EXPLOITING DIFFERENT WORD CLUSTERINGS FOR CLASS-BASED RNN LANGUAGE MODELING IN SPEECH RECOGNITION

Minguang Song[1], *Yunxin Zhao*[1], *Shaojun Wang*[2]

[1]Department of Computer Science, University of Missouri, Columbia, MO, USA
[2]Department of Computer Science and Engineering, Wright State University, Dayton OH, USA

msong@mail.missouri.edu, zhaoy@missouri.edu, swang.usa@gmail.com

ABSTRACT

We propose to exploit the potential of multiple word clusterings in class-based recurrent neural network (RNN) language models for ensemble RNN language modeling. By varying the clustering criteria and the space of word embedding, different word clusterings are obtained to define different word/class factorizations. For each such word/class factorization, several base RNNLMs are learned, and the word prediction probabilities of the base RNNLMs are then combined to form an ensemble prediction. We use a greedy backward model selection procedure to select a subset of models and combine these models for word prediction. The proposed ensemble language modeling method has been evaluated on Penn Treebank test set as well as Wall Street Journal (WSJ) Eval 92 and 93 test sets, where it improved test set perplexity and word error rate over the state-of-the-art single RNNLMs as well as multiple RNNLMs produced by varying RNN learning conditions.

Index Terms— recurrent neural network, language modeling, model combination, speech recognition

1. INTRODUCTION

Language modeling plays an essential role in spoken and natural language processing. With the advances in deep learning, neural network language models (NNLM) [1][2] have gained popularity. Among the NNLMs, Recurrent Neural Network Language Model (RNNLM) [3][4] is becoming widely adopted in recent years, which greatly outperforms traditional n-gram and some more complex language models [5][6]. RNNLM has been successfully applied in different language processing tasks such as speech recognition [3][4][7][8], spoken language understanding [9][10], machine translation, and information extraction [11]. Nowadays efforts are being made to improve RNNLMs [12].

An RNNLM can efficiently represent more complex linguistic patterns than shallow neural networks [3]. An RNNLM is characterized by employing long word histories through a recurrent hidden-to-input connection, using word embedding features, and producing smooth word prediction probabilities. To reduce structural and computational complexities for large vocabulary modeling, class-based language modeling is commonly used in RNN in a similar way as in structured output layer NNLM [13], where the output layer is factorized to two parts: nodes for probabilities of word classes, and nodes for probabilities of words conditioned on the classes. To cluster words into classes, the methods of frequency binning [4] and Brown clustering [14][15] are often used, with the objective of delivering high-quality single RNNLMs. While class-based language modeling helps reduce complexity, the potential of varying the class structures for ensemble language modeling has not been well explored for RNNLM.

In the current work, we propose to exploit the benefit that multiple word clusterings may offer in class-based language models for ensemble RNN language modeling. By varying the clustering criteria or the space of word embedding, different word clusterings can be obtained to define different word/class factorizations. For each such word/class factorization, a base RNNLM is learned, and the word prediction probabilities of the multiple base RNNLMs are then combined to form an ensemble prediction. The ensemble RNNLM will be more accurate and robust than those of the individual RNNLMs if enough diversity exists among the base LMs. This proposed approach is inspired by our previous work [16][17][18][19] in ensemble speech acoustic modeling, where random-forests of phonetic decision trees were used to define multiple triphone-state clustering structures, and the likelihood scores of an observation vector is combined over the multiple state-tying structures in speech decoding search. Significant word error reductions were obtained using this approach in both Gaussian-mixture-density Hidden Markov Model (HMM) and deep-neural-network-HMM based speech recognition experiments. It is worth noting that although combining multiple RNNLMs were previously reported in [20], the different RNNLMs in [20] were generated by varying the RNN learning initialization conditions. Other efforts focused on combining different types of language models, such as combining n-gram LM with structural LM [5] and RNNLM [3][4][20].

In this work, we investigate using several clustering methods to define word classes, including word frequency binning, part-of-speech (POS) tagging, Brown clustering, as well as using k-means clustering on word embedding vectors that are derived from different language processing tasks. For combining multiple RNNLMs, we investigate a backward model selection method to remove redundant models in a greedy fashion while keeping the good models for combination. We evaluate the proposed language modeling method on the test set of the Penn Treebank and two test sets of the 20k-word WSJ tasks, reporting performance on test set perplexity (PPL) and word error rate (WER). Additionally, we measure diversity between different pairs of language models by comparing word partition structures induced by different clustering methods and report our finding.

The rest of this paper is organized as follows. Section 2 briefly reviews RNNLM. The clustering strategies used in the current work and the method of model combination are described in Section 3. Experimental results are provided and analyzed in Section 4. A conclusion is made in Section 5.

2. RECURRENT NEURAL NETWORK LM

The structure of a class-based RNN is illustrated in Fig. 1. The RNN has an input layer x, a hidden layer s (also called context layer), and an output layer y. At time t, the input to the RNN is $x(t)$, the output is $y(t)$ and $c(t)$, and the state of the hidden layer is $s(t)$. The input vector $x(t)$ is formed by concatenating the one-hot vector of the current word $w(t)$ and the previous state $s(t-1)$. In [4], the output layer is factorized to two parts: $c(t)$ for the word classes, and $y(t)$ for words conditioned on the classes, where words are assigned to classes according to their frequency proportions. The probability of a word w_i is approximated as a product of the probability of the class to which w_i belongs and the class conditional probability of w_i.

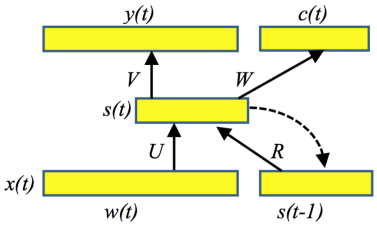

Fig. 1: Class-based RNN

The RNN computations are defined by the following equations:

$$s_j(t) = f(\sum_i w_i(t)u_{ji} + \sum_k s_k(t-1)r_{jk}) \quad (1)$$

$$c_l(t) = g(\sum_j s_j(t)w_{lj}) \quad (2)$$

$$y_c(t) = g(\sum_j s_j(t)v_{cj}) \quad (3)$$

$$\begin{aligned} P(w_i|s(t)) &= \sum_j P(c_j|s(t))P(w_i|c_j, s(t)) \\ &\approx P(c_{j^*}|s(t))P(w_i|c_{j^*}, s(t)) \end{aligned} \quad (4)$$

where in the 2nd line of (4) it is assumed that a word (w_i) belongs to only one class (c_{j^*}), and $f(z)$ and $g(z)$ are the sigmoid and the softmax functions, respectively:

$$f(z) = \frac{1}{1+e^{-z}}, g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}} \quad (5)$$

The RNNLM is trained with the algorithm of truncated backpropagation through time (BPTT). The network thus learns to remember information for several time steps in the hidden layer.

3. WORD CLUSTERING AND MODEL COMBINATION

3.1. Clustering Methods

We investigate the following strategies for word/class factorizations:

- Frequency binning [4]: In this method, words are first sorted by their unigram frequencies. They are next assigned to bins sequentially to give the individual bins relatively even sums of word frequency.
- Brown clustering [14][21]: This method is formulated for statistical n-gram models and it delivers a binary-tree structured hierarchical clustering of words based on the contexts in which they occur, where each internal node of the tree defines a word cluster in that subtree.
- Part-of-speech (POS) tagging: We develop this method to incorporate coarse syntax information into word/class factorization. First, the Stanford parser [22] is used to label training words by their tags which define the initial word clusters, where a word with two or more tags is assigned to its most frequent one. Next, the clusters larger than a predefined size are randomly broken into smaller ones so as to generate a specified number of word classes.
- K-means clustering on Continuous Bag-of-Words (CBOW) word embedding: The CBOW model [23] predicts a word according to its context. First we learn the CBOW word embedding vectors with a dimensionality of 50. Then k-means clustering is performed on the learned vectors to obtain the word classes.
- K-means clustering on Continuous Skip Gram (CSG) word embedding: The CSG model [23] is similar to CBOW model, but it reverses the input and output of CBOW and predicts the context according to word. Again, k-means clustering is performed on the learned vectors to get the word classes.

3.2. Model Combination through Model Selection

We adapt a backward feature selection method for our task of model selection. The selection is carried out on a validation dataset D using the criterion of word error rate minimization. First, all K models, $M_1, M_2, ..., M_K$, are included in the model set $S(K)$, and these models are uniformly combined, yielding a WER of $E(K)$ on D. Next, each model is tentatively removed from the model set, and the remaining models are again combined to evaluate the WER on D. The model whose tentative removal yielding the lowest WER is then formally removed, and the reduced model set becomes $S(K-1)$ with the WER of $E(K-1)$. The procedure iterates so that one model is removed at a time until there is no further improvement on WER or only one model is left. The overall procedure for the model selection method consists of the following four steps:

(1) Initialization: Take $S(K) = \{M_1, M_2, ..., M_K\}$ and compute $E(K)$ on D by using $S(K)$. Set $k = K$.

(2) For $i = 1$ to k, form $S_i(k-1) = S(k) - M_i$ and compute $E_i(k-1)$ on D by using $S_i(k-1)$.

(3) Find i^* and set $E(k-1) = E_{i^*}(k-1)$, where

$$i^* = \underset{i}{\operatorname{argmin}}\, E_i(k-1)$$

(4) If $E(k-1) >= E(k)$ or $k = 2$, stop. Otherwise $S(k-1) = S(k) - M_{i^*}$, $k := k-1$, and go back to Step (2).

Note that $E(k)$ can be any relevant measure of model quality, including PPL.

4. EXPERIMENT RESULTS

4.1. Experimental Setup

The RNNLM toolkit [24] was used to train the RNNLMs on Penn Treebank and Wall Street Journal (WSJ) data sets. The vocabulary sizes were 10k and 20k for the two corpora, respectively. In our experiment, the size of the hidden layer was fixed to 200 for all models, and the number of classes was kept around 100 for all word clustering methods. For each type of word clustering, five RNNLMs were trained with random weight initializations. For comparison, the RNNLMs with the same word clustering but different weight initializations were also combined, similar with the work of [20] on combining frequency binning based class LMs. In the WSJ speech recognition task, the Kaldi toolkit [25] was adopted to generate the word lattices and n-best sentence hypotheses. A trigram language model was trained with the SRILM toolkit for each dataset [26].

4.2. Penn Treebank Results

Penn Treebank corpus is one of the most widely used data sets for language modeling evaluation. It includes 929k training words, 73k validation words, and 82k test words. The vocabulary size is 10k.

As discussed in Section 3.1, we considered 5 different word clustering methods to train 5 types of class-based RNNLMs, and for each clustering method we used 5 different random weight initializations to obtain 5 RNNLMs. We first provide in Table 1 the average and standard deviation of PPLs for each class RNNLM as well as the trigram LM. It is seen that the average PPLs of the RNNLMs varied from 128.3 to 135.8, which amounted to 11.2% to 16.1% reductions relative to the PPL of the trigram LM.

	Mean PPL	STD
Trigram LM	153.0	
Frequency RNNLM	135.8	0.39
Brown RNNLM	128.3	1.52
CBOW RNNLM	129.7	1.10
CSG RNNLM	130.5	0.39
Tag RNNLM	130.9	0.61
All RNNLMs	131.0	2.65

Table 1: Penn Treebank test set PPLs by individual LMs

We next linearly combined the five RNNLMs of different weight initializations in each word clustering by averaging the output word probabilities of the RNNLMs:

$$P(w) = \frac{1}{N}\sum_{j=1}^{N} P_j(w) \tag{6}$$

where $P_j(w)$ is the word probability from the jth LM. The PPL results of the combined models are given in Table 2. We further considered combining five RNNLMs with different types of word clustering. Specifically, we randomly chose five RNNLMs, one from each class LM, to form a combined LM, and repeated the procedure 8 times to generate 8 combined LMs. The mean on the PPLs of the 8 combined LMs is given in the last row in Table 2. It is seen that all model combinations reduced test set perplexity, but combining RNNLMs of different clusterings outperformed combining RNNLMs of the same clustering. Relative to the mean PPL of the individual RNNLMs in Table 1, the combined LM of different clusterings reduced PPL by 17.5% to 21.6%, and relative to the mean PPL of the combined LMs with the same word clustering, it reduced PPL by 5.2%. These results suggest the effectiveness of our strategy of combining different class LMs.

	PPL
5 Frequency RNNLMs	115.8
5 Brown RNNLMs	111.0
5 CBOW RNNLMs	108.9
5 CSG RNNLMs	110.9
5 Tag RNNLMs	111.4
5 different clustering RNNLMs	**105.8**

Table 2: Penn Treebank test set PPLs by combined RNNLMs

4.3. WSJ results

The standard 40M word Wall Street Journal (WSJ) training data set was used to train our RNNLMs. The RNNLMs were evaluated on the Eval 92 and Eval 93 test sets of the 20k-word WSJ task, with 333 sentences (6080 words) and 213 sentences (3738 words) in Eval 92 and Eval 93, respectively. The baseline word error rate was generated by Kaldi [25] HMM-DNN recognition system using the standard DARPA trigram language model, giving WERs of 6.57% for Eval 92 set and 8.00% for Eval 93 set. From the word lattices produced by this system, we derived n-best sentence hypotheses with n = 100. The Dev 93 set was used for model selection. In the backward model selection, the RNNLMs were selected from the 25 RNNLMs (5 clustering methods times 5 different initializations).

4.3.1. Perplexity

	Eval 92	Eval 93
Trigram	120.6	121.3
Frequency	104.7	105.2
Brown	96.1	96.9
CBOW	101.2	99.1
CSG	100.2	99.8
Tag	100.6	102.6
Mean PPL of all RNNLMs	100.6	100.7

Table 3: WSJ test sets PPLs by individual LMs

Table 3 shows the average test set PPLs for the individual RNNLMs of each word clustering as well as the trigram LM. The PPL results of combining models over different weight initializations for each class LM as well as combining models from five different class LMs (similar to the Penn Treebank task) are shown in Table 4. As in Section 4.2, we again observe that model combination clearly improved PPL over the individual models on the two WSJ test sets: combining 5 RNNLMs with the same clustering achieved 5.3% to 13.3% relative PPL reductions over the average PPL of the five models on the two test sets; combining five RNN models from different classes further reduced PPL by 3.3% and 4.1% relative to the best combined model of Brown clustering on Eval 92 and Eval 93 sets, respectively. These results indicate that different word class partitions can capture different properties of language, and this useful property should be exploited in class-based RNNLMs.

	Eval 92	Eval 93
5 Frequency RNNLMs	95.3	95.2
5 Brown RNNLMs	87.3	87.3
5 CBOW RNNLMs	91.2	88.6
5 CSG RNNLMs	91.1	88.8
5 Tag RNNLMs	91.8	92.7
5 different clustering RNNLMs	**84.4**	**83.7**

Table 4: WSJ test sets PPLs by combined RNNLMs

4.3.2. Word Error Rate

To rescore the n-best sentence hypotheses by using combined LMs, two ways of integrating the LM scores were considered. One was averaging the word probabilities of different LMs as defined by Eq.(6), referred to as linear combination. Another was averaging the log word probabilities of different LMs, referred to as log-linear combination:

$$logP(w) = \frac{1}{N}\sum_{j=1}^{N} logP_j(w) \tag{7}$$

The n-best sentence hypotheses were rescored by combining the log scores of acoustic and language models:

$$S(h_i) = logA(h_i) + W * logL(h_i) \tag{8}$$

where h_i is the ith hypothesis in the n-best list of a sentence, W is language model scale, $A(h_i)$ is the sentence likelihood score from acoustic model and $L(h_i)$ is the sentence probability from a combined LM. In our experiments, W was fixed as 15. Additionally, each language model obtained from RNNLM combination was linearly or log-linearly interpolated with the trigram LM by following the way the RNNLMs were combined, and the interpolation weight for the trigram LM was set as 0.3.

	Eval 92	Eval 93
Trigram LM	6.58	8.00
Frequency RNNLM	5.78 ± 0.12	7.37 ± 0.15
Brown RNNLM	5.65 ± 0.09	7.17 ± 0.18
CBOW RNNLM	5.81 ± 0.13	7.22 ± 0.19
CSG RNNLM	5.87 ± 0.18	7.16 ± 0.14
Tag RNNLM	5.65 ± 0.13	7.17 ± 0.17
Mean WER of all RNNLMs	5.75	7.22

Table 5: WERs (%) by individual LMs on WSJ test sets

We first show the mean and standard deviation of WER results from single language model based n-best rescoring in Table 5. The RNNLMs reduced WER over the trigram LM by 0.71% to 0.93% on Eval 92 set, and by 0.63% to 0.84% on Eval 93 set.

The WER results of linearly combining RNNLMs are provided in Table 6. For each word clustering, five RNNLMs were combined for comparison. In addition, we used the model selection procedure of Section 3.2 to select 5 RNNLMs from the pool of 25 RNNLMs.

On the Eval 92 test set, the proposed model selection reduced WER by 0.45% (absolute) and 7.83% (relative) over the mean WER of individual RNNLMs in Table 5. Combining the selected RNNLMs reduced WER by 0.18% (absolute) and 3.3% (relative) over the average WER of the combined RNNLMs of the same clustering (5.48%). It is clear that combining models from the same class LM improved WER, but combining models of different word clusterings in general produced larger error reductions (except for the combined Brown RNNLM). When the models were combined with the trigram model, word errors were further reduced. On the Eval 93 test set, the combined RNNLMs of different word clusterings gave the lowest WERs without or with the trigram LM.

Model combination	Eval 92		Eval 93	
	RNN	+3-gram	RNN	+3-gram
5 Frequency RNNLMs	5.51	5.37	7.13	7.28
5 Brown RNNLMs	**5.25**	**5.09**	6.96	6.67
5 CBOW RNNLMs	5.49	5.19	6.96	6.70
5 CSG RNNLMs	5.64	5.37	6.90	6.55
5 Tag RNNLMs	5.49	5.30	6.99	6.50
5 Selected LMs	5.30	5.19	**6.73**	**6.50**

Table 6: WERs (%) by linearly combined LMs on WSJ test sets

The results for log-linear combination of the RNNLMs are shown in Table 7. The WER patterns are similar to that of linear model combination, and our model selection method achieved best results in three out of four cases.

Model combination	Eval 92		Eval 93	
	RNN	+3-gram	RNN	+3-gram
5 Frequency RNNLMs	5.55	5.42	7.37	7.05
5 Brown RNNLMs	5.37	5.17	7.02	6.58
5 CBOW RNNLMs	5.44	5.21	6.87	6.70
5 CSG RNNLMs	5.67	5.33	6.90	**6.53**
5 Tag RNNLMs	5.51	5.28	6.96	6.73
5 Selected LMs	**5.02**	**4.86**	**6.76**	6.61

Table 7: WERs (%) by log-linearly combined LMs on WSJ test sets

In Table 8, we further give WERs averaged over Eval 92 and Eval 93 test sets, with the averaging weights proportional to the word counts in the two sets. It is seen that on the combined two test sets, linear and log-linear combinations of the selected RNNLMs both gave best results, with log-linear combination performed somewhat better.

Model combination	Linear		Log-linear	
	RNN	+3-gram	RNN	+3-gram
5 Frequency RNNLMs	6.13	6.09	6.23	6.04
5 Brown RNNLMs	5.90	**5.69**	5.99	5.68
5 CBOW RNNLMs	6.05	5.76	5.98	5.77
5 CSG RNNLMs	6.12	5.82	6.14	5.79
5 Tag RNNLMs	6.06	5.76	6.06	5.83
5 Selected LMs	**5.85**	**5.69**	**5.68**	**5.52**

Table 8: Comparison of linearly and log-linearly combined LMs on WERs (%) averaged over WSJ Eval 92 and Eval 93 test sets

4.4. Cluster and Model Diversity Analysis

To assess the differences between the investigated word classes, we used Jaccard index [27] to measure the similarity between each pair of word clusterings. The larger the Jaccard index, the more similar the clusterings, and the index becomes 1 if two clusterings are the same. Table 9 shows the Jaccard index results. It is seen that the largest index value was 0.141 between CBOW and CSG, which was reasonable as both methods used k-means clustering on word embedding features; the smallest value of 0.007 occurred between the Tag and Frequency binning clusterings, which was also sensible as words were clustered by their grammatical roles in the former and by their unigram occurrences in the latter. The index values between different clusterings were all small, which indicates the diversity of the different clustering strategies. This further validates that different word clusterings could capture different properties of words, which makes our proposed combining strategy effective.

	$Freq$	$Brown$	Tag	$CBOW$	CSG
$Freq$	1	0.016	0.007	0.008	0.008
$Brown$	0.016	1	0.04	0.045	0.026
Tag	0.007	0.04	1	0.018	0.01
$CBOW$	0.008	0.045	0.018	1	0.141
CSG	0.008	0.026	0.01	0.141	1

Table 9: Jaccard index between clusters

5. CONCLUSION AND FUTURE WORK

We have investigated using multiple word clusterings in class-based RNNLMs for ensemble RNN language modeling. We have shown that varying the clustering criteria and the space of word embedding help produced different word/class factorizations which are more effective for training diverse RNNLMs than varying the RNN learning initial conditions. Our proposed greedy method for model selection is able to select relatively diverse base RNNLMs for combination. We have evaluated our proposed method on 10k-word Penn Treebank and 20k-word WSJ task. Encouraging results were obtained on both test set perplexity and word error rate. In a future work, we plan to investigate using more clustering criteria and word embedding features for word/class factorization, developing more sophisticated context-sensitive model combining methods, and conducting experiments on larger tasks.

6. ACKNOWLEDGEMENTS

This work is supported in part by National Science Foundation under the grant award IIS – 1218863.

7. REFERENCES

[1] Y. Bengio, R. Ducharme, , and P. Vincent, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[3] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of Interspeech*, 2010.

[4] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proceedings of ICASSP*, 2011.

[5] C. Chelba and F. Jelinek, "Structured language modeling," *Computer Speech & Language*, vol. 14(4), pp. 283–332, 2000.

[6] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Machine Learning*, vol. 42(1), pp. 177–196, 2001.

[7] A. Deoras, T. Mikolov, S. Kombrink, M. Karafiat, , and S. Khudanpur, "Variational approximation of long-span language models for lvcsr," in *Proceedings of ICASSP*, 2011.

[8] M. Sundermeyer, I. Oparin, J. Gauvain, B. Freiberg, R. Schluter, and H. Ney, "Comparison of feedforward and recurrent neural network language models," in *Proceedings of ICASSP*, 2013.

[9] K. Yao, G. Zweig, M. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *Proceedings of Interspeech*, 2013.

[10] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *Proceedings of Interspeech*, 2013.

[11] J. Hough and D. Schlangen, "Recurrent neural networks for incremental disfluency detection," in *Proceedings of Interspeech*, 2015.

[12] X. Chen, X. Liu, M. J. F. Gales, and P. C. Woodland, "Improving the training and evaluation efficiency of recurrent neural network language models," in *Proceedings of ICASSP*, 2015.

[13] HS Le, I Oparin, A Allauzen, and J Gauvain, "Structured output layer neural network language model," in *Proceedings of ICASSP*, 2011.

[14] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18(4), pp. 467–479, 1992.

[15] Y. Shi, W. Zhang, and M. T Johnson J. Liu, "Rnn language model with word clustering and class-based output layer," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2013(22), 2013.

[16] J. Xue and Y. Zhao, "Random forests of phonetic decision trees for acoustic modeling in conversational speech recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 16(3), pp. 519–528, 2008.

[17] X. Chen and Y. Zhao, "Building acoustic model ensembles by data sampling with enhanced trainings and features," *IEEE Trans. on Speech, Language, and Audio processing*, vol. 21(3), pp. 498–507, 2013.

[18] T. Zhao, Y. Zhao, , and X. Chen, "Building an ensemble of cd-dnn-hmm acoustic model using random forests of phonetic decision trees," in *Proceedings of ISCSLP*, 2014.

[19] Y. Zhao, J. Xue, , and X. Chen, "Ensemble learning approaches in speech recognition," *Speech and Audio Processing for Coding, Enhancement and Recognition, Chapter 5, T. Ogunfunmi, R. Togneri, and M. Narasimha (Eds.) Springer*, pp. 113–152, 2014.

[20] T. Mikolov, "Statistical language models based on neural networks," in *PhD Thesis, Brno University of Technology*, 2012.

[21] P. Liang, "Semi-supervised learning for natural language," in *Master's Thesis, Massachusetts Institute of Technology*, 2005.

[22] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *Proceedings of EMNLP*, 2014.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of Workshop at ICLR*, 2013.

[24] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, , and J.H. Cernocky, "Rnnlm - recurrent neural network language modeling toolkit," in *ASRU Demo Session*, 2011.

[25] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011.

[26] A. Stolcke, "Srilm - an extensible language modeling toolkit," in *Proceedings of ICSLP*, 2002.

[27] P. Jaccard, "The distribution of the flora of the alpine zone," *New Phytol*, vol. 11, pp. 37–50, 1912.