Neural Network Language Models for Low Resource Languages

Ankur Gandhe, Florian Metze, Ian Lane

LTI, Carnegie Mellon University
{ankurgan,fmetze,ianlane}@cs.cmu.edu

Abstract

For resource rich languages, recent works have shown Neural Network based Language Models (NNLMs) to be an effective modeling technique for Automatic Speech Recognition, out performing standard n-gram language models (LMs). For low resource languages, however, the performance of NNLMs has not been well explored. In this paper, we evaluate the effectiveness of NNLMs for low resource languages and show that NNLMs learn better word probabilities than state-of-the-art n-gram models even when the amount of training data is severely limited. We show that interpolated NNLMs obtain a lower WER than standard n-gram models, no mater the amount of training data. Additionally, we observe that with small amounts of data (approx. 100k training tokens), feed-forward NNLMs obtain lower perplexity than recurrent NNLMs, while for the larger data condition (500k-1M training tokens), recurrent NNLMs can obtain lower perplexity than feed-forward models.

1. Introduction

Statistical Language modeling is an important component of many natural language processing applications, including spelling correction, machine translation and automatic speech recognition. The majority of previous work has focused on n-gram back-off language models due to their low computational complexity and effectiveness. In particular, modified Kneser-Ney smoothed models [1] have been shown to achieve the best performance[2] within n-gram models. In recent years, however, a variety of novel techniques for language modeling have been proposed, including maximum entropy language models [3], random forest language models [4], and neural network language models ([5],[6]). Of these, neural network language models have been shown to perform the best in automatic speech recognition tasks[7].

Neural network language models were re-introduced recently in [8] to tackle the curse of dimensionality suffered by n-gram models and allow continuous representation of words. Rather than learn the probability distribution over words, initial work, such as [9], used neural networks to label words within a sentence a part-of-speech tag. The models were subsequently modified for application to automatic speech recognition in [5],[10], and proved to perform better than n-gram back-off models. More recently, recurrent NNLMs were proposed in [6],[11], and shown to obtain higher speech recognition accuracy than feed-forward NNLMs. However, they are difficult to train using standard back propagation through time ([12]) approaches because of the large size of output layer. [11] propose a class-based approach to reduce training time, where the faster computation comes at the cost of slight loss in performance. In this paper, we perform an empirical study of performance of feed-forward NNLMs focusing on low-resource languages and transcriptions of conversational speech where the training data is severely limited.

2. Feed-Forward Neural Network Language Model

In this paper we adopt the techniques introduced in [5] to train the feed-forward NNLMs (ff-NNLMs) and follow the notation used in their paper. Figure 1 shows the architecture of our model. The ff-NNLM, is an n-gram language model where the posterior probability distribution of the following word is computed for a given $n-1$ history using a neural network model. Each word in the model's vocabulary is represented as a sparse vector $S_{1\times N}$, where only the j^{th} column is 1 for the word w_j (1-of-N). The probability of current word is then modeled to depend on the n word history, ie :

$$p(w_j|history) = p(w_j|w_{j-1}, w_{j-2}, ...w_{j-n+1})$$

History is represented in the neural network by multiple sparse vectors at the input layer. Each sparse word vector is mapped linearly to a continuous word projection by a $N \times P$ weight matrix (F). The resulting layer formed by concatenating the continuous word vectors is called the *projection layer*. The number of units in the projection matrix determines the number of features used to represent each word. The second layer is the *hidden layer* that uses hyperbolic tangent function as a non-linearity. The *output layer* has N units (equal to the vocabulary size of the model). At this layer a softmax function is applied to the activation of each unit to produce posterior probabilities.

Let c_k represent the projections for the history, d_j be the activations of the hidden units and o_i be the outputs. Then, the $p(w_j|h_j)$ is given by:

$$d_j = tanh\left(\sum_l m_{jl}c_l + b_{1_j}\right) \quad (1)$$

$$o_i = \sum_j v_{ij}d_j + b_{2_i} \quad (2)$$

$$p_i = e^{o_i} / \sum_{k=1}^{N} e^{o_k} \quad (3)$$

where m_{ji} , b_{1_j} correspond to matrix M and bias B_1 between *projection* and *hidden layer*, and v_{ij} , b_{2_j} correspond to the matrix V and bias B_2 between *hidden* and *output layer*. All the weight matrices in ff-NNLMs are trained using standard back-propagation algorithm with an adaptive learning rate. Our optimization function is cross entropy of the data and a "weight

Copyright © 2014 ISCA

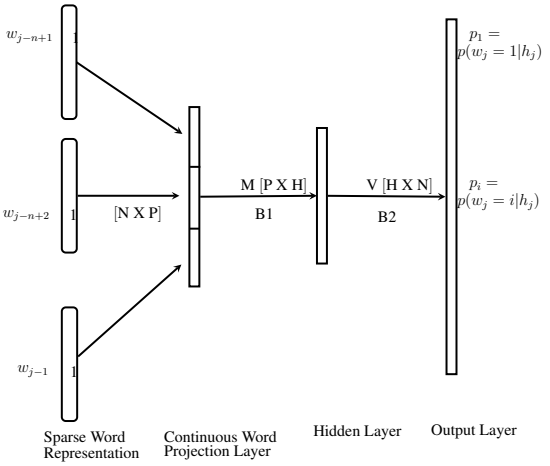

Figure 1: Graphical Model of feed-forward neural network.

decay" L2 regularization, given by:

$$E = \sum_{i=1}^{N} t_i log\, p_i + \beta\big(\sum_{jl} m_{jl}^2 + \sum_{ij} v_{ij}^2\big) \qquad (4)$$

where t_i is the expected activation and p_i is the probability assigned to word i and β is the regularization constant. It can be shown that the outputs of a neural network trained in this manner converge to the posterior probabilities and directly minimizes the perplexity. Neural networks are well suited for conversational language models because:

- In low resource conditions they do not suffer from the curse of dimensionality and can be used to obtain posterior probabilities for any history of words.
- Due to the size of the output layer, the most computationally expensive step when training NNLMs is the back propagation of errors from output layer to hidden layer. Because of the small vocabulary size in our task, the complexity is low and hence the training time is also small.

2.1. Recurrent Neural Network Language Model

Recurrent neural networks were proposed in [6] and have been shown to be effective for language modeling in speech recognition for resource rich languages such as English and Mandarin Chinese. Figure 2 shows a simple recurrent NNLM, with the transforming functions between first and hidden layer, and hidden and output layer being the same as ff-NNLM. The major differences are in representation of each word, which are kept as sparse 1-of-N vector $N_{1\times N}$ w(t), and the first layer, $x(t)$, which is formed by concatenating this word vector with the hidden layer of the previous step $s(t-1)$. This is followed by the hidden layer $s(t)$ and a final output layer $y(t)$. The details of the process are described in [11]. In this paper, we report perplexity results on r-NNLM using the class-based model as described in [11].

3. Experimental setup

3.1. Language models

Both standard n-gram and ff-NNLMs estimate the probability of next word based on a recent history of $n-1$ words. For most speech recognition tasks, a history of two words, known as *tri-gram* model, is sufficient [2]. For our baseline experiments, we trained modified Kneser-Ney (mKN) smoothed trigram language model using the SRILM toolkit [14]. Speech recognition was performed using the Janus recognition toolkit [15] which can apply n-gram language models, NNLMs and combined n-gram+NNLM models during both speech recognition decoding and lattice re-scoring. Using the model structure proposed in [5] and[7], we trained ff-NNLMs using a projection layer of 100 and a hidden layer of 150 units. To train the recurrent NNLMs, we used the rnnlm toolkit [16]. Given the limited vocabulary sizes and training data available for both the LimitedLP and FullLP tasks in BABEL, we used 30 units within the hidden layer. In addition to the recurrent neural network, it also has direct connections from input to output layer. For training the class-based r-NNLMs, we used 100 classes for all languages.

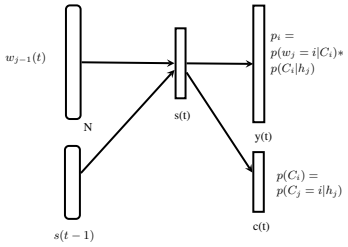

Figure 2: Graphical Model of recurrent neural network.

3.2. Language model training data

The work described in this paper focused on using NNLMs for low resource languages. We evaluate the performance of these methods across five languages (Cantonese, Pashto, Tagalog, Turkish and Vietnamese) using the LimitedLP and FullLP resources made available to participants within phase one the IARPA BABEL project[1]. The LimitedLP resources described in table 1 consist of transcriptions of 10hrs or recorded conversations. We also perform experiments using the FullLP resources which comprises of transcripts of 100 hours of training data, shown in table 2. Although 100 hours of training data is still generally considered a "low resource" task, in this paper we use it as an upper bound to compare the performance of the language modeling techniques with varying data sizes.

Language	Tagalog	Pashto	Cantonese	Turkish	Vietnamese
Sentences	11.5k	8.6k	10.3k	10.3k	10.1k
Tokens	88k	113k	110k	71k	117k
Vocabulary	5581	6185	5986	10186	3218

Table 1: Training data available in the LimitedLP condition

Language	Tagalog	Pashto	Cantonese	Turkish	Vietnamese
Sentences	92.9k	69.6k	81.1k	81.5k	78.6k
Tokens	585k	881k	871k	555k	918k
Vocabulary	21k	17.6k	18.5	38.3k	6.2k

Table 2: Training data available in the FullLP condition

[1]This effort uses the IARPA Babel Program language collection release IARPA-babel101-v0.4c, IARPA-babel104-v0.4Y, IARPA-babel105-v0.4-rc, IARPA-babel106-v0.2f, IARPA-babel107-v0.7

4. Experiment Results

To evaluate the performance of the different language model techniques described earlier in the paper, we report results using two metrics on a 10 hr development set provided with the BABEL corpora : a) Perplexity (PPL): $2^{\sum_x p(x)}$, and b) Word Error Rate (WER)

4.1. Variation with training data size

We began by evaluating the performance of the tri-gram back-off model and tri-gram ff-NNLM and as the training data size is increased from 5k sentences to 93k sentences for Tagalog. The perplexities for these different models are shown in table 3 for different amounts of training data. Even when trained on very limited resources, the ff-NNLM obtain lower perplexities than the baseline mKN-LM model. The last row of table 3 show the performance (perplexity) when the NNLM is interpolated with the baseline mKN-LM. When interpolated with the mKN-LM the reduction in perplexity is significant, 12% relative.

Sentences	5k	10k	25k	50k	93k
Tokens	33k	65k	156k	320k	584k
Vocabulary	3312	5236	9140	14600	21075
mKN	101.5	106.9	111.0	114.9	116.8
ff-NN	100.6	106.5	109.5	109.5	110.8
ff-NN+mKN	**90.0** (11%)	**94.5** (11%)	**97.9** (11%)	**100.6** (12%)	**102.3** (12%)
WER: mKN	62.9	61.3	59.7	58.4	57.5
WER: ff-NN +mKN	62.2 (0.7)	60.5 (0.8)	58.8 (0.9)	57.6 (0.8)	56.7 (0.8)

Table 3: Perplexity and WER for varying amounts of Tagalog training data: modified (mKN), feed-forward Neural Network (ff-NN) and interpolated models (ff-NN+mKN)

As the combination of ff-NNLM + mKN-LM obtained the lowest perplexity of the models evaluated, we evaluated the effectiveness of these models in terms of speech recognition accuracy. Speech recognition was performed using the Janus decoder and the selected language models were used both during decoding and lattice re-scoring. The Gaussian Mixture-Model (GMM)-based Acoustic Model applied during decoding was trained using the FullLP corpora described above (100 hours of training data). Neural Networks were trained to extract Bottle-Neck-Features (BNF) using an approach similar to that described in [17]. The speech recognition performance in terms of WER (Word Error Rate) for these five models are shown in table 3. In this evaluation we observed that the improvement in speech recognition accuracy was consistent across different amounts of training data.

4.2. LimitedLP condition

To verify that the neural network language models perform well across different languages, we evaluated the perplexity of a back-off n-gram model and ff-NNLM across all five languages, as shown in table 4. In the LimitedLP condition (approx. 100k training tokens) we observed that the NNLM performed better on all languages except Turkish and Vietnamese. When combined with the baseline mKN-LM, the NNLMs significantly lower perplexity than the baseline model across all five languages. Compared to the baseline mKN-LM an average reduction of perplexity of 9.4% was obtained for the ff-NNLM after interpolation. One possible reason that the individual NNLMs may perform poorly for Turkish is that the very large vocabulary size and limited training data will significantly skew the distribution of training examples, resulting in a poorly trained model, especially for singleton words. Table 4 also shows the

	Tagalog	Pashto	Cant.	Turkish	Viet.
mKN	108.1	152.6	83.2	242.3	147.1
ff-NN	106.4	149.5	80.7	296.9	152.1
ff-NN+mKN	**95.7** (11%)	**134.5** (11%)	**74.6** (10%)	**233.4** (4%)	**132.9** (9%)
WER: mKN	67.3	73.2	69.7	71.7	68.5
WER: ff-NN +mKN	66.9 (0.4)	72.8 (0.5)	69.0 (0.7)	71.5 (0.2)	68.1 (0.4)

Table 4: Perplexities and WER for five different languages in LimitedLP condition: modified (mKN), feed-forward Neural Network (ff-NN) and interpolated models (ff-NN+mKN)

speech recognition performance (CER for Cantonese and WER for other languages) using the LimitedLP resources. BNFs and GMM-based AMs were trained on the LimitedLP resources (approx 10 hours of training data) and decoding was performed applying the baseline mKN-LM and or an combined of ff-NNLM and mKN-LM model. Similar to the Tagalog case (section 4.1) the speech recognition accuracy when the ff-NNLM is introduced is consistency better across all languages than the the baseline mKN-LM case. From the five languages evaluated here the improvement for Turkish is minimum, which is also observed in the minimal perplexity improvement observed in the table above. Tagalog obtained the largest absolute improvement (0.7% in WER). Despite the very different languages evaluated, the ff-NNLM performed well across all of them without any change in the network architecture.

4.3. FullLP Condition

In addition to evaluating the performance of NNLMs in low resource conditions (approx. 100k training tokens) we also evaluated the performance of the feed forward NNLMs on the FullLP condition, where models are trained with much larger resources (500k - 1M training tokens). The vocabulary sizes and training data available for the FullLP task are listed in table 5. The comparison of perplexity of baseline trigram back off models as compared to the ff-NNLM across the five evaluation languages is shown in table 5. Similar to the LimitedLP case the performance gained on using only neural networks is quite good, except for Turkish. Compared to the baseline mKN-LM, an average reduction of perplexity of 7.7% was obtained for the ff-NNLM after interpolation.

	Tagalog	Pashto	Cant.	Turkish	Viet.
mKN	116.8	145.8	83.0	295.6	120.2
ff-NN	110.8	156.3	81.8	346.5	120.9
ff-NN+mKN	**102.3** (12%)	**142.3** (2%)	**75.5** (9%)	**278.9** (5%)	**108.9** (9%)
WER: mKN	57.5	62.6	57.5	59.2	53
WER: ff-NN +mKN	56.7 (0.8)	61.7 (0.9)	56.7 (0.8)	58.9 (0.3)	52.3 (0.7)

Table 5: Perplexities and WER for five different languages in FullLP condition: modified (mKN), feed-forward Neural Network (ff-NN) and interpolated models (ff-NN+mKN)

For Turkish and Pashto, the ff-NNLM alone is worse than

back-off models. The number of singletons in Turkish are very high compared to other languages (about 60% of the vocabulary), resulting in poorly trained word feature vectors and a worse performance. Handling the Turkish morphology before training the NNLMs might be a solution to this problem, which we plan to investigate as a part of future work.

The ASR experiments using the above mKN-LM and ff-NNLM+mKN-LM combination are performed on acoustic models built on 100 hours of data. Table 5 shows the performance on the different languages. Comparing with table 4, the improvements on Tagalog and Vietnamese are more with 100 hour data models than with 10 hour data models. But the improvement on moving from mKN-LM to NNLMs is certainly clear for all the languages.

4.4. Comparison of Recurrent and Feed-Forward NNLM

From the previous sections, we can conclude that the ff-NNLMs perform better than standard n-gram models in low resource conditions. We also trained recurrent-NNLMs based on the method described in section 2.1 using the toolkit in [16] and compared the performance on the Limited and FullLP dataset. In table 6 and 7, we observe that with small amounts of data (approx. 100k training tokens), feed-forward NNLMs obtain 3% lower perplexity than recurrent NNLMs, while for the larger data condition (500k-1M training tokens), on average recurrent NNLMs obtain slightly (0.3%) lower perplexity than feed-forward models.

	Tagalog	Pashto	Cant.	Turkish	Viet.
mKN	108.1	152.5	83.2	242.4	147.1
ff-NN+mKN	95.7 (11%)	134.5 (11%)	74.6 (10%)	233.4 (4%)	132.9 (9%)
r-NN+mKN	97.3 (10%)	141.1 (7%)	77.2 (7%)	234.3 (3%)	141.8 (3%)

Table 6: Perplexities of ff-NNLM and recurrent-NNLM for five different languages - LimitedLP condition

	Tagalog	Pashto	Cant.	Turkish	Viet.
mKN	116.8	145.8	83.0	295.6	120.2
ff- NN+mKN	102.4 (12%)	142.3 (2%)	75.5 (9%)	278.9 (5%)	108.9 (9%)
r-NN+mKN	103.6 (11%)	132.4 (9%)	76.9 (7%)	274.4 (7%)	113.9 (5%)

Table 7: Perplexities of ff-NNLM and recurrent-NNLM for five different languages - FullLP condition

5. Conclusion and future work

In this paper, we evaluated the performance of ff-NNLMs on very limited amount of language model training data. The improvements were consistent across different training data sizes and both neural network techniques performed well in terms of perplexity and word error rate. The relative improvement from back-off models to neural networks increases with the size of training data. For small amounts of data, ff-NNLMs perform better while for larger data size, ff-NNLMs need to be interpolated with mKN-LM. The absolute improvement in WER was upto 0.7% in WER when the limited-data training set was used and up to 0.8% when full-data training set was used. The trained ff-NNLM models are easy to train compared to the recurrent-NNLMs, and yield significant improvements. Compared to recurrent-NNLMs, the ff-NNLMs has a better relative improvement when the training data was small.

Neural networks offer an easy way of adding more features to capture this abstractness into the model. In future work, we intend to use additional context information to vary the probabilities at the output layer of the NNLMs. NNLMs performed poorly on Turkish, which has rich morphology and low amounts of training data. It can probably benefit to add morphological information in the neural network input layer to model words with the same root more robustly.

6. Acknowledgments

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

7. References

[1] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling", *Proc. ICASSP* vol 1. pp. 181-184, 1995.

[2] S. F. Chen and J. T. Goodman, "An empirical study of smoothing techniques for language modeling", *Proc. Computer Speech and Language*, pp. 359-394, 1999.

[3] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling", *Proc. Computer Speech and Language*, pp. 187228, 1996.

[4] Peng Xu and F. Jelinek, "Random forest in language modeling", *Proc. EMNLP*, pp. 325332, 2004.

[5] H. Schwenk and J. L. Gauvain, "Neural Network Language Models for Conversational Speech Recognition", *Proc. ICSLP*, pp. 1215-1218, 2004.

[6] T. Mikolov, M. Karafiát, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model, *Proc.Interspeech*, pp. 1045-1048, 2010.

[7] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocký, "Empirical Evaluation and Combination of Advanced Language Modeling Techniques" , *Proc. Interspeech*, pp. 605-608, 2011.

[8] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A Neural Probabilistic Language Model", *Proc. J. Machine Learning Research*, vol 3, pp. 1137-1155, 2003.

[9] R. Miikkulainen, and M. G. Dyer, "Natural Language Processing with Modular PDP Networks and Distributed Lexicon", *Proc. Cognitive Science*, vol. 15, No. 3, pp 343-399, 1991.

[10] H. Schwenk, "Continuous space language models", *Proc. Computer Speech and Language*, pp. 492-518, 2007.

[11] T. Mikolov, S. Kombrink, L. Burget, J. Cernocký, and S. Khudanpur, "Extensions of recurrent neural network language model", *Proc. ICASSP*, pp. 5528-5531, 2011.

[12] D. E. Rumelhart, G. E. Hinton, R. J. Williams, “Learning representations by back-propagating errors, *Proc. Nature*, pp. 533-536, 1986.

[13] Boden Mikael “A Guide to Recurrent Neural Networks and Back-propagation”, *In Dallas project*, 2001.

[14] A. Stolcke, “SRILM - an extensible language modeling toolkit”, *Proc. ICSLP*, vol 2, pp. 901-904, 2002.

[15] H. Soltau, F. Metze, C. Fgen, and A. Waibel, “A one-pass decoder based on polymorphic linguistic context assignment”, *Proc. ASRU*, pp. 214-217, 2001.

[16] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocký, “ RNNLM - Recurrent Neural Network Language Modeling Toolkit” , *Proc. ASRU Demo session*, 2011.

[17] J. Gehring, Y. Miao, F. Metze and A. Waibel, “Extracting deep bottleneck features using stacked auto-encoders”, *Proc. ICASSP* pp. 3377 - 3381, 2013.