Comparing Approaches to Convert Recurrent Neural Networks into Backoff Language Models For Efficient Decoding

Heike Adel[1,2], *Katrin Kirchhoff*[2], *Ngoc Thang Vu*[1], *Dominic Telaar*[1], *Tanja Schultz*[1]

[1]Karlsruhe Institute of Technology, Cognitive Systems Lab, Germany
[2]University of Washington, Department of Electrical Engineering, USA
heike.adel@student.kit.edu

10.21437/Interspeech.2014-165

Abstract

In this paper, we investigate and compare three different possibilities to convert recurrent neural network language models (RNNLMs) into backoff language models (BNLM). While RNNLMs often outperform traditional n-gram approaches in the task of language modeling, their computational demands make them unsuitable for an efficient usage during decoding in an LVCSR system. It is, therefore, of interest to convert them into BNLMs in order to integrate their information into the decoding process. This paper compares three different approaches: a text based conversion, a probability based conversion and an iterative conversion. The resulting language models are evaluated in terms of perplexity and mixed error rate in the context of the Code-Switching data corpus SEAME. Although the best results are obtained by combining the results of all three approaches, the text based conversion approach alone leads to significant improvements on the SEAME corpus as well while offering the highest computational efficiency. In total, the perplexity can be reduced by 11.4% relative on the evaluation set and the mixed error rate by 3.0% relative on the same data set.

Index Terms: language modeling, recurrent neural networks, decoding with neural network language models, code switching

1. Introduction

Recurrent neural network language models (RNNLMs) can improve perplexity and error rates in speech recognition systems compared to traditional n-gram approaches [1, 2, 3]. Reasons are their ability to handle longer contexts and their implicit smoothing in the case of unseen words or histories. However, their computational demands are too high to integrate them into the decoding process of large vocabulary continuous speech recognition (LVCSR) systems. They are so far mainly used for rescoring. However, rescoring experiments are based on the extraction of lattices or N-best lists which depend on the language model (mostly n-gram models) used during decoding.

The main contribution of this paper is the presentation and comparison of different techniques for converting RNNLMs into backoff n-gram language models (BNLMs) in order to incorporate the neural network information into the decoding process. Furthermore, we present a novel RNNLM probability based conversion approach and adapt an existing conversion method for feed forward neural networks to RNNLMs. All conversion approaches can be applied to any RNNLM structure. The RNNLM used in this work incorporates both a factorization of the output layer into classes and an extension of the input layer with an additional feature vector.

2. Related work

In the last years, recurrent neural networks have been used for a variety of tasks including language modeling [1]. Due to the recurrence of the hidden layer, they are able to handle long-term contexts. It has been shown that they can outperform traditional language models (LMs), such as n-grams. During the last years, the structure of RNNLMs has been extended for different purposes. Mikolov et al. factorized the output layer of the network into classes to accelerate training and testing [2]. In [3, 4], features, such as part-of-speech tags, were integrated into the input layer to add linguistic information to the LM process and provide more general classes than words to handle data sparseness. In [3], we built an RNNLM for Code-Switching speech. It used part-of-speech tags as input features and clustered the output vector into language classes.

The following subsections describe a method to approximate RNNLMs with BNLMs and an approach to convert feed forward neural networks into backoff language models.

2.1. Approximate RNNLMs with backoff n-gram LMs

Deoras et al. used Gibbs sampling to approximate an RNNLM with a BNLM [5]. They trained an RNNLM and used its probability distribution to generate text data. Afterwards, they built an n-gram LM with that data. Finally, they interpolated their resulting n-gram LM with their baseline n-gram LM which has been built on the same training text as the RNNLM. This final model improved the perplexities compared to the baseline n-gram model and led to different N-best lists and rescoring improvements after decoding. The authors found that the results were better, the more text they generated with their RNNLM.

2.2. Conversion of feed forward neural networks into backoff language models

Arisoy et al. presented an iterative algorithm to build a BNLM based on a feed forward neural network (FFNN) [6]. First, n-gram models and FFNN LMs were trained for each order (2-grams, 3-grams and 4-grams). Second, a BNLM with the probabilities of the FFNNs was created iteratively. If a probability could not be extracted from the FFNNs because the word was not in their output vocabulary V_o, it was calculated by the n-gram models. This is why they were called "background language models" (BLM). Equation 1 shows this.

$$P(w|h) = \begin{cases} \beta(h) P_{NNLM}(w|h) \text{ if } w \in V_o \\ P_{BLM}(w|h) \text{ if } w \notin V_o \end{cases} \quad (1)$$

Initially, unigram probabilities were obtained from a background n-gram model. Then, a 2-gram model containing all

possible bigrams was created. The probabilities from the FFNN and the background 2-gram model were normalized with a history dependent constant β and the model was pruned. Based on the pruned bigrams, a model with all possible trigrams was generated in the next step using the same strategy as before. As the background 3-gram model, the 2-gram result from the previous step was used after it had been extended with the 3-grams of a traditional 3-gram LM. The authors performed these steps for 2-grams, 3-grams and 4-grams. Finally, the resulting 4-gram LM was interpolated with a traditional 4-gram LM. It outperformed the baseline both in terms of perplexity and recognition performance.

3. RNNLM for Code-Switching

This section describes the structure of our Code-Switching RNNLM [3]. It is illustrated in Figure 1.

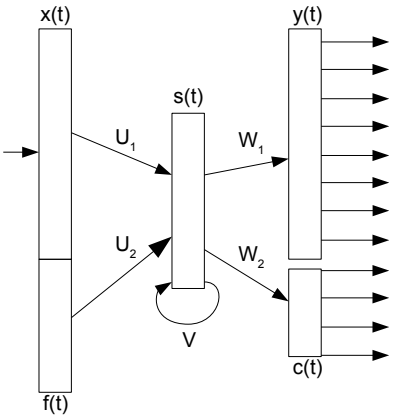

Figure 1: RNNLM for Code-Switching
(based upon a figure in [2])

The network consists of three layers: an input layer, a hidden layer and an output layer. The hidden layer does not only depend on the input layer but also on the hidden layer of the previous iteration. This is why the network is called "recurrent". The input layer is formed by a vector x of the size of the vocabulary. Furthermore, a feature vector f consisting of one entry for each POS tag is added to the input layer. Hence, the values of the hidden layer depend on both the current word and the current POS tag. Based on the values of the hidden layer, the output layer is calculated. The output vector y consists of one entry per vocabulary word. Its softmax-activation function ensures that it provides a probability distribution for the next word. As mentioned before, Mikolov et al. have factorized the output vector into frequency based classes for acceleration [2]. We have proposed to use language classes instead to model the Code-Switching phenomenon [3]. Therefore, the network first computes the probability for the next language class c and based on this the probability for the next word.
This RNNLM is converted to BNLMs in this work in order to use its information directly during decoding.

4. Conversion approaches

For the conversion of the RNNLM into BNLMs, three different approaches are evaluated and compared. They are presented in the following subsections.

4.1. Approach 1: text based conversion

The first approach investigated in this paper is the text based conversion approach suggested by [5]. As described in Section 2.1, the RNNLM is used to generate a large amount of text. Based on this text, we build a 3-gram LM with the SRILM toolkit [7]. We use Witten-Bell discounting for unigrams and Kneser-Ney discounting for bigrams and trigrams [8]. We also investigate the effect of creating different amounts of text data.

4.2. Approach 2: probability based conversion

In our novel approach, we extract unigrams, bigrams and trigrams from the training text similar to traditional n-gram approaches. However, we do not assign count-based probabilities to them but probabilities of our RNNLM. To obtain these probabilities, we extract the RNNLM probability for every word w_i of the training text and assign it to the current unigram, bigram and trigram. If the same unigram, bigram or trigram occurs more than once, the probabilities are averaged. Finally, we normalize the RNNLM outputs y to obtain a probability distribution using the formular $p(w_i|h) = \frac{y(w_i|h)}{\sum_{w_k} y(w_k|h)}$. Then, we smooth the distribution to provide probability mass for backoff. In particular, we ensure that the probabilities for all n-grams ($n = 1, 2, 3$) with the same history h sum to a specific number $0 < S < 1$. Hence, the probability mass available for backoff is $1 - S$. Experiments showed that if we set S history dependently to the same sum as in a baseline 3-gram model built on the same training text, the results are better than if a fixed number is chosen for S independent from the word history. This method of generating a BNLM based on an RNNLM is innovative and one of the contributions of this paper.

4.3. Approach 3: iterative conversion

The third conversion approach is based on the algorithm presented by [6]. Since it has already been described in Section 2.2, this part focuses on the major differences applied in this work. First, Arisoy et al. adjusted the probabilities of the FFNNs with a history dependent normalization constant β to the probabilities of the n-gram models [6]. In this study, we regard the usage of the background n-gram model as backoff from the RNNLM in the case that the probability cannot be extracted from the RNNLM. Hence, we calculate β in the same way as a backoff weight and multiply it with the probabilities of the n-gram model. Second, we did not train 2-gram and 3-gram RNNLMs. On the one hand, this seems to contradict the idea of the (infinitely) recurrent hidden layer. On the other hand, we found that restricting the history of the RNNLM led to substantially worse models in terms of perplexity. In particular, we considered two ways for adjusting the context of the RNNLM: 1) Adjusting the number of unfolding steps for backpropagation, 2) Resetting the values of the hidden layer after each bigram / trigram of the training text. Both approaches increased the perplexity by more than 100% relative. Therefore, we developed a novel method to extract bigram and trigram probabilities from an RNNLM. It is illustrated in Figure 2. For each word of the training text (or for each two subsequent words in the case of trigrams), we obtain the probabilities for all possible next words. Hence, we extract the whole output vector of the RNNLM. Then, we store these probabilities as bigram (or trigram) probabilities given the current word (or the previous and the current word) as histories. Again, we average the probabilities if a bigram or trigram occurs more than once. In contrast approach 2 (see Section 4.2), we do not only extract probabil-

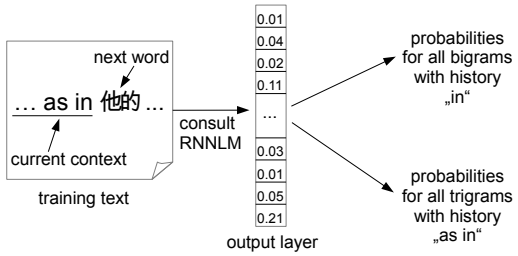

Figure 2: Obtain bigram/trigram probabilities from an RNNLM

ities for every bigram and trigram of the training text but for every possible bigram and trigram based on the histories occuring in the training text. Hence, approach 2 could be regarded as a simplified sub-case of approach 3 (with substantially less computation demands).

5. Experiments and results

The three approaches are evaluated and compared in terms of perplexity and mixed error rate. For these experiments, a Code-Switching data corpus is used.

Code-Switching (CS) refers to speech which contains more than one language. This happens in multilingual communities (e.g. among immigrants). Nowadays, it can also be detected in former monolingual countries due to globalization effects. For the automatic processing of CS speech, it is important to capture long distant dependencies and language changes. Therefore, multilingual models and CS training data are necessary.

5.1. Data corpus

SEAME (South East Asia Mandarin-English) is a conversational Mandarin-English CS speech corpus. It has been recorded in Singapore and Malaysia by [9] and was originally used for the joint research project "Code-Switch" by Nanyang Technological University (NTU) and Karlsruhe Institute of Technology (KIT). The recordings consist of spontaneously spoken interviews and conversations. The corpus contains about 63 hours of audio data with manual transcriptions. The words can be categorized into Mandarin words (58.6% of all tokens), English words (34.4% of all tokens), particles (Singaporean and Malayan discourse particles, 6.8% of all tokens) and other languages (0.4% of all tokens). The average number of CS points between Mandarin and English is 2.6 per utterance. The average duration of monolingual English and Mandarin segments is only 0.67 seconds and 0.81 seconds, respectively. In total, the corpus contains 9,210 unique English and 7,471 unique Mandarin words. We divided the corpus into three disjoint sets (training, development and evaluation set). The data were distributed based on criteria like gender, speaking style, ratio of Singaporean and Malaysian speakers, ratio of the four language categories and the duration in each set. Table 1 provides statistical information about the different sets.

Table 1: Statistics of the SEAME corpus

	Train set	Dev set	Eval set
# Speakers	139	8	8
Duration (hrs)	59.2	2.1	1.5
# Utterances	48,040	1,943	1,018
# Tokens	525,168	23,776	11,294

5.2. Perplexity results

For the perplexity experiments, a 3-gram LM built on the SEAME training text with the SRILM toolkit [7] serves as baseline language model. It will be referred to as "CS 3-gram". To evaluate the closeness of the converted RNNLMs to the original RNNLM, the perplexity results of the unconverted RNNLM are also provided.

Table 2 provides perplexity results for approach 1 (text based conversion) and different amounts of generated text data.

Table 2: PPL results for conversion approach 1 (w denotes the weight of the converted RNNLM when interpolating it with the CS 3-gram)

Model	PPL dev	PPL eval
Baseline: CS 3-gram	268.39	282.86
10M words text	410.67	487.97
+ CS 3-gram (w : 0.290)	256.34	276.61
110M words text	391.24	463.98
+ CS 3-gram (w : 0.328)	251.09	272.01
235M words text	385.05	454.57
+ CS 3-gram (w : 0.342)	**249.53**	**270.55**
300M words text	388.52	459.51
+ CS 3-gram (w : 0.336)	249.96	270.91
650M words text	434.22	509.19
+ CS 3-gram (w : 0.310)	252.63	272.82
RNNLM (unconverted)	219.85	239.21

In contrast to the results of [5], we observe that the benefit obtained by larger amounts of generated text is limited. Increasing the text to 300M or more words does not lead to improvements. An explanation could be shortage of training data for the RNNLM.

Table 3 shows perplexity results for conversion approach 2 (probability based). In each column (unigrams, bigrams, trigrams), it is indicated whether the probabilities are extracted from the RNNLM or from the CS 3-gram model. The first row, for instance, corresponds to the baseline CS 3-gram model since all probabilities are obtained from this model.

Table 3: PPL results of 3-gram LMs obtained by approach 2 (w denotes the weight of the new 3-gram when interpolating it with the CS 3-gram)

Source of Unigrams	Source of bigrams	Source of trigrams	PPL dev	PPL eval
CS 3-gram	CS 3-gram	CS 3-gram	268.39	282.86
RNNLM	RNNLM	RNNLM	574.82	585.71
+ CS 3-gram (w : 0.088)			267.58	280.76
CS 3-gram	RNNLM	RNNLM	287.57	309.03
+ CS 3-gram (w : 0.434)			**260.04**	**274.89**
CS 3-gram	CS 3-gram	RNNLM	285.01	301.81
+ CS 3-gram (w : 0.324)			263.75	277.85

The RNNLM does not seem to provide reliable estimates for unigrams. Hence, the RNNLM needs a rather long context to benefit from the recurrence of the hidden layer.

Finally, Table 4 presents the perplexity results for approach 3 (iterative conversion). The model "2-gram-BO-LM" denotes the 2-gram LM obtained after the first iteration while the name "3-gram-BO-LM" refers to the final 3-gram LM.

The perplexity results of approach 3 are superior to the results of the other two approaches. Nevertheless, they also reveal its main limitation: the probability extraction based on the training text. Since more bigram histories (one word histories) are covered in the training text than trigram histories (two word histo-

Table 4: PPL results of LMs obtained by approach 3 (w denotes the weight of the new 2-gram when interpolating it with the CS n-gram)

Model	PPL dev	PPL eval
Baseline: CS 2-gram	276.86	295.00
Baseline: CS 3-gram	268.39	282.86
2-gram-BO-LM (without pruning)	257.08	281.39
2-gram-BO-LM (pruned to 2M bigrams)	254.79	278.12
+ CS 2-gram (w : 0.716)	243.66	262.13
+ CS 3-gram (w : 0.635)	**235.12**	**250.64**
3-gram-BO-LM (pruned to 9.5M trigrams)	280.57	300.83
+ CS 3-gram (w : 0.406)	249.11	263.06
3-gram-BO-LM (pruned to 1M trigrams)	275.38	294.93
+ CS 3-gram (w : 0.440)	249.39	263.56
3-gram-BO-LM (pruned to 300k trigrams)	258.06	277.51
+ CS 3-gram (w : 0.615)	247.72	263.15
RNNLM (unconverted)	219.85	239.21

ries), the RNNLM can be used for more bigrams than trigrams. For trigrams, the algorithm has to backoff to the background 3-gram LM more often. This can be an explanation why the 2-gram-BO-LM outperforms the 3-gram-BO-LM. The interpolation weights of the results of approach 3 are higher than the weights of the other approaches. This could indicate that approach 3 captures additional information not contained in the original training text.

In summary, all conversion approaches lead to LMs with lower perplexity values than the baseline model. A t-test shows that all improvements are statistically significant (at a level of 0.01). Furthermore, the different models obtained by approach 1 are significantly different from each other and the 2-gram-BO-LM of approach 3 is significantly better than the 3-gram-BO-LM.

Finally, an LM is built by interpolating the best result of each approach. This interpolated LM has a perplexity of 232.48 on the development set and 248.34 on the evaluation set which is slightly better than the 2-gram-BO-LM + CS 3-gram LM. The interpolation weights have been optimized on the SEAME development set using the SRILM toolkit. They are 0.389 for the LM obtained by approach 3, 0.099 for the LM of approach 1, 0.237 for the LM created with approach 2 and 0.274 for the CS 3-gram. Interestingly, the weight for approach 2 is higher than the weight for approach 1. This shows that approach 2 also provides valuable information although its results are not as good as the results of the other two methods.

5.3. Decoding experiments

The BNLMs are used directly in decoding. This section outlines important facts about the ASR system and presents the results.

5.3.1. Description of the decoding system

We apply BioKIT, a dynamic one-pass decoder [10]. The acoustic model of the ASR system is speaker independent. It applies a fully-continuous 3-state left-to-right HMM. The emission probabilities are modeled with bottleneck features [11]. The phone set contains English and Mandarin phones, filler models for continuous speech (+noise+, +breath+, +laugh+) and an additional phone +particle+ for Singaporean and Malayan particles. We use a context dependent acoustic model with 3,500 quintphones. Merge-and-split training is applied followed by three iterations of Viterbi training. To obtain a dictionary, the CMU English [12] and Mandarin [13] pronunciation dictionaries are merged into one bilingual pronunciation dictionary. The number of English and Mandarin entries in the lexicon is 56k. Additionally, several rules from [14] are applied which generate pronunciation variants for Singaporean English. On the language model side, a 3-gram model is built on the SEAME training transcriptions. It is interpolated with LMs built on English and Mandarin monolingual texts (from the NIST and GALE project). This increases the perplexity (from 268.39 to 292.58 on the development set) but reduces the out-of-vocabulary rate (from 2.10% to 1.41% on the development set) and, therefore, improves the error rate results. The resulting LM will be referred to as decoder baseline 3-gram in the following experiments.

5.3.2. Decoding results

As a performance measure for decoding Code-Switching speech, we use the mixed error rate (MER) which applies word error rates to English and character error rates to Mandarin segments [15]. With character error rates for Mandarin, the performance can be compared across different word segmentations. (In this work, we use a manual word segmentation.)

Table 5 shows the results of the converted BNLMs. Each BNLM was interpolated with the decoder baseline 3-gram LM prior to decoding.

Table 5: Decoding results with converted RNNLMs

Model	MER dev	MER eval
Decoder baseline 3-gram	39.96%	34.31%
Approach 1: 235M text	39.37%	33.41%
Approach 2 (bigrams and trigrams from RNNLM)	40.44%	34.65%
Approach 3: 3-gram-BO-LM	39.58%	33.58%
Approach 3: 2-gram-BO-LM	39.37%	**33.28%**
Approach1 + approach2 + approach3	**39.26%**	33.43%

Except for approach 2, all conversion approaches result in LMs which improve the decoding results. A t-test shows that all improvements on the evaluation set are statistically significant (at a level of 0.025). However, the differences within the approaches are not significant. As a result, converting RNNLMs and using them during decoding is beneficial. Whereas, the conversion method does not seem to be relevant on the SEAME corpus.

6. Conclusions

This paper presented and compared three different approaches to convert recurrent neural networks into backoff language models. We applied a text based conversion method and adapted an iterative approach for feed forward neural networks to recurrent neural networks. Moreover, we presented a novel probability based conversion method. The different conversion approaches were evaluated in the context of speech recognition of spontaneous Code-Switching speech. The text based and iterative conversion approaches outperformed the probability based conversion approach. Nevertheless, the combination of the results of all three approaches led to the best results in terms of perplexity and mixed error rate on the SEAME corpus. The perplexity on the SEAME evaluation set was decreased by 11.4% relative and the mixed error rate by 3.0% relative compared to a traditional 3-gram language model. Based on significance analyses of the results, we would suggest to use the text based conversion approach for corpora similar to the SEAME corpus because it led to similar results as the iterative approach while its computation costs were considerably lower.

7. References

[1] Mikolov, T. and Karafiát, M. and Burget, L. and Cernocky, J. and Khudanpur, S., "Recurrent neural network based language model", Proc. of Interspeech, 2010.

[2] Mikolov, T. and Kombrink, S. and Burget, L. and Cernocky, JH and Khudanpur, S., "Extensions of recurrent neural network language model", Proc. of ICASSP. IEEE, 2011.

[3] Adel, H. and Vu, N. T. and Kraus, F. and Schlippe, T. and Li, H. and Schultz, T., "Recurrent neural network language modeling for code switching conversational speech", Proc. of ICASSP. IEEE, 2013.

[4] Shi, Y. and Wiggers, P. and Jonker, C. M., "Towards recurrent neural networks language models with linguistic and contextual features", Proc. of Interspeech, 2012.

[5] Deoras, A. and Mikolov, T. and Kombrink, S. and Karafiát, M. and Khudanpur, S., "Variational approximation of long-span language models for LVCSR", Proc. of ICASSP, IEEE 2011.

[6] Arisoy, E. and Chen, S. F. and Ramabhadran, B. and Sethy, A., "Converting neural network language models into back-off language models for efficient decoding in automatic speech recognition", Proc. of ICASSP, IEEE 2013.

[7] Stolcke, A. and others, "SRILM - an extensible language modeling toolkit", Proc. of SLP, 2002.

[8] Chen, S. F. and Goodman, J., "An empirical study of smoothing techniques for language modeling", Technical Report TR-10-98, 1998.

[9] Lyu, D. C. and Tan, T. P. and Chng, E. S. and Li, H., "An Analysis of a Mandarin-English Code-switching Speech Corpus: SEAME", Oriental COCOSDA, 2010.

[10] Telaar, D. and Wand, M. and Gehrig, D. and Putze, F. and Amma, C. and Heger, D. and Vu, N.T. and Erhardt, M. and Schlippe, T. and Janke, M. and Herff, C. and Schultz, T., "BioKIT - Real-time decoder for biosignal processing", Proc. of Interspeech, 2014.

[11] Vu, N. T. and Metze, F. and Schultz, T., "Multilingual bottleneck features and its application for under-resourced languages", Proc. of SLTU, 2012.

[12] "CMU pronouncation dictionary for English", Online: http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[13] Hsiao, R. and Fuhs, M. and Tam, Y. and Jin, Q. and Schultz, T., "The CMU-InterACT 2008 Mandarin transcription system", Proc. of ICASSP. IEEE, 2008.

[14] Chen, W. and Tan, Y. and Chng, E. and Li, H., "The development of a Singapore English call resource", Oriental COCOSDA, 2010.

[15] Vu, N. T. and Lyu, D.C. and Weiner, J. and Telaar, D. and Schlippe, T. and Blaicher, F. and Chng, E.S. and Schultz, T. and Li, H., "A first speech recognition system for Mandarin-English code-switch conversational speech", Proc. of ICASSP. IEEE, 2012.