Enarvi, Seppo; Kurimo, Mikko

TheanoLM - An extensible toolkit for neural network language modeling

Published in:
Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)

DOI:
10.21437/Interspeech.2016-618

Published: 01/01/2016

Document Version
Peer reviewed version

Please cite the original version:
Enarvi, S., & Kurimo, M. (2016). TheanoLM - An extensible toolkit for neural network language modeling. In Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH): San Francisco, USA, Sept. 8-12 (pp. 3052-3056). (Proceedings of the Annual Conference of the International Speech Communication Association). ISCA. https://doi.org/10.21437/Interspeech.2016-618
TheanoLM — An Extensible Toolkit for Neural Network Language Modeling

Seppo Enarvi, Mikko Kurimo

Aalto University, Finland
firstname.lastname@aalto.fi

Abstract

We present a new tool for training neural network language models (NNLMs), scoring sentences, and generating text. The tool has been written using Python library Theano, which allows researcher to easily extend it and tune any aspect of the training process. Regardless of the flexibility, Theano is able to generate extremely fast native code that can utilize a GPU or multiple CPU cores in order to parallelize the heavy numerical computations. The tool has been evaluated in difficult Finnish and English conversational speech recognition tasks, and significant improvement was obtained over our best back-off n-gram models. The results that we obtained in the Finnish task were compared to those from existing RNNLM and RWTHLM toolkits, and found to be as good or better, while training times were an order of magnitude shorter.

Index Terms: language modeling, artificial neural networks, automatic speech recognition, conversational language

1. Introduction

Neural network language models (NNLMs) are known to outperform traditional n-gram language models in speech recognition accuracy [1, 2]. For modeling word sequences with temporal dependencies, the recurrent neural network (RNN) is an attractive model as it is not limited to a fixed window size. Perhaps the simplest variation of RNN that has been used for language modeling contains just one hidden layer [3]. The ability of an RNN to model temporal dependencies is limited by the vanishing gradient problem. Various modifications have been proposed to the standard RNN structure that reduce this problem, such as the popular long short-term memory (LSTM) [4].

Figure 1 shows a typical LSTM network. Following the architecture by Bengio et al [5], the first layer projects the words \( w_t \) into a continuous lower-dimensional vector space, which is followed by a hidden layer. In recurrent networks the hidden layer state \( h_t \) is passed to the next time step. LSTM is a special case of a recurrent layer that also passes cell state \( C_t \). Sigmoid gates control what information will be added to or removed from the cell state, making it easy to maintain important information in the cell state over extended periods of time. The final neural network layer normalizes the output probabilities using softmax.

Another influential design choice concerns performance when the vocabulary is large. The computational cost of training and evaluating a network with softmax output layer is highly dependent on the number of output neurons, i.e. the size of the vocabulary. Feedforward networks are basically n-gram models, so it is straightforward to use the neural network to predict only words in a short-list and use a back-off model for the infrequent words [6]. Replacing a single softmax layer with a hierarchy of softmax layers [7] improves performance, although the number of model parameters is not reduced. Using word classes in the input and output of the network [8] has the benefit that the model becomes smaller, which may be necessary for using a GPU.

While several toolkits have been made available for language modeling with neural networks [6, 8, 9], they do not support research and prototyping work well. The most important limitation is the difficulty in extending the toolkits with recently proposed methods and different network architectures. Also, training large networks is very slow without GPU support. TheanoLM is a new NNLM tool that we have developed, motivated by our ongoing research on improving conversational Finnish automatic speech recognition (ASR). Our goal is to make it versatile and fast, easy to use, and write code that is easily extensible.

In the next section we will give an introduction on how TheanoLM works. Then we will evaluate it on two conversational ASR tasks with large data sets. In order to verify that it works correctly, the results will be compared to those from other toolkits.
In this work we use class-based models, where the learning rate enables finer adjustments later in the training as opposed to the number of words. The advantage of this solution is that the model size depends on the number of classes instead of the number of words.

The neural network is represented in Python objects as a symbolic graph of mathematical expressions. Theano performs symbolic differentiation, making it easy to implement gradient-based optimization methods. We have already implemented Nesterov’s Accelerated Gradient [11], Adagrad [12], Adadelta [13], Adam [14], and RMSProp optimizers, in addition to plain Stochastic Gradient Descent (SGD).

Evaluation of the expressions is performed using native CPU and GPU code transparently. The compilation does introduce a short delay before the program can start to train or use a model, but in a typical application the delay is negligible compared to the actual execution. On the other hand, the execution can be highly parallelized using a GPU, speeding up training of large networks to a fraction of CPU training times.

Standard SGD training is very sensitive to the learning rate hyperparameter. The initial value should be as high as possible, given that the optimization still converges. Gradually decreasing (annealing) the learning rate enables finer adjustments later in the optimization process. TheanoLM can perform cross-validation on development data at regular intervals, in order to decide how quickly annealing should occur. However, adaptive learning rate methods such as Adagrad and Adadelta do not require manual tuning of the learning rate—cross-validation is needed only for determining when to stop training.

Especially in Finnish and other highly agglutinative languages the vocabulary is large for the final layer to predict words directly. In this work we use class-based models, where each word \( w_t \) belongs to exactly one word class \( c(w_t) \):

\[
P(w_t|w_{t-1}, \ldots) = P(c(w_t)|c(w_{t-1}), \ldots)P(w_t|c(w_t))
\]

When the classes are chosen carefully, this model will not necessarily perform worse than a word-based model in Finnish data, because there are not enough examples of the rarer words to give robust estimates of word probabilities. The advantage of this solution is that the model size depends on the number of classes instead of the number of words.

An arbitrary network architecture can be provided in a text file as a list of layer descriptions. The layers have to be specified in the order that the network is constructed, meaning that the network has to be acyclic. The network may contain word and class inputs, which have to be followed by a projection layer. A projection layer may be followed by any number of LSTM and GRU [15] layers, as well as non-recurrent layers. The final layer has to be a softmax or a hierarchical softmax layer. Multiple layers may have the same element in their input, and a layer may have multiple inputs, in which case the inputs will be concatenated.

The example in Figure 2 would create an LSTM network with a projection layer of 500 neurons and two hidden layers of 1500 neurons. After each of those layers is a dropout [16] layer, which contains no neurons, but only sets some activations randomly to zero at train time to prevent overfitting. This is the configuration that we have used in this paper to train the larger TheanoLM models.

LSTM and GRU architectures help avoid the vanishing gradient problem. In order to prevent gradients from exploding, we limit the norm of the gradients, as suggested by Mikolov in his thesis [2].

### 3. Conversational Finnish ASR Experiment

#### 3.1. Data

In order to develop models for conversational Finnish ASR, we have recorded and transcribed conversations held by students in pairs, in the basic course in digital signal processing at Aalto University. The collected corpus, called *Aalto University DSP Course Conversation Corpus (DSPCON)* is available for research use in the Language Bank of Finland. The corpus is updated annually; currently it includes recordings from years 2013–2015. In addition we have used the spontaneous sentences of SPEECON corpus, FinDialogue subcorpus of FinINTAS, and some transcribed radio conversations, totaling 34 hours of acoustic training data. This is practically all the conversational Finnish data that is available for research.

We have collected language modeling data from the Internet, and filtered it to match transcribed conversations [17]. Similar data is available in the Language Bank of Finland. This was augmented with 43,000 words of transcribed spoken conversations from DSPCON, totaling 76 million words. A 8,900-word development set was used in language modeling.

A set of 541 sentences from unseen speakers, totaling 44 minutes and representing the more natural spontaneous speech

---

1. https://github.com/senarvi/theanolm
2. http://urn.fi/urn:nbn:fi:lb-2015101901
3. http://urn.fi/urn:nbn:fi:lb-20140730194
4. http://urn.fi/urn:nbn:fi:lb-201412171

---

**Figure 2:** An example of a network architecture description.
Table 1: Language model training times and word error rates (%) given by the model alone and interpolated with the best Kneser-Ney model. Finnish results are from the evaluation set, but the same set was used for optimizing language model weights.

| Model          | Training time | Dev | Eval |
|---------------|--------------|-----|-----|
|               |              | Alone | Int. | Alone | Int. |
| **Finnish**   |              |       |     |       |     |
| KN word 4-gram | 7 min        | 52.1 |     |       |     |
| KN word 4-gram weighted | 19 min | 51.0 |     |       |     |
| KN class 4-gram weighted | 55 min | 51.2 | 50.5 |       |     |
| RNNLM 300     | 361 h        | 50.4 | 49.8 |     |     |
| RWTHLM 100+300+300 | 480 h† | 49.4 | 48.6 |     |     |
| TheanoLM 100+300+300 | 20 h | 49.1 | 49.0 |     |     |
| TheanoLM 500+1500+1500 | 139 h | 48.7 | 48.4 |     |     |
| **English**   |              |       |     |       |     |
| KN word 4-gram | 8 min        | 42.1 |     |       |     |
| KN word 4-gram weighted | 15 min | 41.0 |     | 41.2 |       |
| TheanoLM 100+300+300 | 62 h | 40.1 | 39.6 | 41.2 | 40.5 |
| TheanoLM 500+1500+1500 | 290 h | 39.6 | 39.5 | 40.5 | 40.0 |

† Using 12 CPUs.

Both speed of convergence and performance of the final model. Nesterov’s Accelerated Gradient with manual annealing gave a slightly better model with considerably longer training time. The other toolkits used standard SGD.

Kneser-Ney smoothed 4-grams were used in the back-off models. The data sets collected from different sources varied in size and quality. Instead of pooling all the data together, the baseline back-off models were weighted mixtures of models estimated from each subset. The weights were optimized using development data. The back-off model vocabulary was limited to 200,000 of the 2,400,000 different word forms that occurred in the training data, selected with the EM algorithm to maximize the likelihood of the development data [21]. Out-of-vocabulary rate in the evaluation data was 5.06%.

3.3. Results

Evaluation of neural network probabilities is too slow to be usable in the first decoding pass of a large-vocabulary continuous speech recognizer. Another pass is performed by decoding and rescoring word lattices created using a traditional n-gram model. RWTHLM is able to rescore word lattices directly, the other toolkits can only rescore n-best lists created from word lattices.

Word error rates (WER) after rescoring are shown in Table 1. The table also includes word error rates given by interpolation of NNLm scores with the KN word 4-gram weighted model. We interpolated the sentence scores (log probabilities) as

$$
\log P = (1 - \lambda)s_{nn} \log P_{nn}(w_1 \ldots w_n) + \lambda s_{bo} \log P_{bo}(w_1 \ldots w_n).
$$

$s_{bo}$ is the LM scale factor that was optimal for the back-off model. The same value was used for generating the n-best lists. $s_{nn}$ and $\lambda$, the NNLm scale factor and interpolation weight, were optimized from a range of values.

Vocabulary size affects perplexity computation, so we omitted the limited-vocabulary models from the perplexity results in Table 2. Finnish vocabulary was five times larger, so the perplexities are considerably higher than in the English language experiment, as expected. The values are as reported by each tool; there might be differences in e.g. how unknown words and sentence starts and ends are handled. Perplexities of the Kneser-Ney models were computed using SRILM, which excludes unknown words from the computation. With TheanoLM we used the same behaviour (--unk-penalty 0).
We offer a new toolkit that has been implemented using Python.

Another important advantage is speed—Theano provides highly optimized C++ implementations of the expensive matrix operations, and can automatically utilize a GPU.

While back-off n-gram models are still two orders of magnitude faster to train, training a model using TheanoLM was an order of magnitude faster than training a similar model with the other toolkits. The speed advantage of TheanoLM was mainly due to GPU support, but the Adagrad optimizer that we used with TheanoLM was also faster to converge. 4 epochs were required for convergence using Adagrad, while the other toolkits continued to improve perplexity for at least 8 epochs. The faster training time makes it practical to train larger networks with TheanoLM. When more data is available, the benefit of a larger network becomes pronounced, and training without GPU support becomes impractical.

In a conversational Finnish speech recognition task we have used practically all the data that is available for research. A 4-gram Kneser-Ney model, trained simply by concatenating the training data, gave us 52.1 % WER. As a baseline we took a mixture model that was combined from smaller models with optimized interpolation weights. The baseline model gave 51.0 % WER. 48.4 % WER was reached when interpolating TheanoLM and baseline LM scores, a relative improvement of 51.1 %. RWTHLM gives similar results with a similar neural network architecture, which increases our confidence in the correctness of the implementation.

Evaluating the progress we have made in the conversational Finnish task, 48.4 % WER is 10.5 % better than our previous record, 54.1 % [17]. However, we have collected new acoustic data, which explains the 51.0 % baseline in this paper. The acoustic models used in these experiments are still not state of the art. In order to find out the absolute performance of these models, we will also need to obtain a proper development set for optimizing language model scale and interpolation weight.

It appears that some types of NNLMs work better with one language than another. RNNLM did not perform as well as we expected in the Finnish task, probably because the network architecture was not optimal. RNNLM offers only as well a simple network architecture that takes words as input and contains one hidden layer. The number of input connections is huge with the Finnish vocabulary, and the size of the hidden layer is limited if we want training time to be reasonable. Previously interpolating RNNLM with a Kneser-Ney model has been shown to give 3 to 8 % relative improvement in conversational English ASR [23].

In the English meeting recognition task more data was available, and the smaller neural network was not better than the weighted mixture back-off model. Still the neural network brings new information so much that interpolation of the smaller NNLM scores with back-off scores improves from the baseline of 41.2 % WER to 40.5 %. The larger network brings 2.9 % improvement to 40.0 % WER. When more data is available, a larger network is needed, which makes training speed even more important.

So far our research focus has been on conversational Finnish and other highly agglutinative languages. Considering that as much effort has not gone into the English task, the results are satisfactory, and show that TheanoLM works also in a relatively standard conversational English task.

5. Conclusions

Several different NNLM toolkits, implemented in C++, are freely available. However, the field is rapidly changing, and a toolkit should be easily extensible with the latest algorithms. Also, different languages and data sets work best with different algorithms and architectures, so a good solution needs to be versatile.

We offer a new toolkit that has been implemented using Python and Theano, provides implementations of the latest training methods for artificial neural networks, and is easily extensible. Another important advantage is speed—Theano provides highly optimized C++ implementations of the expensive matrix operations, and can automatically utilize a GPU.

6. Acknowledgements

This work was financially supported by the Academy of Finland under the grant number 251170 and made possible by the computational resources provided by the Aalto Science-IT project.
7. References

[1] H. Schwenk and J. L. Gauvain, “Connectionist language modeling for large vocabulary continuous speech recognition,” in Proc. 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 1, May 2002, pp. I–765–I–768.

[2] T. Mikolov, “Statistical language models based on neural networks,” Ph.D. dissertation, Brno University Of Technology, 2012. [Online]. Available: http://www.fit.vutbr.cz/research/view_pub.php?id=10158

[3] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, “Recurrent neural network based language model,” in Proc. 11th Annual Conference of the International Speech Communication Association (INTERSPEECH), Sep. 2010, pp. 1045–1048. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html

[4] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[5] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” The Journal of Machine Learning Research, vol. 3, pp. 1137–1155, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=94919.944966

[6] H. Schwenk, “CSLM - a modular open-source continuous space language modeling toolkit,” in Proc. 14th Annual Conference of the International Speech Communication Association (INTERSPEECH), Aug. 2013, pp. 1198–1202. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2013/i13_1198.html

[7] F. Morin and Y. Bengio, “Hierarchical probabilistic neural network language model,” in Proc. 10th International Workshop on Artificial Intelligence and Statistics (AISTATS). Society for Artificial Intelligence and Statistics, 2005, pp. 246–252.

[8] M. Sundermeyer, R. Schlüter, and H. Ney, “rwwth — the RWTH Aachen University neural network language modeling toolkit,” in Proc. 15th Annual Conference of the International Speech Communication Association (INTERSPEECH), Sep. 2014, pp. 2093–2097.

[9] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Černocký, “RNNLM - recurrent neural network language modeling toolkit,” in Proc. 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU). IEEE Signal Processing Society, Dec. 2011, IEEE Catalog No.: CFP11SRW-USB. [Online]. Available: http://www.fit.vutbr.cz/research/view_pub.php?id=10087

[10] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, “Theano: new features and speed improvements,” NIPS 2012 Workshop on Deep Learning and Unsupervised Feature Learning, 2012.

[11] Y. Nesterov, “A method of solving a convex programming problem with convergence rate O(1/k²),” Soviet Mathematics Doklady, vol. 27, no. 2, pp. 372–376, 1983.

[12] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” The Journal of Machine Learning Research, vol. 12, pp. 2121–2159, Jul. 2011.

[13] M. D. Zeiler, “ADADELTA: an adaptive learning rate method,” Computing Research Repository, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[14] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. International Conference on Learning Representations (ICLR) 2015, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[15] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proc. 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. [Online]. Available: http://arxiv.org/abs/1406.1078

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[17] M. Kurimo, S. Enarvi, O. Tilk, M. Varjokallio, A. Mannikainen, and T. Alumäe, “Modeling under-resourced languages for speech recognition,” Language Resources and Evaluation (LRE), 2016.

[18] T. Hirsimäki, J. Pylkkönen, and M. Kurimo, “Importance of high-order n-gram models in morph-based speech recognition,” IEEE Transactions on Audio, Speech, and Language Processing, vol. 17, no. 4, pp. 724–732, 2009.

[19] R. Kneser and H. Ney, “Forming word classes by statistical clustering for statistical language modelling,” in Contributions to Quantitative Linguistics, R. Köhler and B. Rieger, Eds. Springer Netherlands, 1993, pp. 221–226. [Online]. Available: http://dx.doi.org/10.1007/978-94-011-1769-2_15

[20] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur, “Extensions of recurrent neural network language model,” in Proc. 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2011, pp. 5528–5531.

[21] A. Venkataraman and W. Wang, “Techniques for effective vocabulary selection,” in Proc. 8th European Conference on Speech Communication and Technology (EUROSPEECH), Sep. 2003. [Online]. Available: http://www.isca-speech.org/archive/eurospeech_2003/e03_0245.html

[22] 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 Proc. 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU). IEEE Signal Processing Society, Dec. 2011, IEEE Catalog No.: CFP11SRW-USB.

[23] S. Kombrink, T. Mikolov, M. Karafiát, and L. Burget, “Recurrent neural network based language modeling in meeting recognition,” in Proc. 12th Annual Conference of the International Speech Communication Association (INTERSPEECH), Aug. 2011, pp. 2877–2880. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2011/i11_2877.html