Intelligent Information Systems 9999
ISBN 666-666-666, pages 1–10

Comparing greedy and optimal coverage strategies for shallow-transfer machine translation

Jernej Vicic[1] and Mikel L. Forcada[2]

[1] University of Primorska, 6000 Koper, Slovenia
[2] DLSI, Universitat d'Alcant, E-03071 Alacant, Spain

Abstract

Using the open-source Apertium platform as an example of a shallow-transfer machine translation system and Spanish and Catalan data, we compare greedy (left-to-right, longest-match) and optimal-coverage strategies when chunking the input sentence using the left-hand-side patterns of shallow transfer rules. We find that, when rules are reasonably correct, there is almost no difference between both strategies. The selection of unreasonable rules (such as rules which delete or insert content words) may be curtailed by using a length modification penalty term.

Keywords: machine translation, transfer rule, LRLM, part-of-speach

1 Introduction

One of the key aspects of open source software is that it allows for an easy implementation and sharing of proper ideas and algorithms. The modularity of the basic design of the Apertium open-source machine translation platform Armentano and Forcada (2006); Corbi-Bellot *et al.* (2005) further simplifies the introduction of new algorithms into a working and complete system, in this case a natural language translation system. The process of selecting the most appropriate rule is done using a greedy strategy in a left-to-right longest match (LRLM) manner Divay and Vitale (1997); Armentano and Forcada (2006); Corbi-Bellot *et al.* (2005), meaning that tokens (lexical units) are added to a temporary transfer string until there is no rule matching that string. The last applicable rule, that is, the rule with the longest string of tokens, is applied to the temporary string. Although the LRLM strategy intuitively seems the most natural and certainly one of the fastest strategies (divay1), there are other possible combinations of applicable rules. For instance, if the input string is abcde and the rule set has left-hand sides ab, abc, cde, de, the greedy LRLM would segment the string as [abc][de], and would discard the alternate choice [ab][cde], which could yield to a better translation. In other words, there may exist other coverings of the input sentence using same set of translation rules. Considering only translation quality, the exhaustive search of all possible coverings of the input sentence might be expected to produce a better translation than LRLM. Two problems arise at this point:

1. the need for a way to score each covering of the input sentence

2. the time complexity of the optimal covering (exhaustive search) algorithm

The first problem is addressed by this article and is thoroughly discussed in section 2.2. The second problem is only briefly addressed, and will be left for further investigation. The time complexity is defined by:

$$\sum_{i=1}^{L-1} N_r(i) \tag{1}$$

where L is the sentence length in lexical units and Nr is the number of applicable rules at position i. The rest of the article is organized as follows: section 2 describes the newly devised method for scoring transfer rules, describing the tools and the environment the method was applied to. The next section describes the test system and shows the results. The article ends with conclusions and definition of open issues.

2 Method

This chapter presents the implementation of the proposed method and the environment the method was applied in.

2.1 Apertium

A brief description of Apertium follows; more details can be found in Armentano and Forcada (2006); Corbi-Bellot *et al.* (2005). Apertium is a machine translation platform born as part of a large government-funded project involving universities and language technology companies. Apertium is based on an intuitive approach: to produce fast, reasonably intelligible and easily correctable translations between related languages, it suffices to use an MT strategy which uses shallow parsing techniques to refine word-for-word machine translation. Apertium uses finite-state transducers for lexical processing (powerful enough to treat many kinds of multi-word expressions), hidden Markov models (HMM) for part-of-speech tagging (solving categorical lexical ambiguity), and finite-state-based left-to-right longest-match (LRLM) chunking for structural transfer (local structural processing based on simple and well-formulated rules for some simple structural transformations such as word reordering, number and gender agreement, etc.). The components of the Apertium machine translation toolbox have been released under open-source licenses such as the GNU General Public License2 and one of the Creative Commons licenses.3 Figure 1 shows the sequence of Apertium modules. The de-formatter identifies text that will be translated and separates it from formatting information; the morphological analyzer identifies and analyzes lexical units using monolingual dictionaries, and delivers all possible analyses. The part-of-speech tagger choses one of these analyses according to the context in source text. The structural transfer is composed of a set of rules for shallow transfer Armentano and Forcada (2006); Corbi-Bellot *et al.* (2005); this module is the one which is studied in this paper.

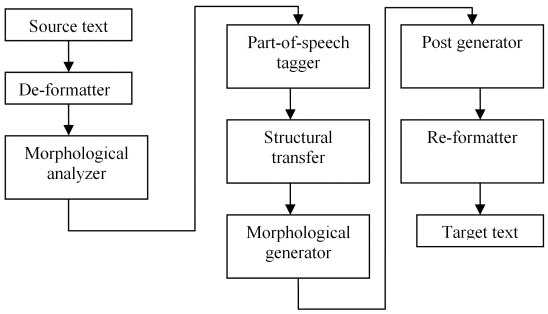

FIGURE 1: Modules of an Apertium translation system. The module which is changed in this paper is the structural transfer module.

The morphological generator uses target-language monolingual dictionaries to inflect the lexical units produced by previous modules. The post-generator applies fine-tune rules (local orthographic transformations such as apostrophizing or contractions) at the end of translation process. The re-formatter reinstates the formatting into the translated text.

2.2 Using a language model to score translation

A target language model Bahl *et al.* (1989); Clarkson (1997), in particular a trigram language model, is used as a metric for scoring the quality of produced translation candidates. A language model assigns a higher score (usually a probability) to a sentence that is most likely to appear in a certain language; the model is trained on a corpus; therefore, what it actually tries to tell is how likely is that a sentence like that would have appeared in that corpus. Note that this score gives the likelihood of the sentence rather than a likelihood of it being a translation of the source sentence. A target language model is constructed using a large monolingual target language corpus that describes the target language as accurately as possible. The language model has to be constructed prior to the testing phase. The probability of appearance of the translation done by the tested method can be calculated from the target language model. Our assumption is that this probability alone can be used as an indication of translation quality, since it is quite unlikely for Apertium's structural transfer to generate sentences which are not related at all to the source sentence, and, therefore, target-language probability will be enough to select the best one among them. As said before, structural transfer rules of a shallow transfer translation system mostly deal with word reordering and agreement of neighboring words. Note that the approach proposed here does not require the use of aligned bilingual corpora, but instead uses target text only. The construction of a bilingual corpus, a lengthy and costly process, can therefore be avoided here.

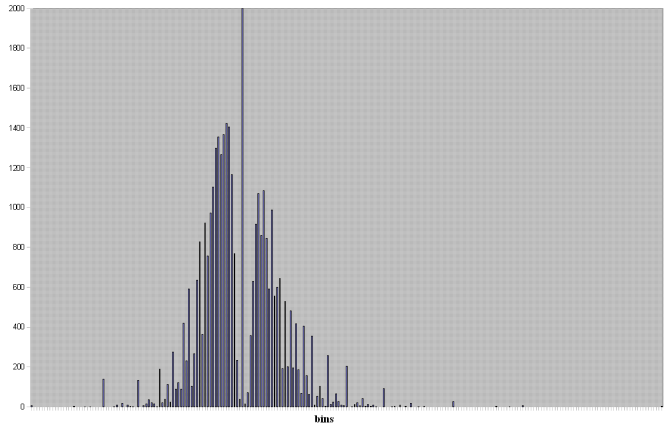

FIGURE 2: The distribution of the quotient between the length of source sentence and that of the translation in Spanish (s-source) and Catalan (t-target). The mean value is 0,9953 and the standard deviation is 0,07.

2.3 Enhanced language model considering the length of the sentence

The language model discussed in section 2.2 (as most probabilistic models do) tends to assign higher (better) values to shorter sentences as these are more likely to appear in a natural language Bahl *et al.* (1989); Clarkson (1997) and the probability is computed as the product of n-gram probabilities. This means that it favors rules that delete words, that is, rules that shorten sentences. Although rules produced by experts seldom show such behavior, this assumption may not hold true for automatically produced rules without restrictions. A factor that penalizes this effect has been added to the original metric. The observation of the quotient x of the length of source sentences in Spanish and to that of their translations in Catalan in a relatively big corpus (around 50000 sentences, see Figure 2) led us to the conclusion that its distribution may be approximated by a normal distribution.

$$f(x, \sigma, \mu) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2}$$

where μ is the mean and σ is the standard deviation of x. The probability computed by the language model is multiplied by this penalizing factor.

2.4 Method implementation

This chapter describes the application of the new method to the Apertium system, specifically to structural transfer module. Apertium is composed of modules that operate sequentially; the result of a module is the input of the succeeding module, see Figure 1. The communication between modules is implemented as a simple input/output text stream. A new module can be easily inserted that parses the input of the preceding module and produces an output which is compatible with

Today is a beautiful day .
is a beautiful day .
a beautiful day . beautiful day .
day .

FIGURE 3: All possible suffixes are generated (in the example, part-of-speech tags are omitted for readability)

Sus acciones han subido de un 75% desde el ano pasado .
^El seu <det><pos><m><pl>$ ^acci<n><f><pl>$
^haver<vbhaver><pri><p3><pl>$
^pujar<vblex><pp><m><sg>$ ^de<pr>$
^un<det><ind><m><sg>$ ^75<num>$
^des de<pr>$ ^el<det><def><m><sg>$
^any<n><m><sg>$ ^passar<vblex><pp><m><sg>$
^.<sent>$
_0_1_-1_25_-1_0_-1_1_5_25_42

FIGURE 4: The triple presents source sentence, the string of rules and the uncompleted translation generated by that string of rules

the succeeding module. Our method was implemented by changing the behavior of an existing module, the structural transfer module. The new system can only be used to test the method, and not in a production system, because the original functionality is changed (the system produces scored translations of all possible coverings of an input sentence and not only the best translation).
The original structural transfer algorithm skips input tokens that are not matched by any transfer rule. To simulate this behavior, a dummy rule has been introduced that is applied every time no proper rule can be applied, for the purpose of these experiments when no rule pattern matches the observed lexical unit. This enables full coverage of each input sentence.
The input of the module is expanded to all possible substrings (the input sentence is expanded to all possible suffixes). All possible suffixes of the input to the transfer module, tokens composed of lexical form (lemmas and part-of-speech tags of input words) are generated as in Figure 3.

Each generated suffix, a string of words, is treated as a separate sentence. All possible rules are applied to each position instead of just the rule with longest applicable pattern. All possible applicable rules and their results, translated values for tokens that these rules were applied on, are stored for each position. The results are in the same format as the original structural transfer data, the only difference is that the new results are expanded for each possible suffix. All possible rule sequences covering the whole sentence are produced from the data generated. Most of the rule sequences are discarded as they do not cover the complete sentence.

Each string of rules covering the whole sentence is represented by a triple holding the source sentence, the string of rules and the uncompleted translation generated by that string of rules (see Figure 4). The final phase of the method

Sus acciones han subido de un 75% desde el ano pasado .
^El seu <det><pos><m><pl>$ ^acci<n><f><pl>$
^haver<vbhaver><pri><p3><pl>$
^pujar<vblex><pp><m><sg>$ ^de<pr>$
^un<det><ind><m><sg>$ ^75<num>$
^des de<pr>$ ^el<det><def><m><sg>$
^any<n><m><sg>$ ^passar<vblex><pp><m><sg>$
^.<sent>$
_0_1_-1_25_-1_0_-1_1_5_25_42
Els seus accions han pujat d'un 75% des de l'any passat .

2.65465883353068E-28

FIGURE 5: The final result of the method is a set of scored 5-tuples like this one, each one representing a single string of rules covering the whole sentence.

steps through the remaining modules of the Apertium system and completes the translations corresponding to each triple. These translations are scored using a trigram language model (bahl1, clarcson1) and stored for further observation. The final result is in a form of a 5-tuple representing each translation candidate holding: (1) the source sentence, (2) the string of rules, (3) the output of the actions of the string of rules from Figure 2. Rules were numbered and this string presents the rules used in this particular translation. The number -1 denotes that no rule was used for the particular lexical chunk. (4)the translation based on Figure 3. (5) score of the translation in Figure 4. using the language model.
An example can be seen in Figure 5. The 5-tuple holding the best score represents the optimal string of rules for the source sentence.

3 Experiments

The method was inserted into the Apertium translation system and evaluated using the Spanish Catalan language pair. Test data was produced from a Spanish corpus which is described in section 3.3. The trigram language model for scoring the translations was computed from a Catalan corpus which is also described in section 3.3.

3.1 The translation system

The method was tested on a newly installed translation system Apertium Armentano and Forcada (2006); Corbi-Bellot *et al.* (2005), version 3.0, for the Spanish-Catalan (es-ca) language pair. This pair was selected as the most tested and widely used on Apertium. The system is described in section 2.1.

3.2 The structural transfer module

The structural transfer module uses finite-state pattern matching to detect (in a left-to-right, longest-match way) fixed-length patterns of lexical forms (chunks

```
<rule> <!– REGLA: DETERMINANT –>
<pattern>
<pattern-item n="det"/>
</pattern>
<action>
<call-macro n="f_concord1">
<with-param pos="1"/>
</call-macro>
<out>
<lu>
<clip pos="1" side="tl" part="whole"/>
</lu>
</out>
</action>
</rule>
```

FIGURE 6: An example of an Apertium structural transfer rule which matches a determiner (a sequence of one lexical form). Rules are written in an XML format as specified by the Apertium platform.

or phrases) needing special processing due to grammatical divergences between the two languages (gender and number changes to ensure agreement in the target language, word reordering, lexical changes such as changes in prepositions, etc.) and performs the corresponding transformations.

An example of a rule is presented in Figure 6. A rule is composed of two parts: *pattern* and *action*. Patterns are usually expressed in terms of lexical categories, for instance, "article-noun" or "article-noun-adjective". The *action* part determines what action should be executed on particular pattern. Macros present an efficient tool to check concordances and agreements between lexical units of the found pattern, the *out* part deals with actual output generation.

3.3 Corpora

Two corpora were used one for each language of the translating language pair: Spanish-Catalan. Corpora were chosen mostly by the availability although used corpora were not domain specific and large enough to perform the tests. Both Spanish and Catalan corpora were collections of articles from the daily newspaper El Periodico de Catalunya, which is published simultaneously in Spanish and Catalan. The size of each corpus was 2 million words.

3.4 Empirical results

Two series of tests were conducted using both scoring functions described in sections 2.2 and 2.3. Descriptive results are presented in Table 1, cumulative and comparative results are presented in Table 2. A malicious (unreasonable) rule was added to the original set of rule to simulate bad rules that could be, for instance, the result of automatic learning or a linguist's mistake. Using language models

Pattern: determiner blank noun blank adjective Action: Output: adjective adjective noun

FIGURE 7: The malicious rule, whenever a pattern of determiner noun adjective occurs, change it to determiner adjective (delete the noun)

Original: Sus acciones han subido de un 75Les seves accions han pujat d'un 75Changed: Sus acciones han subido de un 75Els seus accions han pujat d'un 75

FIGURE 8: Example of the effect of the malicious rule. The first pair represents a Spanish source sentence and an acceptable translation into Catalan; the second pair was produced using the malicious rule.

to discover these malicious rules is one of the primary goals of this research. The malicious rule added was applied to a pattern that is very common and was used in roughly 12% of the test sentences. The rule is presented in Figure 7. The malicious rule produces shorter sentences (which may be favored by the use of a trigram model without a correction factor). An example of the effect of the malicious rule is shown in Figure 8.

Tests were conducted on three test sets of 480, 1000 and 2000 sentences respectively. Table 1 presents descriptive results of the evaluation process. Tests were conducted on 480, 1000 and 2000 Spanish test sentences. The two scoring methods (presented in section 2.2 and 2.3 respectively) were used. When using no malicious rules, a small percentage (a few examples on each test-set) of LRLM sentence coverings were found as non-optimal by both scoring methods; a closer inspection of the examples proved that alternate coverings were selected as optimal because of the selection of a word in different gender that has a larger probability in the corpus, mostly selecting masculine gender instead of feminine, with that choice leading to an incorrect translation. This inspection was done by hand as there were just a small number of such cases. The results show that the LRLM algorithm produces optimal coverage using the original set of rules. This result was expected as LRLM algorithm is used in production translation systems and the rules have been designed with the LRLM model in mind. Table 2 presents the results of the evaluation of a system using a slightly changed set of rules, in which the malicious rule discussed above has been added to the original rule set. The purpose of this test is to determine if the probability distributions can be used to detect malicious rules.

4 Conclusion

We find that, when rules are reasonably correct (such as rules written by an expert), there is almost no difference between applying them using a left-to-right, longest-match (LRLM) greedy strategy or an optimal-coverage strategy. However,

TABLE 1: Results of the evaluation. Tests were conducted on 480, 1000 and 2000 test sentences. The length correction column (on/off) shows which scoring method was used, the target language model score (off) as presented in section 2.2 or target language model with length correction factor (on) as presented in section 2.3. The second column shows how many LRLM coverings were found as being non-optimal using the original rule set. The third column shows the values from column 2 as a percentage.

Length correction	No. of non-optimal sentence coverages by LRLM using the original rule-set	% of errors
480 test sentences		
off	4 (all errors)	0.80%
on	4 (all errors)	0.80%
1000 test sentences		
off	4 (all errors)	0.40%
on	4 (all errors)	0.40%
2000 test sentences		
off	12 (all errors)	0.60%
on	8 (all errors)	0.40%

TABLE 2: Results of the evaluation observed on a rule set with the malicious rule added. Tests were conducted on 480, 1000 and 2000 test sentences. The length correction column (on/off) shows which scoring method was used, the target language model score (off) as presented in section 2.2 or target language model with length correction factor (on) as presented in section 2.3. The second column shows the number of LRLM coverings that were found as non-optimal The third column shows how many coverings were erroneously marked as non-optimal, the fourth column presents the value from the third column in percentage, the fifth column shows the number of non-optimal LRLM coverings that use malicious rule. The sixth column shows the % of sentences using the malicious rule pattern that were found by the method (all sentences with malicious rule pattern should ideally be found).

Length correction	No. of non-optimal sentence coverages by LRLM	errors	in %	No. of non-optimal LRLM coverings found to use malicious rule	*% of non-optimal LRLM coverings found to use the malicious rule*
480 test sentences (60 sentences with malicious rule pattern)					
off	35	4	0,80%	31	52%
on	52	4	0,80%	48	80%
1000 test sentences (120 sentences with malicious rule pattern)					
off	65	4	0,40%	61	51%
on	99	4	0,40%	95	79%
2000 test sentences (218 sentences with malicious rule pattern)					
off	109	11	0,55%	98	45%
on	178	10	0,50%	168	77%

the selection of unreasonable rules (such as rules that delete or insert words, which might have erroneously written by the expert or inferred automatically) may be curtailed by using a length modification penalty term. The research proved that the LRLM algorithm in most cases finds the optimal solution. One of the possible reasons is partially due to the fact that rules have been designed with the LRLM model in mind. The metric can be used as a selection criterion for automatically inferred rules like Sanchez-Martnez (2007); Sanchez-Martnez and Hermann (2006).

5 Acknowledgments

This research was funded by the Vice-rectorate for Research, Development and Innovation of the Universitat d'Alacant. Special thanks go to Sergio Ortiz-Rojas for helping us with the implementation. This work is part of PhD Thesis of Jernej Vicic.

References

Oller Carme Armentano and Mikel L. Forcada (2006), Open-source machine translation between small languages: Catalan and Aranese Occitan, in *Strategies for developing machine translation for minority languages (5th SALTMIL workshop on Minority Languages)*, pp. 51–54, (organized in conjunction with LREC 2006 (22-28.05.2006)).

L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer (1989), A Tree-Based Statistical Language Model for Natural Language Speech Recognition., *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(7)*.

P.R. R. Rosenfeld Clarkson (1997), Statistical Language Modeling Using the CMU-Cambridge Toolkit, in *Proceedings ESCA Eurospeech*.

Antonio M. Corbi-Bellot, Forcada Mikel L., Ortiz-Rojas Sergio, Juan Antonio Perez-Ortiz, Gema Ramrez-Sanchez, Felipe Sanchez-Martinez, Inaki Alegria, Aingeru Mayor, and Kepa Sarasola (2005), An open-source shallow-transfer machine translation engine for the Romance languages of Spain, in *Proceedings of the Tenth Conference of the European Association for Machine Translation*, pp. 79–86.

Michel Divay and Anthony J. Vitale (1997), Algorithms for Grapheme-Phoneme Translation for English and French: Applications for Database Searches and Speech Synthesis Algorithms for Grapheme-Phoneme, *Computational Linguistics, Volume 23, Number 4*.

Felipe Sanchez-Martnez and Ney Hermann (2006), Using Alignment Templates to Infer Shallow-Transfer Machine Translation Rules, in Sampo Pyysalo Tapio Salakoski, Filip Ginter and Tapio Pahikkala, editors, *Advances in Natural Language Processing, Proceedings of 5th International Conference on Natural Language Processing FinTAL*, volume 4139 of *Lecture Notes in Computer Science*, pp. 756–767, Springer-Verlag, ISBN 3-540-37334-9, copyright Springer-Verlag.

Felipe; Forcada Mikel L. Sanchez-Martnez (2007), Automatic induction of shallow-transfer rules for open-source machine translation, in Andy Way and Barbara Gawronska, editors, *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*, volume 2007:1, pp. 181–190, Skovde University Studies in Informatics, ISBN 978-91-977095-0-7, ISSN 1653-2325.