MISTRAL: A Lattice Translation System for IWSLT 2007

Alexandre Patry, Philippe Langlais

RALI – DIRO
Université de Montréal
{patryale,felipe}@iro.umontreal.ca

Frédéric Béchet

LIA
University of Avignon
frederic.bechet@univ-avignon.fr

Abstract

This paper describes MISTRAL, the lattice translation system that we developed for the Italian-English track of the International Workshop on Spoken Language Translation 2007.

MISTRAL is a discriminative phrase-based system that translates a source word lattice in two passes. The first pass extracts a list of top ranked sentence pairs from the lattice and the second pass rescores this list with more complex features. Our experiments show that our system, when translating pruned lattices, is at least as good as a fair baseline that translates the first ranked sentences returned by a speech recognition system.

1. Introduction

While in textual translation the sentences to translate are known for sure, in spoken language translation they must be reconstructed from an audio signal. Instead of translating the raw audio signal, spoken language translation systems usually start from the output of an automatic speech recognition system, which usually takes the form of a word lattice.

A straightforward way to translate a word lattice is to feed a textual translation system with the most probable sentence. To overcome some recognition errors, one can instead translate a set of *n* top ranked sentences extracted from the lattice and select the best translation among the one returned [1, 2]. Finally, a third approach is to translate directly the lattice with a specialized decoder tightly coupled with the speech recognition system [3, 4, 5, 6, 7].

In this paper, we present MISTRAL (Monotone yet Imperfect Statistical TRAnslation of Lattices), a discriminative phrase-based system that translates lattices in two passes. The first pass uses a beam-search decoder to extract a N-Best list of source sentences and their translations from the lattice and the second pass rescores this list with more complex feature functions.

Different word-based lattice translation systems are presented in [3, 4, 5]. A generative phrase-based system is described in [6]. This system translates the lattice with a sequence of weighted finite state machines applied in cascade while our system is dedicated and traverses the lattice only once. In [7], the authors present a discriminative phrase-based system that translates confusion networks and offers an elegant solution to word reordering.

The remaining of the article is organized as follow. The next section presents the theoretical framework of spoken language translation. We describe our decoder in section 3 and evaluate our complete system in section 4. We finally conclude in section 5.

2. Lattice Translation

In textual translation, a source sentence $(\mathbf{f})$ is known for a fact, we thus seek to resolve:

$$\mathbf{e}^\star = \operatorname*{argmax}_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) \quad (1)$$

Since in spoken language translation only a word lattice is known $(\mathbf{o})$, the equation to resolve thus becomes:

$$\mathbf{e}^\star = \operatorname*{argmax}_{\mathbf{e}} \sum_{\mathbf{f}} \Pr(\mathbf{e}, \mathbf{f}|\mathbf{o}) \quad (2)$$

If we pose the assumption that the target sentence and the word lattice are conditionally independent given the source sentence, Eq. (2) can be simplified to:

$$\mathbf{e}^\star = \operatorname*{argmax}_{\mathbf{e}} \sum_{\mathbf{f}} \Pr(\mathbf{e}|\mathbf{f}) \Pr(\mathbf{f}|\mathbf{o}) \quad (3)$$

Most systems solving Eq. (3) proceed in two steps. They first extract a N-Best list of source sentences from the lattice and then use a textual translation system (Eq. (1)) to translate them.

In this work, we describe a monotone decoder using a discriminative phrase-based model to estimate Eq. (2) under the so-called maximum approximation:

$$\hat{\mathbf{e}} = \operatorname*{argmax}_{\mathbf{e}} \max_{\mathbf{f}} \Pr(\mathbf{e}, \mathbf{f}|\mathbf{o}) \quad (4)$$

3. Decoder

Our decoder uses the following process to generate a complete translation from a word lattice:

1. Initialise n to the start node and the translation to an empty sentence.

2. While n is not the end node

(a) Select a path p that starts at n and collect its source words in w.

(b) Select an entry $\langle s, t\rangle$ in the translation table that has w as its source phrase.

(c) Append t at the end of the translation and set n to the destination of p.

This leaves us with two problems: evaluating $\Pr(\mathbf{e}, \mathbf{f}|\mathbf{o})$ and exploring the search space efficiently.

3.1. Model

To approximate the value of $\Pr(\mathbf{e}, \mathbf{f}|\mathbf{o})$, we use an exponential model:

$$p_\lambda(\mathbf{e}, \mathbf{f}|\mathbf{o}) = \frac{1}{Z(\mathbf{o})} \exp\left(\sum_{r=1}^{R} \lambda_r h_r(\mathbf{e}, \mathbf{f}, \mathbf{o})\right) \quad (5)$$

where $Z(\cdot)$ is a normalization factor, $h_r(\cdot,\cdot,\cdot)$ are feature functions returning real values and λ_r are free parameters weighting the feature functions. Using Eq. (5), we can simplify Eq. (4) to:

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \max_{\mathbf{f}} \sum_{r=1}^{R} \lambda_r h_r(\mathbf{e}, \mathbf{f}, \mathbf{o}) \quad (6)$$

The set of feature functions and the algorithm that was used to tune their weights are detailed in section 3.4 and 4.2.

3.2. First Pass

Our decoder uses a beam-search algorithm to select the subset of the search space to explore. This algorithm groups similar incomplete translations together and explores the search space one group at a time, pruning each group independently of the others.

In textual translation, it is common to group translations by the count of source words translated so far. Because this count varies from one translation to another when working with a lattice, we group the translations according to the length of the audio signal they cover. We thus divide the audio signal in slices of equal length and associate a stack with each slice.

The stacks are explored in chronological order. Before a stack is explored, it is pruned and the remaining hypotheses are moved to a temporary stack. Once the hypotheses of the temporary stack are expended, we verify if the stack of the current time slice contains new hypotheses (which means that at least one partial hypothesis was in the same time slice before and after it has been updated). If it is the case, we prune and explore it again, else we explore the next stack.

In this work, we empirically set the length of a slice to one tenth of a second. Short slice lengths induce more stacks and thus less pruning. On the other hand, long slice lengths may be harmful without a good heuristic to approximate the cost of completion of a hypothesis.

The pruning of a stack is done in two steps. In the first step, we only keep the 50 best translations. The remaining translations of the group are then checked for recombination[1]. A set of translations can be recombined if they correspond to the same node in the lattice, their last two source words are the same (because of the source trigram) and their last two target words are the same (because of the target trigram).

When the decoder encounters a node that is followed only by unknown words, it considers that those words are translated by themselves.

3.3. Rescoring

The first pass is used to extract a N-Best list of 500 translations. This list is then reranked using an exponential model with more sophisticated feature functions than in the first pass.

3.4. Tuning

The weights of the first pass are tuned with the following algorithm:

1. Initialise the weights.
2. Extract an N-Best list of 500 translations from each lattice of the tuning corpus.
3. Tune the weights to optimise BLEU on those N-Best lists using the downhill simplex algorithm [8].
4. If the weights were updated, go to 2.

To avoid overfitting, we test the weights obtained at each iteration on a validation corpus. The weights that scored best on this validation corpus are kept.

The first pass is then used to extract the 500 best translations from a new corpus. Rescoring is optimised on this list with the downhill simplex algorithm.

4. Experiments

4.1. Data

The corpus used for the shared task is composed of transcriptions of spontaneous conversations in the travel domain. It is divided in train (TRAIN) and development (DEV) sections containing respectively 19,722 and 996 sentence pairs. We also trained our models on the Italian-English section of the proceedings of the European Parliament (EUROPARL), which contains more than 928,000 sentence pairs [9].

The two corpora were converted to lower case and their punctuation marks were removed. The lattices of DEV, which were originally scored with a language model and an acoustic model, were augmented with posterior probabilities using the *lattice-tool* utility [10].

[1]Recombination is done after pruning because it is slower to execute. We did not observe a degradation in translation quality by doing so.

We then trained a translation table on TRAIN and another on EUROPARL. To do so, we used a script that was provided for the shared task of the NAACL 2006 Workshop on Statistical Machine Translation [11]. This script uses the heuristics described in [12] to extract phrase pairs from a word alignment that was first produced by GIZA++ [13]. It then computes five scores for each phrase pair: the posterior probability in each translation direction, the lexical probability in each translation direction and a constant phrase penalty.

Knowing that the corpora contain many dates and numbers, we manually created a third translation table containing 122 entries translating days, months and numbers[2].

The first 300 sentences of DEV were used to tune the coefficients of the first pass (200 for tuning and 100 for validation, see section 3.4) and the 300 following sentences to tune the coefficients of the rescoring pass. The remaining 396 sentences were used to evaluate different system configurations.

4.2. System

The fifteen following feature functions were used in the first pass:

- The posterior probability of the lattice path.
- Two Italian trigrams trained respectively on the TRAIN and EUROPARL corpora.
- Two English trigrams trained respectively on the TRAIN and EUROPARL corpora.
- The number of words in the source sentence.
- The number of words in the target sentence.
- The sum of the logarithm of each of the five scores in the phrase table.
- Three binary functions associating a pair of phrases with its translation tables.

and the eight following feature functions were added for the rescoring pass:

- Two Italian 4-grams trained respectively on the TRAIN and EUROPARL corpora.
- Two English 4-grams trained respectively on the TRAIN and EUROPARL corpora.
- The lexical probabilities of the complete translations according to four models trained on TRAIN and EUROPARL in both translation directions.

When tuning the first pass, all the weights were initialised to 0.1, except the weight of the posterior probability of the lattice path, which was initialized to 10. The weights for rescoring were initialized to those of the first pass or to 0.1 if their feature function was not used in the first pass.

[2]Numbers from zero to 100, 1000 and one million.

System	First Pass		Rescoring	
	WER	BLEU	WER	BLEU
Ref	0	20.09	0	21.27
Ref (MOSES$_m$)	0	20.91	-	-
Ref (MOSES$_d$)	0	20.39	-	-
1 best	11.90	17.97	11.90	19.37
1 best (MOSES$_m$)	11.90	18.99	-	-
1 best (MOSES$_d$)	11.90	19.03	-	-
BLEU	12.04	17.08	12.07	19.24
WER and BLEU	11.81	17.58	11.87	18.93
BLEU, pruned	10.96	19.21	11.04	20.28

Table 1: Word-error rates of speech recognition and BLEU scores of translations. Values are multiplied by 100 throughout the article to improve readability.

4.3. Results

The source sentences were evaluated with word-error rate (WER) and the translations with BLEU [14][3]. The bottom section of Table 1 presents the results of different configurations of MISTRAL.

To estimate an upper bound score for MISTRAL, we tuned and tested it on the reference corpus, which yielded a BLEU score of 21.27. We also tuned a baseline system on the best sentence returned by the speech recognition system and obtained a BLEU score of 19.37.

In a first experiment, we translated the lattice with a system that was tuned on BLEU. This resulted in a system that had a relative degradation of less than 1% in BLEU relative to the baseline.

In a second experiment, we tried to reduce the WER by optimizing our system on the harmonic mean of the word recognition rate ($1 - \text{WER}$) and BLEU. It did indeed improved the WER, but at the expense of BLEU.

Finally, we ran a system optimized on BLEU, as in the first experiment, but this time we pruned the lattices. We used the *lattice-tool* utility [10] to remove all the edges whose posterior probability was less than one percent of the posterior probability of the best path. Doing so, we were able to get a small improvement over the baseline of about 5% relative in BLEU.

In the original lattices, there were 360 times more words than the number of spoken words. Once the lattices were pruned, this factor went down to 2.7 and the decoding time was divided by seven.

To see if the low BLEU figures were due to an error in our implementation, we compared our system with a monotone (MOSES$_m$) and a non-monotone (MOSES$_d$) MOSES decoders [15]. Those two decoders were tuned on the first 300

[3]WER was computed with an in-house script and BLEU with the *multi-bleu.perl* script available at `http://www.statmt.org/wmt06/shared-task/multi-bleu.perl`

Task	MISTRAL tuning		MOSES$_m$ tuning	
	MISTRAL	MOSES$_m$	MISTRAL	MOSES$_m$
Ref	20.09	20.83	19.52	20.91
1 best	17.97	18.62	17.22	18.99

Table 2: Performances of MOSES and MISTRAL when their configurations are exchanged.

System	BLEU			
	Before	C	P	C + P
Official run	21.03	18.66	16.12	13.90
BLEU, pruned	23.81	20.75	17.60	16.17

Table 3: Results before and after case (C) and punctuation (P) have been restored.

sentences of DEV using the *mert-moses.pl* utility provided with MOSES (see Table 1).

Because BLEU scores of MOSES were low as well, we think that the poor performance of MISTRAL are mainly due to the models. Indeed, the corpora that were used to train the translation tables are either small (TRAIN) or out of domain (EUROPARL) and we took no special care to cope with that.

When looking at Table 1, we see that MOSES is systematically better than MISTRAL. To verify if this difference is due to tuning, we ran MISTRAL on MOSES$_d$ configuration and vice versa. Looking at Table 2, we see the difference is due to our implementation.

4.4. Shared Task

The shared task was evaluated on a test corpus of 724 sentences. Because the evaluation was case and punctuation sensitive, we had to post-process the output of our system.

We treated the capitalization problem as a disambiguation task where each word is ambiguously capitalized or not. To do so, we trained a trigram on the capitalized TRAIN corpus and then used the *disambig* utility [10].

To restore the punctuations, we used a naive Bayes classifier looking at the first word of a sentence to decide whether it should end with a period or a question mark. This algorithm is very limited because it does not restore internal punctuations like comas and has no way to detect when an utterance is made of many sentences (e.g. “Globetrotter Travel. Good morning.”). But we did not have time to investigate this complex problem further.

Table 3 presents the performance of our official run after all post-processing steps. BLEU brevity penalty is really penalizing our system because we added only one punctuation per sentence while the test corpus contained an average of 2.4 punctuations per sentence. If the punctuation marks would have no influence on the brevity penalty, the final BLEU score of our official run would have been 15.87 instead of 13.90.

Our system was still in development when we submitted our official translations. The results of our final system translating pruned lattices are presented in the second row of Table 3.

5. Conclusion

We presented and evaluated MISTRAL, the spoken language translation system that we developed for IWSLT 2007. MISTRAL is a phrase-based system dedicated at translating word lattices.

Our system is deceiving in two ways. Its scores are low in general and it does not clearly surpass a fair baseline. Given that an external system produced translations of low quality when trained on the same data, we think that low scores are due to our models, which were not adapted to the task. We are interested in testing MISTRAL on more data to see if it could do better than the baseline.

On the other hand, MISTRAL is still young and there is room for improvement. We will investigate why MOSES, on the same configuration, outperforms our system. We also plan to work on the integration of new features and the support for non-monotone translations.

6. References

[1] R. Zhang, G. Kikui, H. Yamamoto, T. Watanabe, F. Soong, and W. K. Lo, “A unified approach in speech-to-speech translation: integrating features of speech recognition and machine translation,” in *COLING ’04: Proceedings of the 20th international conference on Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2004, p. 1168.

[2] H. Quan, M. Federico, and C. M., “Integrated n-best re-ranking for spoken language translation,” in *Interspeech 2005 - Eurospeech - 9th European Conference on Speech Communication and Technology*, 2005.

[3] S. Saleem, S.-C. Jou, S. Vogel, and T. Schultz, “Using word lattice information for a tighter coupling in speech translation systems,” in *Proc. ICSLP*, Jeju Island, Korea, Oct 2004.

[4] E. Matusov, S. Kanthak, and H. Ney, “On the integration of speech recognition and statistical machine translation,” in *Proceedings of the 9th European Conference on Speech Communication and Technology (Interspeech)*, September 2005.

[5] R. Zhang, G. Kikui, H. Yamamoto, and W.-K. Lo, “A decoding algorithm for word lattice translation in speech translation,” in *Proceedings of 2005 International Workshop on Spoken Language Translation*, 2005.

[6] L. Mathias and W. Byrne, “Statistical phrase-based speech translation,” in *IEEE Conference on Acoustics, Speech and Signal Processing*, 2006.

[7] N. Bertoldi, R. Zens, and M. Federico, "Speech translation by confusion network decoding," in *Proceedings on the 32nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, April 2007.

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*. New York, NY, USA: Cambridge University Press, 1992.

[9] P. Koehn, "Europarl: A parallel corpus for statistical machine translation," in *2nd. Workshop on EBMT of MT-Summit X*, 2005.

[10] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *Proceedings of ICSLP*, Denver, Colorado, Sept 2002.

[11] P. Koehn and C. Monz, "Manual and automatic evaluation of machine translation between european languages," in *Proceedings on the Workshop on Statistical Machine Translation*. New York City: Association for Computational Linguistics, June 2006, pp. 102–121.

[12] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. Morristown, NJ, USA: Association for Computational Linguistics, 2003, pp. 48–54.

[13] F. J. Och and H. Ney, "Improved statistical alignment models," in *Conference of the Association for Computational Linguistic (ACL)*, Hongkong, China, October 2000, pp. 440–447.

[14] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2001, pp. 311–318.

[15] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*. Prague, Czech Republic: Association for Computational Linguistics, June 2007, pp. 177–180.