STRUCTURED OUTPUT LAYER NEURAL NETWORK LANGUAGE MODEL

Hai-Son Le[1,2]*, Ilya Oparin*[2]*, Alexandre Allauzen*[1,2]*, Jean-Luc Gauvain*[2]*, François Yvon*[1,2]

[1]Univ. Paris-Sud; [2]LIMSI CNRS, Spoken Language Processing Group
B.P. 133, 91403 Orsay, cedex, France
{lehaison,oparin,allauzen,gauvain,yvon}@limsi.fr

ABSTRACT

This paper introduces a new neural network language model (NNLM) based on word clustering to structure the output vocabulary: Structured Output Layer NNLM. This model is able to handle vocabularies of arbitrary size, hence dispensing with the design of short-lists that are commonly used in NNLMs. Several softmax layers replace the standard output layer in this model. The output structure depends on the word clustering which uses the continuous word representation induced by a NNLM. The GALE Mandarin data was used to carry out the speech-to-text experiments and evaluate the NNLMs. On this data the well tuned baseline system has a character error rate under 10%. Our model achieves consistent improvements over the combination of an n-gram model and classical short-list NNLMs both in terms of perplexity and recognition accuracy.

Index Terms— Neural Network Language Model, Automatic Speech Recognition, Speech-To-Text.

1. INTRODUCTION

Standard n-gram back-off language models (LMs) rely on a discrete space representation of the vocabulary, where each word is associated with a discrete index. On the contrary, Neural network language models (NNLMs) are based on the idea of continuous word representation. Distributionally similar words are represented as neighbors in a continuous space. This turns n-grams distributions into smooth functions of the word representations and helps to make use of hidden word and context similarities. These representations and the associated probability estimates are jointly estimated in a neural network.

Neural networks, working on top of conventional n-gram models, have been introduced in [**?**, 2] as a potential means to improve discrete language models. This topic has recently gained much attention in the domain of speech recognition [6, 3]. Both neural network approach and class-based models were shown to pertain to the few approaches that provide significant recognition improvements over n-gram baselines for large-scale speech recognition tasks [**?**, **?**, **?**].

Probably the major bottleneck with NNLMs is the computation of posterior probabilities in the output layer. This layer must contain one unit for each vocabulary word. Using such a design makes handling of large vocabularies, consisting of hundreds thousand words, infeasible due to a prohibitive growth in computation time. Short-list NNLMs, that estimate probabilities only for several thousands most frequent words, were proposed as a practical workaround this problem [**?**].

In this article, we introduce the *Structured OUtput Layer* (SOUL) neural network language modeling approach. It is based on a tree representation of the output vocabulary. This approach successfully combines the benefits of neural network and class-based techniques in one single framework. As opposed to standard NNLMs, SOUL NNLMs make it feasible to estimate the n-gram probabilities for vocabularies of arbitrary size. As a result, all the vocabulary words, and not just the words in the short-list , can benefit from the improved prediction capabilities of the NNLMs.

The LIMSI Chinese STT system serves as a baseline [7]. It is based on a well-tuned 4-gram LM trained on 3.2 billion words corpora (without any pruning and cut-offs) interpolated with standard short-list NNLMs. The GALE Mandarin data were used to carry out the speech-to-text (STT) experiments and evaluate the performance. Our main contribution is to show that training NNLMs on full vocabularies is computationally feasible, and that it allows achieving significant gains on a large STT task.

This paper is organized as follows. Related work on hierarchical neural networks is summarized in Section 2. Then the architecture of SOUL NNLMs is introduced in Section 3. Section 4 describes the experimental setup and, finally, the experimental results are presented in Section 5.

2. RELATED WORK

For large training sets, sampling partially enables to circumvent one of the NNLM training bottlenecks [**?**]. However, the output vocabulary is in practice always restricted to a short-list up to 20k words. Recent work [8] showed the impact of initialization and proposed new faster training schemes. However, there are some issues that occur when resorting to

a short-list: the NNLMs are only used to predict a limited number of words. Thus the probability distribution must be normalized with a standard back-off LM that is still used to deal with words out of the short-list .

To handle large output vocabularies, a hierarchical structure of the output layer was introduced in [9]. In a nutshell, the output vocabulary is first clustered and represented by a binary tree. Each internal node of the tree holds a word cluster which is divided in two sub-clusters and so on. Leaves correspond to words at the end of this recursive representation of the vocabulary. Thus the neural network aims to estimate probabilities of the paths in this binary tree given the history, rather than directly the word itself.

A shortcoming of this approach is the recursive binary structure. If one word is poorly clustered, this error affects all the internal nodes (or clusters) which lead to this word. This is typically the case for rare words that represent most of the vocabulary. Thus an error in one word may have a significant impact on the whole system. By relaxing the constraint of a binary structure, we expect to overcome this shortcoming.

3. STRUCTURED OUTPUT LAYER NEURAL NETWORK LANGUAGE MODEL

In this section we introduce a new class-based neural network language model, namely Structured OUtput Layer Neural Network Language Model - SOUL NNLM. Following the classical work on distributed word representation [5], we assume that the output vocabulary is structured by a clustering tree, where each word belongs to only one class and its associated sub-classes. If w_i denotes the i^{th} word in a sentence, the sequence $c_{1:D}(w_i) = c_1, \ldots, c_D$ encodes the path for the word w_i in the clustering tree, with D being the depth of the tree, $c_d(w_i)$ a class or sub-class assigned to w_i, and $c_D(w_i)$ being the leaf associated with w_i (the word itself). Then the n-gram probability of w_i given its history h can be estimated as follows using the chain rule:

$$P(w_i|h) = P(c_1(w_i)|h) \prod_{d=2}^{D} P(c_d(w_i)|h, c_{1:d-1}) \qquad (1)$$

The Figure 1 represents the architecture of the NNLM to estimate this distribution, for a tree of depth D=3. The SOUL architecture is the same as for the standard model up to the output layer. The main difference is the output structure which involves several layers with a softmax activation function. The first softmax layer (*class layer*) estimate the class probability $P(c_1(w_i)|h)$, while other output *sub-class layers* estimate the sub-class probabilities $P(c_d(w_i)|h, c_{1:d-1})$. Finally, the *word layers* estimate the word probabilities $P(c_D(w_i)|h, c_{1:D-1})$. Words in the short-list are a special case since each of them represents its own class without subclasses (D=1 in this case). It is worth noticing that while D at most equals to 3 in our experiments, it depends on the clustering algorithm. The proposed word representation can be generalized to an arbitrary tree structure.

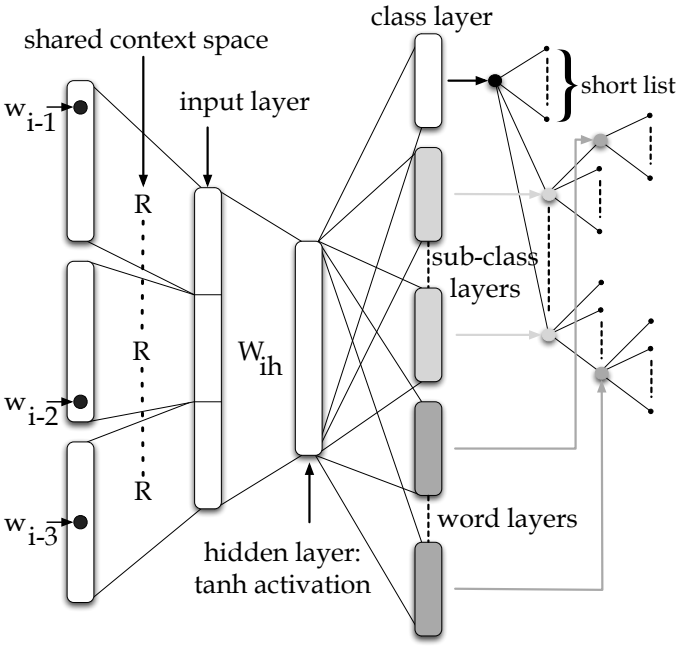

Fig. 1. The architecture of the Structured Output Layer Neural Network language model

3.1. Word clustering

The first step to train a SOUL model is to cluster the words of the output vocabulary. Whereas in [9] a rather sophisticated method based on a randomly initialized model was introduced, we propose a more straightforward method based on the relationship between the two word spaces defined in the standard NNLM [8]: context and prediction spaces. We summarize this clustering algorithm as follows:

Step 1: Train a standard NNLM model with the short-list as an output, following the one vector initialization scheme [8]. In all our experiments, we train this model with 3 epochs and the short-list of 8K words.

Step 2: Reduce the dimension of the context space using a principal component analysis (final dimension is 10 in our experiments).

Step 3: Perform the recursive K-means word clustering based on the distributed representation induced by the context space (except for words in the short-list).

The recursive clustering divides a word class (or sub-class) only if the number of words in this class is above an empirical threshold. In our experiments, the algorithm starts with 4k classes in addition to the short list of 12k. Then a class which contains W words is divided in $|\sqrt{W} + 1|$ sub-classes if $W > 1,000$.

3.2. Training

Training of a NNLM is performed by maximizing the log-likelihood of the parameters. This optimization is performed by stochastic back-propagation as in [2]. The previously trained standard NNLM is used to initialize the shared parameters, while the other parameters are initialized in a usual way. Overall, the training time of each epoch for a SOUL model is only 1.5 times longer than for 8K short-list NNLMs and equal to that of 12K short-list NNLMs.

4. EXPERIMENTAL SETUP

To segment Chinese phrases in words, we make use of the simple longest-match segmentation algorithm based on 56052 word vocabulary used in previous LIMSI Mandarin Chinese STT systems [10]. However, character error rate (CER) is conventionally used to evaluate final recognition performance for Mandarin.

The GALE *dev09* and *eval09* sets are used in this study to evaluate the performance of different models. This data consists of broadcast news and broadcast conversations. A subset of *dev09* called *dev09s* was also defined. It constitutes about a half of *dev09* data. More details concerning the experimental setup, acoustic models and decoding process of the baseline LIMSI Mandarin STT system can be found in [7].

The language model of the LIMSI Mandarin STT system is trained on 3.2 billion word tokens (after segmentation) of Mandarin Chinese data thus providing the system with robust LM estimates. The baseline LM is a word-based 4-gram LM. Individual LMs are first built for each of the 48 Mandarin corpora available by the end of the year 2009. These models are smoothed according to the unmodified interpolated Kneser-Ney discount scheme. No cut-offs and pruning is imposed thus making the LMs to take account of all available information. These individual models are subsequently linearly interpolated together with interpolation weights tuned on *dev09* data. As the number of individual models is small this does not result in a bias to this data.

Each NNLM is trained on about 25M words after resampling of the training data. For each test configuration 4 NNLMs of the same type are trained and interpolated. Each of these 4 NNLMs differs in the dimension of the shared context space, size of the hidden layer and training data resampling. For all our experiments, the learning rate of different NNLMs is 5×10^{-3} , the learning weight decay is 5×10^{-8} and the the weight decay is 3×10^{-5}.

5. EXPERIMENTAL RESULTS

State-of-the-art n-gram language models are rarely of an order larger than 4. Our previous experiments on very large setups indicated that the gain obtained when increasing the n-gram order from 4 to 5 is almost negligible while the size of models increases drastically. Handling such models is thus very impractical and can hardly be done without pruning. However, this is not the case for NNLMs due to the different nature of these models. The increase in context length at the input layer results in only at most linear growth in complexity [**?**]. Thus training longer-context neural network models is still feasible.

As our aim is to improve the performance of the Mandarin STT system, we also investigated the increase in context length from 3 (that corresponds to 4-grams) to 5 for different NNLMs, while keeping the same 4-gram back-off LM at the output layer for standard short-list NNLMs. This remains a valid thing to do with the back-off scheme used for NNLMs with short-lists .

Perplexity and character error rate (CER) results for different models are presented in Table 1. Perplexity results are reported for *dev09* and CER results are on *dev09s* and *eval09*.

The row *+4-gram NNLM 8K* corresponds to the results when the baseline 4-gram model is interpolated with the baseline NNLMs that make use a short-list of 8K most frequent words and take account of the context of the same length (i.e. three). The row *+6-gram NNLM 8K* reports on the interpolation of the baseline 4-gram LM with the longer-context NNLMs.

We tried the longer context neural network setup with the short-list increased up to 12K most frequent words. The results obtained with this setup are reported in the rows *+4-gram NNLM 12K* and *+6-gram NNLM 12K* in Table 1. Finally, results obtained with the whole-vocabulary SOUL NNLMs are represented in rows 4 and 7.

model	**ppx**	**CER**	
	dev09	*dev09s*	*eval09*
Baseline 4-gram	211	9.8%	8.9%
+4-gram NNLM 8K	187	9.5%	8.6%
+4-gram NNLM 12K	185	9.4%	8.6%
+4-gram SOUL NNLM	180	9.3%	8.5%
+6-gram NNLM 8K	177	9.4%	8.5%
+6-gram NNLM 12K	172	9.3%	8.5%
+6-gram SOUL NNLM	**162**	**9.1%**	**8.3%**

Table 1. *Results for different NNLMs for Mandarin Chinese dev09s setup.*

The results presented in this study suggest several conclusions. Contrary to classical back-off n-gram LMs, increasing the NNLM context length significantly improves the results both in terms of perplexity and CER, without any major impact on the training and probability computation time. This is true for all our NNLMs, which all improve the Kneser-Ney baseline LM trained on large amounts of data. This shows the capability of NNLMs to overcome data sparsity issues.

The larger part of the gain attained by SOUL NNLMs,

corresponding to a relative improvement of 23% in perplexity and 7-9% in CER, is due to the fact these models enable to predict all the vocabulary words. This gain is not observed when the short-list is extended from 8K to 12K. The most significant improvement with SOUL models is obtained for the longer-context (6-gram) NNLMs configuration.

6. CONCLUSION AND FUTURE WORK

The SOUL Neural Network approach to language modeling was presented in this paper. It combines two techniques that were proved to improve the STT system performance for large-scale tasks, namely neural network and class-based language models. This approach allows training of neural network LMs with full vocabularies without confining their power to predicting words from limited short-lists. Significant improvement in speech recognition was attained over a challenging baseline provided by a 4-gram LM trained on over 3 billion words (without any pruning and cutoffs) and interpolated with standard short-list based NNLMs.

We also investigated the impact of short-list size and context length on the performance of short-list NNLMs. Longer-context NNLMs were shown to improve the performance without drastic increase in computational costs and model size. At the same time increase of the short-list did not result in consistent improvements in speech recognition. That allows stating that short-list NNLMs were at the current top of their performance when compared to SOUL NNLMs. SOUL NNLMs outperformed standard short-list NNLMs for all test configurations.

The SOUL NNLMs are expected to be even more beneficial for languages with especially large vocabularies. Inflectional languages as, for example, Russian or Arabic, though being completely different in grammar and morphology, are characterized by large number of wordforms for a given lemma. This results in vocabularies that are several times larger than the ones used for Chinese or English. In this context, the short-list strategy is thus expected to be less effective, especially when small short-lists are used. The SOUL neural network model provide us with a more coherent solution in this case. As future work, we thus plan to perform language modeling experiments with SOUL NNLMs on a large-scale Arabic setup with several hundreds thousand words vocabulary.

Acknowledgments

This work has been partially supported by the Quaero program and by the GALE program. Any opinions, findings or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

7. REFERENCES

[1] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: the penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[2] Y. Bengio, R. Ducharme, P. Vincent, and Ch. Janvin, "A neural probabilistic language model," *JMLR*, vol. 3, pp. 1137–1155, 2003.

[3] T. Mikolov, M. Karafiat, L. Burget, J. Černocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. of Interspeech'10*, 2010, pp. 1045–1048.

[4] Holger Schwenk, "Continuous space language models," *Comput. Speech Lang.*, vol. 21, no. 3, pp. 492–518, 2007.

[5] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai, "Class-based n-gram models of natural language," *Comput. Linguist.*, vol. 18, no. 4, pp. 467–479, 1992.

[6] Hong-Kwang Kuo, Lidia Mangu, Ahmad Emami, and Imed Zitouni, "Morphological and syntactic features for arabic speech recognition," in *Proc. ICASSP 2010*, 2010.

[7] I. Oparin, L. Lamel, and J-L. Gauvain, "Improving mandarin chinese STT system with random forests language models," in *Accepted to ISCSLP'10*, 2010.

[8] H.S. Le, A. Allauzen, G. Wisniewski, and F. Yvon, "Training continuous space language models: Some practical issues," in *Proc. of EMNLP'10*, Cambridge, MA, 2010, pp. 778–788.

[9] Andriy Mnih and Geoffrey E Hinton, "A scalable hierarchical distributed language model," in *Advances in Neural Information Processing Systems 21*, 2008, vol. 21, pp. 1081–1088.

[10] J. Luo, L. Lamel, and Gauvain J.-L., "Modeling characters versus words for Mandarin speech recognition," in *Proc. of ICASSP'09*, 2009, pp. 4325–4328.