Modeling Text using the Continuous Space Topic Model with Pre-Trained Word Embeddings

Seiichi Inoue[1], **Taichi Aida**[1], **Mamoru Komachi**[1], **Manabu Asai**[2]
[1]Tokyo Metropolitan University, [2]Soka University
{inoue-seiichi, aida-taichi}@ed.tmu.ac.jp
komachi@tmu.ac.jp, m-asai@soka.ac.jp

Abstract

In this study, we propose a model that extends the continuous space topic model (CSTM), which flexibly controls word probability in a document, using pre-trained word embeddings. To develop the proposed model, we pre-train word embeddings, which capture the semantics of words and plug them into the CSTM. Intrinsic experimental results show that the proposed model exhibits a superior performance over the CSTM in terms of perplexity and convergence speed. Furthermore, extrinsic experimental results show that the proposed model is useful for a document classification task when compared with the baseline model. We qualitatively show that the latent coordinates obtained by training the proposed model are better than those of the baseline model.

1 Introduction

Topic models are statistical models that automatically extract latent topics in documents from a text corpus. Topic models have been used in various applications within and outside of natural language processing. Such applications include information retrieval (Wei and Croft, 2006), collaborative filtering (Marlin, 2003), author identification (Rosen-Zvi et al., 2012), and opinion extraction (Lin et al., 2011).

The latent Dirichlet allocation (LDA) (Blei et al., 2003), which is a representative method for topic modeling, assumes that each document has a latent topic. It uses an unobservable random variable called the latent topic to formulate the factors that produce a set of words that are statistically likely to co-occur. Unlike the LDA, the continuous space topic model (CSTM) (Mochihashi et al., 2013) models documents without using intermediate variables, such as latent topics. Specifically, the CSTM is formulated by introducing latent coordinates of words and considering a function that follows a Gaussian process in the same space to represent the importance of a word in a document. In the LDA, the probability distribution of words is fixed, and the probability of words is controlled by the topic distribution. Therefore, it is not possible to change the probability distribution of words according to each document and thus the text cannot be modeled in a fine-grained way. By contrast, the CSTM controls the probability of words based on the latent coordinates of the words and the function representing the meaning of the document. Hence, it is possible for the CSTM to dynamically change the word distribution according to the document. Additionally, the CSTM outperforms conventional topic models, such as the LDA, in terms of perplexity.

As mentioned above, the CSTM models documents using word embeddings; however, the structure of the model is such that the word embeddings (latent coordinates) are free parameters. Therefore, the estimation of the model is time-consuming because of the large number of parameters. In addition, the only information used for the estimation of the word embeddings is the frequency of words, which makes it difficult to capture the semantics of words.

In this study, we propose a new method in which the latent coordinates of words, which are one of the free parameters of the CSTM, are learned in advance using word2vec (Mikolov et al., 2013), and the learned distributed representation of the words are introduced into the CSTM. As in the Gaussian LDA (Das et al., 2015), when we use the word embeddings that capture the semantics of words and provide them as prior information to the model, we can expect improved performance and faster convergence. In the experiments, we use English and Japanese corpora to compare the proposed method with the baseline CSTM in terms of perplexity and convergence speed. We also perform a document classification task to evaluate the quality of the document representations that are

Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop, pages 138–147

learned by our model. In the discussion, we use the trained model to investigate the importance of words in documents and evaluate the trained model qualitatively. Additionally, we visualize the latent coordinates of words and documents in the same space.

The main contributions of this study are as follows:

- We propose a CSTM-based model that can estimate parameters faster and obtain useful document representation using pre-trained word embeddings.
- Intrinsic experiments using English and Japanese corpora show that the proposed model exhibits a superior performance over the baseline model in terms of perplexity and convergence speed.
- Extrinsic experimental results show that document embeddings obtained by the proposed model are useful for document classification.

2 Related Work

2.1 Word Embeddings and Topic Models

There are several studies that aimed to improve the performance of topic models by using a distributed representation of words. Das et al. (2015) proposed the Gaussian LDA (G-LDA), which uses a multivariate Gaussian distribution in the same space of word embeddings to estimate topics in the embedding space. Compared with the LDA, it has high coherence (Chang et al., 2009) because it introduces prior knowledge of semantics of words by using pre-trained word embeddings. Recently, Dieng et al. (2020) proposed the embedded topic model (ETM). The ETM models each word with a categorical distribution whose natural parameter is the inner product between the embedding of word and an embedding of its assigned topic. It outperformed traditional topic models including the LDA.

However, both topic models use latent topics to model the documents. The G-LDA defines latent topics as multivariate Gaussian distribution, and the ETM uses topic embeddings for formulating the word probability. Therefore, those topic models hardly control word probability directly depending on a document. In Section 2.2, we introduce the CSTM, which can directly control word probability in a document.

2.2 Continuous Space Topic Model

In the CSTM, the probability of a word is modeled through the Polya distribution, which is a compound distribution of the Dirichlet and multinomial distributions, to account for the burstiness of language (Doyle and Elkan, 2009). We denote $\boldsymbol{y} = (y_1, y_2, \ldots, y_V)$ as the frequency of each word in the document, $\boldsymbol{w}$. The Polya distribution is defined as follows:

$$p(\boldsymbol{y}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_v \alpha_v)}{\Gamma(\sum_v (\alpha_v + y_v))} \prod_v \frac{\Gamma(\alpha_v + y_v)}{\Gamma(\alpha_v)}, \tag{1}$$

where $\boldsymbol{\alpha}$ represents the concentration parameter of the Polya distribution. We assume that each word, w_v, has latent coordinates $\phi(w_v) \sim \mathcal{N}(0, I_d)$ in the d-dimension. To increase the probability of semantically related words in each document, we generate a function that follows a Gaussian process with a mean of zero in the same latent space:

$$f \sim \mathrm{GP}(0, \mathrm{K}), \tag{2}$$

where K represents the kernel matrix, and in this case, it is an inner product kernel: $\mathrm{K}_{ij} = k(w_i, w_j) = \phi(w_i)^T \phi(w_j)$. A Gaussian process (Rasmussen and Williams, 2006) is a stochastic process that generates a random regression function, where the closer $k(w_i, w_j)$ is, the closer the corresponding outputs, $f(w_i), f(w_j)$, will be. Intuitively, f represents "what we want to say in this document." The concentration parameter, α_v, of the Polya distribution is then modeled to be larger according to its function value:

$$\alpha_v \propto \alpha_0 G_0(w_v) \exp(f(w_v)), \tag{3}$$

where $\alpha_0 \sim \mathrm{Ga}(a_0, b_0)$ is a free parameter, and $\mathrm{Ga}(a_0, b_0)$ indicates the gamma distribution. Additionally, $G_0(w_v) \sim \mathrm{PY}(\beta, \gamma)$ represents the "default" probability of word w_v, and $\mathrm{PY}(\beta, \gamma)$ denotes the Pitman-Yor process. In practice, the maximum likelihood estimator, $\#(w_v)/\sum_i \#(w_i)$, used as $G_0(w_v)$ ($\#(w_v)$ is the frequency of the word w_v in all documents). Based on this, the generation process of the CSTM that generates N documents is as follows:

1. Draw $\alpha_0 \sim \mathrm{Ga}(a_0, b_0)$.
2. Draw $G_0 \sim \mathrm{PY}(\beta, \gamma)$. (In practice, maximum likelihood estimator is used.)
3. For $v = 1 \ldots V$,

- Draw $\phi(w_v) \sim \mathcal{N}(0, I_d)$.

4. For $n = 1 \dots N$,
   - Draw $f_n \sim \mathrm{GP}(0, \mathrm{K})$.
   - For $v = 1 \dots V$,
     - Set $\alpha_v = \alpha_0 G_0(w_v) e^{f_n(w_v)}$.
   - Draw $\boldsymbol{w} \sim \mathrm{Polya}(\boldsymbol{\alpha})$.

3 Proposed Method

3.1 Word Embeddings

Word2vec (Mikolov et al., 2013) is a probabilistic model for learning distributed representations that capture the semantics of words based on the distributional hypothesis (Harris, 1954). The continuous bag-of-words (CBOW) model, which is one of the learning methods of word2vec, obtains word embeddings by maximizing the predicted probability of the target word, w_t:

$$p(w_t|C_{w_t}) \propto \exp(\eta(w_t)^T \tilde{\eta}(C_{w_t})), \quad (4)$$

where $C_{w_t} = \{w_{t\pm i} | 1 \leq i \leq \delta\}$ represents the set of nearby context words, δ is the context window width, and $\tilde{\eta}(C_{w_t}) := |C_{w_t}|^{-1} \sum_{w \in C_{w_t}} \eta(w)$ denotes the average vector of all context word vectors.

We use the CBOW model to learn word embeddings. In this study, we used a relatively large context window of $\delta = 10$ to learn the topical information (Bansal et al., 2014). In general, it has been shown that the quality of word embeddings improves by centering (Hara et al., 2015; Mu and Viswanath, 2018). Accordingly, acquired distributed representations of the word, $\eta(w_1), \eta(w_2), \dots, \eta(w_V)$, are centered and normalized as follows:

$$\psi(w_v) = \tau S^{-\frac{1}{2}} \left\{ \eta(w_v) - V^{-1} \sum_i \eta(w_i) \right\}, \quad (5)$$

where S is a normalization constant, and defined as follows:

$$S = V^{-1} \sum_i \eta(w_i)^T \eta(w_i). \quad (6)$$

In addition, τ is a hyperparameter that controls the variance of word embeddings, and in this study, we simply set $\tau = d^{-1/2}$.

3.2 Modeling Text with Pre-trained Word Embeddings

Next, as in Mochihashi et al. (2013), we define the function that follows the Gaussian process, whose mean is zero and kernel function is $k(w_i, w_j) = \psi(w_i)^T \psi(w_j)$, in the latent space consisting of the word distributed representations obtained using Eq. (5):

$$f \sim \mathrm{GP}(0, \mathrm{K}_\psi). \quad (7)$$

However, because f is, in principle, infinite in dimension and difficult to estimate directly, we introduce an auxiliary variable representing the latent coordinates of the document in the word latent space, similar to the discrete infinite logistic normal distribution (Paisley et al., 2011), which introduces latent coordinates to correlate between topics in the LDA framework:

$$u \sim \mathcal{N}(0, I_d). \quad (8)$$

We summarize the latent coordinates of the words as $\Psi = (\psi(w_1), \psi(w_2), \cdots \psi(w_V))^T$, and we can obtain the distribution of $f = \Psi u$ by marginalizing u as follows:

$$f|\Psi \sim \mathrm{GP}(0, \Psi^T \Psi) = \mathrm{GP}(0, \mathrm{K}_\psi). \quad (9)$$

f follows the same Gaussian process as expressed in Eq. (7).

Therefore, in the proposed method, we define the Gaussian process representing the meaning of the document using the document vector, u, which is in the same latent space as the word vector:

$$f(w_v) \propto \psi(w_v)^T u. \quad (10)$$

Next, we define α_v as in Eq. (3):

$$\alpha_v \propto \alpha_0 G_0(w_v) \exp(\psi(w_v)^T u), \quad (11)$$

and model the probability of a word using the Polya distribution in Eq. (1).

3.3 Bayesian Markov Chain Monte Carlo (MCMC) Estimation

By combining N documents as $\boldsymbol{D} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_N)$, we can obtain the joint distribution of α_0 and $\boldsymbol{\alpha}$ as follows:

$$p(\alpha_0, \boldsymbol{\alpha}|\boldsymbol{D}) \propto \prod_n p(\boldsymbol{y}_n|\alpha_0, G_0, f_n) p(\alpha_0) p(f_n|\psi). \quad (12)$$

Algorithm 1: MCMC Procedure

Initialize $u \sim \mathcal{N}(0, I_d)$
Initialize $\alpha_0 = 1$
for $j = 1 \dots J$ **do**
  for $n = \text{randperm}(1 \dots N)$ **do**
    Draw $u'_n \sim \mathcal{N}(u_n, \sigma_u^2 I)$
    Draw $\upsilon \sim \text{Uniform}(0, 1)$
    if $\mathcal{A}(u'_n) \geq \upsilon$ **then**
      Update $u_n = u'_n$
    end
  end
  Draw $z \sim \mathcal{N}(0, \sigma_{\alpha_0}^2)$
  Set $\alpha'_0 = \alpha_0 \cdot \exp(z)$
  Draw $\upsilon \sim \text{Uniform}(0, 1)$
  if $\mathcal{A}(\alpha'_0) \geq \upsilon$ **then**
    Update $\alpha_0 = \alpha'_0$
  end
end

Figure 1: The MCMC algorithm of proposed model.

However, because $\boldsymbol{\alpha}$ changes only through the document vector, u, in Eq. (10), in the proposed model, the joint distribution of the estimated parameters, α_0 and $\boldsymbol{u} = (u_1, u_2, \dots, u_N)$, is denoted as follows:

$$p(\alpha_0, \boldsymbol{u}|\boldsymbol{D}) \propto \prod_n p(\boldsymbol{y}_n|\alpha_0, G_0, \psi, u_n) p(\alpha_0) p(u_n). \quad (13)$$

For model estimation, we use the random walk Metropolis-Hastings (MH) algorithm to avoid the problem of local optima, as demonstrated by Mochihashi et al. (2013). [1] We show the MCMC algorithm of proposed model in Figure 1. The estimating parameters are α_0, and the document vector u in Eq. (11). The candidates for each parameter are generated using the following proposal distribution:

$$z \sim \mathcal{N}(0, \sigma_{\alpha_0}^2), \quad (14)$$
$$\alpha'_0 = \alpha_0 \cdot \exp(z), \quad (15)$$
$$u' \sim \mathcal{N}(u, \sigma_u^2 I). \quad (16)$$

[1] We attempted the Hamiltonian MCMC algorithm (Neal et al., 2011) using the gradient of the posterior distribution. However, owing to the high computational cost and need for numerical differentiation, we only used the random walk MH algorithm in this study for the experiments.

Table 1: Statistics for each corpus.

Data	Docs	Vocabulary	Words
NIPS	1,740	37,822	3,971,243
CSJ	3,302	20,001	5,433,871
Mainichi	10,000	38,070	8,070,838

Table 2: Test set perplexity for each corpus.

Data	Ours	CSTM	ETM
NIPS	**980.682**	1148.386	2872.731
CSJ	**288.157**	300.967	1017.658
Mainichi	**362.706**	405.199	2602.808

We also adopt candidates according to the acceptance probability of the following likelihood ratio:

$$\mathcal{A}(\alpha'_0) = \min\left\{1, \frac{\prod_n p(\boldsymbol{y}_n|\alpha')\text{Ga}(\alpha'_0|a_0, b_0)}{\prod_n p(\boldsymbol{y}_n|\alpha)\text{Ga}(\alpha_0|a_0, b_0)}\right\}, \quad (17)$$

$$\mathcal{A}(u') = \min\left\{1, \frac{p(\boldsymbol{y}_n|\alpha')p(u'|0, I_d)}{p(\boldsymbol{y}_n|\alpha)p(u|0, I_d)}\right\}. \quad (18)$$

In this study, we set $\sigma_{\alpha_0} = 0.2$ and $\sigma_u = 0.01$, which are the random walk widths that control efficiency of training, based on the results of preliminary experiments.

4 Experiments

4.1 Corpora

In the experiments, we used the Neural Information Processing Systems (NIPS) [2], which is an English corpus, Corpus of Spontaneous Japanese (CSJ) and Mainichi Newspaper (10,000 randomly selected articles from 2013), which are Japanese corpora. For Japanese, we preprocessed texts using MeCab[3] with IPADic. In all the corpora, words with a frequency of less than five were excluded from the training data. The statistics for each corpus are listed in Table 1.

4.2 Intrinsic Evaluation

To evaluate the performance of topic models, we computed the perplexity of the proposed model, the CSTM and the ETM. Similar to the work of Wallach et al. (2009), we randomly selected 80% of the

[2] https://cs.nyu.edu/~roweis/data.html
[3] https://taku910.github.io/mecab/

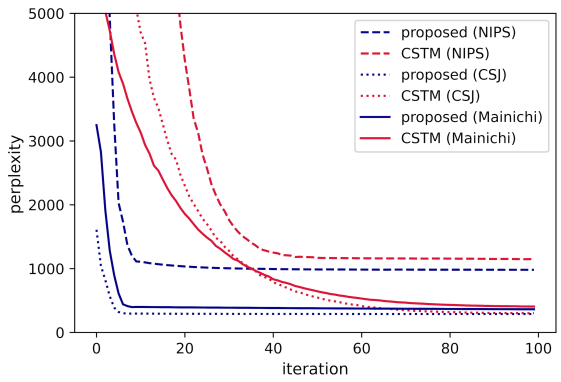

Figure 2: Test set perplexity of the proposed model and the CSTM.

words in each document as training data and calculated the perplexity on the remaining 20% of the words. For the evaluation in the proposed model and the CSTM, we varied the latent dimension size by 10, 20, 50, and 100 and reported the best score on test data. For the evaluation in the ETM, we set the local learning rate to 0.002 and the weight decay parameter to 1.2×10^{-6}, and then selected the model which reported the best validation score by varying the number of topics by 10, 20, 50, and 100.

Perplexity The perplexity of the proposed model, the CSTM and the ETM computed for each corpus is shown in Table 2. The proposed method outperforms the CSTM and the ETM in terms of perplexity for all three corpora. Compared to the CSTM, the proposed method naturally has higher performance because it has the topical information from pre-trained word embeddings. The ETM cannot directly control the word probability in a document because it uses topic embeddings for formulating the word probability, so the proposed model, which can control the word probability flexibly, performs better in terms of predictive power.

Convergence Speed Figure 2 shows the perplexity convergence of the proposed model and the CSTM. The proposed model only takes less than ten iteration to converge, though the CSTM takes fifty to hundred iteration. The proposed model also outperforms the CSTM in terms of convergence speed on all corpora because it has topical information as prior knowledge from the pre-trained word embeddings.

Table 3: Mean classification accuracy on the CSJ corpus using learned embeddings.

Models	Accuracy	P-value
CSTM	0.704	0.000
Ours	**0.866**	
word2vec	0.917	0.111
Ours w/ word2vec	**0.928**	

4.3 Extrinsic Evaluation

To evaluate the quality of representations of the documents that are learned by our model, we perform a document classification task. We evaluate the performance of the proposed model by comparing it with the performances of CSTM and word2vec.

Settings In this experiment, we use the one-versus-one support vector machine implemented in scikit-learn [4]. The data was split between training, 90% and testing, 10%. For the tuning parameter C, which is one of the parameters controlling the extent of penalty, and γ, which is the parameter of RBF kernel, we execute grid search by a 10-fold cross validation on the training data and select the best models in terms of accuracy. For other parameters, we use the default values set by scikit-learn.

We define the features as follows: For the CSTM, we use the document vectors. For word2vec, we use the mean vector of word vectors in the document. For the proposed model, we use the document vector (denoted "Ours") and the concatenation of the mean vector of word vectors and document vector (denoted "Ours w/ word2vec"). Also, we apply the paired t-test to compare the performance between the proposed models and the baseline models. A confidence interval of 95% was considered to identify a significant difference between two compared models.

Results Table 3 shows the classification accuracy on the CSJ corpus using each feature. For document classification using only document vector obtained from the proposed model, we can see that it significantly ($p < 0.05$) outperformed the CSTM but is slightly inferior to word2vec. However, when we use the document vector obtained from the proposed model and the average vector of word vectors obtained from word2vec, the accuracy is better than that of word2vec, although the

[4] https://scikit-learn.org/stable/

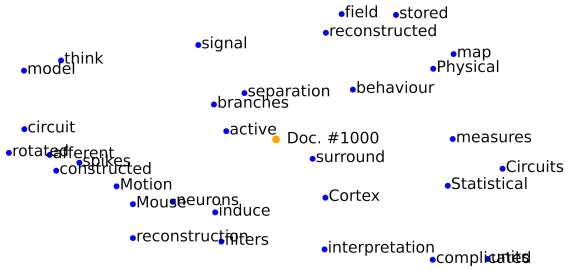

Figure 3: The visualization of reduced embedding space around the 1000th document "The Role of Activity in Synaptic Competition at the Neuromuscular Junction." Words are colored as blue and document as orange.

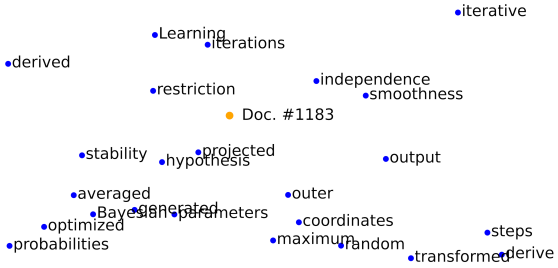

Figure 4: The visualization of reduced embedding space around the 1183rd document "Bayesian Model Comparison by Monte Carlo Chaining." Words are colored as blue and document as orange.

difference is not statistically significant. We will analyze the classification results in detail in Section 5.3.

5 Discussion

5.1 Visualizing Word and Document Embeddings

In the proposed model and the CSTM, word vectors and document vectors are located in the same space, so we can observe the relationships between a word and a document at the same time by visualizing embedding space. We execute the PCA on vectors of words with high frequency and all documents to reduce dimensionality.

The reduced word and document vectors obtained by the proposed model are shown in Figure 3 and 4, and we additionally show the visualization of full embedding space, including those documents, in Figure 5 in Appendix. In these figures, two representative documents are shown—a neuroscience article titled "The Role of Activity in Synaptic Competition at the Neuromuscular Junction," and a computer science article titled "Bayesian Model Comparison by Monte Carlo Chaining." Figure 3 enlarges reduced embedding space around the neuroscience article that shows words such as "signal," "neurons," and "Cortex." Figure 4 enlarges reduced embedding space around the computer science article that shows words such as "Bayesian," "iterations," "optimized," and "parameters."

From these figures, we can see that words related to topics of the article are correctly located. Therefore, we can see that the proposed model can locate document vectors appropriately in the word embedding space, which enhances the performance of the model.

5.2 Analyzing the Importance of Words in a Document

In the proposed model and the CSTM, the document vectors are defined in the same space as the word vectors. Therefore, based on the inner product of the document vector and the word vector, we can quantitatively measure the importance of words in a document, such as words that are likely to appear in a document and words that are not. For the calculation, we used the document and word vectors of all words in the training vocabulary, including words that do not actually appear in the document.

For example, for the proposed model and the CSTM, we used the neuroscience article in the NIPS corpus to compute the ranking of topic-related and topic-unrelated words in the document. Tables 4 and 5 show the results of the proposed model and the CSTM, respectively. We show the words that actually appear in the document in bold. Although both the results of the CSTM and the proposed model contain the words appearing in the document, we can see that the proposed model comparatively captures the topic of the document and gives high score to topic-related words. The topic-related words obtained using the CSTM accounted for a few words that were related to the topic of the document, whereas those obtained by using the proposed model accounted for a significant number of words that were related to the topic of the document, such as "axon," "synapses," and "nervous." This means that the probability of such words in the document will be reflected to a greater extent. Moreover, we observed that words among the topic-unrelated words obtained by applying the proposed model were not related to the topic of the document. Such words include "Euclidean,"

Table 4: Top 30 topic-related words and topic-unrelated words from the NIPS article, "The Role of Activity in Synaptic Competition at the Neuromuscular Junction," using the proposed model. The words that appear in the document are shown in bold.

e^f	Word	e^f	Word
113.7901	**axon**	0.0862	vector
27.7607	**synapses**	0.1197	convex
22.7449	**nervous**	0.1267	hidden
21.7567	brain	0.1280	Fisher
19.4746	**synaptic**	0.1306	derivative
16.0369	**interaction**	0.1308	Euclidean
15.9976	**mechanisms**	0.1332	classifiers
15.5423	fiber	0.1357	norm
15.4603	**stimulation**	0.1359	sigmoidal
15.0863	**presynaptic**	0.1420	observable
14.7511	**sites**	0.1476	gradient
14.6049	animal	0.1565	regression
14.2858	ocular	0.1582	computes
13.9519	interneurons	0.1620	corrupted
13.7734	**areas**	0.1624	squared
13.5084	**role**	0.1643	sampled
13.3584	**postsynaptic**	0.1645	minimized
13.3000	**plasticity**	0.1710	Gaussian
12.9953	inhibition	0.1809	speaker
12.8826	**dominance**	0.1818	discrete
12.8527	**muscle**	0.1843	unknown
12.7587	recordings	0.1909	defined
12.5784	**formation**	0.1910	feature
12.5326	**terminal**	0.1920	written
12.4104	growth	0.1927	LMS
12.2916	pathway	0.1971	PCA
12.0274	caused	0.2029	piecewise
11.8988	cues	0.2065	perceptron
11.6562	**effects**	0.2089	entropy
11.5566	**activated**	0.2138	bounds

Table 5: Top 30 topic-related words and topic-unrelated words from the NIPS article, "The Role of Activity in Synaptic Competition at the Neuromuscular Junction," using the CSTM. The words that appear in the document are shown in bold.

e^f	Word	e^f	Word
7.5986	adding	0.2744	silicon
7.0567	**extent**	0.3063	inequality
6.8850	relatively	0.3491	template
6.2375	recording	0.3565	schedule
6.0914	randomly	0.3582	ICA
5.9904	placed	0.3622	head
5.9894	**other**	0.3811	speaker
5.8748	specified	0.4120	filter
5.8090	write	0.4200	MLP
5.7228	adapted	0.4301	spin
5.1464	**terms**	0.4328	gate
5.0912	speed	0.4355	memory
5.0879	explicitly	0.4355	faces
4.9648	**when**	0.4386	orientation
4.8808	**demonstrate**	0.4503	PCA
4.8080	**range**	0.4520	nucleus
4.7802	share	0.4523	expansion
4.7197	section	0.4543	almost
4.6721	complicated	0.4552	functions
4.6541	partial	0.4593	variational
4.6538	conditions	0.4634	gates
4.6462	approximately	0.4715	boolean
4.6417	**actually**	0.4726	quantization
4.6161	practice	0.4758	contour
4.6149	**journal**	0.4816	Viterbi
4.6034	recognition	0.4845	chip
4.5872	overall	0.4899	pulses
4.5752	basic	0.4918	radial
4.5430	**single**	0.5009	MAP
4.5222	**theoretical**	0.5024	multilayer

"gradient," and "regression." We believe that this is because, unlike the CSTM, the proposed model has prior knowledge of the topical information of words, thereby facilitating the estimation of document vectors that capture a set of topically similar words.

5.3 Error Analysis of Document Classification

Table 6 shows the classification accuracy for eight category labels using each feature. The proposed model outperforms the CSTM substantially in all categories.

For example, the classification of "Speech Processing," the CSTM misclassified some of the documents as "Linguistics," "Psychology," and "Artificial Intelligence," while the proposed model classified almost all of the documents as "Speech Processing" except for some of the documents labeled "Linguistics." We find that the CSTM misclassified one of the documents in "Speech Processing," which discusses statistical methods in detail, as "Psychology," while the proposed model classified it correctly. The CSTM models word co-occurrence on a document-by-document basis as in Eq. 3, though multiple topics might exist in a document. Therefore, the document vectors obtained by the CSTM do not have the information of the semantic difference between psychology and statistics.

Table 6: Classification accuracy on the CSJ corpus for each category using learned embeddings.

Category	Count	CSTM	Ours	word2vec	Ours w/ word2vec
Speech Processing	413	0.761	**0.912**	0.956	**0.971**
Cosmology	248	1.000	1.000	1.000	1.000
Biology	247	1.000	1.000	1.000	1.000
Linguistics	206	0.452	**0.786**	0.790	**0.857**
Psychology	141	0.393	**0.721**	**0.857**	0.843
Artificial Intelligence	120	0.358	**0.592**	**0.825**	0.817
Language Education	62	0.417	**0.833**	**0.833**	0.817
Sociology	28	0.167	**0.400**	0.700	0.700
Total	1465	0.704	**0.866**	0.917	**0.928**

In contrast, the proposed model models word co-occurrence based on the local context of the neighborhood, where topics are considered to be somewhat consistent. Therefore, the proposed model can distinguish the word set that tends to appear in the genre of psychology from the genre of statistics in the embedding space. Hence, because the document vectors are estimated in the space where word vectors have the information of the semantic difference between psychology and statistics, the proposed model can distinguish those documents.

6 Conclusion and Future Work

In this study, we introduced the learned distributed representation of words into the CSTM to provide prior knowledge on the semantics of words. In the experiments, we showed that the proposed model outperformed the baseline method in terms of perplexity and convergence speed. Also, we showed that the proposed model is useful for a document classification task compared with the baseline model. Additionally, we showed that the document vectors obtained by training the model are superior through visualization of the embedding space and analysis of importance of words in a document.

In the future, we would like to investigate better ways of estimating the model, including optimization by applying the Hamiltonian MCMC algorithm, which was not used in this study. Furthermore, we would like to use contextualized word embeddings obtained by ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019) in the proposed model.

References

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 809–815.

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

Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. *Advances in Neural Information Processing Systems*, 22:288–296.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Gabriel Doyle and Charles Elkan. 2009. Accounting for burstiness in topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 281–288.

Kazuo Hara, Ikumi Suzuki, Masashi Shimbo, Kei Kobayashi, Kenji Fukumizu, and Miloš Radovanović. 2015. Localized centering: Reducing hubness in large-sample data. In *Proceedings of*

the AAAI Conference on Artificial Intelligence, volume 4, pages 2645–2651.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Chenghua Lin, Yulan He, Richard Everson, and Stefan Ruger. 2011. Weakly supervised joint sentiment-topic detection from text. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):1134–1145.

Benjamin M Marlin. 2003. Modeling user rating profiles for collaborative filtering. *Advances in Neural Information Processing Systems*, 16:627–634.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations*.

Daichi Mochihashi, Kazuyoshi Yoshii, and Masataka Goto. 2013. Modeling text through Gaussian processes. In *Information Processing Society of Japan Special Interest Groups Technical Report*, volume 2013-NL-213, pages 1–8.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective post-processing for word representations. In *6th International Conference on Learning Representations, ICLR 2018*.

Radford M Neal et al. 2011. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2.

John Paisley, Chong Wang, and David Blei. 2011. The discrete infinite logistic normal distribution for mixed-membership modeling. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 74–82.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Carl Edward Rasmussen and Christopher KI Williams. 2006. Gaussian processes for machine learning. *MA: the MIT Press*.

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2012. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, page 487–494.

Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1105–1112.

Xing Wei and W Bruce Croft. 2006. LDA-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185.

A Visualization of Embedding Space

We show the visualization of full embedding space, including neuroscience article and computer science article, in Figure 5.

Figure 5: The visualization of reduced embedding space using the proposed model. Words are colored as blue and documents as orange.