UBA: Using Automatic Translation and Wikipedia for Cross-Lingual Lexical Substitution

Pierpaolo Basile
Dept. of Computer Science
University of Bari "Aldo Moro"
Via E. Orabona, 4
70125 Bari (ITALY)
basilepp@di.uniba.it

Giovanni Semeraro
Dept. of Computer Science
University of Bari "Aldo Moro"
Via E. Orabona, 4
70125 Bari (ITALY)
semeraro@di.uniba.it

Abstract

This paper presents the participation of the University of Bari (UBA) at the SemEval-2010 Cross-Lingual Lexical Substitution Task. The goal of the task is to substitute a word in a language L_s, which occurs in a particular context, by providing the best synonyms in a different language L_t which fit in that context. This task has a strict relation with the task of automatic machine translation, but there are some differences: Cross-lingual lexical substitution targets one word at a time and the main goal is to find as many good translations as possible for the given target word. Moreover, there are some connections with Word Sense Disambiguation (WSD) algorithms. Indeed, understanding the meaning of the target word is necessary to find the best substitutions. An important aspect of this kind of task is the possibility of finding synonyms without using a particular sense inventory or a specific parallel corpus, thus allowing the participation of unsupervised approaches. UBA proposes two systems: the former is based on an automatic translation system which exploits Google Translator, the latter is based on a parallel corpus approach which relies on Wikipedia in order to find the best substitutions.

1 Introduction

The goal of the Cross-Lingual Lexical Substitution (CLLS) task is to substitute a word in a language L_s, which occurs in a particular context, by providing the best substitutions in a different language L_t. In SemEval-2010 the source language L_s is English, while the target language L_t is Spanish. Clearly, this task is related to Lexical Substitution (LS) (McCarthy and Navigli, 2007) which consists in selecting an alternative word for a given one in a particular context by preserving its meaning. The main difference between the LS task and the CLLS one is that in LS source and target languages are the same. CLLS is not a easy task since neither a list of candidate words nor a specific parallel corpus are supplied by the organizers. However, this opens the possibility of using several knowledge sources, instead of a single one fixed by the task organizers. Therefore, the system must identify a set of candidate words in L_t and then select only those words which fit the context. From another point of view, the cross-lingual nature of the task allows to exploit automatic machine translation methods, hence the goal is to find as many good translations as possible for the given target word. A thorough description of the task can be found in (Mihalcea et al., 2010; Sinha et al., 2009).

To easily understand the task, an example follows. Consider the sentence:

> *During the siege, George Robertson had appointed Shuja-ul-Mulk , who was a **bright** boy only 12 years old and the youngest surviving son of Aman-ul-Mulk, as the ruler of Chitral.*

In the previous sentence the target word is "**bright**". Taking into account the meaning of the word "**bright**" in this particular context, the best substitutions in Spanish are: "**inteligente**", "**brillante**" and "**listo**".

We propose two systems to tackle the problem of CLLS: the first is based on an automatic translation system which exploits the API of Google Translator[1], the second is based on a parallel corpus approach which relies on Wikipedia. In particular, in the second approach we use a structured version of Wikipedia called DBpedia (Bizer

[1] http://code.google.com/p/google-api-translate-java/

Proceedings of the 5th International Workshop on Semantic Evaluation, ACL 2010, pages 242–247,
Uppsala, Sweden, 15-16 July 2010.

et al., 2009). Both systems adopt several lexical resources to select the list of possible substitutions for a given word. Specifically, we use three different dictionaries: Google Dictionary, Babylon Dictionary and Spanishdict. Then, we combine the dictionaries into a single one, as described in Section 2.1.

The paper is organized as follows: Section 2 describes the strategy we adopted to tackle the CLLS task, while results of an experimental session we carried out in order to evaluate the proposed approaches are presented in Section 3. Conclusions are discussed in Section 4.

2 Methodology

Generally speaking, the problem of CLLS can be coped with a strategy which consists of two steps, as suggested in (Sinha et al., 2009):

- *candidate collection*: in this step several resources are queried to retrieve a list of potential translation candidates for each target word and part of speech;
- *candidate selection*: this step concerns the ranking of potential candidates, which are the most suitable ones for each instance, by using information about the context.

Regarding the candidate collection, we exploit three dictionaries: Google Dictionary, Babylon Dictionary and Spanishdict. Each dictionary is modeled using a strategy described in Section 2.1. We use the same approach to model each dictionary in order to make easy both the inclusion of future dictionaries and the integration with the candidate selection step.

Candidate selection is performed in two different ways. The first one relies on the automatic translation of the sentence in which the target word occurs, in order to find the best substitutions. The second method uses a parallel corpus built on DBpedia to discover the number of documents in which the target word is translated by one of the potential translation candidates. Details about both methods are reported in Section 2.2

2.1 Candidate collection

This section describes the method adopted to retrieve the list of potential translation candidates for each target word and part of speech.

Our strategy combines several bi-lingual dictionaries and builds a single list of candidates for each target word. The involved dictionaries meet the following requirements:

1. the source language L_s must be English and the target one L_t must be Spanish;
2. each dictionary must provide information about the part of speech;
3. the dictionary must be freely available.

Moreover, each candidate has a score s_{ij} computed by taking into account its rank in the list of possible translations supplied by the $i-th$ dictionary. Formally, let us denote by $D = \{d_1, d_2, \ldots, d_n\}$ the set of n dictionaries and by $L_i = \{c_1, c_2, \ldots, c_{m_i}\}$ the list of potential candidates provided by d_i. The score s_{ij} is computed by the following equation:

$$s_{ij} = 1 - \frac{j}{m_i} \quad j \in \{1, 2, \ldots, m_i\} \qquad (1)$$

Since each list L_i has a different size, we adopt a score normalization strategy based on Z-score to merge the lists in a unique one. Z-score normalizes the scores according to the average μ and standard deviation σ. Given the list of scores $L = \{s_1, s_2, \ldots, s_n\}$, μ and σ are computed on L and the normalized score is defined as:

$$\overline{s_i} = \frac{s_i - \mu}{\sigma} \qquad (2)$$

Then, all the lists L_i are merged in a single list M. The list M contains all the potential candidates belonging to all the dictionaries with the related score. If a candidate occurs in more than one dictionary, only the occurrence with the maximum score is chosen.

At the end of the candidate collection step the list M of potential translation candidates for each target word is computed. It is important to point out that the list M is sorted and supplies an initial rank, which can be then modified by the candidate selection step.

2.2 Candidate selection

While the candidate collection step is common to the two proposed systems, the problem of candidate selection is faced by using different strategies in the two systems.

The first system, called `unibaTranslate`, uses a method based on google-api-translate-java[2]. The main idea behind `unibaTranslate` is to look for a potential candidate in the translation of the target sentence. Sometimes, no potential candidates occur into the translation. When this happens the system uses some heuristics to discover a possible translation.

For example, given the target word **"raw"** and the potential candidates $M = \{$*puro, crudo, sin refinar, de baja calidad, agrietado, al natural, bozal, asado, frito and bruto*$\}$, the two possible scenarios are:

1. a potential candidate occurs into the translation:
   - S_{en}: *The **raw** honesty of that basic crudeness makes you feel stronger in a way.*
   - S_{es}: *La **cruda** honestidad de esa crudeza de base que te hace sentir mas fuerte en un camino.*
2. no potential candidates occur into the translation, but a correct translation of the target word is provided:
   - S_{en}: *Many institutional investors are now deciding that they are getting a **raw** deal from the company boards of Australia.*
   - S_{es}: *Muchos inversores institucionales estan ahora decidiendo que estan recibiendo un trato **injusto** de los directorios de las empresas de Australia.*

In detail, the strategy can be split in several steps:

1. Retrieve the list M of potential translation candidates using the method described in Section 2.1.
2. Translate the target sentence S_{en} from English to Spanish, using the google-api-translate-java, which results into the sentence S_{es}.
3. Enrich M by adding multiword expressions. To implement this step, the two bigrams which contain the target word and the only trigram in which the target word is the 2^{nd} term are taken into to account. Coming back to the first sentence in the previous example, the following n-grams are built: *"the raw"*, *"raw honesty"* and *"the raw honesty"*. For each n-gram, candidate translations are looked for using Google Dictionary. If translations are found, they are added to M with an initial score equal to 0.
4. Fix a window W[3] of n words to the right and to the left of the target word, and perform the following steps:
   (a) for each candidate c_k in M, try to find c_k in W. If c_k occurs in W, then add 2 to the score of c_k in M;
   (b) if no exact match is found in the previous step, perform a new search by comparing c_k with the words in W using the Levenshtein distance[4](Levenshtein, 1966). If the Levenshtein distance is greater than 0.8, then add 2 to the score of c_k in M.
5. If no exact/partial match is found in the previous steps, probably the target word is translated with a word which does not belong to M. To overcome this problem, we implement a strategy able to discover a possible translation in S_{es} which is not in M. This approach involves three steps:
   (a) for each word w_i in S_{en}, a list of potential translations P_i is retrieved;
   (b) if a word in P_i is found in S_{es}, the word is removed from S_{es}[5];
   (c) at this point, S_{es} contains a list R of words with no candidate translations. A score is assigned to those words by taking into account their position in S_{es} with respect to the position of the target word in S_{en}, using the following equation:
   $$1 - \frac{|pos_c - pos_t|}{L_{max}} \quad (3)$$
   where pos_c is the translation candidate position in S_{es}, pos_t is the target word position in S_{en} and L_{max} is the maximum length between the length of S_{en} and S_{es}.

[2]http://code.google.com/p/google-api-translate-java/

[3]The window W is the same for both S_{en} and S_{es}.

[4]A normalized Levenshtein distance is adopted to obtain a value in [0, 1].

[5]A partial match based on normalized Levenshtein distance is implemented.

Moreover, the words not semantically related to the potential candidates (found using Spanish WordNet[6]) are removed from R. In detail, for each candidate in M a list of semantically related words in Spanish WordNet[7] is retrieved which results in a set WN of related words. Words in R but not in WN are removed from R. In the final step, the list R is sorted and the first word in R is added to M assigning a score equal to 2.

6. In the last step, the list M is sorted. The output of this process is the ranked list of potential candidates.

It is important to underline that both S_{en} and S_{es} are tokenized, part-of-speech tagged and lemmatized. Lemmatization plays a key role in the matching step, while part-of-speech tagging is needed to query both the dictionaries and the Spanish WordNet. We adopt META (Basile et al., 2008) and FreeLing (Atserias et al., 2006) to perform text processing for English and Spanish respectively.

The second proposed system, called `unibaWiki`, is based on the idea of automatically building a parallel corpus from Wikipedia. We use a structured version of Wikipedia called DBpedia (Bizer et al., 2009). The main idea behind DBpedia is to extract structured information from Wikipedia and then to make this information available. The main goal is to have access easily to the large amount of information in Wikipedia. DBpedia opens new and interesting ways to use Wikipedia in NLP applications.

In CLLS task, we use the extended abstracts of English and Spanish provided by DBpedia. For each extended abstract in Spanish which has the corresponding extended abstract in English, we build a document composed by two fields: the former contains the English text ($text_{en}$) and the latter contains the Spanish text ($text_{es}$). We adopt Lucene[8] as storage and retrieval engine to make the documents access fast and easy.

The idea behind `unibaWiki` is to count, for each potential candidate, the number of documents in which the target word occurs in $text_{en}$ and the potential candidate occurs in $text_{es}$. A score equal to the number of retrieved documents is assigned, then the candidates are sorted according to that score.

Given the list M of potential candidates and the target word t, for each $c_k \in M$ we perform the following query:

$$text_{en} : t \text{ AND } text_{es} : c_k$$

where the field name is followed by a colon and by the term you are looking for.

It is important to underline here that multiword expressions require a specific kind of query. For each multiword expression we adopt the PhraseQuery which is able to retrieve documents that contain a specific sequence of words instead of a single keyword.

2.3 Implementation

To implement the candidate collection step we developed a Java application able to retrieve information from dictionaries. For each dictionary, a different strategy has been adopted. In particular:

1. *Google Dictionary*: Google Dictionary website is queried by using the HTTP protocol and the answer page is parsed;

2. *Spanishdict*: the same strategy adopted for Google Dictionary is used for the Spanishdict website[9];

3. *Babylon Dictionary*: the original file available from the Babylon website[10] is converted to obtain a plain text file by using the Unix utility *dictconv*. After that, an application queries the text file in an efficient way by means of a hash map.

Both candidate selection systems are developed in Java. Regarding the `unibaWiki` system, we adopt Lucene to index DBpedia abstracts. The output of Lucene is an index of about 680 Mbytes, 277,685 documents and about 1,500,000 terms.

3 Evaluation

The goal of the evaluation is to measure the systems' ability to find correct Spanish substitutions for a given word. The dataset supplied by the organizers contains 1,000 instances in XML format.

[6] http://www.lsi.upc.edu/∼nlp/projectes/ewn.html

[7] The semantic relations of hyperonymy, hyponymy and "similar to" are exploited.

[8] http://lucene.apache.org/

[9] http://www.spanishdict.com/

[10] www.babylon.com

Moreover, the organizers provide trial data composed by 300 instances to help the participants during the development of their systems.

The systems are evaluated using two scoring types: **best** scores the best guessed substitution, while out-of-ten (**oot**) scores the best 10 guessed substitutions. For each scoring type, precision (*P*) and recall (*R*) are computed. Mode precision (*P-mode*) and mode recall (*R-mode*) calculate precision and recall against the substitution chosen by the majority of the annotators (if there is a majority), respectively. Details about evaluation and scoring types are provided in the task guidelines (McCarthy et al., 2009).

Results of the evaluation using trial data are reported in Table 1 and Table 2. Our systems are tagged as **UBA-T** and **UBA-W**, which denote `unibaTranslate` and `unibaWiki`, respectively. Systems marked as *BL-1* and *BL-2* are the two baselines provided by the organizers. The baselines use Spanishdict dictionary to retrieve candidates. The system *BL-1* ranks the candidates according to the order returned on the online query page, while the *BL-2* rank is based on candidate frequencies in the Spanish Wikipedia.

Table 1: **best** results (trial data)

System	P	R	P-mode	R-Mode
BL-1	24.50	24.50	51.80	51.80
BL-2	14.10	14.10	28.38	28.38
UBA-T	**26.39**	**26.39**	**59.01**	**59.01**
UBA-W	22.18	22.18	48.65	48.65

Table 2: **oot** results (trial data)

System	P	R	P-mode	R-Mode
BL-1	38.58	38.58	71.62	71.62
BL-2	37.83	37.83	68.02	68.02
UBA-T	44.16	44.16	**78.38**	**78.38**
UBA-W	**45.15**	**45.15**	72.52	72.52

Results obtained using trial data show that our systems are able to overcome the baselines. Only the best score achieved by *UBA-W* is below *BL-1*. Moreover, our strategy based on Wikipedia (*UBA-W*) works better than the one proposed by the organizers (*BL-2*).

Results of the evaluation using test data are reported in Table 3 and Table 4, which include all the participants. Results show that *UBA-T* obtains the highest recall using **best** scoring strategy. Moreover, both systems *UBA-T* and *UBA-W* achieve the highest *R*-mode and *P*-mode using **oot** scoring strategy. It is worthwhile to point out that the presence of duplicates affect recall (*R*) and precision (*P*), but not *R*-mode and *P*-mode. For this reason some systems, such as *SWAT-E*, obtain very high recall (*R*) and low *R*-mode using **oot** scoring. Duplicates are not produced by our systems, but we performed an a posteriori experiment in which duplicates are allowed. In that experiment, the first candidate provided by *UBA-T* has been duplicated ten times in the results. Using that strategy, *UBA-T* achieves a recall (and precision) equal to 271.51. This experiment proves that also our system is able to obtain the highest recall when duplicates are allowed into the results. Moreover, it is important to underline here that we do not know how other participants generate duplicates in their results. We adopted a trivial strategy to introduce duplicates.

Table 3: **best** results (test data)

System	P	R	P-mode	R-Mode
BL-1	24.34	24.34	50.34	50.34
BL-2	15.09	15.09	29.22	29.22
UBA-T	**27.15**	**27.15**	57.20	57.20
UBA-W	19.68	19.68	39.09	39.09
USPWLV	26.81	26.81	**58.85**	**58.85**
Colslm	27.59	25.99	59.16	56.24
WLVUSP	25.27	25.27	52.81	52.81
SWAT-E	21.46	21.46	43.21	43.21
UvT-v	21.09	21.09	43.76	43.76
CU-SMT	21.62	20.56	45.01	44.58
UvT-g	19.59	19.59	41.02	41.02
SWAT-S	18.87	18.87	36.63	36.63
ColEur	19.47	18.15	40.03	37.72
IRST-1	22.16	15.38	45.95	33.47
IRSTbs	22.51	13.21	45.27	28.26
TYO	8.62	8.39	15.31	14.95

Table 4: **oot** results (test data)

System	P	R	P-mode	R-Mode
BL-1	44.04	44.04	73.53	73.53
BL-2	42.65	42.65	71.60	71.60
UBA-T	47.99	47.99	81.07	81.07
UBA-W	52.75	52.75	**83.54**	**83.54**
USPWLV	47.60	47.60	79.84	79.84
Colslm	46.61	43.91	69.41	65.98
WLVUSP	48.48	48.48	77.91	77.91
SWAT-E	**174.59**	**174.59**	66.94	66.94
UvT-v	58.91	58.91	62.96	62.96
UvT-g	55.29	55.29	73.94	73.94
SWAT-S	97.98	97.98	79.01	79.01
ColEur	44.77	41.72	71.47	67.35
IRST-1	33.14	31.48	58.30	55.42
IRSTbs	29.74	8.33	64.44	19.89
TYO	35.46	34.54	59.16	58.02
FCC-LS	23.90	23.90	31.96	31.96

Finally, Table 5 reports some statistics about *UBA-T* and the number of times (*N*) the candi-

date translation is taken from Spanish WordNet (*Spanish WN*) or multiword expressions (*Multiword exp.*). The number of instances in which the candidate is a correct substitution is reported in column *C*. Analyzing the results we note that most errors are due to part-of-speech tagging. For example, given the following sentence:

S_{en}: You will still be responsible for the shipping and handling fees, and for the cost of **returning** the merchandise.

S_{es}: Usted seguira siendo responsable de los gastos de envio y manipulacion y, para los gastos de **devolucion** de la mercancia.

where the target word is the verb *return*. In this case the verb is used as noun and the algorithm suggests correctly *devolucion* (noun) as substitution instead of *devolver* (verb). The gold standard provided by the organizers contains *devolver* as substitution and there is no match between *devolucion* and *devolver* during the scoring.

Table 5: *UBA-T* statistics.

Strategy	**N**	**C**
Spanish WN	34	11
Multiword exp.	21	11

4 Conclusions

We described our participation at SemEval-2 Cross-Lingual Lexical Substitution Task, proposing two systems called *UBA-T* and *UBA-W*. The first relies on Google Translator, the second is based on DBpedia, a structured version of Wikipedia. Moreover, we exploited several dictionaries to retrieve the list of candidate substitutions.

UBA-T achieves the highest recall among all the participants to the task. Moreover, the results proved that the method based on Google Translator is more effective than the one based on DBpedia.

Acknowledgments

This research was partially funded by Regione Puglia under the contract POR PUGLIA 2007-2013 - Asse I Linea 1.1 Azione 1.1.2 - Bando “Aiuti agli Investimenti in Ricerca per le PMI” - Fondo per le Agevolazioni alla Ricerca, project title: “Natural Browsing”.

References

Jordi Atserias, Bernardino Casas, Elisabet Comelles, Meritxell González, Lluís Padró, and Muntsa Padró. 2006. FreeLing 1.3: Syntactic and semantic services in an open-source NLP library. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC06)*, pages 48–55.

Pierpaolo Basile, Marco de Gemmis, Anna Lisa Gentile, Leo Iaquinta, Pasquale Lops, and Giovanni Semeraro. 2008. METAL - MultilanguagE Text Analyzer. In *Proceedings of the Language and Speech Technnology Conference - LangTech 2008, Rome, Italy, February 28-29*, pages 137–140.

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia-A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, Issue 7:154–165.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet Physics-Doklady*, volume 10.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 Task 10: English Lexical Substitution Task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic, June. Association for Computational Linguistics.

Diana McCarthy, Rada Sinha, and Ravi Mihalcea. 2009. Cross Lingual Lexical Substitution. http://lit.csci.unt.edu/DOCS/task2clls-documentation.pdf.

Rada Mihalcea, Ravi Sinha, and Diana McCarthy. 2010. SemEval-2010 Task 2: Cross-Lingual Lexical Substitution. In *Proceedings of the 5th International Workshop on Semantic Evaluations (SemEval-2010)*. Association for Computational Linguistics.

Ravi Sinha, Diana McCarthy, and Rada Mihalcea. 2009. SemEval-2010 Task 2: cross-lingual lexical substitution. In *SEW ’09: Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 76–81, Morristown, NJ, USA. Association for Computational Linguistics.