Advances in Natural Language Processing and Applications

Research in Computing Science

Research in Computing Science es una publicación trimestral, de circulación internacional, editada por el Centro de Investigación en Computación del IPN, para dar a conocer los avances de investigación científica y desarrollo tecnológico de la comunidad científica internacional. **Volumen 33** Febrero, 2008. Tiraje: 500 ejemplares. *Certificado de Reserva de Derechos al Uso Exclusivo del Título* No. 04-2004-062613250000-102, expedido por el Instituto Nacional de Derecho de Autor. *Certificado de Licitud de Título* No. 12897, *Certificado de licitud de Contenido* No. 10470, expedidos por la Comisión Calificadora de Publicaciones y Revistas Ilustradas. El contenido de los artículos es responsabilidad exclusiva de sus respectivos autores. Queda prohibida la reproducción total o parcial, por cualquier medio, sin el permiso expreso del editor, excepto para uso personal o de estudio haciendo cita explícita en la primera página de cada documento. Imagen de la portada derivada de: http://www.typophile.com/files/hebrew-script_5538.jpg.
Impreso en la Ciudad de México, en los Talleres Gráficos del IPN – Dirección de Publicaciones, Tres Guerras 27, Centro Histórico, México, D.F. Distribuida por el Centro de Investigación en Computación, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, México, D.F. Tel. 57 29 60 00, ext. 56571.

Editor Responsable: *Juan Humberto Sossa Azuela*, *RFC SOAJ560723*

***Research in Computing Science** is published by the Center for Computing Research of IPN.* ***Volume 33**, February, 2008. Printing 500. The authors are responsible for the contents of their articles. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior permission of Centre for Computing Research. Printed in Mexico City, February, 2008, in the IPN Graphic Workshop – Publication Office.*

Volume 33
Volumen 33

Advances in Natural Language Processing and Applications

Volume Editor:
Editor del Volumen

Alexander Gelbukh

Instituto Politécnico Nacional
Centro de Investigación en Computación
México 2007

ISSN: 1870-4069

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX
Indexada en LATINDEX

Printing: 500
Tiraje: 500

Printed in México
Impreso en México

Preface

Natural Language Processing is a branch of Artificial Intelligence aimed at enabling the computers to perform meaningful tasks related to written or spoken human language, such as English or Spanish, as humans do. For traditional computer programs, a text is just a long string of characters that the program can, for instance, copy, print, or erase. In contrast, intelligent natural language processing programs are designed for operations involving the meaning of the text.

The most important applications of natural language processing include information retrieval and information organization, machine translation, and natural language interfaces for human-computer interaction, among others.

On the other hand, as in any science, the activities of many researchers are concentrated on its internal art and craft, that is, on the solution of the problems arising in the process of analysis or generation of natural language text or speech (as contrast to the applications intended for the end user). Such problems include, for instance, syntactic and semantic analysis of text and its disambiguation.

This volume presents 19 original research papers written by 45 authors representing 17 different countries: Canada, Czech Republic, Finland, France, Germany, Hong Kong, India, Iran, Japan, Jordan, Korea (South), Mexico, Romania, Russia, Spain, UK, and USA. The papers included in this volume were selected on the base of rigorous international reviewing process out of 102 papers considered.

The volume is structured into 7 thematic areas of both internal art-and-craft and applications of Natural Language Processing:

Natural Language Processing

– Semantics
– Parsing and Part of Speech tagging
– Morphology and Word Segmentation
– Word Sense Disambiguation

Applications

– Machine Translation
– Information Retrieval and Question Answering
– Human-Computer Interfaces

I would like to cordially thank all people involved in the preparation of this volume. I am grateful to Shuly Wintner and Sulema Torres for their invaluable contribution to preparation of this volume. I gratefully acknowledge valuable collaboration of the Faculty of Social Sciences, the Rector’s office, and the President’s office of the University of Haifa, as well as the Caesarea Edmond Benjamin de Rothschild Foundation Institute for Interdisciplinary Applications of Computer Science (CRI) at the University of Haifa, especially Profesor Martin Charles Golumbic, Rona Perkis, Avital Berkovich, George Karpatyan, and Orly Ross. I also acknowledge the generous financial support of the Center for Complexity Science in Israel. The submission, reviewing, and selection process was supported for free by the EasyChair system.

Alexander Gelbukh February 2008

Table of Contents
Índice

Semantics

A Language Independent Approach for Recognizing Textual Entailment

Adrian Iftene and Alexandra Balahur-Dobrescu

"Al.I.Cuza" University of Iasi, Romania
adiftene@info.uaic.ro, alexyys13@yahoo.com

Abstract. Textual Entailment Recognition (RTE) was proposed as a generic task, aimed at building modules capable of capturing the semantic variability of texts and performing natural language inferences. These modules can be then included in any NLP system, improving its performance in fine-grained semantic differentiation. The first part of the article describes our approach aimed at building a generic, language-independent TE system that would eventually be used as a module within a QA system. We evaluated the accuracy of this system by building two instances of it - for English and Romanian and testing them on the data from the RTE3 competition. In the second part we show how we applied the steps described in [1] and adapted this system in order to include it as module in a QA system architecture. Lastly, we show the results obtained, which point out significant growth in precision.

1 Introduction

Recognizing textual entailment RTE[1] [3] is the task of deciding, given two text fragments, whether the meaning of one text is entailed (can be inferred) from the other text. The aim in defining this task was to create an application-independent framework for assessing means of capturing major semantic inferences needed in many NLP applications. Examples of such applications are: Information Retrieval (IR), Question Answering (QA), Information Extraction (IE), and Text Summarization (SUM).

Formally, textual entailment is defined in [1] as a directional relation between two text fragments, termed *T - the entailing text*, and *H - the entailed text*. It is then said that *T* entails *H* if, typically, a human reading *T* would infer that *H* is most likely true. This definition is based on (and assumes) common human understanding of language as well as common Background Knowledge.

TE systems compete each year in the RTE competition, organized by PASCAL[2] (Pattern Analysis, Statistical Modeling and Computational Learning) - the European Commission's IST-funded Network of Excellence for Multimodal Interfaces.

Question Answering (QA) Systems are one of the main research topics in the Natural Language Processing field. These systems not only employ various discourse

[1] http://www.pascal-network.org/Challenges/RTE/
[2] http://www.pascal-network.org/

Received 17/10/07
Accepted 07/12/07
Final Version 16/01/08

analysis and processing tools, but also require theoretical studies and formalizations of many language issues, like questions structure and implied knowledge. QA systems receive natural language queries and not keywords and return precise answers and not documents as output. Finding an answer to a question relies heavily on two things: identification of the required information, and the quantity and quality of information available, therefore on the corpus from which the information can potentially be found.

One of the competitions for QA systems is organized within CLEF (Cross-Language Evaluation Forum). CLEF supports the development of applications for digital libraries by creating an infrastructure for the testing, tuning and evaluation of Information Retrieval systems operating in European languages, both in monolingual and cross-lingual contexts. Within the QA@CLEF evaluation track, we have been participating since 2006 with a Romanian-English multilingual system. Having been confronted with the problem of semantic variability, this year we decided to include an English TE module in the QA system.

The results obtained by using the English TE system within the English QA system have proven to be encouraging, by producing significant growth in precision. Therefore, we decided to build a TE system for Romanian which to use within the Romanian QA system. In order to do that, we tried to replace each of the components of the English system with ones that work on Romanian. The original idea consisted in adapting all components previously used for the English TE system for Romanian. However, this approach generated many problems, since Romanian is not a widely used language and thus the linguistic resources available for it are rather sparse. After repeated trials aimed at adapting modules and resources, we decided to build a generic system using the general tools (such as GATE which is available in 9 languages) and resources that exist for many languages (as WordNet and Wikipedia). Out aim is to build a competitive baseline system, which can be improved for any language with additional specific rules. What follows is a description of the components of the generic system. The evaluation is performed on two languages: English and Romanian. Naturally, we expect a difference in the accuracy score of the system between the two languages, since the English WordNet has 117.659 synsets,[3] and Romanian only approximately 45.000 synsets. Also, the English and Romanian Wikipedia are incomparable in information volume[4] (the English Wikipedia has over 2 million articles and the Romanian Wikipedia only 94 thousands). Subsequently, for the system test we used all datasets[5] from the RTE3 competition and in order to test the system behavior on Romanian we translated the English pairs to Romanian.

2 The Generic Textual Entailment System

The main idea is to transform the hypothesis making use of extensive semantic knowledge from sources like WordNet, Wikipedia, and a database of acronyms.

[3] http://wordnet.princeton.edu/man/wnstats.7WN
[4] http://meta.wikimedia.org/wiki/List_of_Wikipedias
[5] http://www.pascal-network.org/Challenges/RTE3/Datasets/

Additionally, we built a system to acquire the extra Background Knowledge needed and applied complex grammar rules for rephrasing in English. These grammar rules can be translated into any language, and we will see how this is done for Romanian.

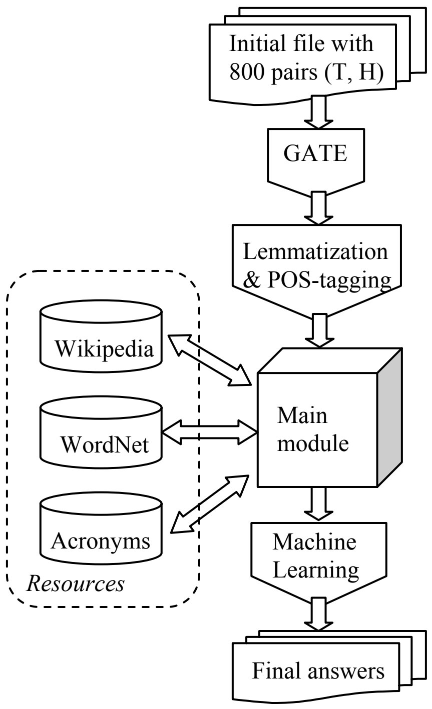


Fig. 1. The process requires an initial pre-processing, followed by the execution of a main module. This uses the output of the first phase and obtains in the end an output that is used as input for two machine learning algorithms. The machine learning algorithms eventually classify all pairs

In the first phase, the initial file is split into 800 pairs of Text and Hypothesis. All these files are then processed with GATE and the contained named entities are identified. The following step consists in applying a module that identifies the lemma and the part of speech for every word. The main module uses all this information and expands the list of words from H and T using Wikipedia, WordNet and the Acronyms resources. Eventually, the main module applies the semantic rules and calculates the fitness for every pair (T, H). The final answer regarding the truth value of the entailment relation between T and H is decided by applying two machine learning algorithms, C4.5 [5] and SMO [6] for Support Vector Machine (SVM). We will further observe the use of each of these components.

3.1 Pre-processing

GATE

GATE (General Architecture for Text Engineering [2]) has the role of identifying named entities in the text and hypothesis. GATE can identify the following types of named entities: person names, countries, locations, organizations, dates, numbers, etc. The rule involving the identification of proper names, numbers and dates was very important to our English system architecture (around 16% of the total system accuracy was gained using this rule). This was an important motivation for its adaptation. The reason for which we used GATE was the number of plugins available for processing in 10 languages: English, French, German, Spanish, Italian, Chinese, Arabic, Romanian, Hindi and Cebuano.

What we used in the generic TE system was a combination between the NEs of the current language of the system used and the English lists of NEs, since the English database is more complex and many NEs are names of persons that have the same form in all languages.

Lemmatization

In order to perform the lemmatization, we apply a Perl module that uses a database containing on separate lines all possible forms for every lemma. This phase also includes the identification of the part-of-speech for every word. As it will further be seen, POS-tagging is important when applying the negation rules, since only verbs can be influenced by negative or positive contexts.

3.2 Main Module

The main module receives separate files for each text-hypothesis pair from the initial data (test or development). For each of the text snippets, the corresponding file contains the lemmas of the words and the marked named entities. We further proceed to identifying and eliminating the stop words using a language dependent list with stop words (this list must be created for every language, and usually contains words with high frequency like articles, prepositions and so on). The remaining resources used are WordNet, the Acronyms Database and the Background Knowledge. They are used to expand each remaining word from the hypothesis to a list of similar or related terms and thus increase the probability to find the initial word or any of its equivalents in the list of words from the text.

WordNet: Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept [4]. Synsets are interlinked by means of conceptual-semantic and lexical relations,[6] such as synonymy, antonymy, hypernymy, etc. WordNet is presently available in 15 languages.

The **acronyms' database** helps our program to find relations between the acronym and its meaning, for example "US - United States". For English, we use a database

[6] http://wordnet.princeton.edu/

with acronyms.[7] For Romanian, we automatically extracted a list of acronyms from a collection of Romanian newspaper articles on topics related to economics and politics. A list of acronyms for Romanian, including the domains of economics, politics etc. can be found also on the web.[8] We think such lists exist for all languages and if not, they can be relatively easily built using a large collection of newspaper articles in the interest language and a set of specific extraction patterns.

The Background Knowledge was used in order to expand the named entities and the numbers. It was built semi-automatically, and it used a module in which language could be set according to the current system working language. Thus, the corresponding Wikipedia[9] could be selected. For every named entity or number in the hypothesis, the module extracted from Wikipedia a file with snippets with information related to them.

It is possible that for languages other than English the file with snippets should be empty. In these cases we used the English Wikipedia, which is larger. Examples of possible relations are presented below:

Table 1. Background Knowledge

Initial NE	Relation	Wikipedia NE
American	[in]	America
America	[is]	United States of America
2	[is]	February
Chinese	[in]	China

Subsequently, we use this file with snippets and some previously set patterns with the aim of identifying relations between NEs. For all languages envisaged, we can apply the same format of patterns used for English and, additionally, we must add rules taking into account the specifics of the language. The patterns were initially built from the rules used to extract definitions in a Question Answering system. In order to reduce the search space and the complexity of rules, the latter have been categorized in six types:

1. **"Is" Definitions** containing the verb "is":
   Example: "*Abbreviation for Hyper Text Mark Up Language, HTML* **is** *also a protocol used by World Wide Web*".
2. **Specific Verbs Definitions** containing verbs, different from "is". The verbs are "denote", "show", "state", "represent", "define", "specify", "consist", "name", and "permit".
   Example: "*Electronic mail* ***represents*** *sending messages through electronic networks*".
3. **Punctuation Definitions** which use punctuation signs like the dash "-" or brackets "()".
   Example: "*Bit – shortcut for binary digit*".

[7] http://www.acronym-guide.com
[8] http://www.abbreviations.com/acronyms/ROMANIAN
[9] http://en.wikipedia.org/wiki/Main_Page

4. **Layout Definitions** that can be deduced by the layout: they can be included in tables when the defined term and the definition are in separate cells or when the defining term is a heading and the definition is the next sentence.
   Example:

Data organizing	The simplest method is the sequential one.

5. **Anaphoric definitions**, when the defining term is expressed in a precedent sentence and it is only referred in the definition, usually pronoun references.
   Example: "*...defining the database concept.* ***It*** *describes methods of modeling real problems in order to define structures which eliminate redundancy in data collecting*".
6. **Other definitions**, which cannot be included in any of the previous categories. In this category are constructions which do not use verbs as the introducing term, but a specific construction, such as "*i.e.*"
   Example: "*equilateral triangle* ***i.e.*** *having all sides equal*".

If such a relation is found, it is saved to an output file. Usually, not all relations are correct, but those that are will help the system at the next run.

Our patterns identify two kinds of relations between words:

- "is", when the module extracts information using the grammar rules presented above;
- "in" when information extracted with rules contains also specific words which characterize the inclusion: ***in, part of, included, region of***, etc.

In this case, the current node does not influence the fitness of the pair for the [is]-type relations, and the fitness receives some penalties for the [in]-type relation.

For Romanian, similar to the English approach, we also built six types of rules in order to identify definition contexts.

Semantic Variability Rules: negations and context terms

For every verb from the text and hypothesis we consider a Boolean value which indicates whether the verb has a negation or not, or, equivalently, if it is related to a verb or adverb **diminishing** its sense or not. For that, we use the POS-tags and a list of words we consider as introducing a negation: "*no*", "*don't*", "*not*", "*never*", "*may*", "*might*", "*cannot*", "*should*", "*could*", etc. For each of these words we successively negate the initial truth-value of the verb, which by default is "*false*". The final value depends on the number of such words.

Since the mapping is done for all verbs in the text and hypothesis, regardless of their original form in the snippet, we also focused on studying the impact of the original form of the verb on its overall meaning within the text. Infinitives can be identified when preceded by the particle "*to*". Observing this behavior, one complex rule for negation was built for the particle "*to*" when it precedes a verb. In this case, the sense of the infinitive is strongly influenced by the active verb, adverb or noun before the particle "*to*", as follows: if it is being preceded by a verb like "*allow*", "*impose*", "*galvanize*" or their synonyms, or adjective like "*necessary*", "*compulsory*", "*free*" or their synonyms or noun like "*attempt*", "*trial*" and their synonyms, the meaning of the verb in infinitive form is stressed upon and becomes **certain**. For all other cases, the particle "*to*" diminishes the certainty of the action expressed in the infinitive-form verb. Based on the synonyms database with the

English thesaurus,[10] we built two separate lists – one of **certainty stressing** (preserving) – "*positive*" and one of **certainty diminishing** – "*negative*" words. Some examples of these words are "*probably*", "*likely*" – from the list of **negative** words and "*certainly*", "*absolutely*" – from the list of **positive** words.

For the Romanian version of the TE system, we identified negation rules and context influencing words and introduced similar rules. For negation, we consider one of the following words (pure form of negation or a modal verb in indicative or conditional form): "*nu*", "*poate*". Subjunctives are identified by the fact that they are preceded by the particle "*să*". In this case, if the subjunctive is preceded by a word like "*permite*", "*impune*", "*indica*", "*propune*" or their synonyms, adjectives like "*necesar*", "*obligatoriu*", "*liber*" or their synonyms, or nouns like "*încercare*", "*posibilitate*", "*opțiune*" and their synonyms, the meaning becomes positive. For the context influencing words, we built, as for the English system, two lists, one containing words like "*sigur*", "*absolut*", "*categoric*", "*cert*", "*precis*", "*inevitabil*", "*infailibil*" for **context stressing words** and "*probabil*", "*posibil*", "*fezabil*", "*realizabil*", "*practicabil*" – for **context diminishing words**.

Rule for Named Entities from hypothesis without correspondence in text
Additionally, we have a separate rule for named entities from the hypothesis without correspondence in the text. If the word is marked as named entity by GATE, we try to use the acronyms' database or obtain information related to it from the background knowledge. In the event that even after these operations we cannot map the word from the hypothesis to one word from the text, we increase a value that counts the problems regarding the named entities in the current pair. We then proceed to calculating a fitness score measuring the syntactic similarity between the hypothesis and the text, further used as one of the features that the two classifiers used are trained on.

3.2 Fitness Calculation

The main idea is to see in which position we find the expanded word from the hypothesis in the text. Below there is an illustration of the manner in which the fitness is calculated for pair 2 of the RTE3 test set:

```
<pair id="2" entailment="NO" task="IE" length="short" >
        <T>The sale was made to pay Yukos' US$ 27.5 billion tax bill,
        Yuganskneftegaz was originally sold for US$9.4 billion to a little known
        company Baikalfinansgroup which was later bought by the Russian
        state-owned oil company Rosneft .</T>
        <H> Yuganskneftegaz costed US$ 27.5 billion.</H>
</pair>
```

In this case, the hypothesis is transformed as follows:

1. After eliminating the stop words, we obtain the following list of keywords, which contains the lemmas of the words in the hypothesis:

[10] http://thesaurus.reference.com/

{*Yugansksneftegaz, cost, US$, 27.5, billion*}

2. This list is then expanded using the English WordNet, resulting in the following list:
   {*Yugansksneftegaz,* {*cost, price, toll*}*, US$, 27.5,* {*billion, one thousand million, one million million, trillion, jillion, zillion*}}
3. In what follows, the expanded list is enlarged by using the Background Knowledge (BK). In the BK, we find *Yugansksneftegaz [in] Russia* and we replace *Yugansksneftegaz* with the list {*Yugansksneftegaz, Russia*}.
4. Lastly, using the acronyms collection, we further expand the list of terms for US with *United States.*

Eventually, the complete list is: {{*Yugansksneftegaz, Rusia*}, {*cost, price, toll*}, {*US$, United States Dollar*}, *27.5*, {*billion, one thousand million, one million million, trillion, jillion, zillion*}}.

Using this list, we build a matrix containing the appearances of words from the hypothesis in the text without stop words:

Table 2. Mapping of the Hypothesis to the Text

Word Number	Word Lexical Family	Positions in Text
1	Yugansksneftegaz, Rusia	12
2	cost, price, toll	3, 13, 33
3	US$, United States Dollar	7
4	27.5	8
5	billion, one thousand million, one million million, trillion, jillion, zillion	9,17

The formula for calculating the **fitness** is the following:

$$Fitness = \frac{\sum_i \max \frac{1}{abs(PositionInText_i - PositionInText_{i-1})}}{NumberOfWords - 1} \quad (1)$$

Where *i* represents the "Word Number" from the above table, and takes values between 2 and the maximum value of "Word Number" (in this case 5). For the example considered, using formula (1), the result is:

$$Fitness = \frac{1 + \frac{1}{4} + 1 + 1}{4} = \frac{3.25}{4} = 0.8125$$

In order to classify the pairs, we train two classifiers – C4.5 and SMO, using as characteristics the fitness score, the number of direct matches, number of indirect matches and number of problems regarding the matching between the Named Entities in the hypothesis and the text.

3.2 Results

In order to evaluate the quality of the generic system, we used two sets of data from the development and testing parts of the RTE3 competition. Every set consists of 800 texts - hypothesis pairs. The output of main module serves as input for two machine learning algorithms: SVM and C4.5. The results are close: SVM with precision equal with 0.634 and C4.5 with precision equal with 0.619. For SVM we can see below, the results of the predicted classification into Yes and No entailment against the gold standard.

Table 3. Detailed results using SVM classification

Class	Precision	Recall	F-Measure
YES	0.620	0.739	0.674
NO	0.656	0.523	0.582
YES+NO	**0.634**	**0.631**	**0.628**

In order to observe the system behavior, we trained and classified the pairs using SVM on two languages: English and Romanian. The results shown for the development data are given for 66% of the set used for development and the rest used for testing. For the Romanian system, we translated both the development and test sets of pairs into Romanian. The data in the table below show that the results of the system running on Romanian are lower than the ones obtained for English. The reasons for this difference are the volume and quality of the resources available in both languages: WordNet and Wikipedia.

Table 4. Evaluation results on English and on Romanian using SVM

Language	Development Data	Test Data
English	0.648	0.634
Romanian	0.567	0.561

It is important to point out the fact that when compared to the results of the systems which participated in the RTE3 competition this year, our system is among the 10 best from 26 groups, over the average of all competitors results.

4 Using the TE System in the QA Competition

Information within a corpus can be expressed in a large variety of ways. QA systems must resolve this semantic variability problem, as they must identify texts from which the expected answer can be inferred. A good solution to this problem could be using a TE system and pursuing the steps described as follows:
Being given the question [1]:

Q: *"Who is John Lennon's widow?"*
It can be transformed in a statement with a variable PERSON:
Statement: *PERSON is John Lennon's widow.*

Among the answers containing the key expression John Lennon, the following could be found:

Answer (Text): *"Yoko Ono unveiled a bronze statue of her late husband, John Lennon, to complete the official renaming of England's Liverpool Airport as Liverpool John Lennon Airport."* From this text, a candidate for the variable PERSON can be extracted – *Yoko Ono*. The hypothesis can be built by replacing the variable PERSON with the found candidate.

Hypothesis: *"Yoko Ono is John Lennon's widow".*

The proving of whether the candidate term is correct and the answer to the question is right is done by evaluating the entailment relation between the Text and the Hypothesis.

4.1 Why Use a TE Module in a Question Answering System?

The aim in using the TE system as a module in the general architecture of a QA system is improving the ranking between possible answers for questions in which the answer type is PERSON, LOCATION, DATE and ORGANIZATION.

The idea is to select all relevant named entities from the extracted snippets for one question and replace with them the variables from the patterns associated to the question. In this manner, we will obtain more hypotheses for one text (represented by the snippet). For every hypothesis, we calculate the fitness score and eventually select the named entity for which the highest value is obtained. Consequently, we compare the highest value obtained for each snippet and select the global best value.

In order to see the behavior of our generic Textual Entailment system, we performed some tests using the Romanian data from the CLEF competition. For question 1: "*Ce faimos romancier, nuvelist şi realizator american de povestiri a trăit între anii 1899 şi 1961?*" (What famous American novelist, and short story writer lived between 1899 and 1961?), the module that was responsible with information retrieval and extraction, returned two snippets:

S1: "*Petru Popescu este un romancier, scenarist şi realizator de filme american de origine română. A emigrat în Statele Unite ale Americii în anii 1980, unde s-a impus drept romancier şi autor de scenarii ale unor filme de la Hollywood.*" ("*Petru Popescu is an American novelist, script writer and movie maker of Romanian origin. He emigrated to the United States around the 80s, where he imposed as novelist and Hollywood movies script writer.*")

S2: "*Americanul Ernest Hemingway (1899-1961), autor de povestiri, nuvelist şi romancier, şi romancierul rus Yuri Olesha (1899-1976) s-au născut la aceeaşi dată.*" ("*The American Ernest Hemingway (1899-1961), tales and short stories writer, and novelist and the Russian novelist Yuri Olesha(1899-1976), were born on the same date.*")

For the first snippet, S1, we have only one possible answer, which is *Petru Popescu.* Our hypothesis will be: *Petru Popescu, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.* Since the hypothesis contains the numbers 1899 and 1961 which don't appear in snippet S1, we use the named entity rule, obtaining a fitness score of 0.52 and a number of named entities with problems of 2. Together with the discussed features, the test pair is introduced in the classifiers as test case. After running the system, the pair is classified with NO entailment.

In the second snippet we have two named entities of type PERSON: *Ernest Hemingway* and *Yuri Olesha*. We obtain two hypotheses:

H2_1: *Ernest Hemingway, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.*

H2_2: *Yuri Olesha, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.*

The fitness for pair (*H2_1, S2*) is 0.75, and for pair (*H2_2, S2*) is 0.647. Together with the discussed features, the two test pairs are introduced in the classifiers as test cases, and both pairs are classified with YES. Since the fitness for the first NE is greater than the fitness of the second NE, we conclude that it is possible for both NEs to represent the correct answer, but the probability that *Ernest Hemingway* should be the correct answer is greater than the probability that *Yuri Olesha* should be the correct answer.

For the types specified, we built specific patterns according to the answer type:

Table 5. Patterns built for Location, Date and Organization answers types

LOCATION	Where was she born?	She was born **in** LOCATION.
DATE	When was the reorganized edition of the poem published?	The reorganized edition of the poem was published **at** DATE.
ORGANIZATION	What computer software company headquartered in San Jose was founded in 1982?	ORGANIZATION, **a** computer software company headquartered in San Jose was founded in 1982.

4.2 Results

Adding the TE module in the QA system improves the capability of the QA system to choose with a higher probability the right answer in the case of complex statements, which express the same idea, but with different actors and contexts. However, in the case of fragments which do not represent coherent statements, the TE module within the QA system is useless.

Including a TE module in a QA system results in clear improvements in the quality of the latter. The tests performed on Romanian, in the case of questions with answers of type PERSON and LOCATION, show an increase in accuracy of up to 5%.

5 Conclusions

Our work offers a solution for implementing a generic TE system. The quality of the implemented system depends on the quality of resources such as WordNet and

Wikipedia for a specific language. Additionally, we use lists of stop words, acronyms, words that create positive contexts and negative contexts. Our aim in this approach was to offer a baseline for any language for which a TE system can be implemented.

We also showed how this generic system was used as module within a Romanian QA system, resulting in improved ranking between the possible answers for questions of the type PERSON and LOCATION.

In the future, we plan to further develop the system in order to also be able to process questions with answers of type DATE and ORGANIZATION. Furthermore, we will try to test the performance of the TE system for Spanish.

References

1. Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini B., Szpektor, I.: The Second PASCAL Recognising Textual Entailment Challenge. In Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment. Venice. Italy. (2006)
2. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July (2002)
3. Dagan I., Glickman O., Magnini, B.: The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela et al., editors, MLCW 2005, LNAI Volume 3944. Springer-Verlag (2006) 177-190
4. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge, Mass. (1998)
5. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning, B. Schoelkopf, C. Burges, and A. Smola, eds., MIT Press (1998)
6. Quinlan, J. R.: C4.5: Programs for machine learning. San Francisco: Morgan Kaufmann (1993)

Summarization by Logic Segmentation and Text Entailment

Doina Tatar, Emma Tamaianu-Morita, Andreea Mihis and Dana Lupsa

University "Babes-Bolyai"
Cluj-Napoca
Romania
{dtatar,mihis,dana}@cs.ubbcluj.ro,etamaian@yahoo.com

Abstract. As the phenomenon of information overload grows day by day, systems that can automatically summarize documents become increasingly studied and used. This paper presents some original methods for text summarization of a single source document by extraction. The methods are based on some of our own text segmentation algorithms. We denote them logical segmentations because for all these methods the score of a sentence is the number of sentences of the text which are entailed by it. The first method of segmentation and summarization is called Logical TextTiling (LTT): for a sequence of sentences, the scores form a structure which indicates how the most important sentences alternate with ones less important and organizes the text according to its logical content. The second method, Pure Entailment uses definition of the relation of entailment between two texts. The third original method applies Dynamic Programming and centering theory to the sentences logically scored as above. The obtained ranked logical segments are used in the summarization. Our methods of segmentation and summarization are applied and evaluated against a manually realized segmentation and summarization of the same text by Donald Richie, "The Koan", [9]. The text is reproduced at [14].

1 Introduction

Systems that can automatically summarize documents become increasingly studied and used. As a summary is a shorter text (usually no longer than a half of the source text) that is produced from one or more original texts keeping the main ideas, the most important task of summarization is to identify the most informative (salient) parts of a text comparatively with the rest. Usually the salient parts are determined on the following assumptions [6]:

- they contain words that are used frequently;
- they contain words that are used in the title and headings;
- they are located at the beginning or end of sections;
- they use key phrases which emphasize the importance in text;
- they are the most highly connected with the other parts of text.

Received 24/10/07
Accepted 07/12/07
Final Version 12/01/08

If the first four characteristics are easy to achieve and verify, the last one is more difficult to establish. For example, the connectedness may be measured by the number of shared words, synonyms, anaphora [7],[8]. On the other hand, if the last assumption is fulfilled, the cohesion of the resulting summary is expected to be higher than if this one is missing. In this respect, our methods assure a high cohesion for the obtained summary while the connectedness is measured by the number of logic entailments between sentences.

In this paper we present some methods for summarization of a single source document by extraction based on a previous segmentation. The methods are genre independent and application independent, and take advantage of discourse structure.

The paper is structured as follows: in Section 2, some notions about textual entailment are discussed. Some basics about segmentation of discourse and Logical TextTilling algorithm are given in Section 3, where some different variants of our methods are presented. Summarization by segmentation is the topic of Section 4. An original dynamic programming algorithm for segmentation and summarization is presented in Section 5. We have applied the above methods of segmentation and summarization to the text [9]. Section 6 summarizes the statistics of the results. We finish the article with conclusions and possible further work directions.

2 Text entailment

In [10] are presented three methods for recognizing the text entailment, based on the following directional principle: A text T entails an hypothesis H, denoted by $T \rightarrow H$, iff H is less informative than T. The first method in [10] is obtained from the text-to-text metric introduced in [1] and from the lexical resolution introduced in [11] and consists in the verification of the relation: $sim(T,H)_T \leq sim(T,H)_H$. Here $sim(T,H)_T$ and $sim(T,H)_H$ are text-to-text similarities introduced in [1]. The second method in [10] calculates the similarity between T and H by *cosine*, the third by a variant of Levenshtein distance. All these methods confirm the above principle, the highest computational precision being realized by the second method. We use in this paper the second method.

A very simple algorithm of segmentation (denoted Pure Entailment) based on the entailment relation is the following:

```
Input: Text= $\{S_1, S_2, \cdots S_n\}$
Output: Segments $Seg_1, ..., Seg_t$.
     $k = 2, t = 1, Seg_t = \{S_1\}$
     while $k < n$ do
        if $(Seg_t \rightarrow S_k)$
          then
             $Seg_t := Seg_t \cup \{S_k\}$
          else
             $t := t + 1, Seg_t := \{S_k\}$
        endif
        k:=k+1
        endwhile
```

A method of summarization based on the entailment relation is:

```
    **Input**: Text= $\{S_1, S_2, \cdots S_n\}$
    **Output**: Summary $S$.
        $S = \{S_1\}; i = 2$
        while $i < n$ do
            if not $(S \rightarrow S_i)$
              then
                  $S := S \cup \{S_i\}$
            endif
            i:=i+1
        endwhile
```

3 Segmentation by Logical TextTiling

To observe the difference between the algorithm to detect subtopic structure (as TextTiling [3]) and our algorithm oriented to detect logical structure of a text, let us emphasize that for us, the unit of discourse is a sentence, with a level of importance which is different from the level of importance of its neighbors. A logical segment is a contiguous piece of text (a sequence of sentences) that is linked internally but disconnected from the adjacent text. This feature assures the local coherence of a segment.

The main differences between TextTiling (TT) and our Logical TextTiling (LTT) method are:

- The tokenization in TT is made by dividing the text into token-sequences (pseudosentences) of a predefined size. In LTT the individual units are sentences;
- In TT a score is assigned to each token-sequence boundary i, calculating how similar the token-sequences $i - k$ through i are to token-sequence from $i + 1$ to $i - k + 1$.
  In LLT the score of a sentence S_i, $score(S_i)$, is the number of sentences of the text which are entailed by S_i (or as in section 3.1). A higher score of a sentence denotes a higher importance. This fact is in accordance with the following text entailment criteria: A text T entails a text H iff the text H is less informative than T [10]. So, the more sentences are entailed by a sentence S_i, the higher its importance is.

A boundary (a shift in discourse or a gap) in LTT is determined by a change in the configuration of logical importance of sentences (see Fig.1. and Fig.2.). In such a way we have a structural basis for dividing up the discourse into a series of smaller segments, each with a different configuration of the importance of components.

The obtained logical segments could be used effectively in summarization. In this respect, our method of summarization falls in the discourse-based category. In contrast with other theories about discourse segmentation, as Rhetorical

Structure Theory (RST) of Mann and Thompson (1988), attentional / intentional structure of Grosz and Sidner (1986) or parsed RST tree of Daniel Marcu (1996), our Logical TextTiling method (and also TextTiling method [3]) supposes a linear segmentation (versus hierarchic segmentation) which results in an advantage from a computational viewpoint.

The algorithm LTT of segmentation is :

```
INPUT: A sequence of sentences $S_1, ...S_n$ and a list of scores $score(S_1), ...score(S_n)$
OUTPUT: A list of segments $Seg_1, ...Seg_j$
    $j = 1; p = 1; Seg_1 = \{S_1\}$
    dir=up;
    while $p < n$ do
        if $score(S_{p+1}) > score(S_p)$
            then
                if dir = up
                    then
                            $Seg_j = Seg_j \cup \{S_{p+1}\}$
                    else
                            $j = j + 1$
                            $Seg_j = \{S_{p+1}\}$
                            dir=up
                endif
            else
                $Seg_j = Seg_j \cup \{S_{p+1}\}$
                dir = down
        endif
        $p = p + 1$
    endwhile
```

A very useful variant of the algorithm is to begin a new segment only if the score $score(S_p)$ is bigger than a given threshold.

3.1 ArcInt and ArcReal scores

As variants of the above score of a sentence as the number of entailed sentences we introduced in this paper two different types of scores: ArcInt and ArcReal. Because the entailment relationship takes place between two sentences, we will consider that those two sentences are bounded together by an "arc". From a single sentence point of view, it can be the starting point of such an "arc", or it can be the destination point, or an in-going arc can pass over it. If i<k<j and sentence i entailes sentence j, then over sentence k passes an "arc" that starts from i and goes to j. If all these "arcs" that start from a sentence, that stop in that sentence and that pass over it are counted, the ArcInt score is obtained. In a way, it shows how the current sentence is linked to the other sentences. But because an entailment relationship is stronger if the sentences involved are close to one another, the sum of all arcs over their length could give a more

precise measure: ArcReal. The following formulas specify more clearly ArcInt and ArcReal measures for a sentence k:

$$ArcInt_k = \sum_{i=1}^{k} \sum_{j=k,i<>j}^{n} entails_{i,j} + \sum_{i=k}^{n} \sum_{j=1,i<>j}^{k} entails_{i,j}$$

$$ArcReal_k = \sum_{i=1}^{k} \sum_{j=k,i<>j}^{n} \frac{entails_{i,j}}{\mid i-j \mid} + \sum_{i=k}^{n} \sum_{j=1,i<>j}^{k} \frac{entails_{i,j}}{\mid i-j \mid}$$

where $entails_{i,j} = 1$ if $s_i \to s_j$ and $entails_{i,j} = 0, otherwise.$

The application of LTT method to sentences scored by ArcInt and ArcReal we will denote in the paper by ArcInt and ArcReal methods. These three methods: LTT, ArcInt and ArcReal are denoted as Logical methods.

4 Summarization by segmentation

The representation of the text as a structure is motivated by our goal of generating summary extracts, where a salience function must tell us which sentences are more salient, to be introduced in the summary. However, given a set of segments we need a criterion to select the segments and those sentences from a segment which will be introduced in the summary. The summarization algorithm from a sequence of segments is the following:

```
INPUT: The segments $Seg_1, ...Seg_j$, the length of summary X (as parameter);
OUTPUT: A summary $SUM$ of length X .

Calculate the "salience" of each segment and rank the segments in $Seg_{i_1}, ...Seg_{i_j}$;
Calculate the number of sentences in each segment $Seg_{i_s}$, $c_{i_s}$;
Select first $k(< j)$ segments such that $\sum_{Seg_{i_s}} c_{i_s} = X$;
Reorder selected k segments by their occurrence in text: $Seg'_1, ...Seg'_k$ ;
$SUM = \{Seg'_1, ...Seg'_k\}$ ;
```

In our paper we considered equal salience of segments and we selected from each segment as the most salient sentence, the sentence with the maximal score.

5 Dynamic Programming algorithm

In this section we describe our method of summarization by Dynamic Programming. The segmentation of the text is made from the summary, choosing as the first sentence of a new segment a sentence from the summary. In order to meet the coherence of the summary the algorithm selects the chain of sentences with the property that two consecutive sentences have at least one common word. This corresponds to the continuity principle in the centering theory which requires that two consecutive units of discourse have at least one entity in common ([7]). We make the assumption that, with each sentence, is associated a score that reflects how representative is that sentence. In these experiments the scores are the logical scores as used in the previous sections. The score of a selected summary is the sum of individual scores of contained sentences. The summary will be selected such that its score is maximum.

The idea of the Dynamic Programming Algorithm is the following.

Let us consider that δ_i^k is the score of the summary with length k that begins with the sentence S_i. Suppose that we ignore for the moment the requirement that each two consecutive sentences in the summary must have at least one common word. Then the next relation holds:

$$\delta_i^k = score(S_i) + max_{j\ (i<j)}\ (\delta_j^{k-1}\ )$$

If we introduce a penalty for the case in which two consecutive sentences have no words in common, the recursive formula becomes: $\delta_i^k = max_{j\ (i<j)}(score(S_i) + \delta_j^{k-1}\ )$ if S_i and S_j have common words and $\delta_i^k = max_{j\ (i<j)}(penalty*(score(S_i)+\delta_j^{k-1}\ ))$ if S_i and S_j have no common words. When running the implementation of the algorithm we used a penalty with value equal to 1/10.

Dynamic Programming Algorithm

```
INPUT:
-the text to be sumarized, that is a sequence of sentences $S_1, ..., S_n$
-$score(S_i)$ – scores associated to each sentence
-the length X of the summary
OUTPUT:
A summary SUM of length X

for $i = n$ downto 1 do
  $\delta_i^1 = score(S_i)$
  for $k = 2$ to $X$ do
    $\delta_i^k = max_{j\ (i<j)}(score(S_i) + \ (\delta_j^{k-1}\ ))$ if $S_i$ and $S_j$have common words
    $\delta_i^k = max_{j\ (i<j)}(penalty * (score(S_i) + \ \ (\delta_j^{k-1}\ )))$ otherwise
    $h_i^k = argmax_{j\ (i<j)}(score(S_i) + \ (\delta_j^{k-1}\ ))$ if $S_i$ and $S_j$ have common words
    $h_i^k = argmax_{j\ (i<j)}(penalty * (scor(S_i) + \ (\delta_j^{k-1}\ )))$ otherwise
  endfor
endfor
$i = argmax_j(\ \delta_j^X\ )$
$SUM = empty$
$for\ k = X, 1, step = -1\ do$
    $SUM = SUM \cup \{S_i\}$
    $i = h_i^k$
endfor
```

A short comparison with greedy algorithm in [7] is the following: our algorithm is based on previously calculated scores for sentences while greedy algorithm starts with previously calculated scores for sentences, but continuously adjusts them, depending on the sentences chosen in the summary. A second aspect is that our algorithm obtains the optimum solution. Because the scores of the sentences in greedy algorithm depend on the obtained summary, it would be difficult to have in this case a non-backtraking algorithm that obtains the optimum solution.

6 Experiments

It is now generally accepted that for single news article systems produce summaries indistinguishable from those produced by humans [8]. However, we apply our algorithm of segmentation and summarization to the narrative literary text by Donald Richie "The Koan" [9] reproduced at http:// www.cs.ubbcluj.ro/ dtatar/ nlp/ Koan-fara-anaph.txt. The manual anaphora resolution is reproduced at http:// www.cs.ubbcluj.ro/ dtatar/ nlp/ anaphEmma.txt. The structures of logically scored sentences for initial text and for anaphora solved text are presented in Fig.1. The structure of ArcInt and ArcReal scored sentences is presented in Fig.2.

6.1 Evaluation of segmentation

There are several ways to evaluate a segmentation algorithm. These include comparing the segments against that of some human judges, comparing the segments against other automated segmentation strategies and, finally, studying how well the results improve a computational task [7]. We will use all these ways of evaluation, including the study how our segmentation method effect the outcome of summarization (Section 6).

Regarding the comparison with the human judge, the method is evaluated according to [3]:

– how many of the same (or very close) boundaries (gaps) with the human judge the method selects out of the total selected (precision);
– how many true boundaries are found out of the total possible (recall)

We compared with Short (19 segments) and Long (22 segments) manual segmentations (as the gold standards) the results of segmentation obtained by the methods: LLT (with the variants of ArcInt and ArcReal scoring, with and without anaphora solved) , Pure Entailment (with and without anaphora solved), Dynamic Programming (with and without resources, with and without anaphora solved). The resources used in Dynamic Programming method are enumerated in Section 6.3.

The evaluation of the segmentation against the manual summaries ($Manual = Short, Long$) is made with the following measures:

$$Precision_{Method,Manual} = \frac{Number\, of\, correct\, gaps}{Number\, of\, gaps\, of\, Method}$$

$$Recall_{Method,Manual} = \frac{Number\, of\, correct\, gaps}{Number\, of\, gaps\, of\, Manual}$$

where $Number\, of\, correct\, gaps$ is the numbers of begins of segments found by $Method$ which differ with -1,0,+1 to the begins of segments found by $Manual$.

The results are presented in the following table:

Table 1. The results of manual segmentation in Short and Long variants. The results of LTT and PE methods

Manual Short		Manual Long		Method LLT		Method PE	
Segment	Sentences	Segment	Sentences	Segment	Senteces	Segment	Sentences
$Seg.1$	1	$Seg.1$	1	$Seg.1$	$1-2$	$Seg.1$	1
$Seg.2$	$2-6$	$Seg.2$	$2-4$	$Seg.2$	$3-5$	$Seg.2$	$2-6$
$Seg.3$	$7-9$	$Seg.3$	$5-6$	$Seg.3$	$6-7$	$Seg.3$	$7-8$
$Seg.4$	$10-12$	$Seg.4$	$7-9$	$Seg.4$	$8-11$	$Seg.4$	9
$Seg.5$	$13-14$	$Seg.5$	$10-12$	$Seg.5$	$12-14$	$Seg.5$	10
$Seg.6$	$14-15$	$Seg.6$	$13-14$	$Seg.6$	$15-17$	$Seg.6$	11
$Seg.7$	$16-20$	$Seg.7$	$15-18$	$Seg.7$	$18-23$	$Seg.7$	12
$Seg.8$	$21-24$	$Seg.8$	$19-20$	$Seg.8$	$24-28$	$Seg.8$	13
$Seg.9$	$25-29$	$Seg.9$	$21-22$	$Seg.9$	$29-32$	$Seg.9$	$14-16$
$Seg.10$	$30-32$	$Seg.10$	$25-29$	$Seg.10$	$33-35$	$Seg.10$	17
$Seg.11$	33	$Seg.11$	$30-32$	$Seg.11$	$36-39$	$Seg.11$	$18-20$
$Seg.12$	$34-38$	$Seg.12$	33	$Seg.12$	$40-43$	$Seg.12$	21
$Seg.13$	$39-43$	$Seg.13$	$34-38$	$Seg.13$	$44-46$	$Seg.13$	22
$Seg.14$	$44-45$	$Seg.14$	$39-43$	$Seg.14$	$47-49$	$Seg.14$	23
$Seg.15$	$46-50$	$Seg.15$	$44-45$	$Seg.15$	$50-51$	$Seg.15$	24
$Seg.16$	$51-52$	$Seg.16$	$46-50$	$Seg.16$	$52-55$	$Seg.16$	25
$Seg.17$	$53-55$	$Seg.17$	$51-52$	$Seg.17$	$56-58$	$Seg.17$	26
$Seg.18$	56	$Seg.18$	$53-55$	$Seg.18$	$59-60$	$Seg.18$	$27-32$
$Seg.19$	$57-65$	$Seg.19$	56	$Seg.19$	$61-65$	$Seg.19$	33
		$Seg.20$	$57-60$			$Seg.20$	34
		$Seg.21$	$61-63$			$Seg.21$	$35-41$
		$Seg.22$	$64-65$			$Seg.22$	42
						...	...

To save space we omit the description of the segments 23 to 32 of PE methods.

The comparison between the Precision and the Recall of different logical methods presented in this paper reported to Short and Long manual segmentation is given in the Fig.3. and Fig.4..

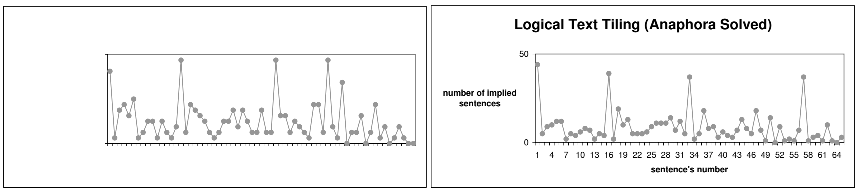


Fig. 1. The logical structure of the text.

The different results of Dynamic programming method are presented in Fig.5. and Fig.6..

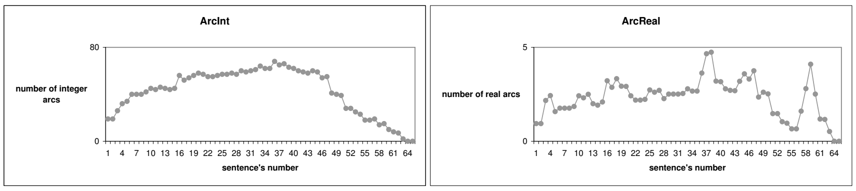


Fig. 2. ArcInt and ArcReal structure of the text.

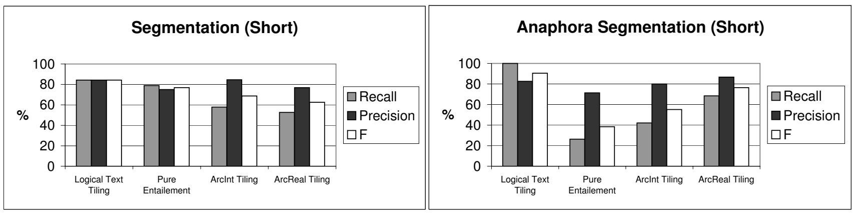


Fig. 3. Comparison of Logical methods with Short segmentation

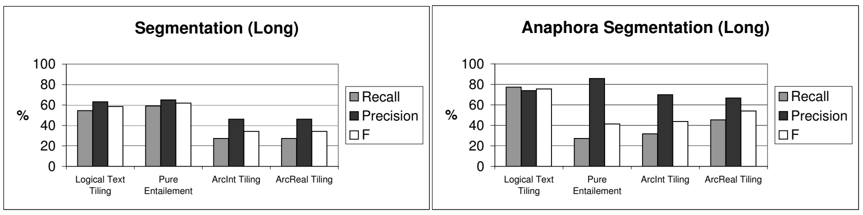


Fig. 4. Comparison of Logical methods with Long segmentation

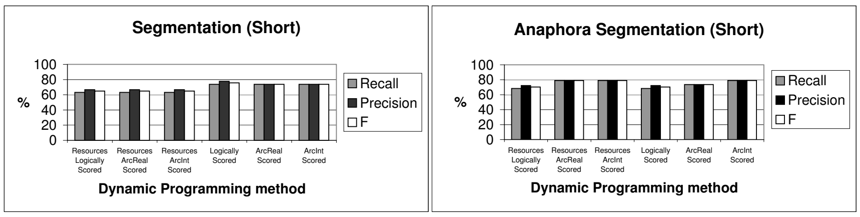


Fig. 5. Comparison of Dynamic Progr. methods with Short segmentation

6.2 Summarization

The gold standards for summaries are obtained manually from manually realized segmentations. The summaries are, in Short and Long variants: Short= $\{1, 2 +$

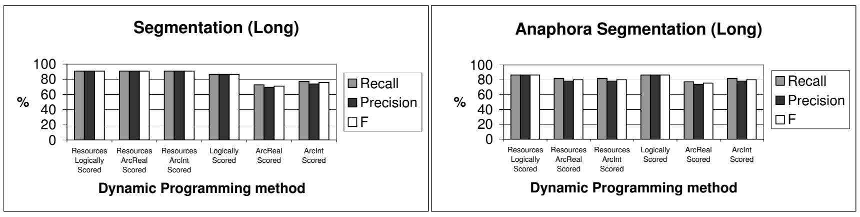


Fig. 6. Comparison of Dynamic Progr. methods with Long segmentation

$4, 7, 10, 14, 16 or 18, 19, 24, 29, 31, 33, 38, 41, 45, 50, 52, 55, 56, 62\, or\, 65\}$ and Long $=$ $\{1, 2 + 4, 6, 7, 10, 14, 16\, or\, 18, 19, 24, 29, 31, 33, 38, 41, 45, 50, 52, 55, 56, 62, 65\}$.

The summary determined by LTT method on initial text is formed by the sentences: $\{1, 4, 6, 9-10, 12, 16, 18, 27, 29, 33, 36, 40, 44-45, 47, 50, 54, 57, 59, 62\}$. The presence of pairs 9-10 and 44-45 in summary is caused of the equal scores for the sentences 9,10 and 44,45 (see Fig.1.). The summary determined by LTT method on the text with Anaphora solved is formed by the sentences: $\{1, 5 - 6, 8, 11, 14, 16, 18, 20, 29, 31, 33, 36, 38, 40, 44, 47, 50, 52, 54, 57, 60, 62, 65\}$.

The precision and the recall of the different methods when the manually summaries are considered are presented in Fig.7-Fig.10.

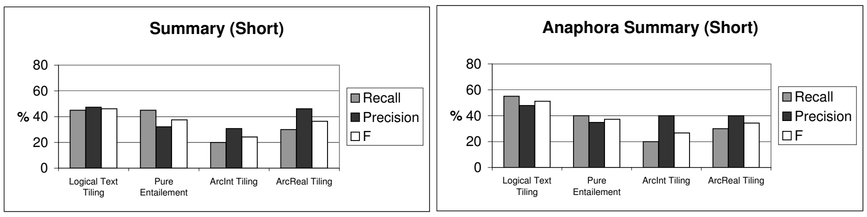


Fig. 7. Precision and Recall for Logical methods (Short summary as standard)

6.3 Implementation details

For Dynamic Programming algorithm we used the LYRL2004 stop-list, which is an ASCII file (english.stop), 3,589 bytes in size. It contains a list of 571 stop words, one per line, and was developed by the SMART project [5],[12] Also, to determine the common words between two sentences we used the files of nouns, adverbs, verbs and adjective of WordNet. To compare words in text with the words in these lists we used Porter stemmer. We used also a part of OpenNLP tools to identify sentences in text, and the tokens (words) at [13]. OpenNLP defines a set of Java interfaces and implements some basic infrastructure for

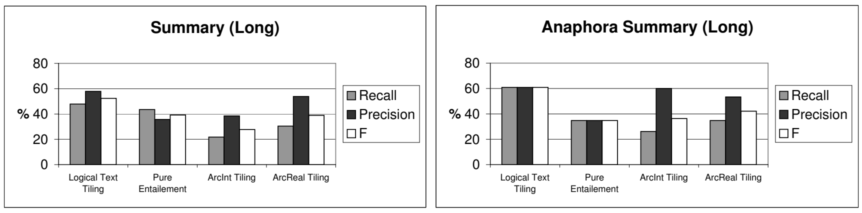


Fig. 8. Precision and Recall for Logical methods (Long summary as standard)

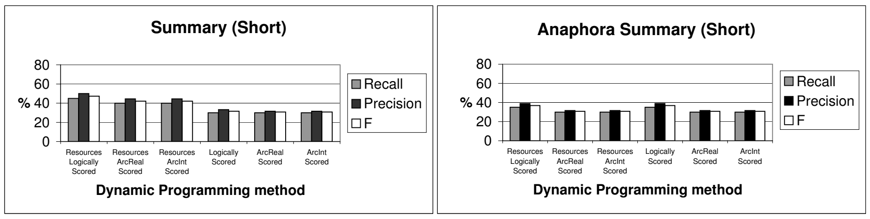


Fig. 9. Precision and Recall for Dynamic Programming method (Short summary as standard)

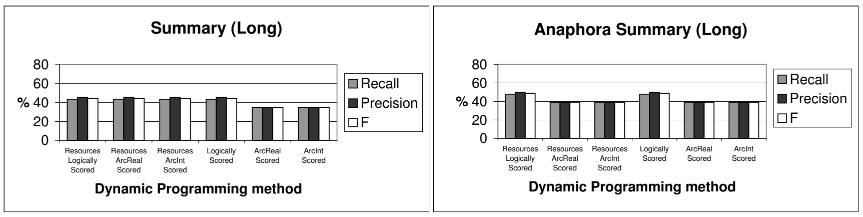


Fig. 10. Precision and Recall for Dynamic Programming method (Long summary as standard)

NLP components. We selected from there sentence detection and tokenization. Also, some of our algorithms have been realized as spreadsheet programs in Microsoft Excel.

7 Conclusion

As the abstraction is harder to be developed in summarization systems, the effort in extraction should be done with a view to increasingly coherent summaries with more accurate ways of identifying the important pieces of information. This paper shows that the discourse segmentation by text entailment relation between sentences is a good basis for obtaining highly accurate summaries (ranging from

30 per cent to over 60 per cent [4]) . Also, scoring the sentences on a logical base can give good results with a neutral method such as Dynamic Programming.

The algorithms described here are fully implemented and use text entailment between sentences without requiring thesaural relations or knowledge bases. The evaluation indices acceptable performance when compared against human judgement of segmentation and summarization. However, our method for computing the logical score of a sentence has the potential to be improved. We intend to improve our text entailment verification tool and to study how the calculation of logical score of a sentence (considering only neighbors of a target sentence, or considering also the number of sentences which imply the target sentence, etc) effects the segmentation and the summarization.

Our method was tested only on narrative texts. We intend to extend the evaluation using other types of texts.

Acknowledgments This work has been supported by CNCSIS Grant 91-037/19.09.2007 CONTO.

References

1. C. Corley, R. Mihalcea: "Measuring the semantic similarity of texts", Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, Ann Arbor, June, 2005, pages 13-18.
2. I. Dagan, O. Glickman and B. Magnini: The PASCAL Recognising Textual Entailment Challenge, Proceedings of the PASCAL Work- shop, 2005.
3. M. Hearst: "TextTiling: A Quantitative Approach to Discourse Segmentation", Technical Report 93/24, University of California, Berkeley, 1993. 1997
4. R. Mitkov (editor): "The Oxford Handbook of Computational Linguistics", Oxford University Press, 2003.
5. Lewis, D. D.; Yang, Y.; Rose, T.; and Li, F. " RCV1: A New Benchmark Collection for Text Categorization Research", Journal of Machine Learning Research, 5:361-397, 2004.
6. D. Marcu: "From discourse structure to text summaries" in Proceedings of the ACL/EACL '97 Workshop on Intelligent Scalable Text Summarization, pp 82-88, Madrid,Spain.
7. C. Orasan: "Comparative evaluation of modular automatic summarization systems using CAST", PhD Thesis, University of Wolverhampton, UK, 2006.
8. D. Radev, E. Hovy, K. McKeown : "Introduction to the Special Issues on Summarization", Computational Linguistics, Vol.28, 2002, pp 399-408.
9. Donald Richie: "The Koan", in Zen Inklings. Some Stories, Fables, Parables and Sermons, New York and Tokyo: Weatherhill, 1991, pp.25-27
10. D. Tatar, G. Serban, A. Mihis and R. Mihalcea: "Text Entailment as directional relation", Proceedings of CALP07 Workshop at RANLP2007, pp 53-58, Borovets, Bulgaria, September 21-23
11. D. Tătar, M. Frenţiu: "Textual inference by theorem proving and linguistic approach", Studia Universitatis "Babeş- Bolyai", Seria Informatics, 2006, nr 2, pages 31-41.
12. http://jmlr.csail.mit.edu/ papers/ volume5/ lewis04a/ a11-smart-stop-list/english.stop
13. http://opennlp.sourceforge.net/
14. http://www.cs.ubbcluj.ro/dtatar/nlp/Koan-fara-anaph.txt

TIL as the Logic of Communication in a Multi-Agent System

Marie Duží

VSB-Technical University Ostrava, 17. listopadu 15, 708 33 Ostrava, Czech Republic
marie.duzi@vsb.cz

Abstract. The method of encoding communication of agents in a multi-agent system (MAS) is described. The autonomous agents communicate with each other by exchanging messages formulated in a near-to-natural language. Transparent Intensional Logic (TIL) is an expressive system primarily designed for the logical analysis of natural language; thus we make use of TIL as a tool for encoding the semantic content of messages. The hyper-intensional features of TIL analysis are described, in particular with respect to agents' attitudes and anaphoric references. We demonstrate the power of TIL to determine the antecedent of an anaphoric pronoun. By an example of a simple dialogue we illustrate the way TIL can function as a dynamic logic of discourse where anaphoric pronouns refer to entities of any type, even constructions, i.e. the structured meanings of other expressions.

1 Introduction

Multi-agent system (MAS) is a system composed of autonomous, intelligent but resource-bounded agents. The agents are active in their perceiving environment and acting in order to achieve their individual as well as collective goals. They communicate with each other by exchanging messages formulated in a standardised natural language. According to the FIPA standards[1], *message* is the basic unit of communication. It can be of an arbitrary form but it is supposed to have a structure containing several attributes. Message semantic *content* is one of these attributes, the other being for instance 'Performatives', like 'Query', 'Inform', 'Request' or 'Reply'. The content can be encoded in any suitable language. The standards like FIPA SL and KIF are mostly based on the First-Order Logic (FOL) paradigm, enriched with higher-order constructs wherever needed.[2] The enrichments extending FOL are well defined syntactically, while their semantics is often rather sketchy, which may lead to communication inconsistencies. Moreover, the bottom-up development from FOL to more complicated cases yields the versions that do not fully meet the needs of the MAS communication. In particular, agents' attitudes and anaphora processing create a problem. In the paper we are going to demonstrate the need for an expressive logical tool of Transparent Intensional Logic (TIL) in order to encode the semantic content of

[1] The Foundation for Intelligent Physical Agents, http://www.fipa.org
[2] For details on FIPA SL, see http://www.fipa.org/specs/fipa00008/; for KIF, Knowledge Interchange Format, see http://www-ksl.stanford.edu/knowledge-sharing/kif/

Received 18/10/07
Accepted 07/12/07
Final Version 12/01/08

messages in a near-to-natural language. Using TIL, the human-computer interface and communication is designed in a smooth way.

Transparent Intensional Logic (TIL)[3] is a system with procedural semantics primarily designed for the logical analysis of natural language. Traditional non-procedural theories of formal semantics are less or more powerful logical languages, from the extensional languages based on FOL approach, through some hybrid systems up to intensional (modal or epistemic) logics. Particular systems are suited well to analysing restricted sublanguages, and they are broadly used and well standardised. However, the logic of attitudes is a stumbling block for all of them. Moreover, if such a great variety of specialised languages were involved in order to design a communication language for a multi-agent system (MAS), the agents would have to keep switching from one logical language to another, which is certainly not a plausible solution.

On the other hand, TIL, due to its strong typing and procedural semantics, operates smoothly with the three levels of granularity: the extensional level of truth-functional connectives, the intensional level of modalities and finally the hyperintensional level of attitudes. The sense of a sentence is an algorithmically structured *construction* of a proposition denoted by the sentence. The denoted proposition is a flat mapping with the domain of possible worlds. Our motive for working 'top-down' has to do with anti-contextualism: any given unambiguous term or expression (even one involving indexicals or anaphoric pronouns) expresses the same construction as its sense (meaning) in whatever sort of context the term or expression is embedded within. And the meaning of an expression determines the respective denoted entity (if any), but not vice versa.

When assigning a construction to an expression as its meaning, we specify *procedural know-how*, which must not be confused with the respective *performancy know-how*.[4] Understanding a sentence *S* involves procedural know-how; one can spell out instructions for evaluating the truth-conditions of *S* in any state-of-affairs *w* at any time *t*. But, of course, one can know how to evaluate *S* without actually being able to do so—that is, without having the performatory skills that enable him to determine the truth-value of *S* in a particular state-of-affairs *W* at time *T*.

The paper is organised as follows. After briefly introducing TIL philosophy and its basic notions in Section 2, the following Section 3 describes the method of analysing sentences with anaphoric references occurring in any context; extensional, intensional, or even hyperintensional context of attitudes. By way of an example we demonstrate in Section 4 how TIL functions as the logic of dynamic discourse. Finally, a few notes on TIL implementation by the TIL-Script language are contained in concluding Section 5.

[3] See, for instance, [5], [10] and [11].

[4] See [7], pp.6-7.

2 Basic Principles of TIL

TIL *constructions* are uniquely assigned to expressions as their algorithmically structured meanings. Intuitively, construction is a procedure (a generalised algorithm), that consists of particular sub-constructions. It is an instruction on how to proceed in order to obtain the output entity given some input entities. Atomic constructions (*Variables* and *Trivializations*) do not contain any other constituent but itself; they supply objects (of any type) on which compound constructions operate. *Variables* $x, y, p, q, \ldots,$ construct objects dependently on a valuation; they *v*-construct. *Trivialisation* of an object X (of any type, even a construction), in symbols $\mathbf{{}^0X}$, constructs simply X without the mediation of any other construction. *Compound* constructions, which consist of other constituents, are *Composition* and *Closure. Composition* $[\boldsymbol{F\, A_1 \ldots A_n}]$ is the instruction to apply a function f (*v*-constructed by F) to an argument A (*v*-constructed by $A_1 \ldots A_n$).[5] Thus it *v*-constructs the value of f at A, if the function f is defined at A, otherwise the Composition is *v*-improper, i.e., it does not *v*-construct anything. *Closure* $[\boldsymbol{\lambda x_1 \ldots x_n\ X}]$ is the instruction to *v*-construct a function by abstracting over variables $x_1,\ldots,x_n$ in the ordinary manner of λ-calculi. Finally, higher-order constructions can be used twice over as constituents of composed constructions. This is achieved by a fifth construction called *Double Execution*, $\mathbf{{}^2X}$**,** that behaves as follows: If X *v*-constructs a construction X', and X' *v*-constructs an entity Y, then ${}^2X$ *v*-constructs Y; otherwise ${}^2X$ is *v*-improper.

TIL constructions, as well as the entities they construct, all receive a type. The formal ontology of TIL is bi-dimensional; one dimension is made up of constructions, the other dimension encompasses non-constructions. On the ground level of the type-hierarchy, there are non-constructional entities unstructured from the algorithmic point of view belonging to a ***type of order 1***. Given a so-called *epistemic (or 'objectual')* ***base*** of ***atomic types*** (ο-truth values, ι-individuals, τ-time moments / real numbers, ω-possible worlds), the induction rule for forming functions is applied: where α, $\beta_1,\ldots,\beta_n$ are types of order 1, the set of partial mappings from $\beta_1 \times\ldots\times \beta_n$ to α, denoted $\boldsymbol{(\alpha\ \beta_1\ldots\beta_n)}$, is a type of order 1 as well.[6] Constructions that construct entities of order 1 are ***constructions of order 1***. They belong to a ***type of order 2***, denoted by $*_1$. This type $*_1$ together with atomic types of order 1 serves as a base for the induction rule: any collection of partial mappings, type $\boldsymbol{(\alpha\ \beta_1\ldots\beta_n)}$, *involving* $*_1$ in their domain or range is a *type of order 2*. Constructions belonging to a type $*_2$ that identify entities of order 1 or 2, and partial mappings involving such constructions, belong to a ***type of order 3***. And so on *ad infinitum.*

An object A of a type α is called an α-object, denoted A/α. That a construction C *v*-constructs an α-object is denoted $C \rightarrow_v \alpha$.

(α-)intensions are members of a type $(\alpha\omega)$, i.e., functions from possible worlds to the arbitrary type α. *(α-)extensions* are members of the type α, where α is not equal to $(\beta\omega)$ for any β, i.e., extensions are not functions from possible worlds. Intensions

[5] We treat functions as mappings, i.e., set-theoretical objects, unlike the *constructions* of functions.

[6] TIL is an open-ended system. The above epistemic base {ο, ι, τ, ω} was chosen, because it is apt for natural-language analysis, but the choice of base depends on the area to be analysed.

are frequently functions of a type $((\alpha\tau)\omega)$, i.e., functions from possible worlds to *chronologies* of the type α (in symbols: $\alpha_{\tau\omega}$), where a chronology is a function of type $(\alpha\tau)$. We use variables $w, w_1, \ldots$ as v-constructing elements of type ω (possible worlds), and $t, t_1, \ldots$ as v-constructing elements of type τ (times). If $C \rightarrow \alpha_{\tau\omega}$ v-constructs an α-intension, the frequently used Composition of the form $[[Cw]t]$, the intensional descent of the α-intension, is abbreviated as C_{wt}.

Some important kinds of intensions are:

Propositions, type $o_{\tau\omega}$. They are denoted by empirical (declarative) sentences.

Properties of members of a type α, or simply *α-properties,* type $(o\alpha)_{\tau\omega}$. General terms (some substantives, intransitive verbs) denote properties, mostly of individuals.

Relations-in-intension, type $(o\beta_1\ldots\beta_m)_{\tau\omega}$. For example transitive empirical verbs, also attitudinal verbs denote these relations.

α-roles, offices, type $\alpha_{\tau\omega}$, where $\alpha \neq (o\beta)$. Frequently $\iota_{\tau\omega}$. Often denoted by concatenation of a superlative and a noun ("*the highest mountain*").

Example: We are going to analyse the sentence "Adam is looking for a parking place". Our method of analysis consists of three steps:

1) *Type-theoretical analysis*, i.e., assigning types to the objects talked about by the analysed sentence. In our case we have:
   a) *Adam*/ι;
   b) *Look_for*/$(o\iota(o\iota)_{\tau\omega})_{\tau\omega}$—the relation-in-intension of an individual to a property of individuals: the seeker wants to find an instance of the property;
   c) *Parking(Place)*/$(o\iota)_{\tau\omega}$—the property of individuals.

2) *Synthesis*, i.e., composing the constructions of the objects *ad* (1) in order to construct the proposition of type $o_{\tau\omega}$ denoted by the whole sentence. The sentence claims that the individual Adam has the 'seeking-property' of looking for a parking place. Thus we have to construct the individual Adam, the 'seeking-property', and then apply the latter to the former. Here is how:
   a) The atomic construction of the individual called Adam is simply ${}^0Adam$;
   b) The 'seeking-property' has to be constructed by Composing the relation-in-intension *Look_for* with a seeker $x \rightarrow \iota$ and the property *Parking*/$(o\iota)_{\tau\omega}$ an instance of which is being sought. But the relation-in-intension cannot be applied directly to its arguments. It has to be extensionalized first: $[[{}^0Look\_for\ w]\ t]$, abbreviated as ${}^0Look\_for_{wt}$. Thus we have $[{}^0Look\_for_{wt}\ x\ {}^0Parking]$ v-constructing a truth value. Abstracting first from x by $\lambda x\ [{}^0Look\_for_{wt}\ x\ {}^0Parking]$ we obtain the class of individuals; abstracting from w and t we obtain the 'seeking-property':
   $$\lambda w\lambda t\ [\lambda x\ [{}^0Look\_for_{wt}\ x\ {}^0Parking]].$$
   c) Now we have to Compose the property constructed *ad* (b) with the individual constructed *ad* (a). The property has to be extensionalised first, i.e., $[\lambda w\lambda t\ [\lambda x\ [{}^0Look\_for_{wt}\ x\ {}^0Parking]]]_{wt}$, and then Composed with the former.[7] Since we are going to construct a proposition, i.e., an intension, we finally have to abstract from w, t:
   $$\lambda w\lambda t\ [[\lambda w\lambda t\ [\lambda x\ [{}^0Look\_for_{wt}\ x\ {}^0Parking]]]_{wt}\ {}^0Adam].$$

[7] For details on predication of properties of individuals, see [3].

This construction is the literal analysis of our sentence. It can still be β-reduced to the equivalent form:

$\lambda w\lambda t\ [{}^0Look\_for_{wt}\ {}^0Adam\ {}^0Parking]$.

3) *Type-Theoretical checking*:

$$\begin{tikzcd} \lambda w\lambda t \arrow[d] & {[{}^0Look\_for_{wt}\quad {}^0Adam\quad {}^0Parking]} \\ o_{\tau\omega} & {\underbrace{(o\iota(o\iota)_{\tau\omega})\quad \iota \quad (o\iota)_{\tau\omega}}_{o}} \arrow[l] \end{tikzcd}$$

The role of Trivialisation and empirical parameters $w \rightarrow \omega$, $t \rightarrow \tau$ in the communication between agents can be elucidated as follows. Each agent has to be equipped with a basic ontology, namely the set of primitive concepts she is informed about. Thus the upper index '^{0}' serves as a marker of the primitive concept that the agents should have in their ontology. If they do not, they have to learn them by asking the others. The lower index '$_{wt}$' can be understood as an instruction to execute an *empirical inquiry (search)* in order to obtain the actual current value of an intension, for instance by searching agent's database or by asking the other agents, or even by means of agent's sense perception.

3 Anaphora and Meaning

The problem of an anaphoric reference to a previously used expression is a well-known hard nut of semantic analysis, because the antecedent of the anaphoric reference is often not unambiguously determined. Thus it is often said that anaphora constitutes a pragmatic problem rather than a problem of logical semantics. We agree that *logical* analysis cannot disambiguate any sentence, because it presupposes understanding and full linguistic competence. Yet our method of logical analysis can contribute to solving the problem of disambiguation in at least two respects; (a) a type-theoretical analysis often unambiguously determines which of the possible meanings is used, and (b) if there are two or more possible readings of a sentence, the logical analysis should make all of them explicit. This often concerns the distinction between *de dicto* and *de re* readings.

In this section we propose the method of *logically* analysing sentences with anaphoric references. The method consists in substituting an appropriate construction of the object to which the anaphora refers for the anaphoric variable. In other words, we perform a *semantic* pre-processing of the embedded anaphoric clause based on the *meaning* of the respective antecedent. In this sense anaphora *is* a *semantic* problem. Moreover, we are going to show that TIL strong typing often unambiguously determines the respective antecedent.

3.1 Semantic Pre-Processing of Anaphoric References

Our hyperintensional (procedural) semantics makes it possible to apply anti-contextualist and compositional analysis to anaphoric sentences. The meaning of a sentence containing a clause with an anaphoric reference is the procedure which is a two-phase instruction that comes down to this:

(i) *execute the substitution based on the meaning of the antecedent for the anaphoric variable;*

(ii) *execute the result (a propositional construction) again to obtain a proposition.*

To specify phase (i) we make use of the fact that constructions are objects *sui generis* that the other constructions can operate on. The substitution is realised by a function $Sub/(*_n *_n *_n *_n)$ that operates on constructions C_1, C_2 and C_3 yielding as output the construction C_4 that is the result of substituting C_1 for C_2 in C_3. The phase (ii) consists in executing the adjusted meaning, namely the construction pre-processed by phase (i). To this end we use the fifth construction defined above, the *Double Execution.* The method is uniquely applicable to all kinds of sentences, including those that express (*de dicto* / *de re*) attitudes to a hyperintension, attitudes to an intension, and relations (-in-intension) to extensional entities. Now we adduce examples that illustrate the method.

(A) “5 + 7 = 12, and Charles knows it.”

The embedded clause “Charles knows it” does not express Charles’ relation(-in-intension) to a truth-value, but to a *construction*, here the *procedure* of calculating the result of *5 + 7 = 12*. Hence $Know(ing)/(o\iota *_1)_{\tau\omega}$ is a relation-in-intension of an individual to a construction. However, the meaning of the clause is incomplete; it is an *open* construction with the free variable *it*: $\lambda w \lambda t\ [{}^0Know_{wt}\ {}^0Charles\ it]$. The variable $it/*_2 \rightarrow *_1$ is the meaning of the pronoun ‘it’ that in (A) anaphorically refers to the meaning of “5 + 7 = 12”, i.e., the construction $[{}^0{+}\ {}^05\ {}^07]$. The meaning of the whole sentence (A) is, however, complete. It is the *closed* construction

(A’) $\lambda w \lambda t\ [[{}^0{=}\ [{}^0{+}\ {}^05\ {}^07]\ {}^012] \wedge$
$${}^2\mathbf{[}{}^0Sub\ {}^{00}[{}^0{=}\ [{}^0{+}\ {}^05\ {}^07]\ {}^012]\ {}^0it\ {}^{\mathbf{0}}\mathbf{[}\lambda w \lambda t\ [{}^0Know_{wt}\ {}^0Charles\ it]\mathbf{]]}_{wt}]$$

Types: $Charles/\iota$; $Know/(o\iota *_1)_{\tau\omega}$; $Sub/(*_2 *_2 *_2 *_2)$; $it/*_2 \rightarrow *_1$; the other types are obvious.

Since (A’) seems to be rather complicated, we now show that (A’) is an adequate analysis meeting our three requirements of compositionality, anti-contextualism and a purely semantic solution. The argument of the second conjunct of (A’), namely

(S) $\mathbf{[}{}^0Sub\ {}^{00}[{}^0{=}\ [{}^0{+}\ {}^05\ {}^07]\ {}^012]\ {}^0it\ {}^{\mathbf{0}}\mathbf{[}\lambda w \lambda t\ [{}^0Know_{wt}\ {}^0Charles\ it]\mathbf{]]}_{wt}] \rightarrow *_1$

constructs a *construction* of order 1, namely the one obtained by the substitution of the construction ${}^0[{}^0{=}\ [{}^0{+}\ {}^05\ {}^07]\ {}^012]$ for the variable *it* into the construction $\mathbf{[}\lambda w \lambda t\ [{}^0Know_{wt}\ {}^0Charles\ it]\mathbf{]}$. The result is the *construction*

(S’) $\mathbf{[}\lambda w \lambda t\ [{}^0Know_{wt}\ {}^0Charles\ {}^0[{}^0{=}\ [{}^0{+}\ {}^05\ {}^07]\ {}^012]]\mathbf{]}$, which constructs a proposition *P*. But an argument of the truth-value function conjunction ($\wedge$) can be neither a propositional construction, nor a proposition, but must be a truth-value. Since (S) constructs the construction (S'), and (S') constructs the proposition *P*, the execution steps have to be: (a) execute (S) to obtain the propositional construction (S'), (b) execute the result (S') to obtain the proposition *P*; hence we need the Double Execution of (S) to construct the proposition *P*, and then (c) *P* has to undergo intensional descent with respect to the external *w*, *t* in order to *v*-construct a truth-value.

Note that the open construction $\lambda w \lambda t\ [{}^0\mathit{Know}_{wt}\ {}^0\mathit{Charles}\ \mathit{it}]$ is assigned to "Charles knows *it*" invariably of a context. The variable *it* is free here either for a pragmatic valuation or for a substitution by means of the meaning of the antecedent that is referred to in a linguistic context. The object—*what* is known by Charles—can be completed by a situation of utterance or by a linguistic context. If the sentence occurs within another linguistic context, then *Sub* substitutes a different construction for the variable *it*.

The other example concerns Charles' attitude of seeking the occupant of an individual office:

(B) "Charles sought the Mayor of Dunedin but *he* did not find *him*."

Suppose now the *de dicto* reading of (B), i.e., that Charles' search concerned the office of Mayor of Dunedin and not the location of its holder. The function *Sub* creates a new construction from constructions and, so, can easily be iterated. The analysis of (B) is:

(B^{d}) $\lambda w \lambda t\ [[{}^0\mathit{Seek}_{wt}\ {}^0\mathit{Ch}\ \lambda w \lambda t\ [{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]] \wedge {}^{\mathbf{2}}\mathbf{[}{}^0\mathit{Sub}\ {}^{00}\mathit{Ch}\ {}^0\mathit{he}$
$\mathbf{[}{}^0\mathit{Sub}\ {}^0[\lambda w \lambda t\ [{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]]\ {}^0\mathit{him}\ {}^{\mathbf{0}}[\lambda w \lambda t\ \neg[{}^0\mathit{Find}_{wt}\ \mathit{he}\ \mathit{him}]]\mathbf{]]}_{\mathbf{wt}}]$.

Types: $\mathit{Seek}/(o\iota\iota_{\tau\omega})_{\tau\omega}$; $\mathit{Find}/(o\iota\iota_{\tau\omega})_{\tau\omega}$; $\mathit{Ch}(\mathit{arles})/\iota$; *Mayor_of* (something)$/(\iota\iota)_{\tau\omega}$; D(unedin)$/\iota$; $\mathit{he}/*_1 \to \iota$; $\mathit{him}/*_1 \to \iota_{\tau\omega}$.[8]

Again, the meaning of (B) is the closed construction (B^{d}), and the meaning of the embedded clause "*he* did not find *him*" is the open construction[9] $\lambda w \lambda t\ \neg[{}^0\mathit{Find}_{wt}\ \mathit{he}\ \mathit{him}]$ with the two free variables *he* and *him*. Note that since $\mathit{he} \to \iota$ and $\mathit{him} \to \iota_{\tau\omega}$, the arguments of *Sub* function are unambiguously type-determined. The only construction of an individual to be substituted for *he* is here ${}^0\mathit{Ch}$; and the only construction of an individual office to be substituted for *him* is the construction of the Mayor office, namely $[\lambda w \lambda t\ [{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]]$.

Of course, another refinement is thinkable. The variables *he* and *him*, ranging over individuals and individual offices, respectively, reduce the ambiguity of 'find' by determining that here we are dealing with finding the occupant of an individual office. But the pronouns like 'he', 'him', or 'she', 'her' also indicate that the finder as well as the occupant of the sought office are male and female, respectively. Thus, e.g., a refined meaning of "He found her" would be

[8] We often use the infix notation without Trivialisation when using constructions of truth-value functions $\wedge$ (conjunction), $\vee$ (disjunction), $\supset$ (implication), all of the type (ooo), and negation ($\neg$) of the type (oo).

[9] Tenses are disregarded.

$\lambda w\lambda t\ [[{}^0\mathit{Find}_{wt}\ \mathit{he}\ \mathit{her}] \wedge [{}^0\mathit{Male}_{wt}\ \mathit{he}] \wedge [{}^0\mathit{Female}_{wt}\ \mathit{her}_{wt}]].$

Additional types: *Male*, *Female*/$(o\iota)_{\tau\omega}$; *her*/$*_1 \rightarrow \iota_{\tau\omega}$.

Now perhaps a more natural *de re* reading of 'seeking sentences' like

(B^r) "Charles is looking for the Mayor of Dunedin (namely the location of him)"

is understood as uttered in a situation where Charles knows who the Mayor is, and is striving to locate this individual. Unlike the *de dicto* case, the sentence understood *de re* has an *existential presupposition*: in order that (B^r) have *any* truth value, the Mayor has to exist. Thus we must not substitute the construction of an office, but of the individual (if any) that occupies the office. To this end we use $[{}^0\mathit{Tr}\ [{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]]$ that fails to construct anything if $[{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]$ is *v*-improper (the Mayor does not exist), otherwise it *v*-constructs the Trivialisation of the occupant of the office. Using the technique of substitutions we can discover the adequate analysis of (B^r):

$\lambda w\lambda t\ [{}^0\mathit{Look}_{wt}\ {}^0\mathit{Ch}\ {}^2[{}^0\mathit{Sub}\ [{}^0\mathit{Tr}\ [{}^0\mathit{Mayor\_of}_{wt}\ {}^0D]]\ {}^0\mathit{him}\ {}^0[\lambda w\lambda t\ [{}^0\mathit{Loc\_of}_{wt}\ \mathit{him}]]]]$

Additional types: *Look(_for)*/$(o\iota\mu_{\tau\omega})_{\tau\omega}$; *Tr*/$(*_1\iota)$; *him* /$*_1\rightarrow \iota$; *Loc_of*/$(\mu\iota)_{\tau\omega}$.[10]

3.2 Donkey Sentences

The following example is a variant of the well-known problem of Peter Geach's *donkey sentence*s:

(D) "If somebody has got a new car then *he* often washes *it*."

The analysis of the embedded clause "*he* often washes *it*" containing the anaphoric pronouns 'he' and 'it' is again an open construction with two free variables *he*—*who* (washes), *it* —*what* (is washed), *he*, *it* $\rightarrow \iota$; *Wash*/$(o\iota\iota)_{\tau\omega}$:

$\lambda w\lambda t\ [{}^0\mathit{Wash}_{wt}\ \mathit{he}\ \mathit{it}]$.[11]

The problem of donkey sentences consists in discovering their logical form, because it is not clear how to understand them. Geach in [1], p.126, proposes a structure that can be rendered in 1st-order predicate logic as follows (*NC* new car):

$\forall x\forall y\ ((NC(y) \wedge Has(x, y)) \rightarrow Wash(x, y)).$

However, Russell objected to this analysis that the expression 'a new car' is an *indefinite description*, which is not rendered by Geach's analysis. Hence Russell proposed the analysis that corresponds to this formula of 1st-order predicate logic:

$\forall x\ (\exists y\ (NC(y) \wedge Has(x, y)) \rightarrow Wash(x, \mathbf{y})).$

[10] The type μ is the type of a location/position.

[11] If we also want to analyze the frequency of washing, i.e., the meaning of 'often', then we use the function *Freq*(*uently*)/$((o(o\tau))\tau)$. The function *Freq* associates each time T with a set of those time intervals (of type $(o(o\tau))$) that are frequent in T (for instance, once a week). The analysis of "*he* often washes *it*" is then $\lambda w\lambda t\ [{}^0\mathit{Freq}_t\ \lambda t'[{}^0\mathit{Wash}_{wt'}\ \mathit{he}\ \mathit{it}]]$.

But the last occurrence of the variable *y* (marked in bold) is free in this formula—out of the scope of the existential quantifier supposed to bind it.

Neale in [6] proposes a solution that combines both of the above proposals. On the one hand, the existential character of an indefinite description is saved (Russell's demand), and on the other hand, the anaphoric variable is bound by a general quantifier (Geach's solution). Neale introduces so-called *restricted quantifiers*:

[every *x*: man *x* and [a *y*: new-car *y*](*x* owns *y*)]
([whe *z*: car *z* and *x* owns *z*] (*x* often washes *z*)).[12]

The sentence (D) does not entail that if the man owns more than one new car then some of these cars are not washed by him. Hence we can reformulate the sentence into

(D_1) "Anybody who owns some new cars often washes *all of them* [each of the new cars he owns]."

However, the following sentence (D_2) means something else:

(D_2) "Anybody who owns some new cars often washes *some of them* [some of the new cars he owns]."

The TIL analysis of (D_1), which in principle corresponds to Geach's proposal, is

(D_1') $\lambda w \lambda t\, [{}^0\forall \lambda x\, [{}^0\forall \lambda y\, [[[{}^0 NC_{wt}\, y] \wedge [{}^0 Own_{wt}\, x\, y]] \supset$
$\quad {}^2[{}^0 Sub\ {}^0x\ {}^0he\ [{}^0 Sub\ {}^0y\ {}^0it\ {}^0[\lambda w \lambda t\, [{}^0\mathrm{Wash}_{wt}\, he\ it]]]]_{wt}]$.

Types: *Own*/$(o\iota\iota)_{\tau\omega}$; *Wash*/$(o\iota\iota)_{\tau\omega}$; *NC* (being a new car)/$(o\iota)_{\tau\omega}$; *x, y, he, it* $\rightarrow \iota$; $\forall/(o(o\iota))$—the general quantifier: $[{}^0\forall^\alpha\ \lambda x A]$ *v*-constructs True iff $[\lambda x A]$ *v*-constructs the whole type ι, otherwise False.

But then an objection due to Neale can be levelled against these analyses, namely that in the original sentence (D) the anaphoric pronoun 'it' stands *outside* of the scope of the quantifier occurring in the antecedent. To overcome this objection, we use a different type of quantifier. Apart the common quantifiers $\forall,\exists/(o(o\iota))$ that operate on sets of individuals, we use quantifiers of another type, namely *Some* and *All*/$((o(o\iota))(o\iota))$. *Some* is a function that associates the argument—a set S—with the set of all those sets which have a non-empty intersection with S. *All* is a function that associates the argument—a set S—with the set of all those sets which contain S as a subset. For instance the sentence "Some students are happy" is analyzed by

$\lambda w \lambda t\, [[{}^0 Some\ {}^0 Student_{wt}]\ {}^0 Happy_{wt}]$.

The analyses of the embedded clauses of (D_1), (D_2), namely "*he* washes all of *them*", "*he* washes some of *them*" are (the anaphoric pronoun '*them*' refers here to the *set of individuals*; we use the variable *them* $\rightarrow (o\iota)$ as the meaning of 'them')

$\lambda w \lambda t\, [[{}^0 All\ them]\ \lambda it\, [{}^0 Wash_{wt}\ he\ it]]$, $\lambda w \lambda t\, [[{}^0 Some\ them]\ \lambda it\, [{}^0 Wash_{wt}\ he\ it]]$

[12] Neale in [6], p. 236, takes into account that the sentence is true even if a man owns *more than one* new car. To avoid singularity he thus claims that the description used in his analysis does not have to be singular (definite) but plural: his abbreviation 'whe *F*' stands for 'the *F* or the *F*s'.

respectively. Now we need to substitute a construction of the set of new cars owned by the man for the variable *them.* Further, we have to substitute the variable x ('anybody') for the variable *he* ('*who* washes'), and then the pre-processed construction has to be Double Executed. To prevent collision of variables, we rename the internal variables w, t.

(D$_1$'') $$\lambda w\lambda t\ [{}^0\forall\lambda x\ [[[{}^0Man_{wt}\ x] \wedge [{}^0\exists\lambda y\ [[{}^0NC_{wt}\ y] \wedge [{}^0Own_{wt}\ x\ y]]]] \supset {}^2[{}^0Sub\ {}^0[\lambda y\ [[{}^0NC_{wt}\, y] \wedge [{}^0Own_{wt}\, x\, y]]]\ {}^0them\ [{}^0Sub\ {}^0x\ {}^0he\ {}^0[\lambda w'\lambda t'\ [[{}^0All\ them]\ \lambda it\, [{}^0Wash_{w't'}\ he\ it]]]]]_{wt}]].$$

Gloss: "For every man, if the man owns some new cars then all of them [i.e., the new cars owned] are washed by him [the man x]."

This construction can be viewed as the most adequate analysis of (D$_1$), because it meets Russell's requirement of an indefinite description in the antecedent, while the scope of $\exists$ does not exceed the antecedent.

The second possible reading of (D) is now analyzed using *Some* instead of *All*:

(D$_2$'') $$\lambda w\lambda t\ [{}^0\forall\lambda x\ [[[{}^0Man_{wt}\ x] \wedge [{}^0\exists\lambda y\ [[{}^0NC_{wt}\ y] \wedge [{}^0Own_{wt}\ x\ y]]]] \supset {}^2[{}^0Sub\ {}^0[\lambda y\ [[{}^0NC_{wt}\, y] \wedge [{}^0Own_{wt}\, x\, y]]]\ {}^0them\ [{}^0Sub\ {}^0x\ {}^0he\ {}^0[\lambda w'\lambda t'\ [[{}^0Some\ them]\ \lambda it\, [{}^0Wash_{w't'}\ he\ it]]]]]_{wt}]].$$

Gloss: "For every man, if the man owns some new cars then some of them [i.e., the new cars owned] are washed by him [the man x]."

As we pointed out above, it is not clear how to exactly understand the sentence (D), simply because the sentence is ambiguous. We thus offered analyses that disambiguate it. Whether these readings are the only possible ones is not for us to decide. In our opinion the reading (D$_1$) is more plausible, and Neale takes into account only this one. However, our method makes it possible to easily analyse particular variants of donkey sentences like "… none of them …", "… most of them…", and suchlike. It might be objected, however, that in the interest of disambiguation, we actually analysed two variants of the original sentence.

Sandu formulates in [8] two principles that every compositional procedure for analysing natural language sentences should obey:

(a) there is a one-to-one mapping of the surface structure of a sentence of (a fragment of) English into its logical form which preserves the left-to-right ordering of the logical constants

(b) the mapping preserves the nature of the lexical properties of the logical constants, in the sense that an indefinite is translated by an existential quantifier, etc.

One can see that our analyses (D$_1$'') and (D$_2$'') obey these principles with respect to the glossed variants, but not with respect to the original sentence (D). Regardless of the disambiguation concerning some/all new cars being washed, principle (b) is violated because 'a man' is analysed as 'every man'. To put our arguments on a still more solid ground, we now propose the literal analysis of the sentence (D). The analysis of the clause "A man has a new car" is as follows:

(NC) $$\lambda w\lambda t\ [{}^0\exists\lambda xy\ [[{}^0Man_{wt}\, x] \wedge [{}^0NC_{wt}\, y] \wedge [{}^0Own_{wt}\, x\, y]]].$$

Additional type: $\exists/(o(o\iota\iota))$.

The consequent of (D) expresses that *all* the couples <*he*, *it*> are such that *he Washes it.* Using a variable *couples*/$*_1 \rightarrow (o\iota\iota)$, and quantifier $All^c/((o(o\iota\iota))(o\iota\iota))$, we have:

$$\lambda w\lambda t\,[[{}^0All^c\, couples]\ \lambda he\ it\ [{}^0Wash_{wt}\ he\ it]].$$

Now composing (NC) with the latter, we substitute the construction of the set of couples constructed by the Closure of (NC) for the variable *couples*:

(D') $\lambda w\lambda t\,[[{}^0\exists\lambda xy\,[[{}^0Man_{wt}x] \wedge [{}^0NC_{wt}y] \wedge [{}^0Own_{wt}\,x\,y]]] \supset$
${}^2[{}^0Sub\ {}^0[\lambda xy\,[[{}^0Man_{wt}x] \wedge [{}^0NC_{wt}y] \wedge [{}^0Own_{wt}\,x\,y]]]\ {}^0couples$
${}^0[\lambda w\lambda t\,[[{}^0All^c\ couples]\ \lambda he\ it\ [{}^0Wash_{wt}\ he\ it]]]]_{wt}].$

As is seen, (D') is fully compositional. Our constituents operate on constructions of sets of couples of individuals, as well as particular individuals, which is impossible within a first-order theory. In this respect Hintikka is right when claiming that the compositional treatment does not work;[13] it does not work within a first-order framework. But as soon as we have a powerful higher-order system like TIL at our disposal, there is no need to give up the desirable principle of compositionality.

One pressing question is whether the anaphoric pronouns should be, in general, bound, and if so, another pressing question is whether this is to be in a standard or non-standard way. The Dynamic Predicate Logic (DPL) applies a mechanism of passing on binding.[14] Note that (D') at the same time provides the semantics of this mechanism. Indeed, the variables *he* and *it* are bound in (D'), but the binding is of another kind. They are not directly bound by the existential quantifier. Technically, they are bound by Trivialization; semantically, they are bound by the condition that the pairs of individuals they *v*-construct have to belong to the set mentioned by the antecedent clause.

4 Outline of the Implementation Method

Now we outline the method of computing the complete meaning of anaphoric sentences, i.e., the method of substituting an appropriate antecedent for an anaphoric reference. The method is similar to the one applied by Hans Kamp's Discourse Representation Theory (DRT). 'DRT' is an umbrella term for a collection of logical and computational linguistic methods developed for dynamic interpretation of natural language, where each sentence is interpreted within a certain discourse, which is a sequence of sentences uttered by the same speaker. Interpretation conditions are given *via* instructions for updating the discourse representation. DPL is a logic belonging to this group of theories. Discourse representation theory as presented in [2] addresses in particular the problem of anaphoric links crossing the sentence boundary. It is a first-order theory, and it can be proved that the expressive power of the DRT language with negation is the same as that of first-order predicate logic. Thus actually only expressions denoting individuals (indefinite or definite noun phrases) introduce the so-called discourse referents, i.e., free variables that are updated when interpreting the

[13] See [9]
[14] See [8].

discourse. Anaphoric pronouns are represented by free variables linked to appropriate antecedent discourse variables.

As we have seen above, our semantics is hyperintensional, i.e., procedural, and higher order. Thus not only individuals, but entities of any type, like properties of individuals, propositions, relations-in-intension, and even constructions (i.e. meanings of the antecedent expressions), can be linked to anaphoric variables. Moreover, strong typing makes it possible to *determine the respective type-appropriate antecedent*.

The specification of the implementation algorithm proposed here is imperative;[15] similarly as in DRT, we update the list of potential antecedents, or rather constructions expressed by them, in order to substitute the type-appropriate entities for anaphoric variables, whenever needed. For each type (ι, $(o\iota)_{\tau\omega}$, $o_{\tau\omega}$, $(o\iota(o\iota)_{\tau\omega})_{\tau\omega}$, $(o\iota\iota)_{\tau\omega}$, $*_n$, etc.) the list of discourse variables is created. The method substitutes the content of type-appropriate discourse variables for anaphoric variables to complete the meaning of anaphoric clauses. Each closed constituent of a resulting construction becomes an updated value of the respective (type-appropriate) free discourse-referent variable. In this way the discourse variables are gradually updated.

We now illustrate the method by an example of a simple dialog between three agents, *Adam*, *Berta* and *Cecil*. The list of discourse variables for the dialog together with the types of entities constructed by their respective content is: $ind{:=}\iota$, $loc{:=}\mu$, $pred{:=}(o\iota)_{\tau\omega}$, $prof{:=}(o\iota)_{\tau\omega}$, $rel_1{:=}(o\iota(o\iota)_{\tau\omega})_{\tau\omega}$, $rel_2{:=}(o\iota\mu)_{\tau\omega}$, $rel_3{:=}(o\iota o_{\tau\omega})_{\tau\omega}$, $prop{:=}o_{\tau\omega}$, $constr{:=}*_n$.

Adam to Cecil: "Berta is coming. ***She*** is looking for a parking".
'Inform' message content:
$\lambda w\lambda t\ [[{}^0Coming_{wt}\ {}^0Berta]$;
(Relevant) discourse variables updates:
$ind{:=}{}^0Berta$; $pred{:=}{}^0Coming$;
$prop{:=}\ \lambda w\lambda t\ [[{}^0Coming_{wt}\ {}^0Berta]$;
$\lambda w\lambda t\ {}^2[{}^0Sub\ ind\ {}^0she\ {}^0[{}^0Looking\_for_{wt}\ she\ {}^0Parking]] \Rightarrow$ (is transformed into)
$\lambda w\lambda t\ [{}^0Looking\_for_{wt}\ {}^0Berta\ {}^0Parking]$.
(Relevant) discourse variables updates:
$rel_1{:=}\ {}^0Looking\_for$; $pred{:=}{}^0Parking$;
$prop{:=}\ \lambda w\lambda t\ [{}^0Looking\_for_{wt}\ {}^0Berta\ {}^0Parking]$;
$prof{:=}\ \lambda w\lambda t\ \lambda x\ [{}^0Looking\_for_{wt}\ x\ {}^0Parking]$; ('propositional function')

Cecil to Adam: "***So*** am I."
'Inform' message content:
$\lambda w\lambda t\ {}^2[{}^0Sub\ prof\ {}^0so\ {}^0[so_{wt}\ {}^0Cecil]] \Rightarrow \lambda w\lambda t\ [{}^0Looking\_for_{wt}\ {}^0Cecil\ {}^0Parking]$
(Relevant) discourse variables updates:
$ind{:=}{}^0Cecil$;

Adam to both: "There is a free parking at p_1".
'Inform' message content: $\lambda w\lambda t\ \exists x\ [[[{}^0Free\ {}^0Parking]_{wt}\ x] \wedge [{}^0At_{wt}\ x\ {}^0p_1]]$
(Relevant) discourse variables updates:
$loc{:=}{}^0p_1$; $pred{:=}[{}^0Free\ {}^0Parking]$;
$prop{:=}\ \lambda w\lambda t\ [\exists x\ [[{}^0Free\ {}^0Parking]_{wt}\ x] \wedge [{}^0At_{wt}\ x\ {}^0p_1]]$

[15] The algorithm was first proposed in [4].

Berta to Adam: “What do you mean by free parking?”
‘Query’ message content: $\lambda w \lambda t\ [{}^0Refine_{wt}\ {}^0[{}^0Free\ {}^0Parking]]$
(Relevant) discourse variables updates: $constr := {}^0[{}^0Free\ {}^0Parking]$

Adam to Berta: “Free parking is a parking and some parts of it are not occupied”.
‘Reply’ message content: $[{}^0Free\ {}^0Parking] =$

$$[\lambda w \lambda t\ \lambda x\ [[{}^0Parking_{wt}\, x] \wedge \exists y\ [[{}^0Part\_of_{wt}\ y\ x] \wedge \neg[{}^0Occupied_{wt}\, y]]]]$$

Berta to Adam: “I don’t believe ***it***. I have just been ***there***”.
‘Inform’ message content (first sentence):

$$\lambda w \lambda t\ [{}^2[{}^0Sub\ prop\ {}^0it\ {}^0[\neg[{}^0Believe_{wt}\ {}^0Berta\ it]]] \Rightarrow$$
$$\lambda w \lambda t\ \neg[{}^0Believe_{wt}\ {}^0Berta\ [\lambda w \lambda t\ [\exists x\ [[{}^0Free\ {}^0Parking]_{wt}\, x] \wedge [{}^0At_{wt}\, x\ {}^0p_1]]]],$$

‘Inform’ message content (second sentence):

$$\lambda w \lambda t\ \exists t'[[t' \leq t] \wedge {}^2[{}^0Sub\ loc\ {}^0there\ {}^0[{}^0Been\_at_{wt'}\ {}^0Berta\ there]]] \Rightarrow$$
$$\lambda w \lambda t\ \exists t'[[t' \leq t] \wedge [{}^0Been\_at_{wt'}\ {}^0Berta\ {}^0p_1]].$$

And so on.

Note that due to the procedural semantics, our agents can learn new concepts by asking the other agents. In our example, after receiving Adam’s reply Berta learns the refined meaning of the ‘free parking’ predicate, i.e., she updates her knowledge base by the respective composed construction defining the property of being a parking with some free parts. Moreover, though our approach is as fine-grained as the syntactic approach of languages like KIF, the content of agent’s knowledge is not a piece of syntax, but its meaning. And since the respective construction is what synonymous expressions (even of different languages) have in common, agents behave in the same way independently of the language in which their knowledge and ontology is encoded. For instance, if we switch to Czech, the underlying constructions are *identical*: ${}^0[{}^0Free\ {}^0Parking] = {}^0[{}^0Voln\acute{e}\ {}^0Parkovi\check{s}t\check{e}]$.

Of course, improvements of the above method are straightforward. For instance, in the example we were substituting the last type-appropriate entity that received mention; if we wanted to take into account ambiguities of anaphoric references, we might store into the discourse-representation file more than one variable for each type, together with the other characteristics or prerequisites of entities (e.g., gender, ISA hierarchies between properties), so as to be able to generate more meanings of an ambiguous sentence, and thus to contribute to their disambiguation.

5 Concluding Remarks

The above described method is currently being implemented in the TIL-Script programming language, the computational FIPA compliant variant of TIL. It is a declarative functional language. Its only imperative feature is the *Let* command for the dynamic assignment of a construction *C* to a discourse variable in order to update its content. The command is also used for recursive definitions of functions. TIL-Script comprises all the higher-order features of TIL, as the hyperintensional logic of partial functions with procedural semantics and explicit intensionalisation and temporalisation, making thus a communication of software-agents smooth and very natural. TIL constructions are encoded by natural-language expressions in a nearisomorphic manner and for the needs of real-world human agents TIL-Script messages are presented in a standardised natural language. *Vice versa*, humans can formulate their requests, queries, etc., in the standardised natural language that is transformed into TIL-Script messages. Thus the provision of services to humans can be realised in a form close to human understanding.

Acknowledgements. This research has been supported by the program 'Information Society' of the Czech Academy of Sciences within the project "Logic and Artificial Intelligence for multi-agent systems", No. 1ET101940420, and by the project "Semantization of pragmatics", No. GACR 401/07/0451.

References

1. Geach, P.: *Reference and Generality***.** Ithaca, NY: Cornell University Press (1962)
2. Kamp, H. and Reyle, U.: *From Discourse to Logic. Introduction to Model-Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Springer (1993)
3. Jespersen, B.: 'Predication and extensionalization'. *Journal of Philosophical Logic*, January 2008; retrievable at: http://www.springerlink.com/content/q31731r311428343/?p=60281b936ca249278691b0f25135cc0a&pi=2
4. Křetínský, J.: *Use–mention Distinction in Transparent Intensional Logic*. Bachelor thesis, Masaryk University Brno (2007); retrievable at: http://is.muni.cz/th/139914/fi_b/bachelor.pdf
5. Materna, P.: *Conceptual Systems.* Logos Verlag, Berlin (2004)
6. Neale, S.: *Descriptions*. The MIT Press, Cambridge (1990)
7. Rescher, N.: *Epistemic Logic.* Pittsburgh: University of Pittsburgh Press (2005)
8. Sandu, G.: 'On the theory of anaphora: dynamic predicate logic vs. game-theoretical semantics'. *Linguistic and Philosophy* 20 (1997), 147-174
9. Sandu, G., Hintikka, J.: 'Aspects of compositionality'. *Journal of Logic, Language, Information 10* (2001) 49-61
10. Tichý, P.: *The Foundations of Frege's Logic*, Berlin, New York: De Gruyter (1988)
11. Tichý, P.: *Collected Papers in Logic and Philosophy*, V. Svoboda, B. Jespersen, C. Cheyne (eds.), Prague: Filosofia, Czech Academy of Sciences, and Dunedin: University of Otago Press (2004)

Parsing and Part of Speech Tagging

Collins-LA: Collins' Head-Driven Model with Latent Annotation

Seung-Hoon Na[1] , Meixun Jin[1], In-Su Kang[2] and Jong-Hyeok Lee[1]

[1] Pohang University of Science and Technology (POSTECH), AITrc, Republic of Korea
{nsh1979,meixunj,jhlee}@postech.ac.kr,
[2] Korea Institute of Science and Technology Information (KISTI), Republic of Korea
dbaisk@kisti.re.kr

Abstract. Recent works on parsing have reported that the lexicalization does not have a serious role for parsing accuracy. Latent-annotation methods such as PCFG-LA are one of the most promising un-lexicalized approaches, and reached the-state-of-art performance. However, most works on latent annotation have investigated only PCFG formalism, without considering the Collins' popular head-driven model, though it is a significantly important and interesting issue. To this end, this paper develops Collins-LA, the extension of the Collins' head-driven model to support the latent annotation. We report its basic accuracy, comparing with PCFG-LA. The experimental results show that Collins-LA has potential to improve basic parsing accuracy, resulting in comparable performance with PCFG-LA even in the naive setting.

1 Introduction

Recent works for parsing have consistently shown that the lexicalization does not have serious effects on parsing accuracy. Gildea mentioned that the high performance of a Collins' model is obtained not from the bi-lexical dependency, showing that parsing accuracy is not decreased even when the bi-lexical dependency is not incorporated to the model [1]. Gildea's result has been re-confirmed by Bikel, during his investigation through the re-implementation of the Collins' parsing model [2].

Another direction is opened from a Klein's work, where fully un-lexicalized parsing models are extensively evaluated through an accurate design of tree-bank annotation [3]. Klein's work includes Johnson's parent annotation [4], and external-internal annotation, tag-splitting, and head annotation, etc, resulting in the parsing accuracy of about 86%, which corresponds to the initial performance of the lexicalized parsing accuracy. Motivated from this, Matsuzaki proposed a new generative model PCFG-LA, an extension of PCFG models in which non-terminal symbols are annotated with latent variables [5]. PCFG-LA successfully replaces the manual feature selection used in previous research such as Johnson's work or Klein's work, by showing that PCFG-LA reaches Klein's result. Based on this, Petrov et al. further explored PCFG-LA where hierarchical splitting and merging of latent non-terminals are utilized, thus differentiating the number of latent variables of each non-terminal symbol [6]. Their result is remarkable, showing about 90% accuracy, which is almost comparable to the-state-of-art of Charniak's parser [7]. All these previous results consistently show

Received 15/10/07
Accepted 07/12/07
Final Version 19/01/08

that the annotation-based parsing strategy, especially the recently developed automatic annotation, is a promising approach which can adequately balance between the naiveness of the original parsing model and the sparseness problem of the complicated lexicalized model.

However, most previous works on the annotation are restricted to PCFG, without investigating other probabilistic parsing models such as a head-driven model, etc. Of course, PCFG has many advantages such as simplicity and clearness to be applied to the parsing problem, but there are many other parsing models which are known as the-state-of-art methods. In this regard, this paper investigates the extension of the popular Collins' head-driven model [8] with the latent annotation, namely *Collins-LA*. The Collins' model deserves to be extended with the latent annotation for following reasons. 1) Collins' model is one of the-state-of-art approaches for parsing, as well as the Charniak's model. 2) Gildea's observation that bi-lexicalization does not play a serous role in the high performance of parsing is revisited through Collins' head-driven parsing model. 3) Previous works on PCFG have adopted the Collins' head-driven idea when binarizing production rules collected from the tree-bank, by using head-driven binarization according to the head-modifier generation principle (e.g. Matsuzaki's center-head and center-parent binarization, Petrov's x-bar binarization). Although their usage of the head-driven approach is limited to the style of merely applying the head-modifier rule, not incorporated with the parsing model, the head-driven idea has been widely accepted in the parsing community.

Our goal is to examine whether or not the Collins' model is accepted even for the un-lexicalized model. To this end, we formulate a generation model based on Collin-LA, and derive an EM training algorithm to estimate Collins-LA. To cover the specialized generation for BaseNP, we derive a new EM algorithm using the inside-outside algorithm embedding a forward-backward algorithm, and evaluate it on the standard Penn tree-bank data set. Experimental results show that Collins-LA marginally improves the original Collins' model, and is comparable to Matsuzaki's PCFG-LA. This result is notable in that the Collins' head-driven model works well even within the extension of latent annotation. Further, it implies that the work on the latent annotation should be further separately investigated for other grammars.

Relating to our work, Prescher proposed Head-LA, the latent annotation framework of a head-driven model [9]. However, Prescher's work refers to Charniak's head-driven parsing model [10], not to Collins' parsing model. Collins' head-driven model is different from Chaniak's one, since Chaniak's model involves the production rule within the generative model, while Collins' model does not include the production rule itself. Therefore, Prescher's head-LA cannot be directly applied to Collins' head-driven model, thus we require a new separate algorithm to support the latent annotation for Collins' model.

This paper is organized as follows. First, Section 2 briefly mentions Collins' head-driven model. Section 3 examines the proposed Collins-LA, and a EM algorithm to train it. Section 4 further extends Collins-LA for dealing with a specialized process of BaseNP, and drives a novel EM algorithm where a forward-backward algorithm is embedded to the inside-outside algorithm. Section 5 presents experimental results. Finally, the conclusion will be given in Section 6.

2 Un-lexicalized Collins' Head-Driven Model

In the principle of Collins' model, the head node is first generated for a given parent node, and other modifier nodes are generated from the head node and the parent node. Formally, suppose that the production rule is given by *r* as follows.

$$r : P \rightarrow L_N \Lambda\ L_1 H R_1 \Lambda\ R_M$$

, where *H* indicates the head-node of parent node *P* and, L_i and R_j are left and right child nodes. The generation of Collins' model consists of two separated processes [8]: 1) generation of head-node *H* from the given parent node *P*, 2) generation of left and right child nodes L_i or R_j from parent-node *P* and head-node *H*. This is formulated as follows.

$$P(r) = P(H \mid P)\prod_i P(L_i \mid P,H)\prod_j P(R_j \mid P,H) \qquad \mathbf{(1)}$$

However, in this setting, the generation process will not be stopped, and would prefer the production rule with a less number of child nodes. Collins introduced *STOP* node, indicating the stop event on head-modifier generations, and defined the following model.

$$P(r) = P(H \mid P)\prod_i P(L_i \mid P,H)\prod_j P(R_j \mid P,H) \qquad \mathbf{(2)}$$
$$P_L(STOP \mid P,H)P_R(STOP \mid P,H)$$

where $P_L(STOP|P,H)$ and $P_R(STOP|P,H)$ indicate the probability that generates *STOP* event for the left-child part and the right-child part, respectively. This *STOP*-added model can partially deal with the preference to production rule with the small number of child nodes. In fact, when L_{N+1} and R_{M+1} are inserted to the rule *r* as a STOP symbol, Eq. (2) is equivalent to Eq. (1). Summing up, Collins' model consists of the following four types of probabilities.

1) Head generation: $P(H|P)$
2) Left modifier generation: $P(L_i|P,H)$
3) Right modifier generation: $P(R_j|P,H)$
4) Lexical generation: $P(w|H)$

For a given parse tree *T*, the generation probability of *T* is formulated as follows:

$$P(T) = \prod_{r \in T} P(r)^{c(r;T)} \qquad \mathbf{(3)}$$

where $c(r;T)$ is the number of counts that applies the rule *r* in tree *T*. Parsing is the work to find the best parse tree that maximizes the above generation probability $P(T)$. The above model is the un-lexicalized version of Collins' head-driven. The final version of Collin's model is fully lexicalized where the head-word and the head-tag are annotated to each non-terminal node in the original tree *T*, using the same head-modifier generation principle.

3 Latent Annotation of Collin's Head-Driven Model

3.1 Brief Background on Tree-Bank Annotation Approaches

Annotation-driven approaches do not use a given tree-bank directly. Instead, it converts the tree-bank into the annotated one by annotating non-terminal symbols with another non-terminal symbol in the tree or an automatically generated symbol. In fact, lexicalization used in the head-driven model belongs to annotation-driven approaches, where the head word and the head tag are attached to each non-terminal node.
The initial work for the annotation is Johnson's parent annotation, where a non-terminal node is annotated with the symbol of its parent node [4]. In spite of the simplicity of Johnson's annotation, PCFG using the annotated tree-bank reaches about 79% parsing accuracy which significantly improves the baseline of 72%. Klein's work further explored Johnson's annotation scheme, by investigating various annotation schemes such as tag split, attachment of pre-nonterminal and BaseNP, reaching the parsing accuracy of about 86% [4]. However, Johnson's and Klein's works applied manually designed annotation, i.e. considering "which non-terminal symbol of nodes should be split?", and "how the symbol should be annotated with another symbol?". To avoid the manual design of annotation, Matsuzaki et al. [5] and Petrov et al. [6] applied the latent annotation where such considerations are automatically decided.

3.2 Collins-LA: Latent Extension of Collins Model

This study extends the original Collins' model to Collins-LA by annotating latent variables to non-terminal symbols. Let k be the number of latent variables. For given non-terminal symbol P, there are k possible latent variables for P - $P[1]$, …, $P[L]$. Given parse tree T, $\varphi(T)$ is a *latent-annotated tree* obtained from T where each non-terminal node is mapped to one of latent variables. Then, we define the generation probability of $\varphi(T)$ as follows.

$$P(\phi(T)) = \prod_{r\in\phi(T)} P(r)^{c(r;\phi(T))} \tag{4}$$

The production rule r for generating $\varphi(T)$ takes the following form.

$$r : P[x] \rightarrow L_{N+1}L_N[z_N]\Lambda\ L_1[z_1]H[y]R_1[w_1]\Lambda\ R_M[w_N]R_{M+1} \tag{5}$$

where x, y, z_i and w_j correspond to the index numbers used for latent variables in $\varphi(T)$: $P, H, L_i,$ and R_j, respectively. L_{N+1} and R_{M+1} indicates *STOP* symbols.
From the totality law of the probability, $P(T)$ is obtained from the summation of generative probabilities of all latent-annotated trees as follows.

$$P(T) = \sum_{\varphi(T)\in\Phi(T)} P(\varphi(T)) \tag{6}$$

where $\Phi(T)$ is the set of all possible latent annotated trees which can be obtained from T. $|\Phi(T)|$ is the number of the set, which follows the exponential function of the number of non-terminal nodes in T. Similar to the original Collins' model, we should estimate the following four types of probabilities.

1) Head generation: $P(H[y]|P[x])$
2) Left modifier: $P(L_i[z]|P[x],H[y])$
3) Right modifier: $P(R_j[z]|P[x],H[y])$
4) Lexical generation: $P(w|H[x])$

where x, y, and z are the index numbers of latent variables – P, H and L_i (or R_j)

3.3 EM Training Algorithm

Given the training set of parse trees $Train = \{T_1, T_2, ..., T_U\}$, we derive an EM-training algorithm similar to the inside-outside algorithm to estimate the above four types of probabilities for Collins-LA.

Let $\beta(P[x])$ be the inside-probability which generates the sub-tree with P as a root node, providing that P is assigned to the latent index number x. Then, $\beta(P[x])$ can be defined according to the following three different cases.

1) P is a pre-nonterminal node that contains w_i as a unique child word.

$$\beta(P[x]) = P(w_i \mid P[x]) \tag{7}$$

2) For the stop symbol STOP, $\beta(STOP[z])$ is 1.0.
3) Otherwise, let H be the head child node of P, and L_i and R_j be i-th left modifier and j-th right modifier, respectively. Then, $\beta(P[x])$ is recursively defined as follows.

$$\beta(P[x]) = \sum_y P(H[y] \mid P[x])\beta(H[y]) \prod_i \gamma^i_{Left}(P[x],H[y]) \prod_j \gamma^j_{Right}(P[x],H[y]) \tag{8}$$

where $\gamma_{Left}{}^i(P[x],H[y])$ is the probability that generates the sub-tree with L_i as a root node (including the generation of L_i) as follows.

$$\gamma^i_{Left}(P[x],H[y]) = \sum_z P(L_i[z] \mid P[x],H[y])\beta(L_i[z]) \tag{9}$$

Note that $\gamma_{Left}{}^i(P[x],H[y])$ contains the generation probability for sub-trees with roots as $L_i[1]$ … $L_i[k]$ of each latent node for node L_i, thus it is independent to the index number of the latent variable. Similar to $\gamma_{Left}{}^i(P[x],H[y])$, $\gamma_{Righj}{}^i(P[x],H[y])$ is defined as the probability that generates the sub-tree with R_j as a root node, for given $P[x]$ and $H[y]$ (including the generation of the root node R_j).

$$\gamma^j_{Right}(P[x],H[y]) = \sum_z P(R_j[z] \mid P[x],H_i[y])\beta(R_j[z]) \tag{10}$$

Let $\alpha(P[x])$ be the outside probability that generates all other sub-trees in T, except for the generation of the subtree of $P[x]$ (but including the generation of $P[x]$). For parent node P, let $\alpha(H[y])$, $\alpha(L_i[z])$ and $\alpha(R_j[z])$ be outside probabilities of the head-node H, and child nodes L_i and R_j, respectively. Then, all these outside probabilities are recursively defined using $\alpha(P[x])$, in a style of the top-down form.

$$\alpha(H[y]) = \sum_x \alpha(P[x])P(H[y] \mid P[x]) \prod_i \gamma^i_{Left}(P[x],H[y]) \prod_j \gamma^j_{Right}(P[x],H[y]) \quad \textbf{(11)}$$

For *i*-th left child node L_i,

$$\alpha(L_i[z]) = \sum_x \sum_y \alpha(P[x])P(H[y] \mid P[x])\beta(H[y]) P(L_i[z] \mid P[x],H[y]) \left(\prod_{k \neq i} \gamma^k_{Left}(P[x],H[y]) \right) \prod_j \gamma^j_{Right}(P[x],H[y]) \quad \textbf{(12)}$$

For *j*-th right child node R_j,

$$\alpha(R_j[z]) = \sum_x \sum_y \alpha(P[x])P(H[y] \mid P[x])\beta(H[y]) P(R_j[z] \mid P[x],H[y]) \left(\prod_{k \neq i} \gamma^k_{Left}(P[x],H[y]) \right) \prod_j \gamma^j_{Right}(P[x],H[y]) \quad \textbf{(13)}$$

Based on inside and outside probabilities of $\alpha(P[x])$ and $\beta(P[x])$, the Expectation step in EM algorithm calculates the expected count of events which correspond to four type of rules.

$$c(P[x], w \mid T) \propto \alpha(P[x])P(w \mid P[x]) \quad \textbf{(14)}$$

$$c(P[x],H[y] \mid T) \propto \alpha(P[x])P(H[y] \mid P[x])\beta(H[y]) \prod_i \gamma^i_{Left}(P[x],H[y]) \prod_j \gamma^j_{Right}(P[x],H[y])$$

$$c(P[x],H[y],L_i[z] \mid T) \propto \alpha(P[x])P(H[y] \mid P[x])\beta(H[y]) \prod_{k \neq i} \gamma^k_{Left}(P[x],H[y]) \prod_j \gamma^j_{Right}(P[x],H[y]) P(L_i[z] \mid P[x],H[y])\beta(L_i[z])$$

(Parse tree *T* should be inserted as a conditional term in the right-hand part in Eq. (14), but we skip.) Now in the Maximization step, the four types of probabilities which we pursue are calculated as follows, using the above expected counts.

$$P(w \mid P[x]) \propto \sum_{T \in Train} P(T)c(P[x], w \mid T) \quad \textbf{(15)}$$

$$P(H[y] \mid P[x]) \propto \sum_{T \in Train} P(T)c(P[x],H[y] \mid T)$$

$$P(L_i[z] \mid P[x],H[y]) \propto \sum_{T \in Train} P(T)c(P[x],H[y],L_i[z] \mid T)$$

where *Train* is the training set of parse trees in a tree-bank. *P(T)* can be rewritten by using these inside and outside probabilities.

$$P(T) = \sum_{z} \alpha(X[z] \mid T)\beta(X[z] \mid T) \tag{16}$$

where *X* is a non-terminal symbol in parse tree *T*.

3.4 Decoding

The decoding problem (i.e. parsing problem) is to find the best parse tree that maximizes *P*(*T*). However, the exact decoding problem is NP-complete [5]. Instead, we adopted *n*-best re-ranking strategy, which is one of the approximation methods taken by Matsuzaki et al [5]. In *n*-best re-ranking, we first generate top *n* candidate parse trees using a non-latent original parsing model (PCFG or other probabilistic grammar), and then re-rank them according to the generation probabilities *P*(*T*) from Collins-LA.

4 Further Extension for BaseNP Generation

4.1 Specialized Generation Model for BaseNP

BaseNP is a non-recursive noun phrase which does not contain any other noun phrases as its descendents. Due to a linguistic reason, Collins dealt with BaseNP, using a different generation model from the generation of other non-terminals in his original head-driven model. While other non-terminals are generated from parent node and head child node (i.e. $P(L_i|P,H)$), BaseNP is generated from parent node and previous left modifier ($P(L_i|P,L_{i-1})$). To discuss this specialized generation in more details, let us suppose that BaseNP *P* is generated by production rule *r*:

$$r : P \rightarrow L_N \Lambda\ L_1 H$$

Then, the generation model of BaseNP is given as follows.

$$P(r) = P(H \mid P)\prod_{i} P(L_i \mid P, L_{i-1}) \tag{17}$$

where L_0 indicates *H*.

4.2 New EM: Inside-Outside Algorithm Embedding Forward Backward Algorithm

The specialized dependency for generating BaseNP is characterized to Markov dependency of Hidden Markov Model (HMM). Thus, a forward-backward algorithm for learning HMM should be embedded into the original inside-outside algorithm. For BaseNP, inside-outside probabilities are further reformulated as follows.

Inside Probability. First, the inside probability is reformulated as follows.

$$\beta(P[x]) = \sum_y P(H[y] \mid P[x])\beta(H[y]) f(P[x], L_0[y]) \quad \textbf{(18)}$$

Again, we regard L_0 as H. $f(P[x],L_i[y])$ indicates the forward probability, which is the probability that generates all next left modifiers for given $P[x]$ and $L_i[y]$. This is recursively defined as follows.

$$\begin{aligned} f(P[x], L_i[y]) &= \sum_z P(L_{i+1}[z] \mid P[x], L_i[y])\beta(L_{i+1}[z]) f(P[x], L_{i+1}[z]) \quad \textbf{(19)} \\ &\qquad \textit{if } i < Ml \\ &= \sum_z P(STOP[z] \mid P[x], L_i[y]) \quad \textit{otherwise} \end{aligned}$$

Outside Probability.

Outside probabilities are reformulated case-by-case as follows. For head-child node H,

$$\alpha(H[y]) = \sum_x \alpha(P[x]) P(H[y] \mid P[x]) f(P[x], H[y]) b(P[x], H[y]) \quad \textbf{(20)}$$

For i-th left-child node L_i,

$$\alpha(L_i[y]) = \sum_x \sum_y \alpha(P[x]) P(H[y] \mid P[x]) f(P[x], L_i[y]) b(P[x], L_i[y]) \quad \textbf{(21)}$$

, where $b(P[x],L_i[y])$ indicates the backward probability which is the probability that *i*-th left child node is generated with $L_i[y]$ for given $P[x]$. The backward probability is defined as the summation of probabilities of all possible right stop events.

$$b(P[x], L_i[z]) = \sum_y \beta(L_{i-1}[y]) b(P[x], L_{i-1}[y]) P(L_i[z] \mid P[x], L_{i-1}[y]) \quad \textbf{(22)}$$

As a special case, the backward probability for head child node - $b(P[x],H[y])$ is defined as the summation of probabilities of all possible right stop events.

$$b(P[x], H[y]) = \sum_z P(STOP[z] \mid P[x], H[y]) \quad \textbf{(23)}$$

EM Algorithm of the Specialized Collins-LA for BaseNP.

In the Expectation step, the expected counts are calculated as follows.

$$\begin{aligned} c(P[x], H[y] \mid T) &\propto \alpha(P[x]) P(H[y] \mid P[x]) \beta(H[y]) \quad \textbf{(24)} \\ &\quad f(P[x], H[y]) b(P[x], H[y]) \end{aligned}$$

$$\begin{aligned} c(P[x], L_{i-1}[z], L_i[w] \mid T) &\propto \sum_y \alpha(P[x]) P(H[y] \mid P[x]) \beta(H[y]) \\ &\quad f(P[x], L_i[w]) b(P[x], L_{i-1}[z]) \beta(L_{i-1}[z]) \beta(L_{i-1}[w]) \\ &\quad P(L_i[w] \mid P[x], L_{i-1}[z]) \end{aligned}$$

In the Maximization step, Eq. (15) is used to estimate each type of probability.

5 Experimentation

5.1 Experimental Setting

We used WSJ sections (1 ~ 21) as a training data, and the first 420 sentences in WSJ section 22 as a test set. Through our preliminary experiments, we found that the parsing accuracy between the test set and the standard test set (using WSJ section 23) does not show a difference of more than 0.5%.

5.2 Generating N-best Candidate Parse Trees

Table 1. Oracle Performances of *N*-best Parse Trees

	N = 1	*N = 50*	*N = 100*	*N = 200*	*N = 300*	*N = 500*
PCFG	70.25	85.67	87.24	88.78	89.1	89.66
Klein's Markovization (v=2, h=1)	77.57	89.68	90.93	92.02	92.45	92.82

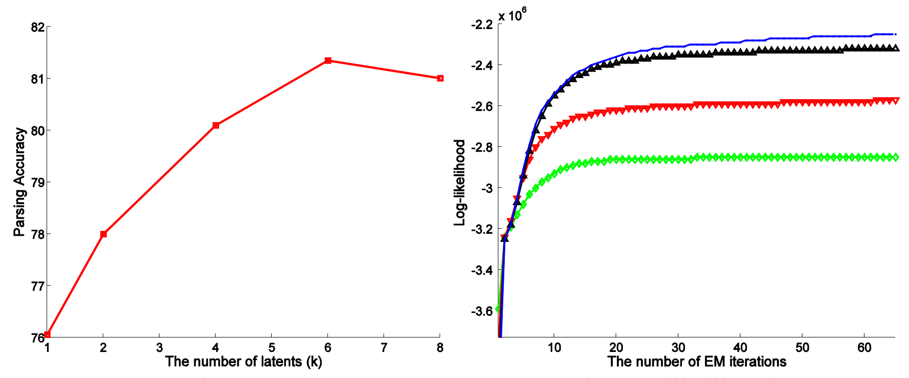
For generating *N*-best candidate parse trees, we adopted Klein's vertical and horizontal markovization [3]. We simply call it *Klein's markovization*. As for vertical markovization level (v) and horizontal level (h), we selected as v = 2 and h = 2. Table 1 shows the oracle performance for *N*-best parse trees of Klein's markovization for different *N*s *(1, 50, 100, 200, 300,* and *500),* compared with the result of PCFG. Klein's markovization showed a large difference from PCFG's result when *N* is 1. As *N* is larger, the performance difference becomes smaller.

5.3 Performance of Collins-LA: Effects of the Number of Latent Variables

We set *N* to 200, thus generated top-200 candidate parse trees and re-ranked them according to the generation probability of Collins-LA (BaseNP version in section 4). Figure 1 shows the parsing accuracy of Collins-LA, and the log-likelihood curve on the number of EM iterations, for different *k*s (1, 2, 4, 6 and 8). As shown in Figure 1, the parsing accuracy increases as k is larger, but decreases when k is more than 6. This result is probably caused by the data sparseness problem that seriously arises when k is 8. Log-likelihood at the final EM iterations is in proportional to k.

5.4 Comparison with Matsuzaki's PCFG-LA

To compare Collins-LA with the performance of Matsuzaki's PCFG-LA, we re-implemented PCFG-LA. The full binarization of PCFG obtained from the tree-bank is not desirable since a serious data sparseness problem is involved. To this end, we used a light backbone grammar based on the binarization with center-head technique which Matsuzaki used [5], and further with Klein's vertical and horizontal markovizations [3]. To see this binarization scheme in more detail, let us consider the original production rule(Figure 2).
Then, figure 3 shows the types of rules of our backbone grammar obtained by binarizing the original rule of figure 2.



(a) Parsing accuracy (b) Log-likelihood curve

Figure 1. Parsing accuracy of Collins-LA for different *k*s, and log-likelihood curve.

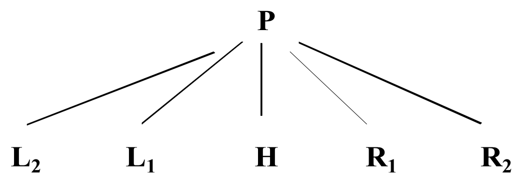
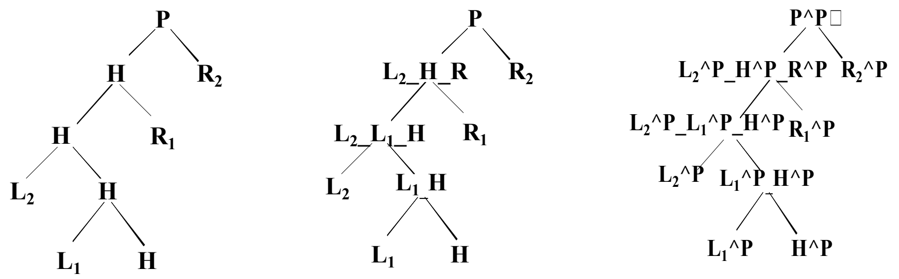


Figure 2. A given production rule.



a) v = 1, h = 0 b) v = 1, h = 2 c) v = 2, h = 2

Figure 3. Three center-head binarizations are presented using a) Klein's markovization, b) Klein's markovization with more horizontal markovization, and c) Klein's markovizatoin with Johnson's parent annotation. We used b) and c) for evaluating PCFG-LA.

Through several experiments, we found that PCFG-LA becomes the best when *h* is 2. If *h* is more than 2, then the parsing accuracy decreased. Thus, we selected this horizontal option (i.e. *h* =2). We optimized the number of latent variables for each non-terminal node. We assigned the number for latent variables differently according to their relative frequencies in the training corpus. During this, we assumed that the maximum number of latent variables is 8. Once our assigned the number, the non-terminal symbols related to noun phrases and verb phrases have 8 different latent variables, due to their large relative frequencies, while a coordination symbol (CC) has only one latent variable. For generating top *N* parse trees, we used the same grammar (Klein's markovization) described in section 5.2.

Table 2 summarizes the best performances of Collins-LA (*k* = 6) and PCFG-LA (*k* = 8 in maximum).

Table 2. The final parsing accuracy of Collins-LA and PCFG-LA.

	All length			Length ≤ 40		
	LP	LR	F1	LP	LR	F1
Baseline	77.78	77.35	77.57	79.21	78.66	78.9
Collins-LA	80.59	79.63	80.11	82.02	80.68	81.35
PCFG-LA(v=1,h=2)	81.58	80.07	80.82	83.23	81.43	82.32
PCFG-LA(v=2,h=2)	82.62	81.46	**82.04**	84.29	82.86	**83.57**

As shown in Table 2, Collins-LA is comparable to PCFG-LA (*v*=1, *h*=2), without showing much difference. Our result of PCFG-LA is less-performed than the result of Matsuzaki's one. The reason for this is that the initial quality of top-N parse trees is different. In Matsuzaki's case, top-1 parsing accuracy of the initial parse trees reaches about 80% performance, which is much more superior to about 77% of our top-1 parse tree. We further evaluated PCFG-LA using the backbone grammar based on the binarization with Johnson's parent annotation (*v*=2, *h*=2). Clearly, PCFG-LA with parent annotation (*v*=2, *h*=2) is superior to Collins-LA and PCFG-LA (*v*=1, *h*=2). Note that PCFG-LA with parent annotation uses a more complex backbone grammar, but Collins-LA uses simple PCFG grammar of (*v*=1, *h*=0). The grammar used in Collins-LA is more simple than even PCFG-LA (*v*=1, *h*=2). Regarding this, the current result of Collins-LA can be more improved when adopting the elaborated backbone grammar. Although some additional experiments are necessary, this experiment identifies that the formalism of Collins' head-driven model provides a useful framework as PCFG does for latent annotation. This is a sufficiently notable result in the way of the research of latent annotation.

6 Conclusion

This paper proposed the extension of Collins' head-driven model with the latent annotation, namely Collins-LA. We provided a novel training algorithm of Collins-LA, based on an inside-outside algorithm, and further extended it to consider the special consideration for BaseNP by embedding a forward-backward algorithm within the inside-outside algorithm. Experimental results are inspiring, showing that Collins-LA is comparable to PCFG-LA which is equipped with a more elaborated backbone grammar. Regarding that our current setting of Collins-LA is at an initial stage, its performances could be much improved if more accurate grammars and formalizations are adopted. The work for improving and extending Collins' head-driven model is important since it can be more flexibly applied to non-English languages than PCFG. In the future, we will continue to explore Collins-LA on a more complicated backbone grammar, and use a more elaborated linguistic considerations, without losing the elegance and principle of the original model.

Acknowledgements. This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Advanced Information Technology Research Center (AITrc), also in part by the BK 21 Project and MIC & IITA through IT Leading R&D Support Project in 2007.

References

1. Gildea, D.: Corpus variation and parser performance. In: EMNLP '01. (2001), 167–172
2. Bikel, D.M.: Intricacies of collins' parsing model. Computational Linguististics 30(4) (2004) 479–511
3. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: ACL '03. (2003) 423–430
4. Johnson, M.: PCFG models of linguistic tree representations. Computational Linguistics 24(4) (1998) 613–632
5. Matsuzaki, T., Miyao, Y., Tsujii, J.: Probabilistic CFG with latent annotations. In: ACL '05. (2005) 75–82
6. Petrov, S., Barrett, L., Thibaus, R., Klein, D.: Learning accurate, compact, and interpretable tree annotation. In: COLING-ACL '06. (2006) 433–440
7. Charniak, E., Johnson, M.: Coarse-to-fine n-best parsing and maxent discriminative reranking. In: ACL 2005. (2005) 173–180
8. Collins, M.: Head-driven statistical models for natural language parsing. Computational Linguististics 29(4) (2003) 589–637
9. Prescher, D.: Head-driven PCFGs with latent-head statistics. In: IWPT '05. (2005) 115–124
10. Charniak, E.: Statistical parsing with a context-free grammar and word statistics. In: AAAI/IAAI 1997. (2005) 598–603

Study on Architectures for Chinese POS Tagging and Parsing

Hailong Cao, Yujie Zhang and Hitoshi Isahara

3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, 619-0289, Japan
{hlcao, yujie, isahara}@nict.go.jp

Abstract. How to deal with part of speech (POS) tagging is a very important problem when we build a syntactic parsing system. We could preprocess the text with a POS tagger before perform parsing in a pipelined approach. Alternatively, we could perform POS tagging and parsing simultaneously in an integrated approach. Few, if any, comparisons have been made on such architecture issues for Chinese parsing. This paper presents an in-depth study on this problem. According to comparison experiments, we find that integrated approach can make significantly better performance both on Chinese parsing and unknown words POS tagging than the pipelined approach. As for known words POS tagging, we find that the two approaches get similar tagging accuracy, but the tagging results of integrated approach do lead to much better parsing performance. We also analyze the reasons account for the performance difference.

1 Introduction

POS tag is an important feature in most of the parsing models as having a word's POS tag can help us determine what kind of syntactic constituent the word can compose. So usually it is necessary to assign a proper POS tag to each word in a sentence which is to be parsed. We could adopt the pipelined approach which performs parsing strictly after POS tagging, or performs POS tagging and parsing simultaneously in an integrated approach. The pipelined approach is simple and fast but is subject to error propagation. Though integrated approach can make decision from global view in theory, whether it can get better accuracy in practice is still an open question since little detailed comparison has been made between pipelined and integrated approaches for Chinese parsing.

This paper presents an in-depth study on such issues for Chinese parsing. We compare the performances of the pipelined approach, the integrated approach and two kinds of compromise strategies. There are three findings in our experiments. First, integrated approach can improve parsing performance by considering POS tag of known word globally though it can not enhance the known words tagging accuracy. Second, integrated approach can get better tagging accuracy on unknown words and therefore get better parsing result. Third, better tagging results do not always lead to better parsing results. Our comparison experiments suggest that fully integrated approach is the best strategy for Chinese parsing if complexity is not a major concern. We also analyze the reasons that account for the performance difference.

Received 11/10/07
Accepted 07/12/07
Final Version 17/01/08

2 Lexicalized POS Tagging Model Based on HMM

As the first step of our investigation, we build a separate POS tagger without considering syntactic information. Our tagger takes segmented sentences as input; formally it is a sequence with n words:

$$W = w_1, w_2, \ldots\ldots w_n$$

We assign each word in the sentence an appropriate POS tag by a lexicalized hidden Markov model (HMM).

Usually there are more than one POS tag sequences for a given word sequence since there are usually more than one POS tags for a single word. The statistical POS tagging method based on Bayesian model is capable of assigning a POS tag sequence with the greatest conditional probability, which is shown as follows:

$$Tag_{best} = \arg\max_{Tag} P(Tag \mid W) = \arg\max_{Tag} \frac{P(Tag, W)}{P(W)} = \arg\max_{Tag} P(Tag, W) \tag{1}$$

Where $Tag = t_1, t_2, \ldots\ldots t_n$ is a candidate POS tag sequence for W.

The classical HMM assumes that the transformation from one state (that means POS here) to another is not affected by the current observation value (that means the current word), and the generation of current observation value is independent from other observation values. That is:

$$P(Tag, W) = P(Tag)P(W \mid Tag) \approx \prod_{i=1}^{n} P(t_i \mid t_1, t_2 \ldots\ldots, t_{i-1}) \prod_{i=1}^{n} P(w_i \mid t_1, t_2 \ldots\ldots, t_n) \tag{2}$$

Furthermore, only N previous states are considered when the current state is generated. And only the current state is involved when the current word is generated:

$$P(Tag, W) = P(Tag)P(W \mid Tag) \approx \prod_{i=1}^{n} P(t_i \mid t_{i-N}, t_{i-N+1} \ldots\ldots t_{i-1}) \prod_{i=1}^{n} P(w_i \mid t_i) \tag{3}$$

This is the so-called N-order model or the (N+1)-gram model. In practice, bi-gram or tri-gram model is often used to alleviate data sparseness.

In fact, we observed that there is a close association between POS tags and words in Chinese text, the above model can not well reflect the characteristic of Chinese. In order to capture the relation between POS tags and words in Chinese text, we augment HMM by the method below:

$$\begin{aligned} Tag_{best} &= \arg\max P(Tag, W) = \\ &\arg\max \prod_{i=1}^{n} P(t_i, w_i \mid t_1, w_1 \ldots\ldots t_{i-1}, w_{i-1}) \approx \arg\max \prod_{i=1}^{n} P(t_i, w_i \mid t_{i-1}, w_{i-1}) \end{aligned} \tag{4}$$

By doing this transformation, we can correct the HMM's unpractical assumption and introduce lexical information into POS tagging model to strengthen its discriminative ability.

2.1 Data Smoothing

Data sparseness problem is more serious after we introduce lexical information. This makes it necessary to utilize some data smoothing method. From equation (4), we get:

$$P(t_i,w_i|t_{i-1},w_{i-1}) = P_1(t_i|t_{i-1},w_{i-1})P_2(w_i|t_{i-1},w_{i-1},t_i) \tag{5}$$

In this way, we can smooth the P_1 and P_2 in equation (5) using the following method:

$$P_1(t_i|t_{i-1},w_{i-1}) = \lambda_1 P_{ML1}(t_i|t_{i-1},w_{i-1}) + (1-\lambda_1)p_{ML1}(t_i|t_{i-1}) \tag{6}$$

$$\begin{aligned} P_2(w_i|t_{i-1},w_{i-1},t_i) &= \lambda_{21} P_{ML2}(w_i|t_{i-1},w_{i-1},t_i) \\ &+ (1-\lambda_{21})[\lambda_{22}(P_{ML2}(w_i|t_{i-1},t_i) + (1-\lambda_{22})P_{ML2}(w_i|t_i)] \end{aligned} \tag{7}$$

λ_1, λ_{21} and λ_{22} are smoothing parameters and $P_{ML}(x|y)$ is the empirical probability estimated from the data in the training set by using maximal likelihood estimation method:

$$P_{ML}(x|y) \equiv \frac{count(x,y)}{count(y)} \tag{8}$$

2.2 Unknown Words Processing

Unknown words[1] processing is an important issue when we build POS tagger. Much work has been done on guessing the POS tag of unknown words. For convenience, the simplest algorithm is applied in our tagger. The tagger pretends that every POS tag in the tag set is a candidate for each unknown word, with equal probability. This means that the POS tag of an unknown word is predicted using lexical forms and POS tags of surrounding words without considering the lexical form of the unknown word. For more details of this algorithm, please refer to (Jurafsky and Martin, 2000).

We found that many unknown words are number words consist of Arabic numeral such as "1, 2, 3" or Chinese numeral such as "一, 二, 三". So other than pretending that every POS tag is possible, we simply tag an unknown word as CD(cardinal number) if it includes any Arabic or Chinese numeral.

2.3 Tagging Results

In our experiments, both the tagging model and the parsing model are trained and tested on the Penn Chinese Treebank 5.1(abbreviated as CTB, Xue et al., 2005) which contains 507,216 words, 18,782 sentences. We use the article 301-325 for testing. Article 001-270 and article 400-1151 are used for training. For comparison, we use the bi-gram HMM as a baseline for the lexicalized HMM tagging model. Table 1 shows the evaluation results. In all this paper, evaluation is implemented on

[1] In this paper, we define the word that does not appear in training set as unknown word.

sentences which have no more than 40 words. Among all the words in test sentences, 8.17% are unknown words.

Table 1. Evaluation of tagging results

	Accuracy on all words	Accuracy on known words	Accuracy on unknown words
Bi-gram HMM	91.79%	95.93%	45.32%
Lexicalized HMM	93.84%	97.09%	57.25%

We can see that the lexicalized HMM outperforms bi-gram HMM significantly both on known words and unknown words tagging.

In order to further confirm the performance of lexicalized HMM, we test it on English Penn tree bank which is much bigger than CTB. We use 39832 sentences from section 02-21 as training data and 2416 sentences from section 23 as test data. We get an accuracy of 96.77% which is close to state of the art of English POS tagging. So we can think 93.84% tagging accuracy on CTB is capable as a baseline in our following comparison experiments.

3 Parsing based on Collins' Model 2

The parsing model we start with is the well-known head-lexicalized model proposed by Collins (Collins, 1999). Given an input sentence $S=(w_1/\ t_1, \ldots\ldots w_n/t_n\ )$ the most likely parse tree defined by a statistical generative model is:

$$T_{best} = \underset{T}{argmax} P(T|S) = argmax\frac{P(T,S)}{P(S)} = argmax P(T,S) \tag{9}$$

Probabilistic context-free grammar (PCFG) is one of the simple methods that is used to model distributions over sentence/parse-tree pairs. If there are k context free grammar rules in the parse tree, then:

$$P(T,S) = \prod_{i=1..k} P(RHS_i \mid LHS_i) \tag{10}$$

where LHS /RHS standards for the left/right hand side of the grammar rule.

Based on PCFG, Collins proposed a lexicalized model by associating a word w and a part of speech tag t to each non-terminal node in the parse tree. Formally, a grammar rule LHS $\rightarrow$ RHS can be written as:

$$Parent(t,w) \rightarrow L_m\ (t,w) \ldots\ldots L_1(t,w)H(t,w)R_1\ (t,w)\ \ldots\ldots\ R_n\ (t,w)$$

where *Parent* is the father node and *H* is the head child, $L_m \ldots\ldots L_1$ and $R_1 \ldots\ldots R_n$ are left and right modifiers of *H*.

To overcome the sparseness problem caused by the introduction of lexical items, the generation of RHS is broken down into a Markov process that makes certain independence assumptions, and the probability of a grammar rule is defined as:

$$P(RHS \mid LHS) =$$

$$P_h(H \mid Parent(t,w)) \cdot \prod_{i=1}^{m+1} P_l(L_i(t,w) \mid Parent(t,w), H) \cdot \prod_{i=1}^{n+1} P_r(R_i(t,w) \mid Parent(t,w), H) \tag{11}$$

where L_{m+1} and R_{n+1} are stop categories. The probability P_h, P_l and P_r are estimated by maximum likelihood estimation method.

When we adopt Collins' model to parse Chinese, the head percolation table from (Xia, 1999) is used to find the head of constituent in CTB. Collins' model 2 also includes sub-categorization frame. So it is necessary to make complement/adjunct distinction in training data. We label the following three types of no-terminal as complement:

(1) NP, CP (Sub clause) or IP (simple clause) whose parent is IP.
(2) NP, CP, VP or IP whose parent is VP.
(3) IP whose parent is CP.

In addition, the non-terminal will not be labeled as complement if it is the head child of its parent. For more details such as parameter estimation and special preprocessing of punctuations, we refer the reader to (Collins, 1999) and (Bikel, 2004(a)).

4 Parsing Experiments on Different POS Tagging Strategies

It is necessary to assign a proper POS tag to each word in a sentence which is to be parsed. We could perform parsing strictly after POS tagging, or perform POS tagging and parsing in an integrated approach simultaneously. So in order to test which architecture is best for Chinese parsing, we perform four experiments with different setting. The first and the fourth experiments are based on pipelined approach and integrated approach respectively. The second and third are based on compromise strategy by which we mean the input sentence is tagged by a front-end tagger first and then some words will be re-tagged by the parser.

4.1 Parsing Strictly after POS Tagging

In this approach, we just input the output of our POS tagger into the parser based on Collins' model. The task of parser is to find the maximum probability tree whose terminal/non-terminal nodes are just W/Tag. Table 2 shows the experimental results which can be used as a baseline for the following experiments. For completeness, table 2 also includes the tagging performance which has been shown in table 1.

4.2 Compromise Strategy 1: Re-tag Known Word by Parser

The main question of the compromise strategy is to determine which words should be re-tagged by the parser. In Collins English parser, the input sentence is tagged by a front-end tagger.

Table 2. Tagging and parsing results

Tagging performance		
Accuracy on all words	Accuracy on known words	Accuracy on unknown words
93.84%	97.09%	57.25%
Parsing performance		
Precision	Recall	F1
81.84%	82.14%	81.99%

Table 3. Tagging and parsing results

Tagging performance		
Accuracy on all words	Accuracy on known words	Accuracy on unknown words
93.94%	97.20%	57.25%
Parsing performance		
Precision	Recall	F1
83.22%	83.11%	83.16%

Table 4. High frequency tagging error types on known words

Lexicalized HMM			Compromise strategy		
Error type		Count	Error type		Count
Gold tag	Error tag		Gold tag	Error tag	
CC	AD	5	M	NN	4
M	NN	5	NR	NN	5
JJ	NN	6	VA	NN	5
VA	NN	6	DEC	DEG	9
NN	M	7	JJ	NN	10
NN	JJ	8	NN	JJ	10
DEG	DEC	17	NN	M	10
VV	NN	17	DEG	DEC	11
NN	VV	18	VV	NN	15
DEC	DEG	27	NN	VV	18

Among the outputs of the tagger, only unknown words' POS tags are kept and known words are re-tagged by the parser. So in this section, we simply follow this strategy. For a known word, the possible POS tags are limited to those which have been seen in training data for that word. Table 3 shows the experimental results.

Comparing table 2 and table 3, we can see that tagging accuracy on unknown words is still 57.25% since the parser does not change them at all. As for known words, tagging accuracy increased from 97.09% to 97.20%. Although the increase in POS tagging is very small, a significant improvement is achieved in parsing performance. In order to find the reason why the compromise strategy obtains improvements on parsing, we analyze the tagging results in detail. We find that differences do exist between the two tagging results in section 4.1 and section 4.2. The two tagging strategies give different POS tag on 2.33% of all the known words. Table 4 shows top 10 frequent error types of two tagging results.

From table 4 we can see that the main advantage of compromise strategy is to disambiguate DEC and DEG. DEC and DEG are two possible POS tag of the auxiliary word “的” which is used frequently in Chinese text. DEC means that there will be a clause named CP before it, such as：

```
[NP [CP [IP [NP [PN 他们]]
              [VP [VV 工作]]]
         [DEC 的]]
     [NP [NN 单位]]]
  (The place where they work.)
```

DEG means that there will be a simple phrase named DNP before it such as:

```
[NP [DNP [NP [NR 约翰]]
            [DEG 的]]
     [NP [NN 书]]]
        (John’s book.)
```

We can see that the POS tag of “的” is very important to determine the syntactic structure of the words before it. So mis-tagging the word “的” will trigger much more parsing errors.

On the other hand, table 4 shows that compromise strategy makes more errors in disambiguating NN-M and NN-JJ than lexicalized HMM. But the parsing errors aroused by such tagging errors are usually limited to the local part of the mis-tagged words. As a result, compromise strategy can improve parsing performance though it can not enhance the overall known words tagging accuracy.

Inspired by the above analysis, a straightforward idea is just re-tag the word “的” with the parser and keep the POS tags of other words unchanged. Table 5 shows the experimental results. As we expected, the tagging accuracy is increased from 97.20% to 97.55%. However, the F1 score of parsing is decreased from 83.16%to 82.94%. It seems that better tagging results do not always lead to better parsing results. The reason for this interesting phenomenon is currently beyond our knowledge and further investigation is necessary.

Table 5. Tagging and parsing results

Tagging performance		
Accuracy on all words	Accuracy on known words	Accuracy on unknown words
94.26%	97.55%	57.25%
Parsing performance		
Precision	Recall	F1
82.93%	82.94%	82.94%

4.3 Compromise Strategy 2: Re-tag Unknown Words by Parser

There is no reason to believe that unknown word must be tagged by a front-end tagger. In contrast of original Collins' model, we try to re-tag unknown words by the parser and do not make any change on known word's tag assigned by lexicalized HMM. When parsing, the parser enumerates every POS tag in tag set for each unknown word. Table 6 shows the experimental results.

Table 6. Tagging and parsing results

Tagging performance		
Accuracy on all words	Accuracy on known words	Accuracy on unknown words
94.20%	97.09%	61.63%
Parsing performance		
Precision	Recall	F1
82.28%	82.33%	82.30%

Comparing table 2 with table 6, we can see that Collins' model is superior to lexicalized HMM on unknown words POS tagging because the accuracy is increased to 61.63% from 57.25%. There are two differences on tagging between Collins' model and the lexicalized HMM. First, HMM generates a word's tag conditioning on information of its previous word, while Collins' model generates a word's tag conditioning on information of the head word which the word depends on. Second, Collins' model uses higher level syntax structure information additionally. We think these two differences account for the improvement on unknown words POS tagging.

However, it should be noted that 61.63% unknown words tagging accuracy is still very low because the approach is too simple. We think there is much room for enhancing the parsing accuracy by applying more effective unknown words tagging algorithm.

4.4 Integrated approach: Tag all Words by Parser

Since both known words re-tagging and unknown words re-tagging can improve parsing performance, we continue to test how much gains we can get when all words are tagged by the parser. In this integrated approach, the input of the parser is a sequence of words without any POS information. Table 7 shows the experimental results.

Comparing table 2 and table 7, we can see the integrated approach can make significantly better performance both on POS tagging and parsing. In detail, the performance gain on tagging is mainly from unknown words while the improvement on known word tagging is trivial. With the effect of both known word re-tagging and unknown words re-tagging, the F1 score is raised from 81.99% to 83.49%.

Furthermore, we can see that known words tagging accuracy is higher than what we get in section 4.2; unknown words tagging accuracy is higher than what we get in section 4.3. This suggests that fully integrated approach is the best strategy for Chinese parsing if complexity is not a major concern.

Table 7. Tagging and parsing results

Tagging performance		
Accuracy on all words	Accuracy on known words	Accuracy on unknown words
94.34%	97.22%	62.02%
Parsing performance		
Precision	Recall	F1
83.64%	83.34%	83.49%

5 Related Work

Much work has been done on parsing CTB. (Bikel and Chiang, 2000) presented the first result of CTB parsing based on BBN model and TIG model. (Chiang and Bikel, 2002) proposed an automatic method to determine the head child based on the EM algorithm. (Hearne and Way, 2004) applied Data-Oriented Parsing approach to CTB. (Xiong et al., 2005) proposed a semantic-class based parsing method. (Wang et al., 2006) presented a deterministic Chinese parser. In all these work, tagging and parsing is performed in the pipelined approach.

(Bikel, 2004(b)) built a multi-language parsing engine which can be extended to Chinese. In this work, tagging and parsing is performed in a compromise approach.
(Levy and Manning, 2003) applied factored model to Chinese parsing and achieved much improvement by grammar transformation. In this work, tagging and parsing is performed in an integrated approach.

The research most similar to ours is (Jiang, 2004). However, on the issue of POS tagging, Jiang's findings are quite different from ours. Jiang found if the parser retags the known words, tagging accuracy can be increased from 90.42% to 92.42%. When Jiang inputted untagged sentences into the parser, the parsing F1 score dropped from 81.1% to 78.07%.

In addition, (Luo, 2003) and (Fung et al., 2004) constructed character based parser. Luo's results showed that higher-level syntactic structures are of little use to word segmentation.

As for English parsing, how to deal with POS tagging is also an open question. (Charniak et al. 1996) investigated the use of POS taggers that output multiple tags for parsing and concluded that single taggers are preferable. However, (Watson, 2006) found that multiple-tag per word can improve on parser accuracy at the cost of efficiency. (Yoshida et al., 2007) also showed that the accuracy of parsers could be improved by allowing POS taggers to output multiple answers for some words.

6 Conclusion and Future Work

It is necessary to assign a proper POS tag to each word in a sentence which is to be parsed. We could adopt the pipelined approach which performs parsing strictly after POS tagging, or perform POS tagging and parsing simultaneously in an integrated approach. This paper presents an in-depth study on such architecture issues for Chinese parsing. There are three findings in our experiments. First, integrated approach can improve parsing performance by considering POS tag of known word globally though it can not enhance the known words tagging accuracy. Second, integrated approach can get better tagging accuracy on unknown words and therefore get better parsing result. Third, better tagging results do not always lead to better parsing results. Our comparisons suggest that fully integrated approach is the best strategy for Chinese parsing if complexity is not a major concern.

There are at least two directions for the future work. First, now the unknown words tagging accuracy in both pipelined approach and integrated approach are very low, therefore more effective unknown words tagging algorithm should be applied in future. Second, lexicalized HMM predicts a word's tag based on local information while parsing model predicts that based on long distance dependency, so simultaneously use both local and long distance feature for tagging is another direction of future work.

References

1. Daniel M. Bikel and David Chiang. 2000. Two Statistical Parsing Models Applied to Chinese Treebank. In Proceedings of the 2nd Chinese language processing workshop.
2. Daniel M. Bikel. 2004(a). Intricacies of Collins' Parsing Model. In Computational Linguistics, 30(4): 479-511.
3. Daniel M. Bikel. 2004(b). On the Parameter Space of Generative Lexicalized Statistical Parsing Models. Ph.D. thesis, University of Pennsylvania.
4. David Chiang and Daniel Bikel. 2002. Recovering Latent Information in Treebanks. In Proceedings of the 19th International Conference on Computational Linguistics.
5. Deyi Xiong, Shuanglong Li, Qun Liu et al. 2005. Parsing the Penn Chinese Treebank with Semantic Knowledge. In Proceedings of the Second International Joint Conference Natural language processing.
6. Daniel Jurafsky, James H. Martin. 2000. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Prentice-Hall.
7. Eugene Charniak, Glenn Carroll, John Adcock et al. 1996. Taggers for Parsers. Artificial Intelligence, 85(1-2):45–57.
8. Fei Xia. 1999. Automatic Grammar Generation from Two Different Perspectives. PhD thesis, University of Pennsylvania.
9. Kazuhiro Yoshida, Yoshimasa Tsuruoka, Yusuke Miyao, et al. 2007. Ambiguous Part-of-Speech Tagging for Improving Accuracy and Domain Portability of Syntactic Parsers. In Proceedings of the Twentieth International Joint Conference on Artificial Intelligence.
10. Mary Hearne and Andy Way. 2004. Data-Oriented Parsing and the Penn Chinese Treebank. In Proceedings of the First International Joint Conference Natural language processing.

11. Mengqiu Wang, Kenji Sagae and Teruko Mitamura. 2006. A Fast, Accurate Deterministic Parser for Chinese. In Proceedings of COLING/ACL.
12. Michael Collins. 1999. Head-driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania.
13. Nianwen Xue, Fei Xia, Fu-Dong Chiou et al. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. Natural Language Engineering, 11(2):207-238.
14. Pascale Fung, Grace Ngai, Yongsheng Yang et al. 2004. A Maximum-Entropy Chinese Parser Augmented by Transformation-Based Learning. ACM Transactions on Asian Language Processing, 3(2):159-168.
15. Rebecca Watson. 2006. Part-of-speech Tagging Models for Parsing. In Proc. of CLUK.
16. Roger Levy and Christopher Manning. 2003. Is it Harder to Parse Chinese, or the Chinese Treebank? In Proceedings of ACL.
17. Xiaoqiang Luo. 2003. A Maximum Entropy Chinese Character-Based Parser. In Proceedings of the conference on Empirical methods in Natural Language Processing.
18. Zhengping Jiang. 2004. Statistical Chinese parsing. Honours thesis, National University of Singapore.

Maximum Entropy Based Bengali Part of Speech Tagging

Asif Ekbal[1], Rejwanul Haque[2], and Sivaji Bandyopadhyay[3]

Computer Science and Engineering Department, Jadavpur University, Kolkata, India
asif.ekbal@gmail.com[1], rejwanul@gmail.com[2], sivaji_cse_ju@yahoo.com[3]

Abstract. Part of Speech (POS) tagging can be described as a task of doing automatic annotation of syntactic categories for each word in a text document. This paper presents a POS tagger for Bengali using the statistical Maximum Entropy (ME) model. The system makes use of the different contextual information of the words along with the variety of features that are helpful in predicting the various POS classes. The POS tagger has been trained with a training corpus of 72, 341 word forms and it uses a tagset [1] of 26 different POS tags, defined for the Indian languages. A part of this corpus has been selected as the development set in order to find out the best set of features for POS tagging in Bengali. The POS tagger has demonstrated an accuracy of 88.2% for a test set of 20K word forms. It has been experimentally verified that the lexicon, named entity recognizer and different word suffixes are effective in handling the unknown word problems and improve the accuracy of the POS tagger significantly. Performance of this system has been compared with a Hidden Markov Model (HMM) based POS tagger and it has been shown that the proposed ME based POS tagger outperforms the HMM based tagger.

Keywords: Part of Speech Tagging, Maximum Entropy Model, Bengali.

1 Introduction

Part of Speech (POS) tagging is the task of labeling each word in a sentence with its appropriate syntactic category called part of speech. Part of speech tagging is a very important preprocessing task for language processing activities. This helps in doing deep parsing of text and in developing Information extraction systems, semantic processing etc. Part of speech tagging for natural language texts are developed using linguistic rules, stochastic models and a combination of both. Stochastic models [1] [2] [3] have been widely used in POS tagging task for simplicity and language independence of the models. Among stochastic models, Hidden Markov Models (HMMs) are quite popular. Development of a stochastic tagger requires large amount of annotated corpus. Stochastic taggers with more than 95% word-level accuracy have been developed for English,

[1] http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf

Received 28/10/07
Accepted 07/12/07
Final Version 18/01/08

German and other European languages, for which large labeled data are available. The problem is difficult for Indian languages (ILs) due to the lack of such annotated large corpus.

Simple HMMs do not work well when small amount of labeled data are used to estimate the model parameters. Incorporating diverse features in an HMM-based tagger is also difficult and complicates the smoothing typically used in such taggers. In contrast, a Maximum Entropy (ME) based method [4] or a Conditional Random Field based method [5] can deal with diverse, overlapping features. A POS tagger has been proposed in [6] for Hindi, which uses an annotated corpus (15,562 words collected from the BBC news site), exhaustive morphological analysis backed by high coverage lexicon and a decision tree based learning algorithm (CN2). The accuracy was 93.45% for Hindi with a tagset of 23 POS tags.

International Institute of Information Technology (IIIT), Hyderabad, India initiated a POS tagging and chunking contest, NLPAI ML [2] for the Indian languages in 2006. Several teams came up with various approaches and the highest accuracies were 82.22% for Hindi, 84.34% for Bengali and 81.59% for Telugu. As part of the SPSAL workshop [3] in IJCAI-07, a competition on POS tagging and chunking for south Asian languages was conducted by IIIT, Hyderabad. The best accuracies reported were 78.66% for Hindi [7], 77.61% for Bengali [8] and 77.37% for Telugu [7].

In this paper, we have developed a POS tagger based on the statistical Maximum Entropy (ME) model. The POS tagger produces an accuracy of 84.6% for the development set with the contextual window of size three, prefixes and suffixes of length up to three characters of the current word, NE information of the current and previous words, POS information of the previous word, digit features, symbol features, length of the word and the various inflection lists. It has been experimentally shown that the accuracy of the POS tagger can be improved significantly by considering unknown word features, named entity recognizer [9] and a lexicon [10] as the means of dealing with the unknown words. Evaluation results with a test set of 20K word forms shows the effectiveness of the proposed model with an accuracy of 88.2%.

The rest of the paper is organized as follows. Section 2 describes briefly the Maximum Entropy framework. Section 3 elaborately describes our approach to the POS tagging task. Experimental results of the development and the test sets are reported in Section 4. Finally, Section 5 concludes the paper.

2 Maximum Entropy Model

The Maximum Entropy framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints are derived from the training data, expressing some relationships between features and outcome. The probability distribution that satisfies

[2] http://ltrc.iiitnet/nlpai_contest06/proceedings.php
[3] http://shiva.iiit.ac.in/SPSAL2007/SPSAL-Proceedings.pdf

the above property is the one with the highest entropy. It is unique, agrees with the maximum likely-hood distribution, and has the exponential form:

$$P(t|h) = \frac{1}{Z(h)} exp(\sum_{j=1}^{n} \lambda_j f_j(h,t)) \tag{1}$$

where, t is the POS tag, h is the context (or history), $f_j(h,t)$ are the features with associated weight λ_j and $Z(h)$ is a normalization function.

The problem of POS tagging can be formally stated as follows. Given a sequence of words $w_1, \ldots, w_n$, we want to find the corresponding sequence of POS tags $t_1, \ldots, t_n$, drawn from a set of tags T, which satisfies:

$$P(t_1, \ldots, t_n | w_1, \ldots, w_n) = \prod_{i=1,2\ldots,n} P(t_i|h_i) \tag{2}$$

where, h_i is the context for the word w_i.

The *Beam search algorithm* is then used to select the sequence of word classes with the highest probability.

The features are binary/multiple valued functions, which associate a POS tag with various elements of the context. For example:

$$\begin{align} f_j(h,t) &= 1 \ \text{ if } \ word(h) = sachin \ \text{ and } \ t = NNP \tag{3} \\ &= 0 \quad \text{otherwise} \tag{4} \end{align}$$

The general-purpose optimization method *Limited Memory BFGS method* [11] has been used for the estimation of MaxEnt parameters. We have used the C++ based Maximum Entropy package. [4]

3 Our Approach for Part Of Speech Tagging in Bengali

Bengali is one of the widely used languages all over the world. In terms of native speakers, it is the seventh popular language in the world, second in India and the national language of Bangladesh. The works on POS tagging in Indian languages, particularly in Bengali, has started very recently as there was neither any standard POS tagset nor any available tagged corpus just one/two years ago. In this work, we have used a Maximum Entropy approach for the task of POS tagging. Along with the variety of contextual and word level features, a lexicon [10] and a HMM based named entity recognizer [9] have been used to improve the accuracy of the POS tagger.

3.1 Features

Feature selection plays a crucial role in the ME framework. Experiments have been carried out to find out the most suitable features for POS tagging in Bengali. The main features for the POS tagging task have been identified based

[4] http://homepages.inf.ed.ac.uk/s0450736/software/maxent/maxent-20061005.tar.bz2

on the different possible combination of available word and tag context. The features also include prefix and suffix for all words. The term prefix/suffix is a sequence of first/last few characters of a word, which may not be a linguistically meaningful prefix/suffix. The use of prefix/suffix information works well for highly inflected languages like the Indian languages. We have considered different combinations from the following set for inspecting the best feature set for POS tagging in Bengali.

F=$\{w_{i-m}, \ldots, w_{i-1}, w_i, w_{i+1}, \ldots w_{i+n}$, $|\text{prefix}| \leq n$, $|\text{suffix}| \leq n$, Current and/or the surrounding named entity (NE) tag(s), Previous POS tag(s), First word, Lexicon, Digit information, Length of the word, Inflection lists$\}$.

Following are details of the set of features that have been applied for POS tagging in Bengali:
• Context word feature: Preceding and following words of a particular word might be used as a feature.
• Word suffix:Word suffix information is helpful to identify POS class. This feature can be used in two different ways. The first and the naive one is, a fixed length (say, n) word suffix of the current and/or the surrounding word(s) can be treated as features. If the length of the corresponding word is less than or equal to *n-1* then the feature values are not defined and denoted by ND. The feature value is also not defined (ND) if the token itself is a punctuation symbol or contains any special symbol or digit. The second and the more helpful approach is to modify the feature as binary/mutiple valued. Variable length suffixes of a word can be matched with predefined lists of useful suffixes for different classes. Here, both the suffixes have been used. The second type of suffixes has been described later in the form of noun, verb and adjective inflections.
• Word prefix: Prefix information of a word is also helpful. A fixed length (say, n) prefix of the current and/or the surrounding word(s) can be considered as features. This feature value is not defined (ND) if the length of the corresponding word is less than or equal to *n-1* or the word is a punctuation symbol or the word contains any special symbol or digit.
• Part of Speech (POS) Information: POS information of the previous word(s) might be used as a feature. This is the only dynamic feature in the experiment.
• Named Entity Information: The named entity (NE) information of the current and/or the surrounding word(s) plays an important role in the overall accuracy of the POS tagger. In order to use this feature, an HMM-based Named Entity Recognition (NER) system [9] has been used. The NER system uses the four major NE classes namely, *Person name*, *Location name*, *Organization name* and *Miscellaneous name.* Date, time, percentages, numbers and monetary expressions belong to the *Miscellaneous name* category. The other words are assigned the 'NNE' tags. The NER system was developed using a portion of the Bengali news corpus [12], developed from the archive of a leading Bengali newspaper available in the web. This NER system has demonstrated 84.5% F-Score value during 10-fold cross validation test with a training corpus of 150k wordforms.

The NE information can be used in two different ways. The first one is to use the NE tag(s) of the current and/or the surrounding word(s) as the features

of the ME model. The second way is to use this NE information at the time of testing. In order to do this, the test set is passed through the HMM based NER system. Outputs of the NER system are given more priorities than the outputs of the POS tagger for the unknown words in the test set. The NE tags are then replaced appropriately by the POS tags (NNPC: Compound proper noun, NNP: Proper noun and QFNUM: Quantifier number).
• Lexicon Feature: A lexicon in Bengali has been used to improve the performance of the POS tagger. The lexicon [10] has been developed in a semi-supervised way from the Bengali news corpus [12] of 34-million word forms. It contains the Bengali root words and their basic POS information such as: noun, verb, adjective, pronoun and indeclinable. The lexicon has 100,000 entries.

This lexicon can be used in two different ways. One way is to use this as the features of the ME model. To apply this, five different features are defined for the open class of words as follows:

1. If the current word is found to appear in the lexicon with the 'noun' POS, then the feature 'Lexicon' is set to 1.
2. If the current word is found to appear in the lexicon with the 'verb' POS, then the feature 'Lexicon' is set to 2.
3. If the current word is found to appear in the lexicon with the 'adjective' POS, then the feature 'Lexicon' is set to 3.
4. If the current word is found to appear in the lexicon with the 'pronoun' POS, then the feature 'Lexicon' is set to 4.
5. If the current word is found to appear in the lexicon with the 'indeclinable' POS, then the feature 'Lexicon' is set to 5.

The intention of using this feature was to distinguish the noun, verb, adjective, pronoun and indeclinable words among themselves.

The second or the alternative way is to use this lexicon during testing. For an unknown word, the POS information extracted from the lexicon is given more priority than the POS information assigned to that word by the ME model. An appropriate mapping has been defined from these five basic POS tags to the 26 POS tags.
•Made up of digits: For a token if all the characters are digits then the feature "ContainsDigit" is set to 1; otherwise, it is set to 0. It helps to identify QFNUM (Quantifier number) tag.
•Contains symbol: If the current token contains special symbol (e.g., %, $ etc.) then the feature "ContainsSymbol" is set to 1; otherwise, it is set to 0. This helps to recognize SYM (Symbols) and QFNUM (Quantifier number) tags.
•Length of a word: Length of a word might be used as an effective feature of POS tagging. If the length of the current token is more than *three* then the feature 'LengthWord' is set to 1; otherwise, it is set to 0. The motivation of using this feature was to distinguish proper nouns from the other words. We have observed that very short words are rarely proper nouns.
•Frequent word list: A list of most frequently occurring words in the training corpus has been prepared. The words that occur more than 10 times in the entire training corpus are considered to be the frequent words. The feature 'RareWord'

is set to 1 for those words that are in this list; otherwise, it is set to 0.
•Function words: A list of function words has been prepared manually. This list has 743 number of entries. The feature 'FunctionWord' is set to 1 for those words that are in this list; otherwise, the feature is set to 0.
•Inflection Lists: Various inflection lists have been created manually by analyzing the various classes of words in the Bengali news corpus [12]. A simple approach of using these inflection lists is to check whether the current word contains any inflection of these lists and to take decision accordingly. Following are the lists of inflections:

– Noun inflection list: A list of inflections that occur with noun words has been manually prepared by observing the various noun words in the Bengali news corpus. This list contains 27 entries. If the current word has any one of these inflections then the feature 'Inflection' is set to 1.
– Adjective inflection list: It has been observed that adjectives in Bengali generally occur in four different forms based on the suffixes attached. The first type of adjectives can form comparative and superlative degree by attaching the suffixes (e.g., *-tara* and *-tamo*) to the adjective word. The second set of suffixes (e.g., *-gato*, *-karo* etc.) make the words adjectives while get attached with the noun words. The third group of suffixes (e.g., *-janok*, *-sulav* etc.) identifies the POS of the word form as adjective. These three set of suffixes are included in a single list and a feature 'AdjectiveInflection' is defined as: if the current word contains any suffix of the list then 'Inflection' is set to 2. This list has been manually constructed by looking at the different adjective words of the Bengali news corpus. At present, this list has 194 entries.
– Verb inflection list: In Bengali, the verbs can be organized into 20 different groups according to their spelling patterns and the different inflections that can be attached to them. Original word-form of a verb word often changes when any suffix is attached to the verb. At present, there are 214 different entries in the verb inflection list. If the current word is in the verb inflection list then the feature 'Inflection' is set to 3.
  The feature 'Inflection' is set to 0 for those words that do not contain any of these inflections.

3.2 Unknown Word Handling Techniques

Handling of unknown word is an important issue in POS tagging. For words, which were not seen in the training set, $P(t_i|w_i)$ is estimated based on the features of the unknown words, such as whether the word contains a particular suffix. The list of suffixes has been prepared. This list contains 435 suffixes; many of them usually appear at the end of verb, noun and adjective words. The probability distribution of a particular suffix with respect to specific POS tag is calculated from all words in the training set that share the same suffix.

In addition to the unknown word suffixes, a named entity recognizer [9] and a lexicon [10] have been used to tackle the unknown word problems. The details of the procedure is given below:

1. Step 1: Find the unknown words in the test set.
2. Step 2: The system assigns the POS tags, obtained from the lexicon, to those unknown words that are found in the lexicon. For noun, verb and adjective words of the lexicon, the system assigns the NN (Common noun), VFM (Verb finite main) and the JJ (Adjective) POS tags, respectively.
   Else
3. Step 3: The system considers the NE tags for those unknown words that are not found in the lexicon
   (a) Step 2.1: The system replaces the NE tags by the appropriate POS tags (NNPC [Compound proper noun] and NNP [Proper noun]).
   Else
4. Step 4: The remaining words are tagged using the unknown word features accordingly.

4 Experimental Results

The ME based POS tagger has been trained on a corpus of 72,341 word forms tagged with the 26 POS tags, defined for the Indian languages. This 26-POS tagged training corpus was obtained from the NLPAI ML Contest-2006 [5] and SPSAL-2007 [6] contest data. The NLPAI ML 2006 contest data was tagged with 27 different POS tags and had 46,923 wordforms. This POS tagged data was converted into the 26-POS [7]tagged data by defining appropriate mapping. The SPSAL-2007 contest data was tagged with 26 POS tags and had 25,418 wordforms. Out of 72,341 word forms, around 15K word forms have been selected as the development set and the rest has been used as the training set of the ME based tagger in order to find out the best set of features for POS tagging in Bengali.

We define the *baseline* model as the one where the NE tag probabilities depend only on the current word:

$$P(t_1, t_2, \ldots, t_n | w_1, w_2, \ldots, w_n) = \prod_{i=1,\ldots,n} P(t_i, w_i).$$

In this model, each word in the test data will be assigned the POS tag, which occurred most frequently for that word in the training data. The unknown word is assigned the POS tag with the help of lexicon, named entity recognizer and word suffixes.

Thirty two different experiments were conducted taking the different combinations from the set 'F' to identify the best-suited set of features for POS tagging in Bengali. From our empirical analysis, we have found that the following combination gives the best result for the development set:
F=[w_{i-1}, w_i, w_{i+1}, |prefix| ≤ 3, |sufix| ≤ 3, NE tags of the current and previous

[5] http://ltrc.iiitnet/nlpai_contest06/data2
[6] http://shiva.iiit.ac.in/SPSAL2007/check_login.php
[7] http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf

words, POS tag of the previous word, Lexicon feature, Symbol feature, Digit feature, Length feature, Inflection lists].

The meanings of the notations, used in the experiments, are defined below:

pw, cw, nw: Previous, current and the next word; pwi, nwi: Previous and the next ith word; pre, suf: Prefix and suffix of the current word; pp: POS tag of the previous word; ppi: POS tag of the previous ith word; pn, cn, nn: NE tags of the previous, current and the next word; pni: NE tag of the previous ith word. Evaluation results of the system for the development set are presented in Tables 1-2.

Table 1. Experimental results for the development set

Feature (word, tag)	Accuracy (in %)
pw, cw, nw	63.11
pw2, pw, cw, nw, nw2	**67.22**
pw3, pw2, pw, cw, nw, nw2, nw3	65.12
pw2, pw, cw, nw, nw2, pp	69.72
pw2, pw, cw, nw, nw2, pp, pp2	69.12
pw2, pw, cw, nw, nw2, pp, $\lvert pre\rvert \le 6$, $\lvert suf\rvert \le 6$	70.42
pw2, pw, cw, nw, nw2, pp, $\lvert pre\rvert \le 5$, $\lvert suf\rvert \le 5$	71.7
pw2, pw, cw, nw, nw2, pp, $\lvert pre\rvert \le 4$, $\lvert suf\rvert \le 4$	72.10
pw2, pw, cw, nw, nw2, pp, $\lvert suf\rvert \le 3$, $\lvert pre\rvert \le 3$	73.81
pw, cw, nw, pp, $\lvert suf\rvert \le 3$, $\lvert pre\rvert \le 3$	**75.16**
pw, cw, nw, pp, $\lvert pre\rvert \le 3$, $\lvert ppre\rvert \le 3$, $\lvert psuf\rvert \le 3$, $\lvert suf\rvert \le 3$	73.36

Evaluation results (3rd row) of Table 1 show that word window [−2,+2] gives the best result without any feature. Results also show the fact that further increase (4th row) or decrease (2nd row) in window size reduces the accuracy of the POS tagger. Experimental results (5th and 6th rows) show that the accuracy of the POS tagger can be improved by including the POS information of the previous word(s). Clearly, it is seen that POS information of the previous word only is more helpful. Experimental results (7th-11th rows) show the effectiveness of prefixes and suffixes up to a particular length for the highly inflective Indian languages as like Bengali. Results (11th row) also show that the window $[-1, +1]$ yields better result than the window $[-2, +2]$ with the same set of features. Observations from the results (12th row) suggest that the surrounding word suffixes and/or prefixes do not increase the accuracy.

It can be decided from the results (2nd-5th rows) of Table 2 that the named entity (NE) information of the current and/or the surrounding word(s) improves the overall accuracy of the POS tagger. It is also indicative from this results (3rd row) that the NE information of the previous and current words, i.e, within the window [−1,0] is more effective than the NE information of the windows [−1,+1], [0,+1] or the current word alone. An improvement of 2.1% in the overall accuracy

Table 2. Experimental results for the development set

Feature (word, tag)	Accuracy (in %)
pw, cw, nw, pp, pn, cn, nn, $\|suf\| \leq 3$, $\|pre\| \leq 3$	77.1
pw, cw, nw, pp, pn, cn, $\|suf\| \leq 3$, $\|pre\| \leq 3$	**78.2**
pw, cw, nw, pp, cn, nn, $\|pre\| \leq 3$, $\|suf\| \leq 3$	77.8
pw, cw, nw, pp, cn, $\|pre\| \leq 3$, $\|suf\| \leq 3$	76.2
pw, cw, nw, pp, pn, cn, $\|pre\| \leq 3$, $\|suf\| \leq 3$, Digit, Symbol, Length, FunctionWord	80.3
pw, cw, nw, pp, pn, cn, $\|pre\| \leq 3$, $\|suf\| \leq 3$, Digit, Symbol, Length, FunctionWord, Lexicon	82.1
pw, cw, nw, pp, pn, cn, $\|pre\| \leq 3$, $\|suf\| \leq 3$, Digit, Symbol, FunctionWord, Lexicon, Inflection	84.2
pw, cw, nw, pp, pn, cn, $\|pre\| \leq 3$, $\|suf\| \leq 3$, Digit, Symbol, Length, Lexicon, Inflection, FunctionWord, RareWord	**84.6**

is observed (6th row) with the introduction of the 'Symbol', 'Digit', 'Length' and 'FunctionWord' features. The use of lexicon as the features of ME model further improves the accuracy by 1.8% (7th row). Accuracy of the POS tagger rises to 84.2% (8th row) by including the noun, verb and adjective inflections. Finally, an overall accuracy of 84.6% is obtained with the inclusion of 'RareWord' feature.

Evaluation results of the POS tagger by including the various mechanisms for handling the unknown words are presented in Table 3 for the development set. The table also shows the results of the *baseline* model.

Table 3. Overall evaluation results of the POS tagger for the development set

Model	Accuracy (in %)
Baseline	55.9
ME	84.6
ME + Lexicon (used for unknown word handling)	85.8
ME + Lexicon (used for unknown word handling) + NER (used for unknown word handling)	87.1
ME + Lexicon (used for unknown word handling) + NER (used for unknown word handling)+ Unknown word features	**88.9**

Clearly, the results of Table 3 show the effectiveness of employing the various strategies for handling the unknown words with the increase in the accuracy values. The ME model exhibits an accuracy of 88.9% with the lexicon, named entity recognizer and unknown word features. The system has demonstrated an improvement in the overall accuracy by 1.2% with the use of lexicon [10] as a mean to handle the unknown word problems. The accuracy of the tagger is further improved by 1.3% when NER system [9] is included additionally to handle the unknown words. This improvement shows the effectiveness of NER during POS tagging. The overall accuracy of the POS tagger increases by 1.8% with the inclusion of unknown word features.

Now, the development set is included as part of the training set and thus the resultant training set consists of 72,341 word forms. A gold standard test set of 20K word forms has been used in order to report the evaluation results of the system. Experimental results of the system along with the *baseline* model are demonstrated in Table 4 for the test set. The POS tagger has demonstrated the overall accuracy of 88.2% for the test set.

Table 4. Experimental results of the system for the test set

Model	Accuracy (in %)
Baseline	54.7
ME	83.8
ME + Lexicon	84.9
ME + Lexicon + NER	86.3
ME + Lexicon + NER + Unknown word features	**88.2**

The performance of the proposed ME based POS tagger has been compared with a HMM based POS tagger [13], in which trigram model was considered. Additional context dependent features were considered for the emission probabilities in this HMM based POS tagger . Also, the lexicon [10], named entity recognizer [9] and unknown word features were considered in a similar way in order to handle the unknown word problems. This system has been trained and tested with the same data. Evaluation results of the HMM based POS tagger is presented in Table 5.

Table 5. Evaluation results of the HMM based POS tagger for the test set

Model	Accuracy (in %)
HMM	75.8
HMM + Lexicon	76.9
HMM + Lexicon + NER	78.1
HMM + NER + Lexicon + Unknown word features	**80.3**

Evaluation results of Table 4 and Table 5 show that the proposed ME based POS tagger outperforms the HMM based POS tagger [13] with the same training and test sets. So, it can be decided that the ME framework is more effective than the HMM framework in handling the morphologically complex Indian languages that contain diverse, overlapping features.

Error analysis of the POS tagger has been done with the help of confusion matrix. Since nouns appear most frequently in a corpus, unknown words have a tendency of being assigned noun tags (NN, in most cases). A close scrutiny of the confusion matrix suggests that some of the probable tagging errors facing the current POS tagger are NNC (Compound common noun) vs NN (Common noun), JJ (Adjective) vs NN (Common noun) and VFM (Verb finite main) vs VAUX (Common noun). A multiword extraction unit for Bengali would have taken care of the NNC vs NN problem. The problems of JJ vs NN is hard to resolve and probably requires the use of linguistic rules. The problems of VFM vs VAUX can be solved with the use of linguistic rules. Errors involving NNP vs NN are reduced but not fully removed even after the use of NER system. We need to improve the accuracy of the NER system further in order to remove this problem.

A close investigation to the evaluation results show that approximately 92% of the errors are involved with the unseen words in the test set. We have incorporated several mechanisms, as discussed earlier, for dealing with the unknown word problems. Experiments have been conducted to observe the accuracies of the unknown words separately. Evaluation results are presented in Table 6. It is observed from the evaluation results that unknown word features are the most effective in handling the unseen words followed by the NER system and the lexicon.

Table 6. Evaluation results of the POS tagger for the unknown words in the test set

Model	Accuracy (in %)
ME	68.4
ME + Lexicon	73.3
ME + Lexicon + NER	79.7
ME + Lexicon + NER + Unknown word features	**88.1**

5 Conclusion

We have developed a POS tagger using Maximum Entropy model that has good accuracy with the contextual window of size three, prefix and suffix of length three, NE information of the current and previous words, POS information of the previous word, digit features, symbol features, length of the word and the various inflection lists. The accuracy of this system has been improved significantly by incorporating several techniques such as word level features, named entity recognizer and lexicon for handling unknown word problem. The performance of the proposed model has been compared with a HMM based POS tagger. The proposed system has demonstrated an accuracy of 88.2%, which is an improvement of approximately 8% over the HMM based tagger.

Analyzing the performance using other methods like Conditional Random Fields and Support Vector Machines (SVMs) will be other interesting experiments.

References

1. Cutting, D., Kupiec, J., Pedersen, J., Sibun, P.: A practical part-of-speech tagger. In: Proceedings of the Third Conference on Applied Natural Language Processing. (1992) 133–140
2. Merialdo, B.: Tagging english text with a probabilistic model. Comput. Linguist. **20**(2) (1994) 155–171
3. Brants, T.: TnT a Statistical Parts-of-Speech Tagger. In: Proceedings of the Sixth Conference on Applied Natural Language Processing ANLP-2000. (2000) 224–231
4. Ratnaparkhi, A.: A maximum entropy part-of -speech tagger. In: Proc. of EMNLP'96. (1996)
5. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: ICML. (2001) 282–289
6. Singh, S., Gupta, K., Shrivastava, M., Bhattacharyya, P.: Morphological richness offsets resource demand-experiences in constructing a pos tagger for hindi. In: Proceedings of the COLING/ACL 2006. (2006) 779–786
7. Avinesh, P., Karthik, G.: Part Of Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning. In: Proceedings of IJCAI Workshop on Shallow Parsing for South Asian Languages, India (2007) 21–24
8. Dandapat, S.: Part Of Specch Tagging and Chunking with Maximum Entropy Model. In: Proceedings of the IJCAI Workshop on Shallow Parsing for South Asian Languages, Hyderabad, India (2007) 29–32
9. Ekbal, A., Naskar, S., Bandyopadhyay, S.: Named Entity Recognition and Transliteration in Bengali. Named Entities: Recognition, Classification and Use, Special Issue of Lingvisticae Investigationes Journal **30**(1) (2007) 95–114
10. Ekbal, A., Bandyopadhyay, S.: Lexicon Development and POS Tagging using a Tagged Bengali News Corpus. In: Proceedings of the 20th International Florida AI Research Society Conference (FLAIRS-2007), Florida (2007) 261–263
11. Malouf, R.: A comparison of algorithms for maximum entropy parameter estimation. In: Proceedings of Sixth Conf. on Natural Language Learning. (2002) 49–55
12. Ekbal, A., Bandyopadhyay, S.: A Web-based Bengali News Corpus for Named Entity Recognition. Language Resources and Evaluation Journal (accepted) (2007)
13. Ekbal, A., Mondal, S., Bandyopadhyay, S.: POS Tagging using HMM and Rule-based Chunking. In: Proceedings of the IJCAI Workshop on Shallow Parsing for South Asian Languages, Hyderabad, India (2007) 31–34

Morphology and Word Segmentation

A Two-Pass Search Algorithm for Thai Morphological Analysis

Canasai Kruengkrai and Hitoshi Isahara

Graduate School of Engineering, Kobe University
1-1 Rokkodai-cho, Nada-ku, Kobe 657-8501 Japan
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289 Japan
{canasai,isahara}@nict.go.jp

Abstract. Considering Thai morphological analysis as a search problem, the approach is to search the most likely path out of all candidate paths in the word lattice. However, the search space may not contain all possible word hypotheses due to the unknown word problem. This paper describes an efficient algorithm called the *two-pass search algorithm* that first recovers missing word hypotheses, and then searches the most likely path in the expanded search space. Experimental results show that the two-pass search algorithm improves the performance of the standard search by 3.23 F_1 in word segmentation and 2.92 F_1 in the combination of word segmentation and POS tagging.

1 Introduction

Morphological analysis has been recognized as a fundamental process in Thai text analysis. In Thai, words are written continuously without word boundaries like other non-segmenting languages (e.g., Chinese and Japanese). However, the Thai writing system has certain unique characteristics. For example, in order to form a smallest linguistic unit, a character cluster, including a consonant, a vowel, and/or a tonal mark, must be formed.

Thai morphological analysis generally involves two tasks: segmenting a character string into meaningful words, and assigning words with the most likely part-of-speech (POS) tags. Considering Thai morphological analysis as a search problem, the approach is to search the most likely path out of all candidate paths in a word lattice. Figure 1 illustrates the word lattice of a string analysis, consisting of possible word hypotheses and their connections. The path indicated by the bold line is the correct segmentation.

Discriminating such path from other candidate paths is a difficult problem itself, and requires a dynamic programming search (e.g., the Viterbi algorithm). The problem becomes more difficult, since the search space is often imperfect. Some word hypotheses are missing due to the unknown word problem. Unknown words are words that do not occur in the system dictionary or the training corpus. The system has no knowledge to use in generating hypotheses for unknown words.

Received 15/10/07
Accepted 07/12/07
Final Version 21/01/08

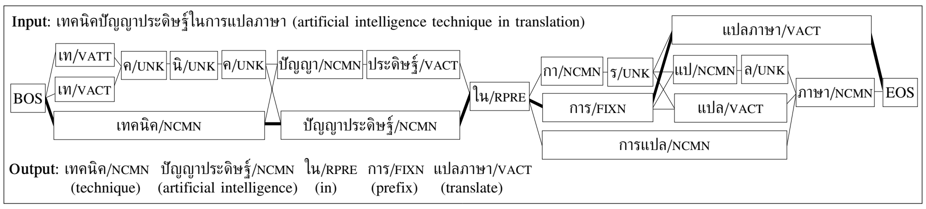

Input: เทคนิคปัญญาประดิษฐ์ในการแปลภาษา (artificial intelligence technique in translation)
BOS
เท/VATT
เท/VACT
ค/UNK
นิ/UNK
ค/UNK
เทคนิค/NCMN
ปัญญา/NCMN
ประดิษฐ์/VACT
ปัญญาประดิษฐ์/NCMN
ใน/RPRE
กา/NCMN
ร/UNK
การ/FIXN
การแปล/NCMN
แปลภาษา/VACT
แป/NCMN
ล/UNK
แปล/VACT
ภาษา/NCMN
EOS
Output: เทคนิค/NCMN ปัญญาประดิษฐ์/NCMN ใน/RPRE การ/FIXN แปลภาษา/VACT
(technique) (artificial intelligence) (in) (prefix) (translate)

Fig. 1. Lattice produced from an unsegmented string

In this paper, we propose a new statistical model for Thai morphological analysis that jointly tackles word segmentation, POS tagging, and unknown word problems. Our assumption is that the system could analyze a given string accurately (both known and unknown words), if the search space contains reasonably potential word hypotheses. Such space can be generated by using the so-called *two-pass search algorithm*. In the first pass, we use a dictionary and writing rules to build an initial lattice for a string input. We then identify suspicious word hypotheses in the lattice and expand new word hypotheses from all possible substrings within uncertainty ranges. Finally, in the second pass, we search the optimal path in the expanded search space by applying a lattice-based Viterbi algorithm. Experimental results show that the two-pass search algorithm improves the performance of the standard search by 3.23 F_1 in word segmentation and 2.92 F_1 in the combination of word segmentation and POS tagging.

2 Framework for Thai Morphological Analysis

2.1 Problem Formulation

We formulate the problem of Thai morphological analysis on word lattice representation. A lattice is an ordered graph that efficiently represents rich information about sequences of word hypotheses. Nodes in the graph are word hypotheses with their morphological information (e.g., POS tags and character types), and arcs are transitions between word hypotheses.

Given an unsegmented string, the lattice is produced by expanding possible word hypotheses from left to right. If a sequence of characters can be matched with a word surface in the system dictionary, word hypotheses are then generated corresponding to lexical entries. If no word can be found in the system dictionary, Thai writing rules are applied to generate character clusters. These character clusters are assigned with a set of open-class POS tags (e.g., noun and verb) to be candidate word hypotheses.

Based on word lattice representation, we define the process of Thai morphological analysis as a search problem by the following equation:

$$\boldsymbol{y}^* = \mathrm{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{y}|\boldsymbol{x}) = \mathrm{argmax}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{y}) \ , \tag{1}$$

where $\boldsymbol{x}$ is a character sequence $c_1 c_2 \ldots c_{|\boldsymbol{x}|}$, $\boldsymbol{y}$ is a sequence of word hypotheses, and $\mathcal{Y}(\boldsymbol{x})$ is a lattice containing a set of whole-sentence analyses for $\boldsymbol{x}$. Note that we can drop $\boldsymbol{x}$ in the last step, since it is a fixed character sequence and does not affect the argmax. Our objective is to search the most likely path $\boldsymbol{y}^*$ in the lattice $\mathcal{Y}(\boldsymbol{x})$.

2.2 Discriminative Learning

In our model, we apply discriminative learning to estimate model parameters. The idea is to learn how to discriminate the correct path from other candidate paths with features. This is in contrast to most previous research for Thai morphological analysis, which estimates model parameters using only the correct path in a given training corpus. Discriminate learning estimates model parameters based on the search space that the system will expect to see in practice. Our problem formulation can be advantageously used for discriminative learning, since the search space is formed in the word lattice. Both learning and decoding are processed in the same problem structure.

We now factorize the probability $p(\boldsymbol{y})$ further by using an instance of the log-linear models with bigram features [7]. Thus, we obtain:

$$p(\boldsymbol{y}) = \frac{1}{Z} \exp\left\{ \sum_{t=1}^{\#\boldsymbol{y}} \sum_{k=1}^{K} \lambda_k f_k(y_{t-1}, y_t) \right\} , \tag{2}$$

where $\#\boldsymbol{y}$ is a number of word hypotheses in a path $\boldsymbol{y}$ which varies among candidate paths, f_k is a feature operated on bigram transition between y_{t-1} and y_t, λ_k is a parameter to be estimated. To ensure that $\sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{y}) = 1$, the normalization factor Z is given by:

$$Z = \sum_{\boldsymbol{y}' \in \mathcal{Y}(\boldsymbol{x})} \exp\left\{ \sum_{t=1}^{\#\boldsymbol{y}'} \sum_{k=1}^{K} \lambda_k f_k(y'_{t-1}, y'_t) \right\} , \tag{3}$$

which is a sum over all paths $\boldsymbol{y}'$ in the lattice.

In learning, given a set of training data $\quad = \boldsymbol{y}^{(i)} \;{}_{i=1}^{N}$ where N is the number of all training samples, we need to find a set of feature weights $\boldsymbol{\lambda}$. We use a penalized log-likelihood maximization, in which the objective function defines:

$$\log p(\boldsymbol{\lambda}| \quad) = \sum_{i=1}^{N} \log p_{\boldsymbol{\lambda}}(\boldsymbol{y}^{(i)}) \quad \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2} . \tag{4}$$

One can use a numerical method for unconstrained optimization to solve Equation 4. In our implementation, we apply the L-BFGS algorithm [8], which is known to be a very efficient optimization method used in large NLP tasks.

To summarize, the system learns the model parameters by representing whole training samples with lattices, and extracting the global feature space. It then iterates on each lattice to estimate feature weights using the Viterbi-style algorithm (explained in the next section). The system tries to discriminate the correct path from incorrect paths using the current estimation of feature weights, and computes the expectation of feature weights based on resulting errors. The system proceeds until it converges to a point where the change of the objective function is less than a given threshold.

2.3 Decoding

In decoding, we need to find the most likely path such that $\boldsymbol{y}^* = \text{argmax}_{\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} p(\boldsymbol{y})$. To avoid an exponential-time search over all candidate paths in the lattice $\mathcal{Y}(\boldsymbol{x})$, we thus adapt the Viterbi algorithm to the lattice structure. Let $\quad(y)$ be a set of nodes that connect to node y from the left. The Viterbi algorithm stores a partial probability δ of the most likely path reaching node y, which can be expressed in a recursive form:

$$\delta_y = \max_{y'\in\mathcal{L}(y)} \left[\delta_{y'} \exp\left(\sum_{k=1}^{K} \lambda_k f_k(y', y)\right)\right] . \tag{5}$$

The partial best path is one that achieves the maximal probability. The recursion terminates at the end of string (eos):

$$\phi^*_{y_{\text{eos}}} = \text{argmax}_{y'\in\mathcal{L}(y_{\text{eos}})} \delta_{y'} . \tag{6}$$

Finally, we can backtrack through the lattice with the back pointer ϕ to recover the global best path $\boldsymbol{y}^*$.

3 Two-Pass Search Algorithm

As mentioned earlier, in practice, the search space is not complete like in learning where all words are known. Some word hypotheses are missing due to the unknown word problem. There are two possible approaches to handle this issue: increasing the coverage of the system dictionary, or using a model that can process unknown words. In this paper, we are interested in the latter approach. Our goal is to recover missing word hypotheses based on the learned model. Here we design the two-pass search algorithm that finds the most likely path in the expanded search space.

The two-pass search algorithm computes the marginal probability of each node in the lattice, which reflects the uncertainty of that node. Nodes that have their probabilities less than a given a threshold, denoted by ϵ, are considered as suspicious nodes. Thus, we can identify uncertainty ranges in the input string that might contain parts of known or unknown words depended on their probabilities. We generate all possible substrings (sequences of character clusters) within uncertainty ranges, and expand nodes in the lattice according to these substrings. We then search the most likely path in the new lattice that contains much better coverage of word hypotheses. Algorithm 1 provides an outline of the two-pass search algorithm.

Algorithm 1: TWO-PASS SEARCH ALGORITHM

```
input  : An unsegmented string $\boldsymbol{x}$.
output : The most likely path $\boldsymbol{y}^* = y_1, \ldots, y_{\#\boldsymbol{y}}$.

  • (1-pass) Build an initial lattice $\mathcal{Y}(\boldsymbol{x})$, and estimate the probability $p_y$ of each
    node $y$ in the lattice using Equations 7, 8 and 9.
  • Find nodes that $p_y < \epsilon$ or node-state = UNK, and mark their surface lengths.
  • Generate all possible substrings within uncertainty ranges, and expand new nodes
    in $\mathcal{Y}(\boldsymbol{x})$ according to these substrings.
  • (2-pass) Perform the Viterbi search to get $\boldsymbol{y}^*$ using Equations 5 and 6.
```

Let α_y be a forward probability and β_y be a backward probability reaching node y from the left and right, respectively. The marginal probability of a particular node y can be calculated by:

$$p_y = \frac{\alpha_y \ \beta_y}{Z} , \tag{7}$$

where

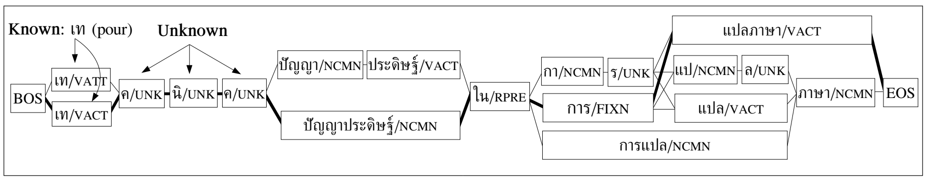
$$\alpha_y = \sum_{y' \in \mathcal{L}(y)} \left[\alpha_{y'} \exp\Big(\sum_{k=1}^{K} \lambda_k f_k(y', y) \Big) \right] , \tag{8}$$

$$\beta_y = \sum_{y' \in \mathcal{R}(y)} \left[\beta_{y'} \exp\Big(\sum_{k=1}^{K} \lambda_k f_k(y, y') \Big) \right] . \tag{9}$$

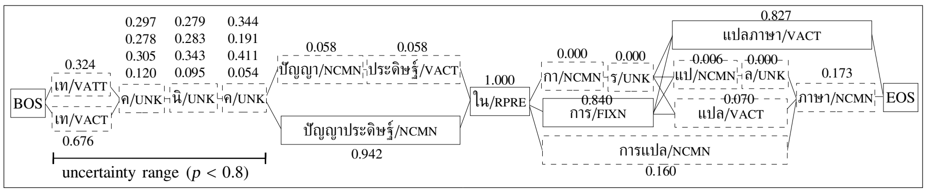
Note that the forward and backward probabilities are calculated with recursion similar to the Viterbi algorithm, except using sum function instead of max function. We can apply the Forward-Backward algorithm to obtain p_y for a given node y.

Let us describe the two-pass search algorithm more clearly with a motivating example. Recall the lattice in Figure 1, and suppose that the word 'technique' does not appear in the system dictionary. The system tries to build a lattice where the predicted path now contains both known and unknown nodes (Figure 2(a)). This situation often occurs in practice, when a word hypothesis is missing and the system searches the most likely path based on available information. If we stop at this point, we obtain an incorrect segmentation.

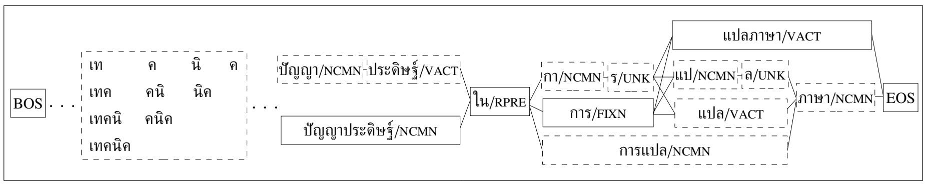
When the two-pass search algorithm is performed, the probability of each node in the lattice is calculated (Figure 2(b)). Note that nodes tagged with UNK show a list of probabilities, since they represent a number of candidate word hypotheses actually tagged with open-class POS tags. In this example, four tags are assigned, including NCMN, NPRP, VACT and VATT. The algorithm identifies the suspicious nodes that their probabilities are less than a threshold $\epsilon = 0.8$, and then all possible substrings within the uncertainty range are generated (Figure 2(c)). Finally, the algorithm can recover the correct path in the new search space (Figure 2(d)).



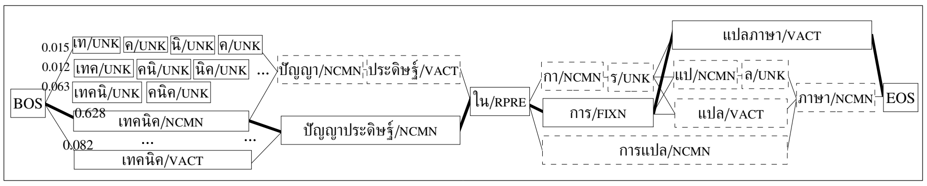
(a) Lattice when a word hypothesis is missing



(b) The suspicious node indicated by the dashed line box are identified, and the uncertainty range which has no potential nodes in the paths is located



(c) Possible substrings are generated within the uncertainty range



(d) Lattice with new candidate word hypotheses. The correct path can be recoverd in the expanded search space

Fig. 2. Examples show how the two-pass search algorithm proceeds

4 Experiments

4.1 Experimental Settings

We conducted our experiments on ORCHID corpus, which is a Thai part-of-speech tagged corpus [9]. The original texts were taken from technical papers published in the proceedings of the National Electronics and Computer Technology Center (NECTEC) annual conference. ORCHID has been known to be a hard dataset for evaluation, since it consists of a large number of unknown words. In experiments, we divided the corpus into the training and test sets according to Table 1. We converted all spacial tags for punctuation to original forms, e.g. , <colon> ‘:’. We also merged space tokens to their next tokens, and considered them as leading spaces of words. The system dictionary was generated from unique tokens occurred in the training set and TCL’s computational lexicon [5].

Table 1. Details of dataset used in our experiments

corpus	ORCHID
POS tags	15 categories and 48 subcategories
# of training samples	9,234 (papers published in ’89-’90)
# of training tokens	116,102
# of test samples	5,808 (papers published in ’91)
# of test tokens	71,097
# of known test tokens	57,381
# of unknown test tokens	9,403 (not include punctuation)
# of entries in the system dictionary	55,791
# of features	2,203,932

Since our problem formulation is based on a log-linear model, a variety of features can be incorporated in the model. Table 2 provides the details of features used in our experiments. We broadly classified character types into 6 categories, including symbol, alphabet, number, ordinal number, Thai and other character types. Note that a word w_i is considered to be a rare word, if it occurs less than 2 times in the training set.

For the propose of comparison, we used the maximal matching algorithm for word segmentation and the unigram baseline method for POS tagging. The idea of the unigram baseline method is to assign each token with the most likely POS tag that can be estimated from the frequency count in the training set [4]. For word segmentation, precision is defined as the percentage of tokens recovered by the system that also occurred in the test set, while recall is defined as the percentage of tokens in the test set recovered by the system. For full morphological analysis (word segmentation and POS tagging), a token is considered to be a correct one only when both the word boundary and its POS tags are correctly identified. The following metrics are used to evaluate the performance.

Table 2. Features are based on three-level, first-order Markov assumption. Each node y contains morphological information of word surface w, POS p and sub-POS p'. $f_k(\cdot)$ is operated on $y_{t-1} \rightarrow y_t$

Unigram Feature	
condition	feature template
$\forall y_t$	$\langle p_t \rangle$
	$\langle p_t, p'_t \rangle$
	character type of $w_t \times \{\phi, \langle p_t \rangle, \langle p_t, p'_t \rangle\}$
w_t is not rare	$w_t \times \{\phi, \langle p_t \rangle, \langle p_t, p'_t \rangle\}$
w_t is rare	up to 2 prefixes of $w_t \times \{\phi, \langle p_t \rangle, \langle p_t, p'_t \rangle\}$
	up to 2 suffixes of $w_t \times \{\phi, \langle p_t \rangle, \langle p_t, p'_t \rangle\}$
Bigram Feature	
condition	feature template
$\forall y_{t-1} \rightarrow y_t$	$\langle w_{t-1}, w_t \rangle$
(if w_t is rare, replace	$\langle p_{t-1}, p_t \rangle$
its surface with a	$\langle w_{t-1}, w_t, p_t \rangle$
symbol '$*$')	$\langle w_{t-1}, p_{t-1}, w_t \rangle$
	$\langle p_{t-1}, p_t, p'_t \rangle$
	$\langle p_{t-1}, p'_{t-1}, p_t \rangle$
	$\langle w_{t-1}, p_{t-1}, p_t \rangle$
	$\langle p_{t-1}, w_t, p_t \rangle$
	$\langle w_{t-1}, p_{t-1}, w_t, p_t \rangle$
	$\langle w_{t-1}, p_{t-1}, p_t, p'_t \rangle$
	$\langle p_{t-1}, p'_{t-1}, w_t, p_t \rangle$
	$\langle p_{t-1}, p'_{t-1}, p_t, p'_t \rangle$
	$\langle w_{t-1}, w_t, p_t, p'_t \rangle$
	$\langle w_{t-1}, p_{t-1}, p'_{t-1}, w_t \rangle$
	$\langle w_{t-1}, p_{t-1}, p'_{t-1}, w_t, p_t, p'_t \rangle$

$$\text{Recall} = \frac{\#\text{ of correct tokens}}{\#\text{ of tokens in the test set}}$$

$$\text{Precision} = \frac{\#\text{ of correct tokens}}{\#\text{ of tokens recovered by the system}}$$

$$F_1 = \frac{2\ \ \text{Recall}\ \ \text{Precision}}{\text{Recall} + \text{Precision}}$$

$$\text{Recall}_{known} = \frac{\#\text{ of correct known tokens}}{\#\text{ of known tokens in the test set}}$$

$$\text{Recall}_{unknown} = \frac{\#\text{ of correct unknown tokens}}{\#\text{ of unknown tokens in the test set}}$$

4.2 Results

Tables 3 and 4 show the performance of the two-pass search algorithm by varying the probability threshold ϵ in the range of [0.5, 0.8]. The probability threshold

Table 3. Results of word segmentation

Algorithm	Recall	Precision	F_1	Recall$_{known}$	Recall$_{unknown}$
Baseline	$65.80(\frac{46780}{71097})$	$66.56(\frac{46780}{70284})$	66.18	$68.19(\frac{39127}{57381})$	$50.32(\frac{4732}{9403})$
Standard Search	$83.47(\frac{59345}{71097})$	$78.24(\frac{59345}{75851})$	80.77	$88.29(\frac{50663}{57381})$	$58.76(\frac{5525}{9403})$
Two-Pass$_{\epsilon=0.5}$	$\mathbf{85.15}(\frac{60541}{71097})$	$\mathbf{82.87}(\frac{60541}{73051})$	**84.00**	$\mathbf{89.35}(\frac{51272}{57381})$	$58.78(\frac{5527}{9403})$
Two-Pass$_{\epsilon=0.6}$	$84.69(\frac{60213}{71097})$	$82.72(\frac{60213}{72795})$	83.69	$88.79(\frac{50949}{57381})$	$58.83(\frac{5532}{9403})$
Two-Pass$_{\epsilon=0.7}$	$84.08(\frac{59777}{71097})$	$82.54(\frac{59777}{72418})$	83.30	$88.00(\frac{50493}{57381})$	$59.13(\frac{5560}{9403})$
Two-Pass$_{\epsilon=0.8}$	$82.84(\frac{58894}{71097})$	$82.03(\frac{58894}{71799})$	82.43	$86.47(\frac{49616}{57381})$	$\mathbf{59.16}(\frac{5563}{9403})$

Table 4. Results of word segmentation and POS tagging

Algorithm	Recall	Precision	F_1	Recall$_{known}$	Recall$_{unknown}$
Baseline	$57.58(\frac{40940}{71097})$	$58.25(\frac{40940}{70284})$	57.91	$61.19(\frac{35109}{57381})$	$31.28(\frac{2941}{9403})$
Standard Search	$77.57(\frac{55149}{71097})$	$72.71(\frac{55149}{75851})$	75.06	$82.46(\frac{47314}{57381})$	$50.14(\frac{4715}{9403})$
Two-Pass$_{\epsilon=0.5}$	$\mathbf{79.05}(\frac{56202}{71097})$	$\mathbf{76.94}(\frac{56202}{73051})$	**77.98**	$\mathbf{83.20}(\frac{47741}{57381})$	$50.51(\frac{4749}{9403})$
Two-Pass$_{\epsilon=0.6}$	$78.63(\frac{55901}{71097})$	$76.79(\frac{55901}{72795})$	77.70	$82.66(\frac{47434}{57381})$	$50.66(\frac{4764}{9403})$
Two-Pass$_{\epsilon=0.7}$	$78.08(\frac{55515}{71097})$	$76.66(\frac{55515}{72418})$	77.36	$81.97(\frac{47033}{57381})$	$50.90(\frac{4786}{9403})$
Two-Pass$_{\epsilon=0.8}$	$76.98(\frac{54729}{71097})$	$76.23(\frac{54729}{71799})$	76.60	$80.60(\frac{46251}{57381})$	$\mathbf{50.95}(\frac{4791}{9403})$

ϵ is used for identifying suspicious nodes. The baseline method cannot produce acceptable results. These results indicate how difficult the morphological analysis in ORCHID corpus is. The standard search which performs a single-pass Viterbi search can boost the performance with a reasonable gap. The best performance (except Recall$_{unknown}$) can be obtained by using the two-pass search algorithm with $\epsilon = 0.5$. The two-pass search algorithm improves the performance of the standard search by 3.23 F_1 in word segmentation and 2.92 F_1 in the combination of word segmentation and POS tagging. The two-pass search algorithm yields the best Recall$_{unknown}$ with $\epsilon = 0.8$. The more the value of ϵ is set, the more the suspicious nodes are identified. The new word hypotheses are subsequently generated, which in turn increase noises. In practice, the optimal value of ϵ can be selected by cross-validation.

In Table 5, we consider the effect of ‘prefix & suffix’ and ‘character type’ features in morphological analysis. We use the two-pass search algorithm with $\epsilon = 0.5$ and perform experiments without each feature. By removing ‘prefix & suffix’ feature, the performance decreases obviously. However, the contribution of character type feature is not significant as we expected. Without it, the performance improves slightly.

4.3 Error Analysis

From overall results, we observe that there are two major cases, which the two-pass search algorithm could not analyze the input strings correctly. The first

Table 5. Results without each feature produced by Two-Pass$_{\epsilon=0.5}$

Feature $\mathbb{F}$	Level	Recall	Precision	F_1	Recall$_{known}$	Recall$_{unknown}$
$\mathbb{F}\backslash\{$prefix & suffix$\}$	word	83.49	81.77	82.62	88.44	58.32
		(-1.66)	(-1.10)	(-1.38)	(-0.91)	(-0.46)
	word & POS	77.55	75.95	76.74	82.48	50.10
		(-1.50)	(-0.99)	(-1.24)	(-0.72)	(-0.41)
$\mathbb{F}\backslash\{$character type$\}$	word	85.41	82.82	84.10	89.73	58.69
		(+0.26)	(-0.05)	(+0.10)	(+0.38)	(-0.09)
	word & POS	79.40	76.99	78.18	83.66	50.60
		(+0.35)	(+0.05)	(+0.2)	(+0.46)	(+0.09)

case is when unknown words are mixed with various character types. Since ORCHID corpus was derived from technical papers, the number of technical terms and jargons is relatively high. For example, the surface 'Gao.65AI035AS' is segmented into smaller tokens, 'Gao', '.', '65', 'AI', '035' and 'AS', according to their character types. In this case, it is not difficult to resolve, because we can use some regular expressions to detect domain-specific word surfaces. The second case which makes Thai morphological analysis much more difficult is when unknown words are formed from known words, and each of them is otherwise found independently in the corpus. This case can be thought of as the problem of compound words. To deal with the problem, we need to find a set of features that can capture the compounding process.

5 Related Work

A wide variety of statistical corpus-based approaches has been applied to Thai morphological analysis. Jaruskulchai [3] used a model selection technique called minimum description length to estimate how likely a sequence of syllables can form a word. Aroonmanakun [1] exploited statistics of collocation to merge syllables to a word. These approaches are only focused on word segmentation, and hard to integrate POS tagging into their frameworks due to the limitation of problem formulation. Charoenpornsawat et at. [2] applied the Winnow algorithm to learn contextual features for handling the unknown word problem. Recently, Kruengkrai et al. [6] have proposed a unified framework for Thai morphological analysis based on conditional random fields (CRFs). Their approach uses a linear-chain formulation and produces a lattice using n-best word/tag sequences, while our approach uses a general graphical model (i.e., an instance of Markov random fields) and directly generates a lattice from a string input.

Discriminative models have been applied to morphological analysis in other non-segmenting languages. Uchimoto et al. [10] proposed a Japanese morpheme model that estimates probabilities of every substring for a given sentence based on maximum entropy. Kudo et al. [7] proposed a CRF-based training method that can avoid the label and length bias problems in Japanese morphological analysis. Our approach can also avoid those problems due to performing whole-sequence normalization.

6 Conclusion

We have presented a discriminative learning approach for Thai morphological analysis. We consider Thai morphological analysis as a search problem. We propose the two-pass search algorithm that finds the most likely path in the expanded search space. The objective of our algorithm is to increase the coverage of word hypotheses based on probability estimation in the lattice. The experimental results on ORCHID corpus show that the two-pass search algorithm can improve the performance over the standard search approach.

References

1. Aroonmanakun, W.: Collocation and Thai Word Segmentation. Proc. of the 5th SNLP & 5th Oriental COCOSDA Workshop. (2002) 68–75
2. Charoenpornsawat, P.: Feature-based Thai Word Segmentation. Master's Thesis, Computer Engineering, Chulalongkorn University. (1999)
3. Jaruskulchai, C.: An Automatic Thai Lexical Acquisition from Text. Proc. of PRICAI. (1998) 289–296
4. Jurafsky, D., Martin, J. H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Prentice-Hall, Inc. (2000)
5. Kruengkrai, C., Charoenporn, T., Sornlertlamvanich, V., Isahara, H.: Acquiring Selectional Preferences in a Thai Lexical Database. Proc. of IJCNLP. (2004)
6. Kruengkrai, C., Sornlertlamvanich, V., Isahara, H.: A Conditional Random Field Framework for Thai Morphological Analysis. Proceedings of the Fifth International Conference on Language Resources and Evaluation. (2006)
7. Kudo, T., Yamamoto, K., and Matsumoto, Y.: Applying Conditional Random Fields to Japanese Morphological Analysis. Proc. of EMNLP. (2004)
8. Liu, D. C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Math. Programming. (1989) 45:503–528
9. Sornlertlamvanich, V., Charoenporn, T., Isahara, H.: ORCHID: Thai Part-Of-Speech Tagged Corpus. Technical Report TR-NECTEC-1997-001, NECTEC. (1997)
10. Uchimoto, K., Nobata, C., Yamada, A., Sekine, S., Isahara, H.: Morphological Analysis of a Large Spontaneous Speech Corpus in Japanese. Proc. of ACL. (2003)

Exploiting Unlabeled Text with Different Unsupervised Segmentation Criteria for Chinese Word Segmentation

Hai Zhao and Chunyu Kit

Department of Chinese, Translation and Linguistics,
City University of Hong Kong,
83 Tat Chee Avenue, Kowloon, Hong Kong, China
{haizhao,ctckit}@cityu.edu.hk

Abstract. This paper presents a novel approach to improve Chinese word segmentation (CWS) that attempts to utilize unlabeled data such as training and test data without annotation for further enhancement of the state-of-the-art performance of supervised learning. The lexical information plays the role of information transformation from unlabeled text to supervised learning model. Four types of unsupervised segmentation criteria are used for word candidate extraction and the corresponding word likelihood computation. The information output by unsupervised segmentation criteria as features therefore is integrated into supervised learning model to strengthen the learning for the matching subsequence. The effectiveness of the proposed method is verified in data sets from the latest international CWS evaluation. Our experimental results show that character-based conditional random fields framework can effectively make use of such information from unlabeled data for performance enhancement on top of the best existing results.

1 Introduction

The task of Chinese word segmentation (CWS) is to segment an input sequence of characters into a sequence of words. It is also a preprocessing task shared by many Asian languages without overt word delimiters. CWS was first formulated as a character tagging problem in [1], via labeling each character's position in a word. For example, the segmentation for following sentences,

他 / 来自 / 墨西哥 。
(he / comes from / Mexico.),

receives the tag (label) sequence $SBEBME$ as segmentation result, where the four tags B, M and E stand for the beginning, middle and ending positions in a word, and S for a single character as a word. A Maximum Entropy (MaxEnt) model was trained for such a tagging task in [1]. Many supervised learning methods have been successfully applied to CWS since the First International Chinese Word Segmentation Bakeoff in 2003 [2]. Among them, the character tagging is a particularly simple but effective formulation of the problem suitable for various competitive supervised machine learning models such as MaxEnt, conditional random fields (CRFs), and support vector machines. [1, 3–8].

Received 07/10/07
Accepted 07/12/07
Final Version 17/01/08

However, few existing methods make use of non-local information of any given sequences as a source of knowledge. In this paper, we will explore a new approach to integrating such useful information from the unlabeled text into the supervised learning for CWS. It attempts to utilize lexicon information derived by various word likelihood criteria, which were intended for unsupervised word segmentation techniques. We know that an unsupervised segmentation strategy has to follow some predefined criterion about how likely a target substring, as a word candidate, is to be a true word. It is important to examine how such information which usually appears as a goodness score for a word candidate can be exploited to facilitate a supervised learning mode for CWS. In this study, we will examine four kinds of such criteria, frequency of substring after reduction, description length gain, accessor variety, and boundary entropy. All of them will be represented as features for integration into our character tagging system for CWS, and their effectiveness will be evaluated using the large-scale data sets for the previous Bakeoff.

The remainder of the paper is organized as follows. The next section describes the baseline supervised learning with the CRFs model for CWS. Section 3 discusses four criteria for unsupervised word extraction and formulates our approach to integrating them to the CRFs learning for CWS. Then, our experimental results are presented in Section 4. Section 5 discusses related work and the possibilities of semi-supervised learning with CRFs. Finally, we summarize our research achievements to conclude the paper in Section 6.

2 Supervised Learning for Word Segmentation

CRFs [9] is a statistical sequence modeling framework that is reported to outperform other popular learning models including MaxEnt method in a number of natural language processing (NLP) applications[10]. CRFs is first applied to CWS in [3], treating CWS as a binary decision task to determine whether a Chinese character in the input is the beginning of a word.

The probability assigned to a label sequence for an unsegmented sequence of characters by a CRFs is given by the equation below:

$$P_\lambda(y|s) = \frac{1}{Z}\exp(\sum_{c\in C}\sum_{k}\lambda_k f_k(y_c, y_{c-1}, s, c)),$$

where y is the label sequence for the sentence, s is the sequence of unsegmented characters, Z is a normalization term, f_k is a feature function and λ_k is the respective weight, C is the label(or tag) set, and c indexes into characters in the sequence to be labeled. For CRFs learning, we use the CRF++ package with necessary modification for training speedup [1].

It is shown in our previous work that the CRFs learning achieves a better segmentation performance with a 6-tag set than any other tag set [11]. Thus, we opt for using this tag set and its six n-gram feature templates as the baseline for our evaluation. The six tags are B, B_2, B_3, M, E and S. Accordingly, we have the tag sequences S,

[1] http://crfpp.sourceforge.net/

BE, BB_2E, BB_2B_3E, BB_2B_3ME and $BB_2B_3M\cdots ME$ for characters in a word of length 1, 2, 3, $\cdots$, and 6 (and above), respectively. The six n-gram feature templates are C_{-1}, C_0, C_1, $C_{-1}C_0$, C_0C_1 and $C_{-1}C_1$, where $0, -1$ and 1 stand for the positions of the current, previous and next characters, respectively.

3 Unsupervised Segmentation Criteria

In general, unsupervised segmentation assumes no pre-segmented data for training and no pre-defined lexicon. It has to follow some predefined criterion to identify word candidates and assign to them a goodness score to indicate their word likelihood. In this study, we explore the effectiveness of utilizing four existing unsupervised segmentation criteria to facilitate supervised learning for CWS. Each of them is applied to compute a goodness score $g(s)$ for an n-gram substring s in the input text. In principle, the higher the goodness score for a substring, the more likely it is to be a true word. We consider all available substrings in the input as possible word candidates for each criterion.

3.1 Frequency of Substring after Reduction

Frequency is not a reliable estimator for how likely a substring is to be a word, although we feel like that a more frequent substring seems to have a better chance to be a true word. Statistical substring reduction [12] is perhaps a workable idea to turn the frequency into a good word-hood criterion. Its underlying assumption is that if two overlapping substrings have the same frequency, then the shorter one can be discarded as a word candidate. To integrate such frequency information after substring reduction (FSR) into our CRFs learning, we define a goodness score as follows,

$$g_{FSR}(s) = \log(p(s)), \tag{1}$$

where $p(s)$ is the frequency of s. That is, we take the logarithm value of the frequency as the goodness score for s. Such a score is the number of bits needed to encode s in a sense of information theory, if the base for the logarithm is 2.

3.2 Description Length Gain

It is proposed in [13], as a goodness measure for a compression-based method for unsupervised word segmentation. The DLG from extracting all occurrences of a substring $s = x_i x_{i+1} ... x_j$ (also denoted as $x_{i..j}$) as a candidate word from a corpus $X = x_1 x_2 ... x_n$ (with a vocabulary V, here, a character list) is defined as

$$DLG(x_{i..j}) = L(X) - L(X[r \rightarrow x_{i..j}] \oplus x_{i..j})$$

where $X[r \rightarrow x_{i..j}]$ represents the resultant corpus from replacing all instances of $x_{i..j}$ with a trace symbol r throughout X and $\oplus$ denotes string concatenation. $L(\cdot)$ is the empirical description length of a corpus in bits that can be estimated as below, following classic information theory [14, 15].

$$L(X) \doteq -|X| \sum_{x \in V} p(x) \log_2 p(x)$$

where $|\cdot|$ denotes the length of a string in number of characters. To effectively integrate DLG into our CRFs model, we define $g_{DLG}(s) = \log(DLG(s))$.

3.3 Accessor Variety

This criterion is formulated in [16]. It has a nice performance in extraction of low-frequent words as reported in [16]. As a measure to evaluate how independent a subsequence is and hence how likely it is a true word, the accessor variety of a substring s is defined as

$$AV(s) = \min\{L_{av}(s), R_{av}(s)\} \tag{2}$$

where the left and right accessor variety $L_{av}(s)$ and $R_{av}(s)$ are defined, respectively, as the number of the distinct predecessor and successor characters. For the similar reason as in Section 3.1, the goodness score $g_{AV}(s)$ for s is set to the logarithm value of AV, $\log(AV(s))$.

3.4 Boundary Entropy

The branching entropy or boundary entropy (BE) is formulated as a criterion for unsupervised segmentation in a number of previous works [17–20]. The local entropy for a given substring $s = x_{i..j}$,

$$h(x_{i..j}) = -\sum_{x \in V} p(x|x_{i..j}) \log p(x|x_{i..j}), \tag{3}$$

indicates the average uncertainty next to $x_{i..j}$. Two scores, namely, $h_L(x_{i..j})$ and $h_R(x_{i..j})$, can be defined for the two directions to extend $x_{i..j}$. Also, let $h_{min} = \min\{h_R, h_L\}$ in a similar manner for AV in equation (2), and then we define $g_{BE}(s) = \log(h_{min})$.

The two criteria AV and BE share a similar assumption as in the pioneering work [21]: If the uncertainty of successive tokens increases, then the location is likely to be at a boundary. In this sense, they are various formulation for a similar idea.

3.5 Feature Templates to Incorporate Unsupervised Segmentation Criteria

The basic idea of exploiting information derived by different unsupervised segmentation criteria is to inform a supervised learner of how likely a substring is to be a true word according to a particular criterion.

To make best of such information, suitable feature templates need to be used to represent word candidates with different lengths. According to [11], less than 1% words are longer than 6-character in segmented corpora of Chinese. Thus, we consider only n-gram of no more than five-character long for feature generation in this work.

We use CRFs as an ensemble model to integrate these features. For each unsupervised segmentation criterion, we consider two types of features. One is concerned with word matching for an n-gram s, which is formulated as a feature function,

$$f_n(s) = \begin{cases} 1, & \text{if } s \in L \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

to indicate whether s belongs to the word candidate list L. Heuristic rules are applied in [16] to remove substrings that consist of a word and adhesive characters for AV criterion. In this study, we do not use any heuristic rules. For each criterion, we only set a default threshold, namely, 0, to get the corresponding list: $g_{FSR}(s) > 0$, $g_{AV}(s) > 0$, $h_{min} > 0$, and $DLG(s) > 0$. The other is concerned with word likelihood information. A feature template for an n-gram string s with a score $g(s)$ is formulated as,

$$f_n(s, g(s)) = \begin{cases} t, \text{ if } t \leq g(w) < t + 1 \\ 0, \text{ otherwise}, \end{cases} \tag{5}$$

where t is an integer to discretize the word likelihood score. For an overlap character of several word candidates, we choose the one with the greatest goodness score to activate the above feature functions for that character. This makes the feature representation robust enough to cope with many infrequent candidates. In this way, feature values will not be sensitive to the threshold about word candidate list generation. Feature function (5) can be actually generated from all possible substrings occurring in the given text. Note that all t in (5) are not parameters but as feature values in the system. Our system is basically parameter-free.

4 Evaluation

Our approach is evaluated in all four corpora from the Third International Chinese Language Processing Bakeoff (Bakeoff-3) [2] [22]. , where corpus size information can be found in Table 1. Word segmentation performance is measured by F-measure, $F = 2RP/(R + P)$, where the recall R and precision P are respectively the proportions of the correctly segmented words to all words in the gold-standard segmentation and a segmenter's output [3]. The recall of out-of-vocabulary words (OOVs), R_{OOV}, is also given to measure the effectiveness of OOV identification.

Table 1. Corpus size of Bakeoff-3 in number of words

Corpus	AS[a]	CityU[b]	CTB[c]	MSRA[d]
Training (M)	5.45	1.64	0.5	1.26
Test (K)	91	220	154	100

[a] Academia Sinica Corpus.
[b] City University of Hong Kong Corpus.
[c] Corpus by University of Pennsylvania and University of Colorado
[d] Microsoft Research Asia Corpus.

[2] http://www.sighan.org/bakeoff2006.
[3] A standard scoring tool is available at http://www.sighan.org/bakeoff2003/score.

4.1 Performance Comparison with Different Criteria

We take the system described in Section 2 as baseline for comparison. Features generated by unsupervised segmentation criteria according to formulae (4) and (5) are derived from the unannotated training and test text. They are integrated the baseline system for evaluation. Our evaluation results are in Table 2 and 3[4].

Table 2. Performance comparison: features derived by different criteria

Criterion	F-score				R_{OOV}			
	AS	CityU	CTB	MSRA	AS	CityU	CTB	MSRA
Baseline	0.9539	0.9691	0.9321	0.9609	0.6699	0.7815	0.7095	**0.6658**
$\text{AV}_{(1)}$[a]	0.9566	0.9721	0.9373	0.9630	0.6847	0.7997	0.7326	0.6584
$\text{AV}_{(2)}$[b]	0.9573	0.9740	**0.9428**	**0.9634**	0.6842	0.8002	**0.7581**	0.6523
$\text{BE}_{(1)}$	0.9566	0.9721	0.9373	0.9630	0.6847	0.7997	0.7326	0.6584
$\text{BE}_{(2)}$	**0.9584**	**0.9743**	0.9421	0.9633	**0.6930**	**0.8029**	0.7569	0.6493
$\text{FSR}_{(1)}$	0.9565	0.9715	0.9367	0.9621	0.6782	0.7931	0.7299	0.6628
$\text{FSR}_{(2)}$	0.9575	0.9735	0.9415	0.9630	0.6775	0.7994	0.7557	0.6420
$\text{DLG}_{(1)}$	0.9554	0.9708	0.9395	0.9616	0.6738	0.7883	0.7459	0.6514
$\text{DLG}_{(2)}$	0.9560	0.9718	0.9401	0.9617	0.6793	0.7970	0.7528	0.6531

[a] (1): using feature formula (4) for all criteria
[b] (2): using feature formula (5) for all criteria

From Table 2, we can see that every unsupervised segmentation criteria do lead to performance improvement upone the baseline. It is also shown that all criteria give further performance improvement upon the cases without word likelihood information included in features, though the improvement is slight in many cases. Among those criteria, it can be observed that AV and BE give the most competitive performance while DLG the least. Such difference is due to supervised learning model and how unsupervised segmentation information is integrated. Although our results show AV and BE are the best criteria to improve supervised learning for CWS, this does not necessarily mean that they are better unsupervised segmentation criteria than DLG for unsupervised segmentation when working alone. Actually, DLG gives better results for unsupervised segmentation as reported in [24]. It is also worth noting that AV and BE results in all evaluation corpora just do as they for unsupervised segmentation tasks [24], revealing the intrinsic similarity of these two criteria.

Table 3 integrating all features from two or more criteria does not necessarily lead to any further improvement. This suggests that though each criterion does give its useful insight to word boundary in Chinese text, their characteristics overlap very much.

[4] The results here are slightly different from those in [23] for the same experimental settings, the only cause to the difference, as we can see, it is the different CRFs implementation this time.

Table 3. Performance improvement with features derived from various unsupervised criterion combinations using feature formula (5)

Criteria				F-score				R_{OOV}			
AV	BE	FSR	DLG	AS	CityU	CTB	MSRA	AS	CityU	CTB	MSRA
				0.9539	0.9691	0.9321	0.9609	0.6699	0.7815	0.7095	**0.6658**
+				0.9573	0.9740	**0.9428**	**0.9634**	0.6842	0.8002	**0.7581**	0.6523
	+			**0.9584**	**0.9743**	0.9421	0.9633	**0.6930**	**0.8029**	0.7569	0.6493
+	+			0.9570	0.9726	0.9421	**0.9635**	0.6798	0.7990	0.7550	0.6614
+		+		0.9574	**0.9739**	0.9425	0.9633	0.6860	0.7975	0.7596	**0.6637**
+			+	0.9569	0.9733	0.9425	0.9629	0.6829	0.7995	0.7587	0.6417
	+	+		**0.9575**	0.9725	0.9423	0.9631	0.6842	0.7996	0.7581	0.6464
	+		+	0.9570	0.9734	**0.9428**	0.9627	0.6860	0.7979	**0.7638**	0.6452
		+	+	0.9573	0.9732	0.9416	0.9632	0.6769	0.8017	0.7541	0.6514
+	+	+	+	**0.9575**	0.9729	0.9413	0.9630	**0.6878**	**0.8039**	0.7535	0.6361

4.2 Comparison against Existing Results

We compare our performance with those best ones in closed test track of Bakeoff. The rule for the closed test is that no additional information beyond training corpus is allowed, while open test of Bakeoff is without such a constraint.

A summary of the best results in the closed test of Bakeoff-3 are presented in Table 4 for a comparison with ours. Our results are obtained by integrating BE features into the baseline system. All six participants with at least a third best performance in the closed test of Bakeoff-3 are given in this table [25, 7, 26, 27, 6, 8].

Table 4. Comparisons of the best existing results and ours in data of Bakeoff-3 (F-scores)

Participant	(Site ID)	AS	CityU	CTB	MSRA
Zhu	(1)	0.944	0.968	0.927	0.956
Carpenter	(9)	0.943	0.961	0.907	0.957
Tsai	(15)	0.957	**0.972**	-	0.955
Zhao	(20)	**0.958**	0.971	**0.933**	-
Zhang	(26)	0.949	0.965	0.926	0.957
Wang	(32)	0.953	0.970	0.930	**0.963**
Best of Bakeoff-3		0.958	0.972	0.933	0.963
Ours		0.958	0.974	0.942	0.963
Error Reduction (%)		-	7.1	13.4	-

From Table 4, we see that our system demonstrates a significant improvement upon the baseline and achieves a better performance on top of the state-of-the-art as in

Bakeoff-3. Especially, our results are achieved only with n-gram information alone, while some official results of Bakeoff-3 were involved in features or techniques that are only allowed in open test [26, 6, 8][5].

To check if those results with slight difference are statistical significant, we perform some statistical significance tests in the results of closed test. Following the previous work [2] and assuming the binomial distribution is appropriate for our experiments, we may compute 95% confidence interval as $\pm 2\sqrt{p'(1-p')/n}$ according to the Central Limit Theorem for Bernoulli trials [28], where n is the number of trials (words). We suppose that the recall represents the probability of correct word identification, and the precision represents the probability that a character string that has been identified as a word is really a word. Thus two types of intervals, C_r and C_p, can be computed, respectively, as p' is set to r and p. One can determine if two results are significantly different at a 95% confidence level by checking whether their confidence intervals overlap. The values of C_r and C_p for the best existing results and ours are in Table 5, where the data of each row with head 'bakeoff-3' are from [22].

Table 5. Statistical significance: comparisons of the best closed results of Bakeoff-3 and ours

Corpus	#word	Best	R	C_r	P	C_p	F
AS	91K	Bakeoff-3	0.961	±0.001280	0.955	±0.001371	0.958
		Ours	0.964	±0.001235	0.953	±0.001403	0.958
CityU	220K	Bakeoff-3	0.973	±0.000691	0.972	±0.000703	0.972
		Ours	0.974	±0.000679	0.974	±0.000679	0.974
CTB	154K	Bakeoff-3	0.940	±0.001207	0.926	±0.001330	0.933
		Ours	0.947	±0.001142	0.937	±0.001238	0.943
MSRA	100K	Bakeoff-3	0.964	±0.001176	0.961	±0.001222	0.963
		Ours	0.960	±0.001239	0.967	±0.001130	0.963

4.3 Early Results in Open Test

Until now, we only consider using the plain text from training and test. Some external segmented corpora are used to strengthen the current segmentation task in [4] and [6]. However, it is well known that segmented corpus is not always easily obtained. Thus it will be meaningful to extend our approach to external unlabeled text, which can be easily obtained as any requirement. Here we report some early segmentation results on using such external resources.

The unlabeled text that we adopt is that of People's Daily[6] from 1993 to 1997 of about 100M characters. The evaluation corpora are CTB and MSRA of Bakeoff-3 that

[5] Although manifestly prohibited by the closed test rules of Bakeoffs, character type features are used in [6] and [8], and a key parameter is estimated by using an external segmented corpus in [26].

[6] This is the most popular official newspaper in mainland of China.

are also in GB encode. AV is selected as the criterion for the goodness score computation. The results are given in Table 6.

Table 6. Performances using external unlabeled data or lexicon

Corpus	Metric	Text[a]	Dict[b]	Text+Dict	Best open
CTB	F-score	0.9401	0.9412	0.9443	0.944
	R_{OOV}	0.7382	0.7412	0.7565	0.768
MSRA	F-score	0.9674	0.9681	0.9716	0.979
	R_{OOV}	0.6905	0.6905	0.7140	0.839

[a] Using AV features from People's Daily text.
[b] Using features from external dictionary.

Note that the way that we extract useful information from unlabeled data is to make use of a word candidate list. It is also a natural way to integrate an external dictionary. The results of using the same online dictionary from Peking University and feature representation as [4] are also given in Table 6. There are about 108,000 words of length one to four characters in this dictionary[7].

We obtain two competitive results compared to the best existing results only by using unlabeled data and an external lexicon, while two best official results in Bakeoff-3 were obtained through large scale external segmented corpora, lexicons and named entity information [29, 30]. This shows that our approach is also effective to exploit external unlabeled data.

5 Discussion and Related Work

In this study, we explore a combination of fully supervised and unsupervised learning for Chinese word segmentation. It is not sure at the first glance whether unsupervised segmentation in the same supervised data can help supervised learning. However, if unsupervised technique can extract global information of the whole text instead from local context inside a sequence, then we can expect the effectiveness, since each type of unsupervised segmentation criterion makes global statistics through the whole text.

When we are applying unlabeled data to improve supervised learning, semi-supervised method is actually introduced into this field. Since Researchers developed techniques for structural semi-supervised learning scheme for large scale unlabeled data in linguistics. As a sequence labeling tool, CRFs with revision for semi-supervised learning has been developed recently.

Semi-supervised CRFs based on a minimum entropy regularizer was proposed in [31]. Its parameters are estimated to maximize the likelihood of labeled data and the negative conditional entropy of the unlabeled data.

[7] It is available from http://ccl.pku.edu.cn/doubtfire/Course/Chinese%20Information %20Processing/Source_Code/Chapter_8/Lexicon_full_2000.zip

In [32], a semi-supervised learning approach was proposed based on a hybrid generative and discriminative approach. Defining the objective function of a hybrid model in log-linear form, discriminative structured predictor (i.e., CRFs) and generative model(s) that incorporate unlabeled data are integrated. Then, the generative model attached unlabeled data is used to increase the sum of the discriminant functions during the parameter estimation.

The idea in our work is close to that of [32]. However, considering that supervised learning for CWS is often a large scale task in computation and lexical information is traditionally used for information transformation. Unsupervised word extraction methods are directly adopted to output lexical information for discriminant model. Our method has been shown efficient and effective in this way.

6 Conclusion

In this paper, we have presented an ensemble learning approach to take advantage of unlabeled data for Chinese word segmentation.

The lexical information plays the central role in information transformation from unlabeled data to supervised learning model. Four types of unsupervised segmentation methods are considered and formulated as word candidate extraction and the respective goodness score computation. Such information about outputs of unsupervised word extraction is integrated as features into CRFs learning model. The effectiveness of different unsupervised criteria for word extraction is studied. We provide evidence to show that character-based CRFs modeling for CWS can take advantage of unlabeled data, especially, the unlabeled text of training corpus and test corpus, effectively, and accordingly achieve a performance better than the best records in the past, according to our experimental results with the latest Bakeoff data sets.

Acknowledgements

The research described in this paper was supported by the Research Grants Council of Hong Kong S.A.R., China, through the CERG grant 9040861 (CityU 1318/03H) and by City University of Hong Kong through the Strategic Research Grant 7002037.

References

1. Xue, N.: Chinese word segmentation as character tagging. Computational Linguistics and Chinese Language Processing **8** (2003) 29–48
2. Sproat, R., Emerson, T.: The first international Chinese word segmentation bakeoff. In: The Second SIGHAN Workshop on Chinese Language Processing, Sapporo, Japan (2003) 133–143
3. Peng, F., Feng, F., McCallum, A.: Chinese segmentation and new word detection using conditional random fields. In: COLING 2004, Geneva, Switzerland (2004) 562–568
4. Low, J.K., Ng, H.T., Guo, W.: A maximum entropy approach to Chinese word segmentation. In: Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, Jeju Island, Korea (2005) 161–164

5. Tseng, H., Chang, P., Andrew, G., Jurafsky, D., Manning, C.: A conditional random field word segmenter for SIGHAN bakeoff 2005. In: Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, Jeju Island, Korea (2005) 168–171
6. Zhao, H., Huang, C.N., Li, M.: An improved Chinese word segmentation system with conditional random field. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 162–165
7. Tsai, R.T.H., Hung, H.C., Sung, C.L., Dai, H.J., Hsu, W.L.: On closed task of Chinese word segmentation: An improved CRF model coupled with character clustering and automatically generated template matching. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 108–117
8. Zhu, M.H., Wang, Y.L., Wang, Z.X., Wang, H.Z., Zhu, J.B.: Designing special post-processing rules for SVM-based Chinese word segmentation. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 217–220
9. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2001) 282–289
10. Rosenfeld, B., Feldman, R., Fresko, M.: A systematic cross-comparison of sequence classifiers. In: SDM 2006, Bethesda, Maryland (2006) 563–567
11. Zhao, H., Huang, C.N., Li, M., Lu, B.L.: Effective tag set selection in Chinese word segmentation via conditional random field modeling. In: Proceedings of the 20th Asian Pacific Conference on Language, Information and Computation, Wuhan, China (2006) 87–94
12. Lü, X., Zhang, L., Hu, J.: Statistical substring reduction in linear time. In et al., K.Y.S., ed.: Proceeding of the 1st International Joint Conference on Natural Language Processing (IJCNLP-2004). Volume 3248 of Lecture Notes in Computer Science., Sanya City, Hainan Island, China, Springer (2004) 320–327
13. Kit, C., Wilks, Y.: Unsupervised learning of word boundary with description length gain. In Osborne, M., Sang, E.T.K., eds.: CoNLL-99, Bergen, Norway (1999) 1–6
14. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal **27** (1948) 379–423, 623–656
15. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley and Sons, Inc., New York (1991)
16. Feng, H., Chen, K., Deng, X., Zheng, W.: Accessor variety criteria for Chinese word extraction. Computational Linguistics **30** (2004) 75–93
17. Tung, C.H., Lee, H.J.: Identification of unknown words from corpus. Computational Proceedings of Chinese and Oriental Languages **8** (1994) 131–145
18. Chang, J.S., Su, K.Y.: An unsupervised iterative method for Chinese new lexicon extraction. Computational Linguistics and Chinese Language Processing **2** (1997) 97–148
19. Huang, J.H., Powers, D.: Chinese word segmentation based on contextual entropy. In Ji, D.H., Lua, K.T., eds.: Proceedings of the 17th Asian Pacific Conference on Language, Information and Computation, Sentosa, Singapore, COLIPS Publication (2003) 152–158
20. Jin, Z., Tanaka-Ishii, K.: Unsupervised segmentation of Chinese text by use of branching entropy. In: COLING/ACL 2006, Sidney, Australia (2006) 428–435
21. Harris, Z.S.: Morpheme boundaries within words. In: Papers in Structural and Transformational Linguistics. (1970) 68－77
22. Levow, G.A.: The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 108–117
23. Zhao, H., Kit, C.: Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In: The Sixth SIGHAN Workshop on Chinese Language Processing, Hyderabad, India (2008)

24. Zhao, H., Kit, C.: An empirical comparison of goodness measures for unsupervised Chinese word segmentation with a unified framework. In: The Third International Joint Conference on Natural Language Processing (IJCNLP-2008), Hyderabad, India (2008)
25. Carpenter, B.: Character language models for Chinese word segmentation and named entity recognition. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia, Association for Computational Linguistics (2006) 169–172
26. Wang, X., Lin, X., Yu, D., Tian, H., Wu, X.: Chinese word segmentation with maximum entropy and N-gram language model. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 138–141
27. Zhang, M., Zhou, G.D., Yang, L.P., Ji, D.H.: Chinese word segmentation and named entity recognition based on a context-dependent mutual information independence model. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 154–157
28. Grinstead, C., Snell, J.L.: Introduction to Probability. American Mathematical Society, Providence, RI (1997)
29. Jacobs, A.J., Wong, Y.W.: Maximum entropy word segmentation of Chinese text. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 108–117
30. Liu, W., Li, H., Dong, Y., He, N., Luo, H., Wang, H.: France Telecom R&D Beijing word segmenter for SIGHAN bakeoff 2006. In: Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing, Sydney, Australia (2006) 108–117
31. Jiao, F., Wang, S., Lee, C.H., Greiner, R., Schuurmans, D.: Semi-supervised conditional random fields for improved sequence segmentation and labeling. In: Proc. of COLING/ACL-2006, Sydney, Australia (2006) 209 – 216 –800
32. Suzuki, J., Fujino, A., Isozaki, H.: Semi-supervised structured output learning based on a hybrid generative and discriminative approach. In: Proceedings of the 45rd Annual Meeting of the Association for Computational Linguistics (ACL'07), Prague, Czech, Association for Computational Linguistics (2007) 791

Word Sense Disambiguation

Defining Relative Strengths of Classifiers at Randomly Generated Datasets

Harri M.T. Saarikoski

Department of Translation Studies
Helsinki University, Finland
Harri.Saarikoski@helsinki.fi

Abstract. Classification is notoriously computing-intensive, especially with large datasets. The common (mis)understanding is that cross-validation (CV) tests on subsets of the training data are the only recourse for the selection of best classifier for given dataset. We have shown in [9] that best classifier can be selected on basis of a few prediction factors (related to training volume, number of features and feature value distribution) calculated from the dataset. In this paper, we report promising results of a series of runs with a number of strong classifiers (Support Vector Machine kernels, Decision Tree variants, Naive Bayes and also K Nearest Neighbor Classifiers and Neural Networks) with randomly generated datasets (gensets). We outline the relative strengths of these classifiers in terms of four prediction factors and generalize the findings to different types of datasets (word sense disambiguation, gensets and a protein recognition dataset).

1 Introduction

Classification is the process of automatically resolving the conflicts that occur when two or more different things have the same name and are thereby ambiguous to machines (e.g. the English noun bank can refer among other things to a financial institution or a river embankment). It is a required procedure for machine-learning application fields as varied as data mining (e.g. recognition of the protein type), text mining and natural language processing (NLP). For instance, word sense disambiguation (WSD) is defined as the resolution of ambiguous words into meanings or senses). To resolve these 'classification tasks' automatically, training instances containing independent observations (instances) of ambiguous classes are gathered (or extracted from text in case of WSD). Then from this training data, a training model is learned by classifiers, and the model is tested or evaluated against unseen (i.e. test) instances. Classification accuracy (number of correctly classified test instances / total number of test instances) is the qualitative measure used in this paper for assessing the performance of classifiers.

In the next sections, we describe the investigated datasets, prediction factors and classifier differences. Then we present the summary of results from those classifiers at those datasets and explain the systematic findings in terms of those prediction factors and finally conclude on the importance of the findings.

Received 28/10/07
Accepted 07/12/07
Final Version 18/01/08

2 Dataset Profiles

Dataset elements are target classes (e.g. senses of word), training and test instances, features and feature values. In classification tasks such as weather prediction (e.g. sunny/rain, hail/snow), target class is predicted using either binary (0/1), numerical (0..1) or set-based (0,1,n) features as probabilistic pointers to those target classes.

Table 1. Dataset matrix (fictional values) for 2-grain (2 target classes) task.

Dataset	Instance	Feature 1 (binary)	Feature 2 (nominal)	Feature 3 (numerical)
Class 1	Instance 1	0	A	0.1
	Instance 2	0	A	0.2
	Instance 3	1	A	0.1
Class 2	Instance 4	1	B	0.9
	Instance 5	1	B	0.8
	Instance 6	1	C	0.7
Feature bias[1]		0.890	-0.212	0.506

Datasets from different fields have been used to define the strengths of classifiers:

1. Word sense disambiguation (WSD) datasets. Word sense disambiguation is typically based on a combination of contextual-statistical methods (e.g. n-grams or parts of speech from around target word). 1-grams are single words around target word from the whole target word instance, 2-grams are consecutive words in the whole target word instance, parts of speech are POS tags from the local context around target word (see Listing 1).

Listing 1. Parts of speech feature set (pos3) (= parts of speech with one-word window around and including target word) for *authority.n* word in Senseval-4 in ARFF format from Weka [11].

```
@attribute 'R1-WDT' {0,1}
@attribute 'R1-WP' {0,1}
@attribute 'senseclass' {1, 3, 4, 5}
{3 1, 29 1, 43 1, 54 3} % authority.n 18:0@30@wsj02
{9 1, 22 1, 49 1, 54 1} % authority.n 32:0@0@wsj02
```

First two lines show two features (R2-WDT and R2-WP: words with those pos tags one word right (R) from target word). Third line lists the target classes ('senseclass'). Last lines show two instances where feature values are represented in sparse format (i.e.

[1]Positive/negative feature biases mean the feature points to class 1 / class 2 respectively as estimated by classifiers.

only showing 1's meaning the part of speech tag is present in current instance).

2. Gensets. To compare performance of classifiers at different datasets, we generated random datasets (with both binary features 0/1 or binfeats, and numerical features or numfeats). We varied the following relevant factors:

- number of training instances (train): 40, 60, 80, 100, 150, 200
- number of target classes (grain): 2, 5, 8, 10, 15, 20
- number of input features (nrfeat):5, 10, 25, 50, 100, 200

This constituted a batch of 216 datasets of both numfeat and binfeat type (totalling at 432 datasets). Example of such dataset follows in Listing 2.

Listing 2. Genset-binfeat with five features and two target classes (excerpt). 2

```
@attribute a8 {false,true}
@attribute a9 {false,true}
@attribute class {c0,c1}}
true, false, false, false, true, c0
true, true, false, false, false, c0
```

3. Protein Recognition. Online SCOP resource (http://scop.mrc-lmb.cam.ac.uk/scop/data/scop.b.html) is a protein categorization system with 126 numerical features and over a thousand proteins. Below is an excerpt (Listing 3).

Listing 3. Protein dataset (7 target classes as class grain).

```
@attribute comp_hyd_one real
@attribute comp_hyd_two real
@attribute class {53931, 51349, 48724, 46456, 56572, 56992,
56835}
@data 26.1745,...,5.36913,46456
```

Shown in this excerpt are two features (*comp_hyd_* prefixed), target class (protein) set in terms of protein id's in the SCOP system and one instance belonging to protein class 46456 (most of the 126 feature values largely omitted for reasons of space).

3 Prediction Factors

In this section, we define the case factors that predict the best system for word and define the properties of some classifiers that help define their respective strengths. Prediction factors are sums or averages or distributions of individual instances and features in the training set. In a previous article [9], we have already presented two of them: P (average number of positive training examples per target class) and N (total number of training examples minus P, i.e. number of negative training examples per

[2]Genset-numfeat datasets are just like binfeats but with fvals 0..1 instead of 0,1.

class). Additionally, in recent tests with gensets, we have discovered the following factors relevant:

1. Number of features (nrfeat). While training volume (P+N) and nrfeat already distinguish between some classifier pairs (families even), we still could not explain the difference of some classifiers different weight functions. We found that of all individual factors investigated thus far there exists highest correlation between nfeat and best classifier (0.35 in correlation coefficient). Even P and N remained less correlated but this can be explained by their codependency and interrelation in determining best classifier.
2. Feature quality (fqual) estimation. When we investigated the difference between different decision trees, we realized that they weight low/middle/high-quality features differently (see Classifier differences). Therefore, average fqual of features in the dataset has to make a difference in selecting best classifier. We do not here measure feature quality explicitly but only note the following difference between the datasets: n-grams are typically high in number but relatively low in quality (occurring very few times) while genset features converge to equal quality (due to randomness of fval assignment).
3. Feature value (fval) distribution. This factor characterizes the distribution profile of feature values: while gensets have equal distribution of fvals from 0 to 1 (i.e. with numfeats average fval of each feature converges to 0.5 and with binfeats both 0 and 1 fvals are almost equally probable). N-grams in WSD in turn are focused around value '0' (i.e. there are many features but each of them is not present in most instances). For example, 2-grams have a very high concentration of '0' fvals (features occur rarely) and pos3 the lowest (features occur more frequently). This distribution profile can therefore be characterized as very different from gensets: 'inverse Gaussian' distribution (high-low-high) with a skew toward value '0'. As for protein dataset (SCOP), we noticed that the distribution of approximately 80% of the numerical features enforces Gaussian distribution (low-high-low), i.e. fvals are focused in the middle range.

These two last factors (fqual and fval) will be shown below to be relevant in defining the strengths of different classifiers in terms of their feature / instance (kernel) weight functions respectively. Next we define the relevant differences between the selected classifiers.

4 Classifier Differences

We make the following relevant four divisions:

Difference 1. Weighted dataset element is either instances (containing features) or features (in isolation). We differentiate between feature and instance weighing classifiers:
(a) Feature-based classifiers (Decision Trees or Dtrees as a group name for Dstump, J48, REPTree, CART and PART classifiers from [8] and the well-known NB or Naive Bayes classifier [11]**).** These classifiers operate by estimating feature quality (or feature-class weights), e.g. probabilities that value TRUE in a binfeat dataset predicts class A vs class B. Formula for calculating feature quality however differs from

classifier to classifier. For instance, decision trees usually have two feature weight functions (linear and logarithmic) and corresponding NB formula is also presented 3:

- Information Gain Ratio used in J48, REPTree and PART [8]: `P(feature|class) = P(class) * log2 (P (class|feature))`
- Gini Index used in CART decision tree [3]): `P(feature|class) = P(class) * P(class|feature)`
- Naïve Bayes formula used in NB [11]: `P(feature|class) = P(class) * P(class|feature)`

As we can see, the main difference between CART and Information Gain Ratio based classifiers, for example, is the feature weight function (linear vs logarithmic respectively). Note also that NB is comparable in terms of weight function (linear) to CART classifier (both have essentially the same formula), only differing in the number of features they include in the training model. When we look at the weight function curves (linear and logarithmic), we can categorize their adaptability with features of different fqual profiles as follows: linear weight function weights extremely low and high quality features more highly while logarithmic weight function weights middle-quality features more highly.

(b) Instance-based classifiers (SVM or support vector machine [10], kNN or k nearest neighbor classifier [1] and FNN or feed-forward neural network [6]): Instance-weighing classifiers use feature weight estimations to calculate distances between training instances in projected Euclidean feature space (either original or augmented). The weight function therefore determines how much emphasis is put on class cores vs borders respectively (we call this 'core-border ratio'). We further investigate the most popular ('k' or instance similarity) functions for SVM. The function `K(X,X')` optimizing the class borders differentiate kernel types:

- Linear (SVMlin): `||x·x'||`
- Polynomial (SVMpoly): `(gamma*||x·x'||)^degree)`
- Radial basis function or RBF (SVMrad): `exp(gamma*||x-x'||)^2`

Kernel function notation `K(x,x')` denotes two-class borders (called hyperplanes) X and X' and the optimizing function K itself that seeks to maximize the distance between X and X'. Inside the kernel function, ||x·x'|| denotes the dot product of any two instances projected into Euclidian space [10].

The fundamental difference between the kernel functions can be described as follows: polynomial function (x^2) upweighs the distance between high-difference instances (which by definition are concentrated in the class borders) and downweighs distances of instances with smallest differences (typically those located at class cores). In linear kernel (x), weights increase linearly with distance. When comparing these two kernels, polynomial kernel tends to equalize small differences (which are typical of core instances). Radial kernel does the opposite to polynomial kernel, upweighing small fval differences between instances (class core instances). Notice however that unlike linear and polynomial kernels, RBF does not transform features into nonlinear

[3] I.e. probability that a feature points to a class is probability of class in training data multiplied by logarithm / linear probability that a class points to a feature.

space: instead distances between instances are rather calculated from the sum of raw differences in fvals (`x-x'`)[4].

Difference 2. Original vs augmented feature spaces. We further distinguish classifiers based on whether original input features are used (kNN, NB, Dtree, FNN) or whether feature space is transformed or augmented (SVM). SVM attempts to discover instances on the class border and reweighing the rest of the instances in relation (distance) to them, resulting in augmented feature space where classes can then be linearly separated. FNN in turn seeks to (gradually) place instances that are most related next to each other, attempting to classify instances in original feature space (as opposed to SVM). kNN also operates in original feature space, calculating pure Euclidean distances between instances. In other words, kNN does not transform feature weights like SVM does nor gradually iterate on instance similarity metric like FNN does.

Difference 3. Training model size. can also distinguish between classifiers that include all features in their training model (kNN, NB, FNN) as opposed to those who reduce features in various ways (Dtree, SVM, Dstump). [12] calls these generative (all features) vs discriminative (some features) classifiers respectively. Dstump only picks a single (best) feature and Dtree picks a majority of features (depending on the set amount of pruning). SVM in a way also reduces dimensions since its training model is constructed on the 'support vectors' or the class-border instances in transformed feature space (and hence features occurring in other instances than these are significantly downweighed). NB and Dtree in turn both estimate feature quality using entropy of `P(class|feature)` utilize class priors `P(class)`. There are therefore only two main differences between NB and Dtree: (a) Dtree creates a tree of conditional dependency (AND) rules consisting of several features pointing together to a class while NB uses feature weights in isolation (not expecting dependency between features, hence the name 'naive'). (b) Dtree discards features of poorest quality (under a configurable threshold) while NB uses all features.

Difference 4. Method of model optimization. are two types of model optimizers (or 'tree pruners' when dealing with Dtrees): (a) those that alter feature-class weights based on previous run misclassifications (error-based) and (b) those that generate multiple models and vote between them (vote-based). Additionally there are those that perform no pruning (let us call them 'single run based'). Within the decision tree family under scrutiny here, the corresponding classifiers are REPTree (error-based), CART (vote-based) and J48/PART (single run based). We further present and test two successful meta-learning schemes that implement these types of pruning (a) and (b) (these operate on the output of a selected base classifier):

- **Boosting** [5] is an error-based iteration method. It produces a series of models (set by number of boosting rounds, here 19) trained on different bootstrapped portions of the training data and performs weighted voting between then. More essentially, Boosting upweighs the features in misclassified instances of

[4]The gamma parameter can in this respect be considered a fine-tuning parameter, altering the angle of the function curve only slightly.

each boosting round. This is essentially the same procedure as with REPTree (reduced-error pruning).

- **Bagging** ('Bootstrap Aggregating') [2]. Bagging can be considered as a vote-based method like CART. In training phase, training data is first split into training and test subsets (set by size of split or bag, here 80%). Then, a chosen base classifier is trained on one bag and tested on the remainder. All these submodels then vote on their combined class decision.

5 Test Results [5]

To produce results that are also robust to any test instance, we ran 10-fold cross-validation 10 times on each of the above classifiers (including Boosting and Bagging), resulting in 100 separate test results for each classifier and each dataset. Cross-validation evaluation in Weka [11] also keeps the same target class distribution for the folds as in the whole dataset (= stratified cross-validation) to ensure that the folds are maximally alike and do not thus influence the classification accuracy (nor classifiers' different capability to deal robustly with new test data) dramatically. This is incidentally why we were motivated to omit training-test set divergence as prediction factor (although it was shown in [12] to differentiate classifiers), since 100 tests on all classifiers serves to downplay the role of divergence (as opposed to the 5-fold single-iteration tests done by [12]).

We here report results from tests of selected classifiers on gensets and proteins datasets. We also report a summary of OE tests, i.e. from utilizing prediction factors for the prediction of best classifier. Corresponding OE tests with WSD systems (both custom and Senseval [6]) have already been reported in [9].
(We discuss the significance of these results in next chapter). We can mention here further that OE gross and net gains with gensets were within range of those found for WSD datasets in [9], with net gain between 2-4%. For instance, accuracy for predicting between Dstump and NB was 0.89, which means that out of the potential (gross gain) of +4.1, a net gain of 0.89 * +4.1 = +3.2 was obtained. Prediction accuracies were again well above most frequent class baseline as they were in [9]. Best OE net gains with genset batches were in fact obtained from using at least one decision tree (CART).
A few notes of these results right away: While NB wins SVM at 5-grain, SVM gains over NB both with lower and higher grains. Polynomial kernel of SVM begins to gain steadily over linear kernel after 10-grain (ends up at stable +7% at higher grains). Linear decision tree (CART) using cross-validation based pruning performs better than logarithmic unpruning J48 at any grain as it did with gensets. Some other grains showed similar trends and were therefore omitted.

[5]All classifier runs were made with Weka implementations of these algorithms [11] (Weka is downloadable from http://www.cs.waikato.ac.nz/ml/weka/). Default configurations were used: complexity parameter for SVM was 1.0, confidence parameter for Dtrees was 0.25 (medium amount of tree pruning). Weka's AdaBoost.M1 was used for boosting with 10 iteration rounds and bag size 80% for Weka's Bagging algorithm.

[6]Evaluation framework for WSD systems (see www.senseval.org).

Table 2. Average accuracies of classifiers at two different types of gensets.

gensets-binfeat		**gensets-numfeat**	
Bagged CART	60	Bagged CART	80
Boosted CART	58	Boosted CART	78
CART	58	CART	77
SVMpoly [7]	56	REPTree	76
SVMlin	56	PART	75
NB	55	J48	75
REPTree	54	NB	69
J48	54	Dstump	66
PART	53	SVMpoly	65
Dstump	50	SVMlin	65
SVMrad	49	SVMrad	56

Table 3. Classifier results on different class grains of protein dataset.

grain	**NB**	**CART**	**J48**	**SVMlin**	**SVMpoly**
2	73	61	55	75	75
5	72	62	57	72	66
10	64	58	55	68	49
50	32	42	36	43	49

6 Classifier Strengths

In this section, we look at classifier pairs to see where their strengths lie with regard to three prediction factors [8].

Finding 1. SVM (linear kernel) vs NB form a hemispherical section in PN space. In all datasets involving linear SVM and NB so far investigated (see e.g. [9] for WSD systems, [4] for text classification systems), SVM tends to take C2-C1 corners (low-P) section of PN space over Dtrees and NB, leaving NB (and Dtree) C3-C4. At gensets we

[7]Using degree (exponent) of 2, i.e. Kernel function is thus ||x·x'||^2.

[8]Due to space constraints, prohibiting the inclusion of dozens of images generated using RapidMiner [7], formations (20) will be posted to the following site (www.mysenseval.com) until they are published in my dissertation later this year.

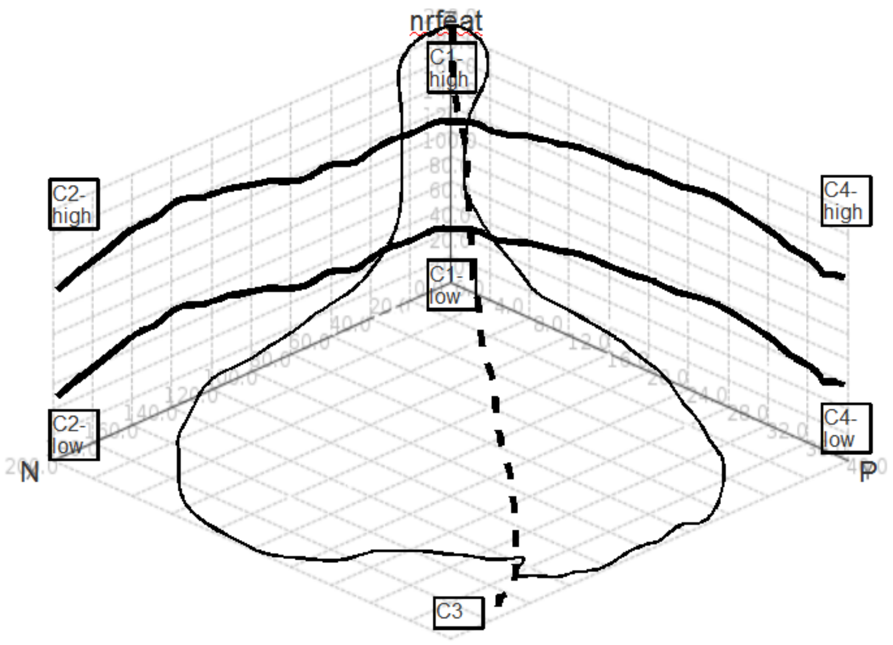


Figure 1. Three-dimensional space for representing classifier formations (= relative strengths) with P, N and nrfeat as dimensions. C stands for 'corner' (e.g. C4 is high-P but low-N, C2 is high-N but low-P) and the region remaining between corners we call 'midfield'. Shown are three types of formations: (a) single pear-shaped formation is called 'yin-yang formation' with one classifier taking the midfield (and the whole of C1 regardless of nrfeat), the other the P and N facets, (b) dotted line in the middle is called 'hemispherical formation' with one classifier taking P facet (C4) and the other N facet (C2), (c) two horizontal lines denote 'layered formation' (or layers), with one classifier taking both low and high nrfeats (regardless of P and N) and the other middle nrfeat range. (Cf. Discussion and [11] for another account on such 1-2-1 phenomena). We refer to these types of formations below when describing classifier performance differences.[9]

found SVM to restrict itself more to C1 (shifting from C2). In [9] with WSD datasets, we showed the border of SVM and NB to be almost linear in PN space.

This hemispherical formation was not as clear with gensets (probably due to the relative weakness of both SVM and NB at gensets), SVM tends to lose at C2 cases more, but it can be seen. There are many separate explanations to this:

- (a) SVM optimizes class-borders while NB makes no explicit difference between cores and borders. The higher the grain, the more borders there are, and so the better this expertise works. SVM is simply better equipped to classify cases where definition of borders are more crucial (i.e. high-grain

[9] We should note that while these are approximations of the actual formations, they are nonetheless clear-cut enough to fall into one of these three main formation types. Also unless specifically mentioned, the formations were similar for both binfeats and numfeats.

cases including C1 where training is very scarce).
- (b) SVM projects features into an augmented feature space with explicit definition of class borders, while NB retains the original feature space and weights features linearly using class-prior and entropy of features per class. Hence, we can consider the border between SVM and NB the border between linear and nonlinear feature space approaches.
- (c) NB operates largely on its class-prior, SVM has no such prior but considers all classes as equally important to classify correctly. When grain is high (as it is at C2), this upweighing of more frequent classes no longer works as effectively while SVM gains in strength with added grain. Hence SVM tends to win at higher grained tasks.

Furthermore, in recent WSD tests (with Senseval-4 English lexical sample dataset), we found the strengths of SVM (linear kernel, c=1.0) and NB different with regard to feature sets (number of word wins are quoted, note that if sum is not 100 rest have been tied):

● 2-grams:	SVM 68, NB 23
● 1-grams:	SVM 38, NB 4 (lot of ties)
● pos3:	SVM 22, NB 35

We can see that SVM performs best when features are numerous but of low individual quality (2-grams) and NB best when features are relatively few but occur frequently (pos3), with both excelling equally at 1-grams (which in terms of both nrfeat and average number of occurrences of features is between 2-grams and pos3). This is an indication that fval distribution does define strength of at least these classifiers: SVM seems to able to perform better when features are very numerous (> 200) but of low occurrence, NB vice versa. This is the most likely reason why SVM did not perform well with gensets (see Table 2) but did perform well with WSD [9].

Finding 2. Dtrees, NB and Dstump form layers. We find clear-cut formation between these three classifiers per nrfeat factor. While Dstump is compatible with highest nrfeat, NB in the middle and Dtree (J48 in this case) takes the lowest nrfeat. We explain this as follows: Dtree operates on class-prior like NB, it additionally cuts (prunes) features of lowest quality, which would be essential in drawing class-borders. For this reason, Dtree cannot build a full-blown valid tree if nrfeat exceeds a certain limit (nrfeat=50) (we can call this 'tree building feasibility requirement') since at high-nrfeat, it becomes increasingly more difficult to find valid feature-class pairings (not to mention codependencies between them) (e.g. “IF rule 1 AND rule 2, THEN select class A”). With lower nrfeats, finding such interrelations between features is much easier (and the classification task itself is much easier), since then on average each feature is defined (validated) by many more training instances and thus the chance for an erroneous tree leaf propagating throughout the tree branch is significantly lower. Dstump in contrast is free of that tree-building feasibility requirement since it builds only a tree stump. Hence it works the better the more nrfeat is provided. As for why NB takes middle nrfeat layer between Dtree and Dstump, we found that this is the nrfeat range where both Dtree's and Dstump's strength degrades, i.e. is not particularly strong due to the abovementioned tree-building requirement. We need also to consider the fact that NB keeps all the features including those with lowest feature quality or fqual (does not reduce dimensions like Dtree and Dstump), and so we can conclude that low-quality features are in a significant role at tasks with mid-nrfeat tasks (than at low/high-nrfeat tasks). This is why NB excels at such tasks in its own merit.

Finding 3. Most decision tree pairs make yin-yang formation. This formation resulted from all four decision tree variants when paired up. As to which classifier takes higher nrfeat position in the yin-yang formation seems to depend on their ranking order of average accuracies per dataset (i.e. CART > REPTree > J48, see results in Table 2). This formation type at least is therefore determined by the relative strength of the classifier (i.e. stronger classifier takes P and N facets at high-nrfeat and weaker one the C1 corner and low-nrfeat midfield). This formation results also when ensembling a strong decision tree like CART with SVM. As for defining the strength of CART (best overall classifier at gensets), we can infer that the datasets where linear decision tree (CART) performs best must have a relative profusion of either low- and high-quality features, since those are upweighed by the linear feature weight function in CART (and respectively mid-quality features must be in slight relative majority where logarithmic classifiers J48, REPTree and PART perform best). Although we have not deployed a quantification of fqual (as mentioned in Prediction factors), we find no other reason than the profusion of mid-quality features as factor that *must* explain why CART as a classifier that upweighs mid-quality features is the superior (decision tree) classifier with gensets.

Finding 4. Boosted vs Bagged decision trees make layers. This is the most clear-cut layer formation investigated in this study: Bagging takes high-nrfeat and Boosting low-nrfeat layer. To find causes for this layeredness, we ran boosters against baggers on two different decision trees with different weight functions (CART and J48 as base classifiers). We also ran error-pruned (equivalent of Boosting) version of J48 and tested two different values of the confidence parameter in J48, responsible for the amount of pruning. Pruned (CART, REPTree) vs unpruned decision trees (J48, PART) mentioned in Finding 3 were also obviously investigated. From these tests, we can confirm that both the amount of pruning and the type of pruning can be used to select best classifier for given dataset: pruning (especially cross-validation type) definitely works better at higher nrfeats. We can motivate this as follows: at high-nrfeat (which are tougher cases to classify anyway), feature-class pairings can only become fully validated, i.e. proper class cores and borders defined using those pairing, when the decision tree is trained multiple times on different subsets. Strength of Boosting in turn can be defined as follows: The primary issue for model optimization at low-nrfeat is determined by the number of outlier (misclassified) instances. Though not reported in the results, we found that the number of misclassified instances increases logistically as nrfeat is increased (i.e. classification accuracy is lower at high nrfeats). That is, at low-nrfeat there are simply fewer outlier instances. When Boosting (features and instances containing them) then upweighs them, they do not disturb the delicate weight balance of features occurring in the already correctly classified instances. Once the case becomes tougher as defined by added nrfeat, Boosting no longer can maintain this balance but gives overweight to misclassified instances (i.e. overfits the training model).

Finding 5. Linear and polynomial SVM kernels form no uniform type of formation in terms of PN+nrfeat factors. Strength of these kernels proved hard to define. Depending on whether gensets were binfeat or numfeat, the formation of the two kernels resembles either layers or hemispheres with some non-convex classifier regions. We explain this with their different instance weight (kernel) functions (see Classifier differences). From looking at the function curves of linear vs polynomial kernel (x vs x^2 as function of instance similarity in augmented Euclidean space), it is obvious that polynomial kernel by downweighing small distances tends to expect more core instances (i.e. core is wider and border is projected farther). In contrast, linear kernel expects rather a smooth transition from one class core to mutual border and over to another class core (i.e. that class cores and borders are of equal 'size'). Since this is their *only* difference, their respective strengths can be said to depend on this expectation.

Therefore, we can conlude that the dataset's core-border ratio (defined above), i.e. how many core vs border instances there are on average per target class, determines the best kernel. With gensets, as said, core-border ratio close to 1 qualifies linear kernel (core is of equal 'size' as borders on average for all classes) and ratios above 1 qualify polynomial kernel (core is larger and needs therefore to fit more instances and the farther we go from that core the less important the instances are estimated to be). In other words, polynomial weight function can be said to perform feature-reduction by way of projecting the far-away instances from the hyperplanes (class border) separating the target classes (cf. [11]).

Core-border ratio is an elusive individual quality of the dataset which could be estimated from fqual factor: high-quality features correlate highly with class core (low-quality features with class borders). However, since fqual measure is out of focus in this paper, the only thing that can be said is the following: gensets that have no pronounced core-border division (gensets) have a core-border ratio of 1 (i.e. there are no real cores vs borders since all fvals converge to equal fval distribution resulting in almost equal fquals for all features). Real-world datasets (such as protein recognition or word sense disambiguation) in contrast tend to have a more clear-cut ('natural') sense of class cores vs borders and a definite division between high and low quality features (e.g. 1-grams *and* and *money* are low- and high-quality features pointing to one of the senses of the English noun *bank*).

One kernel, however, is separable using these three prediction factors: radial kernel (with Gaussian weight function) sets apart from all other kernel types both in terms of formation and (lower) overall accuracy. We have confirmed that it performs best only at C4 cases in all datasets investigated and forms thus hemispherical formation with other kernel types. It seems only as long as class grain is low (2-3) and subsequently class borders are few, it is able to compete with other kernels and can even be superior to other kernel types. In our interpretation, there are two separate reasons for this that can be tracked down to RBF kernel design: (a) RBF makes no nonlinear transformation (border optimization) of feature weights like the other kernel types. (b) Gaussian weight function behind RBF kernel upweighs (counter-intuitively) the smallest differences between instances and downweighs largest ones. This effectively leads to an equalization of differences between all instances, and subsequently cores and borders are diluted. However, as we know [6], Gaussian based weight functions perform well at unsupervised (clustering) tasks where borders are virtually non-existent and emphasis is on finding the cores (of clusters). The nature of classification vs clustering tasks is fundamentally different in the task outset: for clustering the algorithm mainly needs (and is able only) to create class *cores*, while classification algorithm is able to additionally define class *borders* from provided training data. We propose that since with clustering tasks all distances are relatively small, even smaller than with 2-grams, Gaussian style accentuation of small differences is the reason why the Gaussian function has empirically proven the most popular instance similarity function for clustering methods such as SOM and FNN [6].

7 Discussion and Conclusions

In this paper, we have presented probable explanations for the regional formations (relative strengths) of classifiers deriving from the properties of classifiers themselves. Our findings suggest that best classifier prediction is very much possible. Many of the formations and phenomena explained here were left unexplained in [9], (which featured mainly the clearly hemispherical formation of SVM vs NB).

We showed that classifiers respond to varying values of prediction factors (P, N, nrfeat), which furthermore can largely be attributed to their algorithm design (e.g. weight functions, size of training model). For instance, differences in weight functions result in the observed layered or '1-2-1' formation where one classifier outperforms the other at low/high ranges of a factors and the other at mid range. (Cf. definition of Dtree vs NB difference defined in [12]: 1-2-1 formation was shown in terms of P and N factors and 1-2 formation in terms of nrfeat).

To sum up the classifier formations, yin-yang formations result from ensembling decision tree pairs, layers occur when decision trees are ensembled with both NB and Dstump, hemisphere formations are mainly for SVM vs NB. SVM kernels make formations that depend on the core-border ratio of the dataset and cannot thus be well predicted using these three factors. As for prediction factors, the fact that feature-based classifiers are mostly separable by nrfeat (a feature-based factor) and instance-based classifiers (SVMs) are separable primarily by P and N (instance-based factors) is not very surprising after all. Instance-based classifiers kNN and FNN formed no discernible or convex region in PN+nrfeat space in either WSD or genset datasets.

The generalization of these formations to three types help the designers of OE method [9] to gain higher prediction accuracies and classifier developers to perhaps develop better classification algorithms that exploit this new-found knowledge of classifier regionality. Results from separate OE tests with gensets (this paper) and WSD datasets [9] result in standard 2-5% net gain (depending largely on selected best base classifiers which in turn depends on prediction factors). We can state, for example, that SVM and NB do not necessarily form maximal complements for all dataset types although they do for WSD [9] and text classification [4]. This is especially if the datasets are favorable to decision trees (as gensets are), in which case optimal ensembling of at least one decision tree comes into question.

As to why SVM and NB perform best at WSD [9] and Dtrees (CART) at gensets is an issue for further research. However, we can consider the equal fval distribution of gensets as opposed to unequal fval distribution (binary features in WSD and Gaussian distribution in protein datasets) as definitely related to that adaptability. The fact that SVM can compete with gensets-binfeat (which are similar to WSD and protein datasets in their respective fval distributions) is proof of SVM's adaptability to sparse datasets. We can further generalize classifier adaptability to types of datasets: since with real-world datasets there is a marked difference between low- and high-quality features on one hand and core vs border instances on the other, we can define the competence of SVM as deriving from its explicit class border definition. As for decision trees, pruning of either type certainly helps them to higher accuracy, which is seen from the superiority of CART, REPTree Boosting and Bagging over J48 and PART at both gensets and protein dataset. The reason for relative strength of Dtrees over both SVM and NB at gensets can most likely be attributed to the fval distribution or correct fqual estimation, which in our interpretation and that of [12] are the only relevant factors remaining unquantified in this study and [9]. This (dataset-independent definition of classifier strengths) is a subject of further study.

References

1. Aha, D., Kibler, D. Instance-based learning algorithms. Machine Learning. 6:37-66. (1991)
2. Breiman, L. (1996). Bagging Predictors. Machine Learning, 24/2, 123-140.
3. Breiman, L,, Friedman, J., Olshen, R., and Stone, C. (1983). CART: Classification and Regression Trees. Wadsworth, NY.
4. Forman, G., and Cohen, I. Learning from Little: Comparison of Classifiers Given Little Training. In 15th European Conference on Machine Learning and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (2004)
5. Freund, Y. & R.E. Schapire (1996). Experiments with a New Boosting Algorithm, in Proceedings of the Thirteenth International Conference on Machine Learning.
6. Legrand, S., Pulido JGR. A Hybrid Approach to Word Sense Disambiguation: Neural Clustering with Class Labeling. Knowledge Discovery and Ontologies workshop at 15th European Conference on Machine Learning (ECML) (2004).
7. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. and Euler, T. YALE: Rapid Prototyping for Complex Data Mining Tasks. Proceedings of 12th ACM SIGKDD (2006)
8. Quinlan, R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers. (1993)
9. Saarikoski, H., Legrand, S., Gelbukh, A. 2006. Case-Sensitivity of Classifiers for WSD: Complex Systems Disambiguate Tough Words Better. In proceedings of CicLING 2006.
10. 10.Vapnik, V. N. The Nature of Statistical Learning Theory. Springer (1995)
11. 11.Witten, I., Frank, E. Data Mining: Practical Machine Learning Tools and Techniques (Second Edition). Morgan Kaufmann (2005).
12. Yarowsky, D. and Florian, R. Evaluating sense disambiguation across diverse parameter spaces. Journal of Natural Language Engineering, 8(4) (2002) 293-311.

Machine Translation

Features and Categories Design for the English-Russian Transfer Model

Elena Kozerenko
Institute for Informatics Problems of the Russian Academy of Sciences,
44 Corpus 2, Vavilova Str., 119333, Moscow, Russia
elenakozerenko@yahoo.com

Abstract. The paper focuses on the role of features for the implementation of the transfer-based machine translation systems. The semantic content of syntactic structures is established via the contrastive study of the English and Russian language systems and parallel texts analysis. The notion of cognitive transfer is employed which means that a language unit or structure can be singled out for transfer when there exists at least one language unit or structure with a similar meaning in the target language The approach taken is aimed at providing computational tractability and portability of linguistic presentation solutions for various language engineering purposes.

Keywords: Machine translation, syntactic structures, features, categories, cognitive transfer.

1 Introduction

The rapid development of language processing systems within the statistical frameworks has revealed the limitations of purely statistical methods in machine translation projects, and at the same time stimulated the new approaches to linguistic rule systems design making them adjusted to be used together with the modern statistical methods. The rule system described in this paper builds on the awareness of the fact that the meaning of a structure in a source language may shift to another category in the language of translation. This awareness is very important for obtaining reliable statistical information from parallel texts corpora to be further used in statistical machine translation algorithms. Otherwise the existing stochastic methods for language processing bring a lot of excessive inconsistent rules which still require filtering and hand editing.

Generally, major efforts connected with natural language modeling lay emphasis at lexical semantics presentations and less attention is paid to the semantics of structures and establishment of functional similarity of language patterns as a core problem in multilingual systems design. The studies presented in this paper focus on the semantics of language structures, namely, the interaction of categorial and functional meanings for subsequent language engineering design of feature-value structures. The proposed methods of dealing with syntactic synonymy of structures (isofunctionality) and structural (syntactic) polysemy provide an essential linguistic foundation for learning mechanisms.

Received 14/10/07
Accepted 07/12/07
Final Version 15/01/08

An important consideration in our work is that some features of human language appear to be of universal character, for example, every language has nouns and verbs. Even the differences of human languages often have systemic structure [1]. Syntactically languages are most different in the basic word order of verbs, subjects, and objects in declarative clauses. English is an SVO language, while Russian has a comparatively flexible word order. The syntactic distinction is connected with a semantic distinction in the way languages map underlying cognitive structures onto language patterns, which should be envisaged in MT implementations [2]. Besides, there exist syntactic constructions specific of a given language (such as, for example, English constructions with existential "*there*" and "*it*" as formal subjects). Sometimes, a word can be translated by a word of another part of speech in the target language, by a word combination, or even by a clause. The parse aimed at transfer procedures requires a semantic grammar and cannot be efficiently implemented through a combination of monolingual grammars.

This paper focuses on the role of features for the implementation of the transfer-based machine translation systems. The approach taken is aimed at computational tractability and portability of the language presentation solutions for various language engineering purposes. An attempt is made to build a generalization over the feature-value sets of the English and Russian languages to introduce the functional semantic motivation into the categories of language structures. Our theoretical conclusions result from linguistic research of paradigmatic and syntagmatic properties of the languages under study and machine translation developments.

We argue that language engineering presentations should be well-grounded linguistically, hence a detailed and profound theoretical study of language features is indispensable for further language simulation. On the other hand, the computational linguistic models shouldn't be overloaded with detailed rules. We operate with a set of 21 basic features for the Russian language and a set of 18 basic features for the English language. The maximal difference of feature-value structures is observed when comparing the nominal features (a system of cases in the Russian language and the system of cases in the English language). The meanings of "genitivity", "dativity", "instrumentality", etc. have to be represented via non-morhpological means – syntactic configurations, i.e. word order and prepositional phrases. We avoid introducing any abstract categories, unless they naturally "emerge" from the basic features and their configurations. So, we are guided by a "reasonable sufficiency" principle in working out the system of features and values for the Russian and English languages which accords with the minimalist program.

Another dramatic mismatch between the languages is connected with the behavior of the English Nonfinite Verbal Forms, such as Gerunds and Participles. Their double nature directly affects the transfer into the Russian language: we have to choose either verbal or nominal match in the Russian language for the Gerund, and either adjectival or adverbial interpretation for the Participle. Again we have to introduce the syntactic level means, since there are no direct morphological correspondencies in Russian. We construct a presentation system employing the basic morphological features and their combinations to capture the meanings of the higher level units and structures – syntactic and semantic features. Categorial feature structures serve the objective for introducing functional semantics: category is the potential for function (functions).

The presentation mechanism chosen resembles that of HPSG with its attention both to constituency and dependency relations [3]. It is an attempt to build a phrase structure generative system for the Russian language having the objective of multilingual transfer and establishing the synonymy of language structures in a multilingual situation. The set of functional meanings together with their categorial embodiments serves the source of constraints for the unification mechanism in the formal presentation of our grammar. The formalism developed employs feature-based parse and head-feature inheritance for phrase structures. Phrase structures are singled out on the basis of their functional identity in the source and target languages which accords with the approach of [4]. Important evidence of robustness of statistical methods employment in rule-based parse is given in [5].

The parse and transfer procedures accuracy ranges from 34.75 to 73.43 in our model.

2 Language Structures Transferability

To face the problems of language structures transferability for machine translation (MT), it is of great help to consider human translation experience. Translation is a creative and sophisticated human activity, hence, producing automatically a high-quality translation of an arbitrary text from one language to another is a task too far from its complete implementation. However, for simpler tasks, such as acquiring information from the Web, getting acquainted with subject domain information, etc., a rough translation output without post editing can be quite acceptable. One of the domains where MT works best is scientific discourse. Perhaps, it can be accounted for the regularity of syntactic structures which is required by the functional style of scientific prose.

Of the three forms of translation performed by man: written translation, consecutive interpretation and simultaneous interpretation, the one which is nearest to the real-time machine translation is simultaneous interpretation (SI). Therefore, the recommendations for SI are of prime interest to MT designers, as they propose more implementable solutions for lexical grammatical transformations than the first two forms.

The following SI techniques appeared to be of use for MT design in the course of our development.

(1) Full translation of lexical grammatical forms is applied when these forms completely correspond to each other both in the source and the target languages as to their form, function and meaning.

(2) Null translation is applied when a grammatical form exists in the source and target languages but is used differently for explicating a certain referential situation.

(3) Partial translation is used when one and the same grammatical form has several content functions which differ in the source and target languages.

(4) Functional substitution is employed when the functions and meanings of grammatical forms in the source and target languages differ. In that case the source form can be substituted by a form of another type in the target language on the basis of their functional identity.

(5) Conversion is used for substituting a form of one category by a form of another category, and is conditioned by the combinability rules difference in the source and target languages.

Thus it is obvious that the search for equivalence should be carried out starting with the establishment of semantic equivalence of patterns notwithstanding their structural dissimilarity. Pattern-matching approach for the English – Russian transfer was assumed, and the segmentation of structures of the source language was performed on the basis of Cognitive Transfer Fields which were established via contrastive study of the two languages [8].

The segmentation of phrase patterns used for the input language parse was carried out with the consideration of semantics to be reproduced via the target language means.

The absence of full coincidence between the English and Russian language constructions can be found when studying the comparative frequency of the parts of speech use, especially in scientific and technical texts. In general the style of scientific discourse is characterized by a greater rate of “nominativity”, i.e. the use of nouns than the other styles. And the comparative study of translations shows that this tendency is considerably stronger in the Russian language where the verbs of the source English texts are frequently substituted by nouns. Our studies show that the Russian text is approximately by 35% more nominal than the English text. Consider the following examples of verbal-nominal transformations in the English-Russian translations.

These considerations are important for building translation systems employing machine learning methods.

3 Semantic Match Establishing Principles

Studying the categorial and functional meanings of language structures we have established that important tools realizing these meanings are ways of configuring phrase structures, i.e. linearization patterns: possible linear sequences of language objects (of units and structures).

Semiotic linguistics [9] calls these ways of configuring *structural signs* and also introduces the concept of *superposition* of functions, presuming that every language object has its primary function, and shifts of meanings which occur in speech (i.e. language in action) are superposition of secondary and other functions onto the primary one.

Our research is focused on revealing all possible types of structural signs which convey similar meanings, i.e. establishment of syntactic synonymy.

A classical example of syntactic synonymy is the means of expressing case meanings, e.g. morphological in the Russian language (through cases endings) and analytical in English - by means of prepositions and the order of words.

Hence, our problem is to reveal all types of structural signs and compose them into a uniform system of semantic syntactical representations for a language processor.

Superposition of functions is a very useful tool for construction of functional semantic representations of linguistic units and structures.

The concepts of primary and secondary functions of language signs enable us to create the new representations using traditional categories (as, for example, the Verbal_Noun = Verb + Noun).

The method applied for creating a system of rules for functional transfer in machine translation is described in [10]. The establishment of structures equivalence on the basis of functional semantics proved to be useful for developing the syntactic parse and transfer rules module for the English – Russian machine translation [8]. Generally, major efforts connected with natural language modeling lay emphasis at lexical semantics presentations and less attention is paid to the semantics of structures and establishment of functional similarity of language patterns as a core problem in multilingual systems design.

Our interpretation techniques employ the segmentation of structures carried out on the basis of the functional transfer principle. The principal criterion for including a language structure into a field is the possibility to convey the same functional meaning by another structure of the field, i.e. the interchangeability of language structures. To establish whether the structures and units are equal or not, we need some general equivalent against which the language phenomena would be matched. In Contrastive Linguistics the notion of *tertium comparationis* is widely employed to denote this general equivalent, and the approach based on the principle “from the meaning to the form” focusing on Functional Syntax would yield the necessary basis for equivalence search.

What differs our approach is the attention to the semantics of configurations, i.e. the study of the way languages tend to *arrange structures* in order to convey certain meanings. And we focus on the linear patterns of the languages under study, since we assume that linearization is not a random process but it is determined by the cognitive mechanisms of speech production and the way they manifest themselves in syntactic potentials of a given language. The primary object of our contrastive language study was to establish what particular language meanings are represented in the categorial-functional systems of the English and Russian languages. Categorial values embody the syntactic potentials of language units, i.e. their predictable behavior as syntactic structures (syntaxemes). Thus, as it was demonstrated in [9-11], Category is the potential for Function, and multiple categorial values inflict multiple syntactic functions. However, when we analyze language in action, i.e. the utterances of written or sounding speech, the function of a particular language structure determines which of the potentials is implemented in this utterance, hence which of the possible categorial values of a polysemous syntactic structure is to be assigned to the parse result.

4 Cognitive Transfer Structures

Our observation is that the cognitive linguistic process of transfer goes across the functional – categorial values of language units. A language structure which can be subjected to transfer has to be semantically complete from the point of view of its function. The cases of categorial shifts, in particular, when the technique of conversion is employed, require special treatment: the categorial shift of a syntax unit is determined by the functional role of this unit in a sentence (e.g. *noun as a modifier→adjective*). Only by creating the centaur concepts.. 'constituency-dependency', 'linearity-nonlinearity', 'form-function', etc. can we get a reasonably clear picture of linguistic reality [9].

The starting idea for the language structures segmentation strategy was the notion of functional semantic fields [7] in the Russian language. The system of grammar units, classes and categories with generalized content supplementary to the content of lexical units, together with the rules of their functioning, is a system which in the end serves for transmission of generalized categories and structures of mental content which lay the foundation of utterance sense, and constitute the basis of language grammar formation.

The transferability of phrase structures is conditioned by the choice of language units in the source and target languages belonging to the same functionally motivated Cognitive Transfer Fields (CTF), notwithstanding the difference or coincidence of their traditional categorial values. A set of basic CTF was singled out and language patterns employed for conveying the functional meanings of interest were examined.

Primary Predication CTF (non-inverted) bearing the Tense – Aspect – Voice features; this field mainly includes all possible complexes of finite verbal forms and tensed verbal phrase structures.

Secondary Predication CTF bearing the features of verbal modifiers for the Primary Predication CTF. Included here are the non-finite verbal forms and constructions and subordinate clauses comprising the finite verbal forms. All these are united by the functional meanings they convey, e.g. qualification, circumstance, taxis (ordering of actions), etc.

Nomination and Relativity CTF: language structures performing the nominative functions (including the sentential units) comprise this field.

Modality and Mood CTF: language means expressing modality, subjunctivity and conditionality are included here. Here the transfer goes across the regular grammatical forms and lexical means (modal verbs and word combinations) including phrasal units.

Connectivity CTF: included here are lexical – syntactic means employed for concatenation of similar syntactic groups and subordination of syntactic structures.

Attributiveness CTF: adjectives and adjectival phrases in all possible forms and degrees comprise the semantic backbone of this field; included here are also other nominal modifiers, such as nominative language units and structures (stone wall constructions, prepositional genitives – of –phrases), and other dispersed language means which are isofunctional to the backbone units.

Metrics and Parameters CTF: this field comprises language means for presenting entities in terms of parameters and values, measures, numerical information.

Partition CTF: included in this field are language units and phrase structures conveying partition and quantification (e.g. some of, part of, each of, etc.).

Orientation CTF: this field comprises language means for rendering the meaning of space orientation (both static, and dynamic).

Determination CTF: a very specific field which comprises the units and structures that perform the function of determiner (e.g. the Article, which is a good example for grammar – lexical transfer from English into Russian, since in Russian there exist no such grammatical category; demonstrative pronouns, etc.).

Existentiality CTF: language means based on be-group constructions and synonymous structures (e.g. sentential units with existential there and it as a subject: there is…; there exists…; etc.).

Negation CTF: lexical – syntactic structures conveying negation (e.g. nowhere to be seen, etc.).

Reflexivity CTF: this field is of specific character since the transfer of reflexivity meaning goes across lexical - syntactic – morphological levels.

Emphasis – Interrogation CTF: language means comprising this field are grouped together since they employ grammar inversion in English.

Dispersion CTF: individual language structures specific for a given language are included here; these are presented as phrasal templates which include constant and variable elements. To implement the feature-valued inheritance sometimes broader contexts are taken.

A constraint-based formalism of Cognitive Transfer Grammar (CTG) was developed and implemented in the English-Russian machine translation system [8]. It comprised 222 transferable phrase structures together with the transfer rules combined within the same pattern. The formalism provides representation mechanisms for the fine-grained information about number and person, agreement, subcategorization, as well as semantics for syntactic representations. The system of rules based on this formalism consists of transferable phrase structures together with the transfer rules which are combined within the same pattern. Such patterns, or Cognitive Transfer Structures (CTS), are constitutional components of the declarative syntactical processor module of the machine translation system and encode both linear precedence and dependency relations within phrase structures.

The initial variant syntax of a CTS was designed as follows:

CTS → CTS<identifier> CTS<token> <Input Phrase Structure & Feature-Value Set> <Head-Driven Transfer Scheme> <Generation Feature-Value Set & Phrase Structure >

The Cognitive Transfer Grammar provides translation of phrase structures within one CTS. This was the approach employed in the first version of the English-Russian machine translation system which provided one variant of translation to a CTS. At present a Multivariarnt Cognitive Transfer Grammar (MCTG) has been designed which envisages several translations for each CTS. It comprises about 350 transferable phrase structures together with the multivariant transfer rules.

Consider, for example, the functional meaning of Possessiveness, which belongs to the CTF of Attributiveness in the following phrases:

Peter's house; the house of Peter

These phrases have the same meaning, that could be presented by the following semantic network:

Owner ← Having → Had Thing.

However, we see our main objective not in creation of an abstract semantic meta language, but in the careful research of all possible kinds of configurations of language categories, used by natural languages for expression of functional meanings.

To determine the categorial and functional values of language structures we employ the technique of Compositional Categories. This technique consists in the superposition of categorial values of language objects to envisage the possibility of multivariant translations. Thus the Gerund is categorized as "VerbNounIng", the

Infinitive in the Subject function is categorized as “toPlusInfinitiveSubj”, the Russian Adverbial Participle (“Deeprichastie”) and the the English Participle Active in its adverbial function are categirized as “ParticipleAdv”, finite verb forms are referred to as “VerbFinit”, other categiries are also used, as for example, the following:

{[Category: VerbNounIng]: *asking questions*};

{[Category: toPlusInfinitiveSubj]: *She is known to be a skilled typist* };

{[Category: toPlusInfinitiveObj]: *We feel them to be sensitive readers*}.

5 Polysemy and Ambiguity of Syntactic Structures

By syntactic polysemy we mean the immediate realization of more than one categorial meaning within the head element of a language structure. The polysemous structures display variable manifestation of their categorial features depending on the functional role in the sentence. Consider such language phenomena as the Gerund, the Participle and the Infinitive.

The Gerund comprises the features of both the Verb and the Noun, which affects the translation strategy when the appropriate means are to be chosen for representation of the English Gerund via the Russian language forms. The structures similar in category to the English Gerund are the Russian Verbal Nouns denoting “Activity”, e.g. *singing* → *penie, reading* → *chtenie*, and both the English Gerund, and the Russian Verbal Noun allow direct object arguments if derived from transitive verbs. However, the direct transfer of the Gerund into the Russian Verbal Noun is the least probable translation variant of the three possible transfer schemes:

The Gerund (Eng) → Clause with the Finite Verb form (Rus)

The Gerund (Eng) → Clause with the Infinitive (Rus)

The Gerund (Eng) → Verbal Noun (Rus).

This fact can be accounted for by the mechanisms employed in the Russian language for configuring sentential structures and is to be envisaged in the machine translation engine.

Consider the other most productive polysemous language structures which comprise more than one categorial meaning:

The Participle → Verb + Adjective

The Infinitive → Verb + Noun

Nominal Phrase as the Nominal Modifier → Noun + Adjective

Verbal Phrase as the Verbal Modifier → Verb + Adverb.

Thus we introduce the notion “polysemous syntactic structure” to determine the set of possible transfer schemes for a given language structure. When a polysemous structure is assigned specific categorial attributes realized in this structure, the possible and preferable transfer schemes become predictable for the given structure.

The predominant categorial meaning of a polysemous syntactic structure (or syntaxeme) is determined by the syntactic function realized at a given moment. Thus the transfer scheme for a “*stone wall*” construction will be as follows:

Noun1 + Noun2 [Eng.] → Adjective + Noun2 [Rus]

The weight for this transformation will be higher than for the transformation:

Noun1 + Noun2 [Eng] → Noun2 + Noun1 (Genitive) [Rus] if the dictionary contains an Adjective as one of the possible translation equivalents for Noun1, that is the case when the dictionary is composed by various methods including acquisition of lexical units from parallel texts.

Judging by the function we establish the Cognitive Transfer Fields (CTF) within which the translation procedure will be carried out CTF support the possible paraphrasing variants and envisage the synonymous ways of conveying the same functional meaning across languages.

Of special interest is the situation of the categorial shift in translating a syntactic pattern. The category of a syntactic pattern, i.e. a phrase structure, is determined by the category of the head word of this phrase structure. Thus, when transfer employs conversion, and the category of the head word shifts to another category, the whole structure is assigned the feature of the new category. Thus a Nominal modifier of a Nominal Phrase becomes an Adjective in translation; a Verbal unit acting as a Verbal modifier becomes an Adverbial clause containing the Finite Verbal form. The latter case accords with the SUG principle of the Verb being the Sentence Nucleus [9,11].

To illustrate the mechanism of polysemous structures transfer we take the Secondary Predication CTF and the Attributiveness CTF.

The Secondary Predication CTF bearing the features of verbal modifiers for the Primary Predication structures (the non-inverted Finite Verb forms and tensed verbal phrase structures bearing the Tense – Aspect – Voice features) includes the nonfinite verbal forms and constructions, and subordinate clauses comprising the finite verbal forms. All these are united by the functional meanings they convey, e.g. qualification, circumstance, taxis (ordering of actions), etc.

The following schemes of transfer into Russian are applicable to the phrase:

Feeling surprised seemed permanent.

"Gerund + Participle II + Finite Verbal Phrase" → " Sentence " →

"Nominative Clause + Finite Verbal Phrase" (1)

OR

"Verbal Noun Phrase + Finite Verbal Phrase" (2)

The Participle in postposition to a Nominal Phrase most frequently would be transferred into a Russian Clause:

The material processed satisfied all the requirements.

"Nominal Phrase + Participle II + Finite Verbal Phrase" → " Sentence " →

"Nominal Phrase + Qualifying Clause + Finite Verbal Phrase" (1)

OR

"Nominal Phrase + Participle II + Finite Verbal Phrase" (2)

Attributiveness CTF: adjectives and adjectival phrases in all possible forms and degrees comprise the semantic backbone of this field; included here are also other nominal modifiers, such as nominal language units and structures (*stone wall* constructions, prepositional genitives – *of* –phrases), and other dispersed language means which are isofunctional to the backbone units.

Consider the phrases of the kind: "*a woman of means*", "*a man of talent*". Possible contexts might be as follows:

She was a woman of means.

He was a man of talent.

The multivariant transfer would comprise the following Russian phrase structures:

(1) with the Genitive construction;

(2) with the Qualifying Clause;
(3) with the Preposition "*s*" (Russian):
"Nominal Phrase1 + of +Nominal Phrase2" →
"Nominal Phrase2 + Nominal Phrase1 Genitive"
Or
"Nominal Phrase1 + Qualifying Clause"
Or
"Nominal Phrase1 + Prep "*s*" + Nominal Phrase2 Instrumental".

The last variant would mean in Russian "*a woman with means*", "*a man with talent*".

We took into account the computational cost of the rule system which led us to a certain minimalism: we avoided introduction of abstract categories in rule design (having in mind the imperative known as Ockham's Razor: the notion that when presented with a choice of axioms or laws, or explanations, it is wise to choose the one that is the *simplest*). All the functional meanings were presented as feature – value structures based on traditional language categories.

We find it important to differentiate between polysemous and ambiguous syntactic structures. A polysemous structure implies possible realizations of meanings which are compatible within one language structure and can be transferred to the structures of another language which are isofunctional to the source language structure. An ambiguous syntactic structure presupposes alternative ways of interpretation, the meanings being incompatible within one language structure, thus we deal with ambiguity when we try to discern some Finite and Nonfinite verbal forms:

Gerund / Present Participle;
Infinitive / Present Simple;
Past Participle / Past Simple.

Ambiguous structures can be misleading to the parsing procedures and subsequent machine translation, as for example, the "garden path" is a well-known language phenomenon which may give incorrect parse at the early stage of analysis, that could be corrected only at the final stage:

The cars passed by the vessel drowned.

The possible interpretations for the sentence can be as follows:

The cars which were passed via the vessel drowned (the correct variant).
The cars which passed the vessel drowned.

However, the phrase"*The new control system updated continuously displayed robust performance*" was analyzed and translated correctly by all the tested modern MT systems which comprise learning mechanisms within their framework. This fact can be explained by the presence of the broader context.

6 Disambiguation Techniques: Rule-Based and Machine Learning Methods

The impact of differentiation between syntactic polysemy versus syntactic ambiguity consists in the following implementation decisions. An ambiguous structure is analyzed in alternative manner: each possible parse and transfer variant is presented as a separate rule, and constraints are introduced into the rule structure. A polysemous structure is assigned a multiple transfer scheme within one rule.

The mechanism of computational (contextual) reframing (CR) is being designed for treatment of the two major bottlenecks: syntactic derivation history (for words in a secondary, tertiary, etc. syntactic function) and syntactic polysemy of structures. Reframing models the use of the same structural unit in different structural and/or lexical contexts, which results in the difference of the meanings of this unit. The presentations for the syntactic module rest on the basis of traditional word categories. Contextual correlations associated with each function of a structural unit are established via stochastic data obtained from corpora study.

Since parse procedures sometimes may result in more than one possible structure, the rules and lexical entries are supplied with the probabilistic augmentations which serve for syntactic ambiguity resolution. The multivariant rules that envisage the variants of transfer for polysemous structures and separate alternative rules for ambiguous structures have been worked out. They comprise the most probable methods of the language structures transfer, as for example, the Infinitive constructions in the function of the Adverbial Modifier of Goal/Consequence:

Hydrogen and oxygen unite to form water.

The scheme of the multivariant English-Russian transfer of the construction *to form water* will be as follows:

[Category: VerbInf] → {*to form water* }

OR {[Category:ParticipleAdv]; {*образуя воду – forming water*}

[Category: VerbFinit]; {*образуют воду – form water*}

[Category: VerbNounIng]} {*с образованием воды – with formation of water*}

In the course of sentence analysis and parent nodes formation the resulting structures will be marked-up by the compositional categories which provide the appropriate transformations of language structures for transfer.

As natural language generates an infinite number of sequences, learning mechanisms are incorporated into the parse engine: information about unfamiliar words and structures can be inferred from the context. The data on which the inference can be founded is accumulated by learning on parallel texts: a supervised algorithm is trained on a set of correct answers to the learning data, so that the induced model may result in more accurate decisions.

The lexical model employs the concise lexicon entries presenting categorial, morphological and combinatorial information supplied with the statistical data for each lexical item characterizing its distribution.

We studied the existing results in the field of human cognitive mechanisms of language learning, as well as machine learning methods: there is substantial evidence that the way children learn their first language may be understood as information compression [12,13]; the Optimality theory states the importance of grammatical architecture with the strict prioritization or ranking, rather than any scheme of numerical weighting.

Of particular interest for us was the study and comparison of various formal approaches [3,5,14-28], so that practical algorithmic solutions could be worked out, we adhere the strict lexicalism principle of the HPSG [3], i.e. word structure and phrase structure are governed by independent principles.

It is a well-known fact that the underlying tree representation plays a crucial role for the performance of a probabilistic model of grammar [14]. Probabilistic developments of the unification grammar formalism are given in [15,17,18]. A new probabilistic model for Combinatory Categorial Grammar is presented in [16].

The phenomenon of syntactic polysemy determines the possible multiple transfer scheme for a given language pattern. We develop the system of multivariant transfer rules - Multivariant Cognitive Transfer Grammar (MCTG).

The probability values for syntactic analysis variants can be obtained either on the basis of corpus information, or from linguistic expert knowledge. In the latter case we deal with reliable information fixed in grammar systems of languages distilled by the centuries of human language practice.

The values of probabilities for every possible parse variant (i.e. the expansion of a nonterminal node) are calculated on the basis of frequencies of occurrence of each analysis variant in the existing text corpora with syntactic mark-up (treebanks). The calculation is made of the number of times (N), when some variant of expansion of a node ($\alpha \to \beta$) is used with subsequent normalization:

$$P(\alpha \to \beta \mid \alpha) = \frac{N(\alpha \to \beta)}{\sum_{\gamma} N(\alpha \to \beta)} = \frac{N(\alpha \to \beta)}{N(\alpha)} \qquad (0.1)$$

The probability of the full parse of a sentence is calculated with the account of categorial information for each head vertex of every node. Let n be a syntactic category of some node n, and h (n) is the head vertex of the node n, m (n) is a mother node for the node n, hence, we will calculate the probability p(r(n)|n, h(n)), for this we transform the expression (1.2) in such a way, that every rule becomes conditioned by its head vertex:

$$P(T,S) = \prod_{n \in T} p(r(n) \mid n, h(n)) \times p(h(n) \mid n, h(m(n))) \qquad (0.2)$$

Since the ambiguity of some syntactic structure (a node) is understood as an opportunity of realization of more than one categorial value in the head vertex of this structure, the probability of the full parse of a sentence containing ambiguous structures (i.e. nodes, subtrees) will be calculated with the account of the probabilities of the categorial values realized in the head vertices of these structures (nodes).

In our grammar system the functional values of languages structures are determined by the categorial values of head vertices. The probabilities are introduced into the rules of the unification grammar CTG as the weights, assigned to parse trees. Ambiguous and polysemous syntactic structures are modeled by the Multivariant Cognitive Transfer Grammar Structures (MCTGS).

The syntax of a MCTG structure (MCTGS)can be presented in the following way:

MCTGS → MCTGS<identifier> MCTGS<weight> MCTGS<mark-up> <Input Phrase with Feature-Value Structue> <Head-Driven Scheme of Transfer> <Generated Phrase with Feature-Value Structure 1> <weight 1> <Generated Phrase with Feature-Value Structure 2> <weight 2> ...<Generated Phrase with Feature-Value Structure N> <weight N>

The MCTG approach provides the mechanisms for modelling the transfer procedures for polysemous and ambiguous syntactic structures and it can be extended for a greater number of languages.

7 Language Engineering Environment

Our current project INTERTEXT is aimed at creation of systemic presentations of functionally motivated semantic syntactic structures. The above stated methods are being employed for design and development of a language engineering environment comprising the research linguistic knowledge base Phrasenet and the features for multivariant parse and transfer of language structures. It is a linguistic resource with semantic grouping of phrase structure patterns provided with the links to isosemic structures at all language levels for the whole set of languages included into the linguistic base. The categorial systems of a subset of natural languages (English, Russian and some other European languages) and functional roles of language units in a sentence have been explored and the core set of transferable language phrase structures has been established on the basis of generalized cognitive entities manifested in the grammar systems under study. The structures are segmented on the semantic principle of functional transferability, i.e. these structures should be “translatable”.

Our linguistic simulation efforts are aimed at capturing the cross-level synonymy of language means inside a system of one natural language and interlingual semantic configurational matches. This emphasis on the practical human translation experience gives the reliable foundation for statistical studies of parallel text corpora and automated rule extraction in further studies.

Our focus on configurations provides high portability to the language processing software designed under these principles: we can operate with a lexicon which has only standard linguistic information including morphological characteristics, part of speech information and the indication of transitivity for verbs, and no detailed semantic mark-up is required for lexical entries.

The Phrasenet linguistic knowledge base comprises the following components:

- a parallel texts database: the texts are segmented into the functionally relevant structures that are semantically aligned; - a structural parse editor (under development at present) which displays the parse and transfer schemes for indicated text segments; - a multilingual functional treebank; - a functional semantic vocabulary of structural configurations arranged on the basis of the Cognitive Transfer principle.

Special Attention is given to the phenomena of syntactic polysemy and ambiguity.

8 Conclusions

Our analysis and development experiments result in understanding the efficiency of what might be called the “exteriorization of meaning”, i.e. accumulation of relevant data concerning the functional-categorial features of possible structural contexts and/or specific lexical contexts that help to disambiguate the parsed structures of the source language and decide what particular meaning of a language structure is realized in the given text segment. Rather than invent a sophisticated antropocentric heuristics for the rule-based disambiguation techniques via traditional linguistic presentations, we need to design a synthetic mechanism comprising the core rule set and reliable learning methods.

The rule set applicable both for the transfer procedures and for acquiring new linguistic data by corpora study should envisage the phenomena of syntactic polysemy and ambiguity of structures. The solution employed in our project is based on the Cognitive Transfer Structures (CTS) approach grouping isofunctional language structures, and the Multivariant Cognitive Transfer Grammar (MCTG) comprising the rules which state the multiple equivalent structures in the source and target languages. The MCTG linguistic rule set is being augmented by Probabilistic Functional Tree Substitution features. Since the nodes of the MCTG have functional articulation, the trees and subtrees of the possible parses also have functional character, i.e. are tagged by functional values.

Our further research and development efforts are connected with the refinement of the existing presentations, inclusion of specific lexical-based rules into the grammar system, and excessive corpora-based experiments for extending the mechanisms of multiple transfer. Our attention at present is directed at the improvement of statistical mechanisms incorporated in the system of parse and transfer rules for the English-Russian language pair and designing a model which includes other languages (Italian and French).

References

1. Comrie, B. Language Universals and Linguistic Typology. Basil Blackwell, Oxford. Second edition. 1989.
2. Nirenburg, S., Carbonell, J., Tomita, M., and Goodman, K. Machine Translation: A Knowledge-based Approach. Morgan Kaufmann. 1992.
3. Pollard, C. and Sag, I.A. Head-Driven Phrase Structure Grammar. University of Chicago Press, Chicago, 1994.
4. Mustajoki, A. Functional Syntax as the Basis for Contrastive Languae Study. Studia Slavica Finlandesia. Finnish Contributions to the 13th International Congress of Slavists, Ljubljana, August, 15-21, 2003, pp.100-127 (In Russian).
5. Briscoe, E.J. “An introduction to tag sequence grammars and the RASP system parser”. Technical Report N662, University of Cambridge, March, 2006.
6. Shaumyan, S. A Semiotic Theory of Language. Indiana University Press, 1987.
7. Bondarko A.V. Printsipy Funktsional’noi Grammatiki I Voprosy Aspektologhii. Moskwa, URSS, 2001 /Functional Grammar Principles and Aspectology Questions. Moscow, URSS, 2001 (In Russian).
8. Kozerenko, E.B. Cognitive Approach to Language Structure Segmentation for Machine Translation Algorithms // Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications, June, 23-26, 2003, Las Vegas, USA.// CSREA Press, pp. 49-55, 2003.
9. Shaumyan, S. Categorial Grammar and Semiotic Universal Grammar. In Proceedings of The International Conference on Artificial Intelligence, IC-AI’03, Las Vegas, Nevada, CSREA Press, 2003.

10. Kozerenko, E.B., Shaumyan, S. Discourse Projections of Semiotic Universal Grammar // Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications, June, 27-30, 2005, Las Vegas, USA.// CSREA Press, pp. 3-9, 2005.
11. Shaumyan, S. Intrinsic Anomalies in Meaning and Sound and Their Implications for Linguistic Theory and Techniques of Linguistic Analysis. // Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications, June, 27-30, 2005, Las Vegas, USA.// CSREA Press, pp. 10-17, 2005.
12. Chater, N. Reconciling simplicity and likelihood principles in perceptual organization Psychological Review 103 (3), pp. 566-581, 1996.
13. Chater, N. The search for simplicity: a fundamental cognitive principle? The Quarterly Journal of Experimental Psychology. 52 (2), 273-302, 1999.
14. Johnson, M. PCFG Models of Linguistic Tree Representations. Computational Linguistics, 24(4), pp. 613-632, 1998.
15. Osborne, M. and Briscoe, T. Learning Stochastic Categorial Grammars. In T.M. Ellison, editor, Proceedings of CoNLL97: Computational Natural Language Learning, pp. 80-87, Somerset, NJ, 1998.
16. Hockenmaier, Julia. Data and Models for Statistical Parsing with Combinatory Categorial Grammar. Ph.D. thesis, University of Edinburgh , 2003.
17. Brown, P.F., J. Cocke, S.A. Della Pietra, V.J. Della Pietra, F. Jelinek, J.D. Lafferty, R.L. Mercer & P.S. Roossin. A statistical approach to machine translation. Computational Linguistics 16, pp. 79-85, 1990.
18. Habash, Nizar and Bonnie Dorr. Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation. AMTA-2002. Tiburon, California, USA, 2002.
19. Dorr, Bonnie and Nizar Habash. Interlingua Approximation: A Generation-Heavy Approach. AMTA-2002 Interlingua Reliability Workshop. Tiburon, California, USA, 2002.
20. Apresjan, Jurij, Igor Boguslavsky, Leonid Iomdin, Alexandre Lazursku, Vladimir Sannikov, and Leonid Tsinman. "ETAP-2: the linguistics of a machine translation system".META, 37(1):97–112, 1992.
21. Boguslavsky, I., Chardin, I., Grigorieva, S., Grigoriev, N., Iomdin, L., Kreidlin, L. and Frid, N. "Development of a dependency treebank for Russian and its possible applications in NLP". In Proceedings of the 3rd International Conference on Language Resources and Evaluation, Las Palmas, Gran Canaria, Spain, 2002.
22. Kakkonen T. "Dependency treebanks: methods, annotation schemes and tools". Proceedings of the 15th NODALIDA conference, Joensuu, 94-104, 2005.
23. Copestake, A., Lascarides, A., & Flickinger, D. "An algebra for semantic construction in constraint-based grammars". In Proceedings of the 39th Meeting of the Association for ComputationalLinguistics. Toulouse, France, 2001.
24. Flickinger, D. "On building a more efficient grammar by exploiting types". Natural Language Engineering, 6 (1) (Special Issue on Efficient Processing with HPSG), 15 – 28, 2000.
26. Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. "Building a large annotated corpus of English. The Penn Treebank". Computational Linguistics, 19, 313 – 330, 1993.
26. Oepen, S., Callahan, E., Flickinger, D., & Manning, C. D. "LinGO Redwoods. A rich and dynamic treebank for HPSG". In LREC workshop on parser evaluation. Las Palmas, Spain, 2002.
27 McCarthy, D., and Carroll J. "Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences" Computational Linguistics, vol. 29.4, 639-654, 2003.
28. Watson, R., J. Carroll and E. Briscoe "Efficient extraction of grammatical relations", Proceedings of the 9-th International Workshop on Parsing Technologies (IWPT'05), Vancouver, Ca., 2005.

Translation of the Light Verb Constructions in Japanese-Chinese Machine Translation

Yiou Wang[1] and Takashi Ikeda[2]

[1]Graduate school of Engineering, Gifu University
y_wang@ikd.info.gifu-u.ac.jp
[2]Yanagido 1-1, Gifu, Gifu 501-1193, Japan
ikeda@info.gifu-u.ac.jp

Abstract. We study the light verb constructions in Japanese, the constructions of expressions with light verb "suru". Such factors as numerous semantic variants and the syntactic complexity of corresponding Chinese expressions of LVCs put great difficulties in the Japanese-Chinese translation, so current commercially available Japanese-Chinese translation softwares give bad results for LVCs translation. We systematically analyze and describe the phenomena and propose translation rules for the conversion of Japanese LVCs into Chinese. we conducted manual experiments using 200 sentences to evaluate these rules and achieved a translation accuracy of over 80%. We also introduce jaw, a pattern-based translation engine, which can be applied to any target language. We implemented our translation rules for LVCs in the Chinese version, jaw/Chinese and verified their applicability in real MT system by experiments using 100 sentences. Evaluations both by hand and by the machine indicate that it provides high performance and utility.

1 Introduction

In order to examine light verb constructions we must first consider the nature of light verbs. The term "light verb" first occurs in Jespersen [1] and Cattell [2].A light verb is a verb (usually a frequent verb with a very general meaning) which, when combined with certain complements, loses some of its normal semantics: such usages are said to be *semantically bleached*[3].Light verb constructions (hereafter LVCs) is a multiword expression that combines a light verb with a complement of noun, adjective, preposition or verb etc. [4].Light verbs are a cross-linguistic phenomenon, and are found in languages such as Persian, Japanese, and Chinese [5][6].In English the most usual light verbs are "make" "do" "take" "have" and "give" as in such LVCs as "have a look" "take a rest" "do a play" "give a hug" and "make a telephone call". In Grimshaw and Mester[7], the Japanese verb *suru* 'do' is analyzed as a light verb. Like the English light verbs, *suru* is assumed to be thematically "light" or incomplete for the contribution to the meaning of verb phrases. We found many studies about the Japanese LVCs (e.g., [8][9][10][11]). These studies discuss Japanese LVCs from various viewpoints and also define different scopes for Japanese LVCs. In this paper we use the terminology "LVC" in the broadest sense: A construction is a LVC if the light verb *suru* is involved in. It maybe questionable whether some sort of constructions we consider here can still be called light verb constructions, but we also investigate such sort of constructions for the sake of completeness.

Received 12/10/07
Accepted 07/12/07
Final Version 16/01/08

The linguistic importance and cross linguistic frequency of LVCs is well attested (e.g., [6]). LVCs have a high frequency of occurrence across many diverse languages and have particular properties so that draw special attention within linguistic and computational analysis researchers to explore them in great details (e.g., [3][8][12]). However the study of the LVCs in machine translation field has been, in comparison, largely overlooked. Few researchers have discussed LVCs from the viewpoint of machine translation ,and as yet there have been no studies on machine translation of Japanese LVCs into Chinese.

Although Japanese and Chinese both use Chinese characters, they belong to different language families. Chinese is an isolating language without inflexion and is typologically classified as SVO language, while Japanese is an agglutinative language with a basic SOV sentence order. In the study of Japanese-Chinese machine translation, one of the most difficult problems we have encountered so far is the translation of Japanese LVCs. Because of the high occurrence frequency of LVCs (nearly 30% sentences including LVCs in the investigating materials), the quality of its translation has a great impact on the overall Japanese-Chinese MT system. It is very necessary and important to make full analysis of it. Given the diversity and the complexity of Japanese LVCs, in Japanese-Chinese translation, they can be rendered as various parts of speech and different sentence patterns. However, current commercially available translation soft wares do not have a sufficiently deep analysis of these translation ambiguities and many mistranslations occur in areas such as word selection and word order determination.

In this paper, we study the usage of Japanese LVCs, analyze complicated correspondences with Chinese and discuss the application of this analysis to the MT system. We proposed translation rules for them, made manual experiments to evaluate the validity of the rules and conducted tests by the machine to certify the applicability of the rules in real MT system.

2 The Translation of Japanese LVCs into Chinese

In this section, we investigate the semantic usage of Japanese LVCs. We analyze and describe systematically the phenomena and propose a method for disambiguating the meaning of LVCs in Japanese-Chinese machine translation.

2.1 Diverse usage of LVCs in Japanese,

There are a lot of sentences containing LVCs in Japanese. We conducted a statistical investigation over 154,958 Japanese sentences in Japanese-English parallel corpus [13], extracted the sentences containing LVCs and found 48,753 sentences containing LVC (nearly 31.5%). Among these 48,753 sentences, 38,977 sentences contain LVCs with some particle such *as* "*o*" "*ga*" "*ni*" "*to*" etc. directly before the light verb "*suru"* (type1)*,* while 11,103 sentences contain LVCs in which light verb *"suru"* (type2) is used directly in combination with a noun (so-called verbal noun) in order to convert it to a verb, or is used to verbify other parts of speech, such as adjectives, verbal adjectives and adverbs.

The corresponding Chinese expressions are also diversified: LVCs can be translated into parts of speech such as verbs, adverbs, and conjunctions, as well as causative sentences, passive sentences, and so on. The following sentences are examples of the diverse correspondence between Japanese and Chinese expressions.
[1] Although it is said that the light verb *suru* is of little semantic content in Japanese, it seems clear that light verbs can contribute different elements of their semantics to different LVCs. While translated into Chinese, it needs a variety of Chinese verbs in order to deliver the correct meanings for different LVCs. For instrances:

1. Japanese: Manngo wa don na iro wo site imasuka?
   Chinese: Mangguo shi shenme yanse de? (What color is the mango?)
2. Japanese: Samuke ga suru. Kaze wo hiita kamosirenai.
   Chinese: Ganjue faleng,yexu ganmao le. (I feel cold. I may have caught a cold.)
3. Japanese:Kono honn wa sen en suru.
   Chinese: Zhe ben shu zhi 1000 riyuan. (This book is worth 1000 yen.)

[2] LVCs in Japanese are also more productive and syntactically-flexible expressions, and correspond to not only predicate but also various parts of speech in Chinese.

4. Japanese:Dekakeyou to suru to, okyaku san ga kita.
   Chinese:Zhengyaochumen, kere laile. (adverb)
   (I was about to go out when a visitor arrived.)
5. Japanese:Ano toki hajimeteita to suru to imagoro wa mou owatteitadesyou.
   Chinese:Ruguo nage shihou kaishi, xianzai yijing jieshu le ba. (Conjunction)
   (If it began at that time, it should be finished now.)

[3] It can also correspond to different Chinese sentence patterns.

6. Japanese: Sinamono wo kane ni suru. (To change goods into money.)
   Chinese:Ba dongxi biancheng qian. ([ba] sentence)
7. Japanese:Noyama kun wo watasitati no daihyou ni suru.
   Chinese:Rang yeshanjun zuo women de daibiao. (causative sentence)
   (Let's make Noyama our representative.)

These examples of the diversified correspondence between Japanese and Chinese demonstrate the necessity of developing a method for translating LVCs in Japanese-Chinese machine translation.

2.2 Solutions and Translation Rules for LVCs

Given the diversity and the complexity of Japanese LVCs, we try to find method to propose translation rules of LVCs for Japanese-Chinese MT. We extracted 1000 sentences containing the LVCs as investigating materials from154, 958 Japanese sentences in a Japanese-English parallel corpus [13]. We extracted the LVCs from the investigating materials and manually translated them. Based on the examination of these example sentences, we found the Japanese LVCs are roughly employing the following six structures:

(1) (NP+LV) structure: the combination of nominal or noun phrase (NP) with LV
(2)(VP+LV) structure: the combination of verb phrase (VP) with LV
(3)(VN+LV) structure: the combination of verbal noun (VN) with LV
(4)(ADJ+LV) structure: the combination of adjective (ADJ) with LV

(5)(ADV+LV) structure: the combination of adverb (ADV) with LV
(6)(ONO+LV) structure: the combination of Onomatopoeia (ONO) with LV
Furthermore we make a summary for the semantic classification of LVCs according to their corresponding Chinese expressions. Table 1 shows part of semantic classification of for (NP+LV) structure and (VP+LV) structure.

Table 1. Semantic classification of LVCs (partial)

Structure type	Class ID	Semantic classification	Examples
NP+LV	1	Have a feeling of Experience with the senses (sound, color…)	寒気がする.　自動車の音がする. I feel cold.　The sound of the car is heard.
NP+LV	2	Have (a characteristic)	英子さんは長い足をしている。 Eiko has long legs.
NP+LV	3	Engage [be engaged] in Act [officiate, serve] as	彼は以前校長先生をしていた. He used to be a principal.
NP+LV	4	Change [turn into, convert]	母は端切れをベッドカバーにしてしまった My mother made pieces of cloth into a bedspread.
NP+LV	5	Decide to do Make it a rule	わたしは定食にします I decided to order the table d'hote.
NP+LV	6	The passing of time	あれから2か月して赤ん坊が生まれました。 2 months later, the baby was born.
NP+LV	7	Judging from	標準からするとまだまだ足りない Judging from the standard, it is still insufficient.
NP+LV	8	For	この二,三日は冬にしては暖かすぎる It is too warm on these several days for winter.
VP+LV	9	Try; attempt	歩けもしないうちから走ろうとするな Do not try to run while you cannot walk.
VP+LV	10	Be about　to do; be on the point of doing	出かけようとすると,お客さんがきた I was about to go out and the guest came.
VP+LV	11	If, to be supposed to	あの時始めていたとするともう終わっていたでしょう If it began at that time, it should be finished now
VP+LV	12	Though	あるとしても,ごくわずかだ Although there are some, it is tiny
VP+LV	13	No sense, representing inflectional information	彼らは飲んだり食べたりして楽しく遊んだ。 They had a good time eating and drinking.

As we know that most of the meaning of a LVC comes from the complement of the construction, based on the semantic classification, we dug into each structure and proposed the translation rules suitable for different structures.

[1] The Translation of (NP+LV) Structure:
Since the basic characteristics of (NP+LV) constructions are to a large extent determined by those of the NP themselves, in order to characterize (NP+LV) structure properly, it is necessary to characterize NPs into proper types. We typed the NP by the following two conditions:

①Particles : in Japanese, syntactic functions of phrases are indicated with various particles[8]. The particles involved in the LVCs with (NP+LV) structure are *o*, *ga,ni(to)* and *wa* which we assume to specify noun phrases as accusative(object), nominative(subject), dative(target) and topic respectively.

② type of the nominal:　the semantics attributes of nouns involved in the NP

Taking into account the limitations of the possible Chinese combinations, we develop translation rules (see Table 2, we here express the translation rules in this format just for easily understanding, the real translation rules stored in the system are in the form as shown in the two examples of table 9 in section 4.2) which can be processed by *jaw/Chinese* (Section 3). In the following tables, N1, N2, N3… represent nouns and X represents a word or part of a sentence.

Table 2. Translation rules for (NP+LV) structure (Partial)

ID	class	Particles	Japanese Pattern	Conditions and attributions Attribution of N1	Attribution of N2	Attribution of N3	Translation rules
(1)	1	ga+ suru	N1 ga N2 ga suru	subject	feeling		N1 感觉 N2
(2)	1	ga+ suru	N1 ha N2 ga suru	Noun, pronoun	stimulus		N1 有 N2
(3)	2	wo+suru	N1ga N2 wo suru	subject	shape/color /status		N1 是 N2 的
(4)	3	wo+suru	N1 ga N2 wo suru	Human being	job		N1 当/是 N2
(5)	4	ni+suru	N1 ga N2 wo N3 ni suru	Noun, pronoun	Descendant, offspring	occupation	N1 要把 N2 培养成为 N3
(6)	5	ni+suru	N1 ga VP koto ni suru。	Human being			N1 决定 X

[2] The Translation of (VP+LV) Structure:

We studied the (VP+LV) structures and found that the light verb *suru* is usually responsible for representing inflectional information: tense, causative, passive, politeness on speech level and negation and acts as the auxiliary verb. The corresponding Chinese for LVCs in this structure are mainly functions as the conjunctions and adverb. We considered a Japanese VP roughly composed of two sections: propositional content and function words after predicate (FWAP). We proposed the translation rules by judging the FWAP of the VP and light verb *suru* without taking the attributes of verbs into consideration (see Table3).

Table3. Translation rules for (NP+LV) structure (Partial)

class	ID	Conditions Japanese Pattern	FWAP Of VP	FWAP of LV	Translation rules
9	(1)	N1 ga VP rou to suru	rou to		N1 试图 VP
10	(2)	N1 ga VP you to suru	you to		N1 正要 VP
11	(3)	VP to sitara, X	to	tara	如果 VP, 那么 X
11	(4)	VP to sidemo, X	to	demo	即使 VP, 也 X
13	(5)	VP1tari VP 2tari suru	tari		一边 VP1, 一边 VP2

[3] The Translation of (VN+LV) Structure:
In our system, we consider the [VN*suru*] as whole content word (key word) and summarize the rules for each [VN*suru*] respectively. Here we discuss about the [VN-o suru] structure. The basic construction is [N1 ga VN-o suru]. We Check whether VN in Chinese is a pure noun or not. If VN is a pure noun, we need to choose the proper Chinese verb for LV according to the semantic attributes of VN in Japanese. Some rules we proposed are shown in table 4.

Table4. Translation rules for (VN-o LV) structure for pure noun (Partial)

	Conditions and attributions					
ID	Japanese Pattern	Attribution of N1	Attribution of N2	Attribution Of VN	VN in Chinese	Translation rules
(1)	N1 *ga* N2 *to* *VN-o suru*	subject	Human being	conversation	Pure noun	N1 和 N2 进行 VN
(2)	N1 *ga* N2 ni/ *to VN-o suru*	subject	Human being	correspondence	Pure noun	N1 给 N2 打 VN
(3)	N1 *ga* N2 *to* *VN-o suru*	subject	noun	action	Pure noun	N1 和 N2 做 VN
(4)	N1 *ga* N2 *to* *VN-o suru*	subject	noun	sports	Pure noun	N1 和 N2 玩 VN

If VN is also a verbal noun in Chinese, then VN can directly be adopted as a verb. However, we have to pay attention to the treatment of modifiers of VN. We classify the modifiers into the following three types and propose the translation rules (table5).

Table 5. Translation rules for (VN-o LV) structure for verbal noun (Partial)

modifier tpyes	Examples	Conditions: Japanese Pattern	Conditions: VN in Chinese	Translation rules
1.telic	Tokyo e no ryokoo o suru	N1 *ga* N2 e/ni no *VN-osuru*	Verbal noun	N1 去 N2 VN
2.Bipredicational	Eigo no bennkyoo o suru	N1 *ga* N2 no *VN-osuru*	Verbal noun	N1VN N2
3.adjective	Tanosii ryokoo o sita	N1 *ga* ADJ *VN-osuru*	Verbal noun	N1VN 得 ADJ

[4] The Translation of (ADJ+LV), (ADV+LV) and (ONO+LV) Structure:
The Construction of (ADJ+LV) is usually used to express the result of the action, while the Construction of (ADV+LV) and (ONO+LV) can express the static property of something or the action of someone or something, as well as the result of action. When expressing the static property and actions of someone or something, according to the semantic features of Chinese, the meaning of LV is completely bleached. We summarize the translation rules in table 6.

To avoid the overgeneralization of the rules for LVCs, We also summarize numbers of literal patterns based on a fixed word in combination with LV to express

Table 6. Translation rules for (ADJ+LV), (ADV+LV) and (ONO+LV) structures (Partial)

ID	semantics	Japanese Pattern	Translation rules	examples
(1)	Result of an action	N1 ga N2 wo ADJ suru	N1 使 N2 变得 ADJ	女性を美しくする
(2)	Result of an action	N1 ga N2 wo ADV/ONO ni suru	N1 把 N2 弄得 ADV/ONO	家をぴかぴかにする
(3)	Static property or an action	N1 ga ADV/ONO (ni/ to) suru	N1 ADV/ONO	彼はいつものんびりしている。

the Chinese idioms or phrase:

e.g. (1) Pattern: N1 ga X(adjective) <u>ki</u> ga suru.
Chinese: N1 you X de xinqing (N1 is in a X mood).

e.g. (2) Pattern: N1 ga N2 wo <u>mimi</u> ni suru
Chinese: N1 tingshuo N2 (N1 hears N2).

When using these rules for translation, situations may arise in which more than one rule is matched. In this case, the optimal solution is selected according to the following conditions:

(1) The type and number of patterns used to compose the construction tree for the sentence.

According to the types of the pattern, Base type (section 4.2) takes priority than Addition types (section 4.2), and it gives priority to patterns that can compose the construction tree with less number.

(2) Types of limiting conditions.

There are there kinds of limiting conditions, the conditions of content word, the conditions of functional word and the fixed literal conditions. The strictness of each pattern is judged by these three conditions.

(3) The distance between semantic attributes in a thesaurus.

It gives priority to the one with short distance between the semantic attributes given in the pattern and the semantic attributes in the input sentence in a thesaurus.

2.3 The Comparison with Japanese -English Patterns

In this section, we made comparisons of the Japanese-Chinese translation patterns for LVCs mentioned in section 2 with the Japanese-English translation patterns listed in *Goi-Taikei--A Japanese Lexicon*[14](Table 7).

Table 7. The comparison of the number of translation patterns with Japanese-English

	The number of Japanese-English translation patterns	The number of Japanese-Chinese translation patterns
LVCs	319	92

The statistics shows that the number of Japanese-Chinese translation patterns is considerably less than the number of Japanese-English translation patterns. The difference can be analyzed as follows.

1. One Japanese-Chinese translation pattern corresponds to several Japanese-English translation patterns. We observed the following two points as the major reasons:

[1] The number of Japanese-English translation patterns with the structure [*N1 ga N2 wo N3 ni(to) suru*], in which N3 is the literal condition for the patterns, is more than half of the all. While in Chinese, there is a special sentence pattern called [ba] sentence which corresponds to most of the Japanese sentences with the structure [*N1 ga N2 wo N3 ni (to) suru*]. For example:

① Japanese: N1gaN2 wo hokori ni suru English: N1 be proud of N2
② Japanese: N1ga N2 wo urimono ni suru English: N1 feature N 2
③ Japanese: N1ga N2wo aite ni suru English: N1 make a companion of N 2
The Chinese translation for the above patterns is all [*N1 ba N2 dangzuo N3*].

[2] Compared with Chinese, preposition in English is more distinguishably used for different nouns. For example, due to the different prepositions, there are some Japanese-English patterns taking the structure of [*N1ga N2 woN3 no N4 ni(to) suru*] with N4 as the literal condition :

④Japanese: N1ga N2 woN3 no koujitu ni suru
English: N1 make N2 an excuse <u>for</u> N3
⑤Japanese: N1ga N2 wo N3 no mohann ni suru
English: N1makeN2 the example <u>of</u> N3
⑥Japanese: N1ga N2 wo N3 no mokuhyou to suru
English: N1aim at N2 <u>as</u> a target <u>for</u> N3

While for the translation to Chinese, the combination of the pattern ([N1*ga N2 woN3 ni (to) suru*]=>[*N1 ba N2 dangzuo N3*]) and pattern([*N1 no N2*] =>[*N1 de N2*]) can correspond to the above patterns([*N1 ba N2 dangzuo N3 de N4*]).

2. There is no corresponding Japanese-English translation pattern for certain Japanese-Chinese translation patterns. In this case it is because that Japanese-English translation patterns are not complete especially translation patterns for some sentence patterns or idioms. For instance,

⑦Japanese: N1 *kara suru to*X
English: From N1, X Chinese: *cong* N1*laikan* X
⑧Japanese: N1*ga* X(verb-equivalent words) *kao* (expression) wo suru
English: N1show the expression that X Chinese: N1 *louchu yifu* X *de biaoqing*
⑨Japanese: N1*ga ooki kao wo suru*
English: N1 be self-conceit Chinese : N1xiande liaobuqi/ N1baidajiazi

3 Outline of the *Jaw/Chinese* MT System

Jaw/Chinese is a machine translation system developed in our lab, which translates from Japanese to Chinese based on a pattern transfer paradigm. *Jaw* (from **J**apanese to **A**sian and **W**orld languages) is the translation engine, which is applicable to any target language.

Figure 1 shows a rough outline of *jaw*/*Chinese*, which contains three sub-processes: parsing a Japanese sentence, transferring it to a Chinese expression structure, and then generating a Chinese sentence.

First, a Japanese input sentence is morphologically and syntactically analyzed using *Ibuki*, a Japanese syntactical structure analysis system developed in our laboratory, and a dependency tree is constructed. Each node of the tree corresponds to one characteristic Japanese linguistic unit (*bunsetsu*), which is a phrase including one content word and any number of function words (or none). For example: *kowasarete simatta kamo siremasen ga* (however, (it) may have been destroyed) is one *bunsetsu*, composed of one content word (*kowasu*: destroy) and several function words (*reru*: be done to), (*tesimau*: have done), (*ta*: past), (*kamosiremasenn*: may be) and (*ga*: however). In this step, an Input Tree (IT) is created.

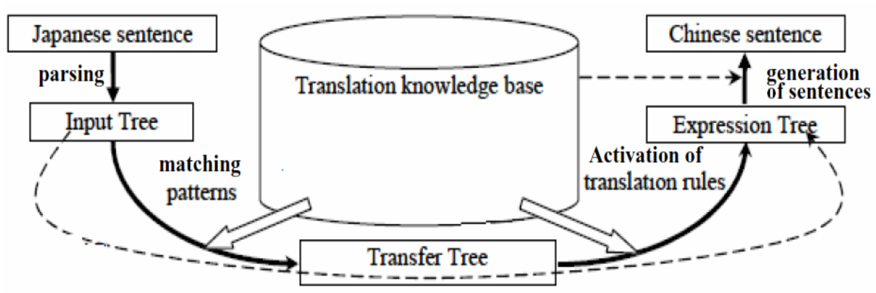


Fig. 1. outline of jaw/Chinese

Secondly, the input sentence (IT) is matched with the most similar expression pattern among those pre-stored in the transfer dictionary. Three types of transfer rule are employed for Japanese expression patterns: a base-type rule and two addition-type rules (see Section 4.2). This step creates a Transfer Tree (TT) with corresponding transfer rules. The rules for translating Japanese expression patterns to Chinese help to disambiguate correspondences in the translation. The conditions of Japanese patterns provide a way for disambiguation.

The translation rules are represented by C++ program stored as a dll file. The execution of this program creates a network of C++ objects, which represents the corresponding Chinese expression structure (Expression Tree, ET). The ET is an assembly of linguistic parts as members of C++ object used to express an input Japanese sentence in Chinese. Finally, execution of a linearization function on the ET places the members of the ET in a single line to give a Chinese output sentence with the correct sentence order. A linearization function is defined for each object as a C++ class method. We designed classes such as CProposition, CNoun, CQuantity, CpConnection, CAdverb, CModeC, and CModeJ for the Chinese ET.

4 Experiments and Evaluation

In order to verify and evaluate the translation rules for LVCs described in Section 2, a manual translation experiment was carried out; in addition, to verify that the translation rules are applicable in the real MT system, translation experiments were conducted using the *jaw/Chinese*.

4.1 Manual Experiments

We extracted 200(excluding the 1000 investigating sentences) Japanese sentences containing LVC*s* from a Japanese-English parallel corpus. Because the dictionary and the database of our machine translation system *jaw/Chinese* is still rather small, these sentences were first translated manually using our rules (translation A), and concurrently using commercially available MT software (translation B). We evaluated the translation results manually, focusing on the LVCs and judging for each individual sentence whether a suitable word for *LV* was used in the translation and whether the word was placed in the correct order. Each translated sentence was graded as follows:

○ = In terms of grammar, as accurate as natural Chinese translation.
△ = A little unnatural grammatically, but the overall meaning of the sentence is correct.
× = Incorrect both grammatically and in terms of overall meaning.

Evaluation of the results was carried out using two alternative criteria: (1) ○ is counted as a correct translation, and △ and × are incorrect translations and (2) ○ and △ are counted as correct translations, and × is incorrect. The results are shown in Table 8.

Table 8. Evaluation of manual experiments

	Evalu ated	Correct (○)		Correct rate (○)		Correct (○ and △)		Correct rate (○ and △)	
		A	B	A	B	A	B	A	B
LVCs	200	166	82	83%	41%	168	102	84%	51%

(A: translation using our rules; B: translation by commercially available MT software)

From the results, we can conclude that our method is valid and could achieve a rather high accuracy, compared with commercially available MT software.

4.2 Experiments Using *Jaw/Chinese*

We also conducted experiments using *jaw/Chinese* to verify the correctness and the applicability of the translation rules mentioned in Section 2. To analyze Japanese sentence patterns, we designed a base-type rule and two addition-type rules, denoted as Base Type, AdditionCW Type, and AdditionFW Type, respectively. The base-type rule is a case-frame-based rule, and deals with the translation of basic propositional content. Addition-type rules deal with adverbial and conjunctional expressions; these may be optionally added to base-type expressions. Table 9 shows two examples of them (one example is for NP+LV structure and the other is for VP+LV structures)

Table 9 also shows the corresponding translation rules for Chinese. Classes, member classes, member names, values, and case markers are the stuffs to build a C++ program for the translation rules; The C++ program is automatically constructed with these stuffs. We developed a *Jaw*-editor to write the Japanese patterns and transfer rules. We implemented the translation rules proposed in section 2 into Jaw/Chinese by Jaw-editor. However, some rules which could not be described using the *jaw*-editor were written in tables of function words after predicate (FWAP), which contain information about modality, tense, aspect, speech acts, adverbs and so on.

Table 9. Japanese expression patterns and translation rules for Chinese

(1-1) Example 1of Japanese pattern for base type

Bunsetsu number	Dependency Bunsetsu Number	Semantic conditions of Dependent Bunsetsu	Function word (particle)	Key-word	Semantic conditions of Key-word
1	4	Noun, pronoun	ga		
2	4	Descendant,offspring	wo		
3	4	occupation	ni		
4	0			suru	action

（1-2）The corresponding translation rules for Chinese of Example 1

PatternID	Rule type	Class	Member Name	Member Class	Value	role Name	case Marker
1114979	Base Type	CProposition	m_centerW	CString	要	target	把
			m_subject	CNoun	1		
			m_directobject	CNoun	2		
			m_clauseObject	CProposition			
			m_centerW	CString	培养成		
			m_directobject	CNoun	3		

(2-1) Example2 of Japanese pattern for AdditionFW type

Bunsetsu number	Dependency Bunsetsu Number	Conditions of Dependent Bunsetsu	Function word	Conditions of content word	Key word	conditions of dependency Bunsetsu
1	2	VP	to			
2	3			suru	to	
3	0					VP

（2-2）The corresponding translation rules for Chinese of Example2

PatternID	Rule type	Class	Member Name	Member Class	Value
5001006	Addition FW Type	CProposition	m_pConnection	CpConnection	
			m_pSubordinate	CProposition	1
			m_connectWord1	CString	如果
			m_connectPos1	CString	head
			m_connectWord2	CString	那么
			m_connectPos2	CString	Head

In this way, we implemented the translation rules in *jaw/Chinese*; next, we carried out closed and open tests using the system. Closed tests were conducted on 30 sentences containing LVCs; all of the translations results were correct. Open tests were carried out using 100 sentences, and the results are shown in Table10. Both of the test shows that the rules is applicable in real MT system and the method gave translation results with a high accuracy, compared with commercially available MT software.

Table10. Evaluation of *jaw/Chinese* open-test experiments

	Evaluated	Correct (○)		Correct rate (○)		Correct (○ and △)		Correct rate (○ and △)	
		A	B	A	B	A	B	A	B
LVCs	100	82	41	82%	41%	84	48	84%	48%

5 Conclusions

We discussed Japanese LVCs, analyzed their diversified correspondence with Chinese, proposed a translation method for these constructions, and implemented translation rules based on this method in our *jaw/Chinese* MT system. We conduct manual experiments using 200 sentences to verify the effectiveness of the rules and made experiments by machine *(jaw/Chinese)* using 100 sentences to verify the applicability of the rules in real MT system. Both of the experiments gave a high translation accuracy of over 80% and verified the high performance and effectiveness of our method.

References

1. Otto Jespersen. A Modern English Grammar on Historical Principles.London: George Allen & Unwin and Copenhagen: Ejnar Munksgaard(1954)
2. R. Cattell. Composite Predicates in English. Academic Press, New York(1984)
3. Butt, M. and Geuder, W.. On the (semi)lexical Status of Light Verbs. In Corver, N. and van Riemsdijk, H., editors, Semi-lexical Categories: On the content of function words and the function of content words. Mouton de Gruyter, Berlin. (2001) 323–370
4. 4. Suzanne Stevenson and Afsaneh Fazly and Ryan North Statistical Measures of the Semi-Productivity of Light Verb Constructions Second ACL Workshop on Multiword Expressions: Integrating Processing (2004) 1-8
5. Gholamhossein Karimi-Doostan. Light verbs and Structural Case. Lingua 115 (2005) 1737–1756
6. Miriam Butt. The Light Verb Jungle. http://ling.uni-konstanz.de/pages/home/butt/harvard-work.pdf (2003).
7. Jane Grimshaw and Armin Mester. Light Verbs and 1 -Marking. Linguistic Inquiry. 19 (1988) 205–232,
8. Atsuko Shimada and Valia Kordoni. Japanese “Verbal Noun and Suru” Constructions. Proceedings of the workshop on Multi-Verb constructions Trondheim Summer School 2003
9. Atsushi Fujita Kentaro Furihata Kentaro Inui Yuji Matsumoto Koichi Takeuchi Paraphrasing of Japanese Light-verb Constructions Based on Lexical Conceptual Structure Second ACL Workshopon Multiword Expressions:Integrating Processing July (2004)
10. Yo Matsumoto. A Syntactic Account of Light Verb Phenomena in Japanese. Journal of East Asian Linguistics 5 (1996) 107-149.
11. Tadao Miyamoto. The Light Verb Construction in Japanese: the role of the verbal noun. John Benjamins Publishing Company. (2000)
12. Yokota, K. The structure and meaning of Japanese light verbs. Language sciences 27,3 (2005) 247-280
13. Tori-bank. http://unicorn.ike.tottori-u.ac.jp/toribank/.
14. NTT Communication Science Basic Research. *Goi-Taikei--A Japanese Lexicon.* Iwanami Publisher. (1997)

Shahmukhi to Gurmukhi Transliteration System: A Corpus based Approach

Tejinder Singh Saini[1] and Gurpreet Singh Lehal[2]

[1] Advanced Centre for Technical Development of Punjabi Language, Literature & Culture,
Punjabi University, Patiala 147 002, Punjab, India
tej@pbi.ac.in
http://www.advancedcentrepunjabi.org
[2] Department of Computer Science, Punjabi University, Patiala 147 002,
Punjab, India
gslehal@yahoo.com

Abstract. This research paper describes a corpus based transliteration system for Punjabi language. The existence of two scripts for Punjabi language has created a script barrier between the Punjabi literature written in India and in Pakistan. This research project has developed a new system for the first time of its kind for Shahmukhi script of Punjabi language. The proposed system for Shahmukhi to Gurmukhi transliteration has been implemented with various research techniques based on language corpus. The corpus analysis program has been run on both Shahmukhi and Gurmukhi corpora for generating statistical data for different types like character, word and n-gram frequencies. This statistical analysis is used in different phases of transliteration. Potentially, all members of the substantial Punjabi community will benefit vastly from this transliteration system.

1 Introduction

One of the great challenges before Information Technology is to overcome language barriers dividing the mankind so that everyone can communicate with everyone else on the planet in real time. South Asia is one of those unique parts of the world where a single language is written in different scripts. This is the case, for example, with Punjabi language spoken by tens of millions of people but written in Indian East Punjab (20 million) in Gurmukhi script (*a left to right script based on Devanagari*) and in Pakistani West Punjab (80 million), written in Shahmukhi script (*a right to left script based on Arabic*), and by a growing number of Punjabis (2 million) in the EU and the US in the Roman script. While in speech Punjabi spoken in the Eastern and the Western parts is mutually comprehensible, in the written form it is not. The existence of two scripts for Punjabi has created a script barrier between the Punjabi literature written in India and that in Pakistan. More than 60 per cent of Punjabi literature of the medieval period (500-1450 AD) is available in Shahmukhi script only, while most of the modern Punjabi writings are in Gurmukhi. Potentially, all members of the substantial Punjabi community will benefit vastly from the transliteration system.

Received 25/10/07
Accepted 07/12/07
Final Version 22/01/08

2 Related Work

Most of the available work in Arabic-related transliteration has been done for the purpose of machine translation. In the paper titled "Punjabi Machine Transliteration (PMT)" Malik A. 2006 [1] has demonstrated a very simple rule-based transliteration system for Shahmukhi to Gurmukhi script. Firstly, two scripts are discussed and compared. Based on this comparison and analysis, character mappings between Shahmukhi and Gurmukhi scripts have been drawn and transliteration rules formulated. Along with this only dependency rules have been formed for special characters like aspirated consonants, non-aspirated consonants, Alif ا[ə], Alif Madda آ[ɑ], Vav و[v] , Choti Ye ی[j] etc. The primary limitation of this system is that this system works only on input data which has been manually edited for missing vowels or diacritical marks (*the basic ambiguity of written Arabic script*) which practically has limited use. Some other transliteration systems available in literature are discussed by Haizhou et al (2004) [3], Youngim et al (2004) [4], Nasreen et al (2003) [5] and Stalls et al (1998) [9].

3 Major Challenges

The major challenges of transliteration of Shahmukhi to Gurmukhi script are as follows:

3.1 Recognition of Shahmukhi Text without Diacritical Marks

Shahmukhi script is usually written without short vowels and other diacritical marks, often leading to potential ambiguity. Arabic orthography does not provide full vocalization of the text, and the reader is expected to infer short vowels from the context of the sentence. Like Urdu, in the written Shahmukhi script it is not mandatory to put short vowels below or above the Shahmukhi character to clear its sound. These special signs are called "Aerab" in Urdu. It is a big challenge in the process of machine transliteration or in any other process to recognize the right word from the written text because in a situation like this, correct meaning of the word needs to be distinguished from its neighboring words or, in worst cases, we may need to go into deeper levels of n-gram.

3.2 Filling the Missing Script Maps

There are many characters which are present in the Shahmukhi script, corresponding to those having no character in Gurmukhi, e.g. Hamza ء [ɪ], Do-Zabar ◌ً [ən], Do-Zer ◌ٍ [ɪn], Aen ع[ʔ] etc.

3.3 Multiple Mappings

It is observed that there is multiple possible mapping into Gurmukhi script corresponding to a single character in the Shahmukhi script as shown in Table 1.

Table 1. Multiple Mapping into Gurmukhi Script

Name	Shahmukhi Character	Unicode	Gurmukhi Mappings
Vav	و [v]	0648	ਵ [v], ੋ [o], ੌ [ɔ], ੁ [ʊ], ੂ [u], ਓ [o]
Ye Choti	ی[j]	0649	ਯ [j], ਿ [ɪ], ੇ [e], ੈ[æ], ੀ[i], ਈ [i]

3.4 Word-Boundary Mismatch

Urdu Zabata Takhti (UZT) 1.01 [2] has the concept of two types of spaces. The first type of space is normal space and the second type of space is given name Hard Space (HS). The function of hard space is to represent space in the character sequence that represents a single word. In Unicode character set this Hard Space is represented as Zero Width Non Joiner (ZWNJ). But it is observed that in the written text normal space is used instead of hard space. Therefore, transliterating a single word of Shahmukhi with space in between will generate two tokens of the same word in Gurmukhi script.

4 Script Mappings

4.1 Gurmukhi Script

The Gurmukhi script, derived from the Sharada script and standardised by *Guru Angad Dev* in the 16th century, was designed to write the Punjabi language. The meaning of "Gurmukhi" is literally "*from the mouth of the Guru*". As shown in Table 2 the Gurmukhi script has forty one letters, including thirty eight consonants and three basic vowel sign bearers (*Matra Vahak*). The first three letters are unique because they form the basis for vowels and are not consonants. The six consonants in the last row are created by placing a *dot* at the foot (pair) of the consonant (*Naveen Toli*). There are five nasal consonants (ਙ[ɲə], ਞ[ɲə], ਣ[ɳ], ਨ[n], ਮ[m]) and two additional nasalization signs, bindi ਂ [ɲ] and tippi ੰ [ɲ] in Gurmukhi script. In addition to this, there are nine dependent vowel signs (ੁ[ʊ], ੂ [u], ੋ[o], ਾ[ə], ਿ[ɪ], ੀ[i], ੇ[e], ੈ[æ], ੌ[ɔ]) used to create ten independent vowels (ਉ [ʊ], ਊ [u], ਓ [o], ਅ [ə], ਆ [ɑ], ਇ [ɪ], ਈ [i], ਏ [e], ਐ [æ], ਔ [ɔ]) with three bearer characters: Ura ੳ[ʊ], Aira ਅ [ə] and Iri ੲ[ɪ]. With the exception of Aira ਅ [ə] independent vowels are never used without additional vowel signs. Some Punjabi words require consonants to be written in a conjunct form in which the second consonant is written under the first as a subscript. There are only three commonly used subjoined consonants as shown here Haha ਹ[h] (usage ਨ[n] +◌੍+ਹ[h] = ਨ੍ਹ [n^h]), Rara ਰ[r] (usage ਪ[p] +◌੍+ਰ[r] =ਪ੍ਰ [pr^h]) and Vava ਵ[v] (usage ਸ[s] +◌੍+ਵ[v] = ਸ੍ਵ [sv]).

Table 2. Gurmukhi Alphabet

ੳ	ਅ[ə]	ੲ				*Matra Vahak*
			ਸ[s]	ਹ[h]		*Mul Varag*
ਕ[k]	ਖ[k^h]	ਗ[g]	ਘ[k^h]	ਙ[ɲə]		*Kavarg Toli*
ਚ[ʧ]	ਛ[ʧ^h]	ਜ[ʤ]	ਝ[ʤ^h]	ਞ[ɲə]		*Chavarg Toli*
ਟ[ʈ]	ਠ[ʈ^h]	ਡ[ɖ]	ਢ[ɖ^h]	ਣ[ɳ]		*Tavarg Toli*
ਤ[t̪]	ਥ[t̪^{h}]	ਦ[d̪]	ਧ[d̪^{h}]	ਨ[n]		*Tavarg Toli*
ਪ[p]	ਫ[p^h]	ਬ[b]	ਭ[b^h]	ਮ[m]		*Pavarg Toli*
ਯ[j]	ਰ[r]	ਲ[l]	ਵ[v]	ੜ[ɽ]		*Antim Toli*
ਸ਼[ʃ]	ਖ਼[x]	ਗ਼[ɣ]	ਜ਼[z]	ਫ਼[f]	ਲ਼[ɭ]	*Naveen Toli*

4.2 Shahmukhi Script

The meaning of "Shahmukhi" is literally “from the King's mouth". Shahmukhi is a local variant of the Urdu script used to record the Punjabi language. It is based on right to left Nastalique style of the Persian and Arabic script. It has thirty seven simple consonants, eleven frequently used aspirated consonants, five long vowels and three short vowel symbols.

4.3 Mapping of Simple Consonants

Unlike Gurmukhi script, the Shahmukhi script does not follow a *‘one sound-one symbol’* principle. In the case of non-aspirated consonants, Shahmukhi has many character forms mapped into single Gurmukhi consonant. This has been highlighted in Table 3 below.

4.4 Mapping of Aspirated Consonants (AC)

In Shahmukhi script, the aspirated consonants are represented by the combination of a simple consonant and HEH-DAOCHASHMEE ھ[h]. Table 4 shows 11 frequently used aspirated consonants in Shahmukhi corresponding to which Gurmukhi script has unique single character except the last one ੜ੍ਹ [ɽ^h] having compound characters.

Table 3. Shahmukhi Non-Aspirated Consonents Mapping

Sr.	Char	Code	Gurmukhi	Code	Sr.	Char	Code	Gurmukhi	Code
1	ب[b]	0628	ਬ [b]	0A2C	20	ع[ʔ]	0639	ਅ [ə]	0A05
2	پ[p]	067E	ਪ [p]	0A2A	21	غ[ɣ]	063A	ਗ਼ [ɣ]	0A5A
3	ت[t̪]	062A	ਤ [t̪]	0A24	22	ف[f]	0641	ਫ਼ [f]	0A5E
4	ث[s]	062B	ਸ [s]	0A38	23	ق[q]	0642	ਕ [k]	0A15
5	ج[ʤ]	062C	ਜ [ʤ]	0A1C	24	ک[k]	06A9	ਕ [k]	0A15
6	چ[ʧ]	0686	ਚ [ʧ]	0A1A	25	گ[g]	06AF	ਗ [g]	0A17
7	ح[h]	062D	ਹ [h]	0A39	26	ل[l]	0644	ਲ [l]	0A32
8	خ[x]	062E	ਖ਼ [x]	0A59	27	م[m]	0645	ਮ [m]	0A2E
9	د[d̪]	062F	ਦ [d̪]	0A26	28	ن[n]	0646	ਨ [n], ੰ [ɲ]	0A28, 0A70
10	ذ[z]	0630	ਜ਼ [z]	0A5B	29	ݨ[ɳ]	06BB	ਣ [ɳ]	0A23
11	ر[r]	0631	ਰ [r]	0A30	30	و[v]	0648	ਵ [v]	0A35
12	ز[z]	0632	ਜ਼ [z]	0A5B	31	ہ[h]	06C1	ਹ [h]	0A39
13	ژ[ʒ]	0698	ਜ਼ [z]	0A5B	32	ی[j]	06CC	ਯ [j]	0A2F
14	س[s]	0633	ਸ [s]	0A38	33	ے[j]	06D2	ਯ [j]	0A2F
15	ش[ʃ]	0634	ਸ਼ [ʃ]	0A36	34	ھ[h]	06BE	੍ਹ [h]	0A4D +0A39
16	ص[s]	0635	ਸ [s]	0A38	35	ٹ[ʈ]	0679	ਟ [ʈ]	0A1F
17	ض[z]	0636	ਜ਼ [z]	0A5B	36	ڈ[ɖ]	0688	ਡ [ɖ]	0A21
18	ط[t̪]	0637	ਤ [t̪]	0A24	37	ڑ[ɽ]	0691	ੜ [ɽ]	0A5C
19	ظ[z]	0638	ਜ਼ [z]	0A5B					

Table 4. Aspirate Consonants (AC) Mapping

Sr.	AC ھ[h]	Code (06BE)	Gurmukhi	Code	Sr.	AC ھ[h]	Code (06BE)	Gurmukhi	Code
1	بھ[b]	0628	ਭ [b]	0A2D	7	دھ[d̪]	062F	ਧ [d̪]	0A27
2	پھ[p]	067E	ਫ [p]	0A2B	8	ٹھ[ʈ]	0679	ਠ [ʈ]	0A20
3	تھ[t̪]	062A	ਥ [t̪]	0A25	9	کھ[k]	06A9	ਖ [k]	0A16
4	ڈھ[ɖ]	0688	ਢ [ɖ]	0A22	10	گھ[g]	06AF	ਘ [g]	0A18
5	جھ[ʤ]	062C	ਝ [ʤ]	0A1D	11	ڑھ[ɽ]	0691	ੜ੍ਹ [ɽ]	0A5C+ 0A4D+ 0A39
6	چھ[ʧ]	0686	ਛ [ʧ]	0A1B					

Table 5. Shahmukhi Long Vowels Mapping

Sr.	Vowel	Code	Mapping	Code	Sr.	Vowel	Code	Mapping	Code
1	ا [ə]	0627	ا →ਅ [ə]	0A05	4	و [o]	0648	و→ਵ [v]	0A35
			ا →◌ਾ [ə]	0A3E				و→◌ੋ [o]	0A4B
2	آ [ɑ]	0622	آ→ ਆ [ɑ]	0A06				و →◌ੌ[ɔ]	0A4C
3	ی[i]	0649	ی→ ਈ [i]	0A08				و→◌ੁ [ʊ]	0A41
			ی→ ਯ [j]	0A2F				و→◌ੂ [u]	0A42
			ی→ ਿ◌ [ɪ]	0A3F				و→ਓ [o]	0A13
			ی→ ◌ੀ [i]	0A40	5	ے[e]	06D2	ے→ਏ [e]	0A0F
			ی→ ◌ੇ [e]	0A47				ے→ਯ [j]	0A2F
			ی→ ◌ੈ [æ]	0A48				ے→◌ੇ [e]	0A47
								ے→◌ੈ [æ]	0A48

Table 6. Shahmukhi Short Vowels Mapping

Sr.	Vowel	Unicode	Name	Gurmukhi	Unicode
1	◌ِ [ɪ]	0650	Zer	ਿ◌ [ɪ]	0A3F
2	◌ُ [ʊ]	064F	Pesh	◌ੁ [ʊ]	0A4B
3	◌َ [ə]	064E	Zabar	-	-

Table 7. Mapping of other Diacritical Marks or Symbols

Sr.	Shahmukhi		Unicode	Gurmukhi	Unicode
1	Noon ghunna	ں [ɲ]	06BA	◌ਂ [ɲ]	0A02
2	Hamza	ء [ɪ]	0621	positional dependent	-
3	Sukun	◌ْ	0652	◌ੂ ਨ [un]	0A42, 0A28
4	Shad	◌ّ	0651	◌ੱ	0A71
5	Khari Zabar	◌ٰ [ə]	0670	◌ਾ [ə]	0A3E
6	do Zabar	◌ً [ən]	064B	ਨ [n]	0A28
7	do Zer	◌ٍ [ɪn]	064D	ਿ◌ ਨ [ɪn]	0A3F, 0A28

4.5 Mapping of Vowels

The long and short vowels of Shahmukhi script have multiple mappings into Gurmukhi script as shown in Table 5 and Table 6 respectively. It is interesting to observe that Shahmukhi long vowel characters Vav و[v] and Ye ی,ے [j] have vowel-vowel multiple mappings as well as one vowel-consonant mapping.

4.6 Mapping other Diacritical Marks or Symbols

Shahmukhi has its own set of numerals that behave exactly as Gurmukhi numerals do with one to one mapping. Table 7 shows the mapping of other symbols and diacritical marks of Shahmukhi.

5 Transliteration System

The transliteration system is virtually divided into two phases. The first phase performs pre-processing and rule-based transliteration tasks and the second phase performs the task of post-processing. In the post-processing phase bi-gram language model has been used.

5.1 Lexical Resources Used

In this research work we have developed and used various lexical resources, which are as follows:
Shahmukhi Corpus: There are very limited resources of electronic information of Shahmukhi. We have created and are using a Shahmukhi corpus of 3.3 million words.
Gurmukhi Corpus: The size of Gurmukhi corpus is about 7 million words. The analysis of Gurmukhi corpus has been used in pre and post-processing phases.
Shahmukhi-Gurmukhi Dictionary: In the pre-processing phase we are using a dictionary having 17,450 words (most frequent) in all. In the corpus analysis of Shahmukhi script we get around 91,060 unique unigrams. Based on the probability of occurrence we have incorporated around 9,000 most frequent words in this dictionary. Every Shahmukhi token in this dictionary structure has been manually checked for its multiple similar forms in Gurmukhi e.g. token اس [əs] has two forms with weights[1] as ਇਸ{59998} [ɪs] (this) and ਉਸ{41763} [ʊs] (that).
Unigram Table: In post-processing tasks we are using around 163,532 unique weighted unigrams of Gurmukhi script to check most frequent (MF) token analysis.
Bi-gram Tables: The bi-gram queue manager has around 188,181 Gurmukhi bi-grams resource to work with.

[1] Weights are unigram probabilities of the tokens in the corpus.

All Forms Generator (AFG): Unigram analysis of Gurmukhi corpus is used to construct AFG Component having 86,484 unique words along with their similar phonetic forms.

5.2 Pre-Processing and Transliteration

In pre-processing stage Shahmukhi token is searched in the Shahmukhi-Gurmukhi dictionary before performing rule-based transliteration. If the token is found, then the dictionary component will return a weighted set of phonetically similar Gurmukhi tokens and those will be passed on to the bi-gram queue manager. The advantage of using dictionary component at pre-processing stage is that it provides more accuracy as well as speeds up the overall process. In case the dictionary lookup fails then the Shahmukhi token will be passed onto basic transliteration component. The Token Converter accepts a Shahmukhi token and transliterates it into Gurmukhi token with the help of Rule Manager Component. Rule Manager Component has character mappings and rule-based prediction to work with. Starting from the beginning, each Shahmukhi token will be parsed into its constituent characters and analyzed for current character mapping along with its positional as well as contextual dependencies with neighboring characters. Shahmukhi script has some characters having multiple mappings in target script (as shown in Table 1 and 5).
Therefore, to overcome this situation extra care has been taken to identify various dependencies of such characters in the source script and prediction rules have been formulated accordingly to substitute right character of target script. Ultimately, a Gurmukhi token is generated in this process and that will be further analyzed in the post- processing activities of transliteration system. Figure 1 shows the architecture of this phase.

5.3 Post-Processing

The first task of this phase is to perform formatting of the Gurmukhi token according to Unicode standards. The second task in this phase is critical and especially designed to enable this system to work smoothly on Shahmukhi script having missing diacritical marks. The input Gurmukhi token has been verified by comparing its probability of occurrence in target script with predefined threshold value. The threshold value is minimum probability of occurrence among most frequent tokens in the Gurmukhi corpus. If the input token has more probability than the threshold value, it indicates that this token is most frequent and acceptable in the target script. Therefore, it is not a candidate for AFG routine and is passed on to the bi-gram queue manager with its weight of occurrence.
On the other hand, a token having probability of occurrence less than or equal to the threshold value becomes a candidate for AFG routine. In AFG routine input Gurmukhi token is examined by All Forms Generator (AFG) with the help of AF manager. AF Manager will generate a phonetic code corresponding to the characters of input Gurmukhi token. This phonetic code will be used by Similar Forms Generator (SFG) routine for producing a list of weighted Gurmukhi tokens with similar phonetic similarities. The suggestion rules will be used to filter out undesired

tokens from the list. This final list of Gurmukhi tokens will then pass on to bi-gram queue manager. The phonetic code generation rules along with suggestion rules play a critical role in the accuracy of this task.

5.4 Bi-gram Queue Manager

The system is designed to work on bi-gram language model in which the bi-gram queue of Gurmukhi tokens is maintained with their respective unigram weights of occurrence. The bi-gram manager will search bi-gram probabilities from bi-gram table for all possible bi-grams and then add the corresponding bi-gram weights. After that it has to identify and mark the best possible bi-gram and pop up the best possible unigram as output. This Gurmukhi token is then returned to the Output Text Generator for final output.

The Output Text Generator has to pack these tokens well with other input text which may include punctuation marks and embedded Roman text. Finally, this will generate a Unicode formatted Gurmukhi text as shown in Figure 2.

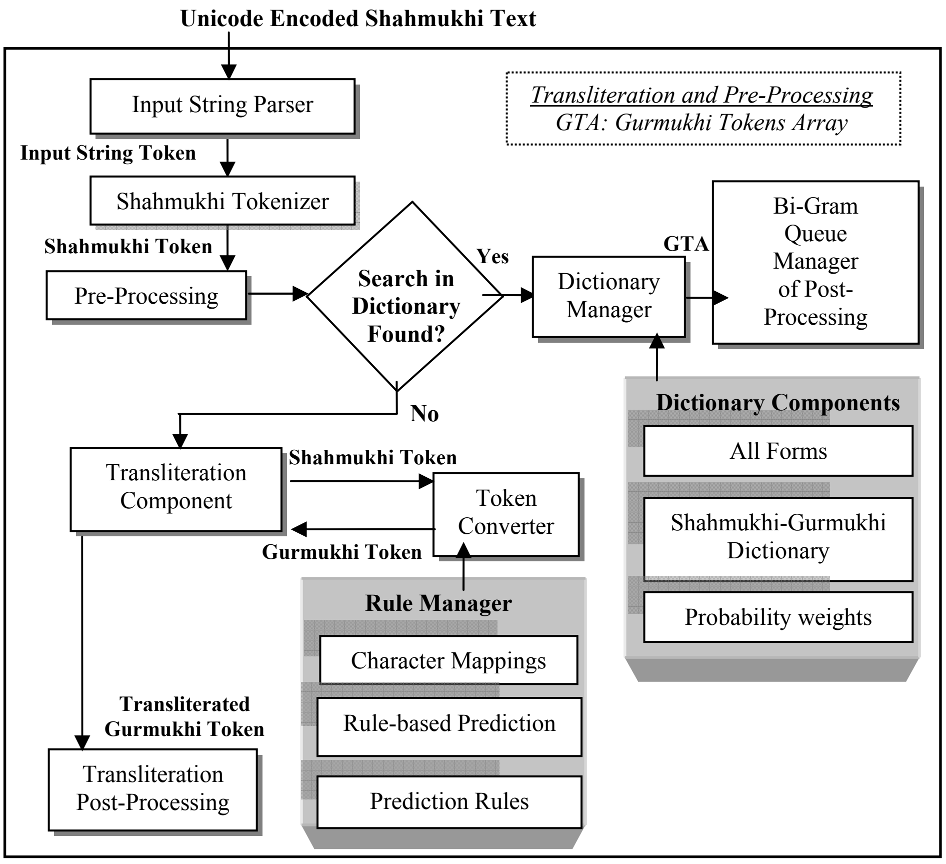

Unicode Encoded Shahmukhi Text
Input String Parser
Transliteration and Pre-Processing
GTA: Gurmukhi Tokens Array
Input String Token
Shahmukhi Tokenizer
Shahmukhi Token
Pre-Processing
Search in Dictionary Found?
Yes
Dictionary Manager
GTA
Bi-Gram Queue Manager of Post-Processing
No
Dictionary Components
All Forms
Shahmukhi-Gurmukhi Dictionary
Probability weights
Transliteration Component
Shahmukhi Token
Token Converter
Gurmukhi Token
Rule Manager
Character Mappings
Rule-based Prediction
Prediction Rules
Transliterated Gurmukhi Token
Transliteration Post-Processing

Fig. 1. Architecture of Transliteration and Pre-Processing

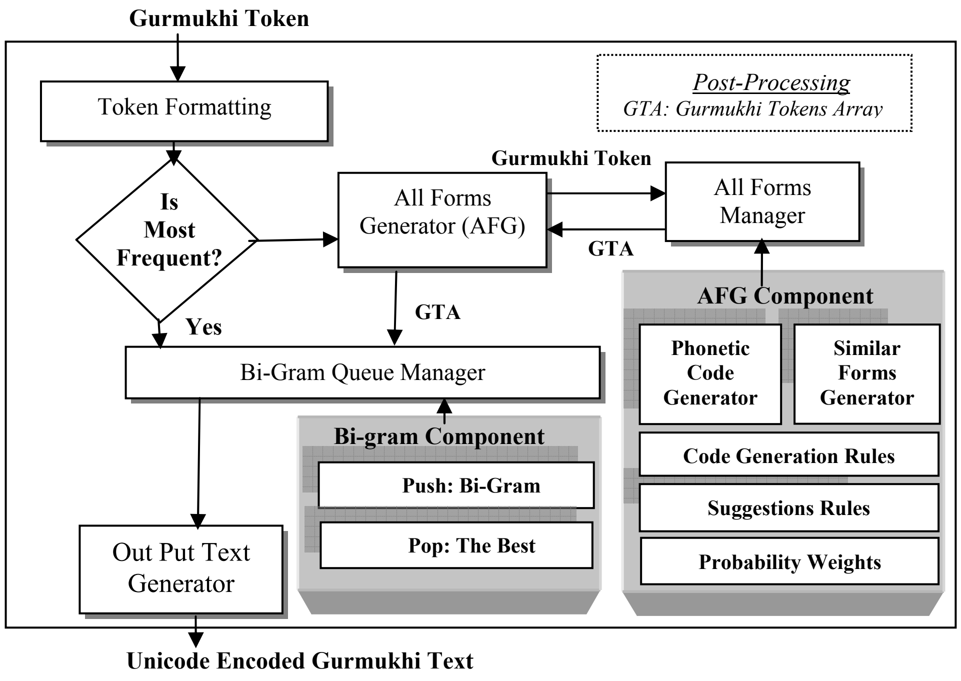


Fig. 2. Architecture of Post-Processing

6 Results and Discussion

The transliteration system was tested on a small set of poetry, article and story. The results reviewed manually are tabulated in Table 8. As we can observe, the average transliteration accuracy of 91.37% has been obtained.

Table 8. Transliteration Results

Type	Transliterated Tokens	Accuracy
Poetry	3,301	90.63769 %
Article	584	92.60274 %
Story	3,981	90.88043 %
Total	7,866	91.37362 %

Comparison with the Existing System

In actual practice, Shahmukhi script is written without short vowels and other diacritical marks. The PMT system discussed by Malik A. (2006) claims 98% accuracy only when the input text has all necessary diacritical marks for removing ambiguities. But this process of putting missing diacritical marks is not practically possible due to many reasons like large input size, manual intervention, person having knowledge of both the scripts and so on. We have manually evaluated PMT system against the following Shahmukhi input published on a web site and the output text is shown as output-A in table 9.The output of proposed system on the same input is shown as output-B. The wrong transliteration of Gurmukhi tokens is shown in bold and italic and the comparison of both outputs is shown in table 10.

Table 9. Input/Output of PMT and Proposed Systems

Input text (right to left)
اس گل وچ جدوں اسیں بہتے پنجابیاں نوں ویکھدے ہاں تاں پرنسپل تیجا سنگھ دے لیکھ وچ بیانیاں گئیاں کوڑیاں سچائیاں ہور وی شدت نال محسوس ہندیاں ہین۔ اسیں دیس نوں پیار کرن دا دعویٰ کردے ہاں پر اپنے صوبے نوں وساری بیٹھے ہاں۔ اس دا سبھ توں وڈا ثبوت ایہہ ہے کہ بھارت دے لگ بھگ بہتے صوبے اپنے اپنے ستھاپنا دوس بڑے اتشاہ تے جذبے نال مناؤندے ہین۔ اپنی زبان، اپنے سبھیاچار، اپنے پچھوکڑ تے اپنے ورثے تے مان کردے ہین۔ اپنی قومی پچھان تے مان کردے ہین۔ پر ساڈا بابا آدم ہی نرالا ہے۔ سرکاراں توں لے کے عام لوکاں تک پنجابی صوبے دے بنن دن بارے پوری طرحاں اویسلے ہی رہندے ہین۔
Output-A of PMT system (left to right)
ਅਸ ਗਲ ਵਚ ਜਦੋਂ ***ਅਸੇਂ ਬਹਤੇ ਪਨਜਾਬੇਆਂ ਨੌਂ*** ਵੇਖਦੇ ਹਾਂ ਤਾਂ ***ਪਰਨਸਪਲ*** ਤੇਜਾ ***ਸਨਘ*** ਦੇ ਲੇਖ ***ਵਚ ਬੇਅਨੇਆਂ*** ਗਈਆਂ ***ਕੌੜੇਆਂ*** ਸਚਾਈਆਂ ਹੋਰ ਵੀ ***ਸ਼ਦਤ*** ਨਾਲ ***ਮਹਸੋਸ ਹਨਦੇਆਂ ਹੇਨ***। ***ਅਸੇਂ*** ਦੇਸ ***ਨੌਂ ਪੇਆਰ*** ਕਰਨ ਦਾ ਦਾਵਾ ਕਰਦੇ ਹਾਂ ਪਰ ***ਅਪਨੇ ਸੋਬੇ ਨੌਂ*** ਵਸਾਰੀ ***ਬੇਠੇ*** ਹਾਂ। ***ਅਸ*** ਦਾ ਸਭ ਤੋਂ ***ਵਡਾ ਸਬੋਤ ਇਹਾ ਹੇ ਕਹ*** ਭਾਰਤ ਦੇ ਲਗ ਭਗ ***ਬਹਤੇ ਸੋਬੇ ਅਪਨੇ ਅਪਨੇ*** ਸਥਾਪਨਾ ***ਦੋਸ*** ਬੜੇ ***ਅਤਸ਼ਾਹ*** ਤੇ ਜਜ਼ਬੇ ਨਾਲ ***ਮਨਾਈਵਨਦੇ ਹੇਨ***। ***ਅਪਨੀ*** ਜ਼ਬਾਨ, ***ਅਪਨੇ ਸਭੇਅਚਾਰ, ਅਪਨੇ ਪਛੋਕੜ*** ਤੇ ***ਅਪਨੇ ਵਰਸੇ*** ਤੇ ***ਮਾਨ*** ਕਰਦੇ ***ਹੇਨ***। ***ਅਪਨੀ ਕੋਮੀ ਪਛਾਨ*** ਤੇ ***ਮਾਨ*** ਕਰਦੇ ***ਹੇਨ***। ਪਰ ਸਾਡਾ ਬਾਬਾ ਆਦਮ ਹੀ ***ਨਰਾਲਾ ਹੇ।*** ਸਰਕਾਰਾਂ ਤੋਂ ***ਲੇ*** ਕੇ ਆਮ ਲੋਕਾਂ ***ਤਕ ਪਨਜਾਬੀ ਸੋਬੇ*** ਦੇ ***ਬਨਨ ਦਨ*** ਬਾਰੇ ***ਪੋਰੀ ਤਰਹਾਂ ਉ◌ੇਸਲੇ*** ਹੀ ***ਰਹਨਦੇ ਹੇਨ।***
Output-B of proposed system (left to right)
ਇਸ ਗੱਲ ਵਿਚ ਜਦੋਂ ਅਸੀਂ ਬਹੁਤੇ ਪੰਜਾਬੀਆਂ ਨੂੰ ਵੇਖਦੇ ਹਾਂ ਤਾਂ ਪ੍ਰਿੰਸੀਪਲ ਤੇਜਾ ਸਿੰਘ ਦੇ ਲੇਖ ਵਿਚ ਬਿਆਨੀਆਂ ਗਈਆਂ ਕੌੜੀਆਂ ਸਚਾਈਆਂ ਹੋਰ ਵੀ ਸ਼ਿੱਦਤ ਨਾਲ ਮਹਿਸੂਸ ਹੁੰਦੀਆਂ ਹੈਨ। ਅਸੀਂ ਦੇਸ ਨੂੰ ਪਿਆਰ ਕਰਨ ਦਾ ਦਾਵਾ ਕਰਦੇ ਹਾਂ ਪਰ ਆਪਣੇ ਸੂਬੇ ਨੂੰ ***ਵਸਾਰੀ*** ਬੈਠੇ ਹਾਂ। ਇਸ ਦਾ ਸਭ ਤੋਂ ਵੱਡਾ ਸਬੂਤ ਇਹ ਹੈ ਕਿ ਭਾਰਤ ਦੇ ਲਗ ਭਗ ਬਹੁਤੇ ਸੂਬੇ ਆਪਣੇ ਆਪਣੇ ਸਥਾਪਨਾ ***ਦੋਸ*** ਬੜੇ ਉਤਸ਼ਾਹ ਤੇ ਜਜ਼ਬੇ ਨਾਲ ਮਨਾਉਂਦੇ ਹੈਨ। ਆਪਣੀ ਜ਼ਬਾਨ, ਆਪਣੇ ਸਭਿਆਚਾਰ, ਆਪਣੇ ਪਿਛੋਕੜ ਤੇ ਆਪਣੇ ਵਿਰਸੇ ਤੇ ਮਾਣ ਕਰਦੇ ਹੈਨ। ਆਪਣੀ ਕੌਮੀ ਪਛਾਣ ਤੇ ਮਾਣ ਕਰਦੇ ਹੈਨ। ਪਰ ਸਾਡਾ ਬਾਬਾ ਆਦਮ ਹੀ ਨਿਰਾਲਾ ਹੈ। ਸਰਕਾਰਾਂ ਤੋਂ ਲੈ ਕੇ ਆਮ ਲੋਕਾਂ ਤੱਕ ਪੰਜਾਬੀ ਸੂਬੇ ਦੇ ਬਣਨ ਦਿਨ ਬਾਰੇ ਪੂਰੀ ਤਰ੍ਹਾਂ ਅਵੇਸਲੇ ਹੀ ਰਹਿੰਦੇ ਹੈਨ।

Table 10. Comparison of Output-A & B

Output Type	Transliteration Tokens			Accuracy %
	Total	Wrong	Right	
A	116	64	52	44.8275
B	116	02	114	98.2758

Clearly, our system is more practical in nature than PMT and we got good transliteration with different inputs having missing diacritical marks. But we are still having erroneous transliterations by the system. The main source of error is the existence of vowel-consonant mapping between the two scripts as already shown in table 5. In some of the cases the bi-gram approach is not sufficient and we need some other contextual analysis technique. In other cases, system makes errors showing deficiency in handling those tokens which do not belong to common vocabulary domain. These observations point to places where the system can be improved and we hope to study them in the near future.

Acknowledgments. This research work is sponsored by *PAN ASIA ICT R&D Grants Programme* for Asia Pacific http://www.apdip.net and the Beta version of this program is available online at http://s2g.advancedcentrepunjabi.org.We would like to thank *Sajid Chaudhry* for providing us data for Shahmukhi corpus.

References

1. Malik, M. G. A.: Punjabi Machine Transliteration. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL (2006) 1137-1144.
2. Afzal, M., Hussain S.: Urdu Computing Standards: Urdu Zabta Takhti (UZT) 1.01. In proceedings of the IEEE INMIC, Lahore (2001).
3. Haizhou, L., Min, Z., and Jian S.: A Joint Source-Channel Model for Machine Transliteration. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (2004) 159-166.
4. Youngim, J., Donghun, L., Aesun, Y., Hyuk-Chul, K.: Transliteration System for Arabic-Numeral Expressions using Decision Tree for Intelligent Korean TTS, Vol. 1. 30th Annual Conference of IEEE (2004) 657-662.
5. Nasreen Abdululjaleel, leah S. Larkey: Statistical Transliteration for English-Arabic Cross Language Information Retrieval. Proceedings of the 12th international conference on information and knowledge management (2003) 139-146.
6. Yan, Q., Gregory, G., David A. Evans: Automatic Transliteration for Japanese-to-English Text Retrieval. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval (2003) 353-360.
7. Arbabi, M., Fischthal, S. M., Cheng, V. C., and Bart E.: Algorithms for Arabic Name Transliteration. IBM Journal of research and Development (1994) 183-193.
8. Knight, K., and Graehl, J.: Machine Transliteration. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (1997) 128-135.
9. Stalls, B. G. and Kevin K.: Translating Names and Technical Terms in Arabic Text. COLING ACL Workshop on Computational Approaches to Semitic Languages (1998) 34-41.

Vector based Approaches to Semantic Similarity Measures

Juan M. Huerta

IBM T. J. Watson Research Center
1101 Kitchawan Road, Yorktown Heights, NY, 10598
huerta@us.ibm.com

Abstract. This paper describes our approach to developing novel vector based measures of semantic similarity between a pair of sentences or utterances. Measures of this nature are useful not only in evaluating machine translation output, but also in other language understanding and information retrieval applications. We first describe the general family of existing vector based approaches to evaluating semantic similarity and their general properties. We illustrate how this family can be extended by means of discriminatively trained semantic feature weights. Finally, we explore the problem of rephrasing (i.e., addressing the question *is sentence X the rephrase of sentence Y?*) and present a new measure of the semantic linear equivalence between two sentences by means of a modified LSI approach based on the Generalized Singular Value Decomposition.

1 Introduction

Measurements of semantic similarity between a pair of sentences[1] provide a fundamental function in NLU, machine translation, information retrieval and voice based automation tasks, among many other applications. In machine translation, for example, one would like to quantitatively measure the quality of the translation output by measuring the effect that translation had in the conveyed message. In voice based automation tasks, for example in natural language call routing applications, one approach one could take is to compare the uttered input against a collection of canonical or template commands deeming the closest category as the intended target.

Current approaches to semantic similarity measurement include techniques that are specific or custom to the task at hand. For example, in machine translation, the BLEU metric [1] is used in measuring similarity of the MT output. In call routing, vector based methods (e.g., [2, 3]) are used to compare the input utterance against a set of template categories. In information retrieval some approaches use the cosine distance between a query and a document-vector mapped into a lower dimension LSI concept

[1] In this paper, for the sake of conciseness, we use the terms *document*, *utterance*, and *sentence* interchangeably. Typically the nature of the task define the specific type (for example, voice automation systems use *utterances* and so on).

Received 15/10/07
Accepted 07/12/07
Final Version 29/01/08

space ([4]) . Furthermore, IR-inspired methods are currently being applied to novel domains like question answering [9].

In this paper we introduce two novel vector based approaches to semantic similarity (namely, discriminatively trained semantic weights and a Generalized Singular Value Decomposition (GSVD) based approach) based on existing vector based approaches (specifically, LSI, cosine distance, BLEU and discriminative approaches). This paper is organized as follows, we first describe cosine distance, the BLEU metric and discriminative approaches and provide some background related to the need for weight inclusion. We then describe a novel approach to obtaining and applying discriminatively trained semantic weights on the features when computing these basic distances. Finally we introduce a novel method to measure semantic similarity based on a common concept space using the Generalized Singular Value Decomposition. We provide some illustrative examples and a set of preliminary experiments based on a rephrase corpus and on a chat corpus.

2 Vector Based Approaches

In this section we provide a brief overview of some existing vector based approaches to utterance similarity measurement or classification and provide some useful background to these metrics. The approaches described in this section are the building blocks for the novel techniques we introduce in sections 3 and 4.

Existing vector based approaches to document classification and retrieval can be categorized based on 3 criteria: (a) the feature type employed, (b) the weighting or functions applied on the feature counts, and (c) vector distance they use. For example, BLEU typically uses (a) n-gram features, (b) flat weight multipliers are applied at the class levels (i.e., one weight for unigram features, one for bigram features etc) and (c) the distance between two documents is an exponential function of modified precisions of the observed features. Another example is Vector Based call routing [3] which uses (a) n-gram features, (b) discriminatively trained weights in the classification matrix vectors, and normalized occurrence counts for the utterance vector and (c) cosine distance between topic matrix vectors and utterance vector.

2.1 Cosine Distance

The cosine distance is one of the simplest ways of computing similarity between two documents by measuring the normalized projection of one vector over the other. It is defined as follows,

$$\cos\theta = \frac{a \cdot b}{|a||b|}$$

One can compute the similarity between two documents simply by computing the cosine distance between their feature vectors. This approach, while coarse, is widely used in IR tasks and call routing tasks, for example. One of its main advantages is that it is domain and model free.

2.2 BLEU

Bleu [1] is a vector based metric intended to measure the quality of the machine translation output and has the important feature of correlating with human evaluator scores. BLEU is based on modified precision. It is defined below,

$$BLEU = \exp\left(\min(1-\frac{r}{c},0)+\sum_{n=1}^{N} w_n \log p_n\right)$$
$$= \exp(\min(1-r/c,0))p_4^{w_4}p_3^{w3}p_2^{w_2}p_1^{w_1}$$

Where c and r are the lengths of the candidate translation sentence and of the reference, respectively; and p_n denotes the modified precision, which is given by,

$$p_n = \frac{\sum_{C\in\{Candidates\}}\sum_{n-gram\in C} Counts_{clip}(n-gram)}{\sum_{C\in\{Candidates\}}\sum_{n-gram\in C} Counts\ (n-gram)}$$

BLEU is essentially the geometric average of weighted modified precisions multilplied by a non-linear term related to length penalty. One can observe that BLEU and cosine distance essentially could share the same features (i.e., vector based on the same n-grams) and could share the same weights. But how are modified precision related to cosine distance? We address this question in section 2.4 with the objective of gaining further insight.

2.3 Discriminant Approaches

MaxEnt (maximum entropy) approaches are designed to maximize the conditional likelihood [5] and thus belong in the discriminant approaches. The conditional likelihood is given here,

$$P(c_j \mid d,\lambda) = \frac{\exp\sum \lambda_{i,j} f_i(c_j,d)}{\sum_{c_j}\exp\sum \lambda_{i,j} f_i(c_j,d)}$$

While the classification rule is given below. Which can be explained as follows: if *f* is a feature vector the class J is the argument that maximizes the dot product of the Lambda matrix and *f*. In other words, classification in MaxEnt approaches can be seen as a dot product between lambda vectors and feature vectors and thus as a vector distance between observed features and template vectors.

$$\text{class J} = \arg\max_j \frac{\exp(\Lambda f)}{sum(\exp(\Lambda f))} = \arg\max_j \exp(\Lambda^j f) = \arg\max_j \lambda_i^j f_i$$

2.4 Relating Cosine and BLUE: Document Perturbation Perspective

So far, we have described the cosine distance and BLUE score, we additionally have explained that MaxEnt discriminative approaches to document classification can be expressed in terms of a cosine distance between query document and a classification matrix. Now we address the question of given a pair of documents, how much similarity there is between the cosine distance and the BLEU score. For this purpose we assume a document perturbation perspective: we assume a document *a* and a document *b* in which *b* is the *perturbed* version of *a* (i.e., the original document plus a small random variation). We then compute the cosine distance and the modified precision between *a* and *b*.

For the case of the cosine distance, we note that the feature counts a_i are always positive while the perturbation noise is a zero mean random variable, thus the cosine distance can be expressed as:

$$a = \text{document}$$

$$b = a + \varepsilon \text{ (perturbed document)}$$

$$\cos\theta = \frac{a\cdot(a+\varepsilon)}{|a||a+\varepsilon|} = \frac{\sum a_i^2 + \sum a_i\varepsilon_i}{\left(\sum a_i^2\right)^{1/2}\left(\sum (a_i+\varepsilon_i)^2\right)^{1/2}}, \varepsilon \text{ is a zero mean Random Variable}$$

$$\cos\theta = \sqrt{\frac{\left(\sum a_i^2\right)}{\left(\sum a_i^2 + \sum \varepsilon_i^2\right)}}$$

For a large summation (i.e., large document) the sum of the square of alphas and of epsilons will become a Gaussian.

For the modified precisions, we express the clipped frequencies (i.e., frequency counts in the candidate that mach the reference, not exceeding the counts in the reference) as the original counts plus a random variable gamma which has with only positive values.

$$p = \frac{\sum a_i - \sum \gamma_i}{\sum a_i}, \gamma > 0 \text{ but } \sum \gamma_i \text{ is Gaussian for large sumation.}$$

From the equations above we can make two observations. The first observation is that the cosine distance and modified precision, which is the building block of BLEU, are most similar whe the difference between documents is relatively small or when the size of the document is very large (ratio of Gaussian Random Variables). The second observation is related to the independence assumption about the features a_i : the behavior of the perturbation, more clearly in the Cosine Distance case, has terms that average out to zero the more the a_i features are truly *Independent and Identically Distributed* (IID); in other words these two metrics blur out by averaging positive and negative perturbations. In reality, natural word frequencies tend to affect this assumption (i.e., not strictly IID), but naturally giving more weight to frequent words and this might not necessarily be a desirable characteristic since it will bias the metric towards frequent features. To minimize this, it is very important to emphasize meaningful words, if what we are interested in is in measuring similarity in the meaning of the utterances. In the next section we propose a method to address this.

3 Toward Discriminant Semantic Weights

In the previous chapter we have provided some background on Cosine Distance, BLEU and discriminative approaches. We illustrated the importance of introducing weights that emphasize semantic relevance and at the same time counterbalance the bias introduced by natural word frequencies. The question we address now is how to identify those words and how to train these weights and integrate them with the metric? We now describe a novel way to retrofit BLEU to incorporate discriminatively trained semantic weights.

In section 2.3 we described the criteria used in MaxEnt and noted that that if the set of weights used for a given feature across the set of classes has little *variance* (i.e., dynamic range), then the contribution of such feature to the overall classification is small. This is precisely what we will use to identify the semantic importance of a feature. In other words we focus on the dynamic range of such feature weight set,

$$w_{f_j} = \max(\lambda_{f_j,i}) - \max(\lambda_{f_j,i})$$

These weights tells us how much contribution to discrimination the feature provide and is always equal or larger than zero. The classification is done across semantic groupings, or classes thus these weights denote semantic-class discriminant importance. Thus we could adjust the BLEU metric to include these weights by making ,

$$\hat{p}_n = \frac{\sum_{C\in\{Candidates\}} \sum_{n-gram\in C} w_{n-gram} Counts_{clip}(n-gram)}{\sum_{C\in\{Candidates\}} \sum_{n-gram\in C} w_{n-gram} Counts\ (n-gram)}$$

In the case of cosine distance the weights are based on weighted versions of *a* and *b* and *W* is the diagonal matrix with the semantic weights on the diagonal.

$$Semantic \cos\theta = \frac{(W^{1/2}a)\cdot(W^{1/2}b)}{\left|W^{1/2}a\right|\left|W^{1/2}b\right|}$$

Thus, our approach utilizes a labeled corpus in which semantic categories are indicated for each utterance. A discriminant criterion is then employed to obtain the matrix Λ from which the weight dynamic ranges are computed in order to obtain the semantic weights, which are used in BLEU and Cosine Distance to weight the feature perturbations accordingly.

While weighted versions of BLEU and the cosine distance have been previously proposed, (e.g., [6]) these approaches have not focused on discriminatively trained semantic weights.

4 Shared Concept Space Approach

We now propose a novel approach to measure utterance similarity in a pair of utterances when one utterance might be a rephrase of the other. We pose the problem as a classification problem: we assume the existence of a parallel corpus in which each sentence in set A has a rephrased version in the set B. While simple cosine distance between utterances in the set A and candidates in the set B can be used in this rephrase classification task, approaches that provide more robustness to noise and sparse input are desirable. The family of approaches to document similarity and document query similarity based on SVD in general and the Latent Semantic Analysis [4] in particular, are commonly used in information retrieval and document similarity evaluation (e.g., [7, 10]) and provide desirable features like the ones described. In LSI the document term matrix A is factored in the following way:

$$A = U\Sigma V^T$$

A query and a document can then be represented in a concept space (using *k* singular values) as shown below, the cosine distance in this concept space is used to compute the similarity between query and document or between documents.

$$\hat{q} = \Sigma_k^{-1} U_k^T q = \Lambda_k q$$

Our approach assumes, as we mentioned at the beginning of this section, the existence of two parallel document term matrices A and B. These two document-term matrices have the same number of columns (because they are parallel), but the number of rows (the lexicons each use) need not be the same. We could perform SVD on each matrix separately. However, we now describe a novel method based on the Generalized Singular Value decomposition [8] that attains decompositions of the matrices that share the same concept space.

4.1 Generalized Singular Value Decomposition

The Generalized Singular Value decomposition of matrices A and B, decomposes these matrices as follows,

$$A = UCX^T$$

$$B = VSX^T$$

Following an LSI interpretation, U and V represent the term to concept mapping matrices and X represents the document to concept matrix. Having matrix X shared by A and B achieves the desired tying of concept spaces.

The way to map documents or queries to concept space is then,

$$(C_k{}^T C_k)^{-1} C_k{}^T U^T A = X^T$$

$$(S_k{}^T S_k)^{-1} S_k{}^T V^T B = X^T$$

In the next section we present some illustrative examples and preliminary experiments of the concepts and techniques we have discussed so far.

5 Experiments

We describe now two sets of experiments we performed. The first one illustrates the discriminative semantic weights using a IT Help Desk corpus, while the second experiment is related to rephrase detection/classification using the GSVD approach.

5.1 Discriminant Weights Experiment

To illustrate the Discriminative Weight approach we used a corpus of text chats between IT users and Help Desk agents. The users and the agents communicate via instant messaging to troubleshoot the user's IT problems. The corpus consist of over 3000 chats. The semantic categories used in the discriminative training correspond the broad product category discussed (e.g., email, network connectivity, web applications, telephony, mainframe, etc.). We used Maximum Entropy training to obtain the classification matrix. Our features consisted uniquely on unigrams. Table 1 below show the outcome of the computation of the words with the highest dynamic range and the words with the lowest dynamic range. Thus, words that carry a substantial amount of semantic importance, in terms of message, are assigned high weights. In general these words describe products and items that can affect the classified semantic category substantially. On the other hand, the words with low dynamic range (and thus low weight in semantically weighted BLEU and cosine distance) are typically verbs which by themselves carry little discriminant semantic power and thus are less important, in this task, in terms of power to convey a message and affect a classification output.

Table 1. Words with highest dynamic range (left column) and lowest dynamic range (right column)

HIGH WEIGHT	LOW WEIGHT
GSA 2.860040	RULE 0.016409
MAIL 3.472591	KICKING 0.017700
BLUEPAGES 3.162921	SWITCHING 0.021317
MANAGENOW 2.502134	CONTAINED 0.013456
PRINTER 2.662830	WORTH 0.012854
EMAILS 3.210260	CHOOSING 0.013268
P/W 1.991220	MONITORING 0.010465
(LOTUS) NOTES 2.402410	
QUICKPLACE 2.626500	
DATABASE 2.775500	
TSM 2.148356	
AT&T 2.648001	

For illustration purposes, we now show a couple of synthetic examples in which BLEU and semantically weighted BLEU scores are compared depending on two sentences. The phrases used are made up and are not part of the corpus (nor representative of the corpus). Table 2 below shows the original sentence and the result of a translation and a roundtrip translation. The output of the back-translation is compared against the original and the scores are computed. We can see that when errors are introduced in important features (e.g., *wireless*) semantic bleu produces a lower score compared to BLEU. Conversely, when errors are introduced in non-important features (e.g., deletion) the score is higher than BLEU. Thus as intended, relative to BLEU, the semantic weighted BLEU produces a score that is more sensitive to perturbation if the perturbation is important, and less if the perturbation is unimportant.

Table 2. Sample phrases, original and perturbed versions, with their BLEU and semantically weighted BLEU scores for two cases.

BLEU > Sem. BLEU	BLEU < Sem. BLEU
Original: Please reset *wireless* connectivity **Perturbed:** That restores connectivity *without threads* please BLEU: 0.323 Sem. BLEU: 0.315	**Original:** Recent calendar deletion **Perturbed:** Recent calendar suppression BLEU: 0.757 Sem. BLEU: 0.792

5.2 Rephrasing Experiments

To conduct our rephrase experiments, we took 2 parallel versions of a portion of the Bible. Specifically we took the book of Proverbs. The structure of the verses in book of Proverbs (915 in total) is relatively simple. Because of the careful attention paid in keeping the message in every translation of the Bible, and because of the size and simplicity in the sentence structure of the verses of the Book of Proverbs, it presents an ideal corpus for preliminary experiment in rephrase analysis. The versions used are the New King James version and the New Life[2] version. The lexicon sizes are: 1956 words for the NKJ version, 1213 for NL, and 2319 for both version. The NKJ version has 14822 tokens while NL version has 17045 tokens. The NL version uses less unique words but it is longer, while NKJ has a bigger vocabulary while being shorter.

The first experiment that we conducted on this corpus is to measure self and cross utterance similarity across versions using the cosine distance. Figure 1 above shows the distribution of the cosine distance value between a sentence and all the other sentences in the other version (cross similarity, left panel) and on the right shows the self similarity which is the distribution of cosine scores between a sentence and its counterpart in the other version. As we can see, from the *detection* point of view, the task is relatively simple, as most of the cross similarity scores lie below 0.5 and most of the self similarity scores lie above 0.5. In terms of *classification* we performed the following experiment: we computed the cosine distance between each verse and the whole other version. If the highest cosine score belongs to the corresponding

[2] Scripture taken from the New King James Version. Copyright © 1982 by Thomas Nelson, Inc. Used by permission. All rights reserved

utterance in the other corpus we count one correct classification (and an incorrect otherwise). We did this both ways (classifying version A first and then classifying version B).The results are shown below in table 3. As we can see, about 80% accuracy is achieved with cosine distance.

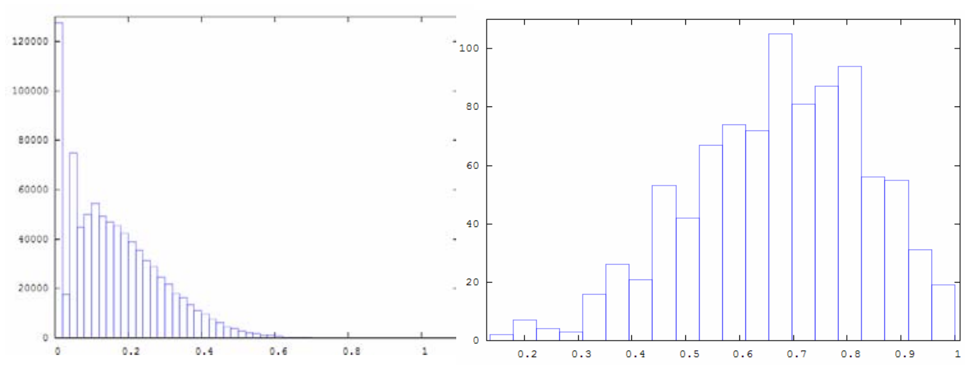


Fig. 1. Cross similarity (left) and self similarity (right) Cosine distance measure distribution.

Table 3. Classification results using the Cosine Distance

	Accuracy
A*b	80%
B*a	78%

We also performed GSVD experiments. We constructed the document-term matrices for each of the translation versions A and B. We then computed the generalized singular value decomposition described in section 4. Using that representation we mapped each sentence to a concept space using several rank values k. Then we repeated the cosine distance experiment described above with the sentences mapped to this concept space. The results are shown in figure 2 below. Interestingly an accuracy of about 99% is obtained with k as low as 26. And with k equal to 16 we get an accuracy comparable to plain cosine distance.

While these results are remarkable, a substantial accuracy improvement over cosine distance, one has to be careful to note that the SVD is performed on the whole corpus. Because of its very small size, it is impractical to perform a breakdown of the corpus into training and testing components and thus an experiment with a much larger corpus is needed. Furthermore, as we mentioned, the sentence constructions in the book of Proverbs are quite simple and thus the linear mappings captured in the GSVD are good enough to attain such high accuracy.

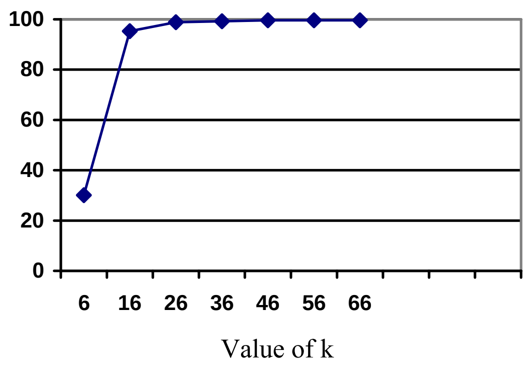


Fig. 2. Classification accuracy in the rephrase task as a function of rank *k*.

6 Discussion

In this paper we reviewed several important vector based approaches to computing similarity between two sentences, documents or utterances: cosine distance, BLEU, discriminative vector based methods, and SVD based methods (like LSI). Besides describing their basic characteristics, in this paper, we pointed out that semantic weighting is important to avoid the undesirable averaging out effect observed in perturbation analysis. We proposed a method to discriminatively train these weights and described ways to incorporate these weights into cosine distance and BLEU. Through some illustrative examples we saw that when contrasting semantically weighted BLEU with BLEU we can provide an improved guidance the semantic distance.

We also analyzed SVD approaches and proposed a novel utilization of the generalized singular value decomposition to vector based computation of semantic similarity. We concluded the paper with a series of illustrative examples and preliminary experiments.

We observed that in our preliminary experiments the basic cosine distance had a classification accuracy of 80% in the rephrase task, while the GSVD based approach performed at around 99%. This result, while very interesting, might be due mostly to linear mappings between rephrases (i.e., the use of features that can be substituted by other features, like synonyms) which might be due to the nature of the simple structure formations and almost deterministic mappings of the sentences conforming the corpus. We pointed out in our experiment section that, in the case of rephrase analysis, it is important to conduct further experiments on corpora that presents larger sentence construction variability and that is large enough to cover a larger lexicon. While the preliminary results we obtained seemed very promising, it will be of much better value to test this approach on a very large set.

In terms of future work and future approaches, we suggest the exploration of directions that are probabilistic in nature. In this paper we proposed extensions to BLEU, SVD-LSI, and cosine distance. These approaches are not probabilistic in nature. Dis criminative methods, and Exponential Models-Maximum Entropy approaches are the only type of probabilistic approaches used in this paper. The authors consider that the methods introduced in this paper would benefit in terms of robustess from being extended into a probabilistic formulation.

References

1. Papineni, K., Roukos, S., Ward, T., and Zhu, W. 2001. BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association For Computational Linguistics (Philadelphia, Pennsylvania, July 07 - 12, 2002). Annual Meeting of the ACL. Association for Computational Linguistics, Morristown, NJ, 311-318.
2. Chu-Carroll, J, and R. Carpenter (1999), Vector-Based Natural Language Call Routing. Journal of Computational Linguistics, 25(30), pp. 361-388, 1999
3. H.-K.J. Kuo,; Chin-Hui Lee Discriminative training of natural language call routers Speech and Audio Processing, IEEE Transactions on Volume 11, Issue 1, Jan 2003 Page(s): 24 - 35.
4. Thomas Landauer, P. W. Foltz, & D. Laham (1998). "Introduction to Latent Semantic Analysis". Discourse Processes 25: 259-284.
5. Berger, A. L., Pietra, V. J., and Pietra, S. A. 1996. A maximum entropy approach to natural language processing. Comput. Linguist. 22, 1 (Mar. 1996), 39-71.
6. J.M. Schultz and M. Liberman, "Topic Detection and Tracking using idfWeighted Cosine Coefficient," Proceedings of the DARPA Broadcast News Workshop, 189-192, 1999.
7. Dasgupta, A., Kumar, R., Raghavan, P., and Tomkins, A. 2005. Variable latent semantic indexing. In Proceeding of the Eleventh ACM SIGKDD international Conference on Knowledge Discovery in Data Mining (Chicago, Illinois, USA, August 21 - 24, 2005). KDD '05. ACM Press, New York, NY, 13-21.
8. Gene Golub, and Charles Van Loan, Matrix Computations, Third Edition, Johns Hopkins University Press, Baltimore, 1996,
9. Gregory Marton and Boris Katz Using Semantic Overlap Scoring in Answering TREC Relationship Questions, Proceedings of LREC 2006, Genoa, Italy; May, 2006
10. Evgeniy Gabrilovich and Shaul Markovitch Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India, January 2007

Information Retrieval and Question Answering

Comparing and Combining Methods for Automatic Query Expansion [⋆]

José R. Pérez-Agüera[1] and Lourdes Araujo[2]
jose.aguera@fdi.ucm.es, lurdes@lsi.uned.es

[1]Dpto. de Ingeniería del Software e Inteligencia Artificial, UCM, Madrid 28040, Spain,
[2]Dpto. Lenguajes y Sistemas Informáticos. UNED, Madrid 28040, Spain,

Abstract. Query expansion is a well known method to improve the performance of information retrieval systems. In this work we have tested different approaches to extract the candidate query terms from the top ranked documents returned by the first-pass retrieval. One of them is the cooccurrence approach, based on measures of cooccurrence of the candidate and the query terms in the retrieved documents. The other one, the probabilistic approach, is based on the probability distribution of terms in the collection and in the top ranked set. We compare the retrieval improvement achieved by expanding the query with terms obtained with different methods belonging to both approaches. Besides, we have developed a naïve combination of both kinds of method, with which we have obtained results that improve those obtained with any of them separately. This result confirms that the information provided by each approach is of a different nature and, therefore, can be used in a combined manner.

1 Introduction

Reformulation of the user queries is a common technique in information retrieval to cover the gap between the original user query and his need of information. The most used technique for query reformulation is query expansion, where the original user query is expanded with new terms extracted from different sources. Queries submitted by users are usually very short and query expansion can complete the information need of the users.

A very complete review on the classical techniques of query expansion was done by Efthimiadis [5]. The main problem of query expansion is that in some cases the expansion process worsens the query performance. Improving the robustness of query expansion has been the goal of many researchers in the last years, and most proposed approaches use external collections [17, 16, 15], such as the Web documents, to extract candidate terms for the expansion. There are other methods to extract the candidate terms from the same collection that

⋆ Supported by project TIC2007-67581-C02-01/ and Dirección General de Universidades e Investigación de la Consejería de Educación de la Comunidad de Madrid and Universidad Complutense de Madrid (Grupo de investigación consolidado 910494)

Received 15/10/07
Accepted 07/12/07
Final Version 17/01/08

the search is performed on. Some of these methods are based on global analysis where the list of candidate terms is generated from the whole collection, but they are computationally very expensive and its effectiveness is not better than that of methods based on local analysis [11, 6, 14]. We also use the same collection that the search is performed on, but applying local query expansion, also known as pseudo-feedback or blind feedback, which does not use the global collection or external sources for the expansion. This approach was first proposed by Xu and Croft [18] and extracts the expansion terms from the documents retrieved for the original user query in a first pass retrieval.

In this work we have tested different approaches to extract the candidate terms from the top ranked documents returned by the first-pass retrieval. After the term extraction step, the query expansion process requires a further step, namely to re-compute the weights of the query terms that will be used in the search process. We have also tested different methods for this step.

There exist two main approaches to rank the terms extracted from the retrieval documents. One of them is the cooccurrence approach, based on measures of cooccurrence of the candidate and the query terms in the retrieved documents. The other one is the probabilistic approach, which is based on the differences between the probability distribution of terms in the collection and in the top ranked set. In this paper we are interested in evaluating the different techniques existing to generate the candidate term list. Our thesis is that the information obtained with the cooccurrence methods is different from the information obtained with probabilistic methods and these two kinds of information can be combined to improve the performance of the query expansion process. Accordingly, our goal has been to compare the performance of the cooccurrence approach and the probabilistic techniques and to study the way of combining them so as to improve the query expansion process. We present the results of combining different methods for the term extraction and the reweighting steps.

Two important parameters have to be adjusted for the described process. One of them is the number of documents retrieved in the first pass to be used for the term extraction. The other one is the number of candidate terms that are finally used to expand the original user query. We have performed experiments to set both of them to its optimal value in each considered method in our configuration.

The rest of the paper proceeds as follows: sections 2 and 3 describe the cooccurrence and probabilistic approaches, respectively; section 4 presents our proposal to combine both approaches; section 5 describes the different reweighting methods considered to assign new weights to the query terms after the expansion process; section 6 is devoted to show the experiments performed to evaluate the different expansion techniques separately and combined and section 7 summarizes the main conclusions of this work.

2 Cooccurrence Methods

The methods based on term cooccurrence have been used since the 70's to identify some of the semantic relationships that exist among terms. In the first works of Keith Van Rijsbergen [12] we find the idea of using cooccurrence statistics to detect some kind of semantic similarity between terms and exploiting it to expand the user's queries. In fact, this idea is based on the Association Hypothesis:

If an index term is good at discriminating relevant from non-relevant documents then any closely associated index term is likely to be good at this.

The main problem with the cooccurrence approach was mentioned by Peat and Willet [8] who claim that similar terms identified by cooccurrence tend to occur also very frequently in the collection and therefore these terms are not good elements to discriminate between relevant and non-relevant documents. This is true when the cooccurrence analysis is done on the whole collection but if we apply it only on the top ranked documents discrimination does occur.

For our experiments we have used the well-know Tanimoto, Dice and Cosine coefficients:

$$\mathrm{Tanimoto}(\mathrm{t_i}, \mathrm{t_j}) = \frac{\mathrm{c_{ij}}}{\mathrm{c_i} + \mathrm{c_j} - \mathrm{c_{ij}}} \tag{1}$$

$$\mathrm{Dice}(\mathrm{t_i}, \mathrm{t_j}) = \frac{2\mathrm{c_{ij}}}{\mathrm{c_i} + \mathrm{c_j}} \tag{2}$$

$$\mathrm{Cosine}(\mathrm{t_i}, \mathrm{t_j}) = \frac{\mathrm{c_{ij}}}{\sqrt{\mathrm{c_i c_j}}} \tag{3}$$

where c_i and c_j are the number of documents in which terms t_i and t_j occur, respectively, and $c_{i,j}$ is the number of documents in which t_i and t_j cooccur.

We apply these coefficients to measure the similarity between terms represented by the vectors. The result is a ranking of candidate terms where the most useful terms for expansion are at the top.

In the selection method the most likely terms are selected using the equation

$$\mathrm{rel}(\mathrm{q}, \mathrm{t_e}) = \sum_{\mathrm{t_i} \in \mathrm{q}} \mathrm{q_i} \mathrm{CC}(\mathrm{t_i}, \mathrm{t_e}) \tag{4}$$

where CC is one of the cooccurrence coefficients: Tanimoto, Dice, or Cosine. Equation 4 boosted the terms related with more terms of the original query.

The results obtained with each of these measures, presented in section 6, show that Tanimoto performs better.

3 Distribution Analysis Approaches

One of the main approaches to query expansion is based on studying the difference between the term distribution in the whole collection and in the subsets of documents that can be relevant for the query. One would expect that terms with little informative content have a similar distribution in any document of the collection. On the contrary, terms closely related to those of the original query are expected to be more frequent in the top ranked set of documents retrieved with the original query than in other subsets of the collection.

3.1 Information-Theoretic Approach

One of the most interesting approaches based on term distribution analysis has been proposed by C. Carpineto et. al. [3], and uses the concept the Kullback-Liebler Divergence [4] to compute the divergence between the probability distributions of terms in the whole collection and in the top ranked documents obtained for a first pass retrieval using the original user query. The most likely terms to expand the query are those with a high probability in the top ranked set and low probability in the whole collection. For the term t this divergence is:

$$KLD_{(PR,PC)}(t) = P_R(t)\log\frac{P_R(t)}{P_C(t)} \tag{5}$$

where $P_R(t)$ is the probability of the term t in the top ranked documents, and $P_C(t)$ is the probability of the term t in the whole collection.

3.2 Divergence from Randomness Term Weighting Model

The Divergence From Randomness (DFR) [2] term weighting model infers the informativeness of a term by the divergence between its distribution in the top-ranked documents and a random distribution. The most effective DFR term weighting model is the *Bo1 model* that uses the Bose-Einstein statistics [10, 7]:

$$w(t) = \mathrm{tf}_x \log_2(\frac{1+P_n}{P_n}) + \log(1+P_n) \tag{6}$$

where tf_x is the frequency of the query term in the x top-ranked documents and P_n is given by $\frac{F}{N}$, where F is the frequency of the query term in the collection and N is the number of documents in the collection.

4 Combined Query Expansion Method

The two approaches tested in this work can complement each other because they rely on different information. The performance of the cooccurrence approach is reduced by words which are not stop-words but are very frequent in the collection [8]. Those words, which represent a kind of noise, can reach a high position in the term index, thus worsening the expansion process. However, precisely because of their high probability in any set of the document collection, these words tend to have a low score in KLD or Bo1. Accordingly, combining the cooccurrence measures with others based on the informative content of the terms, such as KLD or Bo1, helps to eliminate the noisy terms, thus improving the retrieved information with the query expansion process.

Our combined model amounts to applying both, a coocurrence method and a distributional method and then obtaining the list of candidate terms by intersecting the lists provided by each method separately. Finally, the terms of the resulting list are assigned a new weight by one of the reweighting method considered.

In the combined approach the number of selected terms depends of the overlapping between the term sets proposed by both approaches. To increase the intersection area and obtain enough candidate terms in the combined list it is necessary to increase the number of selected terms for the non-combined approaches. This issue has been studied in the experiments.

5 Methods for Reweighting the Expanded Query Terms

After the list of candidate terms has been generated by one of the methods described above, the selected terms which will be added to the query must be re-weighted. Different schemas have been proposed for this task. We have compared these schemas and tested which is the most appropriate for each expansion method and for our combined query expansion method.

The classical approach to term re-weighting is the Rocchio algorithm [13]. In this work we have used Rocchio's beta formula, which requires only the β parameter, and computes the new weight *qtw* of the term in the query as:

$$\mathrm{qtw} = \frac{\mathrm{qtf}}{\mathrm{qtf}_{\mathrm{max}}} + \beta \frac{\mathrm{w(t)}}{\mathrm{w}_{\mathrm{max}}\mathrm{(t)}} \tag{7}$$

where $w(t)$ is the old weight of term t, $w_{\mathrm{max}}(t)$ is the maximum $w(t)$ of the expanded query terms, β is a parameter, qtf is the frequency of the term t in the query and $\mathrm{qtf}_{\mathrm{max}}$ is the maximum term frequency in the query q. In all our experiments, β is set to 0.1.

We have also tested other reweighting schemes, each of which directly comes from one of the proposed methods for the candidate term selection. These schemes use the ranking values obtained by applying the function defined through each method. Each of them can only be applied to reweight terms selected with the method it derives from. This is because these methods require data, collected during the selection process, which are specific of each of them.

For the case of the reweighting scheme derived from KLD, the new weight is directly obtained applying KLD to the candidate terms. Terms belonging to the original query maintain their value [3].

For the scheme derived from the cooccurrence method, that we called *SumCC*, the weights of the candidate terms are computed by:

$$\mathrm{qtw} = \frac{\mathrm{rel(q, t_e)}}{\sum_{\mathrm{t_i \in q}} \mathrm{q_i}} \tag{8}$$

where $\sum_{t_i \in q} q_i$ is the sum of the weights of the original terms [19].

Finally, for the reweighting scheme derived from the Bose-Einstein statistics, a normalization of Bo1 that we call *BoNorm*, we have defined a simple function based in the normalization of the values obtained by Bose-Einstein computation:

$$\mathrm{qtw} = \frac{\mathrm{Bo(t)}}{\sum_{\mathrm{t \in cl}} \mathrm{Bo(t)}} \tag{9}$$

where Bo(t) is the Bose-Einstein value for the term t, and the sum runs on all terms included in the candidate list obtained applying Bose-Einstein statistics.

6 Experiments

We have used the Vector Space Model implementation provided by Lucene[1] to build our information retrieval system. Stemming and stopword removing has been applied in indexing and expansion process. Evaluation is carried out on the Spanish EFE94 corpus, which is part of the CLEF collection [9] (approximately 215K documents of 330 average word length and 352K unique index terms) and on the 2001 Spanish topic set, with 100 topics corresponding to 2001 and 2002 years, of which we only used the title (of 3.3 average word length). Nevertheless, there is evidence in the literature [1, 3] that some of the presented methods are also valid for other languages (English, French, Italian and Spanish).

We have used different measures to evaluate each method. Each of them provides a different estimation of the precision of the retrieved documents, which is the main parameter to optimize when doing query expansion, since recall is always improved by the query expansion process. The measures considered have been MAP[2] *(Mean Average Precision)*, GMAP[3], Precision@X[4], R-Precision[5].

First of all we have tested the different cooccurrence methods described above. Table 1 shows the results obtained for the different measures considered in this work. We can observe that Tanimoto provides the best results for all the measures, except for P@10, but in this case the difference with the result of Dice, which is the best, is very small. According to the results we have selected the Tanimoto similarity function as coocurrence method for the rest of the work.

Table 1. Comparing different cooccurrence methods. The Baseline row corresponds to the results of the query without expansion. P@5 stands for precision after the first five documents retrieved, P@10 after the first ten, and R-PREC stands for R-precision. Best results appear in boldface.

	MAP	GMAP	R-PREC	P@5	P@10
Baseline	0.4006	0.1941	0.4044	0.5340	0.4670
Cosine	0.4698	0.2375	0.4530	0.6020	0.5510
Tanimoto	**0.4831**	**0.2464**	**0.4623**	**0.6060**	0.5520
Dice	0.4772	0.2447	0.4583	0.6020	**0.5530**

[1] http://lucene.apache.org
[2] the average of the precision value (percent of retrieved documents that are relevant) obtained for the top set documents existing after each relevant document is retrieved.
[3] a variant of MAP, that uses a geometric mean rather than an arithmetic mean to average individual topic results.
[4] precision after X documents (whether relevant or non-relevant) have been retrieved.
[5] measures precision after R documents have been retrieved, where R is the total number of relevant documents for a query.

6.1 Selecting the Reweighting Method

The next set of experiments have had the goal of determining the most appropriate reweighting method for each candidate term selection method. Table 2 shows the results of different reweighting methods (Rocchio and SumCC) applied after selecting the candidate terms by the cooccurrence method. We can observe that the results are quite similar for both reweighting methods, though Rocchio is slightly better.

Table 2. Comparing different reweighting methods for cooccurrence. *CooRocchio* corresponds to using cooccurrence as selection terms method and Rocchio as reweighting method. *CooSumCC* corresponds to using cooccurrence as selection terms method and *SumCC* as reweighting method. Best results appear in boldface.

	MAP	GMAP	R-PREC	P@5	P@10
Baseline	0.4006	0.1941	0.4044	0.5340	0.4670
CooRocchio	**0.4831**	**0.2464**	0.4623	0.6060	**0.5520**
CooSumCC	0.4798	0.2386	**0.4628**	**0.6080**	0.5490

Table 3. Comparing different reweighting methods for KLD. *KLDRocchio* corresponds to using KLD as selection terms method and Rocchio as reweighting method. *KLDkld* corresponds to using KLD as selection terms method and *kld* as reweighting method. Best results appear in boldface.

	MAP	GMAP	R-PREC	P@5	P@10
Baseline	0.4006	0.1941	0.4044	0.5340	0.4670
KLDRocchio	0.4788	0.2370	0.4450	0.5960	0.5480
KLDkld	**0.4801**	**0.2376**	**0.4526**	**0.6080**	**0.5510**

Table 3 shows the results of different reweighting methods (Rocchio and kld) applied after selecting the candidate terms with KLD. The best results are obtained using kld as reweighting method.

Table 4 shows the results of different reweighting methods (Rocchio and BoNorm) applied after selecting the candidate terms with Bo1. In this case, the best results are obtained using BoNorm as reweighting method.

The results of this section show that the best reweighting method after selecting terms by cooccurrence is Rocchio, while for the distributional methods in the term selection process, the best reweighting is obtained with the method derived from themselves, though Rocchio also provides results very close to the best one.

Table 4. Comparing different reweighting methods for Bo1 *BoRocchio* corresponds to using Bo1 as selection terms method and Rocchio as reweighting method. *BoBoNorm* corresponds to using Bo1 as selection terms method and *BoNorm* as reweighting method. Best results appear in boldface.

	MAP	GMAP	R-PREC	P@5	P@10
Baseline	0.4006	0.1941	0.4044	0.5340	0.4670
BoRocchio	0.4765	0.2381	0.4450	0.5880	0.5450
BoBoNorm	**0.4778**	**0.2388**	**0.4470**	**0.5960**	**0.5470**

6.2 Parameter Study

We have studied two parameters that are fundamental in query expansion, the number of candidate terms to expand the query and the number of documents from the top ranked set used to extract the candidate terms. The optimal value of these parameters can be different for each method, and thus we have studied them for each case. The reweighting used for each method has been the one that provides de best results, and Rocchio for the combined approach.

Figure 1 shows, for the different expansion methods considered, the MAP and R-PREC measures with different numbers of candidate terms to expand the original query. We can observe that the results of both measures, MAP and R-PREC, indicate similar values, and that this value is different for each considered method: around 25 terms for the cooccurrence method, 40 terms for Bose-Einstein statistics and Kullback-Liebler divergence and 75 terms for our combined approach. The combined approach requires a larger number of selected terms from each basic approach in order to have enough expansion terms in the intersection list.

Figure 2 shows, for the different expansion methods considered, the MAP and R-PREC measures with different numbers of documents used to extract the set of candidate query terms. We can observe that in all cases the best value is around 10 documents.

6.3 Comparing and Combining Both Approaches

The next step of our experiments has been comparing the overall retrieval performance of the different expansion method considered, including our combined approach. The reweighting used for each method has been the one that provides de best results, and Rocchio for the combined approach. Table 5 shows MAP and GMAP measures, while table 6 shows R-precision, precision after 5 documents retrieved (P@5) and after 10 documents (P@10). We can observe that for nearly every measure (except for P@5) the best results are obtained by the combination of the Bo1 model with cooccurrence. The next best result is provided by the other combination considered, KLD with cooccurrence. These results prove that the information provided by methods belonging to different approaches, cooccurrence and distributional analysis, is different and thus its combination improves the results obtained by any of them separately.

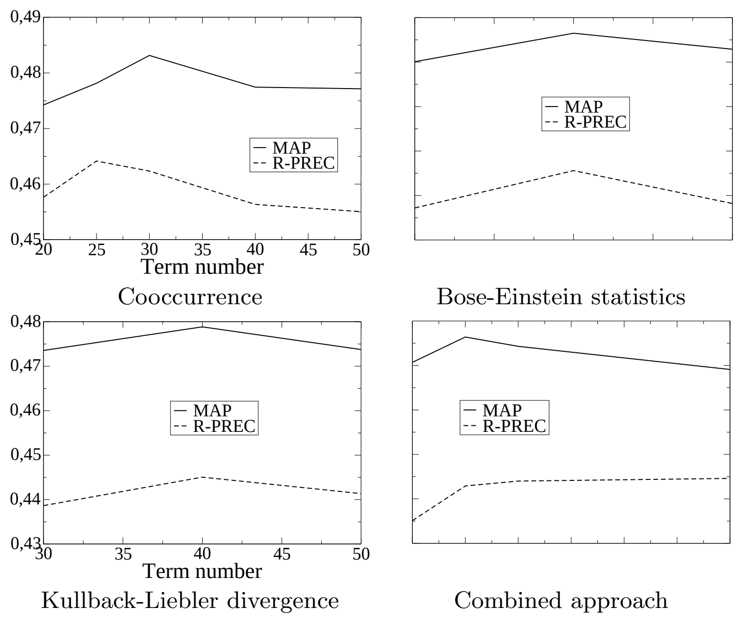


Fig. 1. Study of the best number of candidate terms to expand the original query with the different considered methods. R-PREC stands for R-Precision.

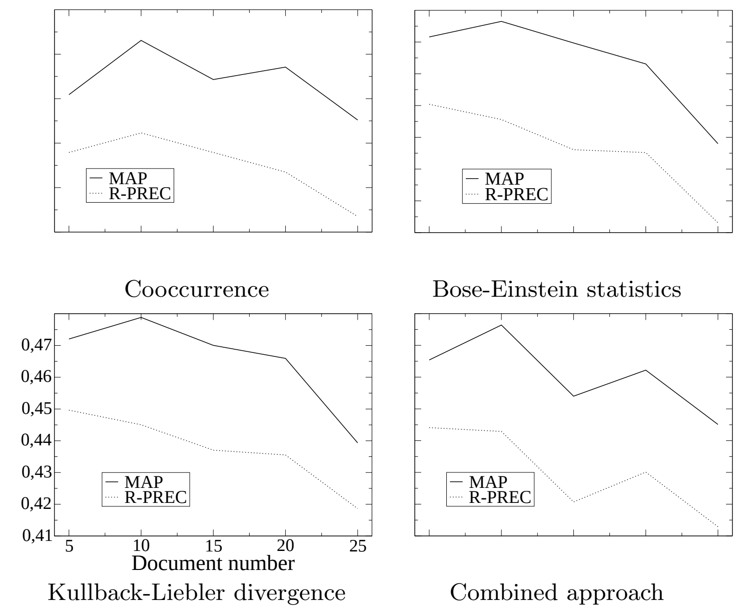


Fig. 2. Study of the best number of documents used to extract the set of candidate query terms. R-PREC stands for R-Precision.

Table 5. Comparing MAP and GMAP for different methods considered for query expansion.

	MAP	GMAP
Baseline	0.4006(-)	0.1941(-)
KLD	0.4801(+16.55%)	0.2376(+18.30%)
Bo	0.4778(+16.15%)	0.2388(+18.71%)
Cooccurrence	0.4831(+17.07%)	0.2464(+21.22%)
BoCo	**0.4964(+19.29%)**	**0.2570(+24.47%)**
KLDCo	0.4944(+18.97%)	0.2483(+21.82%)

Table 6. Comparing R-Precision (R-PREC), precision after 5 documents retrieved (P@5) and after 10 documents retrieved (P@10) for different methods considered for query expansion.

	R-PREC	P@5	P@10
Baseline	0.4044(-)	0.5340(-)	0.4670(-)
KLD	0.4526(+10.64%)	0.6080(+12.17%)	0.5510(+15.24%)
Bo	0.4470(+9.53%)	0.5960(+10.40%)	0.5470(+14.62%)
Cooccurrence	0.4623(+12.5%)	0.6060(+11.88%)	0.5520(+15.39%)
BoCo	**0.4629(+12.63%)**	0.6220(+14.14%)	**0.5630(+17.05%)**
KLDCo	0.4597(+12.02%)	**0.6240(+14.42%)**	0.5600(+16.60%)

6.4 Analysis of the Results

We have analyzed the results for some specific queries of our test set. Table 8 compares the MAP measure obtained for cooccurrence, Bo1 and the combination for the test set queries shown in table 7. We can observe that the best result in each case is provided by a different method, thus showing that these methods provided different information. We can also observe that the combined model not always provides the best result. This suggests to investigate other combination schemes.

Table 7. Queries used to study the performances of each method in particular cases.

C041	Pesticidas en alimentos para bebés
C049	Caída de las exportaciones de coches en Japón
C053	Genes y enfermedades
C055	Iniciativa Suiza para los Alpes
C058	Eutanasia
C122	Industria norteamericana del automóvil

These results are compatible with the previous observations of improvement with the combined approaches because we can observe that BoCo always im-

Table 8. Results of the MAP measure for the queries 41, 49, 53, 55, 58, 122. BoCo stands for the model combining Bo1 and cooccurrence. The best value appears in boldface.

Measure	41	49	53	55	58	122
Baseline	0.62	0.1273	0.3098	0.2334	0.8417	0.0760
Cooccurrence	0.9428	0.2775	**0.4901**	0.6447	0.8960	0.0588
Bo1	0.9270	**0.3594**	0.4005	0.5613	**0.9329**	0.1130
BoCo	**0.9630**	0.3072	0.4724	**0.6588**	0.9223	**0.1252**

proves some of the non-combined methods. This is an indication of the higher robustness of the combined approach. Nevertheless, it will be interesting to analysis the kind of queries more appropriate for each approach.

7 Conclusions and Future Works

We have presented a study of two different approaches, cooccurrence and distributional analysis, for query expansion. For each approach we have considered several models. Results have shown that the query expansion methodology that we apply is very robust and improves the retrieval results of the original query with all tested approaches.

The analysis of the results indicates that the statistical information exploited in each considered approach is different, and this suggests combining them to improve the results.

We have carried out experiments to measure the improvement of each method separately, and the combination of them. Results have shown that a simple combination of the different query expansion approaches is more efficient than the use of any of them separately. This result confirms our thesis that the information exploited by each approach is different and it is worthwhile to investigate more sophisticated ways of performing this combination, what we plan to do in future works.

References

1. Gianni Amati, Claudio Carpineto, and Giovanni Romano. Comparing weighting models for monolingual information retrieval. In *CLEF*, pages 310–318, 2003.
2. Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20(4):357–389, 2002.
3. Claudio Carpineto, Renato de Mori, Giovanni Romano, and Brigitte Bigi. An information-theoretic approach to automatic query expansion. *ACM Trans. Inf. Syst.*, 19(1):1–27, 2001.
4. Thomas M. Cover and Joy A. Thomas. *Elements of information theory.* Wiley-Interscience, New York, NY, USA, 1991.

5. E. N. Efthimiadis. Query expansion. *Annual Review of Information Systems and Technology*, 31:121–187, 1996.
6. Y. Jing and W. Bruce Croft. An association thesaurus for information retrieval. In *Proceedings of RIAO-94, 4th International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 146–160, New York, US, 1994.
7. C. Macdonald, B. He, V. Plachouras, and I. Ounis. University of Glasgow at TREC 2005: Experiments in Terabyte and Enterprise Tracks with Terrier. In *Proceeddings of the 14th Text REtrieval Conference (TREC 2005)*, 2005.
8. Helen J. Peat and Peter Willett. The limitations of term co-occurrence data for query expansion in document retrieval systems. *JASIS*, 42(5):378–383, 1991.
9. Carol Peters and Martin Braschler. European research letter: Cross-language system evaluation: The clef campaigns. *JASIST*, 52(12):1067–1072, 2001.
10. V. Plachouras, B. He, and I. Ounis. University of Glasgow at TREC2004: Experiments in Web, Robust and Terabyte tracks with Terrier. In *Proceeddings of the 13th Text REtrieval Conference (TREC 2004)*, 2004.
11. Yonggang Qiu and Hans-Peter Frei. Concept based query expansion. In *SIGIR*, pages 160–169, 1993.
12. C. J. Van Rijsbergen. A theoretical basis for the use of cooccurrence data in information retrieval. *Journal of Documentation*, (33):106–119, 1977.
13. J. J. Rocchio. Relevance feedback in information retrieval. In G Salton, editor, *The SMART retrieval system*, pages 313–323. Prentice Hall, 1971.
14. Hinrich Schütze and Jan O. Pedersen. A cooccurrence-based thesaurus and two applications to information retrieval. *Inf. Process. Manage.*, 33(3):307–318, 1997.
15. Ellen M. Voorhees. Overview of the trec 2003 robust retrieval track. In *TREC*, pages 69–77, 2003.
16. Ellen M. Voorhees. The trec robust retrieval track. *SIGIR Forum*, 39(1):11–20, 2005.
17. Ellen M. Voorhees. The trec 2005 robust track. *SIGIR Forum*, 40(1):41–48, 2006.
18. Jinxi Xu and W. Bruce Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Trans. Inf. Syst.*, 18(1):79–112, 2000.
19. Angel Zazo, Carlos G. Figuerola, and José Luis Alonso Berrocal. REINA at CLEF 2006 robust task: Local query expansion using term windows for robust retrieval. In A. Nardi, C. Peters, and J.L. Vicedo, editors, *ABSTRACTS CLEF 2006 Workshop, 20-22 September, Alicante, Spain. Results of the CLEF 2006 Cross-Language System Evaluation Campaign*, 2006.

Meta-Search Utilizing Evolutionary Recommendation: A Web Search Architecture Proposal

Dušan Húsek[1], Keyhanipour[2], Pavel Krömer[3], Behzad Moshiri[2], Suhail Owais[4], Václav Snášel[3]

[1] Institute of Computer Science of Academy of Sciences of the Czech Republic, 18207 Prague, Czech Republic
dusan@cs.cas.cz

[2] Control and Intelligent Processing Center of Excellence, School of Electrical and Computer Engineering, University of Tehran, Iran
a.keyhanipour@ieee.org, moshiri@ut.ac.ir

[3] Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VŠB Technical University of Ostrava, 17. listopadu 15, 708 33 Ostrava Poruba, Czech Republic
{pavel.kromer.fei, vaclav.snasel}@vsb.cz

[4] Information Technology, Al-Balqa Applied University - Ajloun University College, P.O. Box 6, JO 26810 Ajloun, Jordan
suhailowais@yahoo.com

Abstract. An innovative meta-search engine named WebFusion has been presented. The search system learns the expertness of every particular underlying search engine in a certain category based on the users preferences according to an analysis of click-through behavior. In addition, an intelligent re-ranking method based on ordered weighted averaging (OWA) was introduced. The re-ranking method was used to fuse the results scores of the underlying search engines. Independently, a progressive application of evolutionary computing to optimize Boolean search queries in crisp and fuzzy information retrieval systems was investigated, evaluated in laboratory environment and presented. In this paper we propose an incorporation of these two innovative recent methods founding an advanced Internet search application.

1 Motivation

WWW consists of more than ten billion publicly visible web documents [1] distributed on millions of servers world-wide. It is a fast growing and continuously changing dynamic environment. Individual general-purpose search engines providing consensual search services have been unable to keep up with this growth. The coverage of the Web by each of the major search engines has been steadily decreasing despite their effort to comprehend larger porting of web space. Several investigations show that no single standalone search engine has complete coverage and it is unlikely any single web search engine ever will [2]. Rather

© A. Gelbukh (Ed.)
Advances in Natural Language Processing and Applications
Research in Computing Science 33, 2008, pp. 189-200

Received 14/10/07
Accepted 07/12/07
Final Version 17/01/08

insufficient coverage of the web by standalone general search engines forced research into the area of meta-search systems as tools sending user queries to multiple search engines and combining the results in order to improve search effectiveness [2, 3]. Therefore, a meta-search engine can be described as an interface on top of multiple local search engines providing uniform access to many local search engines.

Moreover, unified consensual approach to search requirements of all users becomes with growing amount of data and documents on the WWW inefficient in satisfying the needs of large number of individuals desiring to retrieve particular information. Personalized approach to the needs of each user is general trend in state-of-the-art web applications including search engines. Personalization, based on stored knowledge of users general needs, area(s) of interest, usual behavior, long and short term context and search practices can be evaluated when improving web search applications, no matter if they are standalone search engines or more advanced meta-search systems lying on the top of individual search applications.

This paper proposes an incorporation of two recent intelligent methods for improving web search meta-search based on implicit evaluation of local search engines expertness and evolutionary query optimization based on user profiles.

2 Webfusion

Each search engine covers a specific region on the Web. In this regard, meta-search engines use few simultaneous search engines at once to embrace larger parts of the search space. Different techniques of information retrieval which are used in underlying search systems have caused different efficiency of various search engines in particular expertise areas [4]. Majority of contemporary meta-search programs send the query to all of the underlying search engines in their knowledge base which may cause some problems such as resource consumption. The problem is also that each underlying search engine returns too many Web pages, which takes much time for meta-search engine to merge these returned lists and to determine the relevant documents.

Several ranking mechanisms, often based on probabilistic contents, are used to provide search results in a relevant order according to the users queries [5]. An important drawback of most of these techniques is that they do not consider the users preferences. Therefore, the relation between the outcomes and the preferences of the users could not be completely matched [6]. An evidence of user preferences can be found in query logs, created during web search and browsing activities.

There were several investigations concerning query log analysis for search improvement. Joachims [7] proposed a method of utilizing click-through data in learning of a retrieval function. He introduced a novel method for training a retrieval function on the basis of click-through data called Ranking SVM.

Among the more recent methods, QueryFind [8] was based on users feedbacks with respect to the underlying search engines recommendations. They provide more relevant results with the higher rank in the results list. Another new approach is based on exploiting the filtering capabilities of search-engines and the generalized use of weights and aggregation operators to rank documents [9]. These methods do not consider the search engines expertness in a certain category, which is promising when different sources of knowledge with different coverage are used.

2.1 Click-Through Data

Users searching behaviors include queries, click through the list of the returned results, Web pages content information and browsing activities captured in query logs. These query logs contain rich information which can be used to analyze users behaviors and improve the results quality [7, 8]. Formally, click-through data in search engines can be seen as triples *(q,r,c)* consisting of the query q, the ranking r presented to the user, and the set c of links that the user has clicked. Clearly, users do not randomly click on links, but make an (somewhat) informed choice. While Click-through data is typically noisy and clicks are not 'perfect' relevance judgments, the clicks are likely to convey some information [7]. Most users click on rather relevant results and we should benefit from a large quantity of query logs. Experiments show that about 82 procent of the queries are in fact related to the topics of the clicked Web pages [10].

2.2 OWA

The Ordered Weighted Averaging (OWA) operators [10] provide the means for aggregating scores associated with the satisfaction of multiple criteria, which unifies in one operator the conjunctive and disjunctive behavior. The OWA operator of dimension n could be described as: $F : R^n \to R$:

$$OWA(x_1, x_2, x_3, \dots, x_n) = \sum_{j=1}^{n} w_j x_{\sigma(j)} \tag{1}$$

Where σ is a permutation ordering elements $x_{\sigma(1)}, x_{\sigma(2)}, \dots, x_{\sigma(n)}$. The weights are all non-negative and their sum is equal to one. The OWA operators can be seen as a parameterized way to go from the min to the max. In this context, a degree of maxness (initially called orness) was introduced in [11], defined by:

$$Maxness(w_1, w_2, \dots, w_n) = \frac{1}{n-1} \sum_{j=1}^{n} w_j (n-j) \tag{2}$$

For the minimum, we have maxness $(1, 0, \dots, 0) = 0$ and for the maximum maxness $(0, \dots, 0, 1) = 1$.

2.3 Search Engine Expertness modeling and Result re-ranking

To each underlying search engine, an expertness value $expt_i^c$ has been assigned. The value illustrates the expertness of the search engine i in the category c. The expertness value is updated by bootstrapping technique based on previous expertness value and the current usage behavior of the user [12]. In this way, search engines in which the higher ranked results are clicked earlier, are more rewarded than the others. The expertness updating formula and the reward function are given as follows:

$$expt_i^c = (1-\alpha)expt_i^c + \alpha\gamma_l \tag{3}$$

$$\gamma_l = \frac{\sum_{i \in hittedresults}(N-i)t_i}{\sum_{i=1}^{N}(N-i)i} \tag{4}$$

Where N is the number of returned results, i is the index of a result in the ranked list and t_i is the index of hit result i. The learning rate α is also modified by the following formula:

$$\alpha = exp(\beta t) \tag{5}$$

Where t is the iteration number and β is the regulator of the learning rate that was in performed experiments fixed to 0.2. Each iteration starts by query submitting and it will be finished after the user selects the results and closes the session. The learning rate allows at the beginning of learning process more exploration than exploitation and the exploration-exploitation is decreasing in time.

The returned results of the underlying search engines are taken as their decisions for the submitted query which are fused in the decision level of information fusion. In decision level, the more weights are assigned to the higher results of the more expert search engines for the dispatched query in the corresponding category.

For every item, ranks that are assigned by the underlying search engines are assumed as the base of OWA method. For each result item j from the search engine i which is classified in category c, a weight is computed:

$$w(i,j,c) = expt_i^c(1 - \frac{1}{N}) \tag{6}$$

For each result item, search engines are sorted decreasingly based on these weights. Then OWA weights are assigned to these search engines according to the equation (3). Final weight of each result item is calculated:

$$w_F(j) = \sum_{i=1}^{N} w(i,j,c) \cdot w_{OWA}(i) \tag{7}$$

After assigning the weights to each returned result, they are sorted according to their weights decreasingly and the final ordered list is generated.

2.4 Evaluation

An experimental meta-search environment named WebFusion [13] was implemented in order to evaluate the effectiveness of the proposed method. 280 sample queries in the Computers category were taken. These queries were collected using a proxy application mounted on the server of the Instrumentation and Industrial Control Lab. of the ECE department of the University of Tehran. The categories presented by Open Directory Project (http://dmoz.org/), the largest, most comprehensive human-edited directory of the Web, were used to categorize the queries sent by the users. In the experiments, users were asked to manually select the category of their queries. Experiments were shown in the category of Computers.

The performance of the WebFusion is measured by factors such as average click rate, total relevancy of the returned results and the variance of the clicked results. The first measure which is the average position of the clicked results indicates that wither interesting results are settled at the top of the ranked list. If average click rate is relatively little, it shows that users can find useful results at the first portion of the ranked list while saving time and energy. The second criterion, measures the relevancy of the content of the returned results according to judgment of the users. This criterion can be considered as the indicator of nature of responses. The last factor shows the long term performance of the meta-search engine which is determined by the behavior of the users. Consideration of these criteria together, can provide a good insight about the meta-search engine. For calculating the relevancy of the results, we have extended the relevancy measure proposed in [14] into the following:

$$rel = \frac{2 \cdot rel + undecided}{2 \cdot ret} \tag{8}$$

Sample search queries were executed on WebFusion. The results are supposed to be classified to relevant, undecided, and irrelevant documents. Each experiment participant was asked to evaluate the result items in the corresponding classes.

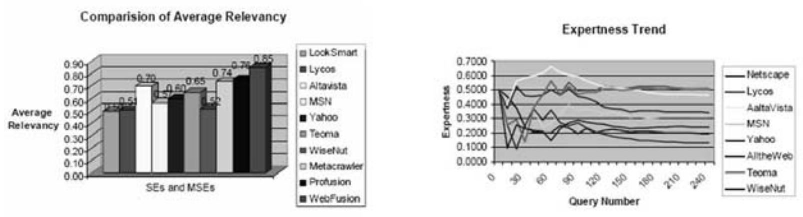


Fig. 1 Average relevancy and expertness learning results.

The comparison of WebFusion to the ProFusion[5] and some other underlying search engines shows an obvious enhancement. Average relevancy of WebFusion results is 85.5%, while this is about 76% for ProFusion. WebFusion has also better performance than the MetaCrawler [6]. It is noticed that the average relevancy of WebFusion is 15.5% better than the best underlying search engine in the sense of relevancy. The learning process of WebFusion along the iterations can be spotted by the means of average click rate.

Average click rate has decreased from 11.77 to below 6.23 in 240 queries, while this is 9.701 for the ProFusion. The proposed method leads to a mapping between query categories and the underlying search engines of the WebFusion. In other words, the most expert search engines of each category are identified. Figure 1 summarizes the comparison of WebFusions average relevancy and captures the expertness learning process for the category 'Computers' along the iterations.

Concluding from above presented experiments, this approach provides more relevant Web pages in higher ranks and hence reduce users time and tension when finding useful information within the retrieved result sets.

3 Personalized Evolutionary Query Optimization

An individual user profile (IUP), containing stored knowledge about system user, could be utilized to improve search results by the means of personalization. Search engine equipped with user profiling can exploit user-specific acquirements to retrieve documents satisfying search queries with respect to individual user, her or his general needs, preferences, abilities, history, knowledge and current context.

Explicit profiles, defined by users themselves, are rather imprecise and not enough flexible. Instead, various techniques for implicit creation and maintenance of user profiles are being investigated [15]. In this section, an evolutionary approach exploiting information from user profiles to optimize search queries by artificial evolution will be discussed.

3.1 Evolutionary Techniques

Evolutionary Algorithms (EA) are family of stochastic search and optimization methods based on mimicking successful strategies observed in nature [16]. EAs emulate the principles of Darwinian evolution and Mendelian inheritance for the use in computer science.

Artificial evolution operates over a population of individuals (chromosomes) encoding possible solutions by applying so called genetic operators (selection, crossover, mutation) [16]. The individuals are evaluated using objective function assigning a fitness value to each individual. Fitness value mirrors the ranking of each individual as a solution of the problem. Competing individuals intensively

[5] http://www.profusion.com
[6] http://www.metacrawler.com

search problems solution space towards optimal solution in more directions simultaneously [16].

EAs are well proven general adaptable concept with good results. The family of evolutionary algorithms consists of Genetic Algorithms (GA), Evolutionary strategies (ES) and Evolutionary programming (EP).

Genetic Algorithms were introduced by Holland when exploring the possibilities of computer simulated evolution [16]. They are widely applied and highly successful EA variant. GAs were designed as a general model of adaptive processes and found most of its applications in the domain of optimization. Basic scheme of original generational GA is:

```
1. Define objective function
2. Encode initial population of possible solutions as fixed length
   binary strings and evaluate chromosomes in initial population
   using objective function
3. Create new population (evolutionary search for better solutions)
   a. Select suitable chromosomes for reproduction (parents)
   b. Apply crossover operator on parents with respect to crossover
      probability to produce new chromosomes (known as offspring)
   c. Apply mutation operator on offspring chromosomes with respect
      to mutation probability. Add newly constituted chromosomes to
      new population
   d. Until the size of new population is smaller than size of
      current population go back to (a).
 e. Replace current population by new population
4. Evaluate current population using objective function
5. Check termination criteria; if not satisfied go back to (3).
```

Termination criteria, crossover probability, mutation probability, population size, maximum number of processed generations and migration strategy are among the most important parameters of each GA based solution.

A high-level variant of GAs, Genetic programming (GP), attracts attention. GP uses hierarchical rather than linear chromosomes and thus is able to evolve computer programs or other hierarchically structured entities such as search queries. Genetic operators are usually applied on nodes of tree chromosomes encoding hierarchical individuals. GP has a good ability to produce symbolic output in response to symbolic input [16].

3.2 Query Optimization by Genetic Programming

More works presenting the use of evolutionary optimization in search and information retrieval have been presented. Kraft et al. [4] published GP method to optimize user search queries. Ramos and Abraham [17] published hybrid method for user profile improvement combining stigmeric paradigms and genetic programming.

In our work, Boolean queries were experimentally evolved over a sample document collection with several setups of GP parameters. Effectiveness of IR system was measured through values indicating the ability of IR system to satisfy users information needs [2]. Among IR effectiveness measures, precision (P) and recall (R) as the most used IR performance measures [2] were applied. P and R are defined as:

$$P = \frac{|Rel \cap RelRet|}{Ret} R = \frac{|Rel \cap RelRet|}{Rel} F = \frac{2PR}{P+R} \tag{9}$$

Rel stands for a set of all relevant documents in the collection, Ret is set of all retrieved documents and RelRet is set of retrieved relevant documents. Precision describes the exactness of result set retrieved in response to user request, the ratio of number of relevant retrieved documents to number of all retrieved documents and recall describes the completeness of result set, the ratio of the number of relevant retrieved documents to the number of all relevant documents. Harmonic mean of precision and recall called F-score (F) was used for assembling precision and recall in one scalar value [2]. In our experiments, all three mentioned fitness measures were used.

We considered user models containing Boolean search queries representing ones individual search needs, long and short term context and document relevance estimation [18]. Several experiments were executed using data extracted from the LISA collection [7]. The collection was indexed for Boolean IR and Extended Bolean IR systems, using Saltons indexing function based on normalized term frequency and normalized inverse document term frequency [19] in the latter case. Indexed collection contained 5999 documents and 18442 unique indexed terms. Due to stochastic character of GP process, all experiments were executed multiple times and mean experimental results evaluated.

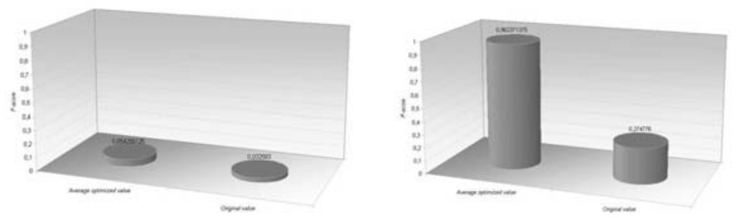


Fig. 2 Query optimization improvement for random initial population (left) and seeded initial population (right).

Several GP parameters were initialized for all experiments and the result of query evolution in cases with different (random initial population and initial population seeded with better ranked individuals) initial populations was ob-

[7] http://www.dcs.gla.ac.uk/idom/ir_resources/test_collections/

served. The experiments were done using both, standard Boolean queries and fuzzified extended Boolean queries.

In the set of experiments with standard Boolean queries and seeded initial population, the average F-score value increased from initial 0,27 to optimized 0,99 (while 1 was maximum). Random initial population started with the average F-score value 0,04 and the F-score value after optimization reached 0,07. Experiments with fuzzified Boolean queries resulted into following: for seeded initial population, the F-score value boosted from average 0,27 to 0,98. Random initial population of average opening F-score 0,03 was optimized up to 0,07. An example illustrating experimental results is shown in Figure 2.

The experiments have shown that GP can be used for optimizing user queries in information retrieval systems. Crucial for successful optimization was the quality of initial population. For successful optimization, initial population must contain at least some quality queries pointing to documents related to users needs. F-score fitness was preferred as a measure combining precision and recall into one value by the means of information retrieval and therefore simplifying query optimization from multi-objective to single-objective task.

4 Advanced Web Search Architecture Proposal

An advanced web search application, preventing some of imperfections summarized in first section of this paper, could incorporate both above presented novel techniques. The personalized meta-search enabled system will consist of:

- Implicit user profiles (IUP) based on click-through data created and updated by analysis of user behavior observed via click-through data. Additionally, profiles will contain previous user queries. User profiles will be exploited when optimizing search queries.
- Implicit search engine expertness profiles (SEEP) created and updated by analysis of users click-through behavior. Search engine profiles will be exploited when retrieving and re-ranking results from underlying local search engines.
- Meta-search and re-ranking subsystem interacting with local search engines and providing data for recommendation component.
- Personalized recommendation subsystem based on queries from user profiles optimized by genetic programming and document relevance estimation. The component will offer alternative queries, created with respect to current user query, previous search behavior and personal relevancy information stored in user profiles.

Introduced architecture, consisting of independent but complementary advanced components, could notably increase search comfort and decrease the time needed for user interaction with web search applications thus improving the efficiency of the search task. Meta-search environment, utilizing and extending services provided by contemporary search engines and featuring personalized recommendation system build upon consensual systems is supposed to offer high added value at low cost.

Implicit observation of user click-through behavior is used to provide data basis for building both, SEEPs and IUPs. The simultaneous exploitation of search engine expertness profiles and individual user profiles will be deployed to improve the result sets retrieved in response to user queries. Results will be re-ranked with respect to detected search engine expertness, preferring for particular category results retrieved by search engines with higher expertness. User profiles will be employed to be a basis for personalized query optimization, performed on the top of retrieved and according to SEEP re-ranked result set. Neither the SEEPs nor IUPs are needed to be specified explicitly. On the contrary, meta-information provided by ordinary user tasks performed during web search activities and hence almost omnipresent (and in majority of search environments omitted), will be extracted and reasonably exploited to discover SEEPs and UIPs. Hereby created and maintained search engine expertness profiles and user profiles do not require any additional feedback from the users, although an explicit feedback should be considered as a mean to speed up the expertness and user learning process and make the profiles more literal.

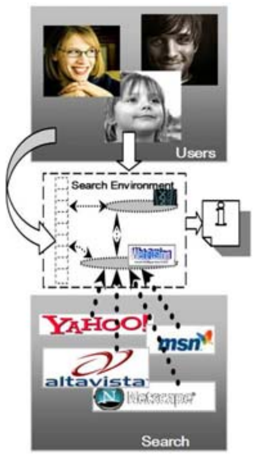


Fig. 3 Proposed search architecture.

A search environment implementation, built according to proposed architectural concepts, will offer the advantages of expertness and user aware advanced search techniques on the top of contemporary systems allowing the exploitation of their todays information retrieval capabilities for more sophisticated search improvement. Our future work should lead to prototype implementation of proposed architecture and experimental confirmation of proposed concepts. Major attention will be paid to the particular methods for creating click-through based implicit SEEPs and IUPs.

References

1. Gulli, A., Signorini, A.: The indexable web is more than 11.5 billion pages. In: WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web, New York, NY, USA, ACM Press (2005) 902–903
2. Losee, R.M.: When information retrieval measures agree about the relative quality of document rankings. Journal of the American Society of Information Science **51** (2000) pp. 834–840
3. Gordon, M., Pathak, P.: Finding information on the world wide web: the retrieval effectiveness of search engines. Inf. Process. Manage. **35** (1999) 141–180
4. Kraft, D.H., Petry, F.E., Buckles, B.P., Sadasivan, T.: Genetic Algorithms for Query Optimization in Information Retrieval: Relevance Feedback. In Sanchez, E., Shibata, T., Zadeh, L., eds.: Genetic Algorithms and Fuzzy Logic Systems, Singapore, World Scientific (1997)
5. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems **30** (1998) 107–117
6. Amento, B., Terveen, L., Hill, W.: Does authority mean quality? predicting expert quality ratings of web documents. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 296–303
7. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM (2002)
8. Xue, G.R., Zeng, H.J., Chen, Z., Yu, Y., Ma, W.Y., Xi, W., Fan, W.: Optimizing web search using web click-through data. In: CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management, New York, NY, USA, ACM Press (2004) 118–126
9. Gomez, M., Abasolo, J.C.: Improving meta-search by using queryweighting and numerical aggregation operators. In: Proceedings of the Information Processing and Management of Uncertainty conference, IPMU 2002. (2002)
10. Yager, R.R.: On ordered weighted averaging aggregation operators in multicriteria decisionmaking. In Dubois, D., Prade, H., Yager, R.R., eds.: Readings in Fuzzy Sets for Intelligent Systems. Kaufmann, San Mateo, CA (1993) 80–87
11. Zhang, D., Dong, Y.: An efficient algorithm to rank Web resources. Computer Networks (Amsterdam, Netherlands: 1999) **33** (2000) 449–455
12. Wang, P.H., Wang, J.Y., Lee, H.M.: Queryfind: Search ranking based on users' feedback and expert's agreement. In: EEE '04: Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04), Washington, DC, USA, IEEE Computer Society (2004) 299–304
13. Keyhanipour, A.H., Moshiri, B., Piroozmand, M., Lucas, C.: Webfusion: Fundamentals and principals of a novel meta search engine. In: Proceedings of International Joint Conference on Neural Networks. (2006)
14. Gauch, S., Wang, G.: Information Fusion with ProFusion. In: Proceedings of WebNet '96: The First World Conference of the Web Society, San Francisco, CA (1996) 174–179
15. Cordn, O., de Moya, F., Zarco, C.: Fuzzy logic and multiobjective evolutionary algorithms as soft computing tools for persistent query learning in text retrieval environments. In: IEEE International Conference on Fuzzy Systems 2004, Budapest, Hungary (2004) 571–576

16. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge, MA (1996)
17. Ramos, V., Abraham, A.: Evolving a stigmergic self-organized data-mining. CoRR **cs.AI/0403001** (2004)
18. Húsek, D., Snášel, V., Neruda, R., Owais, S.S.J., Krömer, P.: Boolean queries optimization by genetic programming. WSEAS Transactions on Information Science and Applications **3** (2006) 15–20
19. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24** (1988) pp. 513–523

Structuring Job Search via Local Grammars

Sandra Bsiri[1], Michaela Geierhos[1], and Christoph Ringlstetter[2]

[1] CIS, University of Munich
[2] AICML, University of Alberta

Abstract. The standard approach of job search engines disregards the structural aspect of job announcements in the Web. Bag-of-words indexing leads to a high amount of noise. In this paper we describe a method that uses local grammars to transform unstructured Web pages into structured forms. Evaluation experiments show high efficiency of information access to the generated documents.

1 Introduction

After years of steady growth, the main source of information about job announcements is the Internet [1]. Though, there is some bastion left for high profile openings and local jobs, the traditional newspaper advertisement is of declining importance. Big organizations such as corporations or universities provide an obligatory *career link* on their home pages that leads to their job openings. According to a recent study [2], for example, 70% of the workforce in France searches for jobs on the Internet.[3] The interface arranging access to the information on job opportunities is provided by specialized job search engines. Due to the sheer amount of data, a sophisticated technology which guarantees relevancy would be required. In reality, though, search results are rife with noise.

As compared to a standard search engine, job engines are specialized in that they only index a certain part of the document space: pages that contain job announcements. Unfortunately, in most cases, at this point the specialized treatment of the data has its end. The majority of engines uses variants of the standard vector space model [3] to built up an index that later on is approached by similarity based query processing. This blind statistical model leads often to poor results caused, for example, by homography relations between job descriptors and proper names: in a German language environment a search for a position as a *restaurant chef* (German: *Koch*) easily leads to a contamination with documents that mention a "Mr. Koch" from human resources, with "Koch" being a popular German name. Problems like these arise because the used bag-of-words model treats all terms equally without being aware of their context.

The starting point of a solution is the structured nature of the data spanning the search space and the queries accessing it. Unlike a general search scenario, job search can be seen as a slot-filling process. The indexing task is then to detect concept-value pairs in the HTML-documents and make them accessible. Other

[3] For certain groups such as IT-professionals this value probably comes close to 100%.

Received 25/10/07
Accepted 07/12/07
Final Version 16/01/08

than fully structured data stored in databases, Web pages for career announcements are set up by humans for humans. Nevertheless, induced by widespread copying of HTML-source-code and corresponding reengineering on account of popular design recommendations, the data are more uniform than is the case for other Web genres: they can be characterized as semi-structured data. Obligatory elements, such as the name of the organization in a home page or the identifier of an announced position combined with a highly restrictive domain vocabulary, make *local grammars* the appropriate tool to discover the logic of a job offer that can then be integrated into a relational structure. As our studies during this research were limited to the Francophone job market, the investigated language was French.

The first step to provide transparent access to announced job openings by an integrated platform was the automatic classification of Web pages into a database of organization home pages. The HTML-structure of these pages was then scanned for anchor elements that lead to job advertisements.

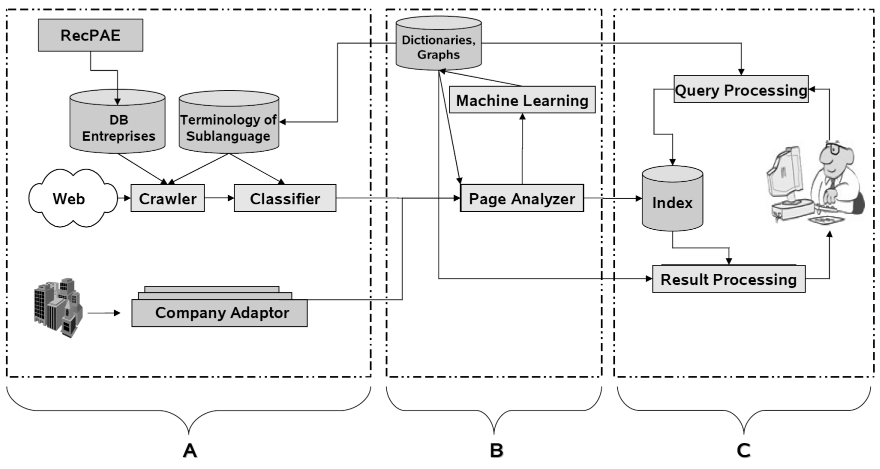


Fig. 1. Overview of the system architecture

Local grammars [4, 5] and electronic lexica [6, 7] were employed for information extraction to transform running text into a semantically structured form. By linguistically motivated analysis, the form slots were automatically filled with values to enable high selectivity for the subsequent retrieval functionality. In Figure 1, we show our architecture for job search that is based on three interactive modules to establish a relational database holding job openings and a query module.

The paper is structured as follows. The next section introduces strategies for the location of job openings on the Web. Section 3 presents methods to extract the structured information of a job announcement and its integration into a frame to enable high selectivity for user queries. In Section 4 we evaluate the proposed methods. The conclusion comments on practical implications of the given approach and the directions of future work.

2 Locating online job offers

To develop a centralized platform for job search, all relevant job announcement documents on the Internet have to be automatically detected. Two different strategies have been explored: firstly, a system for the recognition of the home pages of organizations was developed. From the home pages, we followed the career links and scanned for job opening pages. Secondly, we used genre-specific terminology, bootstrapped from a development corpus, to run a focused crawler that retrieves and classifies pages to detect job announcements.

2.1 Identification of organization home pages

The first approach to detect published job openings on the Web is based on a database of URLs that has to be updated continually because of the steady dynamics of the job market. An automatic binary classification system was developed that for every input URL decides whether its target falls into the class *organization home page* or not. The classification relies on the following main feature classes:

– Formal HTML-structure of the document
– Evaluation of the URL, meta-information and title
– Analysis and classification of the anchor elements into pre-defined semantic classes (cf. Table 1)
– Identification of the organization name
– Address, phone number, registration number, etc.
– Extraction of typical expressions and complex terminology

Table 1. Selected examples of anchor texts of links in organization home pages.

Carrière *(Careers)*	Nous recrutons *(We hire)* Nos offres d'emploi *(Our openings)*
Produits/Services *(Products/Services)*	Nos Produits *(Our Products)* Accès à la boutique *(To the shop)*
Contact *(Contact information)*	Nous contacter *(Contact us)* Pour venir nous voir *(Visit us)* Nos coordonnées *(Contact)*
Société *(Company Information)*	Notre Société *(Our organization)* Qui sommes nous? *(Who are We)* Entreprise *(Company)*
Presse *(Press/Media)*	Communiqués de Presse *(Press Information)* La presse et nous *(In Press)* Presse infos *(Press Information)* media *(Media)*
Clients/Partenaires *(Customers/Partners)*	Nos Clients *(Our Customers)* Espace clientèles *(Customer Area)* Nos partenaires *(Our Partners)* Relation client *(Customer Relations)*

Phrases and terminology of organizations. In France, every company can be identified by its SIREN-number, a 9-digit code, within the directory of the National Institute for Statistics and Economical Studies (INSEE). Together with the location of the company, the SIREN–number comprises the identifier for the commerce register. Additionally, the SIRET number, the SIREN identification extended by a code for the class of business and NTVA, the European tax identification number, were used for organization detection. The three formal company identifiers (SIREN/SIRET/NTVA) were extracted with local grammars. Besides that, the system searched for organization-specific standard phrases (*frozen expressions*) that frequently emerge on home pages of organizations. This task is conducted by a multitude of local grammars that allow the modeling of a high degree of syntactic variation. This flexibility could not be reached by standard string search, where a list of collected phrases would be matched with the document, since minor morpho-syntactic variation prevents the match. Bootstrapping with local grammars [8, 9] is a much more promising method, as illustrated by the following example.

- Notre société , leader mondial sur le marché [...]
- Notre société est leader européen dans le secteur [...]
- Notre société est leader sur le marché mondial [...]
- Notre société leader dans son domaine [...]
- Notre société en position de leader au niveau régional

Though these five phrases are highly different on the morpho-syntactic level, they describe a highly similar context of the word "leader" that refers to the same semantic concept. By a local grammar, as shown in Figure 2, it is possible to efficiently represent all five phrases.

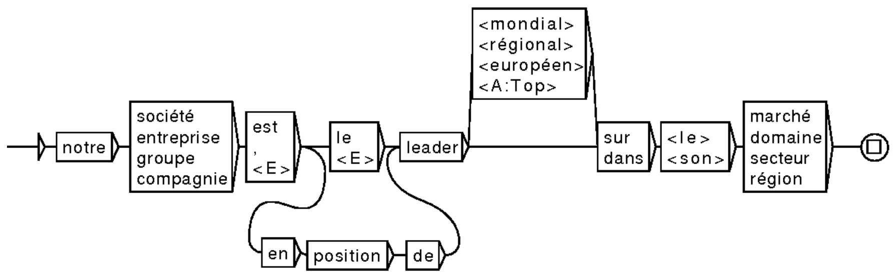


Fig. 2. Local grammar that models the context of the word *leader*.

Organization name and URL. A highly descriptive feature to recognize the home page of an organization is the connection between the name of the organization and the URL of the page. The automatic recognition of company names [10] is a sub-task of *named entity recognition*. Because of ambiguities and sparseness, a list of organization names can not attain sufficient coverage with regards to the annotation of job announcements. Machine learning systems are extremely dependent on the training corpora or the sub-language used [11, 12].

We used linguistically motivated local grammars to describe organizational contexts and to detect the organization identifier within these contexts. Local grammars [5] enable the description of a local context and restrict the emergence of certain lexical or syntactical features to a window of predefined size. Thus, they avoid or reduce ambiguities that occur for a simple keyword search. To establish a match between the candidates for the organization name and the URL we applied a segmentation algorithm for the domain name. If the result of this segmentation algorithm matches one of the candidates found in the text of the page, it will be regarded as the name of the organization publishing the Web page.

Classification algorithm The retrieved feature value sets are then fed into a classification algorithm that provides a decision on the basis of weighted features as for whether a Web page is an organization homepage or not. When an organization home page has been recognized, it is scanned for links to job openings. This search is guided by the HTML-structure of the page. We introduced a class *careers* which refers to the information provided by anchor texts underlying those links that lead to the openings of the organization. During the learning cycle of the system we found more than 80 different linguistic sequences that lead to a classification into the category *careers*.

2.2 Focused crawler

A second method for the retrieval of job advertisements on the Internet concentrates on the typical terminology of a job offer, which comes close to a sub-language [11, 12]. To this end, during the training phase, frozen expressions and complex phrases were collected. We distinguish two types: nominal phrases that semantically structure the job descriptions and manifest themselves in the headings of the paragraphs of the job announcement, and frozen expressions that contain specific verbs and nouns of the sub-language and that only make sense within the context of job announcements. With this terminological specification, a focused crawler can decide whether a Web page falls into the category job announcement even if the HTML-structure of the document gives no insight whatsoever on the semantic organization of the page.

3 IE and document vectors

The second module of the introduced system concerns information extraction (IE) and the automatic transformation of the textual representation of a job announcement into a semantically structured document. With more than 100 local grammars and electronic dictionaries, an automatic population of a relational database of job announcements is realized. The structure of the database can be considered as a form that comprises the information contained in a job advertisement. Unlike other systems, the form filling process is fully automatized. From a document that has been recognized as a job announcement by the system (part A), we extract information to fill a form that is presented in Table 2.

Example of a job offer form	
Date of publication	22. Jan 2007
Application deadline	fin février *(End of Februay)*
Employment date	mi-mars *(Mid-March)*
Job description	ingénieur d'étude en électromécanique *(Poject manager (electromechanics))*
Contract type	intérim à temps partiel : 1 à 2 jours/semaine *(Temporary employment: 1–2 days/week)*
Employment period	8 mois renouvelables *(8 months renewable)*
Workspace	sud-est de Paris *(South-east of Paris)*
Offered salary	selon profil *(Commensurate with experience)*
Job code	MOR34544/ing-21
Professional experience	expérience de 2 à 3 ans dans un poste similaire *(2–3 years of professional experience in a similar position)*
Education	de formation Bac+5 de type école d'ingénieur *(Engineering degree)*
Company name	CGF Sarl
Office	**Address:** 34 bis rue Berthauds, 93110 Rosny **Phone:** 0 (+33) 1 12 34 45 67 **Fax:** 0 (+33) 1 12 34 45 68 **E-mail:** contact@cgf.fr **Homepage:** http:///www.cgf.fr
Contact	Directeur des RH, Mr. Brice *(HR manager, Mr. Brice)*
Company sector	Construction électromécanique *(Electro-mechanical construction)*

Table 2. Form of a structured job offer

For the transformation of the initial HTML-documents into the form schema, we need different operations. The chronological application of the five preprocessing steps is shown in Figure 3 (see next page).

Preprocessing. Each document is labeled with semantic-structural markers, such as [TagMISSION], [TagPROFIL], and [TagFORMATION]. During the training phase, 13 semantic classes were established that contain frozen expressions or phrases constituting the surface forms (the textual realizations) of a tag. For example, the label [*TagMISSION*] represents the tasks a position is concerned with. The following expressions can be found:

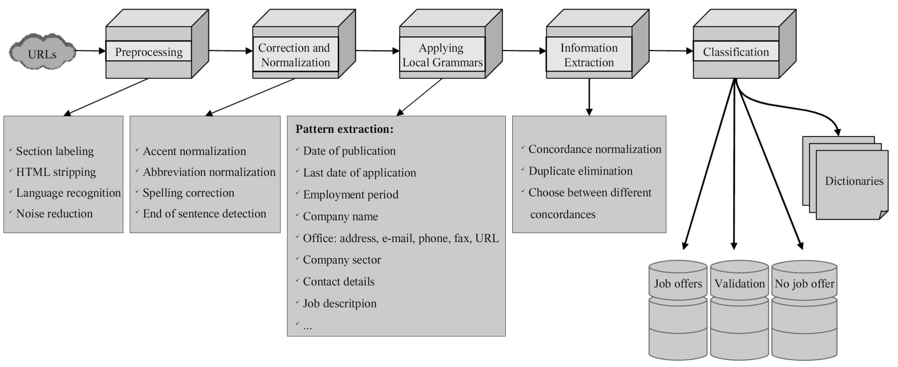


Fig. 3. Steps of the classification and transformation of a possible job annoucement

Your responsibilities
Your duties are
Your field of activities
Duties/activities
We expect
Our expectations
Area of responsibility
We expect from you

During the development of the system we observed three different strategies to compose job advertisements: either frozen expressions are used to structure the document, or we find a mere textual description without a clear structure, and finally a very compact composition with a minimum of information can represent a job advertisement. The third type comprises short announcements such as *welders hired* that do not provide enough linguistic material to be automatically recognized.

After analyzing and labeling the structure, the processed document is stripped of HTML-formating and scripting (JavaScript, ActionScript). Subsequently, the language of the remaining text is classified with the help of the Unitex dictionaries for English, German, and French. If a Web page is composed of less than 50% French words, it is filtered out.

Correction and Normalization. The cleansing step consists of the recognition and correction of orthographic errors with the help of a list of spelling mistakes, which was collected from a large corpus of job advertisements, and the reconstruction of missing accents of French words. During the normalization step, we identify abbreviations and substitute them with their original non-contracted spelling variants, as for example, *ing.* $\mapsto$ *ingénieur, comm.* $\mapsto$ *commercial.* For this purpose, a list of abbreviations and their corresponding non-contracted word sequences was applied. By the end of this processing step, a normalized text is available on which syntactic and lexical analysis can be performed successfully.

Applying local grammars. We created several specialized dictionaries for simple terms as well as for multi-word terms which can deal with a substantial part of the above mentioned sub-language and which conform to the DELA lexicon format [6, 7].

The convention of dictionary development according to the DELA format allows for the application of local grammars within the LGPL[4] software Unitex[5]. This platform provides all linguistic tools necessary for the processing of big corpora and enables the efficient handling of electronic lexica. Additionally, the development of local grammars, represented by directed acyclic graph (DAG) structures (cf. Figure 2), is supported by a graphical development tool.

In order to apply these graphs, a text has to be passed through the following processing pipeline of the Unitex system:

1. *Conversion*: Converts the text into Unicode (UTF-16LE)
2. *Normalization*: Normalizes the special characters, white spaces and line breaks
3. End of sentence detection
4. Dissolving of contractions (e.g. *d'une* $\mapsto$ *de une*)
5. *Tokenization*: Tokenizes the text according to the alphabet of the investigated language
6. *Dictionary*: Performs a lexical analysis by comparing tokens to lexical entries and assigns each word to its possible grammatical categories

The lexical analysis is directly followed by the step of information extraction with local grammars. Each piece of structured information is covered by different extraction units.[6] We created a system of local grammars which work iteratively and sometimes cascaded [13]. The cascaded grammars are executed with different priorities. For example, we developed local grammars to extract job descriptions with eight levels of priority. The graphs of level $n + 1$ are only applied if the grammars of level n with higher priority generate no recall.

Information extraction. The main task of module B (information extraction) is to normalize the extracted sequences and to eliminate duplicates as well as incompletely recognized units. For information extraction, the local grammars organized as DAGs were used. Because sequences are recognized simultaneously by alternative paths of the DAGs that work as annotation transducers a decision on the match strategy had to be made. Since results on training data showed a superior performance, the longest match strategy was used.

Classification. As it is quite unusual that a job offer contains values for all possible information slots, we developed rules modeling the dependency of the

[4] GNU **L**esser **G**eneral **P**ublic **L**icense

[5] http://www-igm.univ-mlv.fr/unitex/

[6] Among them are, for example, job description, company name, office, workspace, and offered salary.

recognized information in the document for the further classification of job advertisements and the transfer to the database. If several of the semantically-structured information bits are realized by phrases, the job description and the employment date are successfully found, the URL will be classified as a job offer. Additionally, it will be indexed according to the retrieved information in the database. By means of a graphical user interface, the annotated job advertisement can be manually enriched with the missing pieces of information. Several functionalities are provided to interactively expand the semantic dictionaries and the local grammars with new surface realizations of semantic concepts.

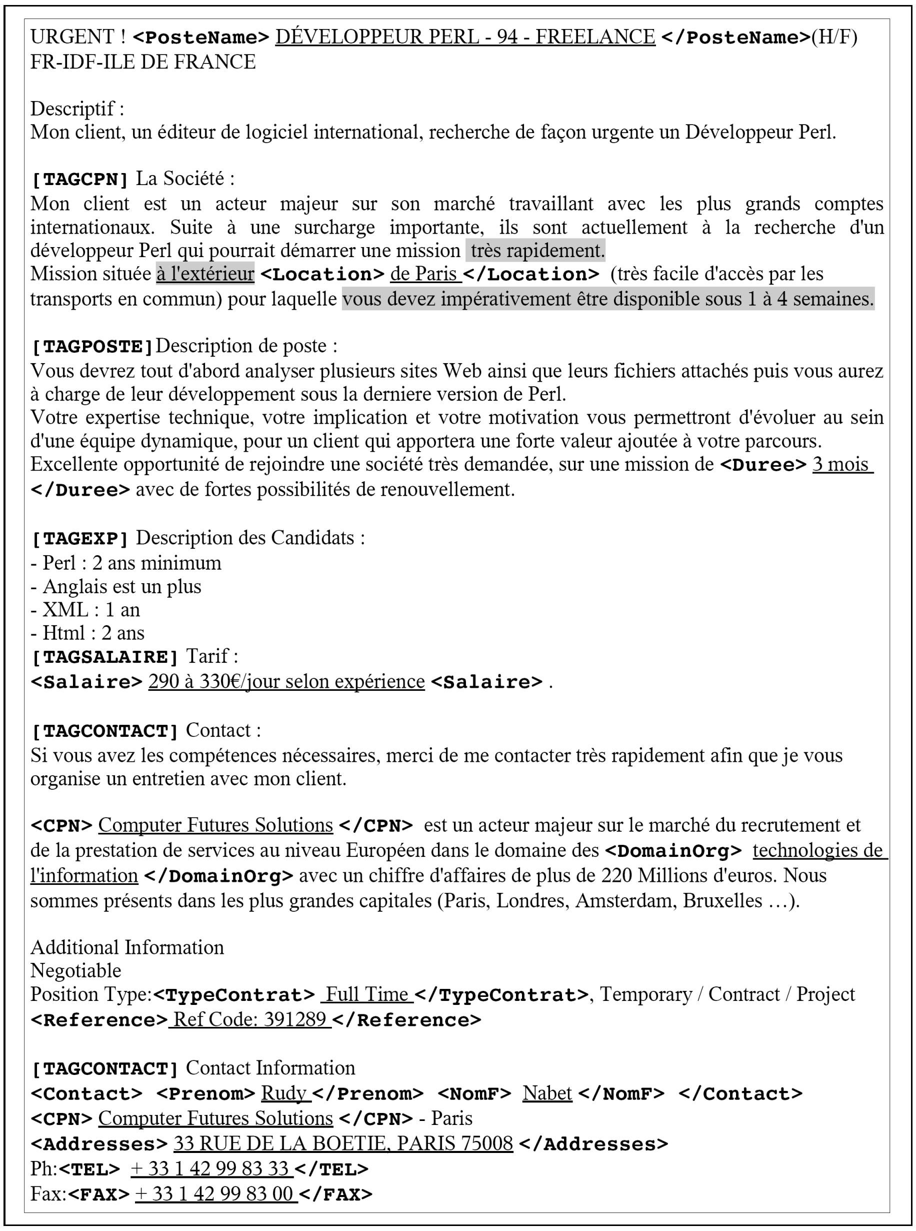
Application. With the help of a concrete example, we want to illustrate typical results as well as the quality of the extraction process, driven by the local grammars developed for our system. In Figure 4 we show a tagged document after being passed through all the processing steps described previously.

The recognized surface realizations of semantic classes are labeled by square bracket tags. Our example in Figure 4 shows that all five instances of the semantic classes occurring in the job offer could be located (*TAGCPN = Company information, TAGPOSTE = Job description, TAGEXP = Qualifications of the employee, TAGSALAIRE = Salary, TAGCONTACT = Contact information*).

By means of the more than 100 different local grammars, 14 of the totally 20 sought-after pieces of information (underlined in Figure 4) could be identified and associated to the correct semantic tags:

`<PosteName>` =	*Job description*
`<Location>` =	*Location*
`<Duree>` =	*Contract period*
`<Salaire>` =	*Offered salary*
`<CPN>` =	*Company name*
`<DomainOrg>` =	*Company sector*
`<TypeContrat>` =	*Type of contract*
`<Reference>` =	*Reference*
`<Contact>` =	*Contact information*
`<Prenom>` =	*First name of the contact person*
`<NomF>` =	*Last name of the contact person*
`<Addresses>` =	*Company's address(es)*
`<TEL>` =	*Phone number of the contact person*
`<FAX>` =	*Fax number of the contact person*

This example shows a sequence not completely recognized by our system which should belong to the concordance of extraction results. Therefore it could not be retrieved as a correct piece of information by our local grammars. According to the automatically gained information, the located place of work would be *"de Paris."* However, it should be *"à l'extérieur de Paris"* (the greater area of Paris). During the primary configuration step of our system, the existing semantics has changed and because of that the system's performance suffers a severe setback concerning this special information slot: if a user looks for a position in the city



URGENT ! **<PosteName>** DÉVELOPPEUR PERL - 94 - FREELANCE **</PosteName>**(H/F)
FR-IDF-ILE DE FRANCE

Descriptif :
Mon client, un éditeur de logiciel international, recherche de façon urgente un Développeur Perl.

[TAGCPN] La Société :
Mon client est un acteur majeur sur son marché travaillant avec les plus grands comptes internationaux. Suite à une surcharge importante, ils sont actuellement à la recherche d'un développeur Perl qui pourrait démarrer une mission très rapidement.
Mission située à l'extérieur **<Location>** de Paris **</Location>** (très facile d'accès par les transports en commun) pour laquelle vous devez impérativement être disponible sous 1 à 4 semaines.

[TAGPOSTE]Description de poste :
Vous devrez tout d'abord analyser plusieurs sites Web ainsi que leurs fichiers attachés puis vous aurez à charge de leur développement sous la derniere version de Perl.
Votre expertise technique, votre implication et votre motivation vous permettront d'évoluer au sein d'une équipe dynamique, pour un client qui apportera une forte valeur ajoutée à votre parcours.
Excellente opportunité de rejoindre une société très demandée, sur une mission de **<Duree>** 3 mois **</Duree>** avec de fortes possibilités de renouvellement.

[TAGEXP] Description des Candidats :
- Perl : 2 ans minimum
- Anglais est un plus
- XML : 1 an
- Html : 2 ans

[TAGSALAIRE] Tarif :
<Salaire> 290 à 330€/jour selon expérience **<Salaire>** .

[TAGCONTACT] Contact :
Si vous avez les compétences nécessaires, merci de me contacter très rapidement afin que je vous organise un entretien avec mon client.

<CPN> Computer Futures Solutions **</CPN>** est un acteur majeur sur le marché du recrutement et de la prestation de services au niveau Européen dans le domaine des **<DomainOrg>** technologies de l'information **</DomainOrg>** avec un chiffre d'affaires de plus de 220 Millions d'euros. Nous sommes présents dans les plus grandes capitales (Paris, Londres, Amsterdam, Bruxelles …).

Additional Information
Negotiable
Position Type:**<TypeContrat>** Full Time **</TypeContrat>**, Temporary / Contract / Project
<Reference> Ref Code: 391289 **</Reference>**

[TAGCONTACT] Contact Information
<Contact> <Prenom> Rudy **</Prenom> <NomF>** Nabet **</NomF> </Contact>**
<CPN> Computer Futures Solutions **</CPN>** - Paris
<Addresses> 33 RUE DE LA BOETIE, PARIS 75008 **</Addresses>**
Ph:**<TEL>** + 33 1 42 99 83 33 **</TEL>**
Fax:**<FAX>** + 33 1 42 99 83 00 **</FAX>**

Fig. 4. Example of an automatically tagged job offer

of Paris itself, he will be successful because of the produced error. But if the user tended to seek after work outside of Paris, he would have no chance to locate this offer. The same was true for synonymous descriptions of the mentioned phrase. So, the user could have typed the query *"IDF" or "Île de France"*, the French name for the greater area of Paris, which should be mapped to *"à l'extérieur de Paris"*, a paraphrase describing the same place. Meanwhile, our system can deal with these linguistic phenomena.

The undiscovered sequences, missed by our local grammars, are gray highlighted. The example shows that the "employment date" paraphrased by two statements like *"vous devez impératiement être disponible sous 1 à 4 semaines" (You should be available within 1 or 4 weeks.)* and *"... pourrait démarrer une mission très rapidement" (... the occupation could possibly start pretty soon.)* could not at all be located. This lack of information has already been eliminated in a new version of the system and the missing structures were included in the corresponding local grammars. In this manner, the local grammars are continuously expanding which can be realized without great effort thanks to the intuitive Unitex interface [14].

A great advantage of local grammars is the clarity of the already prepared but still missing syntactic structures. If a piece of information was not completely extracted, the missing sub-path can be added quickly to the graphs because of the grammars' high level of modularity.

4 Evaluation

To evaluate the quality of our system with regards to the recognition of information bits indicating job offers, subsequently to the completion of the system, we designed a small, manually annotated test corpus composed of approximately 1,000 job offers.[7] This corpus allows us to provide values for recall and precision of the automatically retrieved search results.

Extracted type of information	**Precision**	**Recall**
Job description	96.9 %	93.3 %
Company name	94.3 %	90.6 %
Office (address)	93.0 %	92.3 %
Salary information	97.1 %	91.8 %
Workspace	98.0 %	96.9 %
On average	95.9 %	93.0 %

Table 3. Evaluation results gained on test corpora

Table 3 shows promising results of precision (95.9% on average) and recall (93.0% on average) considering the recognition of entities typically found in job offers. The experimental evaluation presented in this paper was limited to the five highest weighted of the 13 semantic classes. The majority of difficulties could be observed for the explicit identification of company names.

[7] For research purposes the annotated test corpus is available at `http://www.cis.uni-muenchen.de/~sandrab/DA/IE-Korpus1.html`

5 Conclusion

We presented an integrated platform to enable job search on the Internet. Apart from an overview of the location of job documents on the Web, the main focus of this paper is document analysis and information extraction. The core technique to automatically extract structured information from text is the use of local grammars. The evaluation on an annotated corpus shows excellent results for the proposed approach.

Though the linguistic descriptors and the examples of job advertisements refer to the French job market, the methods are generalizable to other language environments: a system working with equally expressive local grammars will show significant advantages as compared to a mere keyword approach.

References

1. Fondeur, Y., Tuchszirer, C.: Internet et les intermédiaires du marché du travail. In: La lettre de l'IRES. Number 67. IRES - Institut de Recherches Economiques et Sociales, Noisy-le-Grand, France (2005)
2. Focus RH: Le guide des 500 meilleurs sites emploi. Jeunes Editions, Levallois-Perret, France (2006)
3. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Tokyo (1983)
4. Gross, M.: Local grammars and their representation by finite automata. In Hoey, M., ed.: Data, Description, Discourse, Papers on the English Language in honour of John McH Sinclair. Harper-Collins, London (1993) 26–38
5. Gross, M.: The Construction of Local Grammars. In Roche, E., Schabès, Y., eds.: Finite-State Language Processing. Language, Speech, and Communication, Cambridge, Mass.: MIT Press (1997) 329–354
6. Courtois, B., Silberztein, M.: Dictionnaires électroniques du français. Langues française **87** (1990) 11–22
7. Courtois, B., Garrigues, M., Gross, G., Gross, M., Jung, R., Mathieu-Colas, M., Silberztein, M., Vivès, R.: Dictionnaire électronique des noms composés DELAC : les composants NA et NN. Rapport Technique du LADL **55** (1997)
8. Gross, M.: A bootstrap method for constructing local grammars. In: Contemporary Mathematics: Proceedings of the Symposium, University of Belgrad, Belgrad (1999) 229–250
9. Senellart, J.: Locating noun phrases with finite state transducers. In: Proceedings of the 17th International Conference on Computational Linguistics, Montréal (1998) 1212–1219
10. Mallchok, F.: Automatic Recognition of Organization Names in English Business News. PhD thesis, Ludwig-Maximillians-Universität Munich, Munich, Germany (2004)
11. Harris, Z.S.: Mathematical Structures of Language. Interscience Tracts in Pure and Applied Mathematics **21** (1968) 230–238
12. Harris, Z.S.: Language and Information. Bampton Lectures in America **28** (1988) 120–128
13. Friburger, N., Maurel, D.: Elaboration d'une cascade de transducteurs pour l'extraction des noms personnes dans les textes. In: TALN 2001, Tours (2001)
14. Paumier, S.: Manuel d'utilisation d'Unitex. (2004)

ExpLSA: An Approach Based on Syntactic Knowledge in Order to Improve LSA for a Conceptual Classification Task

Nicolas Béchet and Mathieu Roche and Jacques Chauché

LIRMM - UMR 5506 - CNRS, Univ. Montpellier 2,
34392 Montpellier Cedex 5 - France

Abstract. Latent Semantic Analysis (LSA) is nowadays used in various thematic like cognitive models, educational applications but also in classification. We propose in this paper to study different methods of proximity of terms based on LSA. We improve this semantic analysis with additional semantic information using Tree-tagger or a syntactic analysis to expand the studied corpus. We finally apply LSA on the new expanded corpus.

1 Introduction

Classification's domain has many research fields like conceptual classification. This one consists in gathering terms in concepts defined by an expert. For example, *exhaust pipe, windshield wiper*, and *rearview mirror* terms can be associated to the *automobile* concept. Then, these terms are classified by semantic proximity with different algorithms like k nearest neighbor (KNN) or k means. The corpora have different types as the language, the syntax, the domain (biology, medicine, etc) using a specialized semantic, etc. Then these complex textual data require a specific process.

In this paper, we describe the first step of a conceptual classification, the study of proximity of the terms. First, we use the Latent Semantic Analysis (LSA) method evolved by [1][1]. LSA is a statistic method applied to high dimension corpora to gather terms (conceptual classification) or contexts (textual classification). After the latent semantic analysis application on the corpus, a semantic space associating each word to a vector is returned. Then, the proximity of the two words can be obtained by measuring the cosine between two vectors. Our aim is to improve the performance of the LSA method by an approach named *ExpLSA*.

The *ExpLSA* approach (context **Exp**ansion with **LSA**) consists in expanding the corpus before the application of a "traditional" latent semantic analysis. This context expansion uses semantic knowledge obtained by syntax, what allows to use *ExpLSA* as well specialized corpus as not. Actually, it is not necessary to use training corpus, so it is not necessary to know the general domain of the corpus.

[1] http://www.msci.memphis.edu/∼wiemerhp/trg/lsa-followup.html

Received 21/10/07
Accepted 07/12/07
Final Version 11/01/08

In this paper, we use a Human Resources corpus of PerformanSe[2] company in French. It uses a specialized vocabulary. This corpus was submitted to a human expertise which validates our experiments.

We propose in the next section to explain theoretical characteristics of LSA method and its limits. Section 3 proposes a state-of-the-art in the field of the syntactic knowledge used with LSA. Then we present the different steps of our method (section 4). We also describe (section 5) the experimental protocol applied. Finally we will present results obtained.

2 LSA

The LSA method is based on the fact that words which appear in the same context are semantically close. Corpus is represented by a matrix. Lines are relating to words and columns are the several contexts (document, section, sentence, etc). Every cells of matrix represent the number of words in the contexts. Two semantically close words are represented by close vectors. The proximity measure is generally defined by the cosine between the two vectors.

LSA is based on the Singular Value Decomposition (SVD) theory. $A = [a_{ij}]$ where a_{ij} is the frequency of the word i in the context j, breaking down in a product of three matrices USV^T. U and V are orthogonal matrices and S a diagonal matrix.

Let us S_k where $k < r$ the matrix built by removing of S the $r - k$ columns which have the smallest singularly values. We take U_k and V_k, matrices obtained by removing corresponding columns of U and V matrices. Then, the $U_k S_k V_k^T$ can be considered like a compressed version of the original matrix A. Experiments presented in the section 5 were made with a factor k equal to 200.

Before performing a singularly value decomposition, a first step of normalization of the original matrix A is applied. This normalization consists in applying a logarithm and an entropy measure on the matrix A. This transformation allows to refer the weight of the words on their contexts. In a similar way to the work of [2], this normalization can also refer on the tf×idf method, well-known approach in the field of the Information Retrieval (IR).

Also let us specify that we do not consider punctuations and some words not relevant in a semantical point of view like: “and”, “a”, “with”, etc. (stop words).

LSA gives many advantages like notions of independence of the language of the corpus. It needs no language or domain knowledge. However, LSA has limits. Firstly, we consider size of chosen contexts. [3] showed during its experiments that if contexts have less than 60 words, the results can be disappointing. In addition, the efficiency of LSA is weak with a proximity of the vocabulary used. For example, a very precise classification of texts based on very close domains can be difficult with LSA.

[2] http://www.performanse.fr/

3 State-of-the-art on the addition of syntax to LSA

[4] presents the problem of the lack of syntactic knowledge with LSA method. They compare their methods to a human evaluation. They propose to human experts to evaluate essays of 250 words on the human heart writing by students. A semantic space have been built from 27 papers about human heart learned by LSA. Tests performed give good results for the LSA method comparing to the human expertise. Bad results was the consequence of a small paucity of syntactic knowledge in the approach used. Thus, the work below demonstrates how these knowledge can be added to LSA.

The first approach of [5] uses the Brill tagger [6] to assign a part-of-speech tag to every word. The tags are attached to each word with an underscore. So LSA can consider each word/tag combination as a single term. Results of similarity calculation with such method stay disappointing. The second approach of [5] is characterized by the use of a syntactic analysis in order to segment text before applying the latent semantic analysis. This approach is called Structured LSA (SLSA). A syntactic analysis of sentences based on different elements (subject, verb, and object) is firstly made. Then, similarity scores (obtained by a cosine computing) between the vectors of the three matrices obtained by LSA are evaluated. The average of the similarities is finally computed. This method gave satisfactory results compared to "traditional LSA".

The approach described in [7] proposes a model called SELSA. Instead of generating a matrix of co-occurrences word/document. A matrix where each line contains all the combinations of words_tags, and a column represents a document. The label "prefix" informs about the syntactic type of the word neighborhood. The principle of SELSA is that the sense of a word is given by the syntactic neighborhood from which it results. This approach is rather similar to the use of the Brill tagger presented in [5]. But SELSA extends and generalizes this work. A word with a syntactic context specified by its adjacent words is seen as a unit knowledge representation. The evaluation shows that SELSA makes less errors than LSA but these errors are more harmful.

The *ExpLSA* approach presented in this paper is placed in a different context. In fact, in our studies, the contexts are represented by sentences. These ones have a reduced size which tends to give low results with LSA [3]. In our approach, we propose to use the regularity of some syntactic relations in order to expand the context as described in the next section.

4 Our approach: *ExpLSA*

The final aim consists in automatically gathering terms extracted by systems adapted to French corpora such as ACABIT [8], SYNTEX [9], EXIT [10]. In our case, we propose to gather nominal terms extracted with EXIT from the Human Resources corpus (written in French). The extracted terms with this system are phrases respecting the syntactical patterns (noun-prep-noun, adj-noun, noun-adj, etc). In addition, EXIT is based on a statistical method to rank terms extracted. It uses an iterative approach to build complex terms.

The first step of the conceptual classification can be done by *ExpLSA*. Its principle is described in the following sections.

4.1 General principle of ExpLSA

Our approach makes an expansion of lemmatized sentences based on a syntactic method. It comes out a richer context by completing words of the corpus by words considered semantically similar. We have for example the sentence (in French): *"Vos interlocuteurs seront donc bien inspirés..."*. We transform it firstly in a lemmatized sentence: *"Votre interlocuteur être donc bien inspiré..."*. Then, it will be expanded by other terms. This sentence becomes *"Votre (interlocuteur collaborateur) être donc bien inspiré..."*. This set of terms semantically close are selected with a measure presented in the next section.

4.2 The syntactic analysis to evaluate the semantic proximity

In order to improve initial corpus, we propose firstly to apply a syntactic analysis with the SYGMART French parser [12]. This one gives the existing syntactic relations in each sentence. In our approach, we only used the Verb-Object relations (Verb_DO, Verb_Preposition_Complement) of the corpus.

When all Verb-Object relations are extracted, we use a measure to evaluate the semantic proximity of words, the Asium measure [13]. This one proposes to evaluate verbs considered as close if they have a significant number of common features. The principle of the approach is similar to the method presented in [11].

We consider verbs, the associated prepositions and features after a syntactic parsing. The Asium measure consists in computing a similarity measure between verbs.

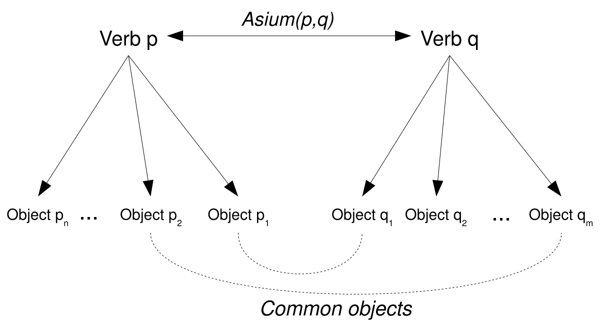


Fig. 1. The Asium measure between p and q verbs.

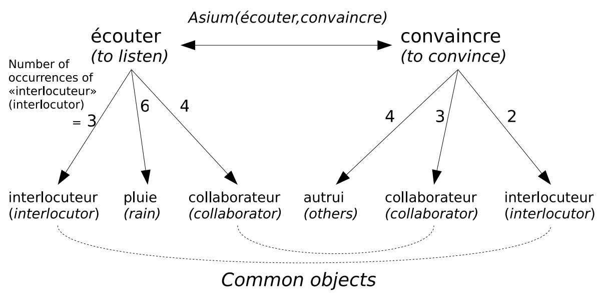
We consider the p and q verbs with their respective p_1,...,p_n and q_1,...,q_n objects illustrated in the figure 1. $NbOC_p(q_i)$ is the occurrences number of q_i

object of q and p verbs (common objects). $NbO(q_i)$ is the occurrences number of q_i objects of q verb. Then, the Asium measure is:

$$Asium(p,q) = \frac{log_{Asium}(\sum NbOC_q(p_i)) + log_{Asium}(\sum NbOC_p(q_i))}{log_{Asium}(\sum NbO(p_i)) + log_{Asium}(\sum NbO(q_i))}$$

Where $log_{Asium}(x)$ equal to :
- for $x = 0$, $log_{Asium}(x) = 0$
- else $log_{Asium}(x) = log(x) + 1$

The example of the figure 2 gives an application of the Asium measure for the *écouter (to listen)* and *convaincre (to convince)* verbs. Next, we consider different similarity threshold values of the Asium measure. Over this one, verbs are considered as close by the Asium measure. The expansion method is described in the next section.


Asium(écouter,convaincre)
écouter
(to listen)
convaincre
(to convince)
Number of occurrences of «interlocuteur» (interlocutor) = 3
6
4
4
3
2
interlocuteur (interlocutor)
pluie (rain)
collaborateur (collaborator)
autrui (others)
collaborateur (collaborator)
interlocuteur (interlocutor)
Common objects

Asium(écouter,convaincre)

$$= \frac{\log_{Asium}(4+3)+\log_{Asium}(3+2)}{\log_{Asium}(3+6+4)+\log_{Asium}(4+3+2)}$$

$$= 0{,}87$$

Fig. 2. The Asium measure between the verbs *écouter (to listen)* and *convaincre (to convince)*

4.3 The steps of ExpLSA

After the description of the Asium measure, we propose to clarify the different steps of *ExpLSA* to expand the contexts.

The first step of the *ExpLSA* approach identifies the different terms extracted by Exit. Then a term is represented like a single word (for instance, the *attitude profondément participative* term becomes *noun234* which is the 234th term of a list extracted by Exit).

After the extraction of the syntactic Verb-Object relations by using a syntactic analysis, the next step of our approach is to study the semantic proximity between verbs using the Asium measure. We deduced for example that verbs *écouter (to listen)* and *convaincre (to convince)* are semantically close by using the Asium measure because they have the common objects *interlocuteur (interlocutor) and collaborateur (collaborator)* (See figure 2).

The next step proposes to gather the objects of the closest semantically verbs. Then, we consider two gathering methods. The first method consists in completing corpus with common words of the two verbs (*interlocuteur* and *collaborateur* in the example of the figure 2). The second method is to consider the common and the complementary objects of the two verbs to expand corpus like the work of Faure and Nedellec [17] (*entourage, autrui, pluie, collaborateur* in the example of the figure 2).

Then we propose to expand initial corpus by catching to each word, the other words judged close. Then, our initial French sentence:
- *Votre* ***interlocuteur*** *être donc bien inspiré...*

becomes with the first gathering method:
- *Votre* ***(interlocuteur collaborateur)*** *être donc bien inspiré...*

and becomes with the second gathering method:
- *Votre* ***(autrui pluie interlocuteur collaborateur)*** *être donc bien inspiré...*

Latent Semantic Analysis can be applied with the expanded corpus.

A list of general words with a poor semantic content are not selected to expand the context (for instance, general words like "chose" (*thing*), "personne" (*person*), etc). This list is manually made.

The evaluation presented in the next section compares results obtained with *ExpLSA* with results of the experts who had manually associated relevant terms to the concepts.

5 Experiments

To discuss of the quality of the results returned by our approach, we describe the experimental protocol in the next section.

5.1 Experimental Protocol

In our experiments, we use a Human Resources corpus manually evaluated. This expertise gives a conceptual classification of all terms extracted by Exit. The concepts are defined by the expert. For instance, with our corpus, the expert defined "Relationnel" (*relational*) concept. The *confrontation ouverte (open adversarial)*, *contact superficiel (superficial contact)*, and *entourage compréhensif (understanding circle)* terms are instances of this concept. Our aim is to evaluate similarities obtained with the following methods:

- *M1:* LSA
- *M2:* the intersections method of *ExpLSA*
- *M3:* the complementaries method of *ExpLSA*
- *M4:* LSA + Tree-tagger

The LSA + Tree-tagger method consists in using the Tree-tagger [18]. We use a part-of-speech tagger like the approach of [5] developed in the section 3.

To compare these methods, we select the most representative concepts, *i.e.* concepts gathering a minimum of 200 distinct terms given by the expertise. We obtain four concepts ($C1$, $C2$, $C3$, and $C4$). Twenty terms of each concept which have the most number of occurrences are selected. Then we measure the similarity (with the cosine) of the couples of terms of the concepts 2-2 (terms of $C1$ and $C2$ concepts, $C1$ and $C3$ concepts, etc). For example, the similarity measure is computed between the terms of the $C1$ concept with the terms of the $C1 + C2$, $C1 + C3$, and $C1 + C4$ concepts. A couple of terms is called relevant if they are an instance of the same concept.

The experiments consist in ranking these couples of terms based on the similarity measure using the M1, M2, M3 (section 5.2), and M4 (section 5.3) methods. Then we compute the precision for the n first couples of terms. Precision gives the proportion of relevant couples returned by the method. The recall has not judged adapted[3].

5.2 The *ExpLSA* methods comparison

We propose a first experiment comparing LSA with both methods of *ExpLSA* used to expand the corpus: The complementaries and the intersections approaches. The table 1 gives average precision of the 100 first couples of terms (for each couple of concepts evaluated) with a threshold of the Asium measure at 0.6. This value of threshold makes a large expansion of the corpus. This table indicates better results for the *ExpLSA* with the intersections method (M2 method) except for the "activity-environment" couple of concepts. But these improvements are mitigate; we have a large majority of cases where the LSA method is not improved because an Asium threshold to 0.6 makes a large expansion with a lot of noise added. We conclude experimentally that the better threshold value of *ExpLSA* is 0.8. It is a good compromise between quantity and quality of expanded knowledge. The table 2 is based on an Asium threshold to 0.8. This one gives better results for the intersections approach (M2 method) in comparison with LSA. The figure 3 confirms the weak results for the complementaries approach. We only conserve M2 method in the next section.

[3] Our approach allows to obtain relevant couples at the beginning of the list. Nevertheless, in a case of few couples are considered ($n < 100$), recall is naturally week and not adapted to evaluate the performance of our approach.

		The n first terms								
Pairs of concepts	*Method*	20	30	40	50	60	70	80	90	100
Activity	M1	**0,551**	0,543	0,536	0,527	0,520	0,518	0,520	0,524	0,521
Behaviour	M2	0,539	**0,553**	**0,545**	**0,541**	**0,545**	**0,546**	**0,537**	**0,534**	**0,528**
	M3	0,533	0,519	0,509	0,521	0,524	0,520	0,516	0,512	0,511
Activity	M1	0,520	**0,528**	0,515	0,518	0,514	0,513	0,513	0,512	0,514
Environment	M2	**0,548**	0,519	**0,524**	0,519	0,513	0,512	0,511	0,508	0,510
	M3	0,538	0,526	0,521	**0,524**	**0,524**	**0,524**	**0,521**	**0,518**	**0,520**
Activity	M1	**0,580**	**0,567**	0,559	0,546	**0,546**	0,536	**0,540**	**0,545**	**0,543**
Relational	M2	0,566	0,563	**0,561**	**0,548**	0,545	**0,539**	**0,540**	0,541	0,538
	M3	0,549	0,538	0,529	0,526	0,528	0,519	0,520	0,519	0,522
Behaviour	M1	**0,586**	**0,566**	**0,556**	**0,546**	**0,542**	**0,538**	**0,533**	**0,533**	**0,531**
Environment	M2	0,526	0,522	0,513	0,519	0,521	0,518	0,513	0,513	0,512
	M3	0,539	0,522	0,523	0,525	0,521	0,524	0,523	0,525	0,523
Behaviour	M1	**0,555**	**0,548**	**0,539**	0,528	0,523	**0,523**	**0,524**	**0,525**	**0,526**
Relational	M2	0,540	0,530	0,527	**0,531**	**0,524**	**0,523**	0,523	0,517	0,519
	M3	0,523	0,516	0,509	0,502	0,507	0,506	0,512	0,516	0,519
Environment	M1	0,559	0,550	0,548	**0,560**	**0,544**	**0,539**	**0,539**	**0,540**	**0,537**
Relational	M2	**0,560**	**0,558**	**0,551**	0,538	0,532	0,529	0,526	0,526	0,526
	M3	0,538	0,506	0,506	0,519	0,520	0,521	0,516	0,518	0,515

Table 1. Precision of the 100 first couples. Method M1 = LSA, Method M2 = ExpLSA with the intersections method, M3 = ExpLSA with the complementaries method. Asium threshold = 0.6.

		The n first terms								
Pairs of concepts	*Method*	20	30	40	50	60	70	80	90	100
Activity	M1	0,551	0,543	0,536	0,527	0,520	0,518	0,520	0,524	0,521
Behaviour	M2	**0,558**	**0,564**	**0,558**	**0,551**	**0,542**	**0,541**	**0,534**	**0,531**	**0,530**
	M3	0,514	0,522	0,524	0,529	0,529	0,518	0,523	0,520	0,516
Activity	M1	0,520	0,528	0,515	0,518	0,514	0,513	0,513	0,512	0,514
Environment	M2	**0,531**	**0,536**	**0,537**	**0,535**	**0,534**	**0,530**	**0,525**	**0,523**	**0,518**
	M3	0,519	0,506	0,506	0,504	0,501	0,500	0,496	0,500	0,500
Activity	M1	**0,580**	0,567	**0,559**	0,546	**0,546**	0,536	**0,540**	**0,545**	**0,543**
Relational	M2	0,576	**0,572**	0,555	**0,549**	0,545	**0,546**	0,539	0,534	0,534
	M3	0,546	0,549	0,548	0,536	0,530	0,523	0,527	0,522	0,516
Behaviour	M1	0,586	0,566	0,556	0,546	0,542	0,538	0,533	0,533	0,531
Environment	M2	**0,600**	**0,583**	**0,573**	**0,572**	**0,560**	**0,551**	**0,545**	**0,547**	**0,543**
	M3	0,526	0,520	0,511	0,517	0,513	0,514	0,511	0,513	0,511
Behaviour	M1	**0,555**	**0,548**	**0,539**	0,528	0,523	0,523	0,524	0,525	0,526
Relational	M2	0,545	0,528	0,536	**0,539**	**0,537**	**0,534**	**0,531**	**0,531**	**0,527**
	M3	0,509	0,515	0,522	0,522	0,519	0,518	0,514	0,511	0,508
Environment	M1	0,559	0,550	0,548	**0,560**	0,544	0,539	0,539	**0,540**	0,537
Relational	M2	**0,579**	**0,563**	**0,559**	0,555	**0,548**	**0,546**	**0,543**	0,539	**0,539**
	M3	0,488	0,486	0,493	0,498	0,497	0,498	0,496	0,494	0,497

Table 2. Precision of the 100 first couples. Method M1 = LSA, Method M2 = ExpLSA with the intersections method, M3 = ExpLSA with the complementaries method. Asium threshold = 0.8.

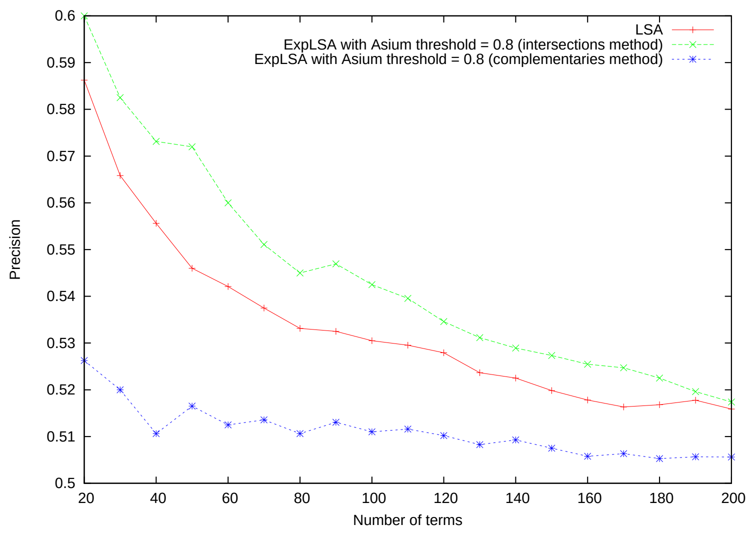


Fig. 3. Precision of the 200 first couples comparing the *ExpLSA* methods between concepts behavior and environment

5.3 ExpLSA compared to the LSA + Tree-tagger approach

In this section we use the syntactic knowledge in two ways: *ExpLSA* and LSA + Tree-tagger. LSA + Tree-tagger adds grammatical knowledge (part-of-speech category) to the corpus by assign a part-of-speech tag to the words. LSA is applied with this tagged corpus. We see in the table 3 and in the figure 4 that the precision of *ExpLSA* is better for the first couples. This is the researched result because that means the first couples provided at the expert are relevant. These encouraging results[4] of *ExpLSA* and LSA + Tree-tagger methods allow to consider a hybrid approach by combine both.

5.4 ExpLSA applied to Text Categorization

In recent works, we proposed to evaluate the *ExpLSA* method for a task of Text Categorization (TC). The aim of these works was to improve TC tasks: SVM (Support Vector Machines), k-NN (k nearest neighbor), Naive Bayes, and Decision Tree (C4.5) algorithms [19] using LSA and *ExpLSA* representations. We use a French news corpus which contains 2828 articles (5,3 MB) provided by yahoo's French web site (http://fr.news.yahoo.com/). In these experiments we calculate the average F-measure by performing a ten-fold cross-validation. We obtained good results of *ExpLSA* representation with the application of SVM,

[4] These results are a little different comparing to results presented in [16] because we proposed a different experimental protocol by considering every words of concept and not only the twenty most common words.

		The n first terms								
Pairs of concepts	*Method*	20	30	40	50	60	70	80	90	100
Activity Behaviour	M1	0,551	0,543	0,536	0,527	0,520	0,518	0,520	0,524	0,521
	M2	**0,558**	**0,564**	**0,558**	**0,551**	**0,542**	**0,541**	0,534	0,531	0,530
	M4	0,551	0,535	0,542	0,547	0,541	0,540	**0,538**	**0,535**	**0,533**
Activity Environment	M1	0,520	0,528	0,515	0,518	0,514	0,513	0,513	0,512	0,514
	M2	**0,531**	**0,536**	**0,537**	**0,535**	**0,534**	**0,530**	**0,525**	**0,523**	0,518
	M4	0,505	0,508	0,505	0,505	0,509	0,513	0,518	0,522	**0,525**
Activity Relational	M1	**0,580**	0,567	0,559	0,546	0,546	0,536	0,540	0,545	0,543
	M2	0,576	**0,572**	0,555	0,549	0,545	0,546	0,539	0,534	0,534
	M4	0,559	**0,572**	**0,572**	**0,567**	**0,560**	**0,559**	**0,553**	**0,551**	**0,550**
Behaviour Environment	M1	0,586	0,566	0,556	0,546	0,542	0,538	0,533	0,533	0,531
	M2	**0,600**	**0,583**	**0,573**	**0,572**	**0,560**	**0,551**	0,545	0,547	0,543
	M4	0,563	0,561	0,561	0,565	0,553	0,549	**0,552**	**0,550**	**0,550**
Behaviour Relational	M1	0,555	**0,548**	0,539	0,528	0,523	0,523	0,524	0,525	0,526
	M2	0,545	0,528	0,536	0,539	**0,537**	**0,534**	**0,531**	**0,531**	**0,527**
	M4	**0,574**	0,545	**0,542**	**0,542**	0,536	0,530	0,529	0,525	0,523
Environment Relational	M1	0,559	0,550	0,548	**0,560**	0,544	0,539	0,539	**0,540**	0,537
	M2	**0,579**	0,563	**0,559**	0,555	**0,548**	**0,546**	**0,543**	0,539	**0,539**
	M4	0,573	**0,565**	0,554	0,550	0,544	0,536	0,535	0,531	0,530

Table 3. Precision of the 100 first couples comparing the LSA (M1), the *ExpLSA* with Asium threshold = 0.8 (M2) and LSA + Tree-tagger approach (M4)

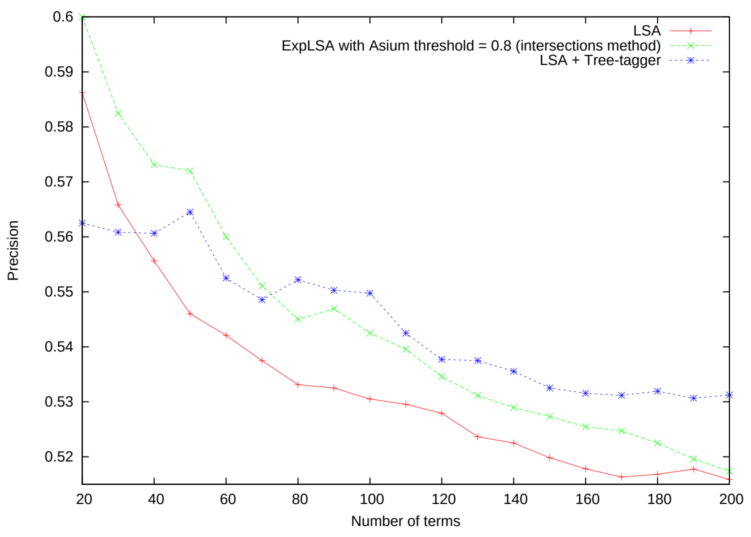


Fig. 4. Precision of the 200 first couples comparing the ExpLSA and LSA + Tree-tagger methods between concepts behavior and environment

k-NN, and C4.5 on medium and long articles of the corpus. In order to improve the classification task of short documents, we will improve quality of expansion using a validation based on the web knowledge.

6 Conclusion and discussion

LSA is a statistical method which can be used to gather terms to build a conceptual classification. However, this method gives medium results in this domain. We can explain these results by the absence of the syntactic knowledge. Quality of results can also be influenced by the size of contexts used. The LSA method gives better results with an important size of contexts.

Then, we propose to improve the LSA performances with small contexts (sentences) by an approach called *ExpLSA*. This one consists in applying a context expansion with LSA by expand the original corpus. We use syntactical resources to make these contexts expansion.

We presented two experiments to compare the *ExpLSA* approach. First, we experiment the *ExpLSA* approach based on the complementaries approach. This expansion is not relevant because this method returns a lot of noise. We consider the Asium threshold value to 0.8, which is the best value experimentally found. The intersections method of *ExpLSA* gives good results. We compare it with another method in the second experiment, the LSA + Tree-tagger approach.

In a future work, we propose firstly to adapt a hybrid approach combining the *ExpLSA* (with the intersections approach) and the LSA + Tree-tagger methods. We envisage to add other set of syntactic knowledge to improve LSA. Finally, we would want to apply *ExpLSA* in order to use other classification tasks (for instance, the text classification task).

References

1. Landauer, T., Dumais, S.: A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge. Psychological Review, Vol. 104 (1997) 211–240
2. Turney P.D.: Mining the Web for Synonyms: PMI–IR versus LSA on TOEFL. Proceedings of ECML'01, Lecture Notes in Computer Science (2001) 491–502
3. Rehder B., Schreiner M., Wolfe M., Laham D., Landauer T., Kintsch W.: Using Latent Semantic Analysis to assess knowledge: Some technical considerations. Discourse Processes, Vol. 25 (1998) 337–354
4. Landauer T., Laham D., Rehder B., Schreiner M. E.: How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. Proceedings of the 19th annual meeting of the Cognitive Science Society (1997)
5. Wiemer-Hastings P., Zipitria I.: Rules for syntax, vectors for semantics. Proceedings of the Twenty-third Annual Conference of the Cognitive Science Society (2001)
6. Brill E.: Some Advances in Transformation-Based Part of Speech Tagging AAAI, Vol. 1 (1994) 722–727

7. Kanejiya D., Kumar A., Prasad S.: Automatic Evaluation of Students' Answers using Syntactically Enhanced LSA. Proceedings of the Human Language Technology Conference (HLT-NAACL 2003) Workshop on Building Educational Applications using NLP (2003)
8. Daille, B.: Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. In: Resnik, P.; Klavans, J. (eds.): The Balancing Act: Combining Symbolic and Statistical Approaches to Language, MIT Press, Cambridge, MA, USA, (1996) 49–66.
9. Bourigault D., Fabre C.: Approche linguistique pour l'analyse syntaxique de corpus. Cahiers de Grammaires, Vol. 25 (2000) 131–151
10. Roche M., Heitz T., O. Matte-Tailliez O., Kodratoff Y.: Exit: Un système itératif pour l'extraction de la terminologie du domaine à partir de corpus spécialisés. Proceedings of JADT'04, Vol. 2 (2004) 946–956
11. Bourigault D.: UPERY : un outil d'analyse distributionnelle étendue pour la construction d'ontologies à partir de corpus. Actes de TALN, Nancy (2002) 75–84
12. Chauché J.: Un outil multidimensionnel de l'analyse du discours. Proceedings of Coling, Standford University, California (1984) 11–15
13. Faure D.: Conception de méthode d'apprentissage symbolique et automatique pour l'acquisition de cadres de sous-catégorisation de verbes et de connaissances sémantiques à partir de textes : le système ASIUM. Phd, Université de Paris Sud (2000)
14. Ferri C., Flach P., Hernandez-Orallo J.: Learning decision trees using the area under the ROC curve Proceedings of ICML'02 (2002) 139–146
15. Mary J.: Étude de l'Apprentissage Actif : Application à la conduite d'expériences. Phd, Université Paris Sud (2005)
16. Béchet N., Roche M., Chauché J.: *ExpLSA* : utilisation d'informations syntaxico-sémantiques associées à LSA pour améliorer les méthodes de classification conceptuelle. Proceedings of EGC'08.
17. Faure D., Nedellec C.: Knowledge acquisition of predicate argument structures from technical texts using machine learning: The system ASIUM. Proceedings of the 11th European Workshop, Knowledge Acquisition, Modelling and Management, number 1937 in LNAI (1999) 329-334
18. Schmid H. Improvements in part-of-speech tagging with an application to German. Technical report, Universitat Stuttgart. Institut fur maschinelle Sprachverarbeitung. (Revised version of a paper presented at EACL SIGDAT, Dublin 1995).
19. Y. Yang An Evaluation of Statistical Approaches to Text Categorization. Information Retrieval, number 1-2, volume 1 (1999) 69–90

Gathering Definition Answers by Information Gain

Carmen Martínez-Gil[1] and A. López-López[1]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica,
Luis Enrique Erro #1, Santa María Tonanzintla, Puebla, 72840, México
{Carmen, allopez}@ccc.inaoep.mx

Abstract. A definition question is a kind of question whose answer is a complementary set of sentence fragments called nuggets, which define the target term. Since developing general and flexible patterns with a wide coverage to answer definition questions is not feasible, we propose a method using information gain to retrieve the most relevant information. To obtain the relevant sentences, we compared the output of two retrieval systems: JIRS and Lucene. One important feature that impacts on the performance of definition question answering systems is the length of the sentence fragments, so we applied a parser to analyze the relevant sentences in order to get clauses. Finally, we observed that, in most of the clauses, only one part before and after the target term contains information that defines the term, so we analyzed separately the sentence fragments before (*left*) and after (*right*) the target term. We performed different experiments with the collections of questions from the *pilot* evaluation of definition questions 2002, *definition* questions from TREC 2003 and *other* questions from TREC 2004. F-measures obtained are competitive when compared against the participating systems in their respective conferences. Also the best results are obtained with the general purpose system (Lucene) instead of JIRS, which is intended to retrieve passages for factoid questions.

1 Introduction

Question Answering (QA) is a computer-based task that tries to improve the output generated by Information Retrieval (IR) systems. A definition question [9] is a kind of question whose answer is a complementary set of sentence fragments called nuggets.

After identifying the correct target term (the term to define) and context terms, we need to obtain useful and non redundant definition nuggets. Nowadays, patterns are obtained manually as surface patterns [5, 6, 12]. These patterns can be very rigid, leading to the alternative soft patterns [2], which are even extracted in an automatic way [5]. Then, once we have the patterns, we apply a matching process to extract the nuggets. Finally, we need to perform a process to determine if these nuggets are part of the definition; where a common criterion employed is the frequency of appearance of the nugget.

According to the state of the art, the highest F-measure in a pilot evaluation [9] for definition questions in 2002 is 0.688 using the nugget set supplied by author, taking

Received 15/10/07
Accepted 07/12/07
Final Version 11/01/08

β=5. For the TREC 2003 [10], the best F-measure was 0.555 also with β=5, and the TREC 2004 [11] F-measure was 0.460, now with β=3.

In contrast to the traditional way to extract nuggets, we propose a method that uses information gain to retrieve the most relevant information. First, we obtain passages from the AQUAINT Corpus using the retrieval system Lucene[1]. Next, from the passages, we extract the relevant sentences, these are further parsed (using Link Grammar [4]) to obtain clauses. Then, from the clauses, we select four kinds of sentence fragments, these are: noun phrases containing an appositive phrase, noun phrases containing two noun phrases separated by comma, embedded clauses, and main or subordinate clauses without considering embedded clauses. Finally, the sentence fragments are separated in two kinds of fragments, i.e. the fragments to the *left* and *right* of the target term. We then assess the information gain of sentence fragments to decide which are the most relevant, and in consequence select them as part of the final answer.

For this task, we work with the questions of the *pilot* evaluation of definition questions 2002 [9], *definition* questions from TREC 2003 [10] and *other* questions from TREC 2004 [11]. First, we test the output of two retrieval systems JIRS[2] and Lucene. In the second experiment, we test balanced and non-balanced sets of sentence fragments from the *right* and *left* sets. Finally, we compare the F-measure obtained with our system DefQuestions_IG against the participating systems in the TREC conferences.

The paper is organized as follows: next section describes the process to extract sentence fragments; Section 3 describes the approaches used and the method to retrieve only definition sentence fragments; Section 4 reports experimental results; finally, some conclusions and directions for future work are presented in Section 5.

2 Sentence Fragments Extraction

Official definition of F-measure used in the TREC evaluations [9] is:
Let

r : be the number of vital nuggets returned in a response
a : be the number of non-vital nuggets returned in a response
R : be the total number of vital nuggets in the assessors list
l : be the number of non-whitespace characters in the entire answer string

Then

$$recall(\Re) = r / R \tag{1}$$

$$allowance(\alpha) = 100 \times (r + a) \tag{2}$$

$$precision(\mathrm{P}) = \begin{cases} 1 & if \quad l < \alpha \\ 1 - \dfrac{l - \alpha}{l} & otherwise \end{cases} \tag{3}$$

[1] http://lucene.apache.org/
[2] http://jirs.dsic.upv.es/

Finally, for a given β, F-measure is: $F(\beta = 3) = \frac{(\beta^2 + 1) \times \mathrm{P} \times \Re}{\beta^2 \times \mathrm{P} + \Re}$ (4)

Thus, a reason to extract sentence fragments is that we need to retrieve only the most important information from relevant sentences. Other reason to extract short sentence fragments is related to the performance F-measure applied to definition systems in the TREC evaluation; this measure combines the recall and precision of the system. The precision is based on length (in non-white-space characters) used as an approximation to nugget precision. The length-based measure starts from an initial allowance of 100 characters for each (vital or non-vital) nugget matched. Otherwise, the measure value decreases as the length the sentence fragment increases.

After our experiments comparing two retrieval systems (and detailed later on), we decide to use Lucene as main system to extract candidate paragraphs from the AQUAINT Corpus of English News Text. From these candidate paragraphs, we extract the relevant sentences, i.e. the sentences that contain the target term. Then, to extract sentence fragments we propose the following process:

1) **Parse the sentences.** Since we need to obtain information segments (phrases or clauses) from a sentence, the relevant sentences were parsed with Link Grammar [6]. We replace the target term by the label **SCHTERM**. As an example, we get the following sentence for the target term **Carlos the Jackal**:

```
The man known as Carlos the Jackal has ended a hunger
strike after 20 days at the request of a radical Pales-
tinian leader, his lawyer said Monday.
```

The Link Grammar the produces the following output with the target term replaced as detailed above:

```
[S [S [NP [NP The man NP] [VP known [PP as [NP
SCHTERM NP] PP] VP] NP] [VP has [VP ended [NP a
hunger strike NP] [PP after [NP 20 days NP] PP]
[PP at [NP [NP the request NP] [PP of [NP a radi-
cal Palestinian leader NP] PP] NP] PP] VP] VP] S]
, [NP his lawyer NP] [VP said [NP Monday NP] . VP]
S]
```

2) **Resolve co-references.** We want to obtain main clauses without embedded clauses or only embedded clauses, so we need to resolve the co-reference, otherwise important information can be lost. To resolve co-reference the relative pronouns WHNP are replaced with the noun phrase preceding it.

3) **Obtain sentence fragments.** An information nugget or an atomic piece of information can be a phrase or a clause. We analyzed the sentences parsed with Link Grammar and we identified four main kinds of sentence fragments directly related to the target and with a high possibility that their information define the target. These fragments are:

Noun phrase (NP) containing an appositive phrase.
Noun phrase (NP) containing two noun phrases separated by comma [NP, NP].
Embedded clauses (SBAR).
Main or subordinate clauses (S) without considering embedded clauses.

To retrieve the four kinds of sentence fragments, we analyze the parse tree, according to the following procedure:

i) Look for the nodes which contain the target, in our case, the label SCHTERM.

ii) Find the initial node of the sentence fragment. The process analyzes the path from the node with the SCHTERM label towards the root node. The process stops when a NP with appositive phrase, NP with [NP, NP], an embedded clause SBAR, or a clause S is reached.

iii) Retrieve the sentence fragment without embedded clauses.

iv) Mark as visited the parent node of the second phrase. For the case [NP1, NP2], we mark as visited the parent node of NP2. For appositive phrase, SBAR or S, the second phrase can be NP, VP or PP.

The steps ii to iv are repeated for the same node with a SCHTERM label until a visited node is found in the path to the node towards the root, or the root node is reached. Also the steps ii to iv are repeated for each node found in step i.

The next module of our definition question system selects definition sentence fragments. In order to select only definition nuggets from all of sentence fragments, we analyze separately, the information that is to the left of SCHTERM and the information that is to the right of SCHTERM, so we form two data sets.

Now, we present some sentence fragments of two sets (right and left) obtained using the process for the target term **Carlos the Jackal**:

```
Right sentence fragments

SCHTERM , a Venezuelan serving a life sentence in a French
prison

SCHTERM , nickname for Venezuelan born Ilich Ramirez Sanchez

SCHTERM , is serving a life sentence in France for murder

SCHTERM as a comrade in arms in the same unnamed cause

SCHTERM refused food and water for a sixth full day

SCHTERM , the terrorist imprisoned in France
```

```
Left sentence fragments

the friendly letter Chavez wrote recently to the terrorist
SCHTERM

The defense lawyer for the convicted terrorist known as SCHTERM

he was harassed by convicted international terrorist SCHTERM

an accused terrorist and a former accomplice of SCHTERM

Ilich Ramirez Sanchez , the terrorist known as SCHTERM

Ilich Ramirez Sanchez , the man known as SCHTERM
```

We found that analyzing separately the sentence fragments before and after the target term is an advantage since in many candidate sentences, only one part contains information defining the target term. When a fragment appears in both sides, this serves to affirm its informative feature, as assessed by information gain.

3 Nuggets Selection

In order to obtain only the most informative nuggets from the *left* and *right* sentence fragments, we use information gain.

3.1 Information Gain

The information gain [1] for each word or term *l* is obtained using the following definition:

Given a set of sentence fragments *D*, the entropy *H* of *D* is calculated as:

$$H(D) \equiv \sum_{i=1}^{c} - p_i \log_2 p_i \tag{5}$$

In this expression, P_i is the probability of *i* word, and *c* is the size of the vocabulary. Then, for each term *l*, let D^+ be the subset of sentence fragments of *D* containing *l* and D^- denotes its complement. The information gain of *l*, *IG(l),* is defined by:

$$IG(l) = H(D) - \left[\frac{|D^+|}{|D|} H(D^+) + \frac{|D^-|}{|D|} H(D^-) \right] \tag{6}$$

3.2 Method to Select Nuggets

The process to obtain informative nuggets using information gain is the following:

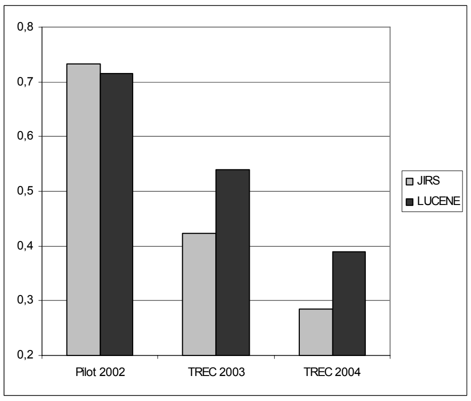
I) Obtain the vocabulary of all the sentence fragments (*left* and *right* sets).
II) Calculate the information gain for each word of the vocabulary using the definition of section 3.1.
III) Using the value of the information gain of each word (except stop words), calculate the sum of each sentence fragment.
IV) Rank the sentence fragments according to the value of the sum.
V) Eliminate redundant sentence fragments.

To eliminate redundancy, we compare pairs (X, Y) of sentence fragments using the following steps:

a) Obtain the word vector without empty words for each sentence fragment.
b) Find the number of similar words (*SW)* between the two sentence fragments.
c) If $\frac{SW}{|X|} \geq \frac{2}{3}$ or $\frac{SW}{|Y|} \geq \frac{2}{3}$, remove the sentence fragment with lower sum of information gains of the vector.

For example, if we have the following sentence fragments for the target **Carlos the Jackal,** with their corresponding sums:

```
2.290 nickname for Venezuelan born Ilich Ramirez Sanchez
```



	Pilot 2002	TREC 2003	TREC 2004
JIRS	0,733	0,424	0,284
LUCENE	0,715	0,54	0,39
% of Improvement	-2,456	27,358	37,324

Fig. 1. Comparison of the F-measures obtained with two different retrieval systems JIRS and Lucene.

```
2.221 Ilich Ramirez Sanchez , the Venezuelan born former
guerrilla

2.157 Ilich Ramirez Sanchez , the terrorist

1.930 Ilich Ramirez Sanchez , the man

1.528 Illich Ramirez Sanchez
```

If we compare the first and the second sentences, the result of the step a) is:

```
[nickname, Venezuelan, born, Ilich, Ramirez, Sanchez]
[Ilich, Ramirez, Sanchez, Venezuelan, born, former, guerrilla]
```

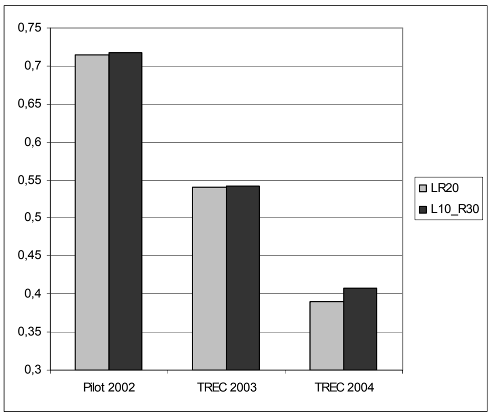
In the step b) we obtained that the number of similar words is *SW*=5.

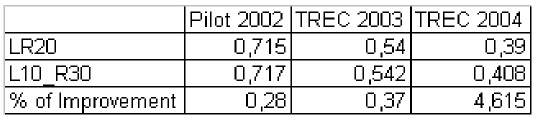
Finally, in the step c) we remove the second sentence fragment since it has a lower sum of information gains. Applying the procedure with the remaining sentence fragments, the result is that we keep only the first:

```
2.290 nickname for Venezuelan born Ilich Ramirez Sanchez
```

4 Experiment Results

We performed experiments with three sets of definition question, the questions from the *pilot* evaluation of definition question 2002 [9], *definition* questions from TREC 2003 [10], and *other* questions from TREC 2004 [11]. (We did not compare our results with the collections of the TREC 2005 and 2006 since in these years the list of nuggets was not readily available). To obtain passages, first we test the output of two retrieval systems JIRS and Lucene, since the overall performance of definition question system depends on the resources and tools used for answer finding [7,8]. Figure 1 shows the comparisons of the F-measure, the best results are obtained with the general propose system (Lucene) instead of JIRS, which is intended to retrieve passages for factoid questions. These results led to the decision to keep using Lucene in further experiments (instead of JIRS), since it provides an improved set of paragraphs.





	Pilot 2002	TREC 2003	TREC 2004
LR20	0,715	0,54	0,39
L10_R30	0,717	0,542	0,408
% of Improvement	0,28	0,37	4,615

Fig. 2. Comparison of the F-measures obtained with balanced sets LR20 and non-balanced sets L10_R30.

In the second experiment, we try to identify which set (*left* or *right*) contributes more for the identification (since we found that the *right* set is usually larger than *left* set). So we set the experiment comparing the results of taking the first 20 sentence fragments from the *left* and the first 20 fragments from *right* sets against taking a ratio of 1:3 between *left* and *right* sets, i.e. we take 10 sentence fragments from the left set and 30 from the right set. We obtained the best results with non-balanced sets, as presented in figure 2.

Thus, we built a system using Lucene to obtain the passages. From these passages we retrieve relevant sentences. Then, we applied a parser (Link Grammar [4]) to analyze the relevant sentences in order to get clauses. Next, from the clauses we obtained the four kinds of sentence fragments detailed above, in section 2. Finally, the sentence fragments were separated in two kinds of fragments, the fragment to the *left* and *right* of the target term. The approach of information gain is then applied to these sentence fragments to obtain the most relevant fragments. Also, we used non-balanced sets of sentence fragments, as the results of the second experiment suggested. Figure 3 displays the F-measure obtained with our system (DefQuestion_IG) compared against the systems of the pilot version of definition questions proposed in 2002. Figure 4 shows the comparison of the F-measures of our system with the systems that participated in the TREC 2003. Finally, figure 5 presents the comparison of the F-measures of the systems in the TREC 2004 and our system. From the figures 3 to 4, we can observe that our system DefQuestions_IG showed very competitive results.

1	**DefQuestion_IG**	**0,717**
2	F	0,688
3	A	0,606
4	D	0,568
5	G	0,562
6	E	0,555
7	B	0,467
8	C	0,349
9	H	0,33

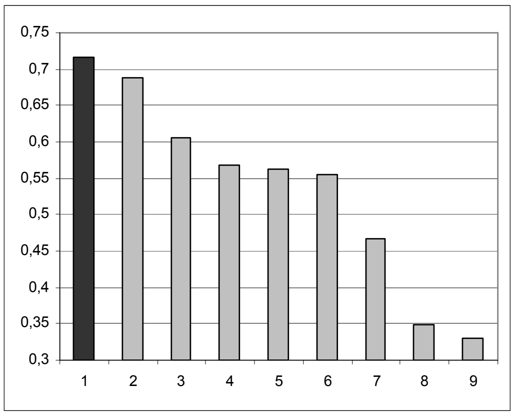


Fig. 3. Comparison of F-measure values of pilot evaluation of definition questions using the AUTHOR list of nuggets.

1	BBN	0,555
2	**DefQuestion_IG**	**0,543**
3	National University of Singapore	0,473
4	University of Southern Calif. ISI	0,461
5	Language Computer Corp.	0,442
6	Univ. Of Colorado/Columbia Univ.	0,338
7	ITC-irst	0,318
8	Univ. Of Amsterdam	0,315
9	MIT	0,309
10	Univ. Of Sheffield	0,236
11	Univ. Of Iowa	0,231
12	Carnegie Mellon University	0,216
13	Fundan University	0,192
14	Univ. Of Pisa	0,185
15	IBM Research	0,177
16	NTT Communication Science Labs	0,169

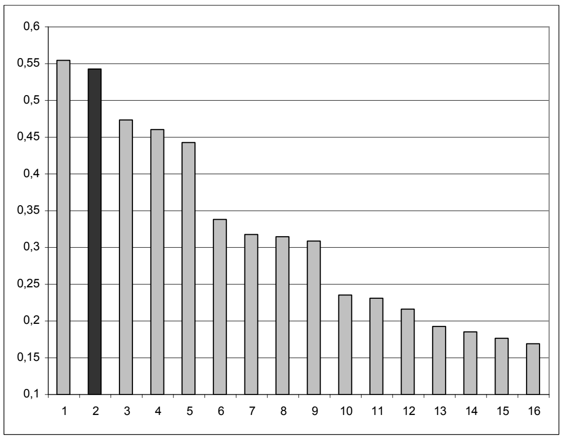
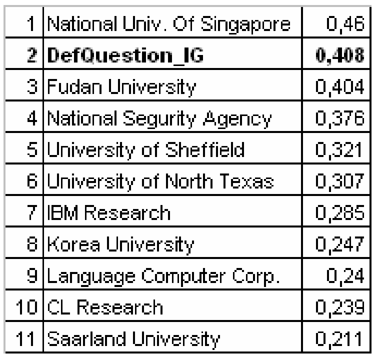


Fig. 4. Comparison of F-measure values of TREC 2003.



1	National Univ. Of Singapore	0,46
2	**DefQuestion_IG**	**0,408**
3	Fudan University	0,404
4	National Segurity Agency	0,376
5	University of Sheffield	0,321
6	University of North Texas	0,307
7	IBM Research	0,285
8	Korea University	0,247
9	Language Computer Corp.	0,24
10	CL Research	0,239
11	Saarland University	0,211

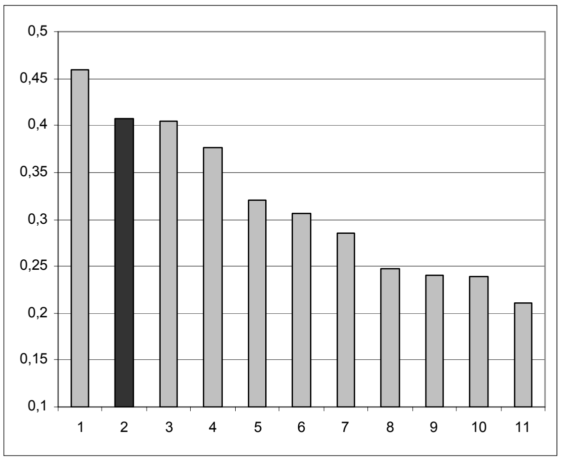


Fig. 5. Comparison of F-measure values of TREC 2004.

5 Conclusions and Future Work

We have presented a method to extract nuggets in an automatic and flexible way and the results obtained are quite competitive when compared to the participating systems in the TREC whose sets of nuggets were used to evaluate the output of our system. The sentence fragments obtained with the process presented are acceptable since these contain only the information directly related to the target. Other advantage is that these sentence fragments present a short length, and this improves the precision of our definition question system.

Future work includes combining Machine Learning algorithms with Information Gain to identify definition sentence fragments since we have showed previously [7] that the combination can improve the performance of the system. Also we plan to categorize the targets in three classes: ORGANIZATIONS, PERSON and ENTITIES and then train three different classifiers. We expect that in this way we can exploit the peculiarities of each kind of entity.

Acknowledgements. We would like to thank CONACyT for supporting this work under scholarship grant 157233 for the first author, and the second author was partially supported by SNI, CONACyT México.

References

1. Carmel, D., Farchi, E., Petruschka, Y., and Soffer, A.: Automatic Query Refinement using Lexical Affinities with Maximal Information Gain. *SIGIR* (2002): 283-290.
2. Cui, H., Kan, M. Chua, T. and Xiao, J.: A Comparative Study on Sentence Retrieval for Definitional Questions Answering. *SIGIR Workshop on Information Retrieval for Question Answering (IR4QA)*, (2004) 90-99.
3. Denicia-Carral, C., Montes-y-Gómez, M., and Villaseñor-Pineda, L.: A Text Mining Approach for Definition Question Answering. *5th International Conference on Natural Language Processing, Fin Tal. Lecture Notes in Artificial Intelligence, Springer* (2006).
4. Grinberg, D., Lafferty, J., and Sleator, D.: A Robust Parsing Algorithm for Link Grammars. Carnegie Mellon University Computer Science technical report CMU-CS-95-125, and *Proceedings of the Fourth International Workshop on Parsing Technologies*, Prague, September, (1995).
5. Harabagiu, S., and Lacatusu, F., Strategies for Advanced Question Answering. *In Proceeding of the Workshop on Pragmatics of Questions Answering at HLT-NAACL*. (2004): 1-9.
6. Hildebranddt, W., Katz, B. and Lin, J.: Answering Definition Question Using Multiple Knowledge Sources. *In Proceeding of HLT/NAACL*, Boston (2004): 49-56.
7. Martínez-Gil, C., and López-López, A.: Answering Definition Questions using a Statistical Method Combined with a Classifier Ensemble, in *Proceedings of the Seventh International Symposium on Natural Language Processing, SNLP2007*, Thailand, December, (2007): 189-194.
8. Moldovan, D., Pasca, M., Harabagiu,S., and Surdeanu, M.: Performance Issues and Error Analysis in an Open-Domain Question Answering System. *Proceeding of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. Philadelphia, July, (2002): 33-40.
9. Voorhees, E.: Evaluating Answers to Definition Questions. *NIST* (2003) 1-3.
10. Voorhees, E.: Overview of the TREC 2003 Question Answering Track. *NIST* (2003): 54-68.
11. Voorhees, E.: Overview of the TREC 2004 Question Answering Track. *NIST* (2004): 12-20.
12. Xu, J., Licuanan A., and Weischedel R.: TREC 2003 QA at BBN: Answering Definitional Questions. In the *Twelfth Text Retrieval Conference* (2003): 28-35.

Human-Computer Interfaces

Improving Word-Based Predictive Text Entry with Transformation-Based Learning

David J. Brooks and Mark G. Lee

School of Computer Science
University of Birmingham
Birmingham, B15 2TT, UK
d.j.brooks, m.g.lee@cs.bham.ac.uk

Abstract. Predictive text interfaces allow for text entry on mobile phones, using only a 12-key numeric keypad. While current predictive text systems require only around 1 keystroke per character entered, there is ambiguity in the proposed words that must be resolved by the user. In word-based approaches to predictive text, a user enters a sequence of keystrokes and is presented with an ordered list of word proposals. The aim is to minimise the number of times a user has to cycle through incorrect proposals to reach their intended word.
This paper considers how contextual information can be incorporated into this prediction process, while remaining viable for current mobile phone technology. Our hypothesis is that Transformation-Based Learning is a natural choice for inducing predictive text systems. We show that such a system can: a) outperform current baselines; and b) correct common prediction errors such as "of" vs. "me" and "go" vs. "in".

1 Introduction

Since the widespread adoption of the Short Message Service (SMS) protocol, "text-messaging" has become a primary communication medium, and the need for efficient text-entry interfaces on mobile phone keypads has become paramount. The T9 – or "text on 9 keys" – interface allows users to enter alphabetic characters using only 9 keys. Each of these 9 keys is mapped to a set of alphabetic characters according to international standard ITU-T Recommendation E.161 [1, p.1], shown in Figure 1. Alternative mappings have been considered [2], but are unlikely to be adopted because some dialing systems require adherence to the ITU standard. Therefore, we consider only the ITU keypad mappings here.

The T9 mapping associates a single key with multiple letters, introducing ambiguity in key-entry, which must be resolved by the input interface. Early systems employed *multi-tap* interfaces, where each letter is unambiguously identified by a number of successive "taps" on a numeric key. More recently, multi-tap interfaces have been superceded by *single-tap* – or "predictive text" – interfaces[1], which have been shown to reduce the number of keystrokes required per character [3].

[1] In many single-tap interfaces, multi-tap entry is still used to enter novel words.

Received 31/10/07
Accepted 07/12/07
Final Version 17/01/08

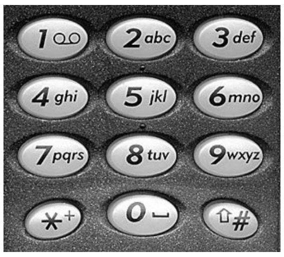


Fig. 1. A mobile phone keypad following the ITU-T E.161 mapping of numeric keys to alphabetic symbols. Non-standard mappings exist for key 1 (symbols such as punctuation), key 0 (spaces), and the * and # keys – one of which is usually assigned to "cycle"

In a single-tap interface, the ambiguity between characters is ignored upon entry: to enter a character requires only a single tap of the associated key. However, because there are multiple letters associated with the key-tap, the system must consider the possibility of extending the current word with each of the associated letters. Some of these extensions will be valid (part-)words, so are retained; others are invalid, so are discarded. Single-tap entry systems are surprisingly effective because, after the first few key-taps of a word, there are usually relatively few words matching that sequence of taps.

Despite improved performance, single-tap systems are still subject to ambiguity at the word level. For instance, the sequence of key-presses 4 – 6 – 6 – 3 corresponds to the words "good", "gone", "home", "hone", "hood" and "hoof", as well as representing the first four letters of words such as "immediate" and "honey". Single-tap entry systems are referred to as "predictive" because they must predict the user's intended word, given the sequence of keystrokes. The present work is concerned with how contextual information can be incorporated into this prediction process. Our hypothesis is that Transformation-Based Learning [4] is a natural choice for inducing effective predictive text systems, and has the advantage that the resulting systems can be implemented on current mobile phone technology.

The remainder of this paper is organised as follows. In Section 2 we give a formal description of word-based predictive text systems, and discuss some existing systems that achieve reasonable predictive behaviour. In Section 3 we discuss part-of-speech (POS) tagging. We argue that there are analogies between this task and word-based text prediction, and, with this in mind, we examine the Transformation-Based Learning (TBL) approach to POS tagging. In Section 4 we describe a novel instantiation of TBL for learning a word-based predictive text system. We describe a series of experiments in Section 5, including evaluations of our system against previous work. Finally, we discuss some limitations of the work in Section 6, and examine some possibilities for future research.

2 Word-Based Predictive Text Systems

In this paper, we consider word-based predictive text (henceforth WBPT) systems, where sequences of key-presses are mapped to sets of matching words. A WBPT system may be formally defined as follows.

Messages in a WBPT system are sequences of words, where words are delimited by space and punctuation characters. The system has a dictionary D containing (a subset of) valid words from the text-entry language[2]. A user enters text by selecting words from the dictionary. Words are selected using a sequence of key-presses, where a key-press is represented by the corresponding numeral in the ITU-T E.161 mapping. A sequence of key-presses is represented as an ordered list of numerals, termed a *signature*. For example, the sequence of key-presses 4 6 6 3 is represented by the signature $\langle 4663 \rangle$.

Each signature maps on to a (possibly empty) set of matching words – termed a *match-set* – which must be retrieved from the dictionary. This set is then ordered into a list, which we call the *match-list*. Match-list ordering is the principal difference between WBPT algorithms, and the most important factor influencing predictive performance. An optimum WBPT system should order the match-list from best to worst, according to context. Match-list ordering is central to our approach, and is used for both supervision and evaluation.

There are, of course, other ways to characterise predictive text systems. LetterWise [5] models transitional probabilities between letters, and so is amenable to novel compounding. However, this approach does not guarantee formation of valid words, and, more importantly, is incapable of resolving word-level prediction errors – such as deciding whether "good" or "home" better reflects a user's intended word. We believe that the approach described below resolves these issues, and may be readily extended to improve compounding systems.

2.1 Determining Match-List Order

The most naïve algorithms for ordering a match-list simply assign arbitrary orderings. This is clearly deficient, since, for a match-list of length N, an arbitrary ordering will start with the best candidate only $\frac{1}{N}$ of the time, while a worse match will be selected first $\frac{N-1}{N}$ of the time – meaning that users are highly likely to experience frustration in response to an arbitrary proposal!

Most existing WBPT systems employ a surprisingly effective algorithm: order the match-list according to descending word-frequency. We will call this the *most-frequent first* (MFF) algorithm. For exposition, we consider word frequencies taken from the 100-million-word British National Corpus (BNC)[3]. Table 1 shows the match-list for the signature $\langle 4663 \rangle$, ordered top-down.

MFF makes sensible predictions most of the time, but inevitably there are exceptions. A user will, at times, wish to words other than the most frequent, but

[2] Novel words may be added to the dictionary, but are ignored here because the associated cost is independent of the WBPT system.

[3] BNC statistics were obtained through the View/BNC interface [6].

Table 1. The match-set for $\langle 4663 \rangle$, with occurrence frequencies from the BNC.

Frequency	Word
80204	good
50539	home
18474	gone
960	hood
125	hoof
45	hone

since rarer words will be chosen less often (by definition) these exceptions are often tolerable. However, there are some predictions that are more problematic.

Consider the signature $\langle 63 \rangle$ and corresponding match-set $\{me,\ of,\ \ldots\}$. Both "of" and "me" are highly frequent words, occurring 2887937 and 2498656 times in the BNC respectively. MFF would use the match-list $(of, me, ...)$, and make at least 2498656 errors when classifying "me". These errors become more conspicuous in certain contexts. If the preceding word is "to" the situation changes dramatically: "to me" occurs 14635 times in the BNC, while "to of" occurs only 76 times[4]. MFF ignores this contextual information, predicting "of" every time.

2.2 Improving Baseline Performance by Considering Context

From the above example, we can see that MFF could be improved by incorporating contextual information – such as whether or not the previous word is "to". The HMS system [7] adopts this approach, supplementing a dictionary with a word bigram model. Each prediction is conditioned on the previous word, and HMS backs-off to MFF where reliable bigram statistics are unavailable. While this approach offers clear performance gains, bigram models have a large memory requirement – $O(n^2)$ where n is the size of the dictionary.

Other recent approaches have employed "common-sense" reasoning [8], using ontological databases to create pragmatic links in order to aid prediction. However, we believe that most prediction errors can be resolved with local syntactic information, and that common-sense approaches deal with more peripheral cases. In addition, common-sense approaches require significant amounts of both memory and computation.

The operating environment for a predictive text system is typically a mobile device such as a cellular phone handset. While such devices have become increasingly powerful, the processing constraints are still relatively modest. Predictive text systems need to operate in real-time – at least as fast as rapid "texters" – and although systems like MFF are viable within these constraints, more expensive n-gram and reasoning techniques are not. Although these approaches may become viable in the long-term, we believe that practical improvements to WBPT may still be achieved in the short-term.

[4] Most occurrences of "to of" may be ascribed to mistranscription of "have" – as in "You ought to of seen his face" (found in document ACB of the BNC).

3 Supervised Part-of-Speech Tagging and Predictive Text

Part-of-speech (POS) tagging is the process of identifying the syntactic role of a word from its surface form and surrounding context. Tagging is non-trivial, as words may be ambiguous, often covering distinct "senses", possibly belonging to different parts-of-speech. For instance, "hope" can be either a noun or a verb.

In many ways, the scenario of POS tagging is similar to the word-based predictive text problem. First, simple taggers attempt to find the best tag for a given word, while we are attempting to find the correct word for a given signature. Second, training a supervised POS tagger requires a corpus where words are annotated with their correct tags. Third, an effective baseline system for supervised tagging is simply to assign the tag that accounts for most tokens of a given word type, which is equivalent to our MFF predictive text system. Finally, supervised tagging systems incorporate contextual information to improve upon the baseline system. Markov Model taggers (such as those described in [9]) condition the choice of the current tag on both the word and the n previous tags – much as n-gram WBPT systems condition predictions based on previous words.

The above similarities suggest that POS tagging systems are analogous to WBPT systems, and effective POS tagging algorithms may yield improved WBPT systems. With this in mind, we will now discuss the Brill tagger – a rule-based POS tagger for incorporating contextual cues into the tagging process – which we believe can be readily adapted for WBPT.

3.1 Transformation-Based Learning

n-gram techniques haven proved successful for both tagging and WBPT, being capable of modelling local contextual information. However, n-grams devote considerable space to modelling rare events. Alternative learning strategies exist that incorporate local context while modelling primarily common events, and this idea is central to Transformation-Based Learning (TBL) [4]. TBL is an error-driven learning technique, which aims to improve a upon any given annotation system by identifying common errors and proposing rules to correct these errors where they occur. We will discuss the nature of a Transformation-Based learner in terms of POS tagging, with a view to treating WBPT within the same framework.

The typical operating environment for a TBL system is shown in Figure 2. A TBL system takes as input an initial guess at the correct POS for a corpus. Often, this is the output of a baseline (like MFF), but may be the output of any tagging system. The TBL learner then compares this initial guess to the "truth" – a gold-standard annotated version of the text, which in the case of tagging is manually-tagged corpus data. From this, the learner creates a list of errors, ordered by some objective criterion (usually just the frequency of the error, but any objective criterion will suffice).

The list of errors is used to propose correction rules, or *transformations*. Each transformation may be decomposed into two parts:

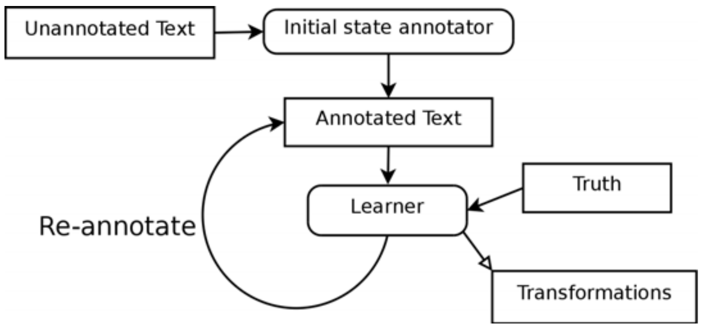


Fig. 2. The training environment for a Transformation-Based Learning system

A rewrite rule A rule that "transforms" an incorrect tag into the correct tag.

A triggering environment Some set of observed contextual features that indicate when a particular transformation should be used.

As an example, we might consider feasible transformations for "hope", which occurs in the BNC as both a verb (VVB - 7827 times) and as a noun (NN1 - 4106 times)[5]. Accordingly, the *most-frequent tag* system will always designate "hope" to be a verb. Clearly, from the frequencies, there are at least 4106 cases where this classification is incorrect, which we might correct using transformation. One conceivable transformation of "hope" might be:

Rewrite rule: change tag from VVB to NN1
Trigger env.: previous word is "the".

In the BNC, this transformation would correct 1490 mistagged instances of "hope".

TBL is a greedy algorithm. At each iteration, the best transformation – according to the objective criterion – is selected and applied to the corpus. This re-annotated corpus is then used as input to the next iteration. Learning continues until the level of improvement becomes negligible.

TBL has been shown to be a very effective algorithm for POS tagging, and this success is due to a number of factors. First, it incorporates contextual information only where it is useful for tagging decisions. Second, it is easy to train, although it does require a manually-tagged corpus as input. Finally, the resulting transformation rules are compact and more easily understandable than stochastic models. We believe that these features make Transformation-Based Learning ideal for WBPT, and will now consider how the learner must be altered for this new domain.

[5] The BNC contains 17192 occurrences of "hope", and VVB and NN1 are the two most frequent tags. Verb-infinitives, proper-nouns and compound tags (e.g. VVB-NN1) account for the remainder, but do not affect our example.

4 A TBL Approach to Predictive Text

In this section, we describe a novel instantiation of Transformation-Based Learning for WBPT. First, we describe our procedures for deriving a training corpus, discussing the form of data and nature of supervision. Second, we specify the form of transformations we use for WBPT. Third, we discuss a definition of error that is based on the human experience of single-tap interfaces. Finally, we describe a novel objective criterion for measuring prediction errors, incorporating this that is similar to the human experience of correction, and finally, considering how transformations should be specified in this domain.

4.1 Data and Supervision

As we observed earlier, TBL is primarily a supervised learning framework, and as such requires a training set that is annotated with correct prediction decisions. For WBPT, we require a data-set that shows the correct word for a given signature in the context of a message. Although such data-sets are not widely available, we can derive them automatically from any raw text corpus. Each word maps to a single signature, which can be trivially retrieved to create our "tagged" corpus. Consider the following sentence:

I want to go home.

For each word, we derive the corresponding signature and label the text:

4/I 9268/want 86/to 46/go 4663/home

From this we can also derive a signature-only version of this sentence:

4 9268 86 46 4663

This method can be used to generate a signature corpus, which can act as input to any WBPT system. The output of such a system will have the same sequences of signatures, but may assign different words according to the prediction model. For instance, MFF would produce the following prediction:

4/I 9268/want 86/to 46/in 4663/good

Thus, we have a gold-standard tagged text, and an input corpus comprising only of signatures. We can use the gold-standard as a training resource for supervised learning, and as a sample from which we can estimate signature-word frequencies for our baseline algorithm.

4.2 Transformations for Predictive Text

The form of transformations for predictive text is similar to that for POS tagging, with some important differences. Recall that transformations comprise a rewrite rule and a triggering environment. For WBPT, our rewrite rules alter the order of the match-list by promoting the correct word to the head of the list.

The triggering environments specify which contextual features are important for correcting erroneous predictions. There are two important differences between POS tagging and predictive text that affect triggering environments. First, a tagger may examine the components (e.g. morphemes) of the word being tagged, whereas in predictive text the mapping between signature and word precludes this. Second, text messages are entered in linear order, whereas POS taggers are conventionally applied to entire sentences. While the Brill tagger considers subsequent POS tags as context, this is not possible in a linear entry system. In our WBPT system, we limit triggering environments to the previous two words in the sentence.

4.3 Measuring Prediction Error

Once we have our gold-standard corpus, we need to be able to measure the prediction error of a given WBPT system. The human experience of correcting predictive text systems provides a natural way to do this. Upon entering a signature, a user is presented with a list of matching words. The user has to cycle through this list until they find their intended word, by pressing a designated *cycle key*[6]. Thus, where the correct word heads the list, no cycles are required; where the correct word is 4th in the list, 3 cycles are required. This quantity – which we will call the *cycle count* – naturally represents the error of any given word prediction.

We can trivially calculate cycle-count from a word and its match-list: the cycle-count is simply the index of that word within the match-list, where indices start from zero. The prediction error for a WBPT system over an entire corpus, E, is obtained by summing cycle-counts over all predictions.

4.4 Selecting Transformations

Transformation-Based Learning systems must be able to identify candidate transformations, from which the "best" selection is selected and applied at each learning iteration. Quality is traditionally defined in terms of prediction error, but in our scenario such a definition can lead to problems.

Recall the example signature ⟨63⟩, where MFF always predicts the word "of" but never predicts "me". Since "me" is a common word, this is a conspicuous errors and a good candidate for transformations. Now consider classifying ⟨63⟩ if we know that the previous word is "made" – the 15th most-frequent word preceding "me" in the BNC (1100 occurrences). Based purely on cycle-counts, "made" seems a strong contextual cue, and we might expect to learn the transformation:

Rewrite rule: change word from "of" to "me"
Trigger env.: previous word is "made".

[6] The cycle key is not part of the ITU-T E.161 specification, but is usually mapped to either the * or # key.

which would correct 1100 misclassifications. However, “made” occurs *more frequently* before “of” (2783 occurrences in the BNC) – all of which would be erroneously transformed by the above transformation. Thus, the above transformation would do our system considerably more harm than good.

To alleviate this problem, we need a criterion that reflects the strength of relation between a rule and its triggering environment. Therefore, we define transformation quality as:

$$TQ = E_M^w \times |w| \times P(w|\sigma, T)^2 \tag{1}$$

where E_M^w is the cycle-count for intended-word w and match-list M, σ is the signature of w and T is the triggering environment. The first term, E_M^w, is just the cycle-count as before, and when multiplied by the second term $|w|$, is the cycle-count across all occurrences of a word-type w across the corpus. The third term $P(w|\sigma, T)$ represents the proportion of the time that w is the correct word, given a signature and triggering environment[7].

We can examine this equation intuitively. The first two terms capture cycle-counts as before, making the system more likely to correct frequently misclassifications (e.g. “of” to “me”), than rare misclassifications (e.g. “good” to “hone”).

The last term represents the *fidelity* of relationship between trigger and word. If a trigger is associated with many members of the match-list it will not help us make a correction. Thus, “to of” warrants a transformation because “to” occurs rarely with “of” but often with “me”; however “made of” does not warrant a transformation because “made” occurs frequently with both “of” and “me”, and therefore affords little information with which to make a decision.

5 Experimental Setup

In the previous section, we described a novel TBL system for predictive text, which can be trained and evaluated on corpus data. In this section we will describe a series of experiments where we train and test our learning system and compare it to baseline and n-gram systems.

We trained our learning system on data taken from the BNC. Specifically, we created a single file containing all sentences of the BNC, and randomised the order between sentences (but not between words). From this corpus, we created 33 disjoint partitions, each comprising 180000 sentences (around 3.3 million words). We split each partition into a training set of 162000 sentences and a test set of 18000 held-out sentences – achieving a rough 90:10 split of training- to test-data in each partition.

We trained three systems on each of these corpora: the MFF baseline system; a bigram system with back-off; and our Transformation-Based learner, which

[7] We square $P(w|\sigma, T)$ only to increase our preference for strong relationships between triggering environments and predictions: large probabilities indicate strong relationships, small probabilities indicate weak relationships, and squaring a small probability will make it even smaller.

uses MFF as its initial input. Table 2 shows the mean and standard deviation in prediction accuracy for each system. We calculate significance levels (p-values) of differences in error-rates using a two-tailed paired t-test. The results show that

Table 2. Average accuracy (including standard-deviation) of three predictive text systems: MFF; a Bigram model with back-off; and Transformation Based Learning

System	Accuracy	Stdev
MFF	89.9189%	(0.0737)
Bigram with backoff	90.5798%	(0.0739)
TBL	90.6149%	(0.0736)

MFF is a relatively good baseline system, performing at close to 89.92% accuracy. Both the bigram model and our TBL system achieve statistically significant increases in accuracy (of 0.66%, with negligible p-values) over the MFF baseline. This is to be expected as both these models incorporate contextual information, which should enhance prediction accuracy. Interestingly, our TBL system also achieves a small and significant increase in accuracy over the bigram model (0.04%, $p = 0.00027$). Initially this seems counter-intuitive because the bigram model encodes statistics even for rare preceding words. However, the results are understandable in terms of novel words in the held-out sample. To make a prediction for word w_n where the preceding word w_{n-1} was not part of the training data, the bigram model backs off to MFF and is therefore only as good as MFF over those cases. In contrast, our instantiation of TBL considers triggering environments covering either or both of two preceding words, w_{n-2} and w_{n-1}. Thus, even where w_{n-1} is novel, some accurate predictions can still be made if w_{n-2} is part of a triggering environment. Such discontinuous affects are not captured in n-gram model.

The performance of our system is encouraging, so we should also examine our claims about memory requirements. For each experiment, around 1200 transformations were proposed – a negligible increase in comparison to the size of dictionary involved. We should also examine the transformations produced by the system, to see if they are only encoding important contextual information.

We chose one of the 33 experiments at random. During this experiment, our system proposed 1225 transformations, the top 10 of which are shown in Table 3. This list includes some of the very frequent errors identified earlier. The system successfully identifies transformations to correct “of” to “me”, and “in” to “go”, amongst others. Many of these transformations make syntactic sense: for instance, we are unlikely to see “to in”, so the transformation “in” “go”: $\langle * \; to \rangle$ corrects this to “to go”. However, we can also see some interesting exceptions. First, there are some contentious triggers: “there” “these”: $\langle * \; in \rangle$ represents the increased likelihood of “in these”, but “in there” is a feasible construction. Second, there are some plausible constructions that would probably not be used much in text messages, such as “civil war”. This reflects a bias specific to our

Table 3. The top 10 transformations for a randomly chosen experiment, ordered by number of corrections in training corpus. For each transformation, the most frequent triggers are shown. Each trigger is an ordered pair of preceding words, where $*$ represents any word

# Corrections	Rule	Example Triggers
3229	them → then	$\langle *\ \textit{and}\rangle$, $\langle *\ ,\rangle$, $\langle *\ .\rangle$, $\langle *\ \textit{is}\rangle$
1930	of → me	$\langle *\ \textit{to}\rangle$, $\langle *\ \textit{for}\rangle$, $\langle *\ \textit{told}\rangle$, $\langle *\ \textit{let}\rangle$
1613	in → go	$\langle *\ \textit{to}\rangle$, $\langle *\ \textit{n't}\rangle$, $\langle *\ \textit{'ll}\rangle$, $\langle *\ \textit{can}\rangle$
1302	on → no	$\langle \textit{there}\ *\rangle$, $\langle \textit{award ref}\rangle$, $\langle \textit{I have}\rangle$, $\langle *\ \textit{oh}\rangle$
1280	there → these	$\langle *\ \textit{of}\rangle$, $\langle *\ \textit{all}\rangle$, $\langle *\ \textit{to}\rangle$, $\langle *\ \textit{in}\rangle$
1136	up → us	$\langle *\ \textit{the}\rangle$, $\langle *\ \textit{of}\rangle$, $\langle *\ \textit{to}\rangle$, $\langle *\ \textit{for}\rangle$
957	he → if	$\langle *\ \textit{even}\rangle$, $\langle *\ \textit{see}\rangle$, $\langle \textit{and}\ ,\rangle$, $\langle *\ \textit{asked}\rangle$
655	might → night	$\langle *\ \textit{last}\rangle$, $\langle *\ \textit{the}\rangle$, $\langle *\ \textit{a}\rangle$, $\langle *\ \textit{all}\rangle$
560	an → am	$\langle *\ \textit{I}\rangle$
556	was → war	$\langle *\ \textit{the}\rangle$, $\langle *\ \textit{world}\rangle$, $\langle *\ \textit{civil}\rangle$, $\langle *\ \textit{of}\rangle$

training corpus – an issue we will discuss below. Finally, there are examples of ambiguity in the triggering environments: in the context $\langle *\ \textit{the}\rangle$ "no" is an abbreviation of "number", and "us" is an abbreviation for "United States".

We should also examine the errors made by our original classifier, and the proportion of those errors that are corrected by our system. Figure 3 shows the 30 most costly words in our chosen test set. The bars show the original error

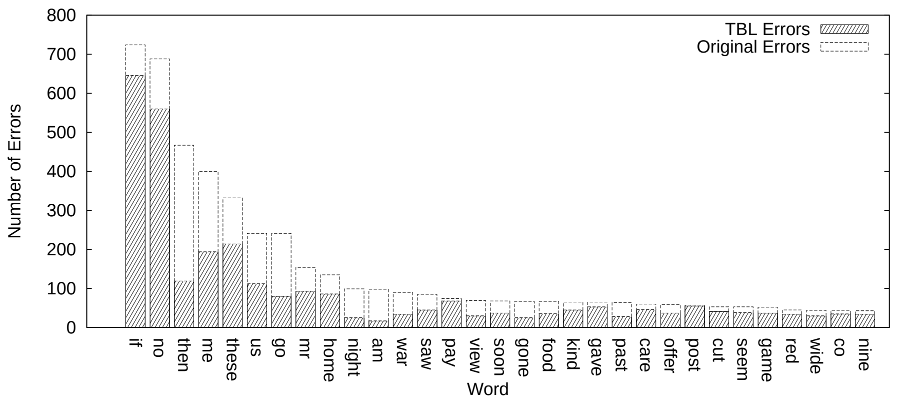


Fig. 3. 30 most-frequent errors made by MFF, with proportion remaining after TBL

count per word (in white), and the TBL error count as a proportion (hatched). For example, the first bar represents the word "if", for which there are 724 errors in MFF, and 646 in TBL system. Although the first two words – "if" and "no" – is largely uncorrected, for many of the subsequent words – including "then", "me", "these", "us", "go" and "am" – our system provides effective correction.

6 Future Work and Conclusions

We have seen that TBL can improve predictive text systems, but there are still areas where our approach could be improved. First, our system has been evaluated only in a Machine-Learning context. User-testing would give a better indication of performance *in situ*, and would permit comparative evaluation against other predictive text systems.

More crucially, our experiments use only BNC data, which does not adequately represent the language used in text messages. However, our methods are language-independent, and can easily be applied to more representative corpora and to other languages. Future work should consider training on SMS or instant-messaging corpora, and adapting our system to users' idiolects.

We have described a novel Transformation-Based learner for improving predictive text systems, in a language-independent manner. We have shown that the resulting WBPT systems achieve statistically significant improvements in prediction accuracy over bigram models, and that these systems are capable of correcting many problematic misclassifications in English. We believe our system offers tangible improvements for predictive text interfaces, and that it works within the constraints of current technology.

References

1. International Telecommunication Union: E.161: Arrangement of digits, letters and symbols on telephones and other devices that can be used for gaining access to a telephone network. http://www.itu.int/rec/T-REC-E.161-200102-I/en (2001)
2. Gong, J., Tarasewich, P.: Alphabetically constrained keypad designs for text entry on mobile devices. In: CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press (2005) 211–220
3. MacKenzie, I.S.: KSPC (keystrokes per character) as a characteristic of text entry techniques. In: Mobile HCI '02: Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction, London, Springer-Verlag (2002) 195–210
4. Brill, E.: Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational Linguistics **21** (1995) 543–565
5. MacKenzie, I.S., Kober, H., Smith, D., Jones, T., Skepner, E.: Letterwise: Prefix-based disambiguation for mobile text input. Technical report, Eatoni (2001)
6. Davies, M.: The View/BNC interface. http://corpus.byu.edu/bnc/ (2004) Accessed on 23rd October 2007.
7. Hasselgren, J., Montnemery, E., Nugues, P., Svensson, M.: HMS: A predictive text entry method using bigrams. In: Proceedings of the Workshop on Language Modeling for Text Entry Methods, 10th Conference of the European Chapter of the Association of Computational Linguistics, Budapest, Hungary, Association for Computational Linguistics (2003) 43–49
8. Hawes, N., Kelleher, J.: Context-sensitive word selection for single-tap text entry. In: STAIRS 2004: Proceedings of the Second Starting AI Researchers' Symposium, IOS Press (2004) 217–222
9. Charniak, E., Hendrickson, C., Jacobson, N., Perkowitz, M.: Equations for part-of-speech tagging. In: AAAI. (1993) 784–789

Author Index

Índice de autores

Editorial Board of the Volume

Comité editorial del volumen

Additional Reviewers

Árbitros adicionales

Jisha Abubaker
Iñaki Alegria
Miguel A. Alonso
Muath Alzghool
Xabier Arregi
Fco. Mario Barcala
Sara Carrera
Hakan Ceylan
Victor Darriba
Alexander Dikovsky
Denys Duchier
Mohamed Abdel Fattah
Marcello Federico
Milagros Fernandez-Gavilanes
Sergio Ferrández
Óscar Ferrández
Davide Fossati
Mary Ellen Foster
Oana Frunza
Alfio Massimiliano Gliozzo
José M. Gómez
Jorge Graña
Marco Guerini
Carlos Gómez-Rodríguez
Eva Hajicova
David Hope
Aminul Islam
Steliana Ivanova
Ruben Izquierdo
Heng Ji
Sylvain Kahane
Cynthia Kersey
Rob Koeling
Anna Korhonen
Zornitsa Kozareva
Svitlana Kurella
Kow Kuroda
Fotis Lazaridis
Mihai Lintean
Erica Lloves-Calviño
Miguel Angel Molinero Alvarez
Andrea Mulloni
Fernando Magán-Muñoz
Sergio Navarro
Roberto Navigli
María Pardiño
Jurgen Paulus
Emanuele Pianta
Juan Otero Pombo
Judita Preiss
Marcel Puchol-Blasco
Jonathon Read
Francisco Ribadas-Pena
Rachel Roxas
Kepa Sarasola
Martin Scaiano
Luciano Serafini
Chung-chieh Shan
Ravi Sinha
Georg Sommer
Aitor Soroa
Mark Stevenson
Rajen Subba
Swati Tata
Stefan Thater
Masatsugu Tonoike
Alberto Tretti
Jesus Vilares
David Weir
Taras Zagibalov
Beñat Zapirain

Impreso en los Talleres Gráficos
de la Dirección de Publicaciones
del Instituto Politécnico Nacional
Tresguerras 27, Centro Histórico, México, D.F.
Febrero de 2008.
Printing 500 / Edición 500 ejemplares.