INTERNATIONAL WORKSHOP

NATURAL LANGUAGE PROCESSING FOR EDUCATIONAL RESOURCES

held in conjunction with the International Conference

RANLP - 2007, September 27-29, 2007, Borovets, Bulgaria

PROCEEDINGS

Edited by

Nerea Ezeiza, Montse Maritxalar and Mathias Schulze

Borovets, Bulgaria

26th September 2007
International Workshop

NATURAL LANGUAGE PROCESSING
FOR EDUCATIONAL RESOURCES

PROCEEDINGS

Borovets, Bulgaria
26th September 2007

ISBN 978-954-452-003-8

Designed and Printed by INCOMA Ltd.
Shoumen, Bulgaria
WORKSHOP ORGANISERS

Nerea Ezeiza
(University of the Basque Country, IXA research group)

Montse Maritxalar
(University of the Basque Country, IXA research group)

Mathias Schulze
(University of Waterloo, Canada)

WORKSHOP PROGRAMME COMMITTEE

Arantza Diaz de Ilarraza (University of the Basque Country)
    Nerea Ezeiza (University of the Basque Country)
    Pius ten Hacken (Swansea University, U.K.)

Trude Heift (Linguistics Department, Simon Fraser University, Canada)

Ola Knutsson (School of Computer Science and Communication, Royal Institute of Technology, Sweden)

Chao-Lin Liu (Department of Computer Science, National Chengchi University, Taiwan)

Montse Maritxalar (University of the Basque Country)

Detmar Meurers (Ohio State University, USA)

Ruslan Mitkov - (University of Wolverhampton, U.K)

John Nerbonne (University of Groningen, Netherlands)

Leonel Ruiz Miyares (Centre for Applied Linguistics, Santiago de Cuba)

Mathias Schulze (University of Waterloo, Canada)

Eiichiro Sumita (NICT/ATR, Japan)
EDITORS’ FOREWORD

In this workshop we discuss how using NLP techniques in the automatic treatment of texts can support pedagogic goals. There are well over 100 documented projects which make or made use of NLP in Intelligent Computer-Assisted Language Learning; however, there are significantly fewer publications on the use of NLP techniques in educational areas other than language learning. This workshop brings together NLP and ICALL researchers with colleagues who are investigating NLP treatment for other educational purposes. For example, Ming-Shin Lu et al. (2007) built a software environment for helping experts translate test items for the Trends in International Mathematics and Science Study from American English to Chinese, and Genoves et al. (2007) have developed a tool to analyze abstracts of scientific papers. Klenner et al. (2007) present a system for self-correction tests which asks learners to identify the correct error message corresponding to sentences made by them.

We also want to discuss how we can use NLP in the automatic treatment of corpora for pedagogical purposes as well as in the analysis of learner corpora, i.e. students’ free responses. Most of the contributors to these proceedings employ general purpose text corpora.

Finally, it is important to stress that the automatic generation of exercises is a recent research avenue which has a remarkable presence in this workshop. In this sense, different prototypes and proposals for assisting the tasks of automatic exercise generation are presented in Aldabe et al. (2007), Ming-Shin Lu et al. (2007) and Nagata et al. (2007).

We would like to thank the members of the international Programme Committee for their constructive comments and speedy response. Our gratitude also goes to Itziar Aldabe for her technical support with creating and maintaining the web page of the workshop.

We are especially grateful to Dr. Kiril Simov and Professor Ruslan Mitkov, organizers of the conference Recent Advances in Natural Language Processing (RANLP 2007), for their continued support and encouragement during the organization of the workshop.

September 2007

Nerea Ezeiza, Montse Maritxalar and Mathias Schulze
# TABLE OF CONTENTS

Ming-Shin LU, Jen-Hsiang LIN, Yu-Chun WANG, Zhao-Ming GAO, Chao-Lin LIU and Chun-Yen CHANG  
*Prototypes of Using NLP Techniques for Assisting Translation and Authoring of Test Items*  
......... 1

Itziar ALDABE, Montse MARITXALAR and Edurne MARTINEZ  
*Evaluating and Improving the Distractor-Generating Heuristics*  
.................................................... 7

Manfred KLENNER, Simon CLEMATIDE and Bojan PERIC  
*What (the Hell) Is Wrong? An Approach to Semi-Automatic Construction of Self Correction Tests*  
................................................................................................................... 15

Luiz GENOVES Jr., Valéria D. FELTRIM, Carmen DAYRELL and Sandra ALUÍSIO  
*Automatically Detecting Schematic Structure Components of English Abstracts: Building a High Accuracy Classifier for the Task*  
................................................................................................................... 23

Ryo NAGATA, Atsuo KAWAI, Koichiro MORIHIRO and Naoki ISU  
*Visualizing the Invisible: a Method for Visualizing the Countability of English Nouns*  
............... 31
Prototypes of Using NLP Techniques for Assisting Translation and Authoring of Test Items

Ming-Shin Lu†  Jen-Hsiang Lin†  Yu-Chun Wang‡  Zhao-Ming Gao‡  Chao-Lin Liu‡  Chun-Yen Chang§
†Nat’l Chengchi University
Dept. of Computer Science
Taipei Taiwan
chaolin@nccu.edu.tw
‡Nat’l Taiwan University
Dept. of Electrical Engineering
Taipei Taiwan
zmgao@ntu.edu.tw
§Nat’l Taiwan Normal University
Dept. of Earth Science
Taipei Taiwan
changcy@ntnu.edu.tw

Abstract

We report two software prototypes that are designed for assisting the tasks of test item translation and test item authoring. We built a software environment for helping experts translate the test items for the Trends in International Mathematics and Science Study (TIMSS). Test items of TIMSS are prepared in American English and will be translated to traditional Chinese. We also built a software environment for composing test items for introductory Chinese courses. The system currently aids the preparation of four important categories of test items, and the resulting test items can be administrated on the Internet.

Keywords
language processing, computer assisted education, controlled languages, test item translation, test item writing, machine translation, TIMSS

1. Introduction

As the presentations in the main conference and this workshop have demonstrated, techniques of natural language processing (NLP) have a wide range of practical applications. In this paper, we briefly outline two applications of NLP techniques to the preparation of educational resources. The first application is to assist the translation of TIMSS [4] test items from English to Chinese. The second application is to assist teachers to prepare test items for introductory Chinese courses.

TIMSS aims at studying the achievements (in mathematics and science) of the fourth and eighth grade students in multiple countries with a common set of test items. The original test items are prepared in American English, and the participating countries must translate the test items into local languages under the guidelines [16]. While the translation must consider the cultural differences of the participating countries [19], all translators must try to make the translation as close to the original items as possible. The bottom line is that the translation must maintain the challenge levels of test items so that the test results remain reliable for further studies. Hence, the translation must be reviewed carefully by the local TIMSS organisation and an international review committee.

Our prototypes aims at helping translators abide by the translation guidelines as much as possible, while making the translation process more efficient. Figure 1 shows the main user interface of the prototype. After translators choose a test item in the upper left corner, the prototype shows the selected item in the upper right area, and recommends translations for the English words in the middle of the window. Translators can select and modify the recommended words, and change the orders of the selected words to make a complete test item at the bottom of the interface.

We also introduce an environment for preparing test items for students who are learning introductory Chinese. Chinese characters are hard to learn and remember. A typical way to test students’ vocabulary is asking students to correct characters in a sentence. In addition, asking students to find a grammatical ordering of a set of shuffled words is a good way to practice Chinese grammars. We discuss two functions of our system: authoring of test items for character correction and sentence reconstruction.

We present the prototype for translating TIMSS items and its evaluation in Section 2, overview the design of the environment for preparing test items for introductory Chinese courses in Section 3, and make a brief con-
2. Translating TIMSS Items

In the current prototype, it takes three major steps to translate a test item, after we convert the TIMSS files from the Microsoft WORD® format to pure texts with a JACOB service [9]. The translator first chooses a test item in English from an item set. Our system will look up the lexicon and provide a list of candidate Chinese translations for words and phrases in the selected item. The translator will then choose the best candidate translation for each word and phrase, and edit the selected sequence of translation into a Chinese test item. During this post-editing phase, the translator can add more Chinese words that do not directly correspond to any English words or phrases in the original item. The translator may need to change the word orders to make the translation grammatically correct in Chinese, and the translator may also remove and/or modify the words that were chosen from the list of candidate translations.

It is undeniable that our system should attempt to recommend a Chinese translation which considers the change of word orders, and allows the translator to improve the recommended Chinese sequence. In order to offer this function, we need to have a sufficient number of translated TIMSS items to learn the correspondence between syntactic structures of English and Chinese items. However, we have only test times for TIMSS 1999 and TIMSS 2003, which include only hundreds of test items. A possible substitute is that we try to learn the syntactic correspondence between English and Chinese from other text. This is a more feasible approach and is on our agenda.

2.1 Consistency in translation

To maintain the quality of the translated items, it is important to translate specific terms and phrases in a consistent way. These terms and phrases include “as shown below”, “explain why”, “one has been done for you” and many others. Every translator must use the same Chinese patterns for these specific phrases, according to the guidelines for all translators. Translations of units for weights and length as well as localisation of English names are taken care of too. Hence, our system must identify these special phrases for recommending appropriate translations.

In addition, there are occasions when translators will want to find how a term or a pattern of terms were previously translated in the TIMSS item bank. Knowing how the patterns were translated in the past years helps the translators maintain consistency in translation even for the optional cases.

Hence our system will help translators find previous test items that contain specific word patterns. We achieve this by implementing a component that can recognise regular expressions, and apply a concordancer (cf. [15]), which aligns the queried terms, to present the items.

2.2 Ordering candidate translations

Except the special patterns that we just discussed in Section 2.1, our system finds all candidate translations for the English words in the test item from the Concise Oxford English-Chinese Dictionary (OECD) [5]. We employ MINIPAR [14] to locate some special patterns, and MXPOST [22], the Porter algorithm [21], and WordNet [5] to determine the part of speech of words and their root forms. Hence, each of the special patterns and individual words has a list of candidate translations.

Let $E_1$, $E_2$, ..., and $E_n$ represent the units, i.e., individual words or idioms, in an English sentence $S_E$. Let $C_i$ denote the set of possible Chinese translations of $E_i$, and $C_i = \{C_{i,1}, C_{i,2}, ..., C_{i,q(i)}\}$, where each $C_{i,j}$ represents a candidate translation of $E_i$, and $E_i$ has $q(i)$ candidate translations. If $E_i$ is a special term, that we explained in Section 2.1, we use the standardised translations for $E_i$. If not, we set up $C_i$ for $E_i$ with OECD. Let $C_{i,q(i)}$ denote a word that is selected from $C_i$. In the following subsections, we use $S_C$ to represent a sequence of $C_{i,q(i)}$, $i = 1, 2, ..., n$.

Given the lists of candidate translations, the translator can choose the best candidate for each word in a pull-down menu. Hence, placing more promising candidates at the tops of the menus facilitates the translator to find the best candidates easier. We would like to offer better orderings of the candidate words, but we will not try to solve this word sense disambiguation [12] problem in the prototype.

We consider four possible factors for ordering the candidate translations. We may record the frequency of a candidate Chinese translation being chosen for an English term, and prefer the one that has the highest frequency. We may look into relevant publications from which we obtain the relative frequency of a word being used. In addition to collecting word frequencies with monolingual corpora, more advanced NLP techniques [15] can be helpful. By aligning English words and their Chinese translations in parallel corpora, we can estimate the probability, $Pr(C \mid E)$, of an English word, $E$, being translated into a particular Chinese word, $C$. We can also learn n-gram statistics from the corpora with public domain software. To explore these possibilities, we employed GIZA++ [18] to learn $Pr(C \mid E)$ and SRILM [23] to learn the bi-gram statistics with the Chinese-English bilingual version of Scientific American [8].

We discuss these factors next. Following the convention used in [15], we use superscripts and subscripts to indicate indexes of words, respectively, in the vocabulary of English (or Chinese) and in a particular sentence.

2.2.1 Usage frequency

After our system is put to work, we can observe the usage frequency, $f_k(C^i \mid E^j)$, of a Chinese translation, $C^i$, being selected for a given English word, $E^j$. If $E_k$ of
resulting list more reliable, although the resulting size used in OECD. Let 

applied a Chinese synonym-finder service \[that had the same English translation with . We also 

23422 types. We recorded the frequency \(f_k\) for each \(w\). (The subscript of \(f_k\) indicates that the fre-

ber of words. \[15\]) Let \(C^i\) denote the different words in a corpus, and \(Types\) that belong to 23422 types. (For each \(w\), we searched a Chinese-

the "no-show" words. 

2.2.2 Corpus frequency 

We may prefer Chinese words that appear more fre-

quently in magazines written for fourth and eighth grad-

ers, so we employed articles published in two such magazines \[7\] to collect word frequencies. Because there are no spaces in between Chinese words, we employed a popular Chinese tokeniser \[10\] to segment the Chinese text in the selected corpora, and obtained 734000 tokens that belong to 23422 types. \((Types\) are the number of different words in a corpus, and \(tokens\) are the total num-

ber of words. \[15\]) Let \(C^i\) denote the \(i\)-th type of these 23422 types. We recorded the frequency \(f_k(C^i)\) for each \(C^i\). (The subscript of \(f_k\) indicates that the frequencies come from the corpora.) 

Since the main purpose of collecting the word frequencies is to order the Chinese translations used in OECD to translate English words, we need to collect frequencies of these words. However, these Chinese translations may not be used in the magazine articles, but their synonyms may. In such cases, we would like to use the frequencies of their synonyms as the frequencies of the "no-show" words. 

Hence, we created a list of synonyms for each of the 23422 types. For each \(C^i\), we searched a Chinese-

English lexicon, HowNet \[5\], for a list of words, \(S_h(C^i)\), that had the same English translation with \(C^i\). We also applied a Chinese synonym-finder service \[11\] to find another list of synonyms of \(C^i\), \(S_s(C^i)\). We then set the list of synonyms of \(C^i\), \(S(C^i)\), to the intersection of \(S_h(C^i)\) and \(S_s(C^i)\). Taking the intersection made the resulting list more reliable, although the resulting size became smaller than those of the original two lists.

With \(f_k(C^i)\) and \(S(C^i)\) for each \(C^i\), we assigned a corpus frequency to each of the 39429 Chinese types used in OECD. Let \(w\) denote a word used in OECD and \(j,s \in [1,23422]\). If \(w = C^i\), we set \(f_k(w)\) to \(f_k(C^i)\), and, if \(w \in S(C^i)\), we set \(f_k(w)\) to \(f_k(C^i)\). If, in the latter case, \(w\) is included in more than one synonym lists, we would choose one of these \(C^i\) arbitrarily. Such an arbitrary decision inevitably introduced inaccuracy into our decisions, but we would have to employ advanced techniques of word sense disambiguation to solve all potential problems. We set \(f_k(w)\) to zero if none of the cases applied. 

Among 39429 Chinese types used in OECD, 22579 types appeared at least once in our corpora \[7\]. With the help of the synonym lists, we set the corpus frequencies of another 4373 types to non-zero values.

2.2.3 Word alignment and n-gram statistics 

GIZA++ can find the best word alignment for a pair of sentences. Yet, before we can take advantage of this service, we have to prepare the text materials for GIZA++. There are two preparatory tasks to do. We have to align sentence in the parallel corpora, and we have to cluster the words in the corpora. 

For the first task, because the paragraphs in our bilingual \textit{Scientific American} are aligned, we align English and Chinese sentences within the aligned paragraph. We segmented the Chinese sentences into tokens with the Chinese tokeniser \[10\] so that we have a basis for judging whether two sentences should be aligned. We designed a primitive sentence-alignment method for the prototype. We located the first and the last \(\alpha\) words in an English sentence. (Currently, we do not consider sentences that contain less than \(2\alpha\) words.) For each of these \(2\alpha\) words, we looked up OECD for their Chinese transla-

tions. Furthermore, we expanded each of these transla-

tions with words in their synonym lists, which we ex-

plained in Section 2.2.2, if such a corresponding list ex-

ists. The located synonyms were then used as if they were the translations for the English word. A Chinese word that matched any of such translations of an English word was considered to be a translation for the English word. A Chinese sequence that contained the translations of all \(2\alpha\) words of an English sentence was considered as the aligned sentence for the English sentence. Currently, we set \(\alpha\) to 5 which appeared to be a strict standard. As a result, we identified only 2780 aligned pairs, although there were 66985 English sentences in our parallel cor-

pus.

For the second task, we employed \texttt{mkclass} \[17\] that was offered by the same provider of GIZA++. This tool clusters words in parallel text, so we used the 2780 pairs we just explained as the input. There were 9313 Chinese types, 71174 Chinese tokens, 10608 English types, and 69403 English tokens in these sentences. We used the default number of clusters, 80, to cluster words in these sentences.

Using the aligned sentence pairs and the information about the word clusters, we applied GIZA++, with its default settings, to find the probabilities for translating English words into their Chinese counterparts, i.e., \(Pr(C | E)\).

Given the probability information about how an English word was translated into another Chinese word, we can compute the probabilistic score of a Chinese sequence \(S_c\) for \(S_e\) with \(\alpha\) \label{eq:score}. (If \(C_{(j,l)}\) is a special transla-

tion that we explained in Section 2.1, we skip \(j\) in \(\alpha\))
because we do not have statistics about these special terms.)

\[
Pr(S_e | S_r) = \prod_{i=1}^{n} Pr(C_{i|d(i)} | E_i)
\]

(1)

Using SRILM to obtain the bi-gram statistics was actually very easy. We simply provided the Chinese portion of our parallel corpus, and SRILM would compute the bi-gram statistics, i.e., \( Pr(C' | C'') \), where \( C' \) and \( C'' \) represent two Chinese words. (We could have obtained bi-gram statistics for English had we provided the English portion of our corpus.) With the bi-gram statistics, we can compute the probabilistic score of a Chinese sequence \( S_e \) with (2). When \( i = 1 \), we set \( Pr(C_{i|d(i)} | C_{i-1|d(i-1)}) \) to 1 in the prototype.

\[
Pr(S_e | S_r) = \prod_{i=1}^{n} Pr(C_{i|d(i)} | C_{i-1|d(i-1)})
\]

(2)

Using (3), which combines (1) and (2), is also a possible alternative.

\[
Pr(S_e | S_r) = \prod_{i=1}^{n} Pr(C_{i|d(i)} | C_{i-1|d(i-1)}) Pr(C_{i|d(i)} | E_i)
\]

(3)

2.3 Evaluation

2.3.1 Test data and measurement of quality

We evaluated our prototypes with some test items used in TIMSS 1999 and all items used in TIMSS 2003. The choice of data was only due to availability. Table 1 shows the quantities of test items and their categories, and the last row shows the names of the subset of items. There are three fields in each entry: the leading two digits come from the last two digits for “Year”; the middle character indicates whether the “Area” is math or science; and the last digit is copied from “Grade.”

Table 1. Quantities of our TIMSS test items

<table>
<thead>
<tr>
<th>Year</th>
<th>1999</th>
<th>2003</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grade</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Area</td>
<td>math</td>
<td>science</td>
</tr>
<tr>
<td>Quantity</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>ID</td>
<td>99m8</td>
<td>99s8</td>
</tr>
<tr>
<td>99m4</td>
<td>30m4</td>
<td>03m4</td>
</tr>
<tr>
<td>99s8</td>
<td>31</td>
<td>70</td>
</tr>
<tr>
<td>99m8</td>
<td>03m8</td>
<td>03s8</td>
</tr>
</tbody>
</table>

To study the influences of the areas of test items and grades that the test items were designed for, we organised the test data in six groups that are shown in Table 2. The grouping of test data aimed at comparing the influences of corpus categories on the quality of our recommendations. For instance, SA includes math items, while SB includes science items for the eighth graders. SB and SD include science items for the eighth and the fourth graders, respectively. SE and SF include all test items, respectively, for the eighth and fourth graders.

Table 2. Groups of test data

| SA | 99m8+03m8 |
| SB | 99s8+03s8 |
| SC | 03m4 |
| SE | 99m8+99s8+03m8+03s8 |
| SF | 03m4+03s4 |

Since we have the Chinese test items that were actually used in 1999 and 2003, we could use these items as the reference answers. We created a raw recommended translation of an English item by concatenating the first word in the pull-down menu for each \( E_i \). With a pair of reference answer and a recommended translation, we can evaluate the quality of translation with NIST and BLEU scores [20].

Recall that our system allows translators to do post edition to the raw recommended translation. The translator can change the word orders, modify the recommended words, and even add more words to improve the sentence patterns. Hence, using the raw recommendation in evaluation offers only the worst-case view of our system.

2.3.2 Test procedures and experimental results

Two popular Internet portals, Google and Yahoo, offer online translation services [6]. Hence we compare the raw recommendations of our prototype with the translations offered by Google and Yahoo. Due to the limits of space, we cannot provide results of all experiments in which are evaluated effects of different factors that we discussed in the previous subsection. In addition, we will show only the cumulative BLEU 4-gram scores that we obtained by running mteval-v10.pl [20], although scores for unigram to 9-gram were available.

Table 3 shows the cumulative BLEU 4-gram scores of Google, Yahoo, our prototype when we consider only corpus frequency \( f_k \), and our prototype when we considered only the formula shown in (3) for ordering the candidate translations. A larger score indicates better match between the recommended translations and the reference answers. Scores on the last row with the “random” heading were obtained when we randomly choose a word from each \( C_i \), and were meant to serve as a baseline for comparing performance of different systems. Clearly, the statistics show that Google, Yahoo, and our prototype outperformed the random method.

Table 3. Cumulative BLEU 4-gram scores

<table>
<thead>
<tr>
<th>Data</th>
<th>SA</th>
<th>SB</th>
<th>SC</th>
<th>SD</th>
<th>SE</th>
<th>SF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>0.0984</td>
<td>0.0677</td>
<td>0.0993</td>
<td>0.0171</td>
<td>0.0434</td>
<td>0.0523</td>
</tr>
<tr>
<td>Yahoo</td>
<td>0.1051</td>
<td>0.0842</td>
<td>0.1111</td>
<td>0.0332</td>
<td>0.0962</td>
<td>0.0705</td>
</tr>
<tr>
<td>( f_k )</td>
<td>0.2130</td>
<td>0.1505</td>
<td>0.2099</td>
<td>0.2229</td>
<td>1.66</td>
<td>2.202</td>
</tr>
<tr>
<td>(3)</td>
<td>0.0786</td>
<td>0.0962</td>
<td>0.1991</td>
<td>0.0290</td>
<td>1.666</td>
<td>1.317</td>
</tr>
<tr>
<td>random</td>
<td>0.0201</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0096</td>
<td>0.0000</td>
<td>0.0093</td>
</tr>
</tbody>
</table>

Our prototype consistently performed better than Google and Yahoo, when we considered only \( f_k \); but performed similarly with, though sometimes better than, Google and Yahoo, when we employed (3).

The fact that using information based on more complex language models, i.e., using (3), did not lead to better performance when we used only the corpus frequencies is disappointing. However, there is at least a reason to it. Since we did not change the word order in Chinese before we optimised the path probability in (3), optimizing (3) might actually impair the overall performance.

Comparing the corresponding data in the SA and SB columns (and the corresponding data in the SC and SD columns) shows that, on average, finding a good \( S_e \) for \( S_r \) is easier for math items than for science items.

Data in the SE and SF columns indicate that the compared systems performed similarly when making recommendations for test items for the fourth and the
eighth graders. (One might insist that the task for the eighth graders was easier.) A more careful comparison shows that the subjects influence the quality. Corresponding data in SB and SD columns show that grades of students affected the quality of recommendations for science items very much. Most of the time, making recommendations for items for the eighth graders is easier than for items for the fourth graders. In contrast, data in SA and SC columns show that there were no obvious differences in making recommendations for math items for the fourth and eighth graders.

3. Authoring Test Items for Chinese

We have implemented an environment for authoring four types of test items for Chinese, including Chinese pronunciation, Chinese cloze tests, reconstructing Chinese sentences, and identifying incorrect Chinese characters. Due to space limits, we present only the last two services.

3.1 Reconstructing Chinese sentences

Word orders are important for Chinese, like many other natural languages. Hence it is important for students to learn the correct word orders, and we would like to help teachers create test items for this capability. We split a Chinese sentence into meaningful chunks, shuffle the chunks, and ask students to reconstruct the original order of the shuffled chunks.

Parsing techniques offer a direct answer to the need of splitting a sentence into meaningful chunks. Figure 2 shows a simplistic example, where a parse tree for a Chinese sentence of five words (denoted by $C_i$) is shown. This tree has four non-terminal nodes, including the sentence node (denoted by $S$) and three internal nodes (denoted by $P_i$).

A parse tree like the one shown in Figure 2 is instrumental to the authoring of increasingly challenging items for sentence reconstruction. If we cut the tree at $L_0$, we split the sentence into two substructures $C_1\cdot C_2$ and $C_3\cdot C_4\cdot C_5$; if we cut the tree at $L_1$, we split the sentence into three substructures $C_1\cdot C_2$, $C_3$, and $C_4\cdot C_5$; and if we cut the tree at $L_2$, we split the sentence into five parts. Although we have not used such test items in real tests, these characters demands a lot of computation. The Cangjie method [1] is one of the popular methods for people to enter Chinese into computers. Characters are decomposed based on the forming components, in general. Figure 4 shows the Cangjie codes for some characters in Figure 3. Each character formation methods, Chinese characters may share the same components [13], making these characters look similar to each other.

An item for testing students’ familiarity with a target character is a sentence which contains an incorrect character that replaces the correct character. Given such a sentence, students need to find and correct the wrong character.

3.2 Identifying incorrect Chinese characters

Learning Chinese characters is a tough task, even for native speakers. Different Chinese characters can have the same pronunciation, i.e., homophones. Due to the pronunciation systems in Chinese, the average number of characters that share the same pronunciation in Chinese far exceeds that in English. In addition, due to the character formation methods, Chinese characters may share the same components [13], making these characters look similar to each other.

We acquire parse trees for Chinese sentences with the CKIP Chinese parser [2]. Our system allows students to read Chinese words in multiple forms, including traditional Chinese characters, simplified Chinese characters, and the Romanised Chinese form (i.e., Hanyu pinyin and Zhuyin [3]), so that the test items can be used for students of various backgrounds.
knowledge about the formation of Chinese characters [13]. Consider the examples in Figure 3. Some characters are decomposed vertically, e.g., the middle row in the middle block; some characters are decomposed horizontally, e.g., the second and the third row in the rightmost block; and some have enclosing components, e.g., the third row in the middle block. Hence, we consider the locations as well as the number of shared components in determining the similarity between characters.

4. Concluding Remarks

We demonstrated the applications of NLP techniques to assist the tasks of item translation and item authoring. The reported prototypes have shown their potential, although we are still strengthening their functions.

The study on computer assisted English-to-Chinese translation has started at least two decades ago, e.g. [24], and has become a major research area. The application of machine translation techniques for assisting test item translation between English and Chinese versions is a relatively new attempt.

Our prototype offers Chinese term sequences for TIMSS items in English so that translators can edit and make the Chinese sequences into test items. The prototype helps the translators abide by guidelines for translating TIMSS items at the same time. Due to the use of information about domain-dependent translation obtained from the translators, the prototype performed comparably or better than Google and Yahoo in terms of the provided n-grams, when evaluated with TIMSS items that were used in 1999 and 2003.

The prototype environment for authoring test items facilitates teachers to prepare practical test items for introductory Chinese. We showed how NLP techniques can help us prepare test items of different challenging levels, and illustrated how information about Chinese character formation can save us work in computing visual similarity between Chinese characters.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments on the submitted manuscript. This final version has been revised to meet some of the comments, and the authors wish to respond to other comments in the oral presentation. This project was funded in part by grants NSC-95-2213-E-004-013-MY2, NSC-94-2213-E-004-008, and NSC-94-2815-C-004-006 from the National Science Council and a grant for excellent research centers from the Ministry of Education in Taiwan.

References

2. CKIP Chinese parsing service. <parser.iis.sinica.edu.tw>
3. Pinyin methods. <en.wikipedia.org/wiki/Pinyin>
4. International Association for the Evaluation of Educational Achievement. <www.iea.nl/>
9. The JACOB Project. <danadler.com/jacob/>
Evaluating and Improving the Distractor-Generating Heuristics

Itziar Aldabe, Montse Maritxalar (1), Edurne Martinez
Department of Computer Languages and Systems
University of the Basque Country
P.O. box 649, E-20080 Donostia
(1) montse.maritxalar@ehu.es

Abstract
ArikIturri is a system developed for the automatic generation of didactic resources. More specifically, it is focused on the automatic generation of questions based on NLP tools. In this paper, we present an evaluation of the error correction and multiple-choice question types generated by ArikIturri. This evaluation has been carried out with the collaboration of four expert editors. In the mentioned question types, heuristics have been used in order to generate the distractors automatically. Although the heuristics have been manually defined, a first attempt for their automatic generation is explained.

Keywords
question generation, evaluation, heuristics

1 Introduction
In the last years, several research on automatic generation of questions for language learning has been carried out. Among others, reading comprehension, vocabulary, cloze questions and grammar tests are automatically generated based on Natural Language Processing (NLP) techniques and resources. These tests deal with different topics related to specific linguistic phenomena or to more general subjects such as the comprehension of a text. We call topic to the subject matter of a question.

The evaluation of the systems can also be made from different perspectives. For example, [8] evaluates the system in terms of time (to produce distractors) and quality of the items. The questions previously approved by a linguistic lecturer are evaluated by students. [6] also evaluates the system giving the cloze tests to 60 students. [7] makes use of authoring and assessment subsystems in order to evaluate the generated questions. In [3] the vocabulary questions are compared to human-generated questions, while in [4] the grammar questions are evaluated by seven professor and students. Finally, [9] examines the quality of the questions with the help of a native speaker of English.

In this paper, we compare the opinions of different human editors about the question automatically generated by ArikIturri. Firstly, the questions created by the generator are evaluated by a single editor, and then, the evaluation is extended to some more editors in order to compare their points of view. Moreover, the improvements of distractor-generating heuristics is also the aim of our work. A distractor is a choice which does not match correctly in the context of the question and a heuristic is the rule or the knowledge the system uses to generate the distractors. The heuristics have been defined manually whereas the distractors are automatically generated words. In this paper, we also explain a first attempt to generate the heuristics automatically.

Section 2 presents the question generator. In section 3, the editors’ agreement is analysed. Section 4 deals with different ways of producing distractors. Finally, some conclusions and future work are outlined.

2 The Question Generator
ArikIturri [1] is a system developed for the automatic generation of didactic resources based on NLP tools. The system generates different types of questions using pedagogical corpora. Although we have developed it for the Basque language, the architecture of the system is language independent.

The input corpus consists of a databank which is composed of morphologically and syntactically analysed sentences where phrase chunks are automatically identified. Question instances of a question model consist the output of the system. Both input and output are represented in XML.

Figure 1 represents the automatic process for question generation.

The sentence retriever module in ArikIturri selects candidate sentences from the source corpus. In a first step, the candidate sentences for the questions are automatically extracted from the databank, depending on the topic of the question. Then, it analyses the occurrences rate of the possible candidates in order to make random selection of the sentences.

Once the sentences are selected, the answer focuses identifier tags the chunked phrases (answer focuses) where the topic to be treated appears. Then, the item generator creates the questions depending on the specified exercise type. That is why this module contains the distractor generator submodule. Distractors are automatically generated words; they are not extracted from any databank. Due to the rich inflection system
of Basque, it is impossible to store every possible word form in a dictionary, even in a compressed way. Therefore, we use a general purpose morphological generator to create the distractors. By contrast, the heuristics used by the generator are not automatically generated, they are based on experts’ knowledge. In section 4 we will explain some experiments carried out for the automatic generation of the heuristics.

As the question generation process is automatic, it is probable that some of the questions are ill-formed. This is why we have included the ill-formed questions rejecter in the architecture of our system.

One of the aims in [1] was to prove the viability of ArikIturri when constructing questions. Even though the system was able to produce four different types of questions, we limited our evaluation to multiple-choice question types. The system used 17 heuristics for the generation of the distractors (see table 1) were based on the knowledge of an expert who took part in the design of the system but not in the evaluation. It is also important to remark that the heuristics were defined all in a row by a single person.

3 Editors’ agreement

One way of evaluating the questions generated by ArikIturri is to give them to different editors. In this section, we present two experiments carried out for the manual evaluation of ArikIturri.

It is necessary to underline the fact that the heuristics used for the generation of the distractors (see table 1) were based on the knowledge of an expert who took part in the two experiments as editors: three of them work as computational linguists in a NLP research group and the last one in HABE (Institute for the Teaching of Basque and Basque Language Literacy to Adults), which is an institute of the Basque government developed for L2 and L1 Basque Language Teaching. The computational linguists do not only have a linguistic profile but also a language teaching background. In the case of the expert language teacher from HABE, he has teaching profile as well as experience in creating didactic resources.

For the two experiments, the editors used a web-based post-editing environment which helped them to set the questions. In the first experiment, the manual evaluation was carried out by a computational linguist while in the second experiment three new editors took part in it. The aim of these experiments was to compare the opinions of different editors.

3.1 Experimental settings

Four experts in both computational linguistics and language teaching took part in the two experiments as editors: three of them work as computational linguists in a NLP research group and the last one in HABE (Institute for the Teaching of Basque and Basque Language Literacy to Adults), which is an institute of the Basque government developed for L2 and L1 Basque Language Teaching. The computational linguists do not only have a linguistic profile but also a language teaching background. In the case of the expert language teacher from HABE, he has teaching profile as well as experience in creating didactic resources.

For the two experiments, the editors used a web-based post-editing environment which helped them to set the questions. In the first experiment, the manual evaluation was carried out by a computational linguist while in the second experiment three new editors took part in it. The aim of these experiments was to compare the opinions of different editors.

The sample of 1,350 question instances mentioned in section 2 was evaluated in the first experiment. One expert evaluated 442 multiple-choice questions and 908 of the error correction type. In the case of multiple-choice, the topic of 153 questions was the declension case and the other 289 questions had the verb tense instances was obtained to be evaluated manually.

Table 1: Heuristics

<table>
<thead>
<tr>
<th>Declension cases</th>
<th>Replacement of declension cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change of the finiteness</td>
<td>Sociative</td>
</tr>
<tr>
<td>INE =</td>
<td>Inessive</td>
</tr>
<tr>
<td>ABS =</td>
<td>ERG =</td>
</tr>
<tr>
<td>DAT =</td>
<td>SOZ =</td>
</tr>
<tr>
<td>DA paradigm</td>
<td>ZAO paradigm</td>
</tr>
</tbody>
</table>

Table 1: Heuristics

<table>
<thead>
<tr>
<th>Verb</th>
<th>Change of the person of the verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>DA paradigm</td>
<td>ZAO paradigm</td>
</tr>
</tbody>
</table>

Fig. 1: Architecture of ArikIturri

1 The input corpus was classified into three different language levels chosen by expert teachers.
2 The paradigms are explained in section 4.1.
3 We did not evaluate them manually because they were badly-formed questions.
as topic. However, in the case of the error correction questions, they all were connected with declension cases.

In the second experiment, we took 100% of the generated multiple-choice questions related to the present indicative verb tense and 25% of the error correction questions related to declension cases. The sample we obtained contained a total of 431 questions nearly the same amount for each linguistic phenomenon.

Table 2 summarises the information of the experimental settings.

<table>
<thead>
<tr>
<th></th>
<th>1st experiment</th>
<th>2nd experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Editors</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Multiple-choice Declension cases</td>
<td>153</td>
<td>0</td>
</tr>
<tr>
<td>Multiple-choice Verb tenses</td>
<td>289</td>
<td>195</td>
</tr>
<tr>
<td>Error correction Declension cases</td>
<td>908</td>
<td>236</td>
</tr>
<tr>
<td>Total amount of questions</td>
<td>1350</td>
<td>431</td>
</tr>
</tbody>
</table>

Table 2: Experimental settings

3.2 First experiment

In the evaluation of the first experiment, we asked the editor to modify or reject questions only if they were badly-formed. Considering that all the questions discarded or modified by the editor were not well generated, the results showed that the rate of the accepted questions was 82.71% in the case of error correction questions and 83.26% in the case of multiple-choice questions.

Let us assume that the probability to generate a proper distractor and consequently a theoretically acceptable question in error correction is \( P(dist) = p \) where \( 0 \leq p \leq 1 \). However, the probability to generate an acceptable question decreases in multiple-choice questions (3 distractors in this case). If we assume that the distractors are independent with each other, the probability to create an acceptable question is \( P(dist_1, dist_2, dist_3) = P(dist_1) \cdot P(dist_2) \cdot P(dist_3) = p \times p \times p \) where \( 0 \leq p \leq p \leq 1 \). The results obtained in the experiment confirm that this probability has influence on the acceptance rate, when dealing with the same topic. For instance, in the case of declension cases the acceptance rate is 82.71% for error correction question type and 64.70% for multiple-choice.

If we split the multiple-choice questions taking into account the number of distractors we find that there is a significant difference in terms of acceptance. Regarding the verb tenses, the acceptance rate is 92.73%, while in the case of declension cases, it is 64.70%. As the system generated two distractors when dealing with verb tenses and three when dealing with declension cases, the probability of creating a correct question for verbs is higher than in the case of declension cases. However, the acceptance rate given by the human editors (92.73%) in the case of verb tenses (2 distractors) is higher than the rate (82.71%) for the declension cases (1 distractor). Table 3 displays all the acceptance rates in this first experiment.

<table>
<thead>
<tr>
<th></th>
<th>Number of distractors</th>
<th>Acceptance rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error correction Declension cases</td>
<td>1</td>
<td>82.71%</td>
</tr>
<tr>
<td>Multiple-choice Declension cases AND Verb tense</td>
<td>2-3</td>
<td>83.26%</td>
</tr>
<tr>
<td>Declension cases</td>
<td>3</td>
<td>64.70%</td>
</tr>
<tr>
<td>Verb tenses</td>
<td>2</td>
<td>92.73%</td>
</tr>
</tbody>
</table>

Table 3: Accepted questions

We conclude that the number of distractors alter the acceptance rate of the generated questions by ArikTurri. We foresaw that the topic could also have influence in the results.

3.3 Second experiment

In order to corroborate the results from the first experiment as well as our hypothesis about the topic, three new editors took part in the second experiment. Two of them were asked to evaluate the accepted questions in the first experiment. The first one had to evaluate the questions related to verb tenses, the second one those related to declension cases, and the third editor examined the questions which were rejected.

<table>
<thead>
<tr>
<th></th>
<th>Verb</th>
<th>Declension cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted in the 1st</td>
<td>94.97%</td>
<td>96.94%</td>
</tr>
<tr>
<td>Rejected in the 1st</td>
<td>75.00%</td>
<td>25.00%</td>
</tr>
</tbody>
</table>

Table 4: Evaluation of the 431 questions

Table 4 shows the results we obtained in the second experiment in comparison with the ones obtained in the first one. For example, the editor of the second experiment accepted 94.97% of the questions related to the verb tenses which were previously accepted in the first experiment and 96.94% of the declension cases. For instance, both editors agreed on the fact that the error correction question “*Industriak milaka galdu du entzumenak errespetatuko zurretan erruztu” is an acceptable question because the correct answer is “Industrian” instead of “Industriak”. 75% of the rejected questions related to verb tenses in the first experiment were also not accepted in the second one, while in the case of declension cases the percentage is 25%.

A more detailed information is given in table 5, where the number of questions in which different editors agree and disagree are displayed.

---

4 The industry lots of professionals have lost their hearing due to the factory noise

5 In the industry lots of professionals have lost their hearing due to the factory noise

---

Workshop Natural Language Processing for Educational Resources’2007 - Borovets, Bulgaria 9
Both tables show good results, in fact, the second experiment also verifies the high percentage of well-formed questions. The favourable opinion of the four editors is also an important aspect since the questions were automatically generated.

However, although it is supposed that all the editors followed the same instructions for the evaluation of the automatically generated questions, we must also consider other aspects, such as chance or some personal factors, which might have influence on the results obtained in the evaluation. Those factors could be i) editors’ own experience when generating questions manually; ii) the final users of the questions they are thinking about; iii) when and how the evaluation was carried out; iv) the number of questions to evaluate, etc. Cohen’s kappa index ($\kappa$) [5] takes into account this variable.

If we apply the kappa concept to our results, we obtain the kappa indexes displayed in table 6.

<table>
<thead>
<tr>
<th></th>
<th>Declension cases</th>
<th>Verb tenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted in 2</td>
<td>190</td>
<td>170</td>
</tr>
<tr>
<td>Rejected in 2</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Accepted in 1</td>
<td>184</td>
<td>170</td>
</tr>
<tr>
<td>Rejected in 1</td>
<td>10</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 5: Comparison of the results of the two experiments

If we take into account that there are more distractors in a multiple-choice question than in an error correction question, the probability that two editors will agree is higher in the case of error correction. In the case of multiple-choice questions, they must agree on all the different distractors. Therefore, as the number of questions evaluated for each question type was almost the same, we should expect better kappa indexes in the case of declension cases since they belong to the error correction type of question. As a consequence, we conclude that the topic of the questions has influence on the results. Indeed, our hypothesis when planning the second experiment was that it was easier to generate questions to learn verb tenses than to learn declension cases.

4 Producing distractors

We have already mentioned the fact that ArikIturri makes use of different heuristics to create the distractors of the questions. The information used to define the heuristics was manually created but it could be also automatically generated. The rules represent some of the unsuitable combinations from the linguist point of view.

As our aim is also to generate the heuristics automatically, in the next sections, we explain the first attempts we have made in this research line.

4.1 Automatic extraction of patterns to define heuristics

The previous experiments have been performed having human experience on the basis of the implementation of the heuristic rules. The new approach considers an automatic process not only for the generation of the distractors but also for the generation of the heuristics.

Before focusing on the automatic extraction of patterns to define the heuristics, we consider necessary to clarify two aspects:

- Verb and ellipsis: The auxiliary verbs in Basque refer to the syntagmatic component where the ergative, absolutive and dative cases occur. Even if those phrases do not appear in the sentence, the auxiliary gives us that information and we can know which phrases have been elided.

  In general, a verb can have from one to four different auxiliary paradigms. These paradigms correspond to the following four auxiliary types:
  
  - DA: the absolutive is the subject of the clause.
  - DU: the ergative is the subject and the absolutive is the direct object of the clause.
  - DIO: the ergative is the subject, the absolutive is the direct object and the dative is the indirect object of the clause.
  - ZAIO: the absolutive is the subject and the dative is the indirect object of the clause.

- Working unit: In this article, the term clause refers to a group of phrases containing a conjugated verb. A sentence that contains only one clause is called a simple sentence; if we have two or more sentences (juxtaposition, coordination or subordination), we speak of complex sentences. Therefore, we consider two different working units: the simple sentence level and the complex sentence level.

The basis of the automatic extraction of patterns to define heuristics comes from [2], where a finite-state syntactic grammar was developed in order to join the verb instances and their corresponding syntactic dependents (arguments and adjuncts) from journalistic corpora. The grammar scores 87% of precision and 66% of recall. The system obtained 688 different patterns for 640 verbs. For each verb more than one of the different 688 patterns can occur.

The patterns which represent the knowledge extracted from automatically analysed corpora were obtained at simple sentence level. Moreover, they automatically retrieved the elided cases in order to reflect
them in the patterns. Each of the patterns offers the following information: i) the syntactic dependents; ii) the auxiliary type and iii) the number of instances. For instance, one of the 143 extracted patterns related to the verb “askatu” (to release) is:

48 askatu: DU: ABS + ERG + INE

Based on the journalistic corpora, the system developed in [2] matched 48 times the DU: ABS + ERG + INE pattern for the verb “askatu” (to release), that is, it reflects the number of times that the absolutive, the ergative and the inessive occur with the auxiliary DU.

We consider these patterns could be the basis for the automatic generation of the heuristics. When the morphological generator creates a distractor, the patterns automatically extracted and the created distractor pattern could be compared. If a matching was detected, the distractor could not be considered a candidate distractor and the question would be automatically rejected.

The experiments previously carried out offered us the chance to compare the heuristics manually generated with the automatic patterns. Moreover, as the questions were already manually evaluated, we could study the measure of success of the patterns.

Once the clauses of the questions are extracted, it is important to specify which phrases are going to be taken into consideration when matching them to the patterns. For example, considering that the system has generated the following error correction question when setting absolutive as the topic, we will study different criteria for comparing the automatically extracted patterns with the phrases of the question generated by ArikIturri:

“Hainbat ariketaren bidez gure gorputzaren blokeoarekin askatu dugu”\(^6\)

The phrase containing the correct answer (“gure gorputzaren blokeoartea”) is absolute and it has been transformed into the sociative (SOZ) case in order to generate the distractor (“gure gorputzaren blokeoarekin”). The phrase “Hainbat ariketaren bidez” refers to the instrumental case (INS) and the auxiliary for the verb “askatu” is DU.

The auxiliary DU for the verb “askatu” tells us there is a subject (ERG) as well as a direct object (ABS). The criteria to compare the clause of the question with the patterns can be summarised as follows:

- **Criterion 1**: to compare the patterns with the declension cases/phrases that appear in the clause explicitly. In the previous example, in the case of the distractor, DU: INS + SOZ would be compared with the patterns from [2]. As there is no matching, ArikIturri would create a distractor.

If we want to take into account the phrases containing some given declension cases that occur in a clause plus those which are elided, we can follow two different options:

- **Criterion 2**: to contrast the cases which have been elided, if they are not part of the topic. In the example, the system would compare DU: INS + SOZ + ERG with the automatic patterns. That is to say, we would take into consideration the ergative case because it is elided and it is not the topic. On the contrary, we would not consider the absolutive case because it is the topic of the question. In this case the system would generate a distractor, since it does not match any of the 143 patterns extracted for the verb “askatu”.

- **Criterion 3**: to include all the elided cases. In that example, we would compare the paradigm DU: INS + SOZ + ERG + ABS with the patterns. As this distractor pattern exists in [2], the system would not generate a distractor.

In the next two sections, we explain the two approaches followed in the automatic generation of the heuristics. The first one was developed to generate complex sentence questions while the second one was carried out to create simple sentence questions.

### 4.2 Heuristics based on patterns to generate complex sentence questions

The first attempt was to compare the evaluated questions in the first experiment with the patterns automatically extracted from journalistic corpora. For that, some steps were followed:

1. To obtain a sample of the error correction questions related to the sociative, inessive, ergative, dative or absolutive cases. This was the same sample as the one used for the second experiment (25% of error correction questions).

2. To extract the simple sentence of each question where the topic appeared. This task was hand-made. When the topic was part of the subordinate clause, the subordinate clause was manually transformed into a main clause.

3. To compare the questions (at clause level) with the patterns in order to observe the acceptance rate if, in the basis, we had an automatic generation of heuristics based on the automatic patterns.

As editors agreed in a high rate in the previous experiments, we first made a study of the heuristics used in the generation of well-formed questions. We compared the questions accepted in the first experiment with the patterns. That is to say, the information of the well-formed questions was divided to compare both the correct answer and the distractors.

If we applied the **criterion 1**, we might expect low results since it only takes into account the explicit phrases of the question, while the patterns automatically assign the explicit declension cases of the verb as well as those that are elliptic. Nevertheless, low results are obtained in the case of the correct answers, but not in the case of the distractors.

Table 7 shows the results related to the three different criteria when comparing both data of the questions accepted in the first experiment:

For instance, if we had, in the basis, the automatically extracted knowledge (patterns) when using **criterion 1**, 66.27 out of the 100 correct answers accepted...
by the editor in the first experiment would be also considered correct answers. Besides, 69.94% refers to the clauses of the questions that were accepted by the editor as distractors. Almost 70% of the distractors of the questions would also be considered distractors if the patterns were used for the automatic generation of the heuristics.

Regarding the results obtained in the case of the criterion 2, we could conclude that they are more realistic and better. As the correct answers were extracted from the source sentence of the corpus, they are presumably correct.

The number of the questions created by ArikIturri and rejected by the editors is not troubling, since it is a low percentage\(^8\). However, the patterns can also be compared with them in order to observe if the distractors rejected by the human editor would not be created having the patterns as basis of the heuristics.

Table 8 represents the percentages related to the rejected distractors in the first experiment together with the three different criteria.

<table>
<thead>
<tr>
<th></th>
<th>Criterion 1</th>
<th>Criterion 2</th>
<th>Criterion 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT distractor in the 1st experiment</td>
<td>15.38%</td>
<td>19.23%</td>
<td>48.00%</td>
</tr>
</tbody>
</table>

Table 8: Rejected questions

As it happens in the case of the accepted questions, in this case the way to compare the questions with the patterns is threefold. This time the given percentages have a different meaning. Since the questions were rejected by the editor, it is supposed that the rejected distractors were not proper ones\(^9\). If we used the automatically extracted patterns to create heuristics and compare them only with the explicit phrases of the clause, 15.38% of the distractors would be considered improper distractors in the case of criterion 1. In this case, better results are obtained from the third comparison.

We foresaw some aspects that could affect the results: the error rate of the patterns, the corpus and the working unit. The error rate of the patterns may alter the results since their precision is 87%. Therefore, 13% of the times the patterns obtained in [2] are incorrect. As regards the corpus, it could also be an aspect to be considered because different corpora have been used in both works. In the case of ArikIturri, the corpus is focused on language learning while in the automatic extraction of patterns the corpus is composed of newspaper texts. Finally, the experiments commented in section 3 were carried out using complex sentences while the extracted patterns refer to simple ones. Therefore, the working unit could have influenced on the results.

4.3 Heuristics based on patterns to generate simple sentence questions

The fact that the working unit could affect in the results made us carry out a new experiment in order to obtain heuristics based on patterns to generate simple sentence questions. This time, we presented the editors new questions to be evaluated. Those questions were simple sentences manually extracted from the complex sentences used in the previous experiments. As we have said, we used a sample of 25% of the error correction questions which were related to the sociative, inessive, ergative, dative and absolutive cases.

The editors evaluated the questions and accepted 75.21% of them. If we compare it with the acceptance rate in the first (82.71%) and second (93.22%) experiments, the acceptance rate decreases. These results correspond to the generated error correction question types related to the declension cases. The only difference lies in the evaluated sentences: the one from the first and second experiment were complex sentences, whereas in the last one they were simple sentences.

Once we obtained a set of questions manually evaluated, we compared them with the automatically generated patterns. This time, we used the three criteria previously mentioned again. Table 9 shows the patterns accuracy taking into account the accepted questions (75.21%).

Table 9: Accepted questions at simple sentence level

<table>
<thead>
<tr>
<th></th>
<th>Criterion 1</th>
<th>Criterion 2</th>
<th>Criterion 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct in the 1st experiment</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Distractor in the 1st experiment</td>
<td>79.19%</td>
<td>78.05%</td>
<td>77.64%</td>
</tr>
</tbody>
</table>

Table 10: Rejected questions at simple sentence level

As it happens in the case of the accepted questions, in this case the way to compare the questions with the patterns is threefold. This time the given percentages have a different meaning. Since the questions were rejected by the editor, it is supposed that the rejected distractors were not proper ones\(^8\). If we used the automatically extracted patterns to create heuristics and compare them only with the explicit phrases of the clause, 15.38% of the distractors would be considered improper distractors in the case of criterion 1. In this case, better results are obtained from the third comparison.

We foresaw some aspects that could affect the results: the error rate of the patterns, the corpus and the working unit. The error rate of the patterns may alter the results since their precision is 87%. Therefore, 13% of the times the patterns obtained in [2] are incorrect. As regards the corpus, it could also be an aspect to be considered because different corpora have been used in both works. In the case of ArikIturri, the corpus is focused on language learning while in the automatic extraction of patterns the corpus is composed of newspaper texts. Finally, the experiments commented in section 3 were carried out using complex sentences while the extracted patterns refer to simple ones. Therefore, the working unit could have influenced on the results.

\(^{8}\) 17.29% in the first experiment for error correction question types and 6.78% in the second one

\(^{9}\) The error correction question type has just one distractor.
Although the results are better than the ones obtained in table 8, they are still quite poor. In case of the criterion 1 and criterion 2 the results are twice better, but still poor.

The comparison of the results obtained in the evaluations show us that a clause considered a question at complex sentence level is not always a question at simple sentence level, and vice versa. Moreover, the editors took into account the elided sentence elements when evaluating the questions at simple sentence level. Finally, we also conclude that the best results are obtained from criterion 2.

5 Conclusions and future work

In this paper, we have presented an automatic question generator, which was evaluated in different ways. First, one editor evaluated multiple-choice and error correction question types created by ArikIturri. The error correction questions were generated to deal with declension cases, and the multiple-choice questions cope with declension cases as well as with verb tenses. Apart from obtaining a high acceptance rate in the automatically generated questions, we can conclude that automatically generate error correction questions are more reliable than the multiple-choice ones.

After that, the evaluation was extended to three new editors in order to compare the different opinions they might have. In all cases, the number of the accepted questions was high, and so it was the agreement among the editors. Moreover, based on the obtained results, we can also conclude that the topic has influence in the automatic generation of questions.

In the generation process of the questions that were manually evaluated by different editors, the distractors were automatically generated, but the heuristics were manually defined. Nevertheless, one of the purposes of this work was to try to automatize this process. Section 4 deals with it, comparing the patterns automatically extracted from journalistic corpora with the patterns of complex sentence questions and simple sentence questions. The automatic generation of heuristics to create distractors is more reliable if we choose simple sentences as questions instead of complex ones. However, the acceptance rate of the same questions is higher in the case of complex sentences: 82.71% in the first experiment, 93.22% in the second experiment with complex sentences and 75.21% in the last experiment with simple sentences.

As future work, we plan to continue working on these heuristics. We must not only to try to automatize the process but also study their combination. We plan to analyse different possibilities such splitting the heuristics taking into account the learning level, combining different heuristics to deal with the same topic, etc. Finally, as regards the evaluation carried out, we consider necessary to extend it to learners in order to obtain more realistic results.

Acknowledgements. This research is supported by the University of the Basque Country (GIU05/52) and the Ministry of Industry of the Basque Government (ANHITZ project, IE06-185).

References

What (the hell) is wrong?
An approach to semi-automatic construction of self correction tests

Manfred Klenner
Simon Clematide
Bojan Peric
Institute of Computational Linguistics
University of Zurich
{siclemat,klenner}@cl.uzh.ch, bojan.peric@access.uzh.ch

Abstract
We present a system that helps language learners to improve their grammar skills, especially the detection and classification of errors in (their own) sentences. A learner corpus of 270 ill-formed sentences has been manually corrected and classified along our error taxonomy that was adapted for the task at hand. 230 well-formed sentences are added in order to make the classification task more challenging. In setting 1, the system randomly selects a sentence and four potential error classifications, including the correct one (if an ill-formed sentence was chosen). The student’s task is to pick out the correct error message ("no error" in the case of a well-formed sentence). In setting II, the system presents an error message and 4 sentences. The task is to mark those sentences that bear the error described in the message. Rules that operate on the parse trees of the corrected versions of the sentences are used to determine those error messages that are applicable given the sentence in question. In order to produce good test items, error statistics and incompatibility filters are integrated.

1 Introduction
A common exercise in second language learning is the production of texts, e.g. the summarising of a larger text or the description of a picture. Normally, a teacher corrects the texts and the students skim over the corrections more or less attentively. Feedback, thus, is not as effective as it could be. If, however, the errors being made by the students were presented to them again as an exercise where they had to find and classify their own mistakes, the learning impact would be much higher. We aim at realizing such a scenario. The core of our conception is a component that selects an ill-formed sentence and generates a number of potential error messages including the correct one.

Take the following ill-formed sentence as an example. "*Aber er hat andere Frau" (*"But he has other wife"). Here, a determiner is missing: "Aber er hat eine andere Frau." The system generates the following error messages (1-4):

Aber er hat andere Frau
1. The word order is wrong.
2. A determiner is missing.
3. The inflection of a pre-nominal word is wrong.
4. The sentence is correct.

To find the correct error message the students have to decide whether there is an error in this sentence or not. Then they have to identify the correct error message. This requires some grammatical knowledge, mostly basic grammar vocabulary (e.g. part-of-speech tags). If the generated error messages apply well to the sentence, that is, if
they are plausible candidates of what might have been wrong in the sentence, the students can’t simply skim over them. They have to attentively check them until the single correct error message is found. The challenge, thus, is to define an algorithm such that only “good error messages” are generated given a particular sentence.

Some error messages are always applicable, e.g., wrong word order. Others depend on the presence or absence of specific part-of-speech tags, e.g., a determiner is missing. Even syntactic structures might be required, e.g., message 3 (the inflection of a pre-nominal word) only makes sense if there is a noun phrase comprising a noun and some pre-nominal word. At least a tagger, but – as we would argue – even a parser is necessary to generate well fitting error messages. We use the Papa parser [5], a weighted constraint-based dependency parser that is based on a robust broad-coverage hand-written dependency grammar.

The ill-formed sentence needs to be manually corrected and classified according to some error taxonomy, i.e., no automatic error diagnosis is presumed in our approach. As Menzel and Schröder [13] put it: “Unfortunately, robust parsing and fault diagnosis are almost incompatible tasks”.

When we parsed the ill-formed sentences in one experiment, we got the following results for the problem of missing determiners (error art-n): 13 instances are present. The parser identifies 109 violations (the constraint violation is called “Determiner fehlt” in the Papa grammar) but misses 4 real ones. Therefore, we have a precision of 8%, a recall of 60% and an f-measure of 15%.

The proposed learning scenario is meant to raise the grammar awareness of the students but also help to improve their self-explanation skills, which is an ability that psychologists claim to be central for good learners.

2 The Sentence Corpus

The empirical basis of our system is a sentence corpus comprising about 20,000 German sentences produced by Japanese second language learners (cf. [3]). The corpus is split into first-, second- and third-year learners. The type of exercise and the original material underlying the production of these sentences is unknown. We only know that the sentences are sent by email to the teachers. We randomly selected 1000 sentences from the first-year learner corpus. 10 students corrected only those sentences with exactly one error in it. This way we got about 270 pairs of ill-formed sentences and their corresponding correct versions. The next step was to analyze the errors and to design a set of error messages. We added 230 well-formed sentences to the sentence pool in order to make the classification task more challenging. The corrected versions of all sentences were parsed with the Papa parser. The parse trees form the basis of our error message selection component. Each error message has application conditions, e.g., the presence of a specific part-of-speech tag. These conditions can easily be fixed on the basis of the parse trees.

We present our set of error messages in the next section. There are 47 such messages. Most of the time, more than 10 messages are applicable per sentence. This is far too much, so some filter criterion is necessary. The n (to be presented) messages are selected according to the probability distribution of errors. This way, the frequency of an error directly determines the frequency of the corresponding error message.

3 Error Diagnosis and Messages

The presentation of error messages to the learner depends on different goals and limitations:

- How much grammatical knowledge is the learner supposed to bring along? E.g., adult vs. teenage language learner.

- What kind of grammatical terminology is he used to? In German, many different linguistic terms are used.

- How detailed are the error diagnostics provided?

In principle, error messages can easily be “localized” to the terminological abilities and
habits of a learner. More difficult questions arise regarding the error diagnostics. From a syntactic point of view one would like to have a diagnosis like "A verb in this sentence subcategorizes for accusative", given a wrong sentence like "Die Nachricht regte ich auf" ("The message excited me"). Understanding such explanatory oriented error diagnostics requires a common set of quite abstract grammatical rules and principles. In order to locate the potential error, one has to semantically identify the intended direct object with its wrong nominative case. In the first stage of our work we used this kind of explanatory error messages that referred to linguistic concepts such as agreement, subcategorization and government. While testing and developing our test scenarios we decided to change our error diagnostics to a more descriptive, superficial, and word oriented analysis. Two reasons motivated us to do so:

- Interpreting explanatory diagnosis is fatiguing, because you have to apply the abstract principles several times (depending on the test scenario). Seeing an error message concerning e.g. a pronoun or a verb gives you more guidance in locating potential errors.

- Error messages in a single choice test setting should be unambiguous. In combination with more general error diagnostics, explanatory messages can lead to conflicting equally true cases. To give the correct answer the learner would have to know which kind of answers are preferred if more than one is applicable.

On an explanatory level, there are quite different diagnostics for an ill-formed sentence like "*Dank ihre Beschäftigung kann sie Tennis spielen." ("Owing to her occupation she can play tennis."). One error analysis may find the culprit in wrong case requirements of the preposition "Dank": accusative instead of the correct dative. Another possible diagnosis may see an agreement error in a dative nominal phrase. Or we just state that "ihre" is morphologically wrong.

We can only speculate about the real motivation of the learner producing such an error. When classifying errors, language teachers should not need to think about the many different reasons for an error. Our classification is therefore optimized for a more descriptive error diagnosis where overlapping error cases are minimized (word orientation). Additionally, on the second more fine-grained level of error diagnosis conflicting error cases are quite easily spotted by our program and suppressed through simple patterns.

### 3.1 Used Error Categories

The main level of error diagnosis applies to part-of-speech (A: involves a single word), word order (B: involves more than one word) or orthography (C).

More fine-grained diagnostics form a second level of diagnostics. The part-of-speech tags follow the fairly traditional classification of the Stuttgart-Tübingen-Tagset (STTS) [14] (also used in the EAGLES standard [2]). This includes some German specific classes as the particle "zu" or fusions of determiners and prepositions ("im") or separable verb prefixes that are relevant for both, language learners and automatic taggers.

- A. adj (adjective), art (determiner and attributive pronouns), adv (adverb), es (impersonal pronoun "es"), konj (conjunction), neg (negation particle), partzu (special particle "zu"), prp (preposition), prpart (fusion of preposition and determiner), v (verb), vaux (auxiliary verbs), vpart (separable verb prefixes). A second level of classification indicates the error type and appends to the first level: -f (generally wrong), -fx (wrong inflection; conjugation for verbs and declination for nouns, determiners, pronouns and attributive adjectives), -fo (wrong orthography). -z (a word too much), -m (a word is missing), -sfw (semantically wrong usage of a word). Special cases: The combination vaux-f means "wrong auxiliary used", vpart-ng means "verb particle not separated", prcn-mr means "reflexive pronoun is missing".

B. we (word order): we-f = wrong word order (general); we-fn = in subordinate clause; we-fh = in main clause;

C. int (punctuation error), 1gk (capitalization error), 1c-g (words are not separated or written together)

Our German standard error messages used in the evaluations are formulated in a regular and concise way. The erroneous part is
the first information conceived when reading from left to right. E.G. “Eine falsch konjugierte Verbform liegt vor.” (There is a wrongly conjugated verb.) Every instance of POS-tx errors follow this pattern “Eine falsch konjugierte/deklinierte ... liegt vor.” For the other error types similar patterns are formulated. Using the same kind of error formulation again and again may be seen as tedious for the learner. On the other hand, the learners may be confused if the same error must be spotted by quite different formulations. A decision on this point must be based on usability studies with real students. Still, our adult non-native testees especially liked the clean and systematic nature of the messages.

3.2 Other Error Classification Taxonomies

For German, at least two other error taxonomies are available. Heringer [7] provides a linguistically fine-grained taxonomy with 157 different error diagnostics for his Augsburger error corpus (circa 7000 errors). The taxonomy is hierarchically organized and the top categories partially consist of part-of-speech categories as in our case. But he also includes top categories as ellipsis (missing elements), “plus” (obsolete elements), congruence, and morphological errors that are related to part-of-speech information on a more fine-grained level. This classification scheme produces quite a lot of diagnostic instances where more than one classification is possible. One of the important points in the introduction to the CD-ROM [6] is to inform the learner that other than the annotated error classifications may be true.

Another error classification developed for a learner corpus in German is described in [12]. The identification, description and explanation of errors is separated in different annotation levels. An interesting point in their corpus is the ability to include more than one target hypothesis. Spelling out the different possible intended utterances allows annotating them with their respective error classification. In our case such a disjunctive annotation would give us support for different interpretations on the learner’s side. However, in our scenarios such an elaborate correction and error description is not feasible since our task does not consist in constructing a detailed learner corpus. As our evaluations indicated it seems that incompatible target hypotheses are not very problematic – our descriptive error messages and the restriction to at most one error per sentence may be helpful.

There are some cases where we had to add a context sentence that makes one target hypothesis mandatory. This is a negative consequence of the fact that our corpus contains isolated sentences. “Ihr hat Unterricht.” (“You have class”) In order to promote a specific target hypothesis we add something like “Ihrer Lehrer erwartet euch.” (“Your teacher is waiting for you.”).

4 Scenario I: Error Message Selection

We now turn to the question of how to determine adequate error messages to be presented for a given sentence? Take e.g. error message 12: “a fusion of the preposition and the determiner is needed”. Of course, a preposition and determiner must be present in the sentence, otherwise the message is not appropriate. But the preposition also must immediately precede the determiner, i.e. a structural condition has to be verified.

Inverting error message 12, we get message 14, i.e. “a fusion of a preposition and a determiner is wrong”. This rule is only appropriate if such a fusion is present in the sentence. Other messages require structural relations to be checked, e.g. whether a subordinate clause is given. Most of the time, however, the presence or absence of one or more part-of-speech tags is sufficient. For example error message 2: “a reflexive pronoun is missing”. No reflexive pronoun must be in the sentence, but since auxiliary or modal verbs do not require reflexives, it also must be guaranteed that at least one main (content) verb is in the sentence.

For convenience, we are using a rule-based approach to specify these triggering conditions. In [9] we have described a robust, rule-based semantic interpreter. Each interpretation rule has an identifier, a condition and an action part. The rule language in the original

\[ \text{In German, a preposition (e.g. 'in') and a definite, masculine or neuter determiner (e.g. 'dem') can be merged (e.g. 'in dem' -> 'in').} \]
interpreter resembles the TIGERSearch search language [11]. We have, however, adapted it to our present needs, namely to check for the presence (existence) and absence (negated existence) of tags and structural constellations. The action part is very simple, it consists of a list of error message codes applicable to the sentence in question.

In Fig. 2 we give an example of such a rule. The rule id is m-12. The condition part is a conjunction of propositions, where a proposition is either a node description or a test predicate operating on part of speech tags or on structural relations.

m-12#

```
X='PRPClass' & % prepositions
Y='ART' & % determiner
X<Y % immediate precedence
=>
error_messages([prpart-m]).
```

**Fig. 2: Rule m-12**

A node description is an assignment of a tree node (internally represented by a number) to a variable. A matching node must correspond to the conditions specified after the equal sign. For example, X='PRPClass' matches every tree node whose syntactic label is one of the various part of speech tags of prepositions. The Papa parser uses the STTS POS tags [14]. To ease the reference to POS classes, the interpreter allows for arbitrary class definitions. Fig. 3 gives the definition of the prepositions according to the inventory of the STTS tagset ('<<' represents the subclass relation, APPRART is a fusion of preposition and determiner, APPR is a "normal" initial preposition, APPC is a dislocated preposition).

```
APPRART << 'PRPClass'. %
APPR << 'PRPClass'. %
APPC << 'PRPClass'. %
```

**Fig. 3: Class Definition**

Often, only the existence or absence of a certain type of node has to be checked. In these cases, the instantiation of a node variable (e.g. X = 'PRPClass') is not necessary. We have defined three predicates: wd_exists (word exists), dep_exists (dependency label exists), pos_exists (part-of-speech tag exists). Besides the features of a single node, sometimes also structural relations (precedence, dominance) between two or more nodes have to be checked.

m-22# % subclause: irregular word order
dep_exists('NEE')
=>
error_messages([ws-fn]).

**Fig. 4: Rule m-13: word order**

Fig. 4 gives another example – the rule for irregular word order in a subclause. The triggering condition simply requests that a subordinate clause (dependency label NEE) be present.

Altogether, there are 37 rules, most of which are simple existence or non-existence checks.

After the set of error messages applicable to a sentence has been fixed, some of them, typically four, are selected for presentation. This is done according to the error distribution of the corpus. The probability of an error message is given by the maximum likelihood estimation of the underlying error. We have altogether 270 ill-formed sentences, with the frequencies as given in Fig. 1. The probability of
an error message e.g. v-fx (verb conjugation error) thus is 49/276. That is, in about 6 out of 10 sentence constellations this message will be generated - provided that it is permitted. A message is permitted, if it does not share the prefix with an already selected error message, e.g. v-fx and v-sfw do share a prefix, so only one is permitted.

5 Scenario II: Sentence Selection

There is another type of exercise that can be generated with our set of error message selection rules. Instead of providing a sentence and 4 error messages, we provide a single error message combined with 4 sentences. The learning task then is to find those sentences that beat the error described in the message.

A wrong preposition is being used.

1. Peter will in diesen Bus einsteigen.
3. In Amerika bauen sie moderne Bauten.
4. Aber sie wartet an ihren Freund.

Fig. 5: Sentence Selection Scenario

Fig. 5 gives an example. Here, a preposition is wrong: rather than “an ihren Freund” (“on her friend”), the correct PP is “auf ihren Freund”. The criterion for the selection of a sentence is simple: a preposition must be present. We have decided to present the correct(ed) versions of these sentences to the learner. So, only one of the four sentences is wrong, namely the one described by the error message. If the incorrect versions were chosen, the learner could remove sentence candidates more easily by simply checking for other errors being made in these sentences. However, we want the learner to focus on the single error that is described in the error message.

6 Evaluation

A (very) preliminary evaluation of our two scenarios was done by a non-native speaker of German and one of the authors of this paper. We evaluated the quality of the error messages being generated given a randomly generated sample of 100 sentences (Setting I). We looked for correctness and appropriateness. We found that only 3 erroneous (misleading) messages were generated - two due to a tagging error of the parser and one due to a faulty error message selection rule. This is a very low error rate. The few resulting non-appropriate error messages could hardly disturb the quality of this setting.

Setting II (Sentence Selection) was evaluated with respect to the quality of the generated sentences given an error message. About 70 such examples were given to a non-native speaker of German. She had to find the target sentence for each example and afterwards evaluate the appropriateness of the non-target sentences. Her feedback was mainly positive.

Of course, this is not a comprehensive evaluation. We plan to extend (or integrate) our prototype to (into) a collaborative writing environment. Given this, we hope to find some teachers that are willing to use and evaluate it with the help of their students (or pupils).

7 Related Work

To the best of our knowledge, [6] was the first who discussed “learning from errors” (“Lernen lernen”) as a CALL setting. We have adopted his idea, but have made it fully operational. Heringer’s original work relies exclusively on a manual encoding of exercises. Moreover, we designed a second exercise type (Sentence Selection) that is based on the same techniques and resources.

G. Faass [3] originally collected the Japanese second language learner sentence corpus that we are using in our work. In her master thesis she described a CALL system that is able to parse ill-formed sentences and to diagnose the assumed errors. Her system is based on the work done by Fortmann & Forst [4], a LFG-based grammar checker. Although we use our own error annotation, our system could easily be combined with the system of Faass and Fortmann/Forst (a mapping of error codes would suffice).

There are various approaches that are complementary to our system. [8] describes how an annotated learner corpus (such as ours) could be used to detect errors automatically.
On the other hand, if no training data is available, methods from machine translation could help (cf. [10]). Finally, [1] examines how an annotated learner corpus can be used to even correct faulty input.

8 Conclusions

We have introduced two CALL scenarios that are based on mature NLP techniques, namely tagging and parsing. Although these techniques still are not perfect, carefully designed learning scenarios producing reliable output seem to be feasible. Our system allows teachers to produce exercises on the fly. The only thing they have to do is to correct ill-formed sentences and classify the error being made—something they have to do anyway.

Finding and understanding one's own mistakes is a crucial skill in any problem-solving task including language learning. Our two settings are meant to support the student in such a learning process.

Acknowledgments. We would like to thank Christian Fortmann who made the second language learner sentence corpus available to us. Krista Linn helped us in evaluating the sentence selection scenario. Thank you, Krista.

References


Automatically detecting schematic structure components of English abstracts: building a high accuracy classifier for the task

Luiz Genoves Jr
University of São Paulo, NILC
CP 668,13560-970 São Carlos/SP Brazil
genoves@gmail.com

Valéria D. Feltrim
State University of Maringá, DIN
Av. Colombo, 5790, 87020-900
Maringá/PR Brazil
valeria.feltrim@din.uem.br

Carmen Dayrell
University of São Paulo, NILC
CP 668,13560-970 São Carlos/SP Brazil
c_dayrell@yahoo.com

Sandra Aluisio
University of São Paulo, NILC
CP 668,13560-970 São Carlos/SP Brazil
sandra@icmc.usp.br

Abstract
The writing of scientific papers poses a heavy burden on non-native English writers who have to master the conventions of scientific writing regarding the use of appropriate schematic structures and conventional expressions common to academic and scientific discourse. In order to minimize problems non-native English writers may face, we have built several types of writing tools. Moreover, we have developed a rubric to analyze abstracts of scientific papers in order to give feedback to the user when he/she writes this section of the paper. Both, the compilation of the writing tools databases and the implementation of the rubric, need a high accuracy system to annotate schematic structure components. The primary aim of this paper is to present AZEA (Argumentative Zoning for English Abstracts), a classifier which even based on a very small corpus (74 abstracts) achieved 80.4% accuracy (Kappa 0.73) in the task of annotating abstracts’ schematic structure components, which is far above the rates reached by other similar systems.

Keywords
Corpus linguistics, academic writing, educational technology, annotation of schematic structure components of abstracts, machine learning, writing support tools.

1. Introduction
The writing of scientific papers poses a heavy burden on non-native English writers at the lexical, syntactic and, most importantly, at the textual level. In addition to the various issues related to the natural complexity of the writing process, non-native English writers also have to deal with the conventions of scientific writing regarding the use of appropriate schematic structures and expressions which are typically employed in academic and scientific discourse. Schematic structure refers to the organization and structure of the text which is expected by a given discourse community [12], [15].

In an attempt to minimize problems that non-native English writers may face when writing academic papers in English, we have developed various types of writing tools [1], [2] and [8]. These tools use a database of authentic papers to capture idiosyncrasies of the discourse community and provide novice students with material related to the one he/she needs to write. They focus on experimental research papers, i.e., the writer puts forward one or more hypotheses and then presents the data which supports or disproves them. This is important to mention because different types of papers may adopt different schematic structure. For example, experimental research papers, theoretical papers, and revision papers use entirely different schematic structure.

The schematic structure of experimental research papers tends to be very similar in all research fields. This type of research paper tends to be divided into sections which are presented in the following order: Introduction, Review of Literature, Methodology, Results, Discussion and Conclusions. The abstract is usually a summary of these sections. However, the emphasis given to each component of the schematic structure may vary among different research fields. For example, while Biological Sciences abstracts focus on the components Purpose, Results and Conclusions [18], Computer Science abstracts focus on Purpose, Methodology and Results [7]. This is therefore the main reason why the abovementioned writing tools are domain dependent and have been designed to present the user the peculiarities of a specific domain.

Another difficulty is related to giving feedback once users have written a given section. In order to tackle this second problem, we are currently working on the implementation of a genre-based rubric to analyze abstracts of scientific papers [3], [11]. When fully automated, this rubric should enable a writing tool to detect errors and offer suggestions for improvements. It includes seven dimensions which deal with: (1) organization and development of the text, (2) balance...
among the components of the schematic structure of scientific genre, (3) coherence among components, (4) cohesive markers, (5) technical errors, (6) style and (7) presence of substantive material in certain components instead of indicative content. The dimensions have two scale values each — high and low — which helps both annotate dimensions and achieve high consistency among the human judges. Five of the seven rubric’s dimensions depend on the high accuracy of the rhetorical segmentation task i.e., the annotation of schematic structure components of the abstract.

The primary aim of this paper is to present AZEA (Argumentative Zoning for English Abstracts), a high accuracy system to automatically detect components of the schematic structure of English abstracts. We have therefore developed a Sequential Minimal Optimization (SMO) classifier which automatically annotates each sentence of a given abstract according to 6 components proposed by Feltrim et al. [7], namely: Background, Gap, Purpose, Methodology, Results, and Conclusion. Despite using a very small corpus (74 abstracts), our classifier achieved 80.4% accuracy (Kappa 0.73), which is far above the rates reached by other similar systems (see Section 2).

2. Related Work

To the best of our knowledge, two methods have been proposed in the literature to automatically detect the components of the schematic structure of scientific texts, namely, Argumentative Zoner [13], [14] and Mover [4].

The Argumentative Zoner (AZ) was initially designed to identify the authorship of each sentence of a scientific text on the basis of three rhetoric functions: general background information, contribution of scholars other than the author and contribution of the author him/herself. The system was later expanded so as to automatically summarise scientific texts [14]. In this case, the summaries are made up of extracts containing the contributions of the author as well as their link to previous research in the area. This is done by associating each sentence of the text to one of the seven rhetoric categories shown in Table 1.

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aim</td>
<td>Statements of author’s scientific aim</td>
</tr>
<tr>
<td>Contrast</td>
<td>Statements of difference from other work</td>
</tr>
<tr>
<td>Basis</td>
<td>Statements of sources of ideas</td>
</tr>
<tr>
<td>Other</td>
<td>Neutral description of other work</td>
</tr>
<tr>
<td>Background</td>
<td>Generally accepted statements in the field</td>
</tr>
<tr>
<td>Textual</td>
<td>Statements about external structure of article</td>
</tr>
<tr>
<td>Own</td>
<td>All other statements about own work</td>
</tr>
</tbody>
</table>

The AZ system uses 16 features for extracting lexical, syntactical and structural knowledge from texts. These features are then used to train a Naïve Bayes classifier. The error rate was estimated by means of a 10-fold stratified cross-validation on a corpus of 80 conference articles in computational linguistics. The resulting values were compared to four baselines (Table 2).

| Table 2. Evaluation of the AZ system [14] |
|-------------------|-------------------|-------------------|
| Category   | Kappa | Accuracy | Macro-F |
| AZ         | 0.45  | 0.73    | 0.50    |
| Baseline I | 0.30  | 0.72    | 0.30    |
| Baseline II| -0.10 | 0.14    | 0.09    |
| Baseline III| 0   | 0.48    | 0.14    |
| Baseline IV| -0.13| 0.67    | 0.11    |

These figures refer to the comparison between the categorization of the AZ system and the baselines with one human annotator. Baseline I is a text categorization experiment using a Rainbow implementation of a Naïve Bayes TF*IDF method [10]; each sentence is treated as a “document”. Baseline II is a random choice of categories. Baseline III is a random choice of categories weighted by their distribution in the corpus; and Baseline IV always assigns the most frequent category.

For this specific task, no reference corpus is used due to the fact that agreement among humans is low. In order to overcome this issue, we have resorted to the Kappa statistics which indicates whether the machine annotation can be said to be as good as a given human annotation. Although the Kappa value between the AZ and the human annotator is low (0.45), these are encouraging results since they show a significant improvement when in relation to the baselines.

In line with Teufel and Moens [14], Feltrim et al. [9] proposed and implemented a system to automatically detect components of the schematic structure of scientific abstracts in Portuguese. Feltrim’s system is based on the method of Argumentative Zoning and it was named AZPort. Unlike the AZ, the AZPort was designed to be part of an academic writing supporting system – SciPo (Scientific Portuguese). One of Scipo’s functionalities is the automatic detection and criticism of thesis abstracts structures. For the classification task, Feltrim et al. [9] also uses the Naïve Bayes algorithm to label each sentence of an abstract according to the following seven rhetoric categories: background, gap, purpose, methodology, results, conclusion and outline, which are illustrated in Figure 1. These categories were identified on the basis of a set of eight features. These features derived from the original AZ feature set and indicate the size and position of the sentence in the abstract, presence of citations, presence of formulaic expressions (phrases which regularly appear in the categories mentioned above), verb tense and voice of the first finite verb in the sentence, if this verb is a modal one or not, and history (the category of the previous sentence) [6], [8].

AZPort was trained by applying a 13-fold cross-validation test to 52 Portuguese abstracts written by computer science students (training sets of 48 texts and testing sets of 4 texts) and the resulting values were compared to two different baselines.
1 Background
“The research article (RA) or paper is one of the most important genre that both scientists and engineers will write.”

2 Gap
“When faced with the tasks of reading and writing a complex technical paper, many non-native scientists and engineers (…) lack an adequate knowledge of commonly used structural patterns at the discourse level.”

3 Purpose
“In this paper, we propose a novel computer software tool that can assist these people in the understanding and construction of scientific papers (…)”

4 Methodology
“The software uses a supervised learning approach, in which the system first “learns” the characteristic features of text structure in a particular discipline using a small number of training examples.”

5 Results
“We can see that the system performs consistently across the different data sets, with an average accuracy of 68%.”

6 Conclusions
“The system is tested using research article abstracts and is shown to be fast, accurate, and useful aid in the reading and writing process.”

7 Outline
“In the next section we present the context of this work and details about the used methodology.”

Figure 1. Example sentence for each category with lexical patterns underlined [9]

The baselines are similar to Teufel’s Baseline III and IV. Table 3 summarises the results. The Kappa statistics value was 0.65 and the system achieved 72% accuracy, which outperforms both baselines.

Table 3. Evaluation of the AZPort system [9]

<table>
<thead>
<tr>
<th></th>
<th>Kappa</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>AZPort</td>
<td>0.65</td>
<td>0.72</td>
</tr>
<tr>
<td>Baseline III</td>
<td>0</td>
<td>0.20</td>
</tr>
<tr>
<td>Baseline IV</td>
<td>0.26</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Another method applied to automatically detect the components of the schematic structure of scientific texts was proposed by Anthony and Lashkia [4]. The method was implemented in the Mover system and was designed to assist learners, non-native speakers in particular, to understand the general structure of scientific texts. Unlike AZ and AZPort systems, which focus on scientific texts in a specific language (English and Portuguese, respectively), Mover is language independent since it does not use linguistic knowledge or pre-built resources. The systems’ primary aim is to associate each sentence with a component of the schematic structure of the corpus in use.

Mover also uses a Naïve Bayes classifier and its set of categorization features are modeled as a bag of clusters. It adapts the bag of words model so as to use phrases containing up to five words, rather than using single words. It also resorts to a statistical calculation – Information Gain (IG) [17] – to measure the relevance of each cluster to the classification task. Irrelevant clusters are therefore discarded. In addition to these features, Mover also takes in account the position of the sentence within the text and makes optimizations based on the structural flow of the abstract. The system was trained on 554 sentences and tested on 138 sentences. All sentences were extracted from a 100-abstract corpus collected from published papers on the field of Information Technology. Five sets of texts randomly selected were used for training and testing, with 68% average accuracy. This is higher than both baselines: classification by chance (16.6%) and assignment of the most frequent category (26%).

3. AZEA
AZEA (Argumentative Zoning for English Abstracts) is a corpus-based machine learning system which is based on the approaches used by the AZ and AZPort systems. AZEA’s primary aim is to automatically identify components of the schematic structure of scientific abstracts in English. Given that AZEA was designed to be part of an academic writing critique and evaluation system, we adopted six out of the seven categories proposed by Feltrim et al. [9]: Background, Gap, Purpose, Methodology, Results, and Conclusion.

3.1 Features
The AZEA system uses a set of 22 features (summarized in Table 4).

| Length, Location, Tense, Voice and Modal are similar to the features described by Feltrim et al. [9]. Length classifies a sentence as short, medium or long length, based on two thresholds that were estimated using the average sentence length present in our corpus. The Location feature identifies the position occupied by a sentence within the abstract. Paragraph breaks are ignored. We use four values for this feature: first, medium, penult and last. Tense, Voice and Modal report syntactic properties of the first finite verb in the sentence.

The Formulaic and Agent features follow Teufel’s [13] definitions. The History feature is similar to Teufel’s and Feltrim’s, but simpler, since no beam search is performed in order to estimate the previous classification. AbsFormulaic and Boolean patterns are 14 new features, which are based on formulaic patterns. These features were proposed with the view to helping the classifier to distinguish the categories Results and Methodology, since these categories are likely to be confused.

AbsFormulaic is similar to Formulaic with the difference that it detects the presence of formulaic expressions which are abstract-specific. Boolean patterns are 13 features whose value can be either 1 or 0, marking the presence of 13 specific formulaic expressions. The main difference between a Boolean Pattern feature and the Formulaic feature is that the latter detects the presence of only one type of formulaic expression (the first type detected), irrespective of whether there are other formulaic patterns in the sentence.
Two other corpora were also used in our experiments for additional testing, namely: ‘atypical’, a corpus of atypical abstracts and ‘student’, a corpus of abstracts written by students. Atypical abstracts are texts which do not follow the criteria suggested by Swales [12] and Weissberg and Buker [15] to describe the conventional organization and structure of abstracts. This corpus consists of 18 abstracts from selected publications within the field of Pharmacology, totaling 121 sentences. The corpus of students’ abstracts comprises 18 abstracts (143 sentences) written by non-native speakers of English as part of an academic writing course.

### Table 5. Performance comparison of the four machine learning algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy (Acc)</th>
<th>Kappa (K)</th>
<th>Macro-F</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMO</td>
<td>80.39%</td>
<td>0.73</td>
<td>0.78</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>77.97%</td>
<td>0.69</td>
<td>0.70</td>
</tr>
<tr>
<td>J48</td>
<td>77.17%</td>
<td>0.69</td>
<td>0.74</td>
</tr>
<tr>
<td>JRip</td>
<td>77.81%</td>
<td>0.69</td>
<td>0.74</td>
</tr>
<tr>
<td>Baseline</td>
<td>44.86%</td>
<td>0.00</td>
<td>0.10</td>
</tr>
</tbody>
</table>

As shown in Table 5, all classifiers outperform the baseline and the SMO outperforms the other three algorithms in all the measures employed (Acc=80.39%; K=0.73; Macro-F=0.78). These are very good results since it is higher than the performance reported to the AZ, AZPort and Mover systems. However, when comparing our results to the AZ’s, it is important to bear in mind that our system classifies sentences of abstracts while the AZ classifies sentences of the whole paper.

Thus, taking into consideration the results presented in Table 5, we decided to include the SMO classifier as part of the AZEA system. Table 6 shows the performance of the SMO classifier for each category in terms of precision, recall and F-measure. According to the F-measure values, the classifier performance is uniform for most categories, except for Gap and Methodology. The low performance for the Gap category may be justified by

### Table 6. Performance comparison of the SMO classifier per category

<table>
<thead>
<tr>
<th>Feature</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Location</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>History</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Tense</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Voice</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Modal</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Formulaic</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Agent</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>AbsFormulaic</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Boolean</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Length</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Location</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>History</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Tense</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Voice</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Modal</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Formulaic</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Agent</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>AbsFormulaic</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>Boolean</td>
<td>0.80</td>
<td>0.80</td>
<td>0.80</td>
</tr>
</tbody>
</table>
lack of sufficient training material (only 13 sentences). As for Methodology, we found that the classifier had much difficulty in distinguishing between Methodology and Results sentences. This justifies the low performance for the former. In fact, distinguishing the pair Methodology-Results may be hard work even for human annotators since it is common to find these categories bound by a single sentence. If we treat these two categories as one, the general performance of the classifier rises to 90% (K=0.83). Similar results regarding the categories Methodology and Results were reported by Feltrim et al. [9].

Table 6. Precision, Recall and F-measure per category using the SMO classifier

<table>
<thead>
<tr>
<th>Category</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>77.8%</td>
<td>77.8%</td>
<td>77.8%</td>
</tr>
<tr>
<td>Purpose</td>
<td>82.1%</td>
<td>82.1%</td>
<td>82.1%</td>
</tr>
<tr>
<td>Methodology</td>
<td>69.4%</td>
<td>76.8%</td>
<td>72.9%</td>
</tr>
<tr>
<td>Results</td>
<td>85.6%</td>
<td>83.2%</td>
<td>84.4%</td>
</tr>
<tr>
<td>Conclusion</td>
<td>82.1%</td>
<td>80.9%</td>
<td>81.5%</td>
</tr>
</tbody>
</table>

As a second experiment, we evaluated SMO using a 10-fold stratified cross-validation on our corpus of 74 typical abstracts. Here, we used different subsets of features, as shown in Table 7.

Table 7. Potential of individual and group of features in terms of Kappa

<table>
<thead>
<tr>
<th>Feature(s)</th>
<th>Feature(s) alone</th>
</tr>
</thead>
<tbody>
<tr>
<td>History</td>
<td>0.64</td>
</tr>
<tr>
<td>Length</td>
<td>0.72</td>
</tr>
<tr>
<td>Formulaic</td>
<td>0.71</td>
</tr>
<tr>
<td>Agent</td>
<td>0.74</td>
</tr>
<tr>
<td>Modal</td>
<td>0.72</td>
</tr>
<tr>
<td>Tense</td>
<td>0.71</td>
</tr>
<tr>
<td>Voice</td>
<td>0.73</td>
</tr>
<tr>
<td>AbsFormulaic</td>
<td>0.71</td>
</tr>
<tr>
<td>Boolean Patterns</td>
<td>0.68</td>
</tr>
<tr>
<td>Contextual</td>
<td>0.44</td>
</tr>
<tr>
<td>Syntactic</td>
<td>0.71</td>
</tr>
<tr>
<td>Textual</td>
<td>0.66</td>
</tr>
</tbody>
</table>

The first column lists our pool of features. The last three lines – Contextual, Syntactic and Textual – represent sets of features (see Table 4). The second column gives the predictiveness of the feature (or group) on its own, in terms of Kappa between the classifier and one annotator. The third column gives Kappa coefficients for experiments using all features except the one shown in the first column. The same applies to the three sets of features. The figures in Table 7 show that the contextual features – History and Position – are the strongest. Syntactic features – Tense, Voice and Modal - and Length are the weakest ones. Much to our surprise, textual features, especially Formulaic, proved far less predictive than others. This contradicts previous claims in the literature ([14]; [4]; [9]) that this kind of feature is very predictive. In fact, some systems (i.e., Mover) have the extraction of formulaic expressions as their core feature. We ascribe the weak performance of the Formulaic feature to the difference in domain between the corpus used for extracting the formulaic patterns and the one used for training the classifier. Our list of formulaic patterns was implemented following Teufel [13], who compiled these patterns based on a corpus of scientific papers in the field of Computational Linguistics. Our training corpus consists of papers within the discipline of Pharmacology. Although Teufel’s list of formulaic patterns captures formulaic expressions that are related to scientific discourse in general, it is widely acknowledged that there are idiosyncrasies among different scientific fields. In order to capture these idiosyncrasies, it would be necessary to expand Teufel’s original formulaic patterns list and include new expressions extracted from a corpus of texts belonging to the same scientific area as the ones in the training corpus.

Our experiments also revealed the History feature is highly predictive. This does not come as a surprise since contextual features have already been said to have good predictiveness. However, our results for History are far above the ones presented by related studies. The high performance of the contextual features may be explained by the nature of our corpus. As mentioned in Section 3.2, the abstracts in our corpus were regarded as typical since they comply with well known models of abstract’s schematic structures. Not surprisingly, these abstracts tend to present similar organization and hence contribute to the high performance of features based on contextual information.

These experiments have shown that the classifier works for well written/organized abstracts. However, it is important to stress that our aim is to develop a system which performs well on both published abstracts and abstracts written by students. This means that we would expect AZEA to have a reasonable performance when classifying abstracts with language and organizational problems. This flexibility is necessary since AZEA was designed to be part of a rubric which analyzes abstracts of scientific papers.

In order to evaluate the SMO classifier performance on non-revised abstracts (i.e., written by students) or on abstracts with an atypical organization, we ran a third experiment using our ‘Atypical’ and ‘Student’ corpora. We tested the SMO classifier on ‘Atypical’ corpus, ‘Student’ corpus and both corpora altogether. The results are presented in Table 8 in terms of raw accuracy, Kappa and Macro-F.

As expected, the classifier performance drops when applied to corpora with organization deviations since its strongest features are based on sentence position and history. However, the results are still above the ones reported by related studies, especially when AZEA is tested on the ‘Student’ corpus (Acc=74.83%). These are very encouraging results and give us confidence to use
AZEA as part of the automatic application of the genre-based rubric, as well as for other tasks that might take advantage of this kind of categorization. In fact, Dimensions 1 and 2 of this genre-based rubric have already been implemented. An interface to access Dimensions 1 and 2 is available at: http://www.nilc.icmc.usp.br/azea-web/.

Table 8. Performance of the SMO classifier when tested with ‘Atypical’ and ‘Student’ corpora

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Measure</th>
<th>Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atypical</td>
<td>Accuracy</td>
<td>61.98%</td>
</tr>
<tr>
<td></td>
<td>Kappa</td>
<td>0.48</td>
</tr>
<tr>
<td></td>
<td>Macro-F</td>
<td>0.49</td>
</tr>
<tr>
<td>Student</td>
<td>Accuracy</td>
<td>74.83%</td>
</tr>
<tr>
<td></td>
<td>Kappa</td>
<td>0.67</td>
</tr>
<tr>
<td></td>
<td>Macro-F</td>
<td>0.70</td>
</tr>
<tr>
<td>Atypical+</td>
<td>Accuracy</td>
<td>68.94%</td>
</tr>
<tr>
<td>Student+</td>
<td>Kappa</td>
<td>0.60</td>
</tr>
<tr>
<td></td>
<td>Macro-F</td>
<td>0.64</td>
</tr>
</tbody>
</table>

3.4 Purpose Splitter

As an attempt to improve the classifier performance, we proposed a new approach to the argumentative zoning method, called Purpose Splitter. Instead of using a single classifier, this approach divides the classification task into two. We first locate purpose sentences using a classifier trained for this specific purpose. The location of the purpose sentence is then used as a new feature to classify the remaining sentences of the abstract. This new feature is called Purpose position and may assume three values: ‘purpose’, ‘before_purpose’ and ‘after_purpose’.

Purpose Splitter assumes that purpose sentences can be profitably used to divide the sentences of other categories into two distinctive groups: before and after the purpose sentence. Taking into consideration a typical abstract schematic structure, it is expected that background and gap sentences appear before the purpose sentence (‘before_purpose’ group). By contrast, methodology, results and conclusion sentences are expected to appear after the purpose sentence (‘after_purpose’ group).

In manual annotation, the group division proved to be very helpful in the identification of categories. However, in order to perform it automatically, we need a very accurate classifier for purpose sentences. Once again, we evaluated the four machine learning algorithms listed in the previous section using a 10-fold stratified cross-validation on our corpus of 74 typical abstracts. However, for this experiment, the sentences in the corpus were labelled with two categories only: Purpose and Non-purpose. The general performance of the algorithms is shown in Table 9. The precision of each algorithm in the identification of Purpose is shown in Table 10.

The four algorithms outperformed the baseline (assignment of the most frequent category) and the Naïve Bayes presented the best results. Although the accuracy of the classifiers is all above 90%, it is not very significant since the corpus has a skewed distribution. An analysis of the Kappa and Macro-F values shows that even the Naïve Bayes performance is not sufficient for our purposes. This is confirmed by the precision values presented in Table 10.

However, we tested Purpose Splitter using the Naïve Bayes classifier for purpose location and the SMO classifier trained with our previous pool of features plus the feature Purpose position. As expected, there were no improvements in the final classification results.

Table 9. Performance comparison of four machine learning algorithms for purpose sentence identification

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy(Acc)</th>
<th>Kappa(K)</th>
<th>Macro-F</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMO</td>
<td>92.9%</td>
<td>0.66</td>
<td>0.83</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>91.5%</td>
<td>0.58</td>
<td>0.79</td>
</tr>
<tr>
<td>J48</td>
<td>92.4%</td>
<td>0.61</td>
<td>0.83</td>
</tr>
<tr>
<td>JRip</td>
<td>87.5%</td>
<td>0.62</td>
<td>0.79</td>
</tr>
</tbody>
</table>

We performed a final experiment on the Purpose Splitter approach in which we assumed that the results of purpose location task were always optimal, simulating a classifier that made no mistakes. Our intention was to examine whether the group division strategy would improve the AZEA’s performance. Once again, we evaluated the SMO classifier using a cross-validation on the 74 abstracts but including the simulated Purpose position feature. We found that by simulating the perfect Purpose classifier the final classification results improve by 6% (Acc= 84.73%, K= 0.79 and Macro-F=0.84).

Although we found Purpose Splitter to be useful in theory, it presented no practical results. Even simulating the perfect Purpose classifier, the results were bellow expected and misclassification problems within groups of categories remain. As previously mentioned, classifiers have problems distinguishing the Methodology and Results categories. Both categories tend to occur after the purpose sentence and hence the Purpose Splitter does not help their identification.

4. Concluding Remarks

In this paper, we presented AZEA, a system to automatically detect components of the schematic structure of English abstracts. Despite using a very small corpus in the training, AZEA achieved 80.4% accuracy (Kappa 0.73), which is far above the rates reached by related works. It will be used to annotate new corpora aimed to build customized writing tools for several
research areas. When applied to corpora with organization deviations, the classifier performance drops since its strongest features are based on sentence position and history. However, it keeps its results above the ones reported by related works (Acc=74.83%). AZEA is also part of the automatic application of a genre-based rubric which is intended to give feedback to novice students. Since this rubric has not yet been fully automated, no real user evaluation was performed. In future work, we intend to adapt AZEA features to build classifiers to all the sections of a scientific paper. These classifiers will be used to spread our approach to help non-native English students to several research areas and to other students than Brazilian Portuguese.

5. Acknowledgements

We would like to thank CNPq for the financial support as well as the annotators for their invaluable work.

6. References


Visualizing the Invisible: A Method for Visualizing the Countability of English Nouns

Ryo Nagata¹, Atsuo Kawai², Koichiro Morihi², and Nacki Isu²
¹Hyogo University of Teacher Education
942-1 Shimokune, Kato, Hyogo 673-1494
{rnagata.mori}@hyogo-u.ac.jp
²Mie University
1577 Kurinamachiya, Tsu, Mie 514-8507
{kawai,isu}@ai.info.mie-u.ac.jp

Abstract
This paper describes a method for visualizing the countability of nouns in context, which is normally invisible. The ability to see the countability property has various advantages in second language learning and teaching. The method first estimates the probability that the target noun is recognized to be countable from its context. It then visualizes the countability with its context based on the probability. Experiments show that the visualized countability agrees with our intuitive interpretation of countability and that example sentences for countable and uncountable usages can be efficiently improved using the method.

Keywords
Countability, Noun, Visualization, Article usage, Language learning

1 Introduction
This paper describes a method for visualizing the countability of nouns in context, which is normally invisible. It first estimates the probability that the target noun is recognized to be countable from automatically generated training data. It then visualizes countability by plotting the fluctuation of the probability as a line graph. In doing so, it reveals the contribution of each word in the context of the target noun toward determining its countability. It also visualizes how the countability of the target noun fluctuates with its context.

Visualization of the countability of English nouns has a number of benefits in second language learning and teaching. Even advanced learners of English have problems determining countability in a given context, and frequently make errors involving countability and article usage [7, 9]. Thus, it learners can efficiently learn the countability system in English, they will be able to avoid making such errors. For example, if they understand how the countability of the noun paper varies depending its surrounding context, they can avoid making errors as in “I read scientific paper.” In addition, a visualization capability can allow teachers of English to devise optimal example sentence patterns for teaching countability usages for learners of English. The capability is of particular benefit to teachers whose native language does not have a countability system similar to that of English. Similarly, the visualization is useful for non-native speakers to select appropriate determiners in English, since countability determines the range of possible determiners.

Most work in countability [2, 3, 5, 10, 11, 15, 16] has been devoted to classifying nouns according to their countability classes or countability preferences [1]. For instance, Bond and Vatikiotis-Bateson [5] used an ontology to determine a noun’s countability class from its semantic class. In their model, nouns are classified into five countability preferences: fully countable, strongly countable, weakly countable, uncountable, and plural only. Other models [2, 3, 10, 15, 16] learn countability from corpus data. As an alternative approach, Nagata et al. [13] have proposed a method for predicting whether a noun in context is countable or uncountable from its surrounding words.

Unfortunately, however, countability preferences are not appropriate for visualizing countability of nouns in context, since most nouns can be used as both countable and uncountable depending on their meaning or context [4, 8]. Nor does the prediction method [13] facilitate second language learning and teaching, since it would only indicate that the noun in question is countable or uncountable. Moreover, it does not help the second language teacher make useful example sentences. For these purposes, it is preferable that the user should see what it is that makes the target noun countable or uncountable in context.

Unlike the other approaches, the method described in this paper makes it possible for the language learner to understand the contribution of each word in the context to the countability of the target noun. Also, in the proposed scheme, words that contribute strongly to the countability of the target noun can be extracted from corpus data. As a result, teachers of English as a second language can efficiently build example sentences covering countability usage in English nouns.

The following section introduces the basic idea of the proposed method. Section 3 describes how to visualize the countability of nouns in context. Section 4 discusses how to extract words which strongly indi-
cate the countability of the target noun in context. Section 5 describes experiments conducted to evaluate the method and discusses the results.

2 Basic Concept

Although, in English, nouns heading noun phrases (NPs) are typically either countable or uncountable [5], there are some instances of a noun which appear to be more countable than others. For instance, it is not clear whether the noun *paper* in the sentence:

Where is the paper?

is countable or uncountable, while in the sentence:

Where is the important paper?

it appears to be more countable, and in turn in the sentence:

Where is the important paper she published?

it appears to be more countable than the other two.

This phenomenon can be modeled by the probability that the target noun is recognized to be countable in a given context. Returning to the previous three examples, let us set a value of 0.5 for the first instance, indicating a probability that it will be recognized to be countable, and thus that the probability of the other two instances show a higher value approaching 1.

The basic idea of the method is that the countability of the target noun in context is represented by the probability. The probability can be estimated from instances labeled with their countability, i.e., training data. After estimating the probability, the countability of the target noun is visualized as a graph of the probability fluctuating with each word in the context. The next section describes the estimation algorithm and visualization scheme.

3 Visualizing countability

3.1 Estimating the Probability

As already mentioned, the countability of the target noun in context is modeled by a probability in the proposed method. To formalize the probability, we use the symbol c and u to represent the values countable and uncountable, respectively. Plural only nouns are excluded here, because they are almost always countable (i.e., plural), and thus there is no need for visualization. We use the symbol x to denote a context of the target noun. Then, the probability that the target noun is recognized to be countable given a context is formalized by

\[
F(c|x) = \frac{F(x|c)F(c)}{F(x)} 
\]  

(1)

\(F(c)\) is the prior probability of c, the probability that the target noun is recognized to be countable without any contextual information. \(F(c)\) is updated with \(F(x|c)/F(x)\) that incorporates the evidence that we have about the context.

Using the Naive Bayes assumption [12], Eq.(1) is approximated as

\[
F(c|x) \approx \frac{\prod_{w_{i} \in z}P(w_{i}|c)\prod_{z \in \{c,u\}}P(z)P(w_{i} \mid z)}{\sum_{z \in \{c,u\}}P(z)\prod_{w_{i} \in z}P(w_{i}|z)}F(c) 
\]  

(2)

where \(w_{i}\) is a word in the context \(x\). The probabilities in the right hand side can be estimated from the training data. \(P(c)\) is estimated by

\[
P(c) = \frac{o(c)}{N}
\]  

(3)

where \(o(c)\) and \(N\) represent the number of occurrences of countable instances and the total number of occurrences of the target noun in the training data, respectively. \(P(w|c)\) is estimated by

\[
P(w_{i}|c) = \frac{o(w_{i},c) + \alpha}{\sum_{c}o(w_{i},c) + \alpha}
\]  

(4)

where \(o(w_{i},c)\) and \(o\) are the number of occurrences of \(w_{i}\) in the context of countable instances in the training data, and \(\alpha\) is a smoothing parameter, respectively. Replacing \(c\) with \(u\) in Eq.(3) and Eq.(4) gives equations for estimating \(P(u)\) and \(P(u|w)\), respectively.

3.2 Obtaining Training Data

To estimate the probabilities in Eq.(2) requires training data where instances of the target noun are labeled with their countability. Fortunately, Nagata et al. [14] have proposed a method for tagging nouns in context with their countability. This paper follows it to obtain the training data.

To obtain the training data, first, instances of the target noun used as a head noun are collected from a corpus with their surrounding words. This can be simply done by an existing chunker or parser. Second, the collected instances are tagged with either countable or uncountable by the following tagging rules. For example, the target noun *paper*:

She read new *papers* in her room.

is tagged as

She read new *papers/uncountable* in her room.

because it is plural.

Figure 1 and Table 1 represent the tagging rules based on the method provided by Nagata et al. [14]. Figure 1 shows the framework of the tagging rules. Each node in Fig.1 represents a question applied to the instance in question. For example, the root node reads "Is the instance in question plural?" Each leaf represents a result of the classification. For instance, if the answer is "yes" at the root node, the instance in question is tagged with *countable*. Otherwise, the question at the lower node is applied and so on. The tagging rules do not classify instances in some cases. These unclassified instances are tagged with the symbol "?". Unfortunately, they cannot readily be included in training data. For simplicity of implementation, they are excluded from the training data.
Third, words are extracted from the context\(^1\) of each instance with the following three types of contextual information: (i) within the NP that the instance heads, (ii) within five words to the left of the NP, and (iii) within five words to its right. Here, function words (except prepositions and words in Table 1 (a) and (b)) are excluded as stop words. Also, the target noun itself is excluded. All words are reduced to their morphological stem and converted entirely to lower case when extracted.

Finally, the extracted words are stored in a file with their corresponding countability label as training data. For example, the sentence:

She read new \textit{papers/countable} in her room.

would give a piece of training data:

-5=\textit{read}, NF=\textit{new}, +5=\textit{in}, +5=\textit{room}, \textbf{LABEL=c}

where "\textbf{LABEL=c}" denotes that its countability label is countable.

### 3.3 Visualizing countability

The proposed method visualizes the countability of the target noun in context by plotting the fluctuation of the probability as a line graph. The horizontal and vertical axes correspond to words in the context of the target noun and the countability probability, respectively. The coordinates are given by \( F(c|x = w_1w_2\cdots w_i) \) where \( w_i \) is the \( i \)-th word in the context of target noun. The labels C, C5, and 1 on the vertical axis are respectively replaced with Uncountable, Neutral, and Countable as shown in Fig. 2 so that the visualization gives a better understanding. Likewise, the label \( i \) for the horizontal axis is replaced with its corresponding word \( w_i \) for the same purpose. The label on the horizontal axis is appended with the dummy start symbol \( \phi \) in order to plot the prior probability.

Figure 2 shows examples of the visualization\(^2\) where the target noun is \textit{paper} in the sentences \textit{She read the new paper in her room} and \textit{She has the paper with her}. The words with the symbol \# (She\#, the\#, etc.) are stop words that are excluded from the training data as explained in Subsect. 3.2.

The left graph shows that the target noun \textit{paper} is recognized to be almost neutral in terms of countability without any context (i.e., \( \phi \)). The first word \textit{She} is excluded from the estimate because it is one of the stop words. The graph then reveals that the next word \textit{read} strongly contributes to the target noun’s countability, which agrees with our intuition. Thereafter, as the sentence proceeds to the end, the countability of the target noun steadily increases and approaches almost fully countable at the end of the sentence.

By contrast, the right graph has a rather flat line, comprising no particular words that contribute to the target noun’s countability. This value corresponds to the fact that the countability of the \textit{paper} is unclear in the context. Thus, the sentence \textit{She has the paper with her} is a poor example sentence for teaching of English noun countability.

As we have just seen, the method visualizes the countability of the target noun in context. The visualization clearly shows how the countability fluctuates with the context of the target noun. It also shows the extent to which each word in the context contributes to the countability.

\(^1\) Words are extracted only from within the sentence in which the instance appears.

\(^2\) The British National corpus\(^6\) was used to estimate the probabilities. Section 5 describes the details.
to the countability of the target noun.

When the visualization is targeted at all nouns in a given text, it becomes difficult to visualize the countability of each noun by using the graph described above. In that case, countability can be visualized by coloring each target noun in a given text; e.g., the redder/bluer the target noun is, the more/less countable it is. Similarly, the degree of contribution of each context word toward determining countability can be visualized by taking the same approach.

4 Extracting indicators of Countability

If we can extract words strongly indicating that the target noun is either countable or uncountable, a poor example such as She has the paper with her, may be improved. The strength of the countability measure can be changed by adding some of the extracted words; if the value approaches either 0 or 1, then we can say that the revised example sentence has improved as an exemplar of countability.

In our framework, good indicators of the countability of the target noun are words \( w \) that have a high probability of being countable \( (P(w|c)) \) and a low probability of being uncountable \( (P(w|u)) \).

Based on this, we define goodness of \( w \) by

\[
g(w) = \log \frac{P(w|c)}{P(w|u)}. \quad (5)
\]

Words \( w \) that give large positive and negative numbers of \( g(w) \) are good indicators for the target noun being countable and uncountable, respectively. Sorting \( w \) by \( g(w) \) in descending order gives a list of which \( w \) is in order of its goodness as an indicator for the target noun being countable. Likewise, sorting them in ascending order gives a list of which \( w \) is in order of its goodness as an indicator for the target noun being uncountable.

One might presume that determiners [1] such as a and each occupy the tops of any list of indicators for countability. However, they are of little use in second language learning, since they almost always make the target noun countable and provide no other information about the extent to which the target noun is interpreted as countable. Also, since second language learners are taught the countability system of English mainly to be able to select appropriate determiners including a and each, determiners are apparently of no use for the purpose. Considering these facts, we exclude determiners from the sorted lists.

Table 2 shows some good indicators for the target noun paper being countable or uncountable. Words in the left and right column are indicators for the target noun being countable and uncountable, respectively. They are appended with their contextual information (e.g., "5=scientific" denotes within five words to the left of the NP that the target noun heads).

Using the indicators in Table 2, we can efficiently make a poor example sentence better. For instance, we can make the poor example sentence She has the paper with her better by adding the indicator "NP=scientific" to it. Figure 3 shows the visualization of the countability corresponding to the resulting sentence.

5 Experiments

5.1 Experimental Conditions

In the experiments, we conducted the following two tasks. In the first, we predicted the countability of a set of target nouns in context using the probability calculated by Eq. (2) to evaluate how accurately the proposed method estimated the probability. In the second, we revised some example sentences of countable and uncountable usages using the visualization and indicators extracted by the method described in Sect. 4. Then, a native speaker of English who was a
of the list, each indicator was examined for semantic felicity in the example sentence; if semantically appropriate, the indicator was added to the example sentence (or one of the words in the example sentence was replaced by it if appropriate). The revision was repeated until $P(c|x) > 0.90$ for the countable-target sentences or $P(c|x) < 0.10$ for the uncountable-target sentences is satisfied. Each revised example sentence was shown to the rater, who was asked to choose the better one as an example of countable/uncountable usage of the target noun from each pair. This evaluation resulted in a determination of the number of revised example sentences judged to be better than their original.

5.3 Experimental Results and Discussion

In the first task where countability was predicted, the proposed method succeeded in 49 target sentences out of 50 achieving an accuracy of 0.98; the Nagata et al. [13]'s method for comparison achieved an accuracy of 0.68. However, it turned out that the difference in accuracy was not statistically significant (binomial test, significance level: 0.05). Besides, we should note that the Nagata et al. [13]'s method does not use information on articles unlike the proposed method. What is important here is that despite the Naive Bayes assumption, the proposed method predicts countability as well as or even better than the Nagata et al. [13]'s method. This suggests that the proposed method estimates the probability that the target noun is recognized to be countable fairly well. In most target sentences, the visualization agreed with the intuition of the first author. Words that are recognized as good indicators of countability actually contributed to the target noun being countable or uncountable according to the visualization; some examples are shown in Fig. 4. In Fig. 4 (a), the line rapidly drops toward uncountable at the word strong and sense from which we can tell that the target noun is uncountable. Figure 4 (b) reveals that the target noun hair is normally uncountable but undergoes conversion to countable when it refers to a strip of hair. In Fig. 4 (c), although the word Another strongly contributes to the target noun sleep being countable, the word good are also informative for its countability.

It failed to predict countability correctly in some cases. One major reason was that in some nouns such as gold the prior probability $P(c)$ was so low or high that the surrounding words had little effects on $P(c|x)$ even if they were informative for countability. Figure 5 shows an example of the case. In Fig. 5, $P(c)$ is so low that $P(c|x)$ remains low although the words won and Olympics are informative for the target noun being countable. For such nouns as gold, a model without $P(c)$, or implicitly assuming $P(c) = 0.5$, might work well. Fortunately, we can know in advance which noun has very high or low $P(c)$ at the time we estimate it.

In the second task, it appeared from the empirical result that the revision had almost no effect on improving

---

5.2 Experimental Procedures

First, for the 25 target nouns, training data were obtained from the BNC. As a result, approximately 700 instances were obtained for each target noun on average.

Second, the probabilities defined by Eq. (3) and Eq. (4) were estimated from the training data. The smoothing parameter $\alpha$ was set to 1.

Third, using the estimated probabilities, the value of $p(c|x)$, as defined by Eq. (2), was calculated for each example sentence in the test data. If $p(c|x) > 0.5$, the target noun was predicted as countable; otherwise it was predicted as uncountable. The predicted countability that agreed with the one given by Huddleston and Pullum [8] was judged to be correct. The overall accuracy was calculated by the number of correctly predicted target nouns divided by that of target nouns in the test data (i.e., 50). For comparison, the Nagata et al. [13]'s method was implemented using the BNC.

For the second task, these countable-target sentences whose countability satisfied $P(c|x) \leq 0.85$, which was a relatively high value and gave a fairly number of instances, were chosen as poor examples of countable usage. Similarly, sentences with uncountable-target nouns whose value for $P(c|x) \geq 0.15$ were selected from the 25 uncountable examples. As a result, 18 sentences were chosen. Each of these sentences was revised by adding strong indicators to the sentence, indicators which had been extracted by the method described in Sect. 4. From top to bottom

---

\[\text{http://www.cs.nyu.edu/~schair/FORPROJECT/OAK/}\]
ing the example sentences. Of 18 pairs, eight revised example sentences were judged to be better than their original; the other seven and three were judged to be not better and equally good, respectively. Namely, only 53% of the revised examples were judged to be better than their original.

After the task, however, an interview with the rater revealed that he had preferred shorter example sentences because he found them easier to judge in terms of the contribution of the context words to their countability. Since some of the revised sentences had more than one indicator word added to satisfy \( P(c|x) > 0.50 \) or \( P(c|x) < 0.10 \), they were longer than their original, and thus more difficult to interpret.

Considering the effect of sentence length, we removed example sentences to which more than one indicator word were added. Of remaining nine example sentences, seven (78%) were judged to be better than their original, a much better improvement ratio compared to the set containing the longer sentences (53%); more than one indicator word had been added to five of the seven not-better example sentences. From these results, it follows that there are at least two major factors in making a good example sentence for countability: one is that it should contain some words that are informative for countability; the other is that it should be relatively short so that the relations between the target noun and its surrounding words can be easily interpreted. Fortunately, the two conditions can be easily satisfied using the proposed method.

6 Conclusions

In this paper, we have described a method for visualizing the countability of English nouns in context. The method visualizes it by means of the probability that the target noun is recognized to be countable in context. The method can also extract good indicators for countability from corpus data which can be used to make example sentences for the countable and uncountable usages of the target noun in second language learning and teaching.

The experiments have shown that the probability can be estimated from corpus data with a high accuracy. Consequently, the visualization based on the probability agrees with our intuitive interpretation of countability. The experiments have also shown that using the proposed method, example sentences for the countable and uncountable usages of the target nouns can be efficiently revised to yield better ones although it should be noted that their length is another important factor.

Acknowledgments

The authors would like to thank Satoshi Sekine who has developed the OAK System. The authors also would like to thank three anonymous reviewers for their useful comments on this paper.

References


