Topic-Based Language Modeling with Dynamic Bayesian Networks

Pascal Wiggers, Leon J.M. Rothkrantz

Man–Machine Interaction Group
Delft University of Technology, Delft, The Netherlands
p.wiggers@tudelft.nl, l.j.m.rothkrantz@tudelft.nl

Abstract

Although n-gram models are still the de facto standard in language modeling for speech recognition, more sophisticated models achieve better accuracy by taking additional information, such as syntactic rules, semantic relations or domain knowledge into account. Unfortunately, most of the effort in developing such models goes into the implementation of handcrafted inference routines. A generic mechanism to introduce background knowledge into a language model is lacking. We propose using dynamic Bayesian networks. Dynamic Bayesian networks are a generalization of the n-gram models and HMMs traditionally used in language modeling and speech recognition. Whereas those models use a single random variable to represent state, Bayesian networks can have any number of variables. As such they are particularly well-suited for the construction of models that take additional information into account. This paper discusses language modeling with Bayesian networks. Examples of Bayesian network implementations of well-known language models are given and a novel topic-based language model is presented.
Index Terms: language modeling, dynamic Bayesian networks.

1. Introduction

The task of a stochastic language model is to assign a probability to every sentence in a language. Using the chain rule of probability theory this is often reformulated as the task of predicting the next word in a sentence based on the previous words. The most common language models, n-grams, are based directly on this idea and predict the next word using a limited history of typically two or three preceding words. Despite their simplicity these models are surprisingly powerful because their locality makes them quite robust while at the same time they capture many of the local syntactic and semantic constraints in language. Nevertheless, a number of more sophisticated models, all of which use additional knowledge of some sort, have been introduced that do perform better than n-grams. The additional knowledge ranges from syntax [1] over semantic similarities between words [2] to knowledge of the application domain [3]. Although the information in different models is at least to some extend complementary, there have been few attempts to combine them. Many potentially useful sources of information for language modeling have not yet been explored, e.g. the type of text or conversation. The structure and vocabulary of read text differs from that of a spontaneous conversation. This is typically dealt with by training different models for different types of data. Indeed there is little else one can do with the flat-structured n-grams.

The main reason that the area of knowledge-rich language models is largely unexplored seems to be the lack of a unifying framework that allows one to do so. Work in this domain typically proceeds along the following lines: a feature that might be useful in language modeling is identified, a probabilistic model that includes this feature is formulated and then most of the effort goes into deriving and implementing the algorithms needed to train the model and do inference with it. Although all of these models can be classified as probabilistic language models their already complex algorithms are highly specialized and therefore difficult to integrate.

In this paper we argue that dynamic Bayesian networks, which have proven themselves in artificial intelligence and recently in speech recognition [4], can provide us with a framework that allows for rapid development and validation of knowledge-rich language models. As an example a topic-based language model is presented that captures contextual coherence.

2. Related work

A wide variety of language models have been developed that use other knowledge than the preceding words. Trigger-based models [2] use the coherence of a text: words have a tendency to occur together with semantically related words. This is implemented in the model as words triggering related words. Multispan language models use information retrieval techniques that calculate semantic similarities between vectors associated with words and histories [5]. Another group of language models exploit coherence by modeling the topic of conversation [6, 7, 8]. Structured language models (SLMs) introduced by [1] use syntactic information to model dependencies between words by formulating probabilistic parsers as generative language models [9, 10]. Finally, some models use knowledge of an application domain to get more accurate word probabilities. In [3] travel frequencies between railway stations are used to better predict station names in a train table dialog system.

3. Dynamic Bayesian networks

Bayesian networks originate in artificial intelligence as a method for reasoning with uncertainty based on the formal rules of probability theory [12]. 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:

$$\prod_V P(V|Parents(V)) \tag{1}$$

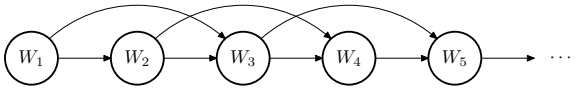

Figure 1: *A trigram Bayesian network. Time progresses in vertical direction. Every time slice contains a random variable* W *that takes the words in the vocabulary as its states.*

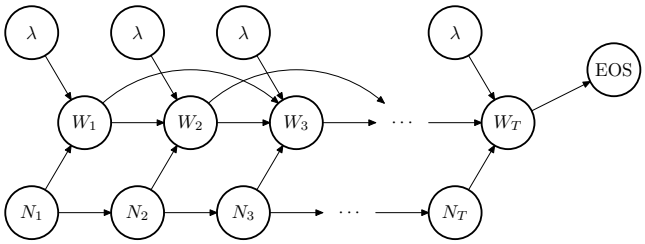

Figure 2: *An interpolated trigram. the hidden variable λ is used to decide on which other parents the word depends.*

Dynamic Bayesian networks (DBNs) model processes that evolve over time. They consist of slices that define the relations between variables at a particular time, implemented as a Bayesian network and a set of arcs that specify how a slice depends on previous time slices. DBNs can be seen as a generalization of both n-gram models and hidden Markov Models. The difference being that HMMs use only one variable to represent the state of the model, whereas a DBN can use any number of variables. Several efficient inference algorithms have been developed for DBNs [11]. Training is typically done with an instance of the Expectation-Maximization (EM) algorithm.

Dynamic Bayesian Networks have already found their way into speech recognition systems. [4, 13] have used DBNs to include articulatory information in acoustic models, and to model relations between observation features. Several types of DBNs have been used to construct multi-modal recognizers that integrate speech recognition and automatic lip-reading [14]. They have also been applied to dialog act classification [15].

We implemented tools for the construction and training of Bayesian networks as well as for recognition and inference (forward, forward-backward, Viterbi algorithms and slice-by-slice prediction). Unlike existing toolkits, these tools are targeted specifically at language processing, they can deal efficiently with variables that can take a large number of states, e.g. the number of words in the vocabulary and distributions that are typically sparse. Other features include parameter tying, mixtures of distributions and methods for pruning and fast approximate inference.

4. Language modeling with DBNs

Although DBNs have been used in speech recognition their use in language modeling and in natural language processing in general remains rather limited, probably because other techniques such as grammars and weighted finite state transducers are more generally known in those areas. Like HMMs, finite state transducers are a special kind of Bayesian network. This section will show how several well-known language models can be formulated as DBNs.

4.1. N-grams

Fig. 1 shows the basic DBN representation of a trigram. Every time slice contains a single word variable that is connected to its two predecessors. However, there are a number of issues we have to deal with. For n-grams one will typically use dummy states to indicate the start and the end of a sentence, where the start symbol is repeated several times to allow the use of trigrams for the first two words [19]. The same can be done in DBNs, but there are better ways. To deal with the first two words a variable N is introduced that counts the words. The value of the counter is used to decide which word distribution will be used. For the first slice a unigram is used, for the second slice a bigram and from the third slice the standard trigram is used.

Another binary variable EOS is added that signals the end of a sentence. This variable makes sure that the model is a proper language model, in the sense that the probabilities it assigns to all sentence in the language will sum to one rather than the probabilities of all sentences of a particular length. The n-gram counterpart of this is a sink state that transitions with probability one to itself. The counter can be conditioned on the end-of-sentence variable to let it restart for every sentence, so the first words of a sentence do not depend on the last words of the previous sentence. To keep the figures clear, we will not draw counter and end-of-sentence nodes in the remaining figures in this paper, but all of the models that will be discussed do include those variables.

The last variable shown in Fig. 2, λ, implements smoothing. Depending on its value the current word does either not depend on it predecessors at all, only on the previous word, or on the previous two words. As λ is a hidden node, the result is a mixture of distributions implementing deleted interpolation:

$$P_\lambda(W_t|W_{t-1},W_{t-2}) = \lambda_1 P(W_t) + \lambda_2 P(W_t|W_{t-1}) + \lambda_3 P(W_t|W_{t-1},W_{t-2}) \quad (2)$$

The values of λ can be found by training on a held-out data set.

4.2. Class-based language models

Class-based language models group words into classes in order to generalize to unseen words and to obtain more reliable statistics. N-gram probabilities over classes are used to predict the class of a word which is then used to predict the word itself. [16] introduced class-based models with part-of-speech (POS) classes. Fig. 3 on the following page shows the DBN counterpart of a POS-model, the POS-tags are added to the model as hidden variables (P) that are connected in time. Compared to n-grams, class-based models achieve better generalization and a smaller parameter set at the cost of less fine-grained modeling. To get the best of both worlds class-based models and n-grams are often combined through interpolation. This is particularly easy to accomplish in a DBN as is shown in Fig. 4 on the next page which combines a POS-model with an interpolated trigram. There is no need to derive or implement special algorithms for this model, the general purpose Bayesian network algorithms are all that is needed.

We can further improve the class-based model by adding first and second order relations on the class level as is depicted in Fig. 4 on the following page. Additionally, we can let the class nodes depend on previous words or the words on previous tags.

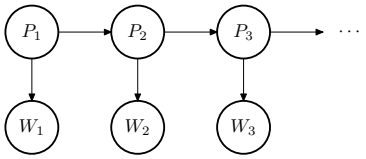

Figure 3: *A class-based* POS*-model. The* P*-variables take part-of-speech tags as their states and are interconnected through time, the word variables only depend on the* POS*-classes.*

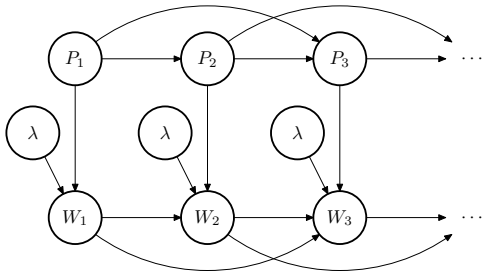

Figure 4: *A combined* POS*-trigram model, words depend on* POS*-classes and on previous words.*

5. A topic-based language model

Words and sentences do not occur in isolation but are part of a coherent discourse. Within a particular conversation some words are more likely to occur than others. One can introduce this idea into a Bayesian network language model in the shape of a variable that takes topics as its states. Every topic is a distribution over the vocabulary. Typically the topic of a conversation is not known beforehand, therefore the topic variable is hidden. As a consequence the model will use a mixture of several topics. The relative likelihoods of those topics may change gradually over time.

Initially all topics will be equally likely, when a new word is observed, it is propagated back through the network as evidence, changing the topic distribution. Those topics that predicted it with a high probability will be supported by the word while the influence of other topics will be moderated. As the topic nodes are connected in time by deterministic links, the updated topic distribution is used to predict the next word. One can think of the topic nodes as a means to capture long-distance dependencies: the exact words are not remembered, but the topic mixture provides a summary of the history.

Topics are especially useful for the prediction of content words. For this reason the model in Fig. 5 differentiates between content words and function words. For every time slice the model first predicts the POS-tag of the word based on previous POS-tags and previous words. It then sets the binary variable F if the word is a function word. Like λ in Fig. 2 this is a switching parent of the word variable, if it is set, the word depends only on its POS-tag and the two preceding words, otherwise the word must be a content word and depends also on the topic. Distributions are combined through deleted interpolation.

6. Experiments

We train all models discussed here on part of the Spoken Dutch Corpus (CGN [17]) of spontaneous speech as spoken by adults in the Netherlands and in Flanders. We use subset *comp-f* of the corpus, which contains interviews and discussions broadcasted on radio and television. The set contains a total of 790.269 words, 80% of which we use for training, 10% for development testing and tuning and the remaining 10% for evaluation. All words that occur only once in the training set are treated as out-of-vocabulary words, resulting in a vocabulary size of 17833 words and 257 POS-tags. The POS-tags include attributes such as number, degree and tense.

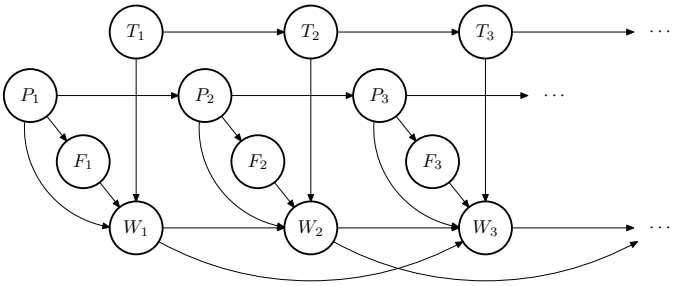

Figure 5: *A topic-based language model.* F *indicates if a word is a function word or a content word. Content words also depend on the topic* T*.*

The most important question for the topic model is how one obtains suitable topics. Ideally, a labeled data set is available, otherwise, unsupervised training of the model using for example the EM-algorithm is possible, at least in theory. Since we found that the model is rather sensitive to its initial distributions we choose a solution somewhere between those extremes, using clusters of recordings as our topics.

We represent the documents in the training set as vectors in lemma space. First all function words and common content words are removed and all other words are replaced by their lemma. The elements of the vectors are *tf-idf* weights as widely used in information retrieval [20]:

$$\text{weight}(i,j) = \begin{cases} (1+\log(\text{tf}_{ij}))\log(\frac{N}{\text{df}_i}) & \text{tf}_{ij} > 0 \\ 0 & \text{tf}_{ij} = 0 \end{cases} \quad (3)$$

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. This quantity is weighted by the inverse document frequency df_i which gives the number of documents lemma i occurs in, as terms that show up in many documents are semantically less specific. Both components are logarithmically scaled to reduce the effect of high counts. We cluster the resulting document vectors using agglomerative clustering with a cosine-measure.

From every cluster we select only half of the documents that are most similar and use those to initialize the distributions in the Bayesian network model. The distributions are interpolated with the global distribution over content words to make sure that all words have a non-zero probability for all topics. If this is not the case, a word that has a zero probability for a topic will set the probability of that topic to zero if it is observed.

Up to this point we treated every document as if it belonged to exactly one topic, but this in not necessarily the case. Therefore we run several iterations of the EM algorithm on the whole training set to update the topic distributions. This is a form of soft clustering,

Table 1: *Perplexity results*

language model	perplexity
interpolated bigram	296.49
interpolated trigram	280.76
interpolated trigram with POS-tags	245.39
topic-based model with 64 topics	242.92

all documents are assigned to all topics weighted by the likelihood of the topics. Monitoring the perplexity of a development test set we found that the model quickly overtrains. Therefore we introduce a damping factor [11]:

$$P_t(W|T) = (1-\delta)\tilde{P}_t(W|T) + \delta P_{t-1}(W|T) \quad 0 \leq \delta \leq 1 \quad (4)$$

Depending on the weight of d distributions are only partially updated in every iteration. If $\delta = 0$ this amounts to normal updating. $\delta = 1$ would result in no updating at all. In addition, the damping factor avoids zero probabilities for words that do not occur in the training data.

7. Results

Table 1 gives the perplexity of various models on our evaluation set. It shows that the model that uses both word trigrams and POS-classes has a much lower perplexity than a simple interpolated trigram. This result corresponds to what one would expect from literature. The perplexity of the topic model is slightly better than that of the POS-model. Earlier experiments with the prediction of content words only have shown that additional topics give better results, so we expect the same to hold for this model.

8. Conclusions

In this paper we have proposed the use of dynamic Bayesian networks for language modeling and shown how well-known language models can be implemented as Bayesian networks. The objective of our work is to explore the usefulness of higher knowledge sources such as syntax and semantics, domain knowledge and user characteristics in language modeling and speech recognition. We believe that Bayesian networks provide an ideal tool for such a task. They make it possible to define new models in a declarative way by simply specifying the variables together with the network structure without the need for special-purpose inference routines. Using a single framework for experimentation with different models has the additional advantage that it is easier to compare models.

We have presented a novel topic-based model that improved in terms of perplexity compared to a standard interpolated trigram model and class-based models. We are currently experimenting with models that have a larger number of topics and are using those models to rescore lattices output by a speech recognizer.

9. References

[1] C. Chelba, F. Jelinek, Exploiting Syntactic Structure for Language Modeling. In Proc. of the 36th Annual Meeting of the ACL, August 1998.

[2] R. Rosenfeld, A Maximum Entropy Approach to Adaptive Statistical Language Modeling. Computer, Speech and Language 10, 187–228, 19

[3] P. Wiggers, L. J. M. Rothkrantz, Using confidence measures and domain knowledge to improve speech recognition, Proceedings of Eurospeech 2003, Geneva Switserland, September 2003.

[4] G. Zweig, Speech Recognition with Dynamic Bayesian Networks, Ph.D. Thesis, Computer Science Division, University of California at Berkeley, 1998.

[5] J. R. Bellegarda, A Multispan Language Modeling Framework for Large Vocabulary Speech Recognition, IEEE Transactions on Speech and Audio Processing, Vol. 6, No. 5, September 1998

[6] D. Gildea, T. Hoffman, Topic-Based Language Models using EM, Eurospeech 1999, Budapest

[7] M. Mahajan, D. Beeferman, X. Huang, Improved Topic-Dependent Language Modeling using Information Retrieval Techniques, in proc. of ICASSP 1999

[8] R. Zhang, A.I. Rudnicky, Improve Latent Semantic Analysis based Language Model by Integrating Multiple Level Knowledge, in proc. of ICSLP 2002, Denver, Colorado

[9] B. Roark, Probabilistic Top-Down Parsing and Language Modeling, Computational Linguistics, Volume 27, Nr. 2

[10] E. Charniak, Immediate-Head Parsing for Language Models, Meeting of the Association for Computational Linguistics", pages 116–123, 2001

[11] K. Murphy, Dynamic Bayesian Networks: Representation, Inference and Learning, Ph.D. Thesis, University of California, Berkeley, 2002.

[12] J. Pearl, Probabilistic Reasoning in Intelligent Systems - Networks of Plausible Inference, 1988, Morgan Kaufmann Publishers, Inc.

[13] J. Bilmes, Natural Statistical Models for Automatic Speech Recognition, Ph.D. Thesis, Dept. of EECS, CS Devision, U.C. Berkeley 1999.

[14] A. V. Nefian, L. Liang, X. Pi, X. Liu, K. Murphy, Dynamic Bayesian Networks for Audio-Visual Speech Recognition, EURASIP Journal on Applied Signal Processing 2002:11, 1–15.

[15] S. Keizer, Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks, Ph.D. Thesis Twenty University, 2003. 96.

[16] F. Jelinek, Self-organized language modeling for speech recognition. In Alex Waibel and Kai-Fu Lee, editors, Readings in Speech Recognition, 1990, Morgan Kaufman Publishers, Inc. San Mateo, Ca.

[17] I. Schuurman, M. Schouppe, H. Hoekstra, T. van der Wouden. CGN, an Annotated Corpus of Spoken Dutch. In Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03). 14 April, 2003. Budapest, Hungary.

[18] J. T. Goodman, A Bit of Progress in Language Modeling, Microsoft Technical Report MSR-TR-2001-72

[19] C. D. Manning, H. Schüze, Foundations of Statistical Natural Language Processing, 1999, The MIT Press, Cambridge, Massachusetts, ISBN 0-262-13360-1.

[20] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley, 1999, ISBN 0-201-39829-X.