A Latent Concept Topic Model for Robust Topic Inference Using Word Embeddings

Weihua Hu[†] **and Jun'ichi Tsujii**[‡§]
[†]Department of Computer Science, The University of Tokyo, Japan
[‡]Artificial Intelligence Research Center, AIST, Japan
[§]School of Computer Science, The University of Manchester, UK
{hu,j-tsujii}@ms.k.u-tokyo.ac.jp,@aist.go.jp

Abstract

Uncovering thematic structures of SNS and blog posts is a crucial yet challenging task, because of the severe data sparsity induced by the short length of texts and diverse use of vocabulary. This hinders effective topic inference of traditional LDA because it infers topics based on document-level co-occurrence of words. To robustly infer topics in such contexts, we propose a latent concept topic model (LCTM). Unlike LDA, LCTM reveals topics via co-occurrence of *latent concepts*, which we introduce as latent variables to capture conceptual similarity of words. More specifically, LCTM models each topic as a distribution over the *latent concepts*, where each *latent concept* is a localized Gaussian distribution over the word embedding space. Since the number of unique concepts in a corpus is often much smaller than the number of unique words, LCTM is less susceptible to the data sparsity. Experiments on the 20Newsgroups show the effectiveness of LCTM in dealing with short texts as well as the capability of the model in handling held-out documents with a high degree of OOV words.

1 Introduction

Probabilistic topic models such as Latent Dirichlet allocation (LDA) (Blei et al., 2003), are widely used to uncover hidden topics within a text corpus. LDA models each document as a mixture of topics where each topic is a distribution over words. In essence, LDA reveals latent topics in a corpus by implicitly capturing document-level word co-occurrence patterns (Wang and McCallum, 2006).

In recent years, Social Networking Services and blogs have become increasingly prevalent due to the explosive growth of the Internet. Uncovering the themantic structures of these posts is crucial for tasks like market review, trend estimation (Asur and Huberman, 2010) and so on. However, compared to more conventional documents, such as news articles and academic papers, analyzing the thematic content of blog posts can be challenging, because of their typically short length and the use of diverse vocabulary by various authors. These factors can substantially decrease the chance of topically related words co-occurring in the same post, which in turn hinders effective topic inference in conventional topic models. Additionally, sometimes small corpus size can further exacerbate topic inference, since word co-occurrence statistics becomes more sparse as the number of documents decreases.

Recently, word embedding models, such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) have gained much attention with their ability to form clusters of conceptually similar words in the embedding space. Inspired by this, we propose a latent concept topic model (LCTM) that infers topics based on document-level co-occurrence of references to the same *concept*. More specifically, we introduce a new latent variable, termed a *latent concept* to capture conceptual similarity of words, and redefine each topic as a distribution over the *latent concepts*. Each *latent concept* is then modeled as a localized Gaussian distribution over the embedding space. This is illustrated in Figure 1, where we denote the centers of the Gaussian distributions as *concept vectors*. We see that each *concept vector* captures a representative concept of surrounding words, and the Gaussian distributions model the small variation between the *latent concepts* and the actual use of words. Since the number of unique concepts that are referenced in a corpus is often much smaller than the number of unique

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 380–386,
Berlin, Germany, August 7-12, 2016.

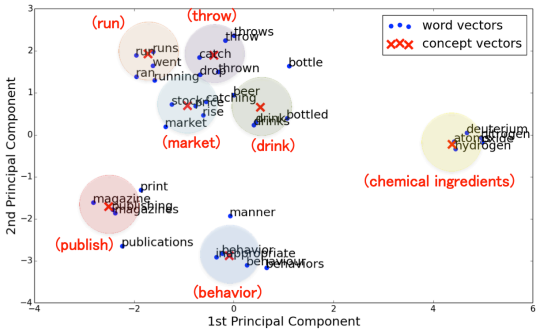


Figure 1: Projected *latent concepts* on the word embedding space. *Concept vectors* are annotated with their representative concepts in parentheses.

words, we expect topically-related *latent concepts* to co-occur many times, even in short texts with diverse usage of words. This in turn promotes topic inference in LCTM.

LCTM further has the advantage of using continuous word embedding. Traditional LDA assumes a fixed vocabulary of word types. This modeling assumption prevents LDA from handling *out of vocabulary* (OOV) words in held-out documents. On the other hands, since our topic model operates on the continuous vector space, it can naturally handle OOV words once their vector representation is provided.

The main contributions of our paper are as follows: We propose LCTM that infers topics via document-level co-occurrence patterns of *latent concepts*, and derive a collapsed Gibbs sampler for approximate inference. We show that LCTM can accurately represent short texts by outperforming conventional topic models in a clustering task. By means of a classification task, we furthermore demonstrate that LCTM achieves superior performance to other state-of-the-art topic models in handling documents with a high degree of OOV words.

The remainder of the paper is organized as follows: related work is summarized in Section 2, while LCTM and its inference algorithm are presented in Section 3. Experiments on the 20Newsgroups are presented in Section 4, and a conclusion is presented in Section 5.

2 Related Work

There have been a number of previous studies on topic models that incorporate word embeddings. The closest model to LCTM is Gaussian LDA (Das et al., 2015), which models each topic as a Gaussian distribution over the word embedding space. However, the assumption that topics are unimodal in the embedding space is not appropriate, since topically related words such as 'neural' and 'networks' can occur distantly from each other in the embedding space. Nguyen et al. (2015) proposed topic models that incorporate information of word vectors in modeling topic-word distributions. Similarly, Petterson et al. (Petterson et al., 2010) exploits external word features to improve the Dirichlet prior of the topic-word distributions. However, both of the models cannot handle OOV words, because they assume fixed word types.

Latent concepts in LCTM are closely related to '*constraints*' in interactive topic models (ITM) (Hu et al., 2014). Both *latent concepts* and *constraints* are designed to group conceptually similar words using external knowledge in an attempt to aid topic inference. The difference lies in their modeling assumptions: *latent concepts* in LCTM are modeled as *Gaussian distributions* over the embedding space, while *constraints* in ITM are sets of conceptually similar words that are interactively identified by humans for each topic. Each *constraint* for each topic is then modeled as a *multinomial distribution* over the constrained set of words that were identified as mutually related by humans. In Section 4, we consider a variant of ITM, whose *constraints* are instead inferred using external word embeddings.

As regards short texts, a well-known topic model is Biterm Topic Model (BTM) (Yan et al., 2013). BTM directly models the generation of biterms (pairs of words) in the whole corpus. However, the assumption that pairs of co-occurring words should be assigned to the same topic might be too strong (Chen et al., 2015).

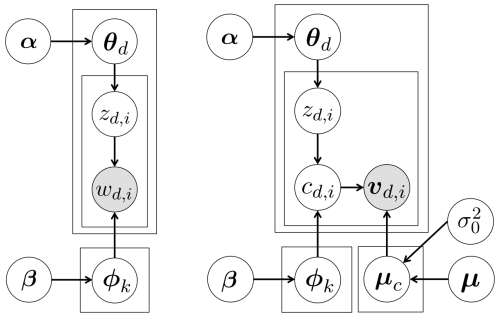
3 Latent Concept Topic Model

3.1 Generative Model

The primary difference between LCTM and the conventional topic models is that LCTM describes the generative process of word vectors in documents, rather than words themselves.

Suppose $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are parameters for the Dirichlet priors and let $\boldsymbol{v}_{d,i}$ denote the word embedding for a word type $w_{d,i}$. The generative model for LCTM is as follows.

1. For each topic k
   (a) Draw a topic *concept* distribution $\boldsymbol{\phi}_k \sim \text{Dirichlet}(\boldsymbol{\beta})$.



(a) LDA. (b) LCTM.

Figure 2: Graphical representation.

2. For each *latent concept* c
   (a) Draw a *concept vector* $\boldsymbol{\mu}_c \sim \mathcal{N}(\boldsymbol{\mu}, \sigma_0^2 \boldsymbol{I})$.
3. For each document d
   (a) Draw a document topic distribution $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha})$.
   (b) For the i-th word $w_{d,i}$ in document d
      i. Draw its topic assignment $z_{d,i} \sim \text{Categorical}(\boldsymbol{\theta}_d)$.
      ii. Draw its *latent concept assignment* $c_{d,i} \sim \text{Categorical}(\boldsymbol{\phi}_{z_{d,i}})$.
      iii. Draw a word vector $\boldsymbol{v}_{d,i} \sim \mathcal{N}(\boldsymbol{\mu}_{c_{d,i}}, \sigma^2 \boldsymbol{I})$.

The graphical models for LDA and LCTM are shown in Figure 2. Compared to LDA, LCTM adds another layer of latent variables to indicate the conceptual similarity of words.

3.2 Posterior Inference

In our application, we observe documents consisting of word vectors and wish to infer posterior distributions over all the hidden variables. Since there is no analytical solution to the posterior, we derive a collapsed Gibbs sampler to perform approximate inference. During the inference, we sample a *latent concept* assignment as well as a topic assignment for each word in each document as follows:

$$p(z_{d,i} = k \mid c_{d,i} = c, \boldsymbol{z}^{-d,i}, \boldsymbol{c}^{-d,i}, \boldsymbol{v}) \propto \left(n_{d,k}^{-d,i} + \alpha_k\right) \cdot \frac{n_{k,c}^{-d,i} + \beta_c}{n_{k,\cdot}^{-d,i} + \sum_{c'} \beta_{c'}}, \quad (1)$$

$$P(c_{d,i} = c \mid z_{d,i} = k, \boldsymbol{v}_{d,i}, \boldsymbol{z}^{-d,i}, \boldsymbol{c}^{-d,i}, \boldsymbol{v}^{-d,i}) \propto \left(n_{k,c}^{-d,i} + \beta_c\right) \cdot \mathcal{N}(\boldsymbol{v}_{d,i} | \overline{\boldsymbol{\mu}}_c, \sigma_c^2 \boldsymbol{I}), \quad (2)$$

where $n_{d,k}$ is the number of words assigned to topic k in document d, and $n_{k,c}$ is the number of words assigned to both topic k and *latent concept* c. When an index is replaced by '$\cdot$', the number is obtained by summing over the index. The superscript $^{-d,i}$ indicates that the current assignments of $z_{d,i}$ and $c_{d,i}$ are ignored. $\mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a multivariate Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. $\overline{\boldsymbol{\mu}}_c$ and σ_c^2 in Eq. (2) are parameters associated with the *latent concept* c and are defined as follows:

$$\overline{\boldsymbol{\mu}}_c = \frac{1}{\sigma^2 + n_{\cdot,c}^{-d,i} \sigma_0^2} \left(\sigma^2 \boldsymbol{\mu} + \sigma_0^2 \cdot \sum_{(d',i') \in A_c^{-d,i}} \boldsymbol{v}_{d',i'} \right), \quad (3)$$

$$\sigma_c^2 = \left(1 + \frac{\sigma_0^2}{n_{\cdot,c}^{-d,i} \sigma_0^2 + \sigma^2}\right) \sigma^2, \quad (4)$$

where $A_c^{-d,i} \equiv \{(d', i') \mid c_{d',i'} = c \wedge (d', i') \neq (d, i)\}$ (Murphy, 2012). Eq. (1) is similar to the collapsed Gibbs sampler of LDA (Griffiths and Steyvers, 2004) except that the second term of Eq. (1) is concerned with topic-*concept* distributions. Eq. (2) of sampling *latent concepts* has an intuitive interpretation: the first term encourages *concept* assignments that are consistent with the current topic assignment, while the second term encourages *concept* assignments that are consistent with the observed word. The Gaussian variance parameter σ^2 acts as a trade-off parameter between the two terms via σ_c^2. In Section 4.2, we study the effect of σ^2 on document representation.

3.3 Prediction of Topic Proportions

After the posterior inference, the posterior means of $\{\boldsymbol{\theta}_d\}$, $\{\boldsymbol{\phi}_k\}$ are straightforward to calculate:

$$\overline{\theta}_{d,k} = \frac{n_{d,k} + \alpha_k}{n_{d,\cdot} + \sum_{k'} \alpha_{k'}}, \quad \overline{\phi}_{k,c} = \frac{n_{k,c} + \beta_c}{n_{k,\cdot} + \sum_{c'} \beta_{c'}}. \quad (5)$$

Also posterior means for $\{\boldsymbol{\mu}_c\}$ are given by Eq. (3). We can then use these values to predict a topic proportion $\boldsymbol{\theta}_{d_{\text{new}}}$ of an unseen document d_{new} using collapsed Gibbs sampling as follows:

$$p(z_{d_{\text{new}},i} = k \mid \boldsymbol{v}_{d_{\text{new}},i}, \boldsymbol{v}^{-d_{\text{new}},i}, \boldsymbol{z}^{-d_{\text{new}},i}, \overline{\boldsymbol{\phi}}, \overline{\boldsymbol{\mu}}) \propto \left(n_{d_{\text{new}},k}^{-d_{\text{new}},i} + \alpha_k\right) \cdot \sum_c \overline{\phi}_{k,c} \frac{\mathcal{N}(\boldsymbol{v}_{d_{\text{new}},i} | \overline{\boldsymbol{\mu}}_c, \sigma_c^2)}{\sum_{c'} \mathcal{N}(\boldsymbol{v}_{d_{\text{new}},i} | \overline{\boldsymbol{\mu}}_{c'}, \sigma_{c'}^2)}. \quad (6)$$

The second term of Eq. (6) is a weighted average of $\overline{\phi}_{k,c}$ with respect to *latent concepts*. We see that more weight is given to the *concepts* whose corresponding vectors $\overline{\boldsymbol{\mu}}_c$ are closer to the word vector $\boldsymbol{v}_{d_{\text{new}},i}$. This to be expected because statistics of nearby *concepts* should give more information about the word. We also see from Eq. (6) that the

topic assignment of a word is determined by its embedding, instead of its word type. Therefore, LCTM can naturally handle OOV words once their embeddings are provided.

3.4 Reducing the Computational Complexity

From Eqs. (1) and (2), we see that the computational complexity of sampling per word is $\mathcal{O}(K + SD)$, where K, S and D are numbers of topics, *latent concepts* and embedding dimensions, respectively. Since $K \ll S$ holds in usual settings, the dominant computation involves the sampling of *latent concept*, which costs $\mathcal{O}(SD)$ computation per word.

However, since LCTM assumes that Gaussian variance σ^2 is relatively small, the chance of a word being assigned to distant *concepts* is negligible. Thus, we can reasonably assume that each word is assigned to one of $M \ll S$ nearest *concepts*. Hence, the computational complexity is reduced to $\mathcal{O}(MD)$. Since *concept vectors* can move slightly in the embedding space during the inference, we periodically update the nearest *concepts* for each word type.

To further reduce the computational complexity, we can apply dimensional reduction algorithms such as PCA and t-SNE (Van der Maaten and Hinton, 2008) to word embeddings to make D smaller. We leave this to future work.

4 Experiments

4.1 Datasets and Models Description

In this section, we study the empirical performance of LCTM on short texts. We used the 20Newsgroups corpus, which consists of discussion posts about various news subjects authored by diverse readers. Each document in the corpus is tagged with one of twenty newsgroups. Only posts with less than 50 words are extracted for training datasets. For external word embeddings, we used 50-dimensional GloVe[1] that were pre-trained on Wikipedia. The datasets are summarized in Table 1. See appendix A for the detail of the dataset preprocessing.

We compare the performance of the LCTM to the following six baselines:

- LFLDA (Nguyen et al., 2015), an extension of Latent Dirichlet Allocation that incorporates word embeddings information.

[1]Downloaded at http://nlp.stanford.edu/projects/glove/

Dataset	Doc size	Vocab size	Avg len
400short	400	4729	31.87
800short	800	7329	31.78
1561short	1561	10644	31.83
held-out	7235	37944	140.15

Table 1: Summary of datasets.

- LFDMM (Nguyen et al., 2015), an extension of Dirichlet Multinomial Mixtures that incorporates word embeddings information.
- nI-cLDA, non-interactive constrained Latent Dirichlet Allocatoin, a variant of ITM (Hu et al., 2014), where *constraints* are inferred by applying k-means to external word embeddings. Each resulting word cluster is then regarded as a *constraint*. See appendix B for the detail of the model.
- GLDA (Das et al., 2015), Gaussian LDA.
- BTM (Yan et al., 2013), Biterm Topic Model.
- LDA (Blei et al., 2003).

In all the models, we set the number of topics to be 20. For LCTM (resp. nI-ITM), we set the number of *latent concepts* (resp. *constraints*) to be 1000. See appendix C for the detail of hyperparameter settings.

4.2 Document Clustering

To demonstrate that LCTM results in a superior representation of short documents compared to the baselines, we evaluated the performance of each model on a document clustering task. We used a learned topic proportion as a feature for each document and applied k-means to cluster the documents. We then compared the resulting clusters to the actual newsgroup labels. Clustering performance is measured by Adjusted Mutual Information (AMI) (Manning et al., 2008). Higher AMI indicates better clustering performance. Figure 3 illustrates the quality of clustering in terms of Gaussian variance parameter σ^2. We see that setting $\sigma^2 = 0.5$ consistently obtains good clustering performance for all the datasets with varying sizes. We therefore set $\sigma^2 = 0.5$ in the later evaluation. Figure 4 compares AMI on four topic models. We see that LCTM outperforms the topic models without word embeddings. Also, we see that LCTM performs comparable to LFLDA and nl-cLDA, both of which incorporate information of word embeddings to aid topic inference. However, as we will see in the next section, LCTM can

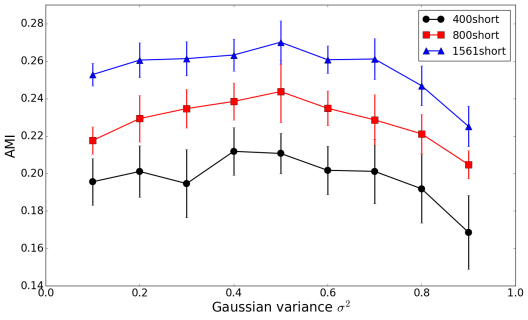


Figure 3: Relationship between σ^2 and AMI.

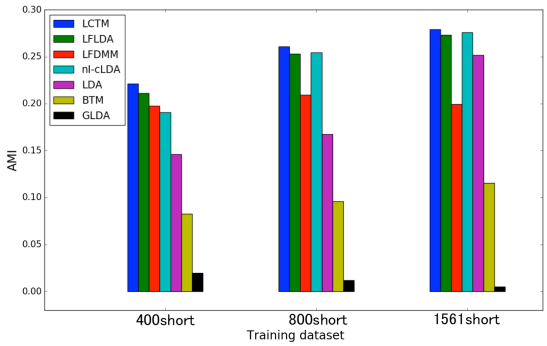


Figure 4: Comparisons on clustering performance of the topic models.

better handle OOV words in held-out documents than LFLDA and nl-cLDA do.

4.3 Representation of Held-out Documents with OOV words

To show that our model can better predict topic proportions of documents containing OOV words than other topic models, we conducted an experiment on a classification task. In particular, we infer topics from the training dataset and predicted topic proportions of held-out documents using collapsed Gibbs sampler. With the inferred topic proportions on both training dataset and held-out documents, we then trained a multi-class classifier (multi-class logistic regression implemented in sklearn[2] python module) on the training dataset and predicted newsgroup labels of the held-out documents.

We compared classification accuracy using LFLDA, nI-cLDA, LDA, GLDA, LCTM and a variant of LCTM (LCTM-UNK) that ignores OOV in the held-out documents. A higher classification accuracy indicates a better representation of unseen documents. Table 2 shows the proportion of OOV words and classification accuracy

[2]See `http://scikit-learn.org/stable/`.

Training Set	400short	800short	1561short
OOV prop	0.348	0.253	0.181
Method	Classification Accuracy		
LCTM	**0.302**	**0.367**	**0.416**
LCTM-UNK	0.262	0.340	0.406
LFLDA	0.253	0.333	0.410
nI-cLDA	0.261	0.333	0.412
LDA	0.215	0.293	0.382
GLDA	0.0527	0.0529	0.0529
Chance Rate	0.0539	0.0539	0.0539

Table 2: Proportions of OOV words and classification accuracy in the held-out documents.

of the held-out documents. We see that LCTM-UNK outperforms other topic models in almost every setting, demonstrating the superiority of our method, even when OOV words are ignored. However, the fact that LCTM outperforms LCTM-UNK in all cases clearly illustrates that LCTM can effectively make use of information about OOV to further improve the representation of unseen documents. The results show that the level of improvement of LCTM over LCTM-UNK increases as the proportion of OOV becomes greater.

5 Conclusion

In this paper, we have proposed LCTM that is well suited for application to short texts with diverse vocabulary. LCTM infers topics according to document-level co-occurrence patterns of *latent concepts*, and thus is robust to diverse vocabulary usage and data sparsity in short texts. We showed experimentally that LCTM can produce a superior representation of short documents, compared to conventional topic models. We additionally demonstrated that LCTM can exploit OOV to improve the representation of unseen documents. Although our paper has focused on improving performance of LDA by introducing the *latent concept* for each word, the same idea can be readily applied to other topic models that extend LDA.

Acknowledgments

We thank anonymous reviewers for their constructive feedback. We also thank Hideki Mima for helpful discussions and Paul Thompson for insightful reviews on the paper. This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Sitaram Asur and Bernardo A Huberman. 2010. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Weizheng Chen, Jinpeng Wang, Yan Zhang, Hongfei Yan, and Xiaoming Li. 2015. User based aggregation for biterm topic model. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, 2:489–494.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. 2015. Gaussian LDA for topic models with word embeddings. In *Proceedings of the 53nd Annual Meeting of the Association for Computational Linguistics*, pages 795–804.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235.

Yuening Hu, Jordan L. Boyd-Graber, Brianna Satinoff, and Alison Smith. 2014. Interactive topic modeling. *Machine Learning*, 95(3):423–469.

Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. 2008. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

James Petterson, Wray Buntine, Shravan M Narayanamurthy, Tibério S Caetano, and Alex J Smola. 2010. Word features for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 1921–1929.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579-2605):85.

Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM.

Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456. International World Wide Web Conferences Steering Committee.

A Dataset Preprocessing

We preprocessed the 20Newsgroups as follows: We downloaded bag-of-words representation of the corpus available online[3]. Stop words[4] and words that were not covered in the GloVe were both removed. After the preprocessing, we extracted short texts containing less than 50 words for training datasets. We created three training datasets with varying numbers of documents, and one held-out dataset. Each dataset was balanced in terms of the proportion of documents belonging to each newsgroup.

B Non-Interactive Contained LDA (nI-cLDA)

We describe nI-cLDA, a variant of interactive topic model (Hu et al., 2014). nl-cLDA is non-interactive in the sense that *constraints* are inferred from the word embeddings instead of being interactively identified by humans. In particular, we apply k-means to word embeddings to cluster words. Each resulting cluster is then regarded as a *constraint*. In general, *constraints* can be different from topic to topic. Let $r_{k,w}$ be a *constraint* of topic k which word w belongs to. The generative process of nl-cLDA is as follows. It is essentially the same as (Hu et al., 2014)

1. For each topic k
   (a) Draw a topic *constraint* distribution $\boldsymbol{\phi}_k \sim \text{Dirichlet}(\boldsymbol{\beta})$.
   (b) For each *constraint* s of topic k
      i. Draw a *constraint* word distribution $\boldsymbol{\pi}_{k,s} \sim \text{Dirichlet}(\boldsymbol{\gamma})$.
2. For each document d
   (a) Draw a document topic distribution $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha})$.
   (b) For the i-th word $w_{d,i}$ in document d
      i. Draw its topic assignment $z_{d,i} \sim \text{Categorical}(\boldsymbol{\theta}_d)$.

[3] `http://qwone.com/~jason/20Newsgroups/`
[4] Available at `http://www.nltk.org/`

ii. Draw its *constraint* $l_{d,i} \sim \text{Categorical}(\boldsymbol{\phi}_{z_{d,i}})$.

iii. Draw a word $w_{d,i} \sim \text{Categorical}(\boldsymbol{\pi}_{z_{d,i},l_{d,i}})$.

Let V be the set of vocabulary. We note that $\boldsymbol{\pi}_{k,s}$ is a multinomial distribution over $W_{k,s}$, which is a subset of V, defined as $W_{k,s} \equiv \{w \in V \mid r_{k,w} = s\}$. $W_{k,s}$ represents a constrained set of words that are conceptually related to each other under topic k.

In our application, we observe documents and *constraints* for each topic, and wish to infer posterior distributions over all the hidden variables. We apply collapsed Gibbs sampling for the approximate inference. For the detail of the inference, see (Hu et al., 2014).

C Hyperparameter Settings

For all the topic models, we used symmetric Dirichlet priors. The hyperparameters were set as follows: for our model (LCTM and LCTM-UNK), nI-cLDA and LDA, we set $\boldsymbol{\alpha} = 0.1$ and $\boldsymbol{\beta} = 0.01$. For nl-cLDA, we set the parameter of Dirichlet prior for *constraint*-word distribution ($\boldsymbol{\gamma}$ in appendix B) as 0.1. Also for our model, we set, $\sigma_0^2 = 1.0$ and $\boldsymbol{\mu}$ to be the average of word vectors. We randomly initialized the topic assignments in all the models. Also, we initialized the *latent concept* assignments using k-means clustering on the word embeddings. The k-means clustering was implemented using sklearn[5] python module. We set M (number of nearest *concepts* to sample from) to be 300, and updated the nearest *concepts* every 5 iterations. For LFLDA, LFDMM, BTM and Gaussian LDA, we used the original implementations available online[6] and retained the default hyperparameters.

We ran all the topic models for 1500 iterations for training, and 500 iterations for predicting held-out documents.

[5]See `http://scikit-learn.org/stable/`.

[6]LFTM: `https://github.com/datquocnguyen/LFTM`
BTM: `https://github.com/xiaohuiyan/BTM`
GLDA: `https://github.com/rajarshd/Gaussian_LDA`