Hidden Topic Markov Models

Amit Gruber
School of CS and Eng.
The Hebrew University
Jerusalem 91904 Israel
amitg@cs.huji.ac.il

Michal Rosen-Zvi
IBM Research Laboratory in Haifa
Haifa University, Mount Carmel
Haifa 31905 Israel
rosen@il.ibm.com

Yair Weiss
School of CS and Eng.
The Hebrew University
Jerusalem 91904 Israel
yweiss@cs.huji.ac.il

Abstract

Algorithms such as Latent Dirichlet Allocation (LDA) have achieved significant progress in modeling word document relationships. These algorithms assume each word in the document was generated by a hidden topic and explicitly model the word distribution of each topic as well as the prior distribution over topics in the document. Given these parameters, the topics of all words in the same document are assumed to be independent.

In this paper, we propose modeling the topics of words in the document as a Markov chain. Specifically, we assume that all words in the same sentence have the same topic, and successive sentences are more likely to have the same topics. Since the topics are hidden, this leads to using the well-known tools of Hidden Markov Models for learning and inference. We show that incorporating this dependency allows us to learn better topics and to disambiguate words that can belong to different topics. Quantitatively, we show that we obtain better perplexity in modeling documents with only a modest increase in learning and inference complexity.

1 Introduction

Extensively large text corpora are becoming abundant due to the fast growing storage and processing capabilities of modern computers. This has lead to a resurgence of interest in automated extraction of useful information from text. Modeling the observed text as generated from latent aspects or topics is a prominent approach in machine learning studies of texts (e.g., [8, 14, 3, 4, 6]). In such models, the "bag-of-words" assumption is often employed, an assumption that the order of words can be ignored and text corpora can be represented by a co-occurrence matrix of words and documents. The probabilistic latent semantic analysis (PLSA) model [8] is such a model. It employs two parameters inferred from the observed words: (a) A global parameter that ties the documents in the corpora, is fixed for the corpora and stands for the probability of words given topics. (b) A set of parameters for each of the documents that stand for the probability of topics in a document. The Latent Dirichlet Allocation (LDA) model [3] introduces a more consistent probabilistic approach as it ties the parameters of all documents via a hierarchical generative model. The mixture of topics per document in the LDA model is generated from a Dirichlet prior mutual to all documents in the corpus.

These models are not only computationally efficient, but also seem to capture correlations between words via the topics. Yet, the assumption that the order of words can be ignored is an unrealistic oversimplification. Relaxing this assumption is expected to yield better models in terms of the latent aspects inferred, their performance in word sense disambiguation task and the ability of the model to predict previously unobserved words in trained documents.

Markov models such as N-grams and HMMs that capture local dependencies between words have been employed mainly in part-of-speech tagging [5]. Models for semantic parsing tasks often use a "shallow" model with no hidden states [9]. In recent years several probabilistic models for text that infer topics and incorporate Markovian relations have been studied. In [7] a model that integrates topics and syntax is introduced. It contains a latent variable per each word that stands for syntactic classes. The model posits that words are either generated from topics that are randomly drawn from the topic mixture of the document or from the syntactic classes that are drawn from the previous syntactic class. Only the latent variables of the syntactic classes are treated as a sequence with local dependencies while latent assignments of topics are treated similar to the LDA model, namely topics extraction is not benefited from the additional information conveyed in the structure of words. During the last couple of years, a few models were introduced in which consecutive words are modeled by Markovian relations. These are the Bigram topic model [15], the LDA collocation model and the Topical n-grams model [16]. All these models assume that words generation in texts depend on a latent topic assignment as well as on the n-previous words in the text. This added complexity seem to provide the models with more predictive power compared to the LDA model and to capture relations between consecutive words. We follow the same lines, while we allow Markovian relations between the hidden aspects. A somewhat related model is the aspect HMM model [2], though it models unstructured data that contains stream of words. The model contains latent topics that have Markovian relations. In the aspect HMM model, documents or segments are inferred using heuristics that assume that each segment is generated from a unique topic assignment and thus there is no notion of mixture of topics.

We strive to extract latent aspects from documents by making use of the information conveyed in the division into documents as well as the particular order of words in each document. Following these lines we propose in this paper a novel and consistent probabilistic model we call the Hidden Topic Markov Model (HTMM). Our model is similar to the LDA model in tying together parameters of different documents via a hierarchical generative model, but unlike the LDA model it does not assume documents are "bags of words". Rather it assumes that the topics of words in a document form a Markov chain, and that subsequent words are more likely to have the same topic. We show that the HTMM model outperforms the LDA model in its predictive performance and can be used for text parsing and word sense disambiguation purposes.

2 Hidden Topic Markov Model for Text Documents

We start by reviewing the formalism of LDA. Figure 1a shows the graphical model corresponding to the LDA generative model. To generate a new word w in a document, one starts by first sampling a hidden topic z from a multinomial distribution defined by a vector θ corresponding to that document. Given the topic z, the distribution over words is multinomial with parameters β_z. The LDA model ties parameters between different documents by drawing θ of all documents from a common Dirichlet prior parameterized by α. It also ties parameters between topics by drawing the vector β_z of all topics from a common Dirichlet prior parameterized by η.

In order to make the independence assumptions in LDA more explicit, figure 1b shows an alternative graphical model for the same generative process. Here, we have explicitly denoted the hidden topics z_i and the observed words w_i as separate nodes in the graphs (rather than summarizing them with a plate). From figure 1b it is evident that conditioned on θ and β, the hidden topics are independent.

The Hidden Topic Markov Model drops this independence assumption. As shown in figure 1c, the topics in a document form a Markov chain with a transition probability that depends on θ and a topic transition variable ψ_n. When $\psi_n = 1$, a new topic is drawn from θ. When $\psi_n = 0$ the topic of the nth word is identical to the previous one. We assume that topic transitions can only occur between sentences, so that ψ_n may only be nonzero for the first word in a sentence.

Formally the model can be described as:

```
1. for z=1...K,
   Draw $\beta_z \sim$ Dirichlet($\eta$)
2. for d=1...D,
   Document $d$ is generated as follows:
   (a) Draw $\theta \sim$ Dirichlet($\alpha$)
   (b) Set $\psi_1 = 1$
   (c) for n=2 ... $N_d$
       i. If (begin_sentence) draw $\psi_n \sim Binom(\epsilon)$
          else $\psi_n = 0$
   (d) for n=1... $N_d$
       i. if $\psi_n == 0$ then $z_n = z_{n-1}$
          else $z_n \sim multinomial(\theta)$
       ii. Draw $w_n \sim multinomial(\beta_{z_n})$
```

where K is the number of latent topics and N_d is the length of document d. Note that if we force a topic transition between any two words (i.e. set $\psi_n = 1$ for all n) we obtain the LDA model [3]. At the other extreme, if we do not allow any topic transitions and set $\psi_n = 0$ between any two words, we obtain the mixture of unigrams model in which all words in the document are assumed to have the same topic [11].

Unlike the LDA and mixture of unigrams models, the HTMM model (which allows infrequent topic transitions within a document) is no longer invariant to a reshuffling of the words. Documents for which successive words have the same topics are more likely than a random permutation of the same words. The input to the algorithm is the entire document, rather than a document-word co-occurrence matrix. This obviously increases the storage requirement for each document,

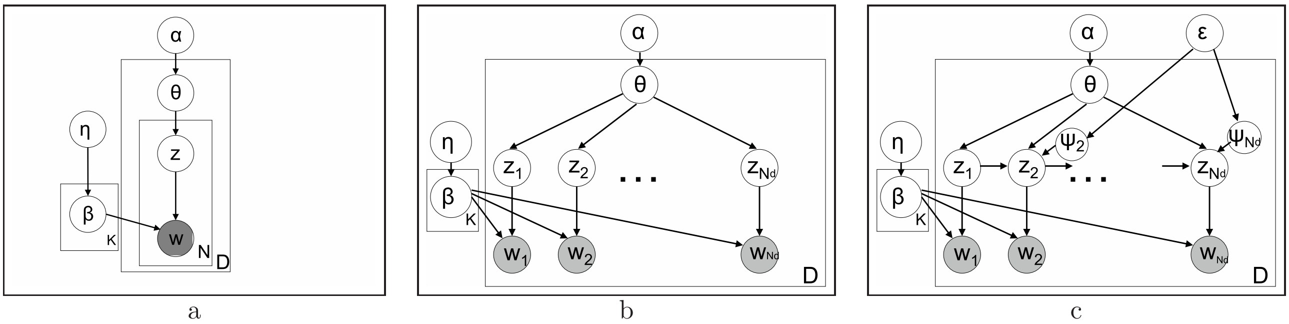


Figure 1: **a.** The LDA model. **b.** The LDA model. The repeated word generation within the document is explicitly drawn rather than using the plate notation. **c.** The HTMM model proposed in this paper. The hidden topics form a Markov chain. The order of words and their proximity plays an important role in the model.

but it allows us to perform inferences that are simply impossible in the bag of words models. For example, the topic assignments calculated during HTMM inference (either hard assignments or soft ones) will not give the same topic to all appearances of the same word within the same document. This can be useful for word sense disambiguation in applications such as automatic translation of texts. Furthermore, the topic assignment will tend to be linearly coherent - subsections of the text will tend to be assigned to the same topic, as opposed to "bag-of-words" assignments which may fluctuate between topics in successive words.

3 Approximate Inference

Due to the parameter-tying in LDA type models, calculating exact posterior probabilities over the parameters β, θ is intractable. In recent years, several alternatives for approximate inference have been suggested: EM [8] or variational EM [3], Expectation propagation (EP) [10] and Monte-Carlo sampling [13, 7]. In this paper, we take advantage of the fact that conditioned on β and θ the HTMM model is a special type of HMM. This allows us to use the standard parameter estimation tools of HMMs, namely Expectation-Maximization and the forward-backward algorithm [12].

Unlike fully Bayesian inference methods, the standard EM algorithm for HMMs distinguishes between latent variables and parameters. Applied to the HTMM model, the latent variables are the topics z_n and the variables ψ_n that determine whether the topic n will be identical to topic $n-1$ or will it be drawn according to θ_d. The parameters of the problem are θ_d , β and ϵ. We assume the hyper parameters α and η to be known.

E-step:

In the E-step the probability $\Pr(z_n, \psi_n | d, w_1 \ldots w_{N_d}; \theta, \beta, \epsilon)$ is computed for each sentence in the document. We compute this probability using the forward-backward algorithm for HMM. The transition matrix is specific per document and depends on the parameters θ_d and ϵ. The parameter $\beta_{z,w}$ induces local probabilities on the sentences. After computing $\Pr(z_n, \psi_n | d, w_1 \ldots w_{N_d})$ we compute expectations required for the M-step. Specifically, we compute (1) the expected number of topic transitions that ended in the topic z and (2) The expected number of co-occurrence of a word w with a topic z.

Let $C_{d,z}$ denote the number of times that topic z was drawn according to θ_d in document d. Let $C^{z,w}$ denote the number of times that word w was drawn from topic z according to $\beta_{z,w}$.

$$E(C_{d,z}) = \sum_{n=1}^{N} \Pr(z_{d,n}=z, \psi_{d,n}=1 | w_1 \ldots w_{N_d}) \quad (1)$$

$$E(C^{z,w}) = \sum_{d=1}^{D} \sum_{n=1}^{N} \Pr(z_{d,n}=z, w_{d,n}=w | w_1 \ldots w_{N_d}) \quad (2)$$

M-step:

The MAP estimators for θ and β are found under the constraint that θ_d and β_z are probability vectors. Standard computation using Lagrange multipliers yields:

$$\theta_{d,z} \propto E(C_{d,z}) + \alpha - 1 \quad (3)$$

$$\epsilon = \frac{\sum_d \sum_{n=2}^{N_d} \Pr(\psi_{d,n} = 1 | w_1 \ldots w_{N_d})}{\sum_d (N_d^{sen} - 1)} \quad (4)$$

where $\theta_{d,z}$ is normalized such that θ_d is a distribution and N_d^{sen} is the number of sentences in the document d. After computing θ and ϵ the transition matrix is updated accordingly.

Similarly, the probabilities $\beta_{z,w}$ are set to:

$$\beta_{z,w} \quad \propto \quad E(C^{z,w}) + \eta - 1 \tag{5}$$

Again, $\beta_{z,w}$ is normalized such that β_z forms a distribution.

EM algorithms have been discussed for word document models in several recent papers. Hofmann used EM to estimate the maximum likelihood parameters in the pLSA model [8]. Here, in contrast, we use EM to estimate the MAP parameters in a hierarchical generative model similar to LDA (note that our M step explicitly takes into account the Dirichlet priors on β, θ). Griffiths et al. [6] also discuss using EM in their HMM model of topics and syntax and say that Gibbs sampling is preferable since EM can suffer from local minima. We should note that in our experiments we found the final solution calculated by EM to be stable with respect to multiple initializations. Again, this may be due to our use of a hierarchical generative model which reduces somewhat the degrees of freedom available to EM.

Our EM algorithm assumes the hyper parameters α, η are fixed. We set these parameters according to the values used in previous papers [13] ($\alpha = 1 + 50/K$ and $\eta = 1.01$).

4 Experiments

We first demonstrate our results on the NIPS dataset [1]. The NIPS dataset consists of 1740 documents. The train set consists of 1557 documents and the test set consists of the remaining 183. The vocabulary contains 12113 words. From the raw data we extracted (only) the words that appear in the vocabulary, preserving their order. Stop words do not appear in the vocabulary, hence they were discarded from the input. We divided the text to sentences according to the delimiters .?!;. This simple preprocessing is very crude and noisy since abbreviations such as “Fig. 1” will terminate the sentence before the period and will start a new one after it. The only exception is that we omitted appearances of “e.g.” and “i.e.” in our preprocessing [2].

On this dataset we compared perplexity with HTMM and LDA with 100 topics. We also considered a variant of HTMM (which we call VHTMM1) that sets the topic transition probability to be 1 between sentences. This is a “bag of sentence” model (where the order of words within a sentence is ignored): all words in the same sentence share the same topic; topics of different sentences are drawn from θ independently. In all the experiments the same values of α, η were provided to all algorithms.

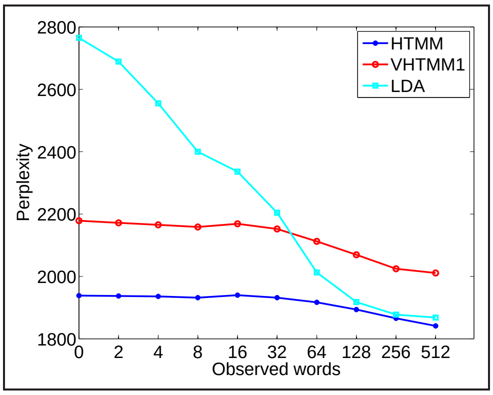


Figure 2: Comparison between the average perplexity of the NIPS corpus of HTMM, LDA and VHTMM1. HTMM outperforms the bag of words (LDA) and VHTMM1 models.

The perplexity of an unseen test document after observing the first N words of the document is defined to be:

$$\text{Perplexity} = \exp\left(-\frac{\log \Pr(w_{N+1} \dots w_{N_{test}} | w_1 \dots w_N)}{N_{test} - N}\right) \tag{6}$$

where N_{test} is the length of the test document. The perplexity reflects the difficulty of predicting a new unseen document after learning from a train set. The lower the perplexity, the better the model.

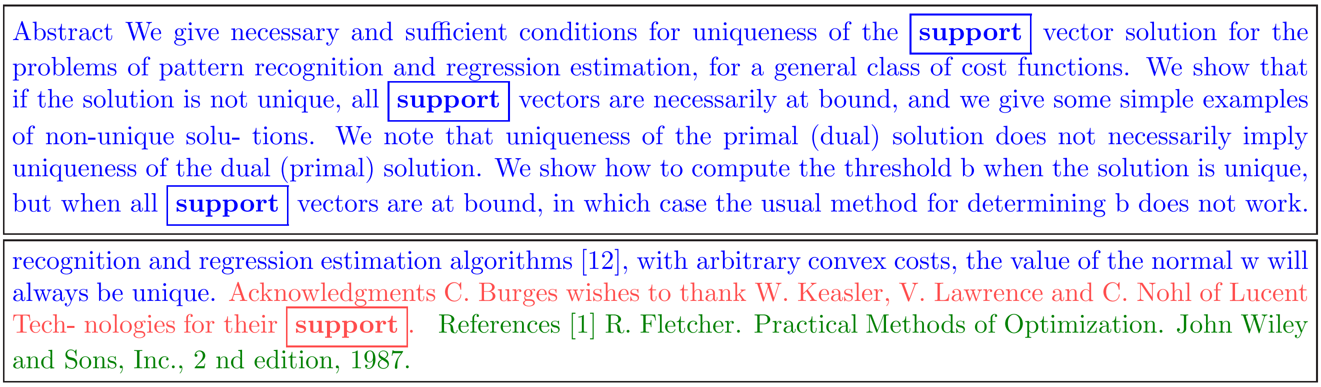
The perplexity for the HTMM model is computed as follows: First θ_{new} is found from the first N words of the new document. θ_{new} is found using EM holding $\beta = \beta_{train}$ and $\epsilon = \epsilon_{train}$ fixed. Second, $\Pr(z_{N+1}|w1 \dots w_N)$ is computed by inference on the latent variables of the first N words using the forward backward algorithm for HMM and then inducing $\Pr(z_N|w1 \dots w_N)$ on $\Pr(z_{N+1}|w1 \dots w_N)$ (using the transition matrix). Third, $\log \Pr(w_{N+1} \dots w_{N_{test}}|w_1 \dots w_N)$ is computed using forward backward with θ_{new}, $\beta_{train}, \epsilon_{train}$ and $\Pr(z_{N+1}|w1 \dots w_N)$ that were previously found.

Figure 2 shows the perplexity of HTMM, LDA and VHTMM1. with $K = 100$ topics as a function of the number of observed words at the beginning of the test document, N. We see that for a small number of words, both HTMM and VHTMM1 have significantly lower perplexity than LDA. This is due to combining the information from subsequent words regarding the topic of the sentence.

When the number of observed words increases, HTMM

[1] http://www.cs.toronto.edu/∼roweis/data.html

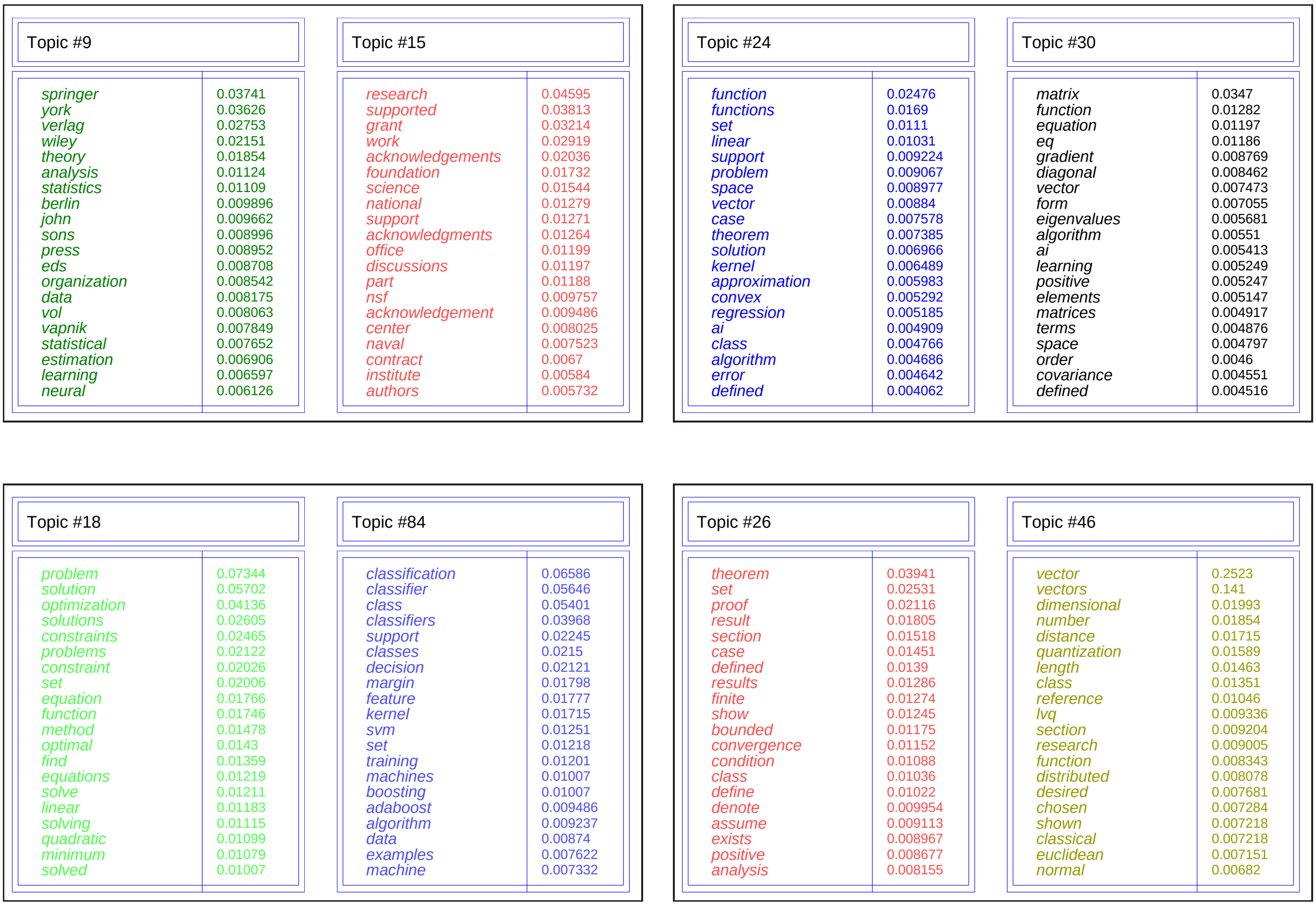
[2] The data we used is available at: http://www.cs.huji.ac.il/∼amitg/htmm.html



Abstract We give necessary and sufficient conditions for uniqueness of the **support** vector solution for the problems of pattern recognition and regression estimation, for a general class of cost functions. We show that if the solution is not unique, all **support** vectors are necessarily at bound, and we give some simple examples of non-unique solu- tions. We note that uniqueness of the primal (dual) solution does not necessarily imply uniqueness of the dual (primal) solution. We show how to compute the threshold b when the solution is unique, but when all **support** vectors are at bound, in which case the usual method for determining b does not work.

recognition and regression estimation algorithms [12], with arbitrary convex costs, the value of the normal w will always be unique. Acknowledgments C. Burges wishes to thank W. Keasler, V. Lawrence and C. Nohl of Lucent Tech- nologies for their **support**. References [1] R. Fletcher. Practical Methods of Optimization. John Wiley and Sons, Inc., 2 nd edition, 1987.

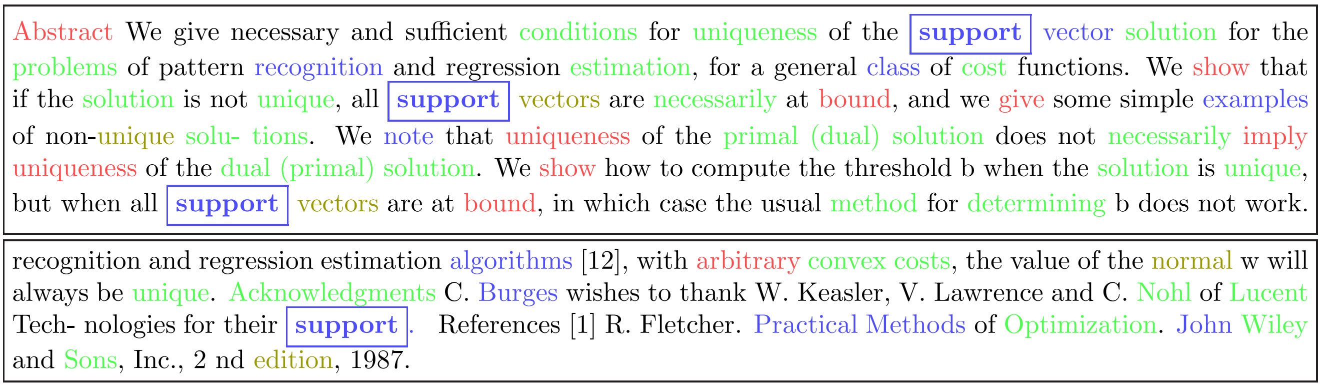
Figure 3: An example of segmenting a document according to its semantic and of word sense disambiguation. **Top** Beginning of an abstract of a NIPS paper about support vector machines. The entire section is assigned latent topic 24 that correspond to mathematical terms (see figure 4). The word "support" appears in this section 3 times as a part of the mathematical term "support vector" and is assigned the mathematical topic.
Bottom A section from the end of the same NIPS paper. The end of the discussion was assigned the same mathematical topic (24) as the abstract, the acknowledgments section was assigned topic 15 and the references section was assigned topic 9. Here, in the acknowledgements section, the word "support" refers to the help received. Indeed it is assigned latent topic 15 like the entire acknowledgments section.



Topic #9		Topic #15		Topic #24		Topic #30	
springer	0.03741	research	0.04595	function	0.02476	matrix	0.0347
york	0.03626	supported	0.03813	functions	0.0169	function	0.01282
verlag	0.02753	grant	0.03214	set	0.0111	equation	0.01197
wiley	0.02151	work	0.02919	linear	0.01031	eq	0.01186
theory	0.01854	acknowledgements	0.02036	support	0.009224	gradient	0.008769
analysis	0.01124	foundation	0.01732	problem	0.009067	diagonal	0.008462
statistics	0.01109	science	0.01544	space	0.008977	vector	0.007473
berlin	0.009896	national	0.01279	vector	0.00884	form	0.007055
john	0.009662	support	0.01271	case	0.007578	eigenvalues	0.005681
sons	0.008996	acknowledgments	0.01264	theorem	0.007385	algorithm	0.00551
press	0.008952	office	0.01199	solution	0.006966	ai	0.005413
eds	0.008708	discussions	0.01197	kernel	0.006489	learning	0.005249
organization	0.008542	part	0.01188	approximation	0.005983	positive	0.005247
data	0.008175	nsf	0.009757	convex	0.005292	elements	0.005147
vol	0.008063	acknowledgement	0.009486	regression	0.005185	matrices	0.004917
vapnik	0.007849	center	0.008025	ai	0.004909	terms	0.004876
statistical	0.007652	naval	0.007523	class	0.004766	space	0.004797
estimation	0.006906	contract	0.0067	algorithm	0.004686	order	0.0046
learning	0.006597	institute	0.00584	error	0.004642	covariance	0.004551
neural	0.006126	authors	0.005732	defined	0.004062	defined	0.004516

Topic #18		Topic #84		Topic #26		Topic #46	
problem	0.07344	classification	0.06586	theorem	0.03941	vector	0.2523
solution	0.05702	classifier	0.05646	set	0.02531	vectors	0.141
optimization	0.04136	class	0.05401	proof	0.02116	dimensional	0.01993
solutions	0.02605	classifiers	0.03968	result	0.01805	number	0.01854
constraints	0.02465	support	0.02245	section	0.01518	distance	0.01715
problems	0.02122	classes	0.0215	case	0.01451	quantization	0.01589
constraint	0.02026	decision	0.02121	defined	0.0139	length	0.01463
set	0.02006	margin	0.01798	results	0.01286	class	0.01351
equation	0.01766	feature	0.01777	finite	0.01274	reference	0.01046
function	0.01746	kernel	0.01715	show	0.01245	lvq	0.009336
method	0.01478	svm	0.01251	bounded	0.01175	section	0.009204
optimal	0.0143	set	0.01218	convergence	0.01152	research	0.009005
find	0.01359	training	0.01201	condition	0.01088	function	0.008343
equations	0.01219	machines	0.01007	class	0.01036	distributed	0.008078
solve	0.01211	boosting	0.01007	define	0.01022	desired	0.007681
linear	0.01183	adaboost	0.009486	denote	0.009954	chosen	0.007284
solving	0.01115	algorithm	0.009237	assume	0.009113	shown	0.007218
quadratic	0.01099	data	0.00874	exists	0.008967	classical	0.007218
minimum	0.01079	examples	0.007622	positive	0.008677	euclidean	0.007151
solved	0.01007	machine	0.007332	analysis	0.008155	normal	0.00682

Figure 4: Twenty most probable words for 4 topics out of 100 found for each model. **Top** HTMM topics. **Bottom** LDA topics. Aside each word is its probability to be generated from the corresponding topic ($\beta_{z,w}$).



Abstract We give necessary and sufficient conditions for uniqueness of the support vector solution for the problems of pattern recognition and regression estimation, for a general class of cost functions. We show that if the solution is not unique, all support vectors are necessarily at bound, and we give some simple examples of non-unique solu- tions. We note that uniqueness of the primal (dual) solution does not necessarily imply uniqueness of the dual (primal) solution. We show how to compute the threshold b when the solution is unique, but when all support vectors are at bound, in which case the usual method for determining b does not work.

recognition and regression estimation algorithms [12], with arbitrary convex costs, the value of the normal w will always be unique. Acknowledgments C. Burges wishes to thank W. Keasler, V. Lawrence and C. Nohl of Lucent Tech- nologies for their support. References [1] R. Fletcher. Practical Methods of Optimization. John Wiley and Sons, Inc., 2 nd edition, 1987.

Figure 5: Topics assigned by LDA to the words in the same sections as in figure 3. For demonstration, the four most frequent topics in these sections are displayed in colors: topic 18 in green, topic 84 in blue, topic 46 in yellow and topic 26 in red. In contrast to figure 3, LDA assigned all the 4 instances of the word “support” the same topic (84) due to its inability to distinguish between different instances of the same word according to the context.

and LDA have lower perplexity than VHTMM1. HTMM is consistently better than LDA but the difference in perplexity is significant only for $N \leq 64$ observed words (the average length of a document after our preprocessing was about 1300 words).

Figure 3 shows the topical segmentation calculated by HTMM. On top the beginning of the paper is shown. A single topic is assigned to the entire section. This is topic 24 that corresponds to mathematical terms (related to support vector machines which is the subject of the paper) and is shown in figure 4. On the bottom we see a section taken from the end of the paper, consisting of the end of the discussion, acknowledgements and the beginning of the references. The end of the discussion is assigned topic 24 (the same as the abstract), as it addresses mathematical issues. Then the acknowledgments section is assigned topic 15 and the references section is assigned topic 9.

Figure 5 shows the topics assigned to the words in the same section by LDA. We see that LDA assigns different topics to different words within the same sentence and therefore it is not suitable for topical segmentation.

The HTMM model distinguishes between different instances of the same word according to the context. Thus we can use it to disambiguate words that have several meanings according to the context. The word “support” appears several times in the document in figures 3 and 5. On top, it appears 3 times in the abstract, always as a part of the mathematical term “support vector”. On bottom, it appears once in the acknowledgments section in the sense of help provided.

HTMM assigns the 3 instances of the word in the abstract the same mathematical topic 24. The instance in the acknowledgments section is assigned the acknowledgments topic 15. LDA, on the other hand assigns all the 4 instances the same topic 83 (see figure 4.

Figure 4 shows 4 topics learnt by HTMM and 4 topics learnt by LDA. In each topic the 20 top words (most probable words) are shown. Each word appears with its probability to be generated from the corresponding topic ($\beta_{z,w}$).

The 4 topics of each model were chosen for illustration out of 100 topics because they appear in the text sections in figures 3 and 5. The complete listing of top words of all the 100 topics found by HTMM and LDA is available at: http://www.cs.huji.ac.il/∼amitg/htmm.html

The 4 topics shown in figure 4 for both models are easy for interpretation according to the top words (among the 100 topics some topics are “as nice” and some are not). However, we see that the topics found by the two models are different in nature. HTMM finds topics that consist of words that are consecutive in the document. For example, topic 15 in figure 4 corresponds to acknowledgements. Figure 3 shows that indeed the entire acknowledgements section (and nothing but it) was assigned to topic 15. On the other hand no such topic exists among the 100 topics found by LDA. Another example for a topic that consists of consecutive words is topic 9 in figure 4 (that corresponds to references as it consists of names of journals, publishers, authors etc.). The value learnt for ϵ by HTMM and reflects the topical coherence of the NIPS documents was $\epsilon = 0.376278$.

Figure 7 shows the value of ϵ learnt by HTMM as a function of the number of topics. It shows how the

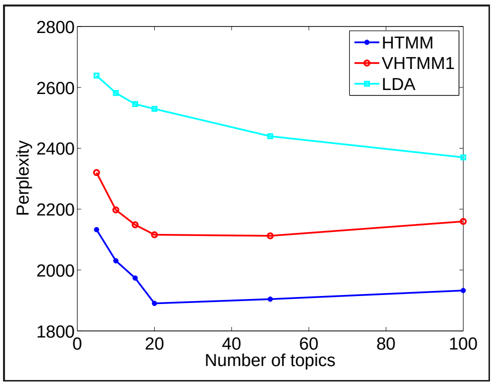


Figure 6: Perplexity of the different models as a function of the number of topics for $N = 10$ observed words.

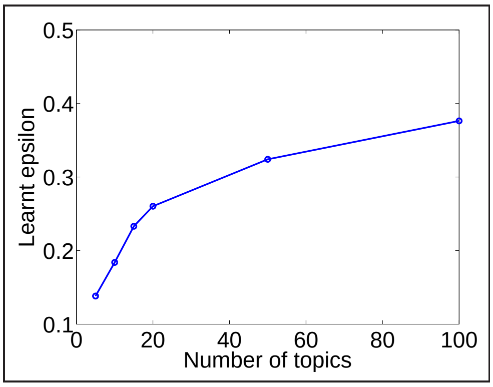


Figure 7: MAP values for ϵ as a function of the number of topics for the NIPS train set. When only few topics are available, learnt topics are general and topic transitions are infrequent. When many topics are available topics become specific and topic transitions become more frequent.

dependency between topics changes with the granularity of topics. When the dataset is learnt with a few topics, these topics are general and do not change often. As more topics are available, the topics become more specific and topic transitions are more frequent. This suggests a correlation between the different topics ([1]). Having learnt always values smaller than 1 means that the likelihood of data is higher when consecutive latent topics are dependent.

The picture of perplexity we see in figure 2 may suggest that the Markovian structure exists in real documents. We would like to eliminate the option that the perplexity of HTMM might be lower than the perplexity of LDA only because it has less degrees of freedom (due to the dependencies between latent topics). We created two toy datasets with 1000 documents each, from which 900 formed the train set and the other 100 formed the test set for each case. The vocabulary size was 200 words and there were 5 latent topics. The first data set was generated using HTMM with $\epsilon = 0.1$ and the second dataset was created using LDA.

In table 1 we compare LDA and HTMM in terms of perplexity using the first 100 words (the average number of words in a document in this dataset is 600), in terms of estimation error on the parameters β (i.e. how well the true topics are learnt) and in terms of erroneous assignments of latent topics to words.

As a sanity check, we find that indeed HTMM recovers the correct parameters and correct ϵ for data generated by a HTMM. We also see that when the data indeed deviates from the "bag of words" assumption - HTMM significantly outperforms LDA in all three criteria. However, when the data is generated with the "bag of words" assumption - the HTMM model no longer has better perplexity. Thus the fact that we obtained better perplexity on the NIPS dataset with HTMM may reflect the poor fit of the "bag of words" assumption to the NIPS documents.

5 Discussion

Statistical models for documents have been the focus of much recent work in machine learning. In this paper we have presented a new model for text documents. We extend the LDA model by considering the underlying structure of a document. Rather than assuming that the topic distribution within a document is conditionally independent, we explicitly model the topic dynamics with a Markov chain. This leads to a HMM model of topics and words for which efficient learning and inference algorithms exist. We find that this Markovian structure allows us to learn more coherent topics and to disambiguate the topics of ambiguous words. Quantitatively we show that this leads to a significant improvement in predicting unseen words (perplexity).

The incorporation of HMM into the LDA model is orthogonal to other extensions of LDA such as integrating syntax [7] and author [13] information. It would be interesting to combine these different extensions together to form a better document analysis system.

Acknowledgements

Support from the Israeli Science Foundation is greatfully acknowledged.

References

[1] David M. Blei and John Lafferty. Correlated topic models. In Lawrence K. Saul, Yair Weiss, and

Table 1: Comparison between LDA and HTMM. The models are compared using toy data generated from the different models where there is ground truth to compare with. Comparison is made in terms of errors in topic assignment to words, l1 norm between true and learnt probabilities and perplexity given 100 words.

Data Generator	Performance criterion	HTMM	LDA
HTMM $\epsilon = 0.1$	Percent mistake	0	37.11%
HTMM $\epsilon = 0.1$	$\|\beta_{true} - \beta_{recovered}\|_1$	0.15	1.74
HTMM $\epsilon = 0.1$	Perplexity	134.00	174.39
LDA	Percent mistake	73.26%	68.29%
LDA	$\|\beta_{true} - \beta_{recovered}\|_1$	3.35	3.05
LDA	Perplexity	184.87	184.64

Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, Cambridge, MA, 2005. MIT Press.

[2] David M. Blei and Pedro J. Moreno. Topic segmentation with an aspect hidden markov model. In *Research and Development in Information Retrieval*, pages 343–348, 2001.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] Wray L. Buntine and Aleks Jakulin. Applying discrete PCA in data analysis. In Max Chickering and Joseph Halpern, editors, *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 59–66, San Francisco, CA, 2004. Morgan Kaufmann Publishers.

[5] Kenneth Ward Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the second conference on Applied natural language processing*, pages 136–143, Morristown, NJ, USA, 1988. Association for Computational Linguistics.

[6] Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101:5228–5235, 2004.

[7] Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. Integrating topics and syntax. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 537–544. MIT Press, Cambridge, MA, 2005.

[8] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 289–29, San Francisco, CA, 1999. Morgan Kaufmann.

[9] David J. C. MacKay and Linda C. Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(3):1–19, 1994.

[10] Thomas Minka and John Lafferty. Expectation-propagation for the generative aspect model. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 352–359, San Francisco, CA, 2002. Morgan Kaufmann Publishers.

[11] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Mach. Learn.*, 39(2-3):103–134, 2000.

[12] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257–286, 1989.

[13] Michal Rosen-Zvi, Tom Griffith, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In Max Chickering and Joseph Halpern, editors, *Proceedings 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494, San Francisco, CA, 2004. Morgam Kaufmann.

[14] N. Slonim and N. Tishby. The power of word clusters for text classification. In *23rd European Colloquium on Information Retrieval Research*, 2001.

[15] Hanna M. Wallach. Topic modeling: Beyond bag-of-words. In *ICML '06: 23rd International Conference on Machine Learning*, Pittsburgh, Pennsylvania, USA, June 2006.

[16] Xuerui Wang and Andrew McCallum. A note on topical n-grams. Technical Report UM-CS-071, Department of Computer Science University of Massachusetts Amherst, 2005.