An Ngram-based Statistical Machine Translation Decoder

Josep M. Crego, José B. Mariño and Adrià de Gispert

TALP Research Center
Universitat Politècnica de Catalunya, Barcelona
{jmcrego,canton,agispert}@talp.upc.es

Abstract

In this paper we describe MARIE, an Ngram-based statistical machine translation decoder. It is implemented using a beam search strategy, with distortion (or reordering) capabilities.

The underlying translation model is based on an Ngram approach, extended to introduce reordering at the phrase level. The search graph structure is designed to perform very accurate comparisons, what allows for a high level of pruning, improving the decoder efficiency. We report several techniques for efficiently prune out the search space.

The combinatory explosion of the search space derived from the search graph structure is reduced by limiting the number of reorderings a given translation is allowed to perform, and also the maximum distance a word (or a phrase) is allowed to be reordered.

We finally report translation accuracy results on three different translation tasks.

1. Introduction

Statistical Machine Translation (SMT) is thought as a task where each source sentence f_1^J is transformed into (or generates) a target sentence e_1^I, by means of a stochastic process. Thus, translation of a source sentence f_1^J can be formulated as the search of the target sentence e_1^I that maximizes the conditional probability $p(e_1^I|f_1^J)$, which can be rewritten using the Bayes rule as:

$$\arg\max_{e_1^I} \left\{ p(f_1^J|e_1^I) \cdot p(e_1^I) \right\} \quad (1)$$

where $p(f_1^J|e_1^I)$ represents the translation model and $p(e_1^I)$ is the target language model.

This decomposition into two sources is commonly called the source-channel approach. The argmax operation denotes the search problem.

Regarding the translation model, the first statistical MT systems worked at the word level [1]. Among these first systems, we find decoders following different search approaches: optimal A* search [2], integer programming [3], greedy search algorithms [4], [5], [6].

In the last few years, new systems tend to use sequences of words, commonly called phrases, aiming at introducing word context in the translation model. These systems model translation through a log-linear maximum entropy framework [7], that makes it easier to introduce additional models.

This work has been partially supported by the Spanish government, under grant TIC-2002-04447-C02 (Aliado Project), the European Union, under FP6-506738 grant (TC-STAR project) and the Universitat Politècnica de Catalunya, under UPC-RECERCA grant.

$$\arg\max_{e_1^I} \{exp(\sum_i \lambda_i h_i(e,f))\} \quad (2)$$

where feature functions h_i are the system models (translation model, language model, reordering model, ...), and weights λ_i are typically optimized to maximize a scoring function.

Different authors have shown how using phrase distortion models, allowing for modelling phrase discontinuities, outperforms monotonous systems in some language pairs and under reordering restricted conditions. These systems are forced to restrict their distortion abilities because of the high cost in decoding time distortion implies. In [8], the decoding problem with arbitrary word reordering is shown to be NP-complete.

Some decoders with reordering capabilities are described in [9], [10]. Among the last systems, in [11] a free available beam search phrase-based decoder with reordering capabilities is described.

This paper addresses the decoding problem when allowing reordering, under an Ngram-based translation model approach [12]. It is organized as follows. Section 2 introduces the particularities of the SMT modelling that underlies the decoder, section 3 describes the decoder characteristics. Some experiments testing the decoder are reported in section 4. Finally, section 5 presents some conclusions and outlines further research.

2. Ngram Translation Model

According to equation 1, translation can also be seen as a stochastic process that maximizes the joint probability:

$$\arg\max_{e_1^I} \{ p(e_1^I, f_1^J)\} \quad (3)$$

The Translation Model can be thought of a Language Model of bilingual units (here called tuples). These tuples define a monotonous segmentation of the training sentence pairs (f_1^J, e_1^I), into K units $(t_1, ..., t_K)$.

Figure 1 shows an example of tuples extraction from a word to word aligned sentence pair.

The Translation Model is implemented using an Ngram language model (B), (with $N = 3$):

$$p(e,f) = Pr(t_1^K) = \prod_{k=1}^{K} p(t_k \,|\, t_{k-2}, t_{k-1}) \quad (4)$$

Through the log-linear approach of equation 2, the decoder extends the modelling with four feature functions (defined as model probabilities):

- A translation model computed using the IBM1 lexicon probabilities in both directions (I):

$$Pr(t_1^K) = \prod_{i=1}^{K} pM1(e_k \mid f_k)^{\lambda_{st}} pM1(f_k \mid e_k)^{\lambda_{ts}} \quad (5)$$

  where each model weight (λ_{st} and λ_{ts}) can be optimized to maximize a scoring function.

- An Ngram target language model (T), (with $N = 3$):

$$Pr(e_1^I) = \prod_{i=1}^{I} p(e_i \mid e_{i-2}, e_{i-1}) \quad (6)$$

- An standard word penalty used to compensate the decoder preference for the shortest translations (P):

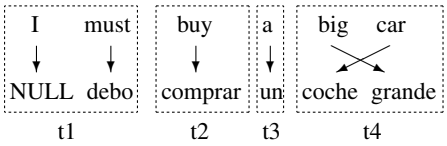
$$Pr(e_1^I) = exp(I) \quad (7)$$

- A word distance-based reordering model (R):

$$Pr(t_1^K) = exp(-\sum_{k=1}^{K} d_k) \quad (8)$$

  where d_k is the distance between the first word of the K^{th} tuple (unit), and the last word +1 of the $K-1^{th}$ tuple (distances are measured in words referring to the units source side).

New models can be introduced extending the sum in equation 2 with additional features.



i must # debo
buy # comprar
a # un
big car # coche grande

Figure 1: *Tuples extraction given a word to word aligned sentence pair. The segmentation of each bilingual sentence pair is carried out by using only the Viterbi word to word alignment, see [13].*

The decoder behaves as a phrase-based SMT decoder, when extracting phrases instead of tuples and using a phrase-based translation model instead of an Ngram-based translation model.

3. Decoder

In SMT decoding, translated sentences are built incrementally from left to right in form of hypotheses, allowing for discontinuities in the source sentence.

To find the optimal path, a Beam search algorithm with pruning is used. Beam search algorithms are widely used in SMT decoding as they offer good possibilities to adjust the trade-off between quality and efficiency.

The search is performed by building partial translations (hypotheses), which are stored in one or several lists. These lists are pruned out according to the accumulated probabilities of their hypotheses. Worst hypotheses with minor probabilities are discarded to make the search feasible.

A main problem when building an SMT decoder is the fact that the search space must be pruned. Some beam-based decoders use only one list, so all hypotheses compete to stay alive after pruning the list. This approach greatly relies on a fair comparison among hypotheses, since decoders tend to bias the search towards those translations with higher probabilities during the first stages (which are not always the best, but just better scored to survive the pruning process).

To avoid this problem, a heuristic function can be used to estimate the future cost of each partial translation. This allows for discarding good hypotheses that do not have an estimated good future path, transforming the beam search into an A* search. A drawback of this solution is the difficulty to find a good heuristic function with acceptable computing cost (see [2], [10] and [11]).

Different lists can be used instead of only one, which helps to fairly compare hypotheses. Commonly, hypotheses are stored in different lists depending on the number of source or target words already covered. Our decoder implements a multi-list approach without heuristic function.

3.1. Core Algorithm and Search Graph Structure

The Beam search algorithm is here outlined:

The search begins adding an initital state, where no source words are covered (translated) yet.

New states (hypotheses) are expanded by adding tuples, translating some source uncovered words. Every hypothesis uses a covering vector indicating wich source words have already been translated (for instance, a hypothesis with a covering vector set to '11000' indicates that the hypothesis has covered the first two words of the source sentence).

Each new expansion is allowed to cover any word positions in the source sentence (restricted to consecutive positions not covered so far), while the tuple target words are added sequentially to the target sentence. This way the target sentence is built monotonously.

The cost of each new state is the result of adding the cost of the predecessor state to the cost derived from the different features used as models.

Table 1 shows the information contained in each state.

A link to the predecessor state
The last N_1 tuples
The last N_2 target words
Covering vector
Positions covered by the last tuple
The cost so far

Table 1: *Every hypothesis is represented by six fields. N_i are set according to the Ngrams used for the translation (B) and target (T) models.*

Figure 2 shows an example of the search graph structure. It can be decomposed into three levels:

- Hypotheses. In figure 2, represented using '*'.
- Lists. In figure 2, the boxes with a tag corresponding to its covering vector. Every list contains an ordered set of hypotheses (all the hypotheses in a list have translated the same words of the source sentence).

- Groups (of lists). In figure 2, delimited using dotted lines. Every group contains an ordered set of lists, corresponding to the lists of hypotheses covering the same number of source words (to order the lists in one group the cost of their best hypothesis is used). When the search is restricted to monotonous translations, only one list is allowed on each group of lists.

The search loops expanding available hypotheses. The expansion proceeds incrementally starting in the group of lists covering 1 source word, ending with the group of lists covering $J-1$ source words (J is the size in words of the source sentence).

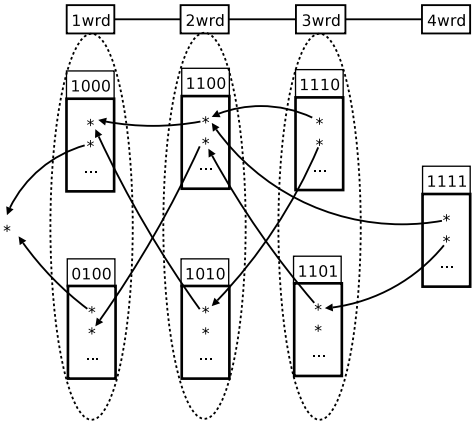


Figure 2: *Search graph corresponding to a source sentence with four words. The graph is constrained by a distortion limit ($m = 1$), and a maximum number of reorderings per sentence ($j = 1$). Details of constraints are given in following sections.*

Among the hypotheses stored in the last group (covering the whole words of the source sentence), the hypothesis with the lowest cost (highest probability) is selected and printed out.

3.2. Pruning the search

To reduce the search space, those hypotheses that agree on the last N_1 tuples, the covering vector and the last N_2 target words are recombined (N_1 and N_2 are set according to the Ngrams used for the translation (B) and target (T) models), keeping only the best scored hypothesis. Recombination is a risk-free way to prune the search space, as those states recombined could not be distinguished in future steps of the search.

When expanding each new list, only the best hypotheses are expanded: those hypotheses with best scores (histogram pruning, b); with a score within a margin (t) given the best score in the list (threshold pruning).

The same pruning strategies are used when expanding the lists of each group: those lists with best scores (histogram pruning, B); with a score within a margin (T) given the best score in the group (threshold pruning). To score a list, the cost of its best hypothesis is used.

When allowing for reordering, the decoder suffers from the apparition of a huge amount of lists (an upper bound is 2^J, where J is the number words of the source sentence). Not only the states, but the lists of the graph structure need to be pruned out. In order to reduce the number of lists, two constraints are used, detailed in the next subsection.

3.3. Reordering Capabilities

A reordering strategy is also key to avoid the search combinatory explosion problem when allowing for distortion. In [14], a comparison among different reordering constraints is shown (namely ITG and IBM constraints).

Our decoder implements two distortion constraints in order to reduce the lists of the graph structure:

- A distortion limit (m). A source word (phrase or tuple) is only allowed to be reordered if it does not exceed a distortion limit, measured in words.
- A reorderings limit (j). Any translation path is only allowed to perform j reordering jumps.

4. Experiments

In this section, different experiments are reported to test the performance of the decoder in terms of translation accuracy (using BLEU and WER). The first subsections explain the framework on which the experiments were performed.

4.1. Corpus

Experiments have been carried out using two databases: the TC-Star[1] corpus (Spanish-English) and the IWSLT 2004 BTEC[2] corpus (Chinese-English). Results with TC-Star are reported using the text version (FTE) of the corpus used in the TC-Star SLT first evaluation.

Tables 2 and 3 show the main statistics of the used data, namely number of sentences, words, vocabulary, and mean sentence lengths for each language.

Set	Lng	Sent	Wrds	Vocab	Mean
trn	eng	1,223,443	33,379,333	104,975	27.3
	spa		34,794,006	168,685	28.4
tst	eng	1,008	26,002	3,208	25.8
	spa		25,658	3,937	25.4

Table 2: *TC-Star Corpus. Two references are available for the test side of both languages.*

Set	Lng	Sent	Words	Vocab	Mean
trn	eng	20,000	188,935	8,191	9.4
	chi		182,904	7,643	9.1
tst	eng	500	4,187	2,496	8.4
	chi		3,794	893	7.6

Table 3: *BTEC Corpus. For the English test side of the corpus, 16 different references are available.*

4.2. Training the Models

We used giza++ to perform the word alignment of the whole training corpus, and refined the links by the union of both alignment directions [10]. Afterwards we segmented the bilingual sentence pairs of the training set, extracting translation units (tuples) using the extract-tuples method described in [13].

[1] www.tc-star.org
[2] www.slt.atr.jp/IWSLT2004

The vocabulary of tuples was pruned out using different pruning techniques, mainly based on limiting bilingual units to those: a) consisting on the N best translation candidates; b) occurring a minimum number of times in the train set; c) not exceeding a size threshold (number of words on each side of the unit); d) not exceeding a fertility threshold (difference in source-side and target-side number of words).

This pruning benefits the estimation of the translation model and improves the efficiency of the search.

To train the Ngram models, we used the SRILM toolkit [15]. The type of discounting algorithm used was the modified Kneser-Ney combining higher and lower order estimates via interpolation.

The weights λ_i for the log-linear combination of models were set in order to minimize the BLEU score [16], using the simplex algorithm.

4.3. Results

Table 4 shows the results obtained by the decoder on the Chinese to English translation direction using the BTEC corpus, and both translation directions results using the TC-Star corpus.

Task	BLEU	WER
chi2eng mon	0.331	51.5
chi2eng reord	0.363	49.68
spa2eng mon	0.545	34.4
eng2spa mon	0.472	41.4

Table 4: *The first two rows show the results of the Chinese to English translation task (in the first row the decoder performed monotonous translations, in the second row reordering was allowed). The last two rows show the decoder results performing monotonous translations for both translation directions.*

The use of distortion is only recommended when required by the language pairs. For instance, the Spanish-English distortion requirements are limited to short-distance reorderings, which are well captured within the bilingual units (tuples or phrases).

5. Conclusions and Further Work

We described MARIE[3], a decoder for Ngram-based Statistical Machine Translation systems with reordering capabilities, which allows to easily incorporate new models by using a log-linear approach.

Despite the preference for monotonous translations of the Ngram-Based translation models, results show how reordering can also be applied.

The search graph structure of the decoder guides to very accurate hypotheses comparisons, improving the decoder efficiency through very high levels of pruning.

The combinatory explosion of the search space that distortion implies is reduced by the use of reordering constraints.

Further work is envisaged to improve the decoder performance by better pruning out the search space, in terms of reordering contraints.

[3]The Decoder can be free downloaded from the internet address: *http://gps-tsc.upc.es/veu/soft/soft/MARIE*.

6. References

[1] P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, 1990.

[2] F. Och, N. Ueffing, and H. Ney, "An efficient A* search algorithm for statistical machine translation," *Data-Driven Machine Translation Workshop, 39th Annual Meeeting of the Association for Computational Linguistics (ACL)*, pp. 55–62, July 2001.

[3] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada, "Fast decoding and optimal decoding for machine translation," *39th Annual Meeting of the Association for Computational Linguistics*, pp. 228–235, July 2001.

[4] U. Germann, "Greedy decoding for statistical machine translation in almost linear time," *HLT-NAACL-2003*, May 2003.

[5] A. Berger, P. Brown, S. Della Pietra, V. Della Pietra, and J. Gillet, "The candide system for machine translation," *Proceedings of the Arpa Workshop on Human Language Technology*, March 1994.

[6] Y. Wang and A. Waibel, "Fast decoding for statistical machine translation." in *ICSLP98*, December 1998.

[7] A. Berger, S. Della Pietra, and V. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–72, March 1996.

[8] K. Knight, "Decoding complexity in word replacement translation models," *Computational Linguistics*, vol. 26, no. 2, pp. 607–615, 1999.

[9] C. Tillmann and H. Ney, "Word re-ordering and dp-based search in statistical machine translation," *Proc. of the 18th Int. Conf. on Computational Linguistics, COLING'00*, pp. 850–856, July 2000.

[10] F. Och and H. Ney, "The alignment template approach to statistical machine translation," *Computational Linguistics*, vol. 30, no. 4, pp. 417–449, December 2004.

[11] P. Koehn, "Pharaoh: a beam search decoder for phrase-based statistical machine translation models," *Proc. of the 6th Conf. of the Association for Machine Translation in the Americas*, pp. 115–124, October 2004.

[12] A. de Gispert and J. Mariño, "Using X-grams for speech-to-speech translation," *Proc. of the 7th Int. Conf. on Spoken Language Processing, ICSLP'02*, September 2002.

[13] J. Crego, J. Mariño, and A. de Gispert, "Finite-state-based and phrase-based statistical machine translation," *Proc. of the 8th Int. Conf. on Spoken Language Processing, ICSLP'04*, pp. 37–40, October 2004.

[14] R. Zens, F. Och, and H. Ney, "Improvements in phrase-based statistical machine translation," *Proc. of the Human Language Technology Conference, HLT-NAACL'2004*, pp. 257–264, May 2004.

[15] A. Stolcke, "Srilm - an extensible language modeling toolkit," *Proc. of the 7th Int. Conf. on Spoken Language Processing, ICSLP'02*, September 2002.

[16] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "Bleu: a method for automatic evaluation of machine translation," IBM Research Division, Thomas J. Watson Research Center, Tech. Rep. RC22176 (W0109-022), 2001.