Human Language Technologies – The Baltic Perspective
A. Utka et al. (Eds.)

doi:10.3233/978-1-61499-442-8-149

Multi-Domain Recurrent Neural Network Language Model for Medical Speech Recognition

Ottokar TILK [a,1] and Tanel ALUMÄE [a]
[a] *Institute of Cybernetics at Tallinn University of Technology, Estonia*

Abstract. We evaluate back-off n-gram and recurrent neural network language models for an automatic speech recognition system for medical applications. We also propose an effective and simple multi-domain recurrent neural network architecture which enables training a joint model for all domains. The multi-domain recurrent neural network model outperforms all other compared models.

Keywords. Language modeling, recurrent neural network language model, multi-domain language model

Introduction

An important part of any automatic speech recognition (ASR) system is the language model (LM) which evaluates the probabilities of word sequences. The traditional approach for language modelling is the back-off n-gram LM which performs well but suffers from data-sparsity problem. Neural network (NN) LMs overcome this problem by projecting input words into continuous space and estimating word probabilities there [1]. While NN LMs enable larger contexts to be utilized than back-off n-gram LMs, they still are an n-gram approach (with fixed context length n-1). Recurrent neural network (RNN) LMs remove the fixed context length limitation and this seems to help as shown in [2] where RNN LMs outperform the feed-forward NN LMs.

LMs are typically trained on large text corpora of different domains with one or more of them being the target domain. Often the best model for the target domain is acquired by exploiting the inter-domain similarities. Back-off n-gram models exploit the similarities by finding optimal interpolation coefficients for combining all the domain-specific models into a target domain model. Maximum entropy and NN LMs can use the adaptation approach where the general model is carefully adjusted for the target domain [3,4]. Multi-domain NN LMs [5] can be trained for all domains simultaneously which is convenient if there are multiple target domains.

In this paper we train LMs for a medical ASR system using medical corpora. We compare n-gram and RNN LMs and propose a method to exploit the inter-domain similarities in the form of a novel multi-domain RNN LM.

[1] Corresponding Author. E-mail: ottokar.tilk@phon.ioc.ee

1. About the Medical Corpora

The medical corpora consist of radiology reports from 10 different domains (X-ray, computed tomography, ultrasound, etc.). All 10 domains are considered as target domains. The training set sizes of the domains vary in the range of 24.0K to 3.6M words with a total size of 10.8M words.

Our initial experiments with back-off n-gram LMs revealed some interesting properties of the medical corpora:

1. Including the general (non-medical) text corpora in the training of the medical LM is not practical;
2. Medical corpora consist of very different domains;
3. Exploiting the inter-domain similarities should give the best results.

Below we elaborate on these points in more detail.

What makes the medical applications model unique and difficult to train is that it has very specific vocabulary and style which is uncharacteristic to normal speech or text, thus limiting the usefulness of the general text corpora. This was reflected by the lack of improvement in terms of perplexity on the test set when we included the general text corpora in the training of the model.

Another challenge is that the medical dataset consists of texts from several very different domains. The differences are indicated by the very big (usually >0.9) optimal interpolation coefficients for the in-domain models when optimizing for the best combination of domain-specific n-gram language models for the target domain. For each domain there is a highly varying amount of training data available. Therefore, it is important for the method to be able to exploit the little similarities there are between the domains and to factor in the relatively big inter-domain differences while not overfitting the potentially small amount of in-domain data.

Our back-off n-gram LM experiments showed that linear interpolation of domain-specific models outperforms both a general model over all domains and using separate domain-specific models. This suggests that while the domains are different, they still share enough similarities to enable training a better model by exploiting the similarities in the whole set of medical corpora than training a separate model on each domain-specific corpus.

2. The Multi-Domain RNN LM Architecture

We implemented a multi-domain RNN LM inspired by the multi-domain NN LM from [5] because of its effectiveness, simplicity and very small number of domain-specific parameters. Small amount of domain-specific parameters enables the model to adapt to domains with little training data without the danger of overfitting. The method is also good at exploiting the similarities between the domains.

This method has not been used with RNN LMs before. Switching from feedforward to recurrent architecture requires finding a new way to apply the adaptation factors.

The first problem is deciding the location of the adaptation layer. The original multi-domain NN LM adds the adaptation layer between the projection and the hidden layer of the feedforward network. Our method puts the adaptation layer between the hidden

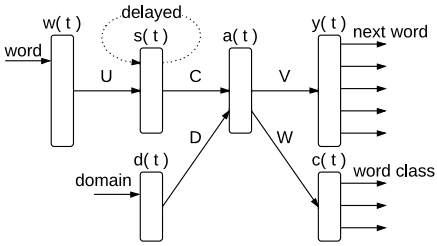

Figure 1. The Multi-Domain RNN LM.

Table 1. Comparison of models in terms of perplexity on the test set.

Model	PPL
n-gram	22.2
RNN	20.9
MD RNN	17.5
MD RNN + n-gram	15.6

and output layer of the recurrent network. This location requires the least amount of domain-specific parameters and enables the adaptation layer to double as a compression layer. Compression layer reduces the computational complexity of the model and was originally proposed in [6].

The second question is concerned with applying the domain-specific parameters in the adaptation layer. The multi-domain NN LM uses domain-specific multiplicative factors on the inputs to the adaptation layer. During experiments on NN LMs we discovered that this method requires shuffling the training samples to work properly. Using multiplicative factors on unshuffled training data was worse than using no adaptation at all. This seemed to be caused by the model parameters overfitting to the factors of more recently seen domains and becoming less compatible with domain factors seen earlier.

The shuffling requirement can not be satisfied with RNN LMs. For sequential training we propose a different method which, though benefits from shuffling, is much less sensitive to seeing long sequences of samples from the same domain. This method uses additive factors instead of multiplicative ones and can alternatively be thought of as a domain input (an idea also considered in [5]) or domain-specific bias to the adaptation layer. The adaptation layer state $a(t)$ at time step t in our approach is computed according to the following formula:

$$a(t) = f\big(s(t)C + d(t)D\big) \tag{1}$$

where $f(z)$ is the logistic sigmoid activation function; C is the weight matrix between the hidden state layer s and the adaptation layer a; D is the weight matrix between the domain input d and the adaptation layer a; $s(t)$ is the state of the hidden state layer and $d(t)$ is the domain input in the form of a one-of-N coded vector indicating the current domain (see Figure 1). The rest of the architecture is identical to the one described in [6].

Using addition instead of multiplication when applying the adaptation factors eliminates the adaptation factors from the gradients of the model parameters behind the adaptation layer. This way the parameters can fit to prediction error rather than adaptation parameters. Similar reasoning applies to factor gradients as well.

In our experiments on NN LMs, where we use shuffling of training samples, the additive factors show similar performance to multiplicative factors.

3. Experimental Results

We use a modified version of Mikolov's RNN LM toolkit [7] in our experiments. Both the simple and multi-domain RNN LM use the following set-up: number of classes in out-

put layer: 230; size of vocabulary: 52293; state layer size: 600; compression/adaptation layer size: 300; number of backpropagation through time steps: 3; learning rate: 0.1; L2 regularization coefficient: 1e-7. Additionally, the multi-domain RNN LM has 10 domain inputs. We use a single model for all domains in both RNN model experiments.

The baseline back-off n-gram model is a separate linearly interpolated model for each domain with interpolation weights optimized on the corresponding development set using the SRILM toolkit [8]. The order of the n-gram models is 4 and we use the modified Kneser-Ney discounting.

The results of the experiments can be seen in Table 1. A simple RNN LM brings a 6% relative perplexity improvement over interpolated back-off n-gram models. The multi-domain RNN LM increases the difference to 21% and linearly combining with the n-gram model raises it to 30%.

4. Conclusion

We trained and compared different types of LMs out of which the newly proposed multi-domain RNN LM turned out to be the most effective one in terms of perplexity. An additional benefit of the multi-domain architecture was the fact that it's a joint model for all domains thus enabling simple switching between target domains in contrast to the n-gram models where we had to use a different model for each domain.

In our future work we plan to perform ASR experiments to see how well the improvements in perplexity translate into improvements in word error rate. Testing the multi-domain RNN LM on other languages and domains is also in our plans.

Acknowledgments

This research was supported by the European Union through the European Regional Development Fund.

References

[1] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models," in *Innovations in Machine Learning*, pp. 137–186, Springer, 2006.

[2] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Černocký, "Empirical evaluation and combination of advanced language modeling techniques.," in *INTERSPEECH*, pp. 605–608, 2011.

[3] T. Alumäe and M. Kurimo, "Domain adaptation of maximum entropy language models," in *Proceedings of the ACL 2010 Conference Short Papers*, pp. 301–306, Association for Computational Linguistics, 2010.

[4] J. Park, X. Liu, M. J. Gales, and P. C. Woodland, "Improved neural network based language modelling and adaptation.," in *INTERSPEECH*, pp. 1041–1044, 2010.

[5] T. Alumäe, "Multi-domain neural network language model," in *INTERSPEECH*, pp. 2182–2186, 2013.

[6] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pp. 5528–5531, 2011.

[7] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Černocký, "RNNLM-recurrent neural network language modeling toolkit," in *Proc. of the 2011 ASRU Workshop*, pp. 196–201, 2011.

[8] A. Stolcke, "SRILM – an extensible language modeling toolkit.," in *Proc. Intl. Conf. on Spoken Language Processing*, pp. 901–904, 2002.