Combining Topic Specific Language Models

Yangyang Shi, Pascal Wiggers, and Catholijn M. Jonker

Man-Machine Interaction Group, Delft University of Technology
Mekelweg 4, 2628CD, Netherlands
shiyang1983@gmail.com

Abstract. In this paper we investigate whether a combination of topic specific language models can outperform a general purpose language model, using a trigram model as our baseline model. We show that in the ideal case — in which it is known beforehand which model to use — specific models perform considerably better than the baseline model. We test two methods that combine specific models and show that these combinations outperform the general purpose model, in particular if the data is diverse in terms of topics and vocabulary. Inspired by these findings, we propose to combine a decision tree and a set of dynamic Bayesian networks into a new model. The new model uses context information to dynamically select an appropriate specific model.

1 Introduction

A language model should capture the regularities in a language so that it can judge the fluency of a sequence of words. Statistical language models implement this notion of fluency as a probability that is assigned to every word sequence.

The n-gram language model, that conditions the probability of a word on the previous n words, is still the most popular language model. It has the advantages that the necessary statistics are easy to collect and that it captures much of the local regularity of language. As it only takes a local context into account, the model is also robust to distortions, a feature that is most useful if a sentence hypothesis processed by the language model may contain incorrect words, as is for example the case in speech recognition.

However, these models fail to capture important long range dependencies between words. Most attempts to improve the n-gram language model focus on modelling a larger part of the context, by taking a larger part of the history into account or by explicitly including syntactic [1] or semantic relations [2,3,4,5].

An assumption behind n-gram models, but also behind many other language models, is that the relative frequency of a word combination is the same for all situations. This is a useful assumption to ensure reliable parameter estimates, but it is clearly incorrect. For example, in this particular paper, the word sequence "language model" is much more likely than in an average text.

An alternative approach would be to create dedicated models for particular contexts, for example, a model trained on papers on language modeling might be a more accurate model for the text in this paper than a general purpose language model trained on a large, diverse set of texts. Such an approach leads to two questions: 1) do dedicated models indeed outperform general purpose models in the proper context? 2) if so, how can one select a dedicated model for a given situation.

I. Habernal and V. Matoušek (Eds.): TSD 2011, LNAI 6836, pp. 99–106, 2011.

In this paper, we investigate these two questions. To answer the first question we compared the performance of a trigram language model trained on our whole data set with the performance of dedicated models trained on specific parts of the data set. Our analysis is presented in Section 3. In Section 4 we turn to the second question. We present two language models based on dynamic Bayesian networks (DBNs) that combine multiple dedicated models. In Section 5 we propose a new adaptive model, called DBN tree, that builds upon these DBN models. This model dynamically selects a language model based on contextual information. We start with a brief overview of related work in section 2.

2 Related Work

The idea of combining multiple specific models has been used before in language modelling [6]. For example [7] interpolate multiple specific n-grams to obtain n-gram estimates. The mixture models of [8] combine multiple topic specific models at the sentence level, using weighted interpolation. Each of their component models is interpolated with a general n-gram model to handle data sparsity problem. The dynamic Bayesian network based language model of [4] is an extension of this work. They argue that sentence level mixture models can outperform other language models because they capture coherence in a text.

In terms of modeling techniques our models are closely related to those Bayesian network models and to the decision tree based language models of [9]. Although decision trees by themselves do not outperform the simpler n-grams in language modeling, they are a valuable component of more sophisticated models, such as random forrest language models [10].

3 Comparing Specific and General Models

The hypothesis behind our work is that models specifically created for a particular domain outperform more general models on the same domain. For this to be true, the advantage of modeling specific syntactic and semantic patterns in the data has to outweigh the data sparsity that comes with a smaller data set.

We tested this hypothesis on the Corpus Spoken Dutch (Corpus Gesproken Nederlands; CGN)[11,12]. An 8 million word corpus of contemporary Dutch spoken in Flanders and Netherlands. This data set is made up of 15 components, each with its own 'genre'. The different components range from spontaneous conversations over dinner, over political discussion to broadcast news (Table 1).

For all experiments discussed in this paper, we randomly selected 80% of the data in every component for training, 10% for development testing and 10% for evaluation. We created a vocabulary with 44368 words, which contains all unique words that occur more than once in the training data. All words in the data that are not in the vocabulary were replaced by an out-of-vocabulary(OOV) token. Using all training data, we created an interpolated trigram model:

$$\hat{p}(w_i|w_{i-2}w_{i-1}) = \lambda_1 p(w_i|w_{i-2}w_{i-1}) + \lambda_2 p(w_i|w_{i-1}) + \lambda_3 p(w_i), \qquad (1)$$

where w_i is the i-th word in a sentence. Each of the components of this model was estimated using maximum likelihood. The interpolation weights λ_1, λ_2 and λ_3 were estimated using Expectation-Maximization on the development test set. In the same way, we created a specific model for 14 components of the CGN corpus[1] We tested every component model on the corresponding component specific evaluation set and the general model on each of the evaluation sets in turn. Table 1 shows perplexity results per component of the evaluation set. Perplexity is calculated according to:

$$PP(w_1 w_2 \dots w_t) = 2^{-\frac{1}{t} \log P(w_1 w_2 \dots w_t)}, \tag{2}$$

where $w_1 w_2 \dots w_t$ is the data in the evaluation set.

Table 1. Comparison in terms of perplexity of specific models and general models per component of the CGN data set

component	specific models	general models		
		trigram	topic	cluster
a (spontaneous face-to-face)	220.85	222.83	226.31	221.37
b (interviews)	196.95	214.39	213.20	212.00
c (spontaneous telephone)	186.56	189.19	193.57	188.22
d (spontaneous telephone)	189.90	194.45	192.14	193.28
e (business negotiations)	110.27	153.35	154.11	152.15
f (interviews broadcast)	274.59	284.32	283.47	281.73
g (debates)	287.19	372.34	366.95	349.62
h (lessons)	298.08	315.01	314.07	313.44
i (live commentaries)	275.65	425.24	402.42	369.33
j (news reports)	345.37	369.28	367.52	361.41
k (news)	366.14	573.01	560.47	563.87
l (interviews broadcast)	425.70	440.11	435.14	434.42
n (lectures)	398.80	444.08	436.57	434.55
o (read book texts)	573.59	705.90	682.52	695.76
overall	-	277.67	274.08	272.50

The second column of Table 1 shows that the perplexity of the specific component models is lower than the perplexity of the general trigram model for every component. For some components the difference is relatively small. This is especially the case for the components that contain spontaneous speech, e.g. component a. For these components we may expect little performance improvement from a model that is based on component specific submodels. For others, most notably the live commentaries, the news, and the read texts the difference is considerable. Based on these results, we may hypothesize that the general model looses performance on those components that contain a diverse mix of topics and and a more diverse vocabulary. Put differently, in those cases it fails to model the exceptions to the rule.

[1] We left out component m, as it it too small to create reliable models with.

4 Bayesian Network Models

The results in the previous section suggest that we can improve upon the general trigram model if we can exploit the information in the component models. Previous work [8,4,5] suggests that we can do so by treating the component as an explicit variable in the language model. As the value of this variable is unknown during use, it will be inferred from the previous words. Dynamic Bayesian networks provide a convenient formalism to express such models in. Before presenting our models we will briefly discuss dynamic Bayesian networks.

4.1 Dynamic Bayesian Networks

Bayesian networks are a method for reasoning with uncertainty that combine probability theory and graph theory [13]. Bayesian networks are directed acyclic graphs of which the nodes are random variables and the arcs indicate conditional independence of the variables, i.e. the absence of an arc between two variables signifies that those variables do not directly depend upon each other. Thus a Bayesian network is a factored representation of a joint probability distribution over all variables given by:

$$\prod_{V} P(V|Parents(V)), \tag{3}$$

where V is a random variable. Dynamic Bayesian networks (DBNs) model processes that evolve over time [14]. They consist of a Bayesian network that defines the relations between variables at a particular time step and a set of arcs that specify how the variables depend on previous time steps. Several efficient inference algorithms have been developed for DBNs [15]. Training is usually done with an instance of the Expectation-Maximization (EM) algorithm.

4.2 A Topic-Based Model

We implemented a socalled topic based Bayesian network model, in which the CGN components are the topics. Figure 1c shows the first three time slices of the corresponding network. As in the trigram, every word w_i depends on the previous two words. As before we use an interpolated model that combines trigram, bigram and unigram statistics. The interpolation weights were set using a held out development test set. In addition, each of these components depends on the topic variable T, that takes as its values the 15 components of the CGN. E signals the end of a sentence and EOS the end of an utterance. The latter is included to ensure that the model is a proper language model in the sense that the sum of the probabilities of all possible word sequences is 1. N is a helper variable that counts the words in a sentence. It is used to make sure that a word is not conditioned on words in a previous sentence.

We first trained this model on the complete data set with a uniform prior for the component variable. As a result every component model corresponds to the general trigram model. We then trained the model on the complete training set, but this time the component variable was set to the correct component value for every document. The

models resulting from this run were interpolated with the models of the previous run to ensure that every model covered the same complete vocabulary. This way, the OOV rate is same for all models.

As before we evaluated the model on every component evaluation set in turn. In this case the component variable was treated as a hidden variable, i.e. the model does not know the component it is dealing with beforehand, but has to infer this value. Table 1 shows the results. With the exception of components a, c and e the model performs slightly better than the general trigram model.

4.3 A Cluster-Based Model

The topic model is based directly on the components of the CGN. However, these components may not form coherent subsets of the data, which makes it difficult for the model to infer which type of data it is dealing with. As discussed above, it is likely that the components that contain spontaneous speech are rather similar, while the data in other components such as news and live commentaries is more diverse. To take this into account, we reclustered the training data. For this we represented every document as a weight vector. The length of the vector corresponds to the number of different semantically salient lemma types in the vocabulary that were found by removing all function words and common content words from the vocabulary. We used lemmas rather than words, as inflections are not important for topicality. The entries of the vectors are weights that indicate the relation between the document and the lemmas. We used term frequency-inverse document frequency (TF-IDF) weights as widely used in information retrieval [16]:

$$\text{weight}(i,j) = \begin{cases} (1+\log(\text{tf}_{ij}))\log(\frac{N}{\text{df}_i}) & \text{tf}_{ij} > 0, \\ 0 & \text{tf}_{ij} = 0, \end{cases} \tag{4}$$

where N is the number of documents and the term frequency tf_{ij} counts the number of times lemma i occurs in document j. High frequency lemmas are thought to be characteristic for the document. Higher counts reflect more saliency of a word for a document but the scale is not linear. Observing a word twice as much does not mean that it is twice as important. Therefore, term frequencies are logarithmically scaled. This quantity is weighted by the inverse document frequency df_i which gives the number of different documents lemma i occurs in. The idea is that lemmas that occur in many documents are semantically less discriminating. This component is also logarithmically weighted.

Together the word vectors span a high-dimensional space in which each dimension corresponds to a lemma. To measure semantic similarity between documents we applied a cosine metric. Clustering was done using k-means clustering, with $k = 16$. Every document was annotated with its cluster number, after which we trained a topic based model as descibed above.

Table 1 shows that the model performs better than the general trigram model on all components. With the exceptions of components d, k and o it also outperforms the topic-based model based on all components. The improvement is highest for the live commentaries and debates, components on which the trigram model did considerably

worse than the baseline set by the component specific models. For the other components the improvements are smaller, but as mentioned above, it should be noted that for several components there is not that much to gain compared to the component specific models.

It is likely that better results can be obtained by optimizing the number of clusters in the model to ensure that each cluster does form a coherent, sufficiently large, data set. However, the computational demands of the DBN based models quickly increase as a function of the number of clusters as it sums over all cluster models to arrive at a probability for a word in every time step. Furthermore, this sum of component probabilities implies that the perplexity of the combined model will typically be higher that that of an appropriate specific model. Therefore, we propose a new type of model that selects a single specific model in every time step in the next section.

5 The Dynamic Bayesian Network Tree Model

A Dynamic Bayesian Network Tree (DBNT) combines a set of dynamic Bayesian network language models and a decision tree. The result is a language model that can change its parameters and structure depending on the context. For every word in a sentence, the decision tree is used to select one of the component language models based on available context information, such as the word history. Each of the nodes of the tree has an associated DBN language model. Figure 1a shows an example of a one-level DBNT. The components of this model can for example be the component specific models of section 3. More specific models are put in lower branches of the tree, to ensure that these models generalize sufficiently, they are interpolated with the more general models along the path from the root to their leaf. For example, every component model in Figure 1a is interpolated with the general model R in the root. While model $T1$ in the two level DBNT shown in Figure 1b is interpolated with the root model R and the intermediate model $C1$.

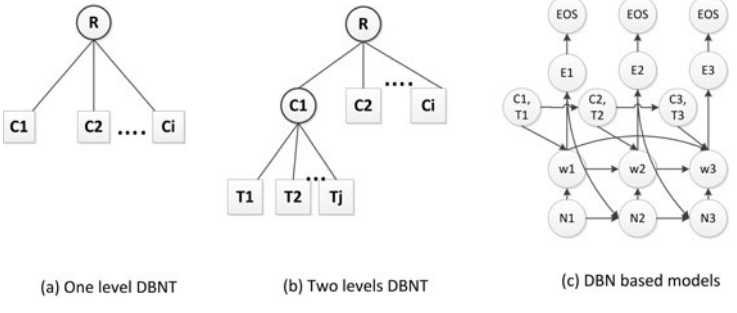

Fig. 1. Two simple dynamic Bayesian network trees and a DBN based model. (a) One level DBNT. (b) Two level DBNT, with one nested clustering procedure. (c) The DBN based models, W_i represents the word at the i-th position, E signals whether current word is the end of a sentence, EOS signals whether current word is the end of a sequence, N gives word position in a sentence, and T, C are the topic and cluster variable, respectively.

5.1 Learning DBNT Language Models

The learning procedure of the DBNT models is a two stage process. In the first step the decision tree is learned and used to split the training data into smaller subsets, where subsets on the same level of the tree do not overlap and node lower in the tree subdivide the data of their parent into smaller sets. For this any decision tree induction algorithm can be used. Alternatively, a hierarchical clustering procedure can be used. In the second stage, a component DBN model is trained for every subset in the data using standard DBN algorithms. The structure of these models is predetermined. To learn the interpolation weights for the models along a path in the model the EM algorithm [17] or re-estimation method [8] can be used.

5.2 Selection of Component Language Models in Prediction

The idea behind DBNT language models is to dynamically apply different component language models in different contexts. Any function that selects an appropriate model based on contextual information can be used. In particular, one can use the probability assigned by every component model to (part of) the word history to select the model that will predict the next word in the sentence.

6 Conclusion

We showed that specific language models outperform general language models if the specific model fits the data. We might improve our general models, if we can exploit the knowledge of the specific language models in a general model. Based on ideas in [8,4,5,16], we created and tested two DBN based models: a general topic model and a general cluster model, that make use of the knowledge of specific language models. The results showed that both models perform better than general trigram models. However, the computational demands of the DBN based models quickly increase as a function of the number of clusters or topics. Therefore, we proposed a new type of model that selects a single specific model in every time step. Our future work is to implement and test this new model.

References

1. Chelba, C., Jelinek, F.: Exploiting syntactic structure for language modeling. In: Proceedings of the 17th International Conference on Computational Linguistics, vol. 1, pp. 225–231. ACL, Stroudsburg (1998)
2. Rosenfeld, R.: A maximum entropy approach to adaptive statistical language modelling. Computer Speech and Language 10, 187–228 (1996)
3. Schwenk, H.: Efficient training of large neural networks for language modeling. In: Proceedings IEEE International Joint Conference on Neural Networks, 2004, vol. 4, pp. 3059–3064 (2004)
4. Wiggers, P., Rothkrantz, L.: Combining topic information and structure information in a dynamic language model. In: Matoušek, V., Mautner, P. (eds.) TSD 2009. LNCS, vol. 5729, pp. 218–225. Springer, Heidelberg (2009)

5. Shi, Y., Wiggers, P., Jonker, C.: Language modelling with dynamic bayesian networks using conversation types and part of speech information. In: The 22nd Benelux Conference on Artificial Intelligence, BNAIC (2010)
6. Clarkson, P., Robinson, A.J.: Language model adaptation using mixtures and an exponentially decaying cache. In: Proc. ICASSP 1997, Munich, Germany, pp. 799–802 (1997)
7. Kneser, R., Steinbiss, V.: On the dynamic adaptation of stochastic language models. In: Proceedings of ICASSP 1993, Minnapolis(USA), vol. II, pp. 586–589 (1993)
8. Iyer, R., Ostendorf, M., Rohlicek, J.R.: Language modeling with sentence-level mixtures. In: HLT 1994: Proceedings of the Workshop on Human Language Technology, pp. 82–87. Association for Computational Linguistics, Morristown (1994)
9. Bahl, L.R., Brown, P.F., de Souza, P.V., Mercer, R.L.: A tree-based statistical language model for natural language speech recognition. IEEE Transactions on Acoustics, Speech and Signal Processing 37, 1001–1008 (1989)
10. Xu, P., Jelinek, F.: Random forests in language modeling. In: Proceedings of EMNLP, pp. 325–332 (2004)
11. Hoekstra, H., Moortgat, M., Schuurman, I., van der Wouden, T.: Syntactic annotation for the spoken dutch corpus project (cgn). In: Computational Linguistics in the Netherlands 2000, pp. 73–87 (2001)
12. Oostdijk, N., Goedertier, W., Eynde, F.V., Boves, L., Pierre Martens, J., Moortgat, M., Baayen, H.: Experiences from the spoken dutch corpus project. In: Proceedings of the Third International Conference on Language Resources and Evaluation, pp. 340–347 (2002)
13. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco (1988)
14. Dean, T., Kanazawa, K.: A model for reasoning about persistence and causation. Computational Intelligence 5, 142–150 (1989)
15. Murphy, K.P.: Dynamic Bayesian Networks: Representation, Inference and Learning. PhD thesis, University of California, Berkeley (2002)
16. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley, Reading (1999)
17. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the royal statistical society, series B 39, 1–38 (1977)