Application of LSTM Neural Networks in Language Modelling

Daniel Soutner and Luděk Müller

University of West Bohemia, Faculty of Applied Sciences, Department of Cybernetics,
Univerzitní 22, Plzeň, Czech rep.
`www.kky.zcu.cz`
`{dsoutner,muller}@kky.zcu.cz`

Abstract. Artificial neural networks have become state-of-the-art in the task of language modelling on a small corpora. While feed-forward networks are able to take into account only a fixed context length to predict the next word, recurrent neural networks (RNN) can take advantage of all previous words. Due the difficulties in training of RNN, the way could be in using Long Short Term Memory (LSTM) neural network architecture.
In this work, we show an application of LSTM network with extensions on a language modelling task with Czech spontaneous phone calls. Experiments show considerable improvements in perplexity and WER on recognition system over n-gram baseline.

Keywords: language modelling, recurrent neural networks, LSTM neural networks

1 Introduction

Statistical language models (LM) play an important role in the state-of-art large vocabulary continuous speech recognition (LVCSR) systems. Statistically computed n-gram models and class-based LMs are the main models used in LVCSR systems, however, subsequent models are becoming more important supplement to existing techniques.

In recent years feed forward neural networks (FFNN) [12] attracted attention due their ability to overcome biggest disadvantage of n-gram models: even when the n-gram is not observed in training, FFNN estimates probabilities of the word based on the full history [15]. That is in contrast to n-gram, where back-off model estimates unseen n-grams with $(n-1)$-gram.

To avoid handling with the parameter n (number of words in n-gram and in FNN LM) we can use the recurrent neural network (RNN) architecture [2]. The RNN is going further in model generalization: instead of considering only the several previous words (parameter n) the recursive weights are assumed to represent short term memory. More in general we could say that RNN sees text as a signal consisting of words.

Long Short-Term Memory (LSTM) neural network [8] is different type of RNN structure. As was shown, this structure allows to discover both long and short patterns in data and eliminates the problem of vanishing gradient by training RNN. LSTM approved themselves in various applications [8][1] and it seems to be very promising course also for the field of language modelling [3].

In this work we present an application of LSTM language model as an extension to the basic n-gram model and the influence of this modification to the perplexity and word error rate analysed on English and Czech corpora.

2 LSTM neural networks

The vanishing gradient seems to be problematic during the training of RNN as shown in [8]. This led authors to re-design of the network unit, in LSTM called as a cell. Fig. 1 shows that every LSTM cell contains *gates* that determine when the input is significant enough to remember, when it should continue to remember or forget the value, and when it should output the value. So designed cells may be interpreted as a differentiable memory.

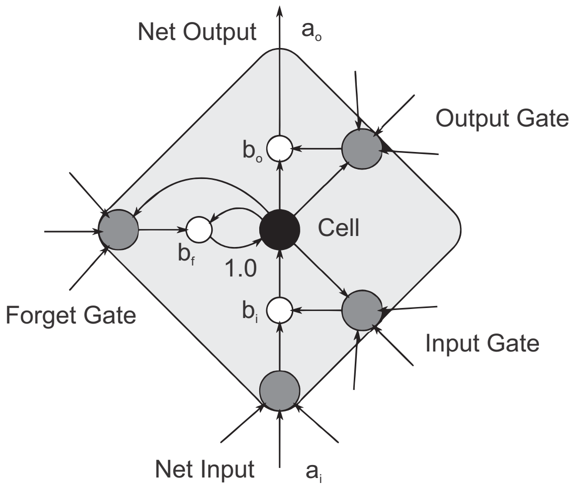

Fig. 1. *LSTM memory cell with gates.*

2.1 LSTM topology

Typical NN unit consists of the input activation which is transformed to output activation with activation function (usually sigmoidal).

The LSTM cell provides this more comprehensively: The three cell inputs called *gates* determine when values are allowed to flow into or out of the block's memory. Firstly, the activation function is applied to all gates. When *input gate* outputs a value close to zero, it zeros out the value from the net input, effectively blocking that value from entering into the next layer. When *forget gate* outputs a value close to zero, the block will effectively forget whatever value it was remembering. The *output gate* determines when the unit should output the value in its memory.

Depends on type of LSTM, consecutions may slightly differ (some modifications and enhancements were introduced), but the main principals are the same. The training algorithm and complete equations of LSTM neural network could be found e.g. in [8] [9].

Due this specific topology of LSTM, especially because of a constant error flow, regular back-propagation could be effective at training an LSTM cell to remember values for very long durations. LSTM can be also trained by evolution strategies or genetic algorithms in reinforcement learning applications [13].

2.2 LSTM language model

The LSTM NN was successfully introduced to the field of language modelling [3]. The topology of our model (shown in Figure 2) is similar to common RNN language models ([2] [3]) and is based on these principals:

- The input vector is word encoded as 1-of-N coding.
- There is a softmax function used in output layer to produce normalized probabilities.
- The cross entropy is used as training criterion.

Normalization of input vector which is generally advised for neural networks is not needed due the 1-of-N input coding.

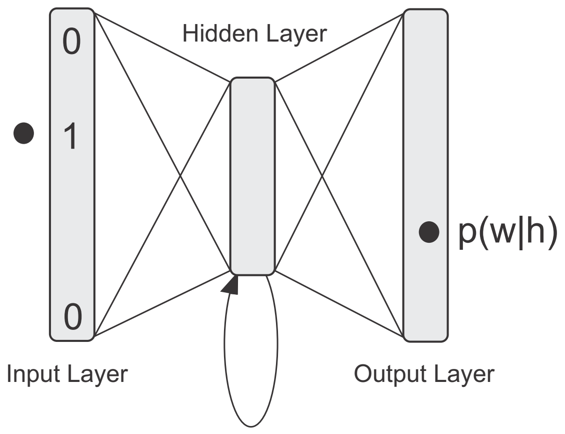

Fig. 2. *Neural network LM architecture.*

3 Input vector modifications

The standard input vector in various neural net language models is mostly 1-of-N. The words on input are encoded by 1-of-N coding, where N is number of words in vocabulary.

We also intended and applied two extensions of the basic model - added Latent Dirichlet Allocation [7] (LDA) for better modelling of longer context [16] and the class extension for dealing with similar words in the same context. Both of these extensions are described in sections bellow.

3.1 Latent Dirichlet Allocation extension

Language models with cache brings improvement in perplexity but not when measured on WER in speech recognition. Thus, to exploit more information from the long span context we decided to use the Latent Dirichlet Allocation (LDA) [7] in our experiments as proposed by T. Mikolov and G. Zweig in [16], where this model is closely described. The LDA process converts word representation of document to low-dimensional vector which represents probability of to topic.

LDA represents documents as mixtures of topics that split out words with certain probabilities. It assumes that documents are produced in the following fashion:

- Deciding on the number of words N the document will have by sampling from a Poisson distribution.
- Choosing a topic mixture for the document (according to a Dirichlet distribution over a fixed set of K topics).
- Generating each word w_i in the document by:
  - Picking a topic (according to the multinomial distribution that you sampled above).
  - Using the topic to generate the word itself (according to the topics multinomial distribution).

Assuming this generative model for a collection of documents, LDA tries to backtrack from the documents to find a set of topics that are most likely to generate the collection.

In our experiments we fixed the length of the word cache for computing topic distribution. For Czech training corpus every phone call is equal to one document, for English Penn Treebank we divided text to documents of 10 non-overlapping sentences; the input vector of NN is modified as original 1-of-N coding and proposed additional LDA feature. The models were created with *gensim* tool [6]. We explored several configurations of trained models with a different number of topics (from 20 to 70) and cache length (50 and 100).

3.2 Class extension

In both - written and spoken language - we use different words for expressing a similar topic or the same fact. There are many approaches in language modelling that are trying to deal with this aspect of language i.e. the class-based models [10]. Assuming this, we tried to investigate whether word classes are able to help us in LSTM LM.

The similar words could be split up to the classes with a lot of different ways, we decided to use one based on inducing word classes from n-gram statistics. Word classes induced from distributional statistics are produced so as to minimize perplexity of a class-based n-gram model given the provided word n-gram counts. This means that words occurring in the similar context should be found in the same class. The classes were prepared using the SRI toolkit [4].

We have modified input vector for our purposes analogously as in LDA extension: we added to the standard input vector (1-of-N) a vector 1-of-C, where C is number of classes. We trained models with various number of classes, from 100 to 2300.

4 Experimental results

4.1 Perplexity results

To maintain comparability with the other experiments, we chose well-known Penn Treebank (PTB) [11] portion of the Wall Street Journal corpus for testing our models. Following preprocessing was applied to the corpora: the vocabulary was short-listed to 10k most frequent words, all numbers were unified into $\langle N \rangle$ tag and punctuation was removed. The corpus was divided into 3 parts (training, development and test) with 42k, 3.3k and 3.7k tokens.

The second part of experiments was performed with Czech spontaneous phone calls (BH). This corpus is further described in Section 4.2.

First, we trained LDA model, word classes and both extensions together on training data and with these models we trained our LSTM neural networks on the same data. All models were trained with 20 cells in hidden layer. Afterwards, we chose models with the best parameters measured on development data. The final results for PTB achieved on test data are shown in Table 1, models were combined with baseline model by linear combination. The proposed extensions seem to be promising, they improve the perplexity of the baseline model by $\approx 2 - 12\%$.

The influence of word cache and number of topics (as parameters of LDA extension) was tested on BH, the Table 2 shows the result, which suggests a cache of length 100 and 50 topics. The influence of the number of classes is shown in Fig. 3, for 500 classes we obtained the best perplexity values.

model	PPL
KN5	140
LSTM +KN5	120
LSTM LDA +KN5	117
LSTM CLASS +KN5	113
LSTM LDA & CLASS +KN5	105

Table 1. *Perplexity results with PTB.*

	cache	
#topics	50	100
20	154.1	163.9
30	164.4	117.2
40	128.5	109.5
50	110.4	**103.7**
70	110.2	107.6
100	**109.6**	125.6

Table 2. *Perplexity results of LSTM with LDA extension on phone calls (BH), different number of topics (20-100) and word cache (50 and 100).*

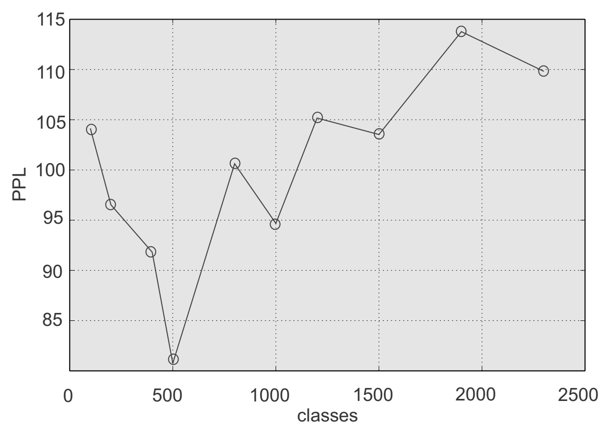

Fig. 3. *Perplexity results of LSTM with class extension combined with KN5 model depending on number of classes; measured on phone calls (BH) corpora.*

4.2 Model evaluation

As a training and test data for models evaluation we used Czech spontaneous speech which was recorded from phone calls. These calls were acquired as "Free calls" where people could phone for free while giving the permission to use anonymously their calls for the speech recognition experiments. We had to deal with the task where recorded data were very different from a common written Czech language. This is not a trivial task, as shown in previous work [5], where the records with spontaneous speech were also processed.

The data are specified by:

- a high inflection of Czech language (cases, various verb forms,...)
- word inflection is partially different from written Czech language
- unusual words used by speakers (slang, diminutives,...)
- only a small set of data available (about 2.8M words)
- the records contain a lot of non-speech events
- the sentences are relatively short
- the vocabulary is relatively small (about 120k words)

The statistics of the used corpus are shown in Table 3; the corpora was divided into tree parts: training, development and test set. The characteristics of the test phone records for models evaluation on speech recognition are shown in Table 4.

We took our state-of-the-art LVCSR system, as a language model we used 3-gram Knesser-Ney back-off model and finally n-best hypothesis with $n = 1000$ from the lattices were extracted. Hereafter, this n-best list will be the base for our experiments with language models.

As the baseline model we used 5-gram Knesser-Ney statistical model (KN5) [14] trained from the same corpus with a full vocabulary. The LSTM language models were

	Sentences	Words	OOVs
Train	400k	2.2M	
Dev	3k	13k	350
Test	3k	14k	385

Table 3. *BH text data.*

Records	
Length h:mm	2:16
Sentences	3582
Speakers	50

Table 4. *BH test records.*

trained from data with a limited vocabulary, where only 10k most frequent words were used, and all were combined with KN5 model with linear interpolation. The width of hidden layer was again fixed to 20 cells for all models. As described above, we chose the parameters of models on development part of data, in terms of perplexity.

We advanced the KN5 baseline by $\approx 3.7\%$ in relative, models with extended features slightly overcome the basic LSTM model, according to perplexity results; the improvement is statistically significant with $p = 0.05$. The complete results are shown in Table 5. The theoretical maximum that we eventually could obtain while we are rescoring this n-best list is 73.7% in accuracy.

model	Acc in %
LSTM LDA&CLASS +KN5	52.25
LSTM CLASS +KN5	52.05
LSTM LDA +KN5	51.95
LSTM +KN5	51.54
KN5 baseline model	50.41

Table 5. *Evaluating on speech recognition (1000-best list rescore).*

5 Conclusions

We have applied the LSTM neural network language model to spontaneous Czech speech. We explored several extensions to this approach: with LDA to explore long span context in dialogue and with classes to find similarities in topics.

We gained some not breakthrough but significant improvements in comparison to the basic model while applying these models in terms of perplexity and speech recognition. For future work it seems interesting to further discover the influence of input feature vectors and realize more experiments with another corpora.

Acknowledgements

This research was supported by the Ministry of Culture Czech Republic, project No. DF12P01OVV022.

References

1. Frinken, V.; Zamora-Martinez, F.; Espana-Boquera, S.; Castro-Bleda, M.J.; Fischer, A.; Bunke, H., “Long-short term memory neural networks language modeling for handwriting recognition, Pattern Recognition,” 21st International Conference on Pattern Recognition (ICPR), 2012, vol., no., pp.701,704, 11-15 Nov. 2012
2. Mikolov, T.; Kombrink, S.; Burget, L.; Cernocky, J.H.; Khudanpur, Sanjeev, “Extensions of recurrent neural network language model,” 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol., no., pp.5528,5531, 22-27 May 2011
3. Martin Sundermeyer, Ralf Schlüter, Hermann Ney; “LSTM Neural Networks for Language Modeling,” INTERSPEECH 2012
4. A. Stolcke (2002), “SRILM – An Extensible Language Modeling Toolkit,” Proc. Intl. Conf. on Spoken Language Processing, vol. 2, pp. 901-904, Denver.
5. Daniel Soutner, Zdeněk Loose, Ludek Müller, Ales Pražák: Neural Network Language Model with Cache. TSD 2012:528-534
6. Řehůřek, R., Sojka, P.; “Software Framework for Topic Modelling with Large Corpora,” In Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks. Valletta, Malta: University of Malta, 2010. p. 46–50, 5 pp. ISBN 2-9517408-6-7.
7. David M. Blei and Andrew Y. Ng and Michael I. Jordan and John Lafferty, “Latent dirichlet allocation,” Journal of Machine Learning Research, 2003, vol.3.
8. Sepp Hochreiter and Jürgen Schmidhuber; “Long Short-term Memory,” Neural Computation 9(8), 1997, pp. 1735-1780.
9. Felix Gers (2001); “Long Short-Term Memory in Recurrent Neural Networks,” Ph.D. Thesis. École Polytechnique Fédérale de Lausanne, Switzerland.
10. P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai and R. L. Mercer, “Class-Based n-gram Models of Natural Language,” Computational Linguistics 18(4), 467-479, 1992.
11. Eugene Charniak, et al.; 2000, BLLIP 1987-89 WSJ Corpus Release 1, Linguistic Data Consortium, Philadelphia.
12. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C., “A neural probabilistic language model,” J.Mach. Learn. Res. 3, 1137-1155 (2003).
13. J. Schmidhuber, D. Wierstra, M. Gagliolo, F. Gomez., “Training Recurrent Networks by Evolino,” Neural Computation, 19(3): 757-779, 2007. PDF (preprint).
14. Kneser, R.; Ney, H., “Improved backing-off for M-gram language modeling,” Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on , vol.1, no., pp.181,184 vol.1, 9-12 May 1995
15. Oparin, I., Sundermeyer, M., Ney, H., Gauvain, J., “Performance analysis of Neural Networks in combination with n-gram language models,” ;In ICASSP(2012)5005-5008
16. T. Mikolov and G. Zweig, “Context Dependent Recurrent Neural Network Language Model,” Microsoft Research Technical Report MSR-TR-2012-92, 2012.