Mini-batch Variational Inference for Time-Aware Topic Modeling

First Author[1] $^{[0000-1111-2222-3333]}$ and Second Author[2] $^{[1111-2222-3333-4444]}$

[1] -

-

[2] -

-

Abstract. This paper proposes a time-aware topic model and its mini-batch variational inference for exploring chronological trends in document contents. Our contribution is twofold. First, to extract topics in a time-aware manner, our method uses two vector embeddings: the embedding of latent topics and that of document timestamps. By combining these two embeddings and applying the softmax function, we have as many word probability distributions as document timestamps for each topic. This modeling enables us to extract remarkable topical trends. Second, to achieve memory efficiency, the variational inference is implemented as a mini-batch maximization of the evidence lower bound. Our inference requires no knowledge of the total number of training documents. This enables us to perform the optimization in a similar manner to neural networks. Actually, our implementation used a deep learning framework. The evaluation results show that we could improve test perplexity by using document timestamps. The results also show that our test perplexity was comparable with that of collapsed Gibbs sampling, which is less efficient in memory usage than our method. This makes our proposal promising for time-aware topic extraction from large data sets.

Keywords: topic models · variational Bayes · time-aware analysis.

1 Introduction

This paper proposes a topic model using document timestamps, which is an extension of latent Dirichlet allocation (LDA) [3], and also provides an inference for the proposed model. Our contribution is twofold. First, we use two vector embeddings in our model: the vector embedding of latent topics and that of document timestamps. Vector embedding enables us to easily incorporate covariates like timestamps in topic modeling. Second, we provide a mini-batch based variational inference for the proposed model. An important merit of mini-batch inference is memory efficiency. These two contributions, explained in the following subsections, provide a viewpoint from which we can regard our time-aware topic modeling as sharing common features with neural networks. We actually implemented the proposed inference by using PyTorch.[3]

[3] `https://github.com/pytorch`

1.1 Vector Embeddings of Topics and Timestamps

The first contribution concerns the modeling. Topic models are a probabilistic model of documents. The following two types of categorical distributions are used in topic modeling: per-document categorical distributions over topics and per-topic categorical distributions over words.

First, each document is modeled with a categorical distribution defined over latent topics. We denote the number of topics by K. Let $\theta_{d,k}$ refer to the probability of the topic k in the document d. Intuitively, $\theta_{d,k}$ for each $k = 1, \ldots, K$ tells the importance of each topic in the document d. Our method introduces no modification with respect to this type of categorical distributions. Therefore, the corresponding posterior parameters are estimated as described in [3].

Second, each latent topic is in turn represented as a categorical distribution defined over vocabulary words. We denote the number of different words, i.e., the vocabulary size, by V and refer to the probability of the word v in the topic k by $\phi_{k,v}$. Intuitively, $\phi_{k,v}$ is the relevancy of the word v to the topic k. In the proposed model, the $\phi_{k,v}$'s for each topic k are obtained by applying the softmax function to the V-dimensional embedding $\boldsymbol{w}_k$ of the topic k plus bias, i.e., $\phi_{k,v} = \exp(w_{k,v} + b_v) / \sum_{v'} \exp(w_{k,v'} + b_{v'})$, where b_v is the bias for the word v and is shared by all topics. Let $\boldsymbol{W}$ be the $V \times K$ matrix whose k-th column is $\boldsymbol{w}_k$. Then $\boldsymbol{\phi}_k = (\phi_{k,1}, \ldots, \phi_{k,V})^\top$ can be written as

$$\boldsymbol{\phi}_k = \mathrm{Softmax}(\boldsymbol{W}\boldsymbol{e}_k + \boldsymbol{b}) \tag{1}$$

where $\boldsymbol{e}_k$ is the one-hot vector whose k-th element is 1 and all other elements are 0. The formula in Eq. (1) is similar to that presented in [7]. However, we here provide a viewpoint from which these per-topic word probabilities $\boldsymbol{\phi}_k$ for $k = 1, \ldots, K$ can be regarded as softmax outputs of a single-layer feedfoward network whose input is always one-hot. We adopt the same approach also for document timestamps. Assume that there are T different timestamps and that $\boldsymbol{u}_t$ is the vector embedding of the timestamp t. We then obtain *time-aware* per-topic word probabilities as

$$\boldsymbol{\phi}_{k,t} = \mathrm{Softmax}(\boldsymbol{W}\boldsymbol{e}_k + \boldsymbol{U}\boldsymbol{o}_t + \boldsymbol{b}) \tag{2}$$

where $\boldsymbol{U}$ is the $V \times T$ matrix whose t-th column is $\boldsymbol{u}_t$, and $\boldsymbol{o}_t$ is the one hot vector whose t-th element is 1 and all other elements are 0. $\boldsymbol{\phi}_{k,t} = (\phi_{k,t,1}, \ldots, \phi_{k,t,V})^\top$ in Eq. (2) represents the word probabilities in the topic k at the time point t (cf. Fig. 1). In the evaluation experiments, we compared the word probability modeling based on Eq. (2) to that based on Eq. (1) in terms of test perplexity. The experiments are expected to show whether document timestamps can improve the quality of extracted topics in terms of perplexity.

1.2 Mini-Batch Variational Bayes

Our second contribution concerns the inference for the proposed model. Mini-batch based variational Bayesian inferences have already been proposed for topic

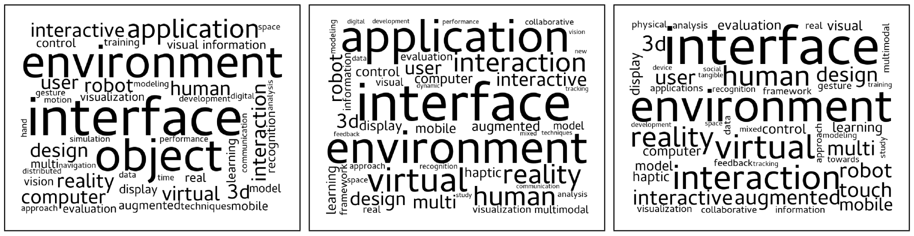

Fig. 1. Word cloud presentation of the same topic at three different time points. This topic, seemingly related to HCI, extracted from 827,141 paper titles in DBLP data set (cf. Section 3). K was set to 50. The words having larger probabilities are shown in larger fonts, because the font size is determined by the value of $\phi_{k,t,v}$. The left, center, and right panels correspond to the time points 1998, 2005, and 2012, respectively.

models [9][4]. However, these inferences require the knowledge of the total number of training documents. Our second contribution is to provide a mini-batch learning for the proposed model, where no knowledge of data size is required.

We perform the parameter estimation for our model in a similar manner to that for neural networks. We iterate over mini-batches, i.e., small subsets of the corpus, and conduct gradient descent where the loss function is the negative of the evidence lower bound (ELBO). Of course, we need to choose an appropriate optimization algorithm like Adam [10] and to adjust its learning rate based on validation set evaluation. However, this is what we usually do when training neural networks. The experimental results show that the test perplexity achieved by the proposed inference was comparable to that achieved by collapsed Gibbs sampling (CGS) for the vanilla LDA [8]. While CGS can give a good test perplexity in many situations [1], it is far less efficient in memory use than mini-batch learning. Since our inference is a mini-batch learning, it can be adopted even when the number of training documents is prohibitively large for CGS.

The rest of the paper is organized as follows. Section 2 describes our proposal in detail. Section 3 provides the settings and results of evaluation experiments. Section 4 discusses some of the existing approaches related to ours. Section 5 concludes the paper by giving a future research direction.

2 Method

2.1 Variational Lower Bound

While there are many ways to perform an inference for topic models, we adopt variational Bayesian inference [3][9][4], because it allows us to perform inference as an optimization. Our topic model is a modification of the vanilla LDA [3]. In our model, a symmetric Dirichlet prior Dirichlet(α) is assigned to the per-document categorical distributions over topics as in the vanilla LDA. In contrast, no prior is assigned to the per-topic categorical distributions over words in our model, because the word probabilities are directly estimated in a time-aware manner. The evidence lower bound (ELBO) to be maximized in variational Bayesian inference can thus be written as follows [3]:

$$\begin{aligned}\sum_d \log p(\boldsymbol{x}_d; \alpha, \boldsymbol{\Phi}) \geq & \sum_{d,v,k} n_{d,v}\gamma_{d,v,k} \log \phi_{k,v} \\ & + \sum_{d,k} \Big(\alpha + \sum_v n_{d,v}\gamma_{d,v,k} - \lambda_{d,k}\Big)\Big\{\Psi(\lambda_{d,k}) - \Psi\Big(\sum_{k'} \lambda_{d,k'}\Big)\Big\} \\ & - \sum_{d,v,k} n_{d,v}\gamma_{d,v,k} \log \gamma_{d,v,k} + \log \Gamma(K\alpha) - K \log \Gamma(\alpha) \end{aligned} \tag{3}$$

where $\boldsymbol{x}_d$ is the bag-of-words representation of the document d. $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_K\}$ are the parameter vectors of the per-topic categorical distributions over words, i.e., per-topic word probabilities. $n_{d,v}$ is the frequency of the word v in d. $\gamma_{d,v,k}$ is the responsibility of the word v with respect to the topic k in d. $\lambda_{d,k}$ is the posterior parameter for the topic k in d. Ψ is the digamma function. We regard the parameter α of the symmetric Dirichlet prior Dirichlet(α) as free parameter. The document-specific parameters $\lambda_{d,k}$ and $\gamma_{d,v,k}$ are estimated in the same manner as in [3][9][4]. However, we estimate $\boldsymbol{\Phi}$ differently, because the per-topic word probabilities are modeled differently. The details are given below.

2.2 Modeling of Time-Awareness

The vanilla LDA represents the per-topic word distributions simply as the categorical distributions or as those to which a Dirichlet prior is assigned. In our model, the parameter vector $\boldsymbol{\phi}_k$ of the word categorical distribution for each topic k is given by

$$\phi_{k,v} = \frac{\exp(w_{k,v} + b_v)}{\sum_{v'} \exp(w_{k,v'} + b_{v'})} \tag{4}$$

where the softmax function is used. $\boldsymbol{w}_k$ can be viewed as a vector embedding of the topic k as discussed in Section 1.1. While we use no prior, the bias term b_v in Eq. (4) plays a similar role to that of Dirichlet prior in smoothing [5][1], because b_v is shared by all topics.

Our proposal also considers the usage of covariates like timestamps, authors, venues, etc. In this paper, we explore chronological trends in document contents and thus use document timestamps in order to make the per-topic word probabilities *time-aware*, though other covariates may also be used in a similar manner. By using timestamps, the per-topic word probabilities are given by

$$\phi_{k,t,v} = \frac{\exp(w_{k,v} + u_{t,v} + b_v)}{\sum_{v'} \exp(w_{k,v'} + u_{t,v'} + b_{v'})} \tag{5}$$

where t represents document timestamps. $\boldsymbol{U} = (u_{t,v})^\top$ is a weight matrix modeling the dependency of word probabilities on timestamps. Each column $\boldsymbol{u}_t$ of $\boldsymbol{U}$

can be viewed as a vector embedding of the timestamp t. As already discussed in Section 1.1, the $\phi_{k,t,v}$'s can also be written as softmax outputs of a single-layer network whose input is always one-hot:

$$\boldsymbol{\phi}_{k,t} = \mathrm{Softmax}(\boldsymbol{W}\boldsymbol{e}_k + \boldsymbol{U}\boldsymbol{o}_t + \boldsymbol{b}) \tag{6}$$

where $\boldsymbol{e}_k$ and $\boldsymbol{o}_t$ are one-hot input vectors corresponding to the latent topic k and the timestamp t, respectively.

2.3 Implementation

In the manner described above, the parameter vector $\boldsymbol{\phi}_k$ and its time-aware version $\boldsymbol{\phi}_{k,t}$ are regarded as an output of the softmax function. The weights and biases, i.e., $\boldsymbol{W}$, $\boldsymbol{U}$, and $\boldsymbol{b}$, are estimated by maximizing the ELBO in Eq. (3). We implemented the maximization as a mini-batch learning by using PyTorch. Thanks to the broadcasting mechanism in PyTorch, we can write Eq. (6) as

```
phi = torch.nn.functional.softmax(
    W.unsqueeze(2)
    + U.unsqueeze(1)
    + b.unsqueeze(1).unsqueeze(2),
    dim=0)
```

by collecting all $\phi_{k,t,v}$'s in one place. In the evaluation experiments, we found that when it was difficult to obtain a good validation perplexity, DropConnect [15] worked very well. This can be implemented by modifying the above code snippet as follows:

```
dropout = torch.nn.Dropout(p=dropout_prob)
phi = torch.nn.functional.softmax(
    dropout(W).unsqueeze(2)
    + dropout(U).unsqueeze(1)
    + b.unsqueeze(1).unsqueeze(2),
    dim=0)
```

The dropout probability `dropout_prob` was chosen from $\{0.2, 0.5\}$ in our experiment. No further fine tuning of probability was required. However, it was mandatory to check if DropConnect could improve test perplexity, because the extent of improvement sometimes reached a significant level.

2.4 Parameterization Efficiency

Here we give an additional discussion on modeling. As Eq. (5) shows, our method combines the parameters indexed by the tuple (k, v) with those indexed by (t, v) to make our topic extraction dependent on document timestamps. Therefore, the memory space for these parameters is $O(KV + TV)$. However, it is also possible to introduce the parameters indexed explicitly by the 3-tuple (k, t, v) as follows in the same manner as in [7] and [13]:

$$\phi_{k,t,v} = \frac{\exp(w_{k,v} + u_{t,v} + r_{k,t,v} + b_v)}{\sum_{v'} \exp(w_{k,v'} + u_{t,v'} + r_{k,t,v'} + b_{v'})} \ . \tag{7}$$

A merit of this approach is that we can separate out the words not depending on topics but exclusively depending on timestamps (e.g. ‘2001’, ‘December’, etc) from the words depending both on topics and on timestamps equally (e.g. ‘ATM’, ‘WDM’, etc[4]). The words of the former type may appear in documents related to a wide range of topics but only appear at some limited set of time points. Therefore, such words may have probabilities mainly determined by the parameters indexed by (t, v), and the contribution of the parameters indexed by (k, t, v) may be small. The words of the latter type may appear in documents related to a narrow range of topics and at the same time only appear at some limited set of time points. Such words may have probabilities determined both by the parameters indexed by (t, v) and by those indexed by (k, t, v).

Our model has no parameters indexed by (k, t, v). Therefore, it may be difficult to make a clear distinction between the above two types of words. However, an obvious drawback of the above approach is the memory space requirement of $O(KTV)$. When we use GPU, it can be prohibitive to allocate a memory space for $K \times T \times V$ parameters each requiring its gradient. Even without using this many parameters, we could obtain interesting time-dependencies as presented in Fig. 1 and in Fig. 3. Therefore, we did not introduce the parameters explicitly indexed by (k, t, v).

2.5 Details of Inference Algorithm

Algorithm 1 is the inference for the proposed model. The inference related to the document-specific parameters are performed in the same manner as [3][9][4]. Only the corpus-wide parameters, i.e., $\boldsymbol{W}$, $\boldsymbol{U}$, and $\boldsymbol{b}$, are updated by backpropagation. The inputs for the algorithm are: (1) the frequency $n_{d,v}$ of the word v in the document d; (2) the parameter α of the symmetric Dirichlet prior distribution; and (3) the initial learning rate η of optimization algorithm. As Algorithm 1 shows, we don’t need to know the total number of training documents.

The elements of $\boldsymbol{W}$ and $\boldsymbol{U}$ were initialized by samples from the Gaussian distribution with zero-mean and a variance of 0.01. The bias terms $\boldsymbol{b}$ were initialized to zero. For updating these parameters, we used Adam optimizer [10]. Let the initial learning rate of Adam be denoted by η. In our implementation, the learning rate for the m-th mini-batch was set to $(1+m/500)^{-0.7} \times \eta$. Further, η was halved every 500 mini-batches until 2,000 mini-batches were seen.

[4] These words are observed often in the titles of computer science papers related to networking and communication but only appear across a limited range of years.

Algorithm 1 Mini-batch variational Bayes

```
 1: Input: $n_{d,v}$ for each mini-batch, $\alpha$, and $\eta$
 2: Initialize $\boldsymbol{W}$, $\boldsymbol{U}$, and $\boldsymbol{b}$
 3: while True do
 4:     $\boldsymbol{\phi}_{k,t} \leftarrow \mathrm{Softmax}(\boldsymbol{W}\boldsymbol{e}_k + \boldsymbol{U}\boldsymbol{o}_t + \boldsymbol{b})$
 5:     Get next mini-batch
 6:     for each document $d$ in mini-batch do
 7:         $t \leftarrow$ timestamp of document $d$
 8:         Initialize $\gamma_{d,v,k}$ randomly
 9:         repeat
10:             $\lambda_{d,k} \leftarrow \alpha + \sum_v n_{d,v}\gamma_{d,v,k}$
11:             $\gamma_{d,v,k} \leftarrow\propto \exp\big(\Psi(\lambda_{d,k})\big) \times \phi_{k,t,v}$
12:         until change in $\lambda_{d,k}$ is negligible
13:     end for
14:     Make computational graph of the negative of ELBO
15:     Backpropagate
16:     Update $\boldsymbol{W}$, $\boldsymbol{U}$, and $\boldsymbol{b}$ by using $\eta$
17:     Update $\eta$ if necessary
18: end while
```

3 Experiments

3.1 Data Sets

The specifications of the four document sets used in the evaluation experiments are given in Table 1, where the total number of documents, the vocabulary size, and the number of different document timestamps are denoted by D, V, and T, respectively. A detailed explanation is given below.

The first data set 'MAI' is a set of newswire articles from the Mainichi, a Japanese newspaper. A morphological analysis was performed by using MeCab.[5] The dates of the articles range from November 1, 2007 to May 15, 2008. We split the date range into 13 slices of nearly equal size and used the slices as document timestamps. The second data set 'TDT4' is a set of English documents from TDT4 Multilingual Text and Annotations.[6] The dates of the documents range from December 1, 2000 to January 31, 2001. We split the date range into 15 slices of nearly equal size and used them as document timestamps. The third data set 'DBLP' is a set of paper titles obtained from DBLP web site.[7] We regarded each title as a single document. Therefore, this is a set of documents of quite short length. The publication years ranging from 1998 to 2012 were used as document timestamps. The last data set 'STOV' is a subset of the questions in the Stack-Overflow data set available at Kaggle.[8] We used as document timestamps the months on which the questions were published, ranging from January 2014 to

[5] `http://taku910.github.io/mecab/`
[6] `https://catalog.ldc.upenn.edu/LDC2005T16`
[7] `http://dblp.uni-trier.de/xml/`
[8] `https://www.kaggle.com/stackoverflow/rquestions`

Table 1. Specifications of data sets

	D	V	T		D	V	T
MAI	32,775	15,161	13	DBLP	1,034,067	18,940	15
TDT4	96,246	15,153	15	STOV	25,608	13,184	34

December 2016. For all data sets, English words were lower-cased, and highly frequent words and infrequent words were removed. The training:validation:test ratio was 8:1:1 for all data sets. We used one random split as in the usual experiments for neural networks. The mini-batch size was 200 for MAI, TDT4, and STOV data sets and was 10,000 for DBLP data set, because DBLP is a set of short documents.

3.2 Evaluation Method

We compared the following three approaches for topic extraction: the mini-batch inference for the topic model with document timestamps (cf. Eq. (5)), the mini-batch inference for the topic model without document timestamps (cf. Eq. (4)), and collapsed Gibbs sampling (CGS) [8] for the vanilla LDA. These compared methods are referred to by ‘VB w/ t’, ‘VB’, and ‘CGS’, respectively in Table 2, which summarizes the evaluation results. With respect to K, i.e., the number of latent topics, we tested the following two settings: $K = 50$ and $K = 100$.

The comparison is performed in terms of test perplexity [3]. However, each approach has its own free parameters to be tuned. Therefore, we tuned the free parameters based on the perplexity computed over validation set. For ‘VB w/ t’ and ‘VB’, the parameter α of the symmetric Dirichlet prior distribution and the initial learning rate η of Adam were tuned based on validation perplexity. The tuned values of η and α are given in Table 2. With respect to ‘CGS’, not only α but also the parameter β of the symmetric Dirichlet prior distribution assigned to the per-topic word categorical distributions were tuned based on validation perplexity. The tuned values of α and β are given in Table 2.

Both the validation and test perplexities were computed by using fold-in procedure [1], where randomly selected two-thirds of word tokens were used for obtaining topic posterior probabilities in each document. There are myriads of ways to select two-thirds of word tokens. Therefore, when computing test perplexity, i.e., when performing the final evaluation, and not when computing validation perplexity, we repeated the computation of perplexity ten times, each over a different set of randomly chosen two-thirds of word tokens. Also for CGS, we performed an evaluation in the same manner. We then calculated the mean and standard deviation of ten resulting test perplexities as given in Table 2.

3.3 Comparison Results

Table 2 provides the mean and standard deviation of ten test perplexities obtained in a manner described above for each data set. It can be observed that the mini-batch variational Bayes for the topic model using document timestamps

Table 2. Evaluation results in terms of test perplexity

	K	method	perplexity (mean±stdev)	free parameters (η: learning rate) (α, β: Dirichlet)
MAI	100	VB w/ t	**1115.64±3.94**	$\eta = .07, \alpha = .03$
		VB	1138.99±3.11	$\eta = .07, \alpha = .03$
		CGS	1130.93±3.04	$\alpha = .02, \beta = .05$
	50	VB w/ t	**1283.05±5.74**	$\eta = .08, \alpha = .02$
		VB	1315.10±5.94	$\eta = .08, \alpha = .02$
		CGS	1336.75±5.56	$\alpha = .04, \beta = .15$
TDT4	100	VB w/ t	1480.82±3.85	$\eta = .06, \alpha = .02$
		VB	1481.13±3.79	$\eta = .06, \alpha = .04$
		CGS	**1425.64±3.08**	$\alpha = .02, \beta = .10$
	50	VB w/ t	1587.45±3.25	$\eta = .05, \alpha = .01$
		VB	1589.99±3.10	$\eta = .10, \alpha = .02$
		CGS	1586.48±3.05	$\alpha = .05, \beta = .05$
DBLP	100	VB w/ t	**1274.45±3.88**	η =6e-4, $\alpha = .09^\dagger$
		VB	1325.30±5.25	η =8e-4, $\alpha = .08^\dagger$
		CGS	1341.07±4.84	$\alpha = .03, \beta = .04$
	50	VB w/ t	**1300.09±5.69**	η =5e-4, $\alpha = .12^\dagger$
		VB	1335.71±5.72	η =6e-4, $\alpha = .10^\dagger$
		CGS	1384.36±6.21	$\alpha = .04, \beta = .05$
STOV	100	VB w/ t	1083.10±4.82	$\eta = .015, \alpha = .20^\dagger$
		VB	1125.78±5.62	$\eta = .010, \alpha = .30^\dagger$
		CGS	**1042.18±4.19**	$\alpha = .05, \beta = .01$
	50	VB w/ t	1255.72±3.59	$\eta = .018, \alpha = .30$
		VB	1266.16±3.43	$\eta = .020, \alpha = .35$
		CGS	**1178.20±6.16**	$\alpha = .07, \beta = .04$

(i.e., the model based on Eq. (5)) gave the best test perplexity among the three compared methods when $K = 50$ or 100 for MAI and when $K = 50$ or 100 for DBLP. Further, it can be also observed that even when we adopt the topic model using no timestamps (i.e., the model based on Eq. (4)), the test perplexity is better than that of CGS in three cases, i.e., when $K = 50$ for MAI and when $K = 50$ or 100 for DBLP. When we consider the fact that CGS often gives a good test perplexity [1], it can be said that even when we use no timestamps, the proposed mini-batch variational Bayes is a promising alternative to CGS when the size of training data set is prohibitively large for CGS.

Another important observation is that DropConnect [15] led to a remarkable improvement when $K = 50$ or 100 for DBLP and when $K = 100$ for STOV. These cases are marked by dagger '†' in Table 2. DropConnect worked both for the modeling with timestamps and for that without timestamps. Fig. 2 shows to what extent DropConnect could improve the quality of extracted topics in terms of validation perplexity for DBLP data set when $K = 50$. The horizontal axis gives the number of seen mini-batches. The vertical axis gives the validation perplexity computed during the course of inference. As Fig. 2 shows, DropCon-

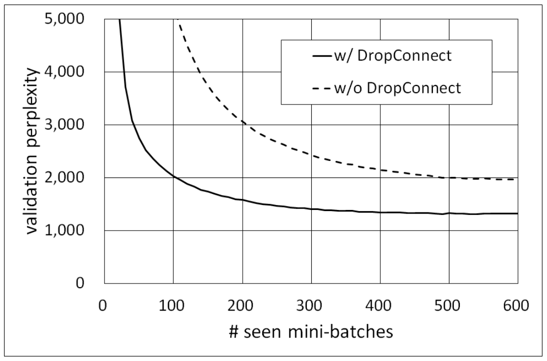

Fig. 2. Comparing the inference with DropConnect (solid line) to the inference without it (dashed line) in terms of validation perplexity for DBLP data set when $K = 50$. The horizontal axis gives the number of seen mini-batches. The vertical axis gives the perplexity computed over the validation set. The learning rate η and the Dirichlet hyperparameter α were tuned separately for each inference. The validation perplexity given by the inference without DropConnect (dashed line) reached a stable value, which was around 1,870, after 2,000 mini-batches were seen. The validation perplexity given by the inference with DropConnect reached around 1,320 as presented in the above chart. That is, the perplexity was improved by around 500.

nect improved the validation perplexity by around 500. Based on this kind of result, we adopted DropConnect also for the other two cases.

3.4 Examples of Extracted Topical Trends

Fig. 1 and Fig. 3 present the latent topics extracted by the proposed model as word clouds.[9] These two topics were extracted from 827,141 paper titles contained in the training set of DBLP data set. In the three panels of each figure, we provide top 50 words according to their probabilities at three different time points, 1998, 2005, and 2012, respectively. The words having larger probabilities are depicted in larger fonts. For the topic given in Fig. 1, seemingly related to HCI, the word ‘object’ is more relevant to this topic in 1998 than in 2005 and 2012. In contrast, the word ‘application’ is more relevant in 2005 than in 1998 and 2012, and the words ‘interaction’ and ‘augmented’ are more relevant in 2012 than in 1998 and 2005. For the topic given in Fig. 3, seemingly related to wireless network, the word ‘cellular’ is more relevant to this topic in 1998 and 2005 than in 2012, the word ‘wireless’ is more relevant in 2005 and 2012 than in 1998, and the word ‘power’ is more relevant in 2012 than in 1998 and 2005. In this manner, we can observe remarkable trends by comparing word probability distributions coming from different time points. Our approach enables us to conduct this type of time-aware topic analysis.

[9] `https://github.com/amueller/word_cloud`

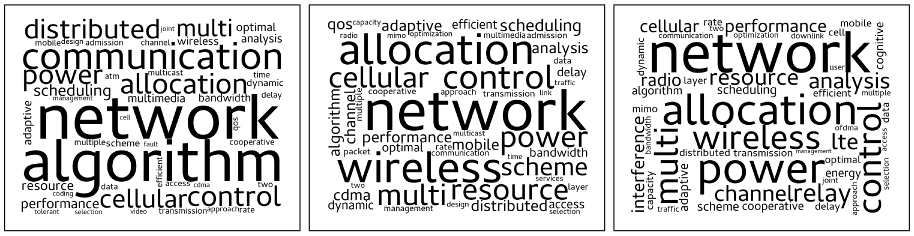

Fig. 3. Word cloud presentation of the same topic at three different time points. This topic, seemingly related to wireless network, extracted from 827,141 paper titles in DBLP data set. K was set to 50. The left, center, and right panels correspond to the time points 1998, 2005, and 2012, respectively.

4 Related work

While NVDM [11] is the first proposal of document modeling using neural networks, several other proposals combine topic modeling with neural networks in a more interesting and interpretable manner. Mikolov and Zweig [12] devised an RNN-based language model using LDA. The output of the inference for LDA is used as an additional context information for language modeling. However, this is neither a new topic model nor a new inference for topic models. Srivastava and Sutton [14] proposed a topic model, called ProdLDA, in order to apply autoencoding variational Bayes, which was implemented by using neural networks. Dieng et al. [6] proposed a topic model, called TopicRNN, where RNN is used to obtain per-document word probabilities. Both of these models use the softmax function to carry out the word-level mixture over per-topic word probabilities after marginalizing each $z_{d,i}$, a hidden variable representing the topic assignment of the i-th word token in the document d. However, since both models marginalize the $z_{d,i}$'s out, they widely deviate from the vanilla LDA [3]. Our method introduces only a limited complication to the vanilla LDA and keeps the coordinate ascent related to the document-specific parameters untouched to make the inference simple and efficient.

5 Conclusion

This paper proposed a topic model using document timestamps as covariate and provided a mini-batch inference, whose implementation was realized by using a deep learning framework. While learning for topic models is performed in an unsupervised manner, we can regard it as an optimization by adopting variational inference approach. Our mini-batch variational inference updated per-topic word probabilities in a similar manner to neural network parameters and required no knowledge about the training data size. We also showed that DropConnect worked in our inference. It is an important future research direction to consider covariates other than timestamps and to introduce more flexibility in modeling by referring to the recent work discussed in Section 4.

References

1. Asuncion, A., Welling, M., Smyth, P., and Teh, Y. W.: On smoothing and inference for topic models. In Proc. the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09), 27–34 (2009)
2. Blei, D. M., Carin, L., and Dunson, D.: Probabilistic Topic Models. IEEE Signal Process. Mag. 27, no. 6, 55–65 (2010)
3. Blei, D. M., Ng, A. Y., and Jordan, M. I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003)
4. Broderick, T., Boyd, N., Wibisono, A., Wilson, A. C., and Jordan, M. I.: Streaming Variational Bayes. In Proc. Advances in Neural Information Processing Systems (NIPS), 1727–1735 (2013)
5. Chen, S. F. and Goodman, J.: An Empirical Study of Smoothing Techniques for Language Modeling. In Proc. Annual Meeting on Association for Computational Linguistics (ACL), 310–318 (1996)
6. Dieng, A. B., Wang, C., Gao, J., and Paisley, J.: TopicRNN: A recurrent neural network with long-range semantic dependency. arXiv preprint, arXiv:1611.01702 (2016)
7. Eisenstein, J., Ahmed, A., and Xing, E. P.: Sparse Additive Generative Models of Text. In Proc. International Conference on Machine Learning (ICML), 1041–1048 (2011)
8. Griffiths T. L., and Steyvers, M.: Finding Scientific Topics. Proc. Natl. Acad. Sci. U.S.A 101, Suppl. 1, 5288–5235 (2004)
9. Hoffman, M. D., Blei, D. M., and Bach, F.: Online learning for latent Dirichlet allocation. In Proc. Advances in Neural Information Processing Systems (NIPS), 856–864 (2010)
10. Kingma, D. P. and Ba, J. L.: Adam: a Method for Stochastic Optimization. In Proc. International Conference on Learning Representations (ICLR) (2015)
11. Miao, Y., Yu, L., and Blunsom, P.: Neural Variational Inference for Text Processing. In Proc. International Conference on Machine Learning (ICML), 1727–1736 (2016)
12. Mikolov, T. and Zweig, G.: Context Dependent Recurrent Neural Network Language Model. In Proc. IEEE Spoken Language Technology Workshop, 234–239 (2012)
13. Roberts, M. E., Stewart, B. M., Tingley, D., and Airoldi, E. M.: The Structural Topic Model and Applied Social Science. In Proc. Advances in Neural Information Processing Systems Workshop on Topic Models: Computation, Application, and Evaluation (2013)
14. Srivastava, A. and Sutton, C.: Autoencoding Variational Inference For Topic Models. In Proc. International Conference on Learning Representations (ICLR) (2017)
15. Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R.: Regularization of Neural Networks using DropConnect. In Proc. International Conference on Machine Learning (ICML), 1058–1066 (2013)