Modeling Text using State Space Models

Rasmus E. Madsen
Department of Mathematical Modelling
Technical University of Denmark
Lyngby, DK-2800
rem@imm.dtu.dk

Abstract

Generic "bag-of-words" text categorization methods are only based on the information contained in word count histograms. These methods does therefore not capture the information contained in the order in which the words appear in a document. We here consider models that is acting on both parts of information at the same time, that is the information about what words appear and in what order they appear. State-space models has the ability to capture information from the order in which the words appear, and combine it with the word appearance probabilities. The state-space models should therefore conceptually super-seed the bag-of-words/vector-space models, in ability to model documents correctly. In the following we experiment with two state space model approaches, for making categorization better.

1 Introduction

The document vector space model (Salton et al., 1975), the bag-of-words model and its varieties are effective document simplifications, that make machine learning approaches to text modeling and classification simple. The two document representations has resulted in the development of many different algorithms (Deerwester et al., 1990; Hofmann, 1999; Sebastiani, 2002; Blei et al., 2003) who are effective for text classification. The models that use these representations however loose a big fraction of the information contained in the documents, by considering only the counts of how many times words appear in a given document. The other part of information contained in documents is the information about the order in which the words appears. Though the major part of document information is contained in the knowledge about which word occur, some important information might be captured from the word appearance order, that could make document classification accuracy better. One way to interpret the word order information is as being the authors style of writing, i.e. a fingerprint that tells how the author constructs his sentences. Some authors might construct grammatically different sentences from others. This grammatical difference might not be captured when only word histograms are considered.

It is easy to extract the word appearance information from a document and form it into some meaningful representation that can be used for machine learning, i.e. vectors or histograms. The word order information is however harder to extract to some simple low dimensional representation, which is easily portable to a machine learning algorithms. We therefore consider state space models, which can model sequences of data, instead of the counts.

State space models have previously been used for language modeling, e.g. in context of predicting the next word in handwritten text recognition systems (Zimmermann & Bunke, 2004), and has been successful so. It is therefore further likely that the state-space model can capture valuable information that can be used for text classification.

We here consider two different state-space based approaches, both based on an underlying Markov state space model. Both approaches suggest a method to overcome the dimensionality problem of text, which otherwise makes the state-space models extremely slow. The first approach suggested here generates a new lower dimensional vocabulary, which is later used in a hidden Markov model. Using the second approach, the state part of a hidden Markov model is used in conjunction with LSI emission probabilities.

2 Discrete Markov Process

The discrete Markov process (Rabiner & Juang, 1986) is a state space model that can model and generate sequences of discrete symbols. The discrete Markov process considers a system with K states s_k, where for each time-step t the process changes state, where the new state can be the same as the previous state. The actual state at time t is denoted q_t, which can be interpreted as the discrete symbol generated at time t. The probability of changing state to a new state $q_{t+1} = s_j$ from the state $q_t = s_i$ is determined by the transition probabilities $a_{s_i,s_j} = P(q_{t+1} = s_j | q_t = s_i)$, where $\sum_{j=1}^{K} a_{s_i,s_j} = 1$ and $a_{s_i,s_j} \geq 0$. The transition probabilities are therefore only dependent on the current state of the process and not the time t or previous states $q_{t-t'}$. A tutorial on Markov processes can be found in (Rabiner, 1989).

The discrete Markov process assembles an urn scheme where there is one urn for each state in the Markov process. When the time-step changes, a new urn is selected according to the transition probabilities, and a ball from that urn is drawn, and the color noted, whereafter the ball is returned into the urn. Each urn contains only balls with the same color.

The urn model analogy to text modeling is straight forward. Instead of balls, each urn is filled with words, again only one kind of words for each urn. When a document is generated, we start out with one particular urn and draw a word from it, and continue to another urn and draw a new word here. The transition probabilities determines what words are likely to appear after the present one. The Markov process will therefore be able to model parts of the semantics of the language model, by the transition probabilities. These semantics are not modeled at all when only word appearances alone are considered, i.e. using the vector space model representation.

Different kinds of documents might contain the same kinds of words, where the order of the appearances of the words, can change the meaning of the content. The word “train” could for example be used in documents about transportation or in documents about exercising in the gym. The words appearing around the word train, will therefore change the meaning of that particular word. The difference in meaning could therefore be captured by the Markov model. Another example of when the order of the words appearances can change the meaning of a sentence, is when the word “not” is used. Yet another example where transition probabilities could be useful is in spam email detection systems.

The drawback of the Markov model is that it models a huge probability space, since it considers all the possible word-pairs in the vocabulary. Since most document collection vocabularies considers about $100,000$ words, the model must consider $10,000,000,000$ possible transition probabilities. The transition probabilities would therefore consume to much memory for holding this data representation. By use of a grammar, many of the transition probabilities could be pruned away, while many word pairs can’t be used in grammatically correct sentences. Though the pruning approach would reduce the amount

of modeled probabilities tremendously, the amount of memory used to represent the model would still be very large. On top of the memory consumption, the model would also need a lot of data to be able to estimate all the transition probabilities. For existing document collections, the amount of data is far too limited to estimate the probabilities, making a huge need for smoothing, which usually result in bad modeling performance. Human brains can probably work with some variety of this modeling approach, while we can generalize many probabilities in the model by use of grammar and can therefore easily prune away the unlikely Markov model transition probabilities.

3 Hidden Markov Model

The hidden Markov model (HMM) (Rabiner & Juang, 1986; Rabiner, 1989) extends the discrete Markov process by adding an additional emission parameter to each state. The emission parameters controls the output that is generated from each state, i.e. a discrete symbol. For the HMM, each state therefore has the potential to generate all the symbols in the vocabulary of symbols. For each time-step t the HMM still changes state according to the transition probabilities a_{s_i,s_j}, but the the symbol is now generated using the emission probabilities $b_{s_j,v_m} = P(x_t = v_m|q_t = s_j)$, where x_t is the symbol generated at time t and v_m is symbol number m from the vocabulary of M symbols.

The HMM assembles an urn scheme that is similar to the Markov process urn scheme. A new urn is still selected at each time-step according to the transition probabilities, and a ball from the new urn is drawn. The color of the ball is noted whereafter the ball is returned into the urn. Using the HMM each urn now contains a distribution of balls that each has one of M different colors.

Since the number of symbols that can be generated M is independent of the number of states K, the memory consumption of the model can be reduced remarkably when the vocabulary is huge. If we consider a vocabulary of about $100,000$ words and use a state-space of 100 states, the amount of probabilities used to describe the model is approximately $10,000,000$, which is only $1/1000$ of the amount of memory needed to describe the Markov process for the same vocabulary.

The HMM parameters can be estimated using the expectation maximization (EM) algorithm (Dempster et al., 1977), resulting in an iterative update procedure that estimates the model parameters using the so called forward-backward approach (Rabiner, 1989),

$$\pi_{s_i} = \gamma_{1,s_i} \tag{1}$$

$$a_{s_i,s_j} = \frac{\sum_{t=1}^{T-1} \xi_{t,s_i,s_j}}{\sum_{t=1}^{T-1} \gamma_{t,s_i}} \tag{2}$$

$$b_{s_j,v_m} = \frac{\sum_{t=1}^{T} (O_t = v_m)\xi_{t,s_i,s_j}}{\sum_{t=1}^{T} \gamma_{t,s_j}} \tag{3}$$

where π_{s_i} is the probability of starting in state s_i and $\gamma_{t,s_i} = \sum_{j=1}^{K} \xi_{t,s_i,s_j}$ and $\xi_{t,i,j}$ is the probability of being in state s_i at time t and in state s_j at time $t+1$ and $(O_t = v_m)$ is 1 if the observation at time t is symbol v_m, and zero otherwise. The full description of the learning rules can be found in (Rabiner, 1989).

4 HMM with LSI GMM Vocabulary

The HMM approach reduces the memory needs, comparing it with a Markov process with a similar vocabulary size, making it possible to represent the model in a standard computer of today. The HMM model is however still fairly large and the EM updates that estimates the parameters are very demanding, computationally. In the approach described here, the vocabulary is therefore projected to a lower dimensional representation using latent semantic indexing (LSI) (Deerwester et al., 1990) with a SVD basis (Madsen et al., 2003) and gaussian mixture models (GMM). In Figure 1, the the lower dimensional representation of the vocabulary is shown.

The procedure of transforming the vocabulary to a lower dimensional representation, takes place in the following way:

1. Documents are cut into substrings of length L, with 50% overlap.
2. A common LSI representation for the substrings in all the documents is estimated using SVD.
3. The substrings are clustered using GMM on the first H dimensions of the LSI representation.
4. The clusters are now forming a new and much smaller vocabulary for the substrings, where each substring is transformed to an the index associated with the closest cluster.
5. A HMM is trained for each class of documents using the new vocabulary.
6. New documents are classified using the HMM forward backward classification algorithm.

The classification algorithm is using the forward-backward approach which is also used to estimate the parameters.

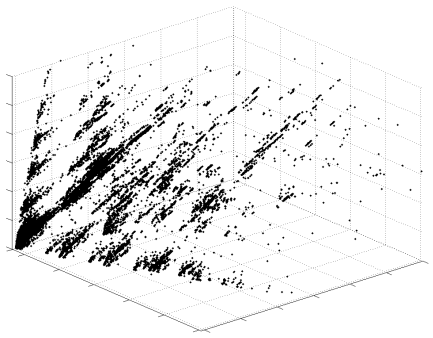


Figure 1: Space for the new vocabulary.

5 HMM with LSI emission probabilities

In the section about the hidden Markov model, we reject the model for use on text directly, while the high number of parameters for the model would make it converge slowly, due to size of the vocabulary. It is further undesirable to use the HMM directly on each single class while the classes wont be able to share the emission probabilities. It is desirable to share

the emission probabilities for all the classes while they can be thought of as latent topics, where there is a latent topic for each single state in the HMM. This idea is conceptually similar to the ideas from latent semantic indexing and it's varieties (Furnas et al., 1988; Deerwester et al., 1990; Hofmann, 1999; Kolenda et al., 2002; Blei et al., 2002; Blei et al., 2003).

The problems of shared latent topic emissions could be overcome by redefining the HMM to be a model with more state space transition models, but only one single state emission model. This model would be likely to inherit the slow convergence property of the normal HMM. We therefore reject the model here, knowing that it probably would be the best modeling approach to the problem.

The alternative to a redefined HMM, is to estimate the emission probabilities b_{s_j,v_m} using another algorithm and keeping them fixed when first estimated. Using this approach it would only be necessary to estimate the state transition parameters a_{s_i,s_j} and initial state probabilities π_{s_i} for each separate class. This estimation procedure would further not need to run in an iterative EM-loop where the one set of parameters are estimated based on an estimate of the other set of parameters. The transition parameters would therefore only need one or very few iterations to converge.

There are more alternative ways to determine a set of shared latent topic emission parameters. Three possible approaches are *independent component analysis* (ICA) (Bell & Sejnowski, 1995b; Bell & Sejnowski, 1995a; Molgedey & Schuster, 1994), *singular value decomposition* (SVD) (Madsen et al., 2003) and *non-negative matrix factorization* (NMF) (Lee & Seung, 1999; Lee & Seung, 2001). The latter approach has the advantage of estimating non-negative values when factorizing the data, which is valuable since these values reflect emission probabilities, i.e. they have to be positive and sum to zero. NMF has also shown valuable for text clustering previously (Xu et al., 2003). In practise however the NMF does not work well with the sparse structure of the text data, resulting in very few active words in each NMF latent topic. When only few words are active it is necessary to either use a lot of smoothing or use many latent topics. Neither of these fixes are likely to give us a good model or classifier, so we turn to SVD approach instead. The latent topics estimated by the SVD all have many active words. The problem of probabilities being negative is solved by simply setting negative values equal to zero, and then normalize the distribution.

The procedure of using the HMM state space model with LSI estimated emission probabilities, takes place in the following way:

1. A common set of HMM emission parameters are estimated using the LSI approach on the documents using the histogram representation.
2. A set of HMM state space parameters are estimated for each class using the word sequences for each document.
3. New documents (sequences of words) are classified using the HMM forward backward classification algorithm.

6 Experiments

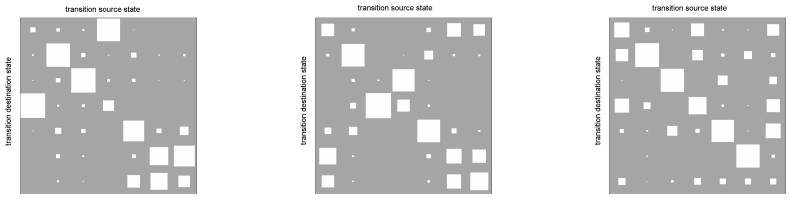
We are here working with the three corpora: email, WebKB and multimedia. The number of words in the three corpora are reduced by use of stemming and stop-word removal. Though we here only show results for the email-data, similar results where gained by use of the two other data-sets. The TF-IDF transformation has been applied to the document collections, when performing experiments using the HMM with LSI-GMM generated vocabulary. In the experiments where using the HMM with LSI emission probabilities, the TF-IDF transformation has not been applied. The reason is that the HMM works on sequences where

each unit in the sequence must be unity. A weighting scheme could be applied to the HMM, where the TF-IDF coefficients could be applied as weights. At first we are interested in investigating if the model works conceptually, and have therefore skipped the transformation step.

We start by training the HMM with LSI-GMM generated vocabulary (HLG) using the email-data. The largest class in the email data-set (spam) accounts for 0.55% of the emails. A naive classifier should therefore have a classification accuracy of about 0.55. The two models considered should therefore have generalization error below 0.45%.

We find that the HLG approach works best when a LSI subspace of 4 dimensions is used to form the new HLG generated vocabulary. A set of 100 gaussian's is used to cover the 4-dimensional space forming an new vocabulary of 100 words. The first three dimensions of the subspace are shown in Figure 1, where the structure of the data are much different from the structure found by the generic LSI representation Figure **??**. Each cluster that is put in the space in Figure 1 now represents a word in the new vocabulary.

Estimating the HMM for the new sequences, the transition probabilities for the three classes in the email-set, show us whether there is a sequential difference between the three classes that is captured by the model. In Figure 2, a graphical illustration of the transition probabilities is shown. Seven states is used in the HMM to best model the new sequences.



(a) Job (b) Conference (c) Spam

Figure 2: Graphical illustration of the transition probabilities for the HLG model. For the Job category (a), state 1 and 4 are paired and almost isolated from the other states making them a semantic chain for the category. Similarly state 5, 6 and 7 forms a state group that are likely to generate long sequences of words. The Conference category (b) has a similar group formation where the states 3 and 4 forms a group, and state 1,7 and 8 forms a group. The spam category (c) does not have the same strong group formation as the two other categories, but is instead less symmetric. There is however weak group formation between the states 1, 2 and four, and the states 3 and 5. The illustration of the transition probabilities reveal that there is a sequential pattern that is captured by the model.

The illustration in Figure 2 shows clearly that there is information in the order in which the words appear in a document, and that this information can be captured by the HMM.

There are more settings that determines the optimal HLG model, i.e. number of LSI dimensions, number of states in the HMM, the number of gaussian mixtures and the length of the substrings used to form the vocabulary. In Figure 3 the classification accuracy for the HLG model as function of the substring length is plotted, where the settings for the remaining parameters are close to optimal.

The HLG model has an accuracy that is lower than the accuracy of the LSI model, for all possible substring-length values.

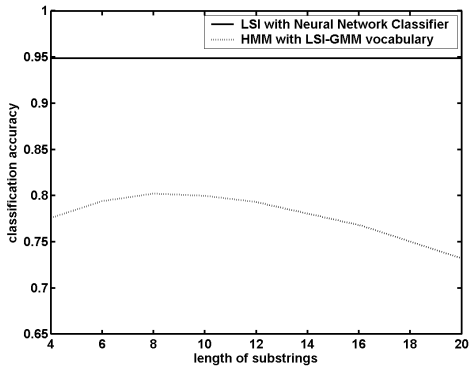
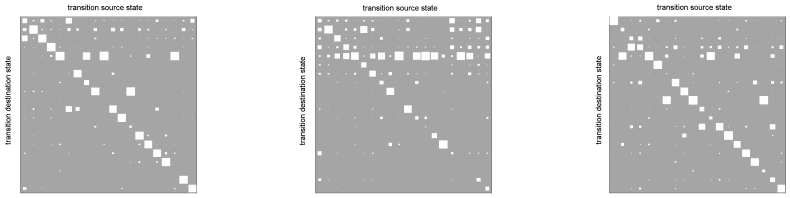


Figure 3: Classification accuracy using the HMM with LSI-GMM reduced vocabulary, as function of the substring length. The classification results are compared with a neural network classifier using a LSI subspace on TFIDF normalized data. The data used are email data where 20% of the data are used for training.

We next turn to the HMM model with LSI estimated emission probabilities. We again take a look at the transition probabilities for the three classes in the email-set, for discovering whether there is a sequential difference between the three classes that is captured by the model. In Figure 4 an illustration of the transition probabilities is shown. The Illustration shows transition probabilities for a model with 20 states, where the best model instead uses about 120 states. The smaller model is shown while it is easier to survey.



(a) Job (b) Conference (c) Spam

Figure 4: Graphical illustration of the transition probabilities for the LSI-HMM model. The group formations for the LSI-HMM state space is harder to discover than those for the HLG model. There is however small groupings like state 1 and 3 for the Job category (a).

The formation of state groups is not as obvious as it was for the HLG model. It is therefore less obvious whether or not, the LSI-HMM model has captured much sequential information about the documents.

In Figure 5 we show the learning curves for the LSI-HMM compared with the generic LSI model. The LSI model performs slightly better than the LSI-HMM model when used for classification. The performance of the two models follow along for the whole range of training set sizes.

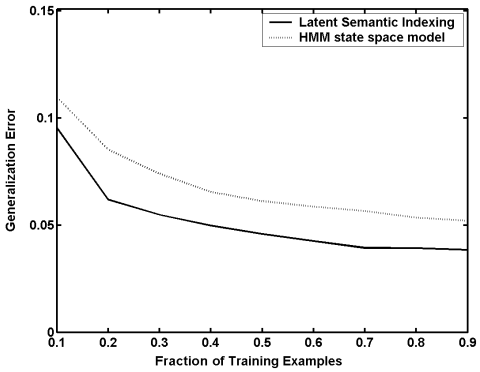


Figure 5: Learning curves for the LSI HMM model. The LSI-HMM model is slightly worse at classifying documents correctly than the generic LSI model.

7 Discussion

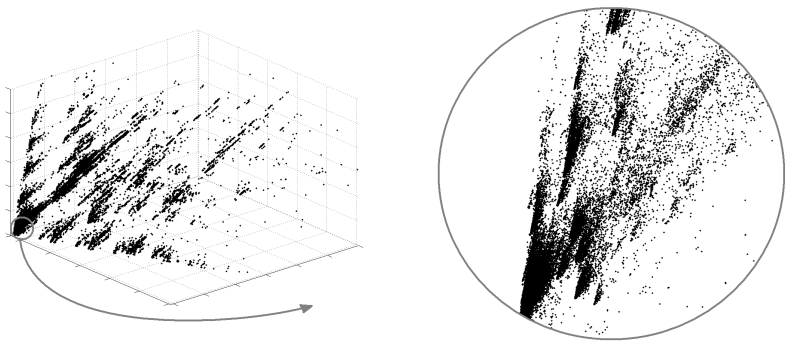
The first approach to capture information from the word sequences in documents, the HLG model did not perform well at the classification task. The state transition probabilities however showed that word order information was captured in the model. The reason for the lack of classification performance is therefore not to be found in the use of the state space model, but rather in the transformation of the vocabulary to a lower dimensional LSI-GMM vocabulary.

Previous experiments has shown that the 50 or more LSI components are needed to create an efficient classifier. It is therefore likely that valuable information is lost when we only use 4 LSI principal component directions here. The reason for only using few LSI components is that the use of many components makes it hard for the GMM to model the semantic space correctly. As illustrated in Figure 6, the density in the LSI subspace is very high in some areas, and the clusters are not very gaussian in shape. A very high amount of gaussian mixtures is therefore needed if they should cover a higher dimensional subspace. In practise the gaussian mixtures are poor at modeling the new LSI subspace, while they tend to cluster around high density areas when too many LSI dimensions are used. This gives a bad fit to many of the outer data-points, resulting in poor classification performance.

The HMM model with LSI estimated emission probabilities was much better at classifying documents correctly than the HLG model. The state transition probabilities did however not seem to capture any valuable information about the differences in word sequences for the three classes. It is likely that a true EM estimate of emission probabilities would have resulted in different transition probabilities that would capture more of the word order information, leading to better classification. A true EM estimate of shared latent topic emissions will however require that the a new HMM must be redefined and update rules determined.

8 Conclusion

We have used two state space model approaches to capture the information, that is contained in the order in which words appear in documents. The first approach involved a transformation of the document vocabulary, into a smaller LSI vocabulary, whereon a HMM could be trained. This approach lacked in classification ability but was conceptual



(a) full space (b) zoom

Figure 6: Zooming in on the LSI space of the document substrings. Some of the spaces are very dense on data, making the areas very attractive for the gaussian mixtures. When using high dimensional representations of the LSI substring space, the outer data points therefore tend to be badly modeled. Since much variation exists for the data in non dense areas, lack of modeling in these areas are likely to result in loss of information.

successful at capturing word order information. The second approach involved making an estimate of latent topic emission probabilities for at HMM using LSI. This approach had less success at capturing word order information, but was better at the classification task. We have hope that the HMM approach will have greater success by the development of a HMM with shared latent topic emission probabilities.

References

Bell, A., & Sejnowski, T. (1995a). Blind separation and blind deconvolution: An information-theoretic approach. *International Conference on Accoustics Speech and Signal Processing (ICASSP)* (pp. 3415–3418). Detroit, US.

Bell, A., & Sejnowski, T. (1995b). An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*, 1129–1159.

Blei, D., Ng, A., & Jordan, M. (2002). Latent dirichlet allocation. *Advances in Neural Information Processing Systems 14* (pp. 601–608). Cambridge, MA: MIT Press.

Blei, D., Ng, A., & Jordan, M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, *41*, 391–407.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B*, *39*, 1–38.

Furnas, G., Deerwester, S., Dumais, S., Landauer, T., Harshman, R., Streeter, L., & Lochbaum, K. (1988). Information retrieval using a singular value decomposition model of latent semantic structure. *The 11th International Conference on Research and Development in Information Retrieval* (pp. 465–480). Grenoble, France: ACM Press.

Hofmann, T. (1999). Probabilistic latent semantic indexing (PLSI). *Proceedings of the*

22nd Annual ACM Conference on Research and Development in Information Retrieval (pp. 50–57). Berkeley, California: ACM.

Kolenda, T., Hansen, L., Larsen, J., & Winther, O. (2002). Independent component analysis for understanding multimedia content. *Proceedings of IEEE Workshop on Neural Networks for Signal Processing XII* (pp. 757–766). Piscataway, New Jersey: IEEE Press.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 789–792.

Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13* (pp. 556–562). Cambridge, MA: MIT Press.

Madsen, R., Hansen, L., & Winther, O. (2003). Singular value decomposition and principal component analysis, isp technical report.

Molgedey, L., & Schuster, H. (1994). Separation of independent signals using time-delayed correlations. *Physical Review Letters*, *72*, 3634–3637.

Rabiner, L. (1989). A tutorial on hidden markov models. *Proceedings of the IEEE*, *77*, 257–286.

Rabiner, L., & Juang, B. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine*, *74*, 4–15.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Xu, W., Liu, X., & Gong, Y. (2003). Document clustering based on non-negative matrix factorization. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 267 – 273). Toronto, CA: ACM press.

Zimmermann, M., & Bunke, H. (2004). Optimizing the integration of a statistical language model in hmm based online handwritten text recognition. *Proceedings of the 17th International Conference on (ICPR'04)* (pp. 541–544).