CONNECTIONIST LANGUAGE MODELING FOR
LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION

Holger Schwenk and Jean-Luc Gauvain

LIMSI-CNRS
91403 Orsay cedex, bat. 508, B.P. 133, FRANCE
{schwenk,gauvain}@limsi.fr

ABSTRACT

This paper describes ongoing work on a new approach for language modeling for large vocabulary continuous speech recognition. Almost all state-of-the-art systems use statistical n-gram language models estimated on text corpora. One principle problem with such language models is the fact that many of the n-grams are never observed even in very large training corpora, and therefore it is common to back-off to a lower-order model. In this paper we propose to address this problem by carrying out the estimation task in a *continuous* space, enabling a *smooth* interpolation of the probabilities. A neural network is used to learn the projection of the words onto a continuous space and to estimate the n-gram probabilities. The connectionist language model is being evaluated on the DARPA HUB5 conversational telephone speech recognition task and preliminary results show consistent improvements in both perplexity and word error rate.

1. INTRODUCTION

Language modeling is known to be a very important aspect of speech recognition. Almost all state-of-the-art large vocabulary continuous speech recognition (LVCSR) systems use statistical language models based on n-grams, i.e. the model predicts the following word based on the previous $n-1$ words, ignoring all other context. Due to data sparseness and computational complexity during decoding, n is usually limited to two or three words. Although these statistical language models (LM) perform quite well in practice, there are several drawbacks from a theoretical point of view due to the high dimensionality in the discrete space representation of the words. The vocabulary size in most current LVCSR systems is at least 64k words, which means that many of the $(64k)^2$ bigrams and $(64k)^3$ trigrams are never observed during training. Inevitably a number of the word sequences in the test data are likely to be different from the word sequences seen during training. This is particularly true in LVCSR where the decoder is likely to request probabilities for n-grams that are syntactically or semantically incorrect, i.e., sequences that would never be observed in any training corpus of any size. We will provide some statistics that seem to support this observation.

"True generalization" is difficult to obtain in a discrete word indice space, since there is no obvious relation between the word indices. The probability distributions are not smooth functions since any change of the word indices can result in an arbitrary change of the LM probability. Various techniques for generalization to new word sequences have been proposed, in particular

backing-off and smoothing. These approaches rely on the utilization of probabilities available for shorter contexts. Another approach is to use word classes in order to improve generalization, but these do not seem to scale well to very large training corpora.

Recently, a new approach has been developed that proposes to carry out the estimation task in a *continuous space* [1]. The basic idea is to project the word indices onto a continuous space and to use a probability estimator operating on this space. Since the resulting probability functions are smooth functions of the word representation, better generalization to unknown n-grams can be expected. In this paper a neural network is used as probability estimator since it can learn both the projection and the estimates of the n-gram probabilities.

The connectionist LM has been previously evaluated on two text corpora ("Brown": 800K training words, English textbooks; and "Hansard": 32M words, Canadian Parliament proceedings) and achieved perplexity improvements of up to 30% with respect to a standard 3-gram [1]. In this paper we extend the approach to large vocabulary continuous speech recognition for the DARPA HUB5 task. Several improvements to increase efficiency during decoding are discussed.
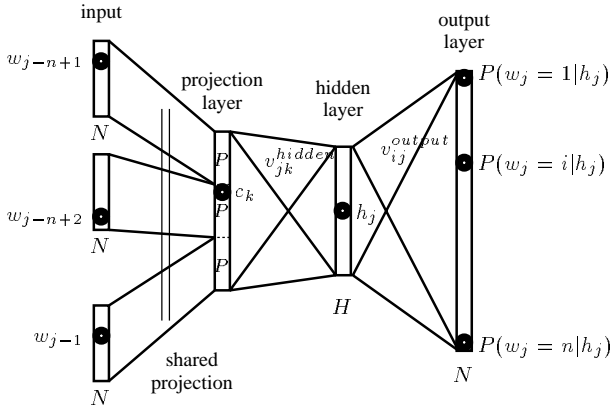
2. ARCHITECTURE OF THE APPROACH

The architecture of the connectionist LM is shown in Figure 1. A standard fully-connected multi-layer perceptron is used. The inputs to the neural network are the indices of the $n-1$ previous words in the vocabulary $w_{j-n+1}, ..., w_{j-2}, w_{j-1}$ and the outputs are the posterior probabilities of *all* words of the vocabulary:

$$P(w_j = i | w_{j-n+1}, ..., w_{j-2}, w_{j-1}) \qquad \forall i \in [1, N]$$

This can be contrasted to standard language modeling where only one probability is calculated. The input uses the so-called 1-of-n coding, i.e., the *i-th* word of the vocabulary is coded by setting the *i-th* element of the vector to 1 and all the other elements to 0. This coding substantially simplifies the calculation of the projection layer since we only need to copy the *i-th* line of the $N \times P$ dimensional projection matrix, where N is the size of the vocabulary and P the size of the projection. The hidden layer activities h_j are calculated by applying the *tanh* function to the weighted sum of the projection layer activities c_k:

$$h_j \;=\; \tanh\left(\sum_k v_{jk}^{hidden} c_k + b_j^{hidden} \right) \qquad \forall j = 1...H$$

where b_j^{hidden} is the bias of the *j-th* hidden layer neuron. The outputs are calculated in a similar way, using a softmax normalization

Fig. 1. Architecture of the connectionist language model. h_j denotes the context $w_{j-n+1}, ..., w_{j-1}$.

to obtain posterior probabilities:

$$o_i = \sum_j v_{ij}^{output} h_j + b_i^{output}$$

$$p_i = \frac{e^{o_i}}{\sum_{k=1}^{N} e^{o_k}} \qquad \forall i = 1...N$$

The value of the *i-th* output neuron corresponds directly to the probability $P(w_j=i|w_{j-n+1}, ..., w_{j-2}, w_{j-1}) = P(w_j=i|h_j)$.

Training is performed with the standard back-propagation algorithm using cross-entropy as the error function (see for instance [2] for a general description of neural network training):

$$E = \sum_{i=1}^{N} d_i \, log \, p_i$$

where d_i denotes the desired output, i.e., the probability should be 1.0 for the next word in the training sentence and 0.0 for all the other ones. It can be shown that the outputs of a neural network trained in this manner converge to the posterior probabilities. Therefore, the neural network minimizes directly the perplexity on the training data. Note also that the gradient is back-propagated through the projection-layer, which means that the neural network learns by itself the projection of the words onto the continuous space that is best for the probability estimation task.

2.1. Complexity analysis

Let us analyse the complexity of calculating the probability of one n-gram $P(w_j|w_{j-n+1} \cdots w_{j-1})$. The activities of the projection layer are obtained by a simple table look-up and can be neglected in the complexity analysis. The calculation of the hidden- and output-layer activities correspond to a matrix/vector multiplication followed by the application of a non-linear function. This gives the following number of floating point operations (Flops):

$$((n-1)P \times H) + H + (H \times N) + N$$

where H the size of the hidden layer. Since N is much larger than H, the complexity is dominated by the calculation at the output layer. For usual values of n=3, N=64k, P=50 and H=200, about 13 MFlops are needed to calculate one LM probability, which is

prohibitive for use in LVCSR. Note that due to the softmax normalization, all of the output activities need to be calculated even if only one probability is needed.

However, a LVCSR decoder usually requests many different LM probabilities for the same context when expanding the search trees and during LM look-ahead. Using caching techniques, the proposed LM can calculate these additional predictions for the same input context at no cost since they are already available at the output! As a result, if the LM scores for all 64k possible words are needed then the complexity is only 200 Flops for each probability. In practice a decoder probably won't request n-gram probabilities for all 64k possible words (see Section 3 for statistics on HUB5 decoding), and it is not very reasonable to spend a lot of computation power on words that appear very rarely. Therefore, we chose to use the neural network only on "interesting words", which are refered to as a *shortlist* in the following discussion. This of course requires defining what words are interesting. Two cases are considered:

1. **Static shortlists :**
   the neural network is used to predict the posterior probabilities of the $l \ll N$ most frequent words, *independently* of the input context.

2. **Dynamic shortlists :**
   the neural network is used to predict the posterior probabilities of $l \ll N$ words, where this set of words *depends* on the current context.

Let us denote $h_j = w_{j-2}, w_{j-1}$ the word history of a 3-gram. The LM probabilities of words in the shortlist are calculated by the network ($\hat{P}_N$) and the LM probabilities of the remaining words by a standard 3-gram backoff LM (P_B):

$$P(w_j|h_j) = \begin{cases} \hat{P}_N(w_j|h_j) \cdot P_S(h_j) & \text{if } w_j \in \text{shortlist} \\ P_B(w_j|h_j) & \text{else} \end{cases}$$

$$\text{with} \quad P_S(h_j) = \sum_{w \in shortlist(h_j)} P_B(w|h_j)$$

In other words, one can say that the neural network redistributes the probability mass of all the words in the shortlist.[1] These probability masses can be precalculated and easily stored in the data structures of the standard 3-gram LM. A standard back-off technique is used if the probability mass for a requested input context is not directly available.

Since the set of words in the dynamic shortlist needs to be determined at each calculation of LM probability, an efficient algorithm is needed. The following procedure, which takes advantage of information already available by the standard 3-gram LM, was used. The word w_3 is part of the dynamic shortlist for the context $w_1 w_2$ if a 3-gram $w_1 w_2 w_3$ or a 2-gram $w_2 w_3$ has been encountered in the training data. In addition a (small) number of very frequent words are included in the dynamic shortlist. Static and dynamic shortlists of different sizes are compared in Section 3.

Finally, all the required computations involve matrix operations that can be highly optimized on standard computing architectures. Optimized BLAS libraries [3, 4] were used to take advantage of machine characteristics such as the cache size, memory architecture and instructions set (e.g. SSE on Intel processors). Using these libraries up to 800 MFlops per second can be

---

[1]Note that the sum of the probabilities of the words in the shortlist for a given context is normalized $\sum_{w \in shortlist} \hat{P}_N(w|h_j) = 1$.

achieved on DEC Alpha workstations or Intel CPUs. These optimization techniques make it possible to use the connectionist LM for LVCSR.

3. RESULTS ON HUB5

In this section we present results for the DARPA HUB5 conversational telephone speech recognition task [5]. This task was chosen since it is known to be very difficult, in particular with respect to language modeling for spontaneous speech.

The speech recognizer used in these experiments was derived from the LIMSI broadcast news transcription system[6]. Context-dependent phones are modeled using tied-state left-to-right CD-HMMs with Gaussian mixture, where the state tying is obtained by means of a phonemic decision tree. The BN 3-gram language model used in these experiments was trained on BN transcriptions and on newspaper and newswire texts. Word recognition is done using a single pass dynamic network decoder [7].

In order to adapt the LIMSI BN system to the HUB5 task, acoustic models were trained on about 280h of conversational speech data distributed by the LDC:

- Switchboard1 corpus (swb1): 248h of speech, 2.9M words.

- Call Home English (che): 17h of speech, 218k words.

In addition to the BN language model, 3-gram LMs were also trained on the manual transcriptions of the HUB5 acoustic training data. Table 1 summarizes the perplexities obtained with standard backoff language models on the 1998 evaluation data set (eval98, 35k words). The first two columns give the perplexities using LMs trained on the individual HUB5 corpora. Interpolating these two LMs (swb1+che) achieves a perplexity of 138.8, which is lower than building an LM on the combined transcripts from the 2 corpora (swb1che, perplexity=143.8). We did not try to optimize the vocabulary, since the OOV rate with LIMSI's standard 64k broadcast news lexicon is only 0.5%. The wordlist contains 263 compounds like I_AM or A_LOT_OF. Therefore we report also the decompounded complexity, i.e counting the actual number of words in the test sentence.

train. corpus	che	swb1	swb1che	swb1+che
eval98 perplexity	215.3	152.1	143.8	138.8
decompounded	131.5	95.9	91.2	88.3

Table 1. Perplexities of standard 3-gram backoff LMs. swb1che: trained on the combined corpora, swb1+che: interpolated LM from swb1 and che.

3.1. Importance of the shortlist

The type and the length of the shortlist directly influence the complexity of the calculations and the expected reduction in perplexity since fewer cases are handled by the network as the size of the shortlist decreases. While this results in a lower overall complexity for smaller shortlists (this effect is less important when many probabilities for the same context must be calculated during decoding), there is also less room for improvement. We expected the dynamic shortlist to be more powerful since the context dependency should allow a higher probability mass to be covered by the network. Table 2 summarizes the results obtained for 4 different shortlists (training was done using swb1che). The first line shows the perplexity on the eval98 corpus and the second line gives the

percentage of probabilities that are actually calculated by the neural network, i.e. the word to be predicted is in the shortlist. Note that the network is not used to calculate the bigram probabilities at the begining of each sentence (this accounts to a 10% coverage loss).

shortlist type & length	dynamic			static
	600	1000	2000	2000
eval98 perplexity	141.6	138.5	134.7	134.5
eval98 coverage	77.1%	80.1%	82.3%	83.2%

Table 2. Comparison of different shortlists (see text for details).

It is clear that as the length of the shortlist is increased, more LM probabilities are calculated by the neural network, which leads to a decrease in the perplexity. Surprisingly, for the same length better results were obtained with a static shortlist than with a dynamic one. We believe that this can be explained by the fact that the underlying backoff 3-gram LM was trained on a small corpus, which means that there are very few n-grams that can be used to determine the shortlist for a given context, and these shortlists may not be representative.

On the other hand, the results of the static shortlist are rather encouraging since this version is much easier to train and to optimize. Based on these results, we decided to only use static shortlists of length 2000 for the following experiments. This means the neural network predicts the LM probabilities for the 2000 most frequent words independently of the context.

train. corpus	che	swb1	swb1che	swb1+che
eval98 perplexity	196.2	141.7	134.5	132.4
rel. improvement	-8.9%	-6.8%	-6.5%	-4.9%
decompounded	120.9	89.9	85.8	84.5

Table 3. Perplexities of connectionist 3-gram LMs on eval98. The relative improvements are calculated with respect to the standard backoff 3-gram LMs (see Table 1).

Table 3 gives the perplexities for connectionist LMs trained on the HUB5 corpora. A small but consistent reduction in perplexity with respect to the backoff 3-gram LM is observed in all cases. Although the interpolated connectionist LM (swb1+che) gives the lowest perplexity, we decided to use the swb1che LM since the use of only one network results in faster processing. The network parameters are as follows: $n=2$, $c=30$, $h=50$ for che and $n=2$, $c=50$, $h=200$ for all the other ones. All weights were initialized randomly and standard stochastic gradient descent was performed for 20 iterations over the training material. More sophisticated initialization methods were tried for the projection matrix, but they did not lead to any significant improvements.

Our current experiments focus on an evaluation of the connectionist LM using only 3-grams, but the proposed model has several properties that makes it very promising for much longer contexts. In fact, longer contexts increase only slightly the number of parameters and the complexity of the model since only the projection layer is affected.[2] Note that the proposed LM always uses the full context for the posterior probability estimation, i.e it never backs-off to lower orders. Words that often appear in similar

---

[2]Eventually an increased hidden layer is also needed in order to deal with the more complicated learning problem.

contexts will probably get similar projection codes, independently of their position in the context. This is expected to lead to good generalization behavior due to the smooth probability estimation function.

3.2. Decoding experiments

BN LM +	backoff LM		connectionist LM	
	swb1che	swb1+che	swb1che	swb1+che
perplexity	119.1	118.5	113.8	113.3
decompounded	76.8	76.4	73.7	73.4

Table 4. Eval98 perplexities when interpolating the HUB5 LM with a large Broadcast News 3-gram LM.

The decoding experiments were performed by interpolating the LMs trained on the swb1 and che corpora with LIMSI's standard BN 3-gram LM. The corresponding perplexities on eval98 are given in Table 4. Decoding was performed with a slightly modified version of the LIMSI Broadcast News system. This system (using the standard backoff 3-gram LM) achieves a word error rate of 46.3% without adaptation, and 42.8% after unsupervised MLLR adaptation (1 iteration with 2 regression classes). Note that this system is not yet tuned to the HUB5 task and many changes are in progress that are expected to lead to further word error reductions.

	backoff LM		connectionist LM	
	no adapt	adapt	no adapt	adapt
word error	46.3%	42.8%	45.8%	42.5%

Table 5. Word error rates on eval98 with standard 3-gram and connectionist LMs.

Although the connectionist LM gave only a small reduction in perplexity, a reduction in word error from 46.3 to 45.8% without adaptation is obtained with this LM (see Table 5). Note that a 1% absolute error reduction is not easy to obtain on the HUB5 task eventhough the word error rate is quite high. The experiments were done using a full decode with the connectionist LM.

Further insight can be gained with the help of statistics collected during decoding. Averaging over the 4317 sentences of the 3h eval98 data set, 4.66M LM probabilities were requested by the decoder for each sentence, among which 2.17M were handled by the neural network (46.6 %). This percentage is much lower than the coverage on the eval98 transcriptions (see table 2) and the training data transcriptions (85.9 %). This means that the decoder is requesting many n-grams that are very unlikely to be observed in the training corpus, even of very large size, and a good generalization of the LM to these unseen n-grams appears to be important during decoding. It seems difficult to measure the quality of this generalization behavior by calculating the perplexity on a necessarily rather small development corpus.

One could expect further improvements when using the neural network for more than 2000 words and it appears that the overall complexity will not increase too much. In fact, on average only 12.1k of the 2.17M calls to the network resulted in a complete forward pass, all other probabilities were cached and directly available at the network output. Averaging over all sentences, the mean and maximum of the number of sequentially requested probabilities for the same context is 540 and 17.7k respectively. These values depend of course on the pruning parameters of the decoder.

4. DISCUSSION AND FUTURE WORK

In this paper we have described ongoing work on a new approach to language modeling for LVCSR. The word indices are projected onto a continuous space, allowing by these means smooth interpolations. The current experimental results are insufficient to make strong conclusions, but they illustrate the potential of the approach. Although the connectionist language model was only used for 47% of the requested LM probability calculations during decoding, a word error reduction of 0.5% was obtained on the HUB5 task.

Several extensions are currently under investigation that are expected to lead to further improvements. First, longer static short-lists will be considered. From what has been observed thus far, increasing the length of the shortlist always reduces the perplexity. Due to the efficient caching algorithm, this should not result in a significant increase in decoding time. Another not yet well exploited potential of the proposed algorithm is the possibility to learn long-span LMs with $n \gg 3$.

The move to a continuous representation of the words should open some promising research directions. In this work neural networks are used as probability estimators as well as to learn the projection. In fact, once this projection is learned any probability estimator can be used, in particular Gaussian mixtures. Although Gaussian mixtures are not usually trained in a discriminative way (like neural networks), they can be evaluated much faster.

Finally the continuous projection matrix opens the way to several powerful LM adaptation techniques. It is for instance possible to apply a transformation to this matrix in order to accommodate for new LM data. Similar ideas have also been proposed for standard n-gram LM (see for instance [8] for an review of LM adaptation techniques), but the discrete representation makes it mathematically less tractable.

6. REFERENCES

[1] Yoshua Bengio and Réjean Ducharme, "A neural probabilistic language model," in *NIPS*. 2001, vol. 13, Morgan Kaufmann.

[2] Christopher M. Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, Oxford, 1995.

[3] "Compaq Extended Math Library (CXML)," http://www.compaq.com/hpc/software/dxml.html.

[4] "Automatically Tuned Linear Algebra Software (ATLAS)," http://www.netlib.org/atlas.

[5] A. Martin and M. Przybocki, "The 2001 NIST evaluation for recognition of conversational speech over the telephone," in *2001 NIST LVCSR workshop*, 2001.

[6] Jean-Luc Gauvain, Lori Lamel, and Gilles Adda, "The LIMSI broadcast news transcription system," in *Speech Communication*. 2002, vol. to appear, North Holland.

[7] J.-L. Gauvain and L. Lamel, "Fast decoding for indexation of broadcast data," in *Proc. ICSLP*, 2000.

[8] R. DeMori and M. Frederico, "Language model adaptation," in *Computational Models od Speech Pattern Processing*, K. Ponting, Ed. Springer, 1999.