ASR Word Lattice Translation with Exhaustive Reordering is Possible

Evgeny Matusov, Björn Hoffmeister, and Hermann Ney

Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6 - Computer Science Department
RWTH Aachen University, D-52056 Aachen, Germany
{matusov,hoffmeister,ney}@cs.rwth-aachen.de

Abstract
This paper shows how ASR word lattices can be translated even when exhaustive reordering is required for good translation quality. We propose a method for labeling lattice word hypotheses with position information derived from a confusion network (CN). This information is effectively used in the statistical phrase-based machine translation (MT) search to reduce its complexity, which makes even long-range reordering possible. The proposed method has the benefits of the CN-based MT without having its theoretical drawbacks.

We compare our novel search with the search based on single-best recognition output and on confusion networks. We obtain significant improvements on two translation tasks over the single-best search and gain over the CN search on a task requiring heavy reordering.

Index Terms: speech translation, confusion network

1. Introduction
The goal of this work is to show how multiple automatic speech recognition (ASR) sentence hypotheses represented in a word lattice can be used to improve speech translation quality even when a full-scale phrase reordering has to be performed in the translation process.

The idea of coupling ASR and MT by considering multiple ASR hypotheses has been actively researched in the recent years. A good overview of the theory for integrated speech translation was presented in [9]. Since then, improvements in translation quality were reported when the alternative ASR hypotheses were represented in the form of N-best lists, word lattices [11, 7] or confusion networks [2].

A theoretically well-grounded way of coupling ASR and MT is to use word lattices which represent a large portion of the true ASR search space. The word hypotheses in these lattices are provided with acoustic and source language model (LM) scores. In [8] it was shown how these scores can be effectively included in the log-linear phrase-based MT model framework. However, in the previous work the lattice-based search was either monotonic [1] or with a very limited word or phrase reordering, e.g. by allowing the search algorithm to skip one word or phrase and translate it at a later stage [13, 7]. The reason for this is the high computational complexity.

In order to solve the reordering problem, translation of confusion networks has been proposed [2]. In a confusion network (CN) the recognized word hypotheses are aligned to specific positions, or slots. The structure of a CN allows for an MT search that is similar to the established search for text input, where translation hypotheses with the same cardinality (number of already covered slots) are expanded under certain reordering constraints. Long-range reordering becomes possible. Given a clear definition of the slots in a CN, other, more sophisticated types of decoding e.g. using hierarchical phrases (CYK-style search) can also be performed [3].

However, the CN representation has its drawbacks. Extra paths may be introduced which were not part of the search space. As a consequence, current MT models may often be too weak to differentiate between “good” and “bad” paths in the confusion network, especially if reordering is involved. Furthermore, the original ASR scores can not be used as features.

In this paper, we combine the advantages of a cardinality-based search with the advantages of using the original word lattice. We introduce an approach for labeling a general lattice with slot information as in a CN and then use this information in MT search. We also modify the search to keep track of the already covered lattice states in order to ensure that only real lattice paths are covered. Although our approach resembles the one described in [3], it was developed independently. The crucial difference to that work is that we apply the search to real ASR lattices (vs. lattices with e.g. Chinese word segmentation alternatives in [3]) and that we use slots instead of the whole set of lattice states to guide the cardinality-synchronous search.

We show experimentally that the computational complexity of the proposed lattice search is manageable even with extensive reordering, and that this search yields translations that are significantly better in terms of BLEU and TER than translations of single-best ASR output and comparable or better than CN-based translations. To our best knowledge, this is the first published experimental comparison between translations of word lattices and confusion networks.

2. Coupling ASR and Statistical MT
Following [8] and [3], we formulate the problem of speech translation as the problem of maximizing the posterior probability $P_T(e^1 | x^1)$ of the target language sentence $e^1$ which is the translation of an utterance represented by the acoustic vectors $x^1$. The source words $f^1_j$ hypothesized by an ASR system are introduced as a hidden variable.

$$e^1 \ = \ \underset{e^1}{\text{argmax}} \ \sum_{f^1_j} P_T(e^1 | f^1_j | x^1)$$  \ \ \ \ \ \ \ (1)$$

To solve the problem in Eq. 1, we employ a log-linear model for $P_T(e^1 | f^1_j | x^1)$ and approximate the sum by a maximum over all speech recognition sentence hypotheses $f^1_j$ represented in a word lattice.

$$\hat{e}^1 \ = \ \underset{\hat{e}^1}{\text{argmax}} \ \max_{f^1_j} \ \exp \left\{ \sum_{m=1}^{M} \lambda_m h_m(e^1, f^1_j, x^1) \right\}$$  \ \ \ \ \ \ \ (2)$$

The features $h_m$ we use in the log-linear model in Eq. 2 are the acoustic model and the source LM probabilities, as well as the target LM probabilities and translation probabilities of the word and phrase level as described in [8]. The scaling factors $\lambda_m$ are optimized on a development set for translation quality as measured by an automatic error measure.

---

This is also the main difference to the work of [11].
3. Search

The optimization problem in Eq. 2 is solved using a modified version of the phrase-based search described in [14]. In that work the search is performed using only the single-best ASR hypothesis \( f^* \) as follows.

3.1. Search Using Single-best ASR Output

The search proceeds synchronously with the cardinality of the already translated source slots \( c \), and partial target translations are created from left to right in a monotonic way. With each hypothesis a coverage set \( C \subset \{1, \ldots, j\} \) is associated, it holds \( c \in C \). With each hypothesis with cardinality \( c \), the decoder selects a range of source slots \( j'_1, \ldots, j'_n \) for which a target phrase translation exists and extends the current hypothesis with this phrase. The extension is valid if there is no overlap with the already translated slots, i.e. \( C \cap \{j'_1, \ldots, j'_n\} = \emptyset \).

In case of monotonic search, the next extension must start with slot \( j' \) that is the next left slot of the last slot in \( C \); in case of non-monotonic search, this can be any slot that does not violate reordering constraints. These constraints are used to either make the search more efficient or are motivated by linguistic reordering rules. The constraints are computed from the coverage vector \( C \), the latest covered slot, and the first candidate slot \( j' \). The search is performed using dynamic programming. Histogram pruning is applied separately to lexical hypotheses for each coverage and coverage hypotheses for each cardinality.

This type of search is useful to model word and phrase reordering. However, it is not directly applicable to word lattices. The problem is that a lattice contains ASR hypotheses of different lengths so that a coverage vector can not be defined based on a single path in the lattice. To overcome this problem, [3] and [11] use the whole set of lattice states to define the coverage sets. In contrast, we modify the lattice by labeling each arc (word hypothesis) with the slot information, see Section 4. After this modification, for each word hypothesis in the lattice we can reach the lattice state from which the nearest covered slot to the left of \( j' \) is reachable from the lattice state which corresponds to the nearest already covered slot to the left of \( j' \).

3.2. Phrase Matching for Word Lattices

As pointed out in [3], the number of source phrases which can be extracted from a word lattice is exponential in the number of lattice nodes. However, there exists an efficient phrase-matching procedure for matching the possible translations of every span in a lattice as described e.g. in [2]. This implementation is based on a prefix-tree representation for the source phrases in the phrase table. From every lattice state, the translation alternatives are generated incrementally over the span length, until a leaf of the prefix tree is reached, or the source phrase of the span is not found in the prefix tree. The intermediate expansions are stored on a stack. In practice, the phrase matching procedure is computationally inexpensive since translation options exist only for very few spans of length \( \geq 2 \).

For each lattice span with an existing translation, we extract phrase translation candidates with the following information: source and target word sequence, the set of covered slots, as well as the ids of the beginning and the end state of the span. In case of gaps in the slot enumeration, we assume that one word covers two or more slots and add the right number of slots to the coverage set. By saving the span state boundaries we separate translation candidates which have the same coverage sets and target phrases, but which arise from different lattice spans. For example, given the lattice in Figure 1 we differentiate between two candidates translating \( \text{questo in} \), one starting at state 1 and ending at state 5, and the other one starting at state 2 and ending at state 5. This separation will be of value in the search, as described in the next subsection.

3.3. Search Modification for Word Lattices

In the search, when extending a hypothesis with a phrase translation candidate, we have to ensure that only valid lattice paths are followed. Considering a possible extension covering slots \( j'_1, \ldots, j'_n \) with start and end states \( n' \) and \( n'' \), we make sure that:

- \( n' \) is reachable from the lattice state which corresponds to the nearest already covered slot to the left of \( j' \)
- from \( n'' \) we can reach the lattice state from which the nearest covered slot to the right of \( j'' \) has been translated.

In case no slot has yet been covered to the right of \( j'' \), we only have to check the first condition since there is always a path to a final state of the lattice. Similarly, in case no slot has been covered to the left of \( j' \), only the second condition must be checked. The constraint is explained on the example lattice in Figure 1. Here, given a possible translation candidate translating \( \text{aereo} \) at slot 4 from state 5 to 8, we can use this candidate only if there is a path to state 5 from the nearest covered slot to the left of slot 4 (e.g. to state 5 with the source word \( \text{in} \) in case of monotonic translation). This condition is not fulfilled for a hypothesis translating \( \text{e} \) and ending at state 6, since there is no path from state 6 to state 5. Indeed, the translation of \( \text{aereo} \) should not follow the translation of \( \text{e} \) since the two words are not on the same path. Similarly, if e.g. due to reordering the word \( \text{di} \) between states 7 and 8 has already been translated, the translation can not continue with \( \text{aereo} \) since there is no path from state 8 to state 7.

The procedure that checks the above constraints can be efficiently implemented. The test whether there exists a path between two arbitrary lattice states is performed once in advance for all state pairs using the all-pairs shortest path algorithm.

If the constraints are fulfilled, the hypothesis is expanded. The costs of the phrase translation extension are added to the total costs of the currently considered hypothesis; they include the source LM and acoustic model costs for the words in the span covered by the extension, with corresponding scaling factors.

3.4. Difference to Confusion Networks

The presented lattice translation approach is a generalization of the CN translation presented e.g. in [2]. A CN is a special case
of a lattice, in which each path from the start state to the end state goes through all the other states. As a consequence, a CN derived from a lattice contains all the paths of the lattice and — normally many — extra paths which have to be searched in the MT process. In our experiments we find that translating a general lattice with the proposed approach is faster than translating a confusion network created from the same lattice. Moreover, translating the original lattices may be also a better option in terms of translation quality (see Section 5).

4. Preparing Word Lattices for MT Search

Lattices produced by ASR systems are usually highly redundant w.r.t. the information required by an MT system. They include symbols representing non-word events like noise, and they store time information for each word. By omitting all the information not relevant for MT and applying standard graph algorithms the lattice size can be drastically reduced as described in [8]. The compression of the lattices significantly reduces MT runtime without changing the result.

4.1. Inserting Slot Information

For the translation approach described we have to associate a slot label with each arc in the lattice. To compute the slot labels we collapse the lattice into a confusion network and enumerate the CN slots. Each original lattice arc is now associated with a CN slot and gets the slot number as its label.

The translation runtime depends on the number of slots in the CN. In order to reduce it, we remove slots containing only ε-arcs. Empty words from the lattice are not considered in translation and hence ε-arcs do not need a slot label. In ASR, confusion networks are a common approach for computing word posterior probabilities. For a slot S and word f we calculate the slot-wise word posterior probability

$$p_S(f|x_S^T) = \sum_{s \in S, word(a)=f} FB(a),$$

where a are the lattice arcs assigned to slot S and FB(a) is the forward-backward probability of arc a. In Section 5, we test using the posterior scores of the instead of the arcs. To this end, we set the score of each lattice arc to the negated log-

$$\delta - arcs.$$

Empty words from the lattice are not considered in translation and hence ε-arcs do not need a slot label.

5. Experiments

5.1. System Details

The speech translation experiments were carried out on two different tasks. The Italian-English Basic Travel Expression Corpus (BTEC) task contains tourism-related sentences usually found in phrase books for tourists going abroad. We were kindly provided with this corpus by ITC-irst (now FBK, Trento, Italy). Another BTEC corpus was available to us for Chinese-to-English translation through participation in the evaluation campaign of the International Workshop on Spoken Language Translation (IWSLT, [4]). The corpus statistics for the training corpora of the two tasks are shown in Table 1. In case of Chinese, we count characters because the translation system was trained on the character level to alleviate the large mismatch between the Chinese vocabulary used in ASR lattice generation and the vocabulary used for the MT training corpora.

The statistics of the development and test data are given in Table 2. For Italian-to-English, we used the CSTAR’03 corpus divided in two parts as the development and test data. For Chinese-to-English, the CSTAR’03 corpus was the development set, and the IWSLT’05 corpus was the test set. The lattice density in Table 2 is defined as the number of arcs in a lattice divided by the segment reference length, averaged over all segments. It is measured after minimization of the original ASR lattices. The ASR graph error rate is the minimum WER/CER among all paths through the lattice.

The parallel training corpora as in Table 1 were word-aligned and phrase pairs of maximum source length 7 were extracted. For Chinese-to-English translation, punctuation marks were kept in the target phrases so that MT system could insert them in the translation process. For Italian-to-English translation, the training was done without punctuation marks.

In translation, we used the same features as in [8] with an addition of phrase count features and a distance-based distortion model. We used IBM reordering constraints on the word level with the window size of 4 for the Italian-to-English task and phrase-level IBM constraints with the maximum number of gaps 3 for the Chinese-to-English task. We used a 4-gram and
Table 4: Translation results on the Italian-to-English task.

<table>
<thead>
<tr>
<th>input</th>
<th>BLEU[%]</th>
<th>TER[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>correct transcript</td>
<td>68.2</td>
<td>25.3</td>
</tr>
<tr>
<td>single best</td>
<td>58.2</td>
<td>35.0</td>
</tr>
<tr>
<td>lattice with ASR scores</td>
<td>61.2</td>
<td>33.8</td>
</tr>
<tr>
<td>with posterior score</td>
<td>62.7</td>
<td>32.1</td>
</tr>
<tr>
<td>confusion network</td>
<td>61.1</td>
<td>32.5</td>
</tr>
<tr>
<td>with posterior score</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the automatic evaluation, we used the established measures BLEU score [10] and translation error rate (TER, [12]). The BLEU score measures accuracy, i.e. larger scores are better. The two measures were computed with respect to 16 reference translations. The evaluation was performed without considering word case; for the Italian-to-English task, the evaluation was also without punctuation marks. The BLEU score was used as the optimization criterion for tuning the log-linear model scaling factors on the development set.

5.2. Chinese-to-English

Table 3 presents the results for the Chinese-to-English translation direction on the test set. Translation of single-best ASR output with the CER of 20.6% has a 10% lower BLEU score than the translation of the correct transcript. Using a word lattice without ASR features in the log-linear MT model improves both BLEU and TER by about 1% absolute. In contrast, translating confusion networks created from the same lattices results in a degradation of translation quality. We attribute this to the fact that the translation and the target language model alone cannot differentiate between correct and incorrect paths, especially since exhaustive reordering takes place. Translations of incorrect word hypotheses may become well-formed word sequences after a permutation; yet these sequences may have nothing to do with the spoken utterance. Including the word posteriors in the log-linear model improves the CN translations dramatically, we observe a 2.0% absolute improvement over the single-best baseline. Similarly, translating word lattices using acoustic and source LM probabilities with optimal scaling factors results in an even larger improvement of 2.3% in BLEU. Using the word posterior probabilities in lattice-based translation is inferior to using the ASR scores on this task.

Note that all ASR output translation results in Table 3 are significantly better than the official submissions of the IWSLT’05 evaluation [4], even those trained on large amount of unrestricted data. A large share of this improvement is due to better and more exhaustive reordering, which also works successfully on the word lattices. Translating word lattices with ASR scores monotonically results in BLEU of 45.3% and TER of 44.6%, which is 3.1% and 2.0% absolute worse than the result in Table 3, respectively. Thus, the improvement due to reordering is much larger than for the lattice-based beam search with a skip of a single phrase as reported in [13].

5.3. Italian-to-English

Table 4 presents the test set results for the Italian-to-English translation direction. Note that all of these results are obtained using the same training and test data as in [8], but are better by a large margin. The improvement over the translation of single-best ASR output using lattices with ASR scores is about as large as using the CNs with the word posterior probabilities. However, on this task the lattice-based system has to search through fewer paths and is at 7 words/sec about 6 times faster than the CN-based system under the same pruning settings. This illustrates the efficiency of the proposed lattice-based search. The largest improvement of 4.5% absolute in BLEU and 2.9% in TER is obtained by using lattices with word posterior probabilities. This contradicts the results on the Chinese-to-English task where the best result was achieved using theoretically motivated acoustic and source LM scores. The reason for this may be parameter overfitting or the differences in lattice structure and quality. In the future, we plan to further investigate the role of word posterior probabilities.

6. Conclusions

In this paper we showed how speech translation can be improved by using word lattices augmented by confusion network information. Significant improvements could be obtained on two translation tasks including a Chinese-to-English task where exhaustive reordering is required for good performance. To make the reordering in lattice translation possible, we introduced a procedure for labeling lattices with slot information derived from a CN that helps to guide the cardinality-synchronous search and reduces its complexity. We also showed experimentally that the proposed MT search algorithm for general ASR lattices compares favorably in terms of efficiency and quality with the search that processes confusion networks. In the future we would like to apply the method to larger tasks.

7. Acknowledgements

This paper is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023.

8. References