RECURRENT NEURAL NETWORK LANGUAGE MODELING APPLIED TO THE BRNO AMI/AMIDA 2009 MEETING RECOGNIZER SETUP

Stefan Kombrink, Tomáš Mikolov
Doctoral Degree Programme (2,4), FIT BUT
E-mail: kombrink@stud.fit.vutbr.cz,imikolov@stud.fit.vutbr.cz

Supervised by: Lukáš Burget
E-mail: burget@fit.vutbr.cz

Abstract: In this paper we use recurrent neural network (RNN) based language models to improve our 2009 English meeting recognizer originated from the AMI/AMIDA project, which to date was the most advanced speech recognition setup of the Speech@FIT. On the baseline setup using the original language models we decrease word error rate (WER) from 20.3% to 19.1%. When language models in the system are replaced by models trained on a tiny subset of the original language model data, WER drops from 22.2% to 20.4%. Adding data sampled from two RNN models for language model training improves the overall system, yielding the performance of the original baseline (20.2%).

Keywords: automatic speech recognition, language modeling, recurrent neural networks

1 INTRODUCTION

Neural network (NN) based language models as previously proposed by [2] had been continuously reported to outperform other language modeling techniques. The best results so far yielded RNN based language models as proposed before in [9] and extended later in [10]. The RNN is very similar to approaches based on feed-forward networks, except that a recurrency between hidden and input layer is being added, allowing the hidden neurons to remember information about the entire history processed so far and thus track a potentially infinite history.

Neural networks in language modeling offer the following advantages over competing approaches: In contrary to commonly used n-gram language models, there is no neccessity of smoothing in cases of sparse training data. Due to the projection of the entire vocabulary into a small hidden layer, semantically similar words get clustered. This explains, why data sampled from the distribution defined by RNN models can contain frequent n-grams, which may have never been seen during training: Words get substituted by other words which the RNN learned to be related. Whereas no such relation could be learned by a standard n-gram model using the original sparse training data, we already showed in [1] how we can incorporate some of the improvements gained by RNN language models into systems which use just standard n-gram language models by generating a large amount of additional training data from the RNN distribution.

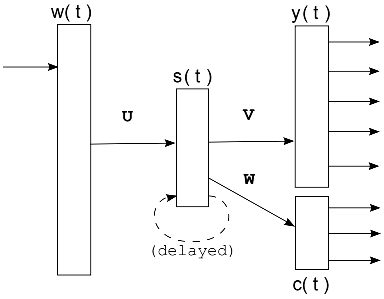

Figure 1: Simple recurrent neural network with output layer factorized by a class layer.

The RNN language model operates as a predictive model for the next word given the previous one.

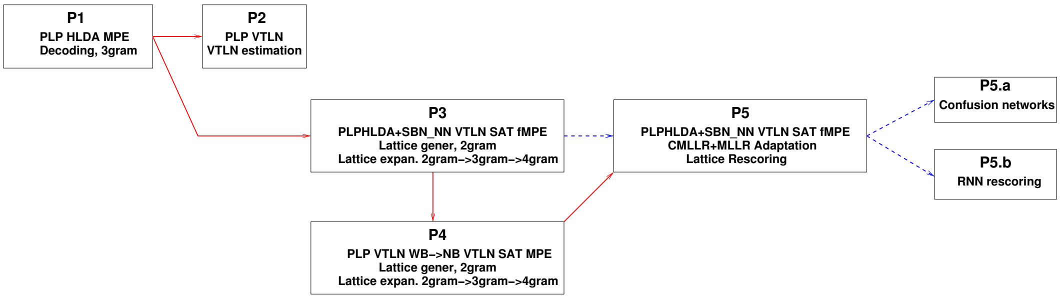

Figure 2: Structure of the Automated Speech Recognition (ASR) System.

It processes every sentence $w_1 w_2 ... w_n$ word by word, yielding the summed log-likelihood:

$$\log P(w_1 w_2 ... w_n) = \sum_{i=1}^{n} logP_{rnn}(w_i | w_1 w_2 ... w_{i-1}) \tag{1}$$

By assuming, that words can be mapped to classes, we utilize the RNN to estimate a probability distribution over classes and hence assume a unigram distribution of words within a class[1]. The utilized RNN architecture is shown in figure 1, where $s(t)$ is the hidden layer using the sigmoid and $y(t)$, $c(t)$ are the output layers using softmax activation functions, respectively. Each words is assigned a class based on its frequency (also known as frequency binning), where the number of classes is a parameter. This is done mainly to reduce computational complexity, but also leads to slight improvements in accuracy. The probability of a word w_t in class c_t given a history h can be expressed in terms of the joint probability of two distributions:

$$P(w_t|h) = P(c_t|h)P(w_t|c_t) = P(c_t|s(t))P(w_t|s(t)) \tag{2}$$

The hidden state vector $s(t)$ is assumed to encode the entire history of words processed so far. Finally, the probability of the next word can be predicted by propagation through the RNN as shown in [10].

2 SETUP

2.1 Baseline system

Our baseline speech recognition system is described in figure 2. It uses acoustic and language models from the AMIDA Rich Transcription 2009 system [7]. The original system structure has been reconfigured in order to be able to generate word lattices. Furthermore, we extended the block scheme by another complementary branch (P4) which we found beneficial in the system for NIST RT07 evaluation [8].

The entire system runs in five passes (P1-P5): Initial decoding in P1 uses PLP features with HLDA [6]. The output was used for the estimation of Vocal Tract Length Normalization (VTLN) warping factors in P2 and per-speaker CMLLR [4] adaptation in P3. In P3, MEL-filter banks over the warped spectrum were generated and forwarded through a stacked bottleneck NN (SBN_NN) [3]. The following MLLT transformations used features suitable for diagonal covariance modeling [5]. This NN based features were further concatenated with standard PLP/VTLN/HLDA 69-dimensional features. Lattices were generated using the bigram RT09 language and HMM MPE trained acoustic models. Finally, these lattices were expanded by higher order ngrams (3-gram and subsequently 4-gram) and a new ASR one-best output was decoded. In P4, PLP/VTLN based MPE models were adapted by CMLLR on a per-speaker base using the P3 ASR output. In P5, P3 lattices were rescored with

[1] It is assumed that each word belongs to exactly one class, but less classes exist than words.

adapted models using P4 output. For our experiments, we used three variants (see table 2) of our baseline system, which were identical to the baseline setup except that the original n-gram language models (RT09) have been substituted by newly build ones.

2.2 Language models

Corpus	Words	RT09	RT11	RNN/rnn
Web data	931M	✓	–	–
HUB4-LM96	152M	✓	33M	–
Fisher 1+2	21M	✓	✓	✓
Swbd+CHE	3.4M	✓	✓	✓
Meetings	2.1M	✓	✓	✓
Total	1.1G	1.1G	60M	26.5M

Model	PPL	Data
VarApx1	94.2	100M words from rnn
VarApx2	89.6	200M words from RNN
RT11	82.5	see left table
VA	82.4	interp. VarApx1+VarApx2
RT11+VA	76.6	interpolated RT11+VA
RT09	72.2	see left table
RT09+VA	69.2	interpolated RT09+VA

Table 1: Language Models - utilized corpora (left) and interpolated model perplexities (right)

In the left part of table 1 we show the corpora used for training the language models[2]. We trained two RNNs using backpropagation through time of depth 6 on a subset (AMI meetings + Fisher1/2 + Call-Home English + Switchboard) of the original training data. The small model (rnn) used 350 hidden units and 39k words, the large one (RNN) full vocabulary (65k words) and 500 hidden units. In the right part we show an overview of n-gram models used in building the derived systems in decreasing order of perplexity. We generated two training data files using the small and the large RNN language model using what we proposed earlier in [1] as variational approximation. The LMs trained on that data (VarApx1 and VarApx2) were interpolated (VarApx), which still reduced perplexity considerably. The RT11 model, which used additional 33M words of train data to increase the vocabulary by parts of the Hub4 corpus. For the RT11 and RT09 4-grams modified Kneser-Ney smoothing has been used, for all other 2-gram, 3-gram and 4-gram LMs Goodman-Turing discounting was used. The rt06seval data set (30k words) served as validation data in RNN training and for tuning interpolation weights for all n-gram models.

3 EXPERIMENTS

All experiment have been carried out using the large RNN model, which showed better performance than the small one. Furthermore, the small RNN model was not adaptable i.e. adaptation in fact degraded recognition performance.

3.1 N-best rescoring

First, the 4-gram lattices which originally served as input for the confusion network (CN) generation, were taken instead to extract n-bests. As shown in table 2, column 1+2, 1-best extraction from these lattices came already very close to the baseline. The RNN model processed each sentence in the n-best list and estimated an updated log-likelihood score for each n-best hypothesis s:

$$\log P(s) = n \cdot wp + \sum_{i=1}^{n} asc_i + lms \sum_{i=1}^{n} \log P_x(w_i|h) \tag{3}$$

where n is the number of words, wp is the word insertion penalty, asc_i is the acoustic score for word w_i, h the history $w_1 ... w_{i-1}$ and lms the language model scale applied in the generation of the input lattices. P_x is a placeholder for the combined probability estimate of standard 4-gram and RNN

[2] The web data actually consists of four separate data sets described more thoroughly in [8].

Model	*4-gram*	*CN*	*RNN*	*Adapt*	*#n-grams*
RT09	20.3	20.2	19.6	19.1	51.2M
RT09+VA	20.4	20.2	–	–	76.7M
RT11	22.2	22.0	20.7	20.4	14.4M
RT11+VA	21.5	21.3	20.5	20.2	46.5M

Table 2: Baseline and derived systems and their word error rates (WER)

models, which was either obtained by linear interpolation (eq. 4) or by linear interpolation of log likelihood estimates (eq. 5):

$$P_x(w_i|h) = \lambda P_{rnn}(w_i|h) + (1-\lambda)P_{ngram}(w_i|h) \tag{4}$$

$$\log P_x(w_i|h) = \lambda \log P_{rnn}(w_i|h) + (1-\lambda)\log P_{ngram}(w_i|h) \tag{5}$$

The interpolation weight λ for the RNN model usually yielded optimal performance around 0.75, except for the RT09 system, whose language model is trained on much more data (mostly the retrieved web data shown in table 1) than the RNN models, where it still was around 0.5. It might be noteworthy, that performance using lattices generated from the RT11+VA system appeared to be less sensitive to changes of λ than when lattices from the RT11 system have been used. The results of the following 1-best extraction for the various system setups can be seen in the forth column of table 2. In overall, no consistent differences between both interpolation methods could be observed, hence we report the results for linear interpolation only.

3.2 Adaptation

The effect of the learning rate setting on adaptation has been examined using the RT09 system, where the largest improvement could be observed. As shown in figure 3, we ran an exhaustive grid search over a wide range of the learning rate, and observed the PPL on the recognition output (dashed red curve) and on the validation data (blue curve). The initial learning rate in RNN training has been 0.1, and the X-axis denotes a multiplicative factor used to set the learning rate for one iteration of retraining. Three markers have been placed at learning rates for which a second RNN rescoring pass was run to compute the improved WER. In the first case (optimal valid) the learning rate is set to minimize PPL on validation data, and in the third case (optimal 1-best) PPL on the recognition output is minimized. Because performances differed considerably, and the learning rates ranged far from each other, WER was additionally computed for the learning rate in between (initial valid), where validation PPL again exceeds the PPL obtained by the unadapted RNN. Similarly, we determined a learning rate for adaptation using jack-knifing. We split the recognition output into two equally sized parts and determined two learning rates, whose average led to a slightly pessimistic guess of the learning rate factor of about 10^{-2}.

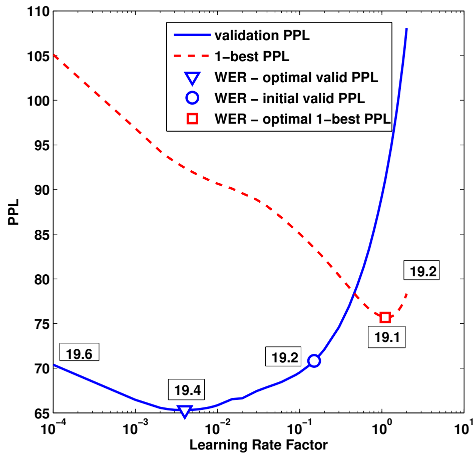

Figure 3: Adaptation learning rate and final WER.

4 CONCLUSIONS

RNNs show great potential in case just limited amount of training data is available. A light-weight system finally reaches similar performance, although the baseline system uses about 20 times more data. But also by combination with standard n-gram models trained on the entire data good improvements are observable. Here, RNN adaptation yields up to 50% of all observed improvement. Variational approximation provides a simpler way than web data retrieval to build better n-gram models, although improvements are likely to be smaller. Using such models leads to slightly better performance even after RNN rescoring. Speeding up training times would allow the exploitation of the entire data available, and speeding up the rescoring process would make RNN models more interesting for application in light-weight ASR systems.

ACKNOWLEDGEMENTS

This work was partly supported by European project DIRAC (FP6-027787), Grant Agency of Czech Republic project No. 102/08/0707, Czech Ministry of Education project No. MSM0021630528 and by BUT FIT grant No. FIT-10-S-2. Also, we would like to thank Martin Karafiát for fruitful discussions and help with the LVCSR setup.

REFERENCES

[1] A. Deoras, T. Mikolov, S. Kombrink, M. Karafiát and S. Khudanpur. Variational Approximation of Long-Span Language Models in LVCSR. In *IEEE Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, CZ, May 2011. Accepted for publication.

[2] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, J. K, T. Hofmann, T. Poggio, and J. Shawe-taylor. A neural probabilistic language model. In *Journal of Machine Learning Research*, 2003.

[3] F. Grezl, M. Karafiát and L. Burget. Investigation into bottle-neck features for meeting speech recognition.

[4] M. Gales. Maximum Likelihood Linear Transformations for HMM-based Speech Recognition. *Maximum Likelihood Linear Transformations for HMM-based Speech Recognition, Tech. Report, CUED/FINFENG/TR291, Cambridge University*, 1997.

[5] R. Gopinath. Constrained Maximum Likelihood Modeling With Gaussian Distributions for Classification. In *Proc. ICASSP*, Seattle, USA, 1998.

[6] N. Kumar and A.G. Andreou. Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. pages 283–297, 1998.

[7] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiát, D.v. Leeuwen, M. Lincoln and V. Wan. The 2007 AMI(DA) system for meeting transcription. In *Proc. Rich Transcription 2007 Spring Meeting Recognition Evaluation Workshop*, Baltimore, Maryland USA, May 2007.

[8] T. Hain, L. Burget, J. Dines, N.P. Garner, A.H. El, M. Huijbregts, M. Karafiát, M. Lincoln and V. Wan. The AMIDA 2009 Meeting Transcription System. In *Proc. of INTERSPEECH 2010*, volume 2010, pages 358–361. International Speech Communication Association, 2010.

[9] T. Mikolov, M. Karafiát, L. Burget, J. Černocký and S. Khudanpur. Recurrent neural network based language model. In *Proc. of INTERSPEECH 2010*, number 9, pages 1045–1048. International Speech Communication Association, 2010.

[10] T. Mikolov, S. Kombrink, L. Burget, J. Černocký and S. Khudanpur. Extensions of Recurrent Neural Network Language Models. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, CZ, May 2011. Accepted for publication.