



# Computer Sciences Department

## **TagLDA: Bringing Document Structure Knowledge Into Topic Models**

Xiaojin (Jerry) Zhu  
David Blei  
John Lafferty

Technical Report #1553

February 2006

UNIVERSITY OF  
WISCONSIN  
MADISON



# TagLDA: Bringing document structure knowledge into topic models

Xiaojin Zhu

jerryzhu@cs.wisc.edu

Computer Science Department

University of Wisconsin, Madison, WI 53706

David Blei

blei@cs.princeton.edu

Computer Science Department

Princeton University, Princeton, NJ 08544

John Lafferty

lafferty@cs.cmu.edu

School of Computer Science

Carnegie Mellon University, Pittsburgh, PA 15213

## Abstract

Latent Dirichlet Allocation models a document by a mixture of topics, where each topic itself is typically modeled by a unigram word distribution. Documents however often have known structures, and the same topic can exhibit different word distributions under different parts of the structure. We extend latent Dirichlet allocation model by replacing the unigram word distributions with a factored representation conditioned on both the topic and the structure. In the resultant model each topic is equivalent to a set of unigrams, reflecting the structure a word is in. The proposed model is more flexible in modeling the corpus. The factored representation prevents combinatorial explosion and leads to efficient parameterization. We derive the variational optimization algorithm for the new model. The model shows improved perplexity on text and image data, but not significant accuracy improvement when used for classification.

## 1 Introduction

Latent Dirichlet Allocation (LDA) is a powerful topic model [1] [2]. LDA model is completely data driven. But sometimes there are certain domain knowledge one wishes to incorporate into an LDA model. In this paper we consider domain knowledge in the form of tags on words. The tags can be quite general. For example in text documents each word can be tagged with its part-of-speech (POS), obtained from a POS tagger. In

HTML web pages each word can be tagged as whether it appears on a hyperlink (anchor text), or body text. For scholarly papers there is usually a fixed structure (abstract, body, references, etc.), and each word can be tagged by the section it is in. We assume the set of tags are pre-defined and known. We also assume for each word in the corpus, its corresponding tag is given. Therefore the tags constitute domain knowledge. In this paper we do not consider higher order tags that apply to a pair or a group of words.

How should tags affect a topic model? Tags and topics can be thought of as orthogonal to each other. It is important to note that in LDA the same unigram is used throughout the document whenever a given topic is about to generate a word. But the same topic can have different word distribution under different tags. Knowing the tags should allow us to build a better model than using the topic model alone.

However the interaction between tags and topics can be subtle. On one hand, for the same topic the word distributions under different tags may be different. For instance if the tags represent part-of-speech, on a *space* topic the high probability words with a *noun* tag might be “space, shuttle, mission, launch, ...”, while those with a *verb* tag might be “make, launch, plan, schedule, ...”. On the other hand, these distributions may also be similar, depending on the nature of the tags. Consider the case where tags are section information in scholarly papers. On a *neural network chip* topic, the high probability words with an *abstract* tag might be “neural, network, chip, system, parallel, ...”, and those with a *body* tag might be “network, neural, time, chip, system, ...”.

One naive way to incorporate tags is to treat different tags separately. If we were to build a  $k$ -topic model without the tags, we can now build one  $k$ -topic model for each tag by ignoring all words in a document with other tags. Simple as it is, this approach has several shortcomings: 1. It fragments the corpus so that rare tags cannot be trained well. 2. It ignores the similarity between tags like *abstract* and *body*. 3. It results in a large number of parameters. With  $k$  topics,  $N_t$  different tag types and a vocabulary of size  $V$ , the number of parameters is  $k \times N_t \times V$ .

In this paper we propose *tagLDA*, a topic model which combines latent Dirichlet allocation (LDA) and tag knowledge using a factored representation. *tagLDA* addresses all three shortcomings above at the same time.

## 2 Representation

*tagLDA* model assumes the following generative process for each document  $w$  in a corpus  $D$ , given the tags  $t$  of the words:

1. Choose  $\theta \sim \text{Dir}(\alpha)$ .  $\theta$  is the topic multinomial with  $k$  outcomes for the document.  $\text{Dir}(\alpha)$  is a Dirichlet distribution with hyperparameter  $\alpha$ .
2. For word positions  $n = 1 \dots N$ , with tag  $t_n$ :
  - (a) Choose a topic  $z_n \sim \text{Multinomial}(\theta)$ .
  - (b) Choose a word  $w_n \sim P(w_n|z_n, t_n, \tau, \pi)$ , a word multinomial with  $V$  outcomes.

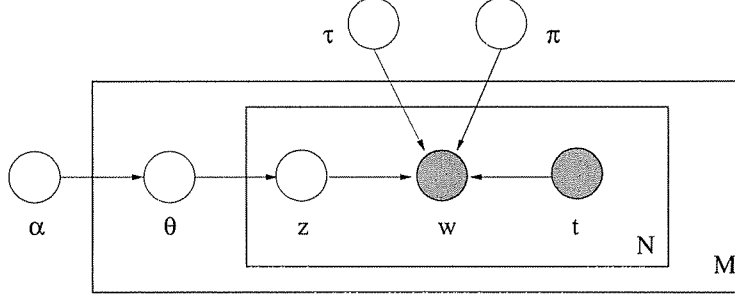


Figure 1: Graphical model representation of tagLDA. The outer plate represents documents, the inner plate represents words. The dark nodes are observed variables.

As in standard LDA, the dimensionality  $k$  of the Dirichlet distribution (i.e. the number of topics) is assumed given and fixed.

The key difference between the standard LDA model and the tagLDA model is how words are generated. The word probabilities are parameterized by a factored representation.  $\tau$  is a  $k \times V$  topic-word matrix which corresponds to the logarithm of the word multinomial parameters  $\beta$  in [1].  $\pi$  is a  $N_t \times V$  tag-word matrix, where  $N_t$  is the number of unique tags. Given that  $w_n$  is from topic  $z_n$  and has tag  $t_n$ , the word probability is

$$P(w_n = v | z_n, t_n, \tau, \pi) \propto \exp(\tau_{z_n, v} + \pi_{t_n, v}). \quad (1)$$

The factored representation has only  $(k + N_t) \times V$  parameters.

Given the set of  $N$  tags  $\mathbf{t}$  and parameters  $\alpha, \tau, \pi$ , the joint distribution of a topic mixture  $\theta$ , a set of  $N$  topics  $\mathbf{z}$ , and a set of  $N$  words  $\mathbf{w}$  is given by:

$$p(\theta, \mathbf{z}, \mathbf{w} | \mathbf{t}, \alpha, \tau, \pi) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, t_n, \tau, \pi). \quad (2)$$

The marginal probability of a document is

$$p(\mathbf{w} | \mathbf{t}, \alpha, \tau, \pi) = \int p(\theta | \alpha) \left( \prod_{n=1}^N \sum_{z_n=1}^k p(z_n | \theta) p(w_n | z_n, t_n, \tau, \pi) \right) d\theta. \quad (3)$$

The marginal probability of a corpus  $D$  with  $M$  documents is

$$p(D | \mathbf{t}_1, \dots, \mathbf{t}_M, \alpha, \tau, \pi) = \prod_{d=1}^M \int p(\theta_d | \alpha) \left( \prod_{n=1}^{N_d} \sum_{z_{dn}=1}^k p(z_{dn} | \theta_d) p(w_{dn} | z_{dn}, t_{dn}, \tau, \pi) \right) d\theta_d. \quad (4)$$

### 3 Variational Inference and Parameter Learning

The inference problem is to compute the posterior distribution of hidden variables  $\theta, \mathbf{z}$  given a *single* document and its tags:

$$p(\theta, \mathbf{z} | \mathbf{w}, \mathbf{t}, \alpha, \tau, \pi). \quad (5)$$

Unfortunately this distribution is intractable just as in standard LDA. Like [1] we use variational inference to approximate the above posterior. We first lower bound the document marginal log likelihood with Jensen's inequality using an auxiliary distribution  $q(\theta, \mathbf{z} | \gamma, \phi)$ :

$$\log p(\mathbf{w} | \mathbf{t}, \alpha, \tau, \pi) \quad (6)$$

$$= \log \int_{\theta} \sum_{\mathbf{z}} p(\mathbf{w}, \theta, \mathbf{z} | \mathbf{t}, \alpha, \tau, \pi) d\theta \quad (7)$$

$$= \log \int_{\theta} \sum_{\mathbf{z}} \frac{q(\theta, \mathbf{z} | \gamma, \phi) p(\mathbf{w}, \theta, \mathbf{z} | \mathbf{t}, \alpha, \tau, \pi)}{q(\theta, \mathbf{z} | \gamma, \phi)} d\theta \quad (8)$$

$$\geq \int_{\theta} \sum_{\mathbf{z}} q(\theta, \mathbf{z} | \gamma, \phi) (\log p(\mathbf{w}, \theta, \mathbf{z} | \mathbf{t}, \alpha, \tau, \pi) - \log q(\theta, \mathbf{z} | \gamma, \phi)) d\theta \quad (9)$$

$$\equiv L(\gamma, \phi; \alpha, \tau, \pi). \quad (10)$$

We choose a particular form for the auxiliary distribution

$$q(\theta, \mathbf{z} | \gamma, \phi) = q(\theta | \gamma) \prod_{n=1}^N q(z_n | \phi_n), \quad (11)$$

where  $\gamma$  is a Dirichlet parameter vector of length  $k$ , and  $\phi$  is a  $N \times k$  matrix whose rows are topic multinomials.

#### 3.1 The variational distribution

The lower bound (10) can be written as

$$L(\gamma, \phi; \alpha, \tau, \pi) \quad (12)$$

$$= E_q[\log p(\mathbf{w}, \theta, \mathbf{z} | \mathbf{t}, \alpha, \tau, \pi)] - E_q[\log q(\theta, \mathbf{z} | \gamma, \phi)] \quad (13)$$

$$= E_q[\log p(\theta | \alpha)] + E_q[\log p(\mathbf{z} | \theta)] + E_q[\log p(\mathbf{w} | \mathbf{z}, \tau, \pi)] \\ - E_q[\log q(\theta)] - E_q[\log q(\mathbf{z})], \quad (14)$$

which is the same as eq (14) in [1] except the third term. The third term is:

$$\mathbb{E}_q [\log p(\mathbf{w}|\mathbf{z}, \tau, \pi)] \quad (15)$$

$$= \mathbb{E}_q \left[ \log \prod_{n=1}^N p(w_n|z_n, \tau_{z_n}, \pi_{t_n}) \right] \quad (16)$$

$$= \mathbb{E}_q \left[ \sum_{n=1}^N \log \frac{\exp(\tau_{z_n, w_n} + \pi_{t_n, w_n})}{\sum_{v=1}^V \exp(\tau_{z_n, v} + \pi_{t_n, v})} \right] \quad (17)$$

$$= \mathbb{E}_q \left[ \sum_{n=1}^N \left( \tau_{z_n, w_n} + \pi_{t_n, w_n} - \log \sum_{v=1}^V \exp(\tau_{z_n, v} + \pi_{t_n, v}) \right) \right] \quad (18)$$

$$= \sum_{n=1}^N \mathbb{E}_q [\tau_{z_n, w_n} + \pi_{t_n, w_n}] - \sum_{n=1}^N \mathbb{E}_q \left[ \log \sum_{v=1}^V \exp(\tau_{z_n, v} + \pi_{t_n, v}) \right]. \quad (19)$$

Because of the log-sum-exp in the second term of (19), parameter learning for  $\tau, \pi$  is difficult. Following the technique used in [3], we upper bound the second term of (19) with  $N$  more variational parameters  $\zeta_n$ . We make use of the inequality

$$\log(x) \leq \zeta^{-1}x + \log(\zeta) - 1, \forall \zeta > 0, \quad (20)$$

which gives

$$\sum_{n=1}^N \mathbb{E}_q \left[ \log \sum_{v=1}^V \exp(\tau_{z_n, v} + \pi_{t_n, v}) \right] \quad (21)$$

$$\leq \sum_{n=1}^N \mathbb{E}_q \left[ \zeta_n^{-1} \left( \sum_{v=1}^V \exp(\tau_{z_n, v} + \pi_{t_n, v}) \right) + \log \zeta_n - 1 \right] \quad (22)$$

$$= \sum_{n=1}^N \left[ \zeta_n^{-1} \left( \sum_{v=1}^V \mathbb{E}_{q(z_n)} [\exp(\tau_{z_n, v} + \pi_{t_n, v})] \right) + \log \zeta_n - 1 \right] \quad (23)$$

$$= \sum_{n=1}^N \left[ \zeta_n^{-1} \left( \sum_{v=1}^V \sum_{i=1}^k \phi_{ni} \exp(\tau_{i, v} + \pi_{t_n, v}) \right) + \log \zeta_n - 1 \right]. \quad (24)$$

Putting everything together, we obtain a lower bound  $L2$  on the original lower bound  $L$ :

$$L(\gamma, \phi; \alpha, \tau, \pi) \tag{25}$$

$$\begin{aligned} &= \mathbb{E}_q[\log p(\theta|\alpha)] + \mathbb{E}_q[\log p(\mathbf{z}|\theta)] \\ &\quad + \mathbb{E}_q[\log p(\mathbf{w}|\mathbf{z}, \tau, \pi)] \\ &\quad - \mathbb{E}_q[\log q(\theta)] - \mathbb{E}_q[\log q(\mathbf{z})] \end{aligned} \tag{26}$$

$$\begin{aligned} &\geq \mathbb{E}_q[\log p(\theta|\alpha)] + \mathbb{E}_q[\log p(\mathbf{z}|\theta)] \\ &\quad + \sum_{n=1}^N \mathbb{E}_q[\tau_{z_n, w_n} + \pi_{t_n, w_n}] - \sum_{n=1}^N \left[ \zeta_n^{-1} \left( \sum_{v=1}^V \sum_{i=1}^k \phi_{ni} \exp(\tau_{i,v} + \pi_{t_n, v}) \right) + \log \zeta_n - 1 \right] \\ &\quad - \mathbb{E}_q[\log q(\theta)] - \mathbb{E}_q[\log q(\mathbf{z})] \end{aligned} \tag{27}$$

$$\begin{aligned} &= \log \Gamma\left(\sum_{j=1}^k \alpha_j\right) - \sum_{i=1}^k \log \Gamma(\alpha_i) + \sum_{i=1}^k (\alpha_i - 1)(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)) \\ &\quad + \sum_{n=1}^N \sum_{i=1}^k \phi_{ni} (\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)) \\ &\quad + \sum_{n=1}^N \sum_{i=1}^k \phi_{ni} (\tau_{i, w_n} + \pi_{t_n, w_n}) - \sum_{n=1}^N \left[ \zeta_n^{-1} \left( \sum_{v=1}^V \sum_{i=1}^k \phi_{ni} \exp(\tau_{i,v} + \pi_{t_n, v}) \right) + \log \zeta_n - 1 \right] \\ &\quad - \log \Gamma\left(\sum_{j=1}^k \gamma_j\right) + \sum_{i=1}^k \log \Gamma(\gamma_i) - \sum_{i=1}^k (\gamma_i - 1)(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)) \\ &\quad - \sum_{n=1}^N \sum_{i=1}^k \phi_{ni} \log \phi_{ni} \end{aligned} \tag{28}$$

$$\equiv L2(\gamma, \phi, \zeta; \alpha, \tau, \pi). \tag{29}$$

The lower bound  $L2$  is a function of the variational parameters  $\gamma, \phi, \zeta$  – one finds the optimal  $\gamma, \phi, \zeta$  to maximize it. The combined lower bound  $L2$  on the whole corpus  $D$  is then viewed as a function of the model parameters  $\alpha, \tau, \pi$ , which are optimized holding the variational parameters fixed. Variational parameter learning and model parameter learning proceed alternatively to improve  $L2$ .

We use the variational distribution  $q(\theta, \mathbf{z}|\gamma, \phi)$  to approximate the true posterior distribution  $p(\theta, \mathbf{z}|\mathbf{w}, \mathbf{t}, \tau, \pi)$  for inference. Notice the variational distribution is w.r.t. the maximizing variational parameters  $\gamma, \phi, \zeta$  explicitly, and model parameters  $\alpha, \tau, \pi$  implicitly.

### 3.2 Variational parameter Learning

We maximize  $L2$  w.r.t. the variational parameters  $\gamma, \phi, \zeta$  by coordinate ascend. First we maximize  $L2$  with respect to  $\zeta$ :

$$\frac{\partial L2}{\partial \zeta_n} = \zeta_n^{-2} \left( \sum_{v=1}^V \sum_{i=1}^k \phi_{ni} \exp(\tau_{i,v} + \pi_{t_n, v}) \right) - \zeta_n^{-1}. \tag{30}$$



Setting it to zero and we find

$$\hat{\zeta}_n = \sum_{v=1}^V \sum_{i=1}^k \phi_{ni} \exp(\tau_{i,v} + \pi_{t_n,v}), \quad (31)$$

which is the maximum by verifying the second derivative.

Second we maximize  $L2$  with respect to  $\phi$ . With the constraint  $\sum_{j=1}^k \phi_{nj} = 1$  we form the Lagrangian, and take the derivative:

$$\begin{aligned} & \frac{\partial L2 + \lambda \left( \left( \sum_{j=1}^k \phi_{nj} \right) - 1 \right)}{\partial \phi_{ni}} \\ &= \Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right) \\ & \quad + (\tau_{i,w_n} + \pi_{t_n,w_n}) - \sum_{v=1}^V \zeta_n^{-1} \exp(\tau_{i,v} + \pi_{t_n,v}) \\ & \quad - \log \phi_{ni} - 1 \\ & \quad + \lambda. \end{aligned} \quad (32)$$

Setting it to zero, the maximizing value is

$$\hat{\phi}_{ni} \propto \exp \left( \Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right) + (\tau_{i,w_n} + \pi_{t_n,w_n}) - \sum_{v=1}^V \zeta_n^{-1} \exp(\tau_{i,v} + \pi_{t_n,v}) \right). \quad (34)$$

Finally we maximize  $L2$  with respect to  $\gamma$ . It can be shown the maximum is at

$$\hat{\gamma}_i = \alpha_i + \sum_{n=1}^N \phi_{ni} \quad (35)$$

Notice the maximizing values of  $\zeta$ ,  $\phi$ ,  $\gamma$  depend on each other. Therefore we need to iteratively optimize the three until  $L2$  converges.

### 3.3 Model parameter Learning

Fixing the variational parameters  $\zeta$ ,  $\phi$ ,  $\gamma$ , the variational marginal likelihood of the corpus  $D$  as a function of  $\alpha$ ,  $\tau$ ,  $\pi$  is  $L2_D = \sum_{d=1}^M L2_d$ .

First the maximizing model parameter  $\alpha$  can be found with the same linear-time Newton-Raphson algorithm in [1] A.4.2.

Then we maximize  $L2_D$  with respect to  $\tau$ . The relevant terms in  $L_D$  are:

$$\begin{aligned} & L2_{D[\tau,\pi]} \\ &= \sum_{d=1}^M \sum_{n=1}^{N_d} \sum_{i=1}^k \phi_{dn,i} (\tau_{i,w_{dn}} + \pi_{t_{dn},w_{dn}}) \\ & \quad - \sum_{d=1}^M \sum_{n=1}^{N_d} \left[ \zeta_{dn}^{-1} \left( \sum_{v=1}^V \sum_{i=1}^k \phi_{dn,i} \exp(\tau_{i,v} + \pi_{t_{dn},v}) \right) + \log \zeta_{dn} - 1 \right]. \end{aligned} \quad (36)$$

The derivative with respect to  $\tau_{i,v}$  is:

$$\frac{\partial L2_D}{\partial \tau_{i,v}} = \sum_{d=1}^M \sum_{n=1}^{N_d} \phi_{dn,i} \delta(w_{dn}, v) - \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \zeta_{dn}^{-1} \phi_{dn,i} \exp(\pi_{t_{dn},v}) \right) \exp(\tau_{i,v}), \quad (37)$$

where  $\delta(x, x') = 1$  if  $x = x'$ , and 0 otherwise. For  $v = 1 \dots V$ , setting the derivative to zero we find

$$\tau_{i,v} = \log \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \phi_{dn,i} \delta(w_{dn}, v) \right) - \log \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \zeta_{dn}^{-1} \phi_{dn,i} \exp(\pi_{t_{dn},v}) \right). \quad (38)$$

Finally we maximize  $L2_D$  with respect to  $\pi$ . The derivative with respect to  $\pi_{t,v}$  is:

$$\frac{\partial L2_D}{\partial \pi_{t,v}} = \sum_{d=1}^M \sum_{n=1}^{N_d} \sum_{i=1}^k \phi_{dn,i} \delta(t_{dn}, t) \delta(w_{dn}, v) - \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \sum_{i=1}^k \zeta_{dn}^{-1} \phi_{dn,i} \exp(\tau_{i,v}) \delta(t_{dn}, t) \right) \exp(\pi_{t,v}). \quad (39)$$

For  $v = 1 \dots V$ , setting the derivative to zero we find

$$\pi_{t,v} = \log \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \sum_{i=1}^k \phi_{dn,i} \delta(t_{dn}, t) \delta(w_{dn}, v) \right) - \log \left( \sum_{d=1}^M \sum_{n=1}^{N_d} \sum_{i=1}^k \zeta_{dn}^{-1} \phi_{dn,i} \exp(\tau_{i,v}) \delta(t_{dn}, t) \right). \quad (40)$$

Notice  $\tau$  appears in  $\pi$ 's maximum solution and vice versa. Therefore we iterate (38) and (40) until  $\tau$  and  $\pi$  converge, which happens quickly in practice.

## 4 A Toy Example

To illustrate the benefit of tagLDA, we create a toy example as follows. On a vocabulary of 9 words, we specify 3 topic parameters  $\tau$  which are the upper three rows in Figure 2. We also specify 3 tag parameters  $\pi$  which are the lower three rows in the same figure. These parameters are smoothed and do not contain zero.

With these parameters, we generate a corpus of 300 documents. All documents are 40 words long. For each word position, a tag is chosen with probability 0.6, 0.3, 0.1 for tags 1, 2, 3 respectively. All words in a document share the same topic, which is chosen uniformly from topics 1, 2, 3. Given tag  $t$  and topic  $i$ , a word is generated according to the multinomial proportional to  $\exp(\tau_i + \pi_t)$ . Since there are three topics and three tags, nine word multinomials are possible; They are plotted in Figure 3.

For tagLDA, the input is the corpus and the tags. That is, for each word in the corpus we give the corresponding tag (1, 2, or 3) as domain knowledge to tagLDA. Therefore there are three tag parameters  $\pi$  to learn. We ask tagLDA to learn three topics parameters  $\tau$ . tagLDA optimizes the lower bound  $L2$  (29), which converges to -20816. We plot the learned parameters  $\tau$  and  $\pi$  in Figure 4. tagLDA is able to approximate the intended factored representation. With the learned parameters  $\tau$  and  $\pi$ , the word posterior given a topic  $i$  and tag  $t$  in tagLDA is the multinomial proportional

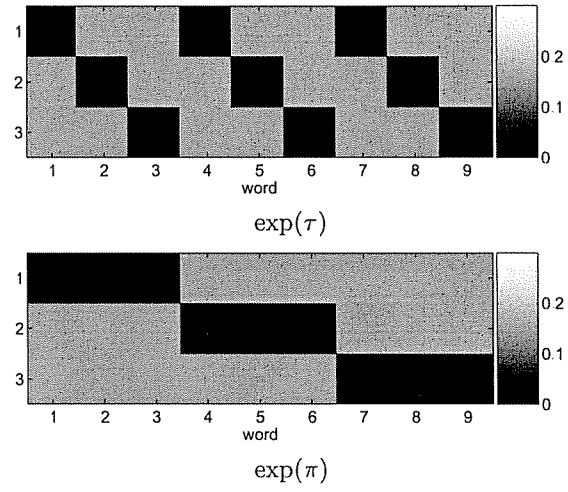


Figure 2: The original parameters  $\tau$ ,  $\pi$  used to generate the toy corpus

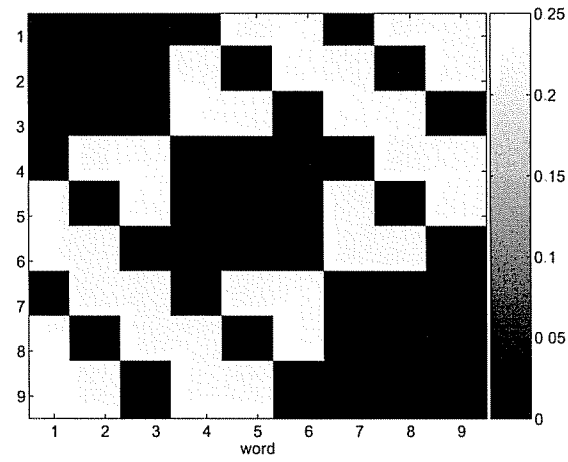


Figure 3: The nine word multinomials out of the combination of  $\tau$  and  $\pi$

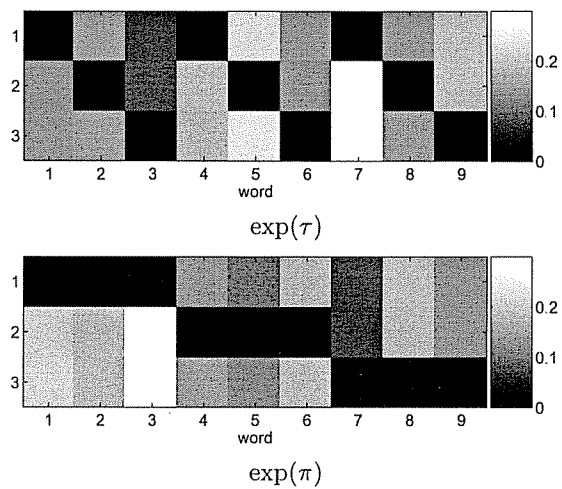


Figure 4: The parameters  $\tau$ ,  $\pi$  learned by tagLDA

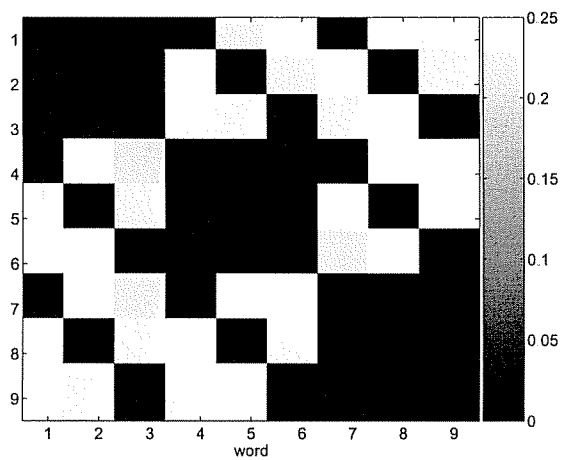


Figure 5: The nine multinomial distributions of tagLDA.

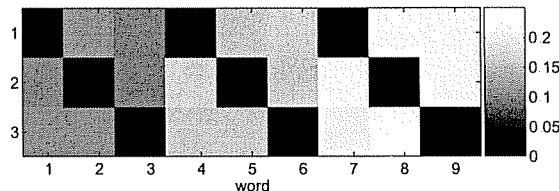


Figure 6: Topics learned by standard LDA

to  $\exp(\tau_i + \pi_i)$ . We show the nine multinomials in Figure 5, which are similar to Figure 3 as they should be.

We ran standard LDA on the toy example for comparison. We ask LDA to learn three topics too. LDA optimizes a variational lower bound on the log likelihood, which converges to -23414. The log likelihood bound is worse than tagLDA. We plot the topics learned by LDA in Figure 6. The topic probabilities are not uniform as those in the upper panel of Figure 2. This is because LDA attempts to explain the asymmetry introduced by uneven tag distributions.

## 5 A Small Text Corpus

We selected 45 documents from AP news<sup>1</sup> to form a small corpus. 15 of the documents are about politics, 15 about finance, and the remaining 15 about war. We converted all words into lower case, used a stop list of 300 words, and a frequency cutoff of 5 on the whole AP data, to obtain a vocabulary of about 10,000 words. No other preprocessing was carried out. We then ran a link parser on the documents. For this experiment, the parser was used as a part-of-speech tagger. Each word is tagged as ‘noun’ (n), ‘verb’ (v) or ‘other’ (o).

Given these three kinds of tags we ran tagLDA on the 45 documents, asking for three topics. The lower bound on log likelihood,  $L2$ , converges to -43916. tagLDA learned topic parameters  $\tau_1, \tau_2, \tau_3$  and tag parameters  $\pi_n, \pi_v, \pi_o$ , each is a vocabulary-sized vector. Given topic  $t$  and tag  $p$  the word probability is  $P(w|\tau_t, \pi_p) \propto \exp(\tau_{tw} + \pi_{pw})$ . We show the top 10 words with the largest probabilities in each  $\tau, \pi$  combination in Figure 7. Not surprisingly, tagLDA learned to separate nouns, verbs and other words according to the tag.

As a comparison we also ran standard LDA on the 45 documents, asking for three topics. The lower bound on log likelihood converges to -51264. We show the top 10 words in each topic in Figure 8. The distinction of noun, verb, other is not present since this information is not available to standard LDA. We could have trained a three-topic LDA model separately for noun, verb and other words. This would achieve similar word distributions as tagLDA, but the number of parameters would be nine for LDA, while tagLDA uses only six parameters.

<sup>1</sup>The same AP corpus in the LDA distribution at <http://www.cs.berkeley.edu/~blei/lda-c/ap.tgz>

tag=n, topic=politics		tag=n, topic=finance		tag=n, topic=war	
0.020710	campaign	0.054672	market	0.018956	army
0.015500	state	0.035832	index	0.015796	hostages
0.015302	poll	0.032898	trading	0.014217	government
0.014452	convention	0.027564	prices	0.014217	gunmen
0.014118	support	0.026186	volume	0.009478	attack
0.013945	primary	0.026184	stock	0.009478	soldiers
0.011902	sen	0.024808	shares	0.008760	forces
0.011052	delegates	0.022024	stocks	0.008688	miles
0.010925	president	0.017917	session	0.008688	troops
0.010201	voters	0.017917	average	0.008688	israeli
tag=v, topic=politics		tag=v, topic=finance		tag=v, topic=war	
0.027641	going	0.051953	rose	0.048936	killed
0.022763	think	0.033524	came	0.016312	freed
0.014633	vote	0.031971	outnumbered	0.014022	told
0.011381	asked	0.027975	fell	0.011845	fighting
0.011381	brokered	0.023978	listed	0.011263	shot
0.010295	made	0.017555	reported	0.010875	opened
0.009755	see	0.015986	totaled	0.010875	died
0.009755	got	0.015986	traded	0.010875	wounded
0.009755	campaigning	0.011989	go	0.009062	go
0.008650	added	0.011989	led	0.009062	kidnapped
tag=o, topic=politics		tag=o, topic=finance		tag=o, topic=war	
0.030569	i	0.034266	million	0.025995	police
0.029899	dukakis	0.029793	stock	0.019496	two
0.023104	south	0.026817	dow	0.018568	people
0.021745	jackson	0.026788	exchange	0.017639	south
0.020386	percent	0.022347	unchanged	0.013926	red
0.018133	bush	0.019368	jones	0.011141	thursday
0.017667	gephardt	0.017878	big	0.010212	new
0.016988	dole	0.017873	nyse	0.010212	15
0.016308	democratic	0.016658	new	0.008798	military
0.016185	new	0.016443	wall	0.008355	i

Figure 7: The top 10 words and their probabilities in each topic / tag combination learned by tagLDA on the small AP corpus.

topic=politics		topic=finance		topic=war	
0.013957	i	0.024101	market	0.009347	police
0.013922	dukakis	0.023651	stock	0.009347	army
0.012024	bush	0.015768	index	0.009013	killed
0.010758	south	0.014531	trading	0.007891	two
0.010441	dole	0.013948	million	0.006677	people
0.010045	jackson	0.012129	prices	0.006677	hostages
0.009492	percent	0.012126	exchange	0.006343	south
0.009492	primary	0.011523	volume	0.006009	government
0.009475	gore	0.010916	shares	0.006009	gunmen
0.008752	campaign	0.010916	dow	0.005341	bank

Figure 8: The top 10 words learned by standard LDA on the small AP corpus.

## 6 AP: A Larger Text Corpus

We ran tagLDA on 2243 AP news articles <sup>2</sup>. Again each word is tagged as noun, verb or other. The corpus has 434979 words. The vocabulary size is 10174. We asked for 50 topics. tagLDA converged in 25 iterations with log likelihood bound -2799598. We show the top 10 words from selected topics learned with tagLDA in Figure 9.

We also ran standard LDA under the same settings. LDA converged in 30 iterations with log likelihood bound -3158008. Figure 10 shows the top 10 words in the corresponding topics.

## 7 The WebKB Corpus

For the WebKB corpus <sup>3</sup> we used two tags: b='body text' for words in the body text of an html page, and a='anchor text' for words in a hyperlink. The corpus has 8099 html pages. Each page is treated as a document. We kept words that consists of more than one letter, converted them into lower case, and used a stopwords list. The processed corpus has 1.3 million words. The vocabulary size is 9898 with a frequency cutoff. We ran tagLDA with 30 topics. tagLDA converged in 30 iterations, with a lower bound of -9674409 on log likelihood.

In Figure 11 we show the top 10 words in six selected topics, separately with the two tags. For the anchor text tag, tagLDA learns the words that frequently appear on hyperlinks, such as *postscript*, *publications*, *resume*, *tar.gz*, *slide*, *solution*, etc.

<sup>2</sup>The same AP corpus in the LDA distribution at <http://www.cs.berkeley.edu/~blei/lda-c/ap.tgz>

<sup>3</sup><http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/webkb-data.gtar.gz>

tag=o, topic=health		tag=n, topic=health		tag=v, topic=health	
0.016932	fda	0.017016	patients	0.011801	say
0.016657	medical	0.015987	drug	0.010072	think
0.016476	dr	0.011704	heart	0.009529	done
0.015216	new	0.010520	health	0.009292	approved
0.010832	i	0.009013	research	0.008741	treated
0.009916	people	0.008909	children	0.008541	says
0.009315	last	0.008291	researchers	0.008359	tested
0.008834	federal	0.007752	company	0.007660	take
0.008361	two	0.007538	officials	0.006936	produce
0.007884	year	0.007335	aspirin	0.006456	make
tag=o, topic=space		tag=n, topic=space		tag=v, topic=space	
0.025630	nasa	0.033668	space	0.018064	made
0.017886	earth	0.030260	shuttle	0.017410	make
0.014641	two	0.015557	mission	0.014456	launch
0.014353	venus	0.012612	launch	0.011008	launched
0.013671	spacecraft	0.010747	time	0.009746	planned
0.011961	magellan	0.009480	rocket	0.009159	take
0.010080	first	0.008723	planet	0.008764	go
0.009874	mars	0.008507	telescope	0.008667	released
0.009604	space	0.007826	astronauts	0.008078	scheduled
0.009204	soviet	0.007532	system	0.008002	manned
tag=o, topic=iraq		tag=n, topic=iraq		tag=v, topic=iraq	
0.053580	iraq	0.034619	iraqi	0.014466	asked
0.034202	kuwait	0.021629	troops	0.012714	sent
0.028902	military	0.017273	american	0.011458	go
0.024496	saudi	0.015365	war	0.011372	made
0.020978	united	0.013507	forces	0.010901	held
0.018748	gulf	0.012611	president	0.008841	say
0.018244	arabia	0.011997	invasion	0.008632	invaded
0.016660	bush	0.011673	oil	0.008474	leave
0.015744	saddam	0.011476	defense	0.008038	told
0.014341	persian	0.011141	officials	0.007746	take

Figure 9: The top 10 words and their probabilities in selected topic / tag combinations learned by tagLDA on the large AP corpus.



topic=health		topic=iraq		topic=space	
0.008081	drug	0.024015	iraq	0.016574	space
0.007592	patients	0.015315	kuwait	0.011573	shuttle
0.006566	health	0.014820	iraqi	0.009906	nasa
0.005777	fda	0.014201	military	0.008528	earth
0.005624	heart	0.012394	saudi	0.007515	launch
0.005415	dr	0.011004	gulf	0.006466	mission
0.005250	medical	0.009460	united	0.006079	two
0.005168	new	0.009104	war	0.005547	venus
0.005025	research	0.008582	arabia	0.005283	spacecraft
0.004526	children	0.008456	bush	0.005123	time

Figure 10: The top 10 words (and their probabilities) in selected topics learned by standard LDA on the large AP corpus.

## 8 The NIPS Corpus

We manually tagged 1602 papers from Neural Information Processing Systems (NIPS) vol. 0 – 12<sup>4</sup>. We tagged each word with the section it is in. There are seven tag types:

- h header, anything before abstract: title, authors, address
- a the abstract
- i the introduction section of the paper
- b main body of the paper
- c the conclusion / discussion / summary section of the paper
- k the acknowledgments
- r the references

We kept words that consists of more than one letters, converted them into lower case, and used a stopword list. The processed corpus has 2.3 million words. The vocabulary size is 8100 with a frequency cutoff. We ran tagLDA with 30 topics. tagLDA converged in 20 iterations with a lower bound of -16099655 on log likelihood. In Figure 12 we show the top 10 words in three selected topics, with all tag combinations.

It is interesting to see the interaction between the 30 topic parameter vectors  $\tau$  and 7 tag parameter vectors  $\pi$  in the factored representation. First of all, some tags seem to be dominated by the topic parameter. This effect is especially prominent for tags a, i, b, c. Intuitively it is what one would expect, as the abstract, introduction, body and conclusion sections are about the same topic. Meanwhile some tags have its distinct word distribution that overwhelms the topic parameter, most noticeably tag k. All topics with tag k exhibits common acknowledgment words like ‘grant, supported, thank’. Finally the tags h and r are in the middle: their word distribution is roughly a half-half combination of the topic and tag parameters. Tag h contains both topic-specific words ‘neural, networks, speech, recognition’ from the titles, and tag-specific words ‘university, department’ from the affiliations. Tag r contains similar topic-specific words, and tag-specific words ‘journal, press, proc, vol’. We conclude that the factored represen-

<sup>4</sup>The un-tagged corpus comes from Sam Roweis at <http://www.cs.toronto.edu/~roweis/data.html>

tag=b, topic 1		tag=a, topic 1		tag=b, topic 2		tag=a, topic 2	
0.027352	pp	0.043408	postscript	0.042366	file	0.360304	ps
0.025519	proceedings	0.023498	acm	0.033877	ps	0.031480	ftp
0.025408	conference	0.016093	ieee	0.032590	hello	0.028412	gz
0.022164	ieee	0.014476	real	0.028990	files	0.027834	tar
0.019816	acm	0.014272	report	0.020195	ftp	0.027094	pub
0.016230	international	0.013964	publications	0.019446	directory	0.024495	file
0.014353	symposium	0.013829	conference	0.011910	window	0.022174	directory
0.014010	vol	0.013779	computing	0.011171	public	0.018556	readme
0.013411	time	0.012498	abstract	0.010971	seed	0.017426	latex
0.011079	th	0.012086	technical	0.010481	version	0.015414	tex
tag=b, topic 3		tag=a, topic 3		tag=b, topic 4		tag=a, topic 4	
0.038483	home	0.090302	cs	0.044209	lecture	0.110431	slide
0.034131	am	0.065067	edu	0.018009	logic	0.105939	lecture
0.028080	cs	0.059314	home	0.016922	tree	0.025189	format
0.020557	interests	0.014164	resume	0.014621	data	0.024708	notes
0.018242	student	0.011802	columbia	0.013322	structures	0.024291	slides
0.017917	edu	0.011165	student	0.013154	notes	0.022500	gif
0.016436	phone	0.010680	homepage	0.011297	trees	0.016721	logic
0.014113	fax	0.010626	graduate	0.010682	chapter	0.015202	cs
0.012872	office	0.009934	new	0.009741	format	0.014314	postscript
0.012539	last	0.009767	publications	0.009087	functions	0.012711	tree
tag=b, topic 5		tag=a, topic 5		tag=b, topic 6		tag=a, topic 6	
0.030285	due	0.080319	postscript	0.047212	programming	0.050997	programming
0.026808	program	0.071853	assignment	0.024635	languages	0.022725	languages
0.026076	assignment	0.071069	homework	0.019574	language	0.020991	language
0.015785	homework	0.032510	solution	0.015835	design	0.014813	software
0.011901	problem	0.026703	program	0.015237	software	0.013375	program
0.011669	code	0.025780	solutions	0.012700	program	0.012944	design
0.010428	project	0.023100	cs	0.010638	compiler	0.012457	ece
0.009513	class	0.018283	project	0.010404	programs	0.011931	compiler
0.009420	file	0.016141	due	0.010088	object	0.011874	object
0.009191	assignments	0.014712	problem	0.009591	oriented	0.011028	cs

Figure 11: The top 10 words (and their probabilities) in six selected topics, with b='body text' and a='anchor text' tags respectively, learned by tagLDA on WebKB corpus.

tation is appropriate for the corpus.

## 9 Perplexity

Both tagLDA and LDA models are trained unsupervised. The natural measure of performance is the log likelihood on some unseen held-out corpus. Equivalently we compute the *perplexity* on the held-out set, a conventional measure in language modeling. The perplexity is a monotonically decreasing function of the log likelihood, and can be understood as the predicted number of equally likely words for a word position on average. For a held-out set  $D$  of  $M$  documents the perplexity is defined as

$$\text{perplexity}(D) = \exp \left( -\frac{\sum_{d=1}^M \log p(\mathbf{w}_d | \mathbf{t}_d)}{\sum_{d=1}^M N_d} \right), \quad (41)$$

where  $N_d$  is the number of words in document  $d$ . Note (41) computes the conditional perplexity given the tags  $\mathbf{t}_d$ . For LDA it reverts to  $p(\mathbf{w}_d | \mathbf{t}_d) = p(\mathbf{w}_d)$  since the tag information is not used. Also note we can only compute an upper bound of the perplexity, because we lower-bound the log likelihood with variational methods for both tagLDA and LDA.

We compute and compare the perplexity of tagLDA and LDA on the AP, WebKB and NIPS corpora with aforementioned tags respectively. For each corpus we run 5 trials in which we randomly hold out 20% of the document for test, and train on the remaining 80%. We train tagLDA and LDA with the same stopping criteria. The number of topics in both models is systematically varied.

We plot the average perplexity on held-out data in Figure 13. By utilizing the tag information tagLDA has lower perplexity on all three corpora. That is, tagLDA is a better model by assigning higher likelihood to held-out documents than LDA. We also note that the reduction in perplexity strongly depends on the type of tags. Intuitively when different tags tend to mark distinctive words, perplexity reduction is large. For example the part-of-speech tags in the AP corpus mark nouns, verbs etc., and tagLDA has a 50% perplexity reduction. On the other hand when different tags mark similar words the perplexity reduction is small, as in the anchor-text / body-text tags for WebKB corpus.

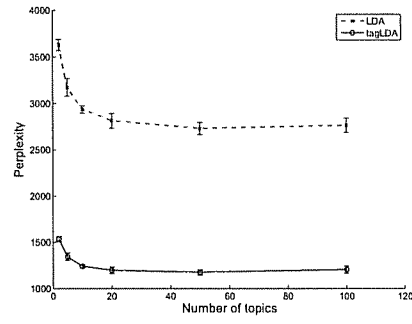
## 10 Document Classification

The ‘topics’ learned by tagLDA and LDA can be viewed as a concise representation of the original document. Here the topics are represented by the variational posterior vector  $P(\gamma | \mathbf{w})$ . One can perform document classification in a discriminative framework, by using  $\gamma$  as feature vectors in a support vector machine (SVM). It should be emphasized that both tagLDA and LDA are trained with unsupervised learning, and are therefore not geared towards classification.

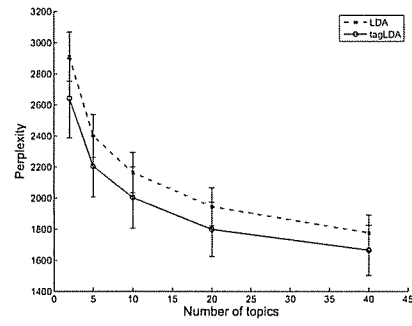
We are interested in whether the topics learned with tagLDA or LDA perform differently in classification. It is important to note that the tag information is what makes

tag=h	a	i	b	c	k	r
university	eye	eye	eye	model	supported	neural
institute	model	head	model	system	grant	eye
department	system	model	head	cells	acknowledgments	visual
edu	abstract	system	system	head	research	journal
salk	head	cells	figure	eye	acknowledgements	brain
electrical	visual	visual	cells	direction	thank	press
science	cells	motor	direction	visual	foundation	motor
brain	motor	direction	position	motor	project	neuroscience
neurobiology	vor	neurons	velocity	neurons	work	movements
ca	neurons	sensory	map	target	eye	auditory
university	abstract	function	function	functions	grant	neural
department	networks	functions	theorem	networks	supported	networks
neural	functions	networks	let	function	research	pp
networks	neural	neural	functions	bounds	acknowledgement	theory
edu	function	number	bound	theorem	work	ieee
engineering	network	network	threshold	bound	gratefully	computation
science	bounds	bounds	number	results	afosr	proc
computer	show	paper	proof	number	thank	vol
ca	bound	class	case	neural	nsf	bounds
australia	paper	threshold	size	complexity	discussions	functions
recognition	recognition	recognition	training	recognition	darpa	recognition
speech	speech	speech	hmm	speech	work	speech
university	system	hmm	word	hmm	acknowledgements	neural
neural	hmm	system	system	system	arpa	morgan
street	network	training	recognition	training	research	ieee
department	abstract	models	network	context	thank	systems
systems	speaker	context	context	models	acknowledgments	networks
technology	context	network	speech	model	steve	pp
ca	models	neural	probabilities	hybrid	acknowledgement	proc
networks	word	model	model	network	authors	processing

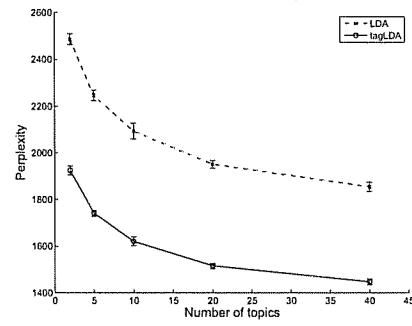
Figure 12: The top 10 words in three selected topics (the top, middle and bottom panel respectively), and with all tag combinations (the columns), learned by tagLDA on NIPS corpus.



(a) the AP corpus



(b) the WebKB corpus



(c) the NIPS corpus

Figure 13: Held-out set perplexity

tagLDA different from LDA. The tags however may or may not have direct correlation with the classification goal.

For these experiments we used the WebKB corpus with 8106 documents. We train a tagLDA and a LDA model respectively, both with 60 topics. The tags for tagLDA are ‘anchor text’ vs. ‘body text’ as mentioned before. Each document therefore is represented by a 60-dimensional  $\gamma$  vector under tagLDA model, and a different 60-dimensional vector under LDA. The classification goals are the set of natural categories of WebKB data, namely whether the document is a *course*, *faculty*, *student*, or *other* web page. For simplicity we create four binary classification tasks of one category vs. the rest. We use the *SVM<sup>light</sup>* software [4] with exactly the same settings for tagLDA and LDA. We vary the proportion of training data from 10% to 70%. Figure 14 compares the classification accuracy. Each point is an average of 30 random trials (the same random split is applied to both tagLDA and LDA). For completeness we also include the accuracy where individual words are used as SVM features.

The tag of anchor/body text is not directly related to any of the classification tasks. Therefore it is interesting to note that in three out of four experiments (a – c), the tagLDA topic representation gives a small improvement over the LDA representation; but on (d) tagLDA is worse than LDA.

## 11 Image Categorization

We use the image dataset from [5], which contains 13 categories of natural scenes and a total of 3859 images. Each image is gray-scale. The average size of an image is about  $250 \times 300$  pixels. Image patches are sampled at an evenly spaced grid of  $10 \times 10$  pixels. The patches have side lengths randomly between 10 to 30 pixels. Each patch is then resized to  $11 \times 11$  pixels. The histogram of each patch is stretched over the whole range.

We randomly split the images into a training set and a test set. Each scene category has 100 training images and the rest are test images. In total there are 1300 training images and 2559 test images.

We learn a codebook by randomly sampling 2% of the patches from a random half of the training set. We run k-means algorithm on the sampled patches to generate  $V$  clusters. We use the Euclidean distance between patches. The  $V$  cluster centers are then used as codewords. In the experiments we used  $V = 200$  and 1600 respectively. With the codebook, all images are converted into a bag-of-patch representation, where each patch is quantized to its closest codeword.

For tagLDA we tag each patch by its vertical position in the image. This is motivated by the intuition that many images show vertical inhomogeneity. Specifically we divide an image evenly into four parts along its height. A patch is tagged by the part its top-left corner falls into.

We note that both LDA and tagLDA are unsupervised learning models. They are used only to generate feature vectors. We separately train a LDA and a tagLDA model on the 1300 training images *without using the labels*. This is different from [5], who use a supervised variant of the LDA model that incorporates the labels. In all experiments we use 40 topics. We then perform inference on the test images. Table 1 lists the test set

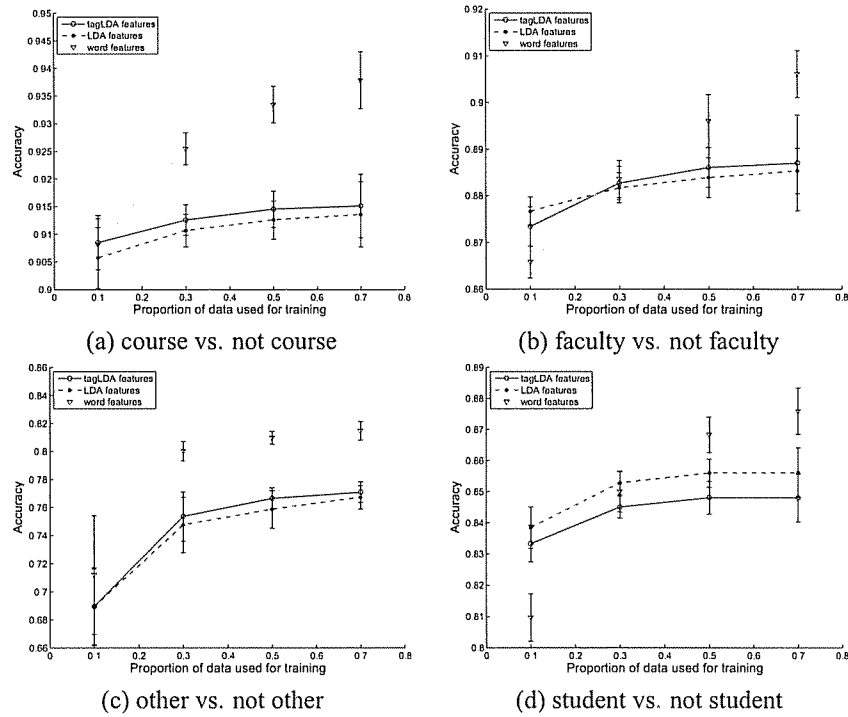


Figure 14: Classification accuracy on WebKB data with different proportions of training data.

codebook $V$	LDA	tagLDA
200	130.0	126.0
1600	575.1	555.5

Table 1: Perplexity on test images for LDA and tagLDA

codebook $V$	kernel	bag-of-patch	LDA $\gamma$	tagLDA $\gamma$
200	linear	58.8%	53.0%	53.0%
	poly3	63.0%	57.3%	55.4%
1600	linear	60.9%	56.2%	56.8%
	poly3	66.1%	58.6%	58.0%

Table 2: Accuracies on test images using  $SVM^{light}$

perplexity. Similar to text tasks, tagLDA gives lower perplexity (i.e. higher predicted probability) on test images than LDA.

We use the inferred posterior variational Dirichlet parameters  $\gamma$  (a 40-vector) as a reduced feature representation for each image. Note the original bag-of-patch representation has  $V = 200$  or 1600 features. Thus the  $\gamma$  representation achieves 80% and 97.5% feature reduction.

For classification, we compare the performance of  $SVM^{light}$  under the three feature representations: bag-of-patch raw counts (200 or 1600 dimensions), LDA  $\gamma$  vector (40 dimensions), and tagLDA  $\gamma$  vector (40 dimensions). We train the SVMs on the training set and test them on the test set with one-vs-all for 13-class classification. For each feature representation, we use a linear kernel and a cubic (poly3) kernel  $K(x, y) = (x^\top y + 1)^3$  respectively. Table 2 lists the test set accuracies. We observe that:

- A larger codebook is better.
- Raw bag-of-patch features are better than the reduced LDA and tagLDA  $\gamma$  features when using SVM. This is consistent with the text classification experiments in the previous section.
- LDA and tagLDA  $\gamma$  features perform similarly for classification. This is disappointing, but also consistent with previous text classification experiments.
- We achieve 66.1% accuracy with 1600 codewords, simple bag-of-patch features, and SVM poly3 kernel. This is comparable to the 64.0% accuracy reported in [5] with a LDA-like theme model. We suspect the fact that SVM is a discriminative model while theme model is generative might have played a role, since it is known that discriminative models tend to yield better classification accuracies.



## 12 Discussions

We introduced tagLDA model, a factored representation that models the structure of a document. In experiments tagLDA is better than LDA in terms of test set perplexity. But there is no significant advantage when tagLDA is used for classification.

TagLDA is only the first step towards incorporating domain knowledge into topic models. There are several future directions:

- Allowing unknown tags for some of the words.
- Allowing multiple tags per word.
- Extending tagLDA to express higher order domain knowledge. The tags describe knowledge on individual words. An obvious extension is to use links that describe knowledge on pairs of words, for example those obtained using a link parser.

## Acknowledgment

We thank Charles Dyer for suggestions on image processing, and Li Fei-Fei for providing the natural scene image dataset.

## References

- [1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [2] T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems (NIPS) 17*, 2004.
- [3] David M. Blei and John D. Lafferty. Correlated topic models. In *Advances in NIPS 18*, 2005.
- [4] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.
- [5] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.