SUBWORD LANGUAGE MODELING WITH NEURAL NETWORKS

Tomáš Mikolov[1], Ilya Sutskever[2], Anoop Deoras[3], Hai-Son Le[4], Stefan Kombrink [1], Jan Černocký[1]

[1]Brno University of Technology, [2]University of Toronto,
[3]Johns Hopkins University, [4]Université Paris Sud and LIMSI/CNRS,
{imikolov,kombrink,cernocky}@fit.vutbr.cz
ilya@cs.utoronto.ca, adeoras@jhu.edu, lehaison@limsi.fr

ABSTRACT

We explore the performance of several types of language models on the word-level and the character-level language modeling tasks. This includes two recently proposed recurrent neural network architectures, a feedforward neural network model, a maximum entropy model and the usual smoothed n-gram models. We then propose a simple technique for learning sub-word level units from the data, and show that it combines advantages of both character and word-level models. Finally, we show that neural network based language models can be order of magnitude smaller than compressed n-gram models, at the same level of performance when applied to a Broadcast news RT04 speech recognition task. By using sub-word units, the size can be reduced even more.

***Index Terms*—** language modelling, compression, neural network, maximum entropy

1. INTRODUCTION

Most statistical language models use words rather than characters as their atomic units due to their superior performance in terms of accuracy and the number of parameters to be estimated [1]. However, word-level models are unable to deal with new words, commonly called Out-of-Vocabulary words. Recently, there has been a lot of research effort aiming at the OOV problem in the context of word-level models [2], with subword units that are learned from the data [3], and based on more linguistically motivated approaches [4]. A thorough overview of OOV modeling in automatic speech recognition (ASR) is given by Bazzi [5].

The root of the OOV problem lies in the complete inability of word-level language models to sensibly assign nonzero probability to previously unseen words. The problem can be addressed using smoothing techniques: for example, we could directly train a model on the subword-level, or follow a hybrid approach, where the unknown words are delegated to a character-level language model [6].

Despite their intuitive appeal, hybrid approaches can be problematic for languages whose words make poor atomic units, as is the case for Czech, Finnish, and Turkish, among others. The OOV rate of such languages remains unacceptably high even when the vocabularies exceed a million of word forms.

Thus, in this paper we study language models trained on subword units, which are applicable to the vast majority of languages. While the previous work mainly focused on either effective implementations of hybrid ASR systems or on defining and learning the set of the subword units, in our work we study models built on top of the subword units.

First, we compare the performance of several different techniques on character-level modelling tasks. We study the performance of smoothed n-gram models, several types of neural networks (including two recently proposed recurrent architectures), and a maximum entropy model. We find that smoothing techniques work relatively poorly for character-level language modeling, and other character-level techniques perform significantly better.

Next, we perform experiments with subword-level models. We find that performance of the subword-level models is significantly better than that of character-level models, while the OOV rate is still zero (we can assign a sensible probability to any sequence of characters). We find that subword-level models can be competitive with word-based models, and the amount of parameters that need to be estimated is significantly lower for the neural network based language models trained on subwords.

Finally, using these findings, we propose a novel compression technique for neural network language models. Using quantization, we reduce the memory requirements by around 90% which makes the resulting models orders of magnitude smaller than n-gram models, while maintaining the same level of word error rate on a Broadcast news RT04 speech recognition task. We then discuss the possibility of further reduction of size of the neural network language model by decomposing infrequent words into subwords.

2. CHARACTER-LEVEL MODELS

Character-level language models are commonly used for modeling new words in open-vocabulary speech recognition and in keyword spotting systems, and many successful phonotactic language identification techniques use generative language models based on characters or phonemes. Usually, smoothing techniques such as Witten-Bell discounting are used to build such models [7]. Alternatively, we can apply neural network language models [8] (NNLM) to this problem, because NNLMs have been reported to achieve the state-of-the-art performance on several standard word-level language modelling tasks [9, 10, 11]. Hybrid character-word-level NNLMs have been already investigated in [12] and recurrent neural networks were applied to character-level language modeling in [13].

2.1. Penn Treebank Corpus experiments

We performed experiments on a tokenized Penn Treebank Corpus whose words were split to individual characters and the spaces be-

The paper was written while the first author was visiting the University of Montreal. The second author was supported by a Google fellowship. This work was partly supported by Technology Agency of the Czech Republic grant No. TA01011328, Czech Ministry of Education project No. MSM0021630528, and Grant Agency of the Czech Republic project No. 102/08/0707.

Table 1. *Results on Penn Treebank corpus (test set) with character-level models.*

Model	Bits/character
NNLM	1.57
N-discounted n-gram	1.48
BPTT-RNN	1.42
HF-MRNN	1.41
Maximum Entropy n-gram (ME)	1.37

tween words were replaced by a special symbol. Sections 0-20 were used as training data (5017k characters), sections 21-22 as validation data (393k characters) and sections 23-24 as test data (442k characters). The original vocabulary size was 10K words and all words outside vocabulary were rewritten to a special `<unk>` token.

We trained n-gram models using SRILM [14] toolkit using a number of smoothing techniques. We tuned the order and the count cutoffs on the validation set, and found 8-grams with Ristad's natural discounting to be the best n-gram model. We then trained several neural network based language models; with feedforward architecture [8] (NNLM), recurrent architecture trained by stochastic gradient descent [15, 16] (BPTT-RNN) and Hessian-free optimized recurrent network [17] with multiplicative connections [13] (HF-MRNN). Also, we trained a hash-based maximum entropy model with up to 15-gram features [10]. The results are summarized in Table 1.

The usual smoothing techniques work poorly when applied to character-level language models, and even the best n-gram models perform significantly worse than the RNN models. The feedforward neural network architecture was unable to achieve good performance. Recurrent NNLMs work reasonably well, and both training techniques lead to similar results on this data set. The most surprising fact is that a simple maximum entropy model (ME) with n-gram features is the best performing method on this task. Its computational complexity is orders of magnitude lower than of RNN models that use up to 1000 hidden units in the hidden layer (further comparison and relationship between ME and NN models is provided in [10], as well as description of the hash-based ME implementation). On the other hand, the memory complexity of the RNN models is much lower than that of models based on n-grams – we will study this interesting fact later.

As the Penn Treebank does not contain any OOVs (all words outside vocabulary are rewritten as `<unk>`), we can compare character-level to word-level models. We can easily compute the bits-per-character of the word-based models by calculating the entropy of the entire test set and dividing it by the number of characters. This way, we obtained 1.32 bpc for the test set using a 5-gram word-based LM with modified Kneser-Ney smoothing and no count cutoffs. This suggests that word-based models are generally better for modeling text sequences, at least for English. It can be however argued that character-level models reserve some probability for novel words, and thus the comparison is not completely fair because their ability to sensibly assign probability to OOV words will not be utilized.

2.2. Large data set experiments

We have performed additional experiments on 'text8' data set from Matthew Mahoney's website[1]. This data set contains 100M characters, from which we used first 90M as train set, the next 5M as held-out set and the last 5M as test set. We were unable to train the BPTT-RNN models with a sufficiently large hidden layer, as well as feedforward NNLM, in a reasonable amount of time. However, the HF-MRNN model was successfully trained with 1500 hidden units thanks to the data-parallel nature of the HF optimizer and the use of eight GPUs [13]. We trained maximum entropy models with up to 20-gram features. We also used a novel architecture, where a small RNN model is trained jointly with the ME model (we denote this architecture RNNME which is described in [10]). We report the results for all the models in Table 2.

[1] Available at `http://mattmahoney.net/dc/text8.zip`

Table 2. *Results on text8 (test set) with character-level models.*

Model	Bits/character
N-discounted n-gram	1.64
ME	1.55
RNNME	1.55
HF-MRNN	1.54
ME interpolated with HF-MRNN	1.47

With increased amounts of data, we still observe poor performance of standard smoothed n-gram models. The maximum entropy model performs very well, and the HF-MRNN achieves the best result among single models. Moreover, it discovers complementary information to the ME model, as their linear interpolation provides further significant improvement. Still, the individual models seem to perform worse than word-level models – with Kneser-Ney smoothed 5-gram, we obtained 1.43 bpc on the test set, with 1.17% OOV rate. Thus, if we suppose that each OOV word can be encoded using 4x more bits than an average word, the entropy of word-level 5-gram model would be 1.48 bpc.

3. SUBWORD-LEVEL MODELS

It appears that training highly accurate character-level models is difficult and that performance is generally lower than of word-level models. Neural networks with recurrent architecture require large hidden layers (more than 1000 neurons), and feedforward networks do not provide good results at all. An alternative approach is to use subword-level models, that can potentially share the advantages of word-level models and are as general as character-level models. While a simple approach is to start with syllables as subword units, we learned the set of subwords from the data.

Elman showed that most of the entropy is concentrated at the first few characters of each word [18]. It would therefore be good if we could avoid making expensive predictions in settings where there is little to no uncertainty. Thus, to not waste computational resources, we chose to keep W most frequent words, and split all the remaining words into syllable-like units based on very simple rules (we split words at vowels and limit the minimum size of subword to 2 characters). Then, we keep the S most frequent syllables plus words and split all remaining tokens into individual characters. It is easy to convert regular text into this alphabet and back with no information loss. For example,

```
new company dreamworks interactive
new company dre+ am+ wo+ rks:  in+ te+ ra+ cti+ ve:
```

Every word can be decomposed into subword units in multiple ways; for example, a word can be spelled with characters or with syllables. However, as long as we follow the above approach con-

Table 3. *Results on text8 (test set) with subword-level models.*

Model	Bits/fragment	Bits/character
Witten-Bell n-gram	4.71	1.58
ME	4.61	1.55
HF-MRNN	4.44	1.49
RNNME	4.36	1.46

sistently, the model will learn to assign negligible probabilities to "incorrect" spellings.

For the following experiment performed on the 'test8' data set, we used $W = 1000$, $S = 2000$, and 26 unique characters, yielding a vocabulary of size 2026. We can see in Table 3 that performance of almost all models trained on word/subword units is improved, compared to character-based models (Table 2). Among the n-gram models, the Witten-Bell discounted 8-gram performed best in our experiments. We trained ME models also with up to 8-gram features (higher orders did not provide significant improvements). The RNNME model was trained with 160 hidden units and the HF-MRNN had 1500 hidden units and 1500 factors.

4. COMPRESSION OF NEURAL NETWORK LANGUAGE MODELS

In the previous experiments, we have observed that character and subword-level language models based on n-gram statistics are very memory inefficient compared to neural networks models, so we therefore consider ways of compressing language models using them.

Compression of backoff n-gram language models is a well-studied problem, as the size of a language model is usually a very significant part of LVCSR and MT systems [19, 20]. However, we are not aware of any successful attempt to compress language models with neural networks. Most of the previous NNLM research was focused on obtaining the best possible accuracy and processing speed, while the size of models was not studied before. However for practical applications, the size of models can be an important factor.

The main motivation for neural networks for language modeling lies in their continuous representations of words and in their ability to generalize to novel contexts. The NNLM's component that transforms a context to a prediction is reused in all possible contexts (i.e., the recurrent connections), causing substantial space savings over n-gram models which explicitly store every context ever observed.

4.1. Quantization

We found that after NNLMs are trained, weights can be quantized to very few bits with only small degradation of performance, as high precision weights (doubles) were needed only during training but not for testing.

We used state-of-the-art setup for Broadcast News RT04 speech recognition from IBM, based on Attila decoder [21]. The language models are trained on 400M words and vocabulary size is 84K. We have previously described experiments with RNNLMs on this setup in [10] and found that with just 80 neurons in the hidden layer, we can obtain better speech recognition performance than with baseline 4-gram model. Although more neurons lead to better results (we were able to obtain more than 10% reduction of WER in our previous work), in this paper we focus just on comparison of models with similar performance in speech recognition.

Table 4. *Re-scoring experiments on RT04 Broadcast News (evaluation set). Sizes of n-gram models are in the ARPA text format.*

Model	WER [%]	size (MB)	size(MB) compressed
unpruned 4-gram	-	2792	242
4.7M 4-gram	14.1	122	14
54M 4-gram	13.11	1862	162
RNN-80 (text format)	12.98	130	-
RNN-80 (quantized)	13.00	-	13

We performed K-means clustering of the neural network weights, with K=128. Every weight is replaced by index of the nearest cluster, and can be encoded using 7 bits, instead of 64 bits (for doubles). In Table 4, it can be seen that there is no significant degradation in WER when using the quantized model. For comparison, we used three baseline n-gram models: 4-gram without any pruning, but using SRILM default cutoffs (all single occurring trigrams and four-grams are discarded); 4-gram pruned down to 4.7 million n-gram entries that is used in the decoding, and a 4-gram pruned down to 54M n-grams that is used for rescoring in the baseline system. In our experiments, we replaced the 54M n-gram model by RNN model, using lattice rescoring technique described in [22]. Storing the RNN model in plain text takes about 130 MB, while a binary representation of the quantized RNN model takes as little as 13 MB. Thus, we can completely avoid using the huge 54M n-gram model, and use much more compact RNN model instead.

The baseline n-gram models can be of course compressed as well. To save space, we can use standard gzip which reduces the size of the 54M n-gram model to 525 MB. However, many n-gram LM-specific compression techniques were developed, and the best among them can reduce the size of the models to about 3 bytes per n-gram entry [20]. In Table 4, we estimate the size of compressed n-gram models by counting 3 bytes for each n-gram entry. Nonetheless, the 54M n-gram model takes more than 10 times the size of quantized RNN-80 model, which even achieves slightly better performance.

We can push the idea even further: different parts of the neural network very likely require different precision, as for example infrequent words are associated in the model with the same amount of parameters as frequent words. We can perform more aggressive quantization of parameters associated with the infrequent words, without losing much of precision. By quantizing different parts of the network to different amount of bits, we were able to further reduce the size of the model to 10 MB with no significant degradation of performance.

4.2. Subword based neural network models

We can perform further compression by following the previously proposed approach for reducing the size of the vocabulary: by dividing all infrequent words into subword sequences, we can greatly reduce the number of parameters in the NNLM (the extent of the reduction is comparable for the feedforward and recurrent architectures). Letting V be the size of the vocabulary and H be the size of the hidden layer, the number of parameters of the basic RNN model is $(2 \times V + H) \times H$.

Thus, by reducing the vocabulary from 100K words to 10K units and using $H = 100$, we can reduce amount of parameters by almost 90%. However, as was mentioned previously, the smaller the vocabulary is, the larger the hidden layer needs to be, at the same level of accuracy. Clearly, the optimal size of vocabulary is a task-specific

Table 5. *Re-scoring experiments on NIST RT05 Meeting recognition setup with subword-level RNN models.*

Model	WER [%]	size (MB)	size (MB) compressed
Word-based bigram	27.0	93	11
Word-based 4-gram	25.1	464	43
Word-based 4-gram (pruned)	25.5	142	15
Subword RNN-160 (text format)	26.5	6.7	-
Subword RNN-160 (quantized)	26.5	-	0.6
Subword RNN-480 (text format)	25.0	21.3	-
Subword RNN-480 (quantized)	24.9	-	1.7

parameter that needs to be tuned.

For the following experiment, we used the setup for NIST RT05 Meeting recognition further described in [23]. We trained the baseline LM on 30M words (meeting transcriptions, Switchboard and Fisher data) with a vocabulary size of 50K. We report the results after 300-best list rescoring in Table 5. The subword models were trained with $W = 1000$ and $S = 2000$. With 480 neurons in the hidden layer, the performance is already better than of 4-gram model with Kneser-Ney smoothing, and the difference in size of models is substantial.

5. CONCLUSION

We performed experiments with different types of language models for character-level language modeling task. We found that the usual smoothing techniques and the feedforward neural network architecture to not work well for such problem. Recurrent NNLMs performed much better, both the one trained by stochastic gradient descent and the one trained by Hessian-free optimization. The most surprisingly, maximum entropy model with n-gram features seems to do very well on character-level language modelling.

Next, we performed experiments with models trained on words and subword units, with limited size of the vocabulary. We learned the subword units from the data, based on very simple rules and frequency of occurrence of these units in the training data; despite simplicity of this approach, we have observed improved performance. This means that none of the studied models, including RNNs, is powerful enough to learn all discoverable patterns from the character-level input. This provides further motivation for research of training algorithms for recurrent neural network models.

The subword-level models are interesting because of three reasons: they outperform character-level models, they have zero out-of-vocabulary rate, and their size is smaller. In fact, the whole word-based language modeling fails for many inflectional and agglutinative languages. In contrast, subword-based language models are unlikely to face difficulties with such languages because they will focus on the meaningful morphological units that construct the extremely large vocabularies of these languages.

Finally, we have described experiments on Broadcast News RT04 and Meeting recognition RT05 setups. We have shown that a simple quantization of weights is sufficient to reduce size of NNLMs several times. We were able to obtain slightly lower word error rate with tiny RNN models, compared to huge 4-gram backoff models. We believe that this result makes neural networks even more attractive for usage in practical applications, as it is possible to avoid having huge n-gram models in the ASR systems.

6. REFERENCES

[1] M. Mahoney, "Adaptive Weighing of Context Models for Lossless Data Compression," Tech. Rep. CS-2005-16, Florida Tech., 2005.

[2] S. Kombrink, M. Hannemann, L. Burget, and H. Heřmanský, "Recovery of Rare Words in Lecture Speech," in *Proceedings of Text, Speech and Dialogue*, 2010.

[3] C. Parada, M. Dredze, A. Sethy, and A. Rastrow, "Learning Sub-Word Units for Open Vocabulary Speech Recognition," in *Proceedings of ACL*, 2011.

[4] M. A. S. Shaik, A. E. Mousa, R. Schlüter, and H. Ney, "Hybrid Language Models Using Mixed Types of Sub-lexical Units for Open Vocabulary German LVCSR," in *Proceedings of Interspeech*, 2011.

[5] Issam Bazzi, *Modelling Out-Of-Vocabulary Words For Robust Speech Recognition*, Ph.D. thesis, MIT, 2002.

[6] Igor Szöke, *Hybrid word-subword spoken term detection*, Ph.D. thesis, Brno University of Technology, 2010.

[7] Pavel Matějka, *Phonotactic and acoustic language recognition*, Ph.D. thesis, Brno University of Technology, 2009.

[8] Yoshua Bengio, Rjean Ducharme, Pascal Vincent, et al., "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[9] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Černocký, "Empirical evaluation and combination of advanced language modeling techniques," in *Proceedings of Interspeech*, 2011.

[10] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černocký, "Strategies for Training Large Scale Neural Network Language Models," in *Accepted to ASRU*, 2011.

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

[12] Moonyoung Kang, Tim Ng, and Long Nguyen, "Mandarin word-character hybrid-input Neural Network Language Model," in *Proceedings of Interspeech*, 2011.

[13] Ilya Sutskever, James Martens, and Geoffrey Hinton, "Generating Text with Recurrent Neural Networks," in *Proceedings of ICML*, 2011.

[14] Andreas Stolcke, "SRILM - an extensible language modeling toolkit," 2002, pp. 901–904.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Mit Press Computational Models Of Cognition And Perception Series*, pp. 318–362, 1986.

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

[17] James Martens and Ilya Sutskever, "Learning Recurrent Neural Networks with Hessian-Free Optimization ," in *Proceedings of ICML*, 2011.

[18] Jeffrey L. Elman, "Finding structure in time," *COGNITIVE SCIENCE*, vol. 14, no. 2, pp. 179–211, 1990.

[19] T. Watanabe, H. Tsukada, and H. Isozaki, "A succinct n-gram language model," in *Proceedings of ACL-IJCNLP*, 2009.

[20] K. Church, R. Wa, T. Hart, and J. Gao, "Compressing Trigram Language Models With Golomb Coding," in *Proceedings of EMNLP*, 2007.

[21] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila speech recognition toolkit," in *Proc. IEEE Workshop on Spoken Language Technology*, 2010.

[22] Anoop Deoras, Tomas Mikolov, and Kenneth Church, "A Fast Rescoring Strategy to Capture Long-Distance Dependencies," in *Proceedings of EMNLP*, 2011.

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