Multi-domain Neural Network Language Model

Tanel Alumäe

Institute of Cybernetics at Tallinn University of Technology, Tallinn, Estonia

tanel.alumae@phon.ioc.ee

Abstract

The paper describes a neural network language model that jointly models language in many related domains. In addition to the traditional layers of a neural network language model, the proposed model also trains a vector of factors for each domain in the training data that are used to modulate the connections from the projection layer to the hidden layer. The model is found to outperform simple neural network language models as well as domain-adapted maximum entropy language models in perplexity evaluation and speech recognition experiments.

Index Terms: neural network language model, language model adaptation

1. Introduction

Neural network language models (NNLMs) have become a popular choice for various large vocabulary continuous speech recognition (LVCSR) tasks in the recent decade. NNLMs have been proved to consistently outperform traditional back-off n-gram language models (LMs). Usually, NNLMs are used to rescore word hypothesis lattices generated using an n-gram model, resulting in a typical word error rate improvement of about one percentage point [1, 2]. More recently, NNLMs have been also successfully applied in machine translation [3]. Increase of CPU speed and memory sizes, improvements of NNLM training algorithms and their efficient implementations have made it possible to train NNLMs on large text corpora.

In LVCSR, a LM is usually estimated from large amounts of written text data. However, LMs are typically applied to speech that is stylistically different from written language. For example, speech recognition is often applied to broadcast news, that includes introductory segments, conversations and spontaneous interviews. To decrease the mismatch between training and test data, often a small amount of speech data is human-transcribed. A LM is then built by interpolating the models estimated from large corpus of written language and the small corpus of transcribed data.

The model interpolation approach is however impractical for NNLMs. NNLMs learn a hidden layer that projects words into continuous space. Learning a separate projection layer for each domain does not allow to take advantage of the inherent similarity of data in different domains. Furthermore, NNLMs have typically numerous hyperparameters that must then be optimized for each individual model. Instead, domain-optimized NNLMs are often trained by using large amounts of out-of-domain data and available in-domain data together, with sampling a small random subset of out-of-domain data at each epoch to make training faster and bias the model towards in-domain data [4]. Another method for creating a domain-optimized NNLM is to first train a general model from large amounts of out-of-domain data, and then adapt the model to a certain domain by introducing a new adaptation layer, while keeping the previously trained layers fixed [5].

This paper describes a new NNLM architecture that aims to learn the language model of multiple domains jointly. The idea of jointly training a statistical model to many domains is not new. For example, such approaches have been used with maximum entropy models [6] and have been also applied to maximum entropy LMs [7]. Our model uses a somewhat similar idea, but using a very different architecture. It is also different from previous adaptation approaches applied to NNLMs: instead of first training a general model and then introducing an adaptation layer, our model includes the adaptation layer from the beginning. Furthermore, the adaptation layer is able to handle any number of domains in the data. Many parameters of the adaptation layer are tied across domains, helping the model to leverage the similarities between domains.

The paper first gives a short review of NNLMs, followed by the motivation and description of the multi-domain NNLM. The proposed model is tested on three different datasets using perplexity evaluations and on one dataset using lattice rescoring experiments.

2. Review of neural network language models

Back-off n-gram models estimate contextual probabilities of different words in a discrete space. Word n-gram probabilities are based on smoothed frequencies of word n-grams in the training corpus. Such models contain probability estimates only for n-grams that are seen in the training data. Probabilities of unseen n-grams are estimated using the back-off technique. This prevents any kind of interpolation to estimate word n-gram probabilities based on other similar n-grams that occurred in the training data. NNLMs try to solve this problem by projecting words into continuous space and performing the probability estimation in this space. Learning of the projection and probability estimation parameters are usually performed jointly by training a multi-layer neural network [8].

The most widely used NNLM architecture is shown in Figure 1. Word contextual probabilities are computed based on $n-1$ previous words (the word history), with a fixed context vocabulary size $|V|$. Input to the neural network consists of indexes of the $n-1$ previous words. The indexes use the so-called one-hot encoding, i.e., word k at position $i-1$ is encoded by setting the k-th element of the corresponding vector to one and all other elements to zero. The resulting feature vector is first processed by a projection layer. Projection layer is a linear hidden layer with $(n-1) \times |V|$ inputs and $(n-1) \times N_{proj}$ outputs (N_{proj} is thus the output dimensionality of the projection layer elements). Weights of the projection layer are shared for different word positions in the history context, i.e., the corresponding projection layer parameters are *tied*. Outputs of the projection layer are fed to a hidden layer with N_{hid} units which usually use

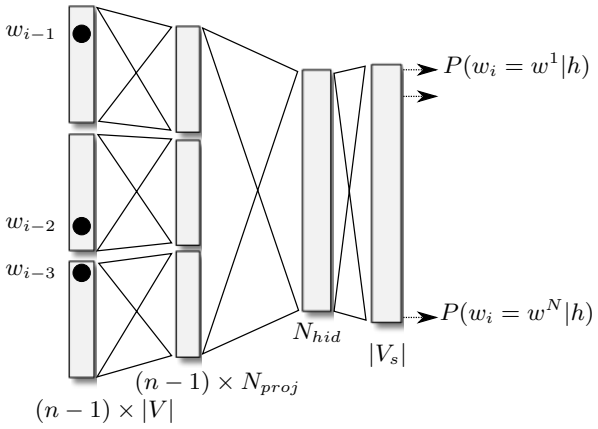

Figure 1: Architecture of a typical neural network language model.

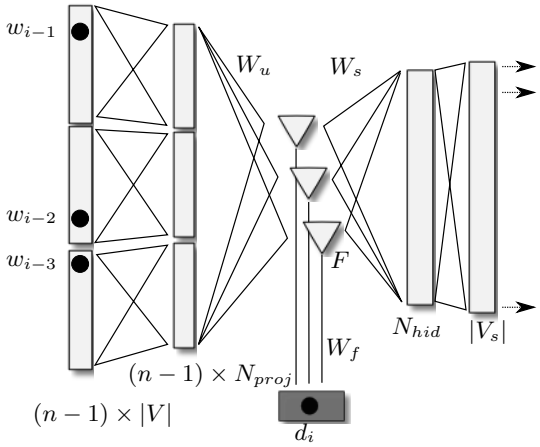

Figure 2: Architecture of the multi-domain neural network language model.

a non-linear activation function (typically sigmoid or the hyperbolic tangent function). Hidden layer activities are processed by a linear output layer which also applies the softmax function to its outputs in order to make it a valid probability distribution. Often, in order to make NNLM training time feasible, the likelihoods are calculated only for a subvocabulary of N most frequent words (the *shortlist*, V_S). Likelihoods of out-of-shortlist (OOS) words are calculated using a traditional n-gram model. Probabilities of in-shortlist words given the word history h must thus be normalized as follows:

$$P(w_t|h) = \begin{cases} P_{NNLM}(w_t|h)\alpha_S(h) & \text{if } w_t \in V_s \\ P_{ngram}(w_t|h) & \text{otherwise.} \end{cases}$$

$$\alpha_S(h) = \sum_{w \in V_s} P_{ngram}(w|h)$$

Alternatively, one can have a special output node in the NNLM that estimates the likelihood over all OOS words [5]. In this case, the NNLM probability estimates for the in-shortlist don't require further normalization, however, the n-gram estimates for OOS words have to be normalized using the NNLM likelihood of the OOS node. The use of shortlists can be avoided altogether if a structured ouput layer is used [9]. The structured output layer consists of several softmax layers applied in hierarchical manner which helps to reduce the overhead of calculating the normalization factor over the words in the output vocabulary.

NNLMs can be trained using back-propagation. Training data negative log-likelihood with L^2 regularization penalty (often called weight decay in the context of neural network models) is used as a cost function. Training is carried out until convergence or until the perplexity of a development set stops to decrease (to prevent overfitting). Stochastic gradient descent with batch mode updates is commonly used.

3. Multi-domain neural network language model

The main idea of the proposed multi-domain NNLM is to use the domain of the text (both during training and testing) as an additional context element, in addition to the context words. The context of the data thus cooperates with the word history to model the output distribution jointly. Thus, instead of building a general model first and then adapting it to a certain domain, we build a model where the existance of different domains is built in. The motivation behind this idea is that many domains are inherently similar and using the joint model we wish to take advantage of data sharing between the domains.

There is no single obvious way how to blend the context of the word history with the context of the domain in the NNLM. One could simply add the context as an additional input to the hidden layer and hope that the model learns the complex relationships of context and the domain. However, we know that the word history is a more dominant element of the two context elements and the domain should merely act as a modifier. Another somewhat naive approach would be to add an additional layer between the projection layer and the output layer, and swap this layer (i.e., the parameter matrix) depending on the active domain. This would however require $N_{domain} \times (n-1) \times N_{proj} \times N_{hid}$ parameters and could very possibly make the neural network overfit unless we had huge amounts of training data from each domain available. Our goal is to have a different parameter matrix for each domain, but we want these matrices to share parameters, as we know that many aspects of the language overlap across different domains, and similar domains should have similar weight matrices (e.g., the web and newspaper domains are likely to be much more similar to each other than newspaper and conversational domain).

We propose to model the interactions between the word context and the domain multiplicatively using factors (see Figure 2). The idea is inspired by the factored conditional restricted Boltzmann machine model [10] that was used to similarly model the interactions between the type and style of human motion. Their model uses multiplicative three-way interactions that allow the the transition between two units to be modulated by the dynamic state of the third unit. In our model, the transition between the projection layer and the hidden layer is computed using F factors. Each factor first computes a weighted sum of the projection layer outputs, using a factor-specific (and domain independent) weight vector (the weights are thus represented by a $((N_{proj} \times (n-1)) \times F)$ matrix W_u). Then, the weighted sum is multiplied by a domain and factor-specific scalar (we also add a factor-specific bias term to each scalar, thus the domain-specific scales are represented by a $((N_{domains}+1) \times F)$ matrix W_f). Finally, the resulting scalar

is used to provide weighted input to the hidden layer (using a matrix of $(F \times N_{hid})$ weights W_s).

The transition between the projection layer and hidden layer can be efficiently calculated using vector-matrix algebra. Weighted input vector to the hidden layer z_{hidden} is calculated from the output (column) vector of the projection layer y_{proj} as follows:

$$z_{hidden} = \left[\left(y_{proj}^T W_u \odot \left(w_f^{(d_i)} + w_f^{(bias)} \right) \right) W_s \right]^T$$

where $\odot$ stands for elementwise multiplication and $w_f^{(d_i)}$ is the d_i-th row vector of the matrix W_f, with d_i being the domain of the text.

Our goal was to have a different transition from the projection layer to the hidden layer for each domain, with sharing of the parameters between domains. In the proposed scheme, the transition is governed by three matrices W_u, W_f and W_s. The weight matrices W_u and W_s are shared by the domains. Only the matrix W_f, i.e., F-dimensional scale vector for each domain, is used to model the differences between the domains. The optimal value for F has to be determined by validation. In our experiments, we found that values of F in the same range as the output dimensionality of the projection layer and input dimensionality of the hidden layer worked consistently well.

The multiplicative layer adds relatively little to the complexity of the NNLM. The complexity to calculate one probability distribution with the basic NNLM is roughly:

$$O = (n-1) \times N_{proj} \times N_{hid} + N_{hid} + N_{hid} \times |V_s|$$

For the multi-domain NNLM, the complexity is:

$$O = (n-1) \times N_{proj} \times F + F + F \times N_{hid} + N_{hid} + N_{hid} \times |V_s|$$

With typical values of $n = 4$, $N_{proj} = 100$, $F = 300$, $N_{hid} = 500$ and $|V_s| = 1024$, the complexity increases from 662 500 to 752 800.

4. Experiments

We analyzed the performance of the multi-domain NNLM on English, French and Estonian data. The models were tested based on perplexity results. For Estonian data, also speech recognition experiments were performed.

We admit that sizes of LM training data are very small compared to the state-of-the-art systems. This is caused by the high complexity of the NNLM training algorithm and the unoptimized nature of our implementation. However, it is relatively straightforward to implement the multi-domain NNLM training procedure so to handle very large corpora, as has been proposed for traditional NNLMs [4, 3].

4.1. Data

The sizes of training corpora and the vocabulary sizes for different languages are listed in Table 2. In the following, detailed description of the origin and properties of the corpora is given.

English Broadcast News. Data comes from two domains: newswire and broadcast news (BN) transcriptions. The training data for the newswire domain consists of randomly selected 500K sentences from the Gigaword (2nd ed.) corpus. Training and development data for the BN domain contains transcriptions from the TDT4 corpus. About 950K words were used as training data and 24K words as development data. The 2003 NIST Rich Transcription Evaluation Data (25K words) was used as evaluation data for the BN domain.

Table 2: Characteristics of the training data and vocabulary sizes for different language modeling tasks. The last two columns give the out-of-vocabulary (OOV) and out-of-shortlist (OOS) rates of the corresponding evaluation sets.

Language	|Vocab|	Domain	#Words	OOV	OOS
English	26K	Newswire	13M		
		BN	950K	2.5	26.4
French	40K	Newspaper	4.5M		
		BN	500K	2.6	24.9
		BC	300K	2.3	22.0
Estonian	60K	Newspaper	1.4M		
		Web	1.7M		
		BN	133K	3.5	40.6
		BC	293K	3.3	34.7

French Broadcast News and Broadcast Conversations. French data comes from three domains: newspapers, BN and broadcast conversations (BC). Newspaper training data contains texts from the newspaper *Le Monde*, randomly selected over its 2006 edition. Training, development and evaluation data for the BN domain contain manual transcriptions from the ESTER corpus (500K, 83K, and 121K words for training, development and evaluation). BC data contain manual transcriptions of conversational broadcast data from the EPAC project (300K, 22K, 49K for training, development and evaluation).

Estonian Broadcast News and Broadcast Conversations. Estonian data is divided into four domains: newspaper, web, BN and BC. Newspaper data contains a random subset of a larger newspaper corpus that contains data from different newspapers from 1994 to 2006. Web data is a random selection of a large web corpus, containing material from mainly 2007 onwards. BN data contains manual transcriptions of mostly dictated short radio news programs (133K, 18K and 18K words for training, development and evaluation). BC data contains manual transcriptions of spontaneous radio and TV interviews (293K, 9K and 6K words, respectively).

4.2. Model training

Both simple as well as multi-domain NNLMs were trained and evaluated using our own implementation based on the Theano [11] and Pylearn2[1] libraries for Python.

All NNLMs used a context of three previous words, a shortlist of 1024 most frequent vocabulary words, a projection layer of 100 linear units and a hidden layer of 500 units with a "rectified linear" activation function. All multi-domain NNLMs used 300 factors in the multiplicative layer. Models were trained using backpropagation, stochastic gradient descent, batch size of 200, learning rate of 0.1, momentum of 0.5. Weight decay was the only parameter that was optimized for all models based on the development sets. Training was performed until the perplexity of a development set didn't decrease for the last five epochs, and the model from the best-performing epoch was used for evaluation. For multi-domain models with many target domains, the average of the perplexity over the development sets was used as a termination criterion.

4.3. Perplexity experiments

We compared the perplexity of NNLMs against several different models and their interpolations. The baseline LM is a Kneser-Ney smoothed 4-gram LM that is built by estimating an indi-

[1] https://github.com/lisa-lab/pylearn2

Table 1: Perplexities of different language models and their interpolations on evaluation data, and the corresponding relative improvement over the N-gram model.

		N-gram	Adapted MaxEnt		Simple NNLM		Multi-domain NNLM		Simple NNLM + N-gram		Multi-domain NNLM + N-gram	
Estonian	BN	351	310	-11.7%	332	-5.4%	310	-11.7%	310	-11.7%	293	-16.5%
	BC	434	405	-6.7%	454	4.6%	405	-6.7%	406	-6.5%	386	-11.1%
English	BN	252	234	-7.1%	253	0.4%	236	-6.3%	224	-11.1%	218	-13.5%
French	BN	143	133	-7.0%	137	-4.2%	135	-5.6%	129	-9.8%	127	-11.2%
	BC	141	126	-10.6%	147	4.3%	133	-5.7%	130	-7.8%	126	-10.6%
Average				-8.6%		-0.1%		-7.2%		-9.4%		-12.6%

vidual LM from the training corpus of each domain, and then interpolating the domain-specific LMs into the final one using interpolation coefficients optimized on the development set of a particular domain.

As a second comparison, we used maximum entropy (ME) models with 4-gram features. The models were adapted for each domain using the implementation in the SRILM extension [12]: first, a "global" ME model over all training data was estimated. Then, the global model was used as a prior for estimating a new model for each target domain.

The perplexities of the NNLMs were calculated using the optimized N-gram for the particular domain as the back-off model for computing the likelihoods of the OOS words. The proportions of tokens for which the back-off model had to be used (i.e., OOS rates) are reported in Table 2. For the simple and multi-domain NNLM, we also measured their performance when interpolated with the optimized N-gram model. The interpolation weights were optimized on development data. Perplexity results on evaluation data are reported in Table 1.

Multi-domain NNLMs gave better perplexity results than simple NNLMs in all cases. However, the difference was less evident when the NNLMs were interpolated with N-gram models. This is expected, since the N-gram models were optimized for the particular domain and hence helped to bring some domain-specific knowledge to the otherwise domain-independent simple NNLMs.

4.4. Speech recognition experiments

As we didn't have access to English and French acoustic models, the recognition experiments were performed only on the Estonian data.

Details of the Estonian transcription system are described in [13]. We use 140 hours of spoken data from various sources for training acoustic models (AMs). The models are continuous triphone HMMs with 2000 Gaussian mixtures that use 385 150 Gaussian distributions. Tied-state cross-word triphones estimated using maximum likelihood training are used to model 25 phoneme and two silence/filler units.

The transcription system performs speech/non-speech detection, speaker diarization and three decoding rounds, with CMLLR and MLLR adaptation between the passes. The same language models as used in perplexity experiments were applied. The optimized N-gram models were used during the decoding. The final pass produces a recognition lattice for each speech segment. The lattices were then rescored using the adapted maximum entropy models and NNLMs. NNLM scores were interpolated with scores of the baseline N-gram model. AM and LM weights and word insertion penalties were optimized on the development set, using grid search. To have a fair comparison, such optimization was also performed on the baseline system. Word error rate results on the evaluation sets of the two Estonian target domains are reported in Table 3. Statistically significant improvements over the N-gram system according to the Wilcoxon test are marked in bold. It can be seen that for the both domains, the multi-domain NNLM gave the best results, although the gains were not always statistically significant. The improvement over the baseline system was 0.9% (3.5% relative) and 1.2% (4.6% relative) for the BN and BC domains, respectively. The same improvements when using the simple NNLM were 0.7% (2.0% relative) and 0.7% (3.6% relative).

Table 3: Word error rates on Estonian BN and BC evaluation data after rescoring recognition lattices with different LMs.

LM	BN	BC
N-gram	19.4	34.8
Adapted MaxEnt	19.2	33.8
Simple NNLM + N-gram	**18.7**	34.1
Multi-domain NNLM + N-gram	**18.5**	33.6

5. Conclusion

The paper described a neural network language model architecture that jointly models language in multiple distinct but possibly related domains. The model uses the identity of the domain as an additional input to the neural network. The model includes a vector of factors for each domain, which are used to modulate the interaction between the projection and hidden layer of the NNLM. The resulting model can be used to estimate word likelihoods in any of the training domains and eliminates the need to optimize a separate model for each target domain. Experiments showed that this approach can improve LM perplexity and reduce speech recognition error rates.

Future work includes experiments with larger datasets, as well as modelling hierarchical or multilevel domain structures. The multilevel approach would make it possible to characterize each domain by a topic and style, and apply the model for any topic and style combination. For instance, this would allow to train the model on lecture and textbook data, and apply the model on lectures of a certain topic.

6. Acknowledgements

This research was supported by the Estonian Ministry of Education and Research target-financed research theme No. 0140007s12.

7. References

[1] L. Lamel, J.-L. Gauvain, V. B. Le, I. Oparin, and S. Meng, "Improved models for Mandarin speech-to-text transcription," in *ICASSP 2011*, 2011.

[2] G. Saon, H. Soltau, U. V. Chaudhari, S. M. Chu, B. Kingsbury, H.-K. Kuo, L. Mangu, and D. Povey, "The IBM 2008 GALE Arabic speech transcription system," in *ICASSP 2010*, 2010.

[3] H. Schwenk, A. Rousseau, and M. Attik, "Large, pruned or continuous space language models on a GPU for statistical machine translation," in *NAACL Workshop on the Future of Language Modeling*, 2012.

[4] H. Schwenk and J.-L. Gauvain, "Training neural network language models on very large corpora," in *EMNLP 2005*, ser. HLT '05, Vancouver, BC, Canada, 2005, pp. 201–208.

[5] J. Park, X. Liu, M. J. F. Gales, and P. C. Woodland, "Improved neural network based language modelling and adaptation," in *Interspeech 2010*, Chiba, Japan, 2010.

[6] H. Daume III, "Frustratingly easy domain adaptation," in *ACL 2007*, Prague, Czech Republic, 2007, pp. 256–263.

[7] T. Alumäe and M. Kurimo, "Domain adaptation of maximum entropy language models," in *ACL 2010*, Uppsala, Sweden, 2010, pp. 301–306.

[8] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[9] H. S. Le, I. Oparin, A. Messaoudi, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Large vocabulary SOUL neural network language models," in *Interspeech 2011*, Florence, Italy, 2011, pp. 1469–1472.

[10] G. W. Taylor and G. E. Hinton, "Factored conditional restricted Boltzmann machines for modeling motion style," in *ICML 2009*, Montreal, Quebec, Canada, 2009, pp. 1025–1032.

[11] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Python for Scientific Computing Conference (SciPy)*, Austin, TX, 2010.

[12] T. Alumäe and M. Kurimo, "Efficient estimation of maximum entropy language models with N-gram features: an SRILM extension," in *Interspeech 2010*, Chiba, Japan, 2010.

[13] T. Alumäe, "Transcription system for semi-spontaneous Estonian speech," in *Baltic HLT 2012*, Tartu, Estonia, 2012.