Direct Word Graph Rescoring Using A* Search and RNNLM

Shahab Jalalvand, Daniele Falavigna

Human Language Technology unit, Fondazione Bruno Kessler, via Sommarive 18, Trento, Italy
{jalalvand,falavi}@fbk.eu

Abstract

The usage of Recurrent Neural Network Language Models (RNNLMs) has allowed reaching significant improvements in Automatic Speech Recognition (ASR) tasks. However, to take advantage of their capability for considering long histories, they are usually used to rescore the N-best lists (i.e. it is in practice not possible to use them directly during acoustic trellis search). We propose in this paper a novel method for rescoring directly the hypotheses contained in the word graphs, which are generated in the first pass of ASR decoding. The method, based on the A* stack search, rescores the partial theories of the stack with a log-linear combination of the acoustic model score and a linear combination of multiple language model scores (including RNNLM). We compared, on an ASR task consisting of the automatic transcription of English weather news, the A* based approach with N-best rescoring and iterative confusion network decoding. Using the proposed method, we measured a relative word error rate improvement of about 6%, on the given task, with respect to using the baseline system. The latter improvement is comparable with that obtained with N-best list based rescoring method.

Index Terms: recurrent neural network language model, word graph, rescoring, confusion network, A* stack search

1. Introduction

The Language Model (LM), usually trained on a large set of text data, allows predicting the a priori probability of a word sequence \(W\). In ASR, it is used in combination with the acoustic model (AM) to estimate the maximum a posterior probability of a sequence of acoustic observations.

The popular, widely used N-gram LMs [1] are limited by both the use of discrete probability densities to compute \(P(W)\), and the need for implementing back-off procedures to handle unseen n-grams. In the literature, different smoothing methods have been discussed for discounting and redistributing probabilities [2][3][4], whose performance mostly depend on the size of training data.

Recently, to overcome above shortcomings, continuous space LMs based on: back-propagation neural network [5], neural network exploiting deep learning [6] and recurrent neural network LM (RNNLM) [7], have demonstrated excellent performance to predict the a priori probability of a sequence of words. Because of exponential increase in computation time to learn the parameters of RNNs given a large amount of training data, RNNLMS are usually trained on a limited set of text data (e.g. millions of words). This data is usually selected to be in a specific application domain, where the ASR system will have to work. In addition, it is difficult to take into account the full long-spanning capability of RNNLM when the search process is carried out over the trellis of acoustic frames [25][26], as the trellis has to be expanded according to the LM history. A simple solution is to generate N-best lists by means of a “general-purpose” n-gram LM and rescore them with the domain-specific RNNLM [8]. Nevertheless, N-best list generation is rather complex if \(N\) is large and there is no guarantee that the best hypothesis is present in each list. For this reason, some approaches exploiting hill climbing search over either confusion networks (CNs) or word lattices have been proposed in [9][24] and [10]. The decoding approach, described in [24], iteratively rescores the sentential hypotheses contained in a CN with an RNNLM and reorder the bin transitions at each step. We will briefly describe this method, later in Section 3. The work in [25] proposes to expand and rescore word graphs (WGs) with RNNLM: to avoid exponential increase of the number of nodes during the expansion process, node word histories are clustered.

All the aforementioned methods perform transformations from the original search space, before rescoring with RNNLM. In [9][10], starting from a given WG, new paths are artificially introduced (in case of CN iterative decoding even the acoustic probabilities of the original WG transitions are lost) and in [25] many paths contained in the original WG can be ignored due to word history clustering.

This work aims at investigating the possibility of rescoring directly the original search space defined by WGs taking advantage, at the same time, of the long-spanning prediction capability of RNNLM. To this purpose, we propose to rescore all of the partial hypotheses contained in a A* stack, with an interpolation of LMs, including RNNLM. Partial WG hypotheses are generated via an exact A* stack decoding [11][12][13], using as look-ahead function the score of the best path following each stack theory. The latter function is computed with a preliminary backward step applied to the given WG.

Note that the integration of RNNLM in Machine Translation (MT) systems has just been proposed in [26], while in [5] an approach for integrating Continuous Space LM (CSLM) in ASR systems has been described. However, to the best of our knowledge, the method which is described in this paper for direct rescoring of ASR partial hypotheses using RNNLM is novel.

We tested the proposed approach on a task consisting of automatic transcription of English weather reports. We compared the performance of A* stack rescoring approach with both N-best list and iterative CN rescoring. A considerable improvement in terms of Word Error Rate (WER on the 1-bests) and Oracle WER (OWER on the N-bests) has been obtained using this method.

The rest of this paper is organized as follows. In Section 2 we describe audio and text corpora, used to train the various acoustic models (AMs) and LMs. In the same Section we also give some details on the ASR systems used to generate the search spaces for rescoring. In Section 3 the rescoring approaches, namely: N-best list based method, iterative decoding over CNs and the proposed A* based approach are described in details. Results and experiments are reported in Section 4, and finally, Section 5 concludes the paper.
2. Description of AM/LM training

Within the framework of the European project EU-BRIDGE\(^1\) (see www.eu-bridge.eu), an ASR evaluation campaign has been organized in order to transcribe weather reports. To do this, a set of training, development and test data have been acquired and delivered to the partners involved in the campaign.

2.1. AM training

Audio training data consists of around 120 hours of English weather forecast. A part of these recordings is associated with captions which are not verbatim transcription of the audio itself, but sometimes they are closely related to it.

A lightly supervised approach \([14]\) was employed in order to train domain-specific triphone Hidden Markov Models (HMMs). The procedure consisted of:

1. Training a set of acoustic models over a large general-domain corpus from EURONEWS channel (about 525 hours).
2. Automatically transcribing the above mentioned 120-hour weather news using the latter acoustic model and a 4-gram LM adapted to the weather-domain text data.
3. Selecting the audio segments where the related automatic transcription and caption are matched.
4. Training weather-specific HMMs on the selected speech segments.

The last three steps of the above procedure were repeated twice, resulting in a portion of selected in-domain speech that grew to 57 hours and then to 65 hours. Given the modest improvement in the third iteration, the procedure was not repeated further. In conclusion, the method allowed to automatically select about 55% of the provided training data.

Automatically selecting about 55% of the provided training data resulted in a portion of selected in-domain speech segments that grew to 57 hours and then to 65 hours. Given the modest improvement in the third iteration, the procedure was not repeated further. In conclusion, the method allowed to automatically select about 55% of the provided training data.

Given the modest improvement in the third iteration, the method allowed to train domain-specific triphone Hidden Markov Models (HMMs). The procedure consisted of:

1. Training a set of acoustic models over a large general-domain corpus from EURONEWS channel (about 525 hours).
2. Automatically transcribing the above mentioned 120-hour weather news using the latter acoustic model and a 4-gram LM adapted to the weather-domain text data.
3. Selecting the audio segments where the related automatic transcription and caption are matched.
4. Training weather-specific HMMs on the selected speech segments.

The last three steps of the above procedure were repeated twice, resulting in a portion of selected in-domain speech that grew to 57 hours and then to 65 hours. Given the modest improvement in the third iteration, the procedure was not repeated further. In conclusion, the method allowed to automatically select about 55% of the provided training data.

2.2. LM training

As for audio data, a set of weather reports, containing about one million words (named Subtitles in Table 1), has been made available. From this latter we trained an in-domain 4-gram LM which was successively used to automatically select documents from a general corpus of news, named "Google-news". This latter is an aggregator of news provided and operated by Google that collects news from many different sources, in different languages. We download daily news from this site, filter-out useless tags and collect texts. Up to now our version of "Google-new" text corpus contains around 1.6 billion words. We ordered documents of “Google-news” according to increasing perplexity (computed with the previously mentioned in-domain LM) and we selected the documents with lowest perplexities to build a corpus of about 100 million words \([15]\), over which a 4-gram back-off LM was trained with the IRSTLM toolkit \([16]\), using the modified Kneser-Ney interpolation method. Then, the latter LM was adapted to the weather domain using the 1MW weather subtitles and the mixture adaptation method as provided by the same IRSTLM toolkit. This LM (hereinafter we will name it as LMbase), together with a lexicon, was exploited to build the Finite State Network used during two ASR decoding passes.

Finally, on the same in-domain corpus, we trained an RNNLM with 450 hidden neurons and 1000 classes using RNNLM toolkit \([18]\). This configuration has shown the best performance in our previous work \([19]\) on the data at hand. Table 1 reports some statistics about LM training data, as well.

<table>
<thead>
<tr>
<th>Set</th>
<th>Audio data</th>
<th>Text data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train1 (AM1)</td>
<td>Train2 (AM2)</td>
<td>Train3 (AM3)</td>
</tr>
<tr>
<td>525 hr</td>
<td>57 hr</td>
<td>65 hr</td>
</tr>
<tr>
<td>3 G</td>
<td>638 K</td>
<td>711 K</td>
</tr>
<tr>
<td>General</td>
<td>Weather</td>
<td>Weather</td>
</tr>
</tbody>
</table>

Table 1. The exploited audio and text data

<table>
<thead>
<tr>
<th>Set</th>
<th>Audio data</th>
<th>Text data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subtitles</td>
<td>Auto-sel</td>
<td>Google-news</td>
</tr>
<tr>
<td>1 G</td>
<td>100 G</td>
<td>1.6 T</td>
</tr>
<tr>
<td>8 K</td>
<td>277 K</td>
<td>300 K</td>
</tr>
<tr>
<td>Weather</td>
<td>Related to weather</td>
<td>General</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LM</th>
<th>PPL</th>
<th>OOV%</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMbase</td>
<td>45.8</td>
<td>0.0</td>
</tr>
<tr>
<td>LMin</td>
<td>39.6</td>
<td>0.04</td>
</tr>
<tr>
<td>RNNLM</td>
<td>34.7</td>
<td>0.04</td>
</tr>
<tr>
<td>RNNLM+LMin</td>
<td>31.6</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Table 2. Perplexity values (PPL) and Out-Of-Vocabulary (OOV) rate of the different LMs

On the weather development set we measured the perplexity values of the various trained LMs, as well as of an additional linearly interpolated LM: \(RNNLM+LMin\). In this case interpolation coefficients were computed by means of the expectation-maximization algorithm with the aim of minimizing the perplexity on the development data. Table 2 gives the obtained perplexity results (more details on the latter evaluation have been published in \([19]\)) and the corresponding Out-Of-Vocabulary (OOV) rates. It is worthy to point the PPL improvement gained with the linearly interpolated LM: \(RNNLM+LMin\). This LM is used for rescoring experiments reported in Table 2.

\(^1\) This work has been partially founded by the European project, EU-BRIDGE, under the contract FP7-287658.
2.3. Word graphs generation

The manually detected segments of the development/test set are grouped by a segment clustering method, based on the Bayesian information criterion, then, cluster-wise feature normalization and acoustic model adaptation are applied. The ASR system employs LMbase, along with continuous density, state-tied, cross-word, gender-independent triphone HMMs as the acoustic models in a two decoding recognition passes.

Speaker adaptively trained HMMs used in the first decoding pass were trained [20][21] with acoustic observations obtained through: 1) unsupervised, cluster based normalization of 52 dimensional acoustic feature vectors (to this purpose constrained maximum, likelihood linear regression [22] is used) and 2) Heteroscedastic Linear Discriminant Analysis (HLDA) projection of 52 dimensional normalized feature vectors into 39 dimensional ones. In the second decoding pass, speaker adaptively trained triphone HMMs were trained on normalized, HLDA projected, acoustic feature vectors by applying a cluster based, affine transformation estimated w.r.t triphone HMMs, with a single Gaussian.

In both cases, triphone HMMs were trained through a conventional maximum likelihood procedure. At recognition stage, the output of the first decoding pass is exploited as supervision for CMLLR-based feature normalization and MLLR-based acoustic model adaptation. A frame synchronous Viterbi beam-search is used to find the most likely word sequence.

In the second recognition pass, the decoder generates for each given speech utterance the best word sequence, that was used to evaluate the baseline performance, as well as a word graph. From the latter the list of related N-best hypotheses was computed as well as a confusion network. The graph error rate measured on the utterances of the development set, using the best set of available AMs, resulted to be 4.3%, which is around 1/3 of the related WER.

3. Rescoring approaches

In this section, we describe the state-of-the-art iterative CN decoding approach and our proposed method which is based on A* stack rescoring. Afterwards, in the next section we compare these methods with N-best list rescoring.

3.1. Iterative CN decoding

The word graph transitions having a specific amount of time overlap could be merged into a bin. A confusion network [23] is a chain of these bins. All the transitions in a bin are ordered according to their posterior probability. Therefore, the 1-best hypothesis is obtained by concatenation of the first transitions in the consequent bins.

Iterative decoding [24] reorders each transition in each bin according to a linear combination of scores. For experiments reported below we have used: posterior probability of the transition itself, baseline LM probability of the “locally best” sentence hypothesis (the latter is obtained joining the best left context and the best right context), RNNLM probability of the “locally best” sentence hypothesis and in-domain LM probability of the “locally best” sentence hypothesis. Over a grid of values, we estimated the interpolation weights in order to minimize the WER on the development set.

At each step, just one word is changed. Regarding this change, if the better score is achieved, the bin becomes updated. Therefore, it is guaranteed that after each step a better solution has been found. Like any other hill climbing algorithm, there is the possibility of getting stuck at local minima. To reduce this effect, the simulated annealing algorithm proposed in [9] can be used.

Note that, due to the merging procedure applied during CN generation, the OWER of a CN decreases in comparison to the corresponding WG or N-best list. At the same time, however, acoustic likelihoods contained in the original WG are missed. This fact prevents direct combination of the LM probability with AM probability for computing hypothesis scores.

3.2. A* stack rescoring

A* stack search algorithm [11][12][13] on a WG starts with expanding the first node, pushing the partial paths (sometimes called partial theories) into a stack and sorting them with regard to a total score. The total score of a partial theory reaching node i (\( S_i(T_i) \)) is given by the sum of the acoustic model score of the partial theory \( (S_i(T_i)) \), its language model score \( (S_{LM}(T_i)) \) and a score furnished by a look-ahead function \( (LH(i)) \). Here, the look-ahead function of node i, \( LH(i) \), is the score of the best path from i to the final state. In the proposed A* stack rescoring method, all the scores except language model are kept unchanged. The latter score is recomputed by means of the new LM components. Hence, for each partial theory \( T_i \) (including the word sequence \( \text{word}_1 \ldots \text{word}_n \)), the total score \( S_i \) is given by:

\[
S_i(T_i) = S_{LM}(T_i) + LH(i) \]

3.3. A* stack rescoring

A* stack search algorithm [11][12][13] on a WG starts with expanding the first node, pushing the partial paths (sometimes called partial theories) into a stack and sorting them with regard to a total score. The total score of a partial theory reaching node i (\( S_i(T_i) \)) is given by the sum of the acoustic model score of the partial theory \( (S_i(T_i)) \), its language model score \( (S_{LM}(T_i)) \) and a score furnished by a look-ahead function \( (LH(i)) \). Here, the look-ahead function of node i, \( LH(i) \), is the score of the best path from i to the final state. In the proposed A* stack rescoring method, all the scores except language model are kept unchanged. The latter score is recomputed by means of the new LM components. Hence, for each partial theory \( T_i \) (including the word sequence \( \text{word}_1 \ldots \text{word}_n \)), the total score \( S_i \) is given by:

\[
S_i(T_i) = S_{LM}(T_i) + LH(i) \]

where, \( S_{LMbase} \) is the language model score given by the background LM (LMbase in Table 2). \( S_{LM} \) is obtained by log-linear interpolation of the background LM and a new LM component which is, in turn, the linear interpolation of RNNLM and LMin (\( S_{RNNLM-LMin} \)). \( Pr_{RNN} \) is the word probability given by the RNNLM and \( Pr_{LMin} \) is the word probability given by LMin. The parameter \( \alpha \) is the LM scale factor used in the ASR decoder; while, \( \beta \) and \( \gamma \) are the LM weights, tuned for minimizing the WER on the development set. We illustrate the effectiveness of the A* stack rescoring approach using a real example given in Figure 1, where, a WG is plotted for an utterance which is to result: “some sunshine in-between”.

On this example, the traditional A* stack search works as shown in Figure 1.b, where At each step (1, 2, … , 5), it popes the top element of the stack, expands the node and pushes the partial theories into the stack and sorts the stack. At any crucial step, like step 3, if the algorithm fails to rank the partial theories, the correct solution might be missed even in the final N-best list (here N=4). In this example, at step 4, the stack overflows, and therefore, the algorithm must drop the last theory which is indeed the correct one. Hereby, we have missed the correct solution not only in the 1-best, but also in the final N-best list. To alleviate this issue, we enhance the algorithm of ranking the partial theories, by updating the total scores using Eq. (1).

Using the new approach, at the crucial step 3, the algorithm succeeds in ranking the partial theories and places the better theory on the top.

3
Therefore, at step 4, the top element of the stack is popped; expanded and labeled as a terminator solution (i.e. this theory will be no more expanded).

The main benefit of this method over the traditional WG rescoring approaches (like in [17]) is that with this method, we could apply a long-span LM on the word graph without doing any expansion.

4. Experiments and results

Experiments have been carried out on the given development/test sets using the acoustic models AM1, trained on English broadcast news domain; AM2 and AM3, trained on weather domain (see Section 2.1). Using each set of acoustic models, along with LMbase as the language model, we generated the 1-best, 100-best, WGs and CNs for each utterance in the development/test set. Results obtained with each one of the rescoring approaches described in Section 3 are reported in Table 3. Both word error rate (WER) and oracle word error rate (OWER) are given (OWER is defined as the WER of the best hypothesis contained in the 100-best list from either word graph or confusion network).

<table>
<thead>
<tr>
<th>AM</th>
<th>Rescoring method</th>
<th>OWER of 100-best Dev</th>
<th>WER on Dev</th>
<th>WER on Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM1</td>
<td>Baseline (no rescoring)</td>
<td>9.9 (+7.9)</td>
<td>10.2 (+6.1)</td>
<td>10.6 (+5.9)</td>
</tr>
<tr>
<td></td>
<td>N-best list rescoring</td>
<td>9.9 (+7.9)</td>
<td>10.2 (+6.1)</td>
<td>10.6 (+5.9)</td>
</tr>
<tr>
<td></td>
<td>Iterative CN rescoring</td>
<td>9.9 (+7.9)</td>
<td>10.2 (+6.1)</td>
<td>10.6 (+5.9)</td>
</tr>
<tr>
<td></td>
<td>A* stack rescoring</td>
<td>10.9 (+7.9)</td>
<td>11.2 (+6.1)</td>
<td>11.5 (+5.9)</td>
</tr>
</tbody>
</table>

As expected, from AM1 to AM3, by improving the expertise of the acoustic models, the performance of the baseline system improves. N-best list rescoring shows a significant improvement for all AMs, over both development and test set. This is in line with the previous works in the literature that used RNNLM for N-best list rescoring. However, a very slight improvement is observed by using iterative CN decoding. Note that the works in [9][24], on a broadcast news corpus, showed the efficiency of iterative decoding approach with respect to the N-best list rescoring, in terms of search effort reduction. As mentioned in the introduction, iterative decoding could also be applied to rescoring directly the word graphs. This would be considered as a part of future works.

Finally, we observe from Table 3 that the proposed A* based rescoring approach outperforms other rescoring methods, in terms of %WER, when AM1 and AM2 are used as the acoustic models. However, the same %WER (10.6%) as the N-best list rescoring is obtained when the best acoustic model (AM3) is used. One reason is that, in the latter case, the acoustic model is such accurate that the 1-best hypothesis, found from the WG, is probably the best solution. Therefore, the language model plays the minor role for selecting the better solutions. Despite this fact, we observe that %ower improves significantly when using A* stack rescoring, i.e. there is still room for decreasing the related %WER, e.g. with RNNLM trained on a bigger set of data.

5. Conclusions

In this paper, we have proposed a new method based on directly rescoring the partial theories, contained in A* stack by means of RNNLM. We have reported preliminary results showing improvements in performance with respect to the baseline system, N-best list and iterative CN decoding approaches. Much more work still need to be done in order to: 1) show the effectiveness of the proposed method on larger domains (such as TED talks); 2) estimate the computational requirements of the approach and compare it with other rescoring methods; 3) verify if larger improvements could be achieved with better LMs used for rescoring (e.g. bigger RNNLM, factored LM, exponential LMs, etc).
References