Topic Structure-Aware Neural Language Model: Unified language model that maintains word and topic ordering by their embedded representations

Noriaki Kawamae
NTT COMWARE
Tokyo
kawamae@gmail.com

ABSTRACT

Our goal is to exploit a unified language model so as to explain the generative process of documents precisely in view of their semantic and topic structures. Because various methods model documents in disparate ways, we are motivated by the expectation that coordinating these methods will allow us to achieve this goal more efficiently than using them in isolation; we combine topic models, embedding models, and neural language models. As we focus on the fact that topic models can be shared among, and indeed complement embedding models and neural language models, we propose Word and topic 2 vec (Wat2vec), and Topic Structure-Aware Neural Language Model (TSANL). Wat2vec uses topics as global semantic information and local semantic information as embedding representations of topics and words, and embeds both words and topics in the same space. TSANL uses recurrent neural networks to capture long-range dependencies over topics and words. Since existing topic models demand time consuming learning and have poor scalability, both due to breaking the document's structure such as order of words and topics, TSANL maintains the orders of words and topics as phrases and segments, respectively. TSANL reduces the calculation cost and required memory by feeding topic recurrent neural networks, and topic specific word networks with these embedding representations. Experiments show that TSANL maintains both segments and topical phrases, and so enhances previous models.

KEYWORDS

Word embeddings, Topic Models, Recurrent Neural Network, Neural Language Model

ACM Reference Format:
Noriaki Kawamae. 2019. Topic Structure-Aware Neural Language Model: Unified language model that maintains word and topic ordering by their embedded representations. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313757

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313757

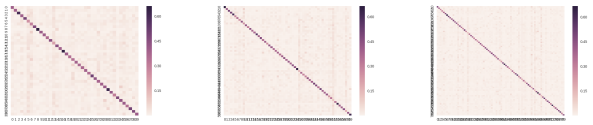

Figure 1: Heat map of topic transition in LDA with (left) #topics=40, (center) #topics=60, (right) #topics=80. Both x and y denote topic id. The higher the probability is, the darker its color is.

1 INTRODUCTION

As many people are expressing their opinions on-line and their volume exceeds the analysis capabilities of any individual or group of individuals, Latent Dirichlet Allocation (LDA) [2] and related topic models have become popular tools for modeling documents. These models uncover the latent structure of a given document collection as topics, and so are enjoying commercial success. Each topic has own multinomial distribution over terms, and has statistical strength and structure that is shared among documents. While their topic representations offer some degree of human interpretability, they are strictly non-sequential models.

As phrases carry more meaning than the sum of the individual words and word meaning depends on context, phrases simplify the human interpretation of topics compared to unigrams. Although some models focus on the order of words [16], they assume that the assignment of a topic is independent from that of other topics. We ran LDA over 2022 ACM papers and show the transition probability between adjacent tokens in Figure 1. This figure shows that 1) these transitions probabilities are asymmetric, and 2) the transition probability to the same topic, p_{ii}, is higher than to other probabilities, $p_{ij}(j \neq i)$, and increases as the #topics increases (e.g. the average of p_{ii} with #topics=40, #topics=60, and #topics=80 is 0.51, 0.57, and 0.61, respectively). Here, we can say that topic appearance and order is affected by their continuity in each document and forms the syntactic structure seen.

Motivated by the above background, we propose both Word and Topic 2 Vec (Wat2Vec) and Topic Structure-Aware Neural Language Model (TSANL) to explain the generative process of documents as based both semantic information and syntactic structures. As we focus on the fact that topic models can be shared among, and indeed complement embedding models [1] and neural language models, we extend embedding models to Wat2vec, and propose TSANL as neural

Table 1: TSANL(+Wat2vec) inherits the interpretability of topic models, the semantic relationships of embedding models, and the prediction power of neural language models: The learning approaches u, and t, denote unified and two-step, respectively.

		Topic model			Embedding models		Neural Language models			
		NTSeg	PDLDA	NOC	CLM	TWE	Topic RNN	TDLM	LLA	TSANL
Reference		[10]	[16]	[12]	[28]	[17]	[4]	[13]	[30]	[this paper]
Semantic	word embeddings	-	-	-	√	√	-	√	√	√
	topic embeddings	-	-	-	√	-	-	√	√	√
Structure	Word	√	√	√	-	-	√	√	-	√
	Topic	-	-	√	-	-	-	-	√	√
embedding learning		-	-	-	u	t	t	u	t	t

language model. Wat2vec uses topic models as global statistical information and local context information as embedding representations of topics and words. Unlike existing topic models, which break the document's structure such as the order of words and topics, TSANL uses these representations with LSTMs [8] to capture their long-range dependencies.

Through experiments, we confirm the key advantages of Wat2vec and TSANL as follows.

Theoretical contribution: Wat2vec embeds both words and topics in the same semantic space, and computes both the contextual word embeddings and topic embeddings for a word and topic given their context. As TSANL marries topic models with neural language models, it maintains the order of words and topics as phrases and segments, respectively.

Practical contribution: As learning phrases and segments is very time-consuming, we develop an efficient sampling technique. TSANL detects candidate phrases and assigns them to the same topic as a constraint in the inference stage.

2 PREVIOUS WORK

2.1 Topic models

To explore word co-occurrence patterns and their order, several topic models form N-grams via a predictive probability distribution for the next word conditioned on the previous words [11, 26, 31]. For example, Phrase Discovering LDA [16] segments a corpus into phrases of varying lengths and assigns topics. NTSeg [10] maintains the document's structure, such as paragraphs and sentences, by employing the notion of word-topics and segment-topics. NOC [12] reveals a tree of topics that captures the semantic relationship between topics from a given corpus as context, and enables each document to maintain its thematic coherence and form N-grams over context. As we are motivated by the observations that a sequence of topics exhibits their dependencies in the form of phrases, we extend these models to preserve the topic order in order to extract the syntactic structure.

2.2 Embedding models

Distributed representations with neural probabilistic language [1] have been proposed to represent words and documents as low-dimensional and dense real value vectors in single semantic spaces, and significant results in many tasks have been demonstrated [23]. Herbelot and Vecchi [7] explored word embeddings and their utility for modeling language semantics. Word2vec [21] is designed to capture a large number of syntactic and semantic word relationships by introducing the continuous Skip-gram model [19]; it achieves better NLP task performance by grouping similar words and gaining distributed representations of words.

Although the probability distributions gained from topic models describe the statistical relationships of word occurrences in the corpus rather than the best choice for feature representation [23], the combination of embedding models and topic models was proposed to represent words in one semantic space. Topical Word Embedding (TWE) [17] forms the topical word embedding by using pre-trained topic structures and concatenating the topical embedding with the word embedding. By integrating this local semantic information with neural language models, our approach captures the long-range dependencies at the embedding level and reduces the computation cost by decreasing the number of parameters.

2.3 Neural language models

While topic models reflect the co-occurrence relationships of words, they fail to sufficiently model the sequential dependency across sentences. Another approach to modeling the sequences of words and topics is employing RNN [20], such as Long Short Term Memory (LSTM) [8]. LSTMs require a large number of parameters, notwithstanding the simplicity of the underlying dynamics, rending the results uninterpretable [30], and models characterize the long range dependencies of a sequence. Topically Driven Neural Language Model (TDLM) [13] has two components, a language model and a topic model, which are jointly trained using neural networks. While Topic RNN [4] can be learned in an end-to-end fashion to directly capture long-range dependencies between words via latent topics, Latent LSTM Allocation (LLA) [30] models sequence of topics using LSTM and generate word similar to LDA. bridges the gap between RNN's and topic models, and model the dynamics of topic evolution. capture semantic meaning in an end-to-end fashion. Our approach uses topic models to learn topics, explore the embeddings of words and topics via learnt topics, and their embedding representation neural language models to maintain phrases, and segments.

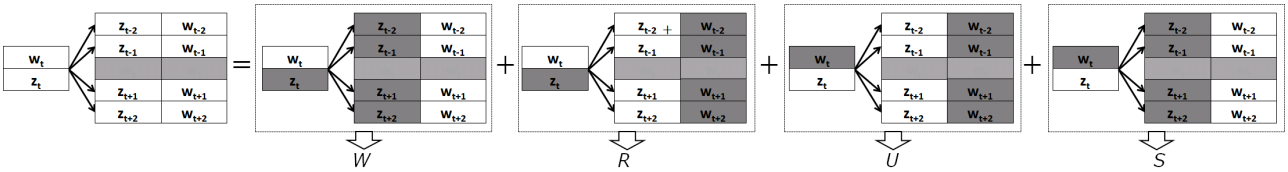

Figure 2: Learning architectures of Skip-gram of Wat2vec: In this figure, w_t is the t-th word token and z_t is the corresponding topic. Skip-gram predicts the surrounding words $(w_{t-2}, w_{t-1}, w_{t+1}w_{t+2})$ and topics $(z_{t-2}, z_{t-1}, z_{t+1}z_{t+2})$ from current word w_t and current topic z_t. We decompose this task into four tasks, where each word and topic predicts the surrounding words and topics individually. Then, Wat2vec learns embeddings using this framework individually with word-word co-occurrence matrix, W, word-topic co-occurrence matrix, R, topic-topic co-occurrence matrix, U, and topic-word co-occurrence matrix, S.

3 WORD AND TOPIC EMBEDDINGS

Inspired by TWE [17], our approach learns both topic and word embeddings to enhance the neural probabilistic language model. Our model receives embedding representations instead of words as input, as these representations learn the latent semantic for each word and so capture useful syntactic and semantic properties. Our proposal Wat2vec (embedding task, Word and Topic 2 vec) learns distributed word representations along with distributed topic representations. Like previous embedding models, Wat2vec requires Skip-gram [19], the state-of-the-art word embedding model, to gain continuous vector representations of words and topics from a given corpus. While Skip-Gram aims to predict context words given a target word in a sliding window, our model extends it to predict surrounding words $(w_{t-2}, w_{t-1}, w_{t+1}w_{t+2})$ and surrounding topics $(z_{t-2}, z_{t-1}, z_{t+1}z_{t+2})$ given current w_t and z_t. As TSNAL aims to predict context topics and words for a target topic and word in a sliding window, Wat2vec learns their embeddings symmetrically to exploit, in equal degree, their semantic and contextual information in the given corpus. That is, Wat2vec allows each word-topic pair w_t, and z_t to have own embedding parameters $\mathbf{a_t}$, and $\mathbf{c_t}$, and to share these parameters between the same words, and topics, respectively. As shown in Figure 2, the target word and its corresponding topic predict the identical surrounding words, and identical surrounding topics, individually.

As mentioned in Levy and Goldberg [14], the Skip-Gram framework with negative sampling (SGNS) is equivalent to implicitly factorizing a word-context matrix, the Pointwise Mutual Information (PMI) of the respective word and context pairs. After making the connection between Skip-Gram and PMI, Levy and Goldberg [14] show how to perform word embedding by spectral dimensionality reduction on the shifted positive PMI (SPPMI) matrix:

$$SPPMI(i,j) = max\{PMI(i,j) - logk, 0\}, \quad (1)$$

where k is a hyperparameter that controls the density of the SPPMI matrix. This matrix can be factorized by either using an eigenvalue method or solving the matrix factorization problem iteratively [29]. As SGNS does not require optimization procedure tuning and can capture the relationships between a word and its context within small sliding windows [19, 21], Wat2vec follows this approach, as do other models [15, 25]. Defining the word co-occurrence matrix, $\mathbf{W} \in \mathbb{V}^V (v = 1, ..., V)$ as the co-occurrence SPPMI matrix instead of the raw frequency matrix, yields word embedding matrix $\mathbf{A}$ and context word embedding matrix $\mathbf{B}$ by factorizing $\mathbf{W}$:

$$L_W = ||W - A^T B||_2^2 + \lambda_w(||A||_2^2 + ||B||_2^2), \quad A \ \ and \ \ B \geq 0, \quad (2)$$

where λ_w is the parameter that prevents overfitting.

Likewise, we construct the topic co-occurrence matrix, $\mathbf{U} \in \mathbb{O}^O (o = 1, ..., O)$ as the co-occurrence SPPMI matrix, and learn topic embedding matrix $\mathbf{C}$ and context topic embedding matrix $\mathbf{D}$ by factorizing matrix $\mathbf{U}$.

$$L_U = ||U - C^T D||_2^2 + \lambda_u(||C||_2^2 + ||D||_2^2), \quad C \ \ and \ \ D \geq 0, \quad (3)$$

where λ_u is the parameter that prevents overfitting.

Following the idea of SPPMI, Wat2vec exploits word and topic embeddings and the corresponding context topic and word, respectively, by factorizing matrix word-topic co-occurrence matrix, $\mathbf{R} \in \mathbb{V}^O$, and topic-word co-occurrence matrix, $\mathbf{S} \in \mathbb{O}^V$:

$$\begin{aligned} L_R &= ||R - A^T D||_2^2 + \lambda_r(||A||_2^2 + ||D||_2^2), \quad A \ \ and \ \ B \geq 0, \\ L_S &= ||S - C^T B||_2^2 + \lambda_s(||C||_2^2 + ||B||_2^2), \quad C \ \ and \ \ B \geq 0, \end{aligned} \quad (4)$$

where λ_r and λ_s are parameters that prevent overfitting.

Because word embedding matrix $\mathbf{A}$ appears in the factorization of both word-word co-occurrence matrix, $\mathbf{W}$, and word-topic co-occurrence matrix ,$\mathbf{R}$, and topic embedding matrix $\mathbf{C}$ appears in the factorization of both topic-topic co-occurrence matrix, $\mathbf{U}$, and topic-word co-occurrence matrix, $\mathbf{S}$, our approach exploits word and topic embeddings by factorizing matrix $\mathbf{W}$, $\mathbf{U}$, $\mathbf{R}$, and $\mathbf{S}$. The objective functions that combine these matrixes are written as::

$$\begin{aligned} L_{W,U,R,S} &= \omega_w(||W - A^T B||_2^2 + \lambda_w(||A||_2^2 + ||B||_2^2)) \\ &+ \omega_u(||U - C^T D||_2^2 + \lambda_u(||C||_2^2 + ||D||_2^2)) \\ &+ \omega_r(||R - A^T D||_2^2 + \lambda_r(||A||_2^2 + ||D||_2^2)) \\ &+ \omega_s(||S - C^T B||_2^2 + \lambda_s(||C||_2^2 + ||B||_2^2)) \end{aligned} \quad (5)$$

where ω_w, ω_u, ω_r, and ω_s are the parameters that control the weights of matrixes $\mathbf{W}$, $\mathbf{U}$, $\mathbf{R}$, and $\mathbf{S}$. This combination enables us to learn, in a balanced manner, the embeddings of words and topics that allow measurement of their similarities.

Following the Alternative Least Squares (ALS) matrix factorization method [9], we obtain the following closed form

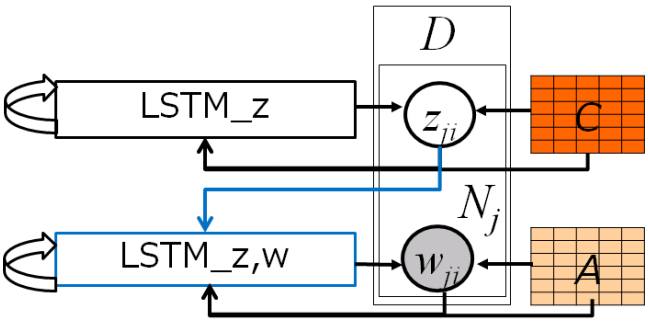

Figure 3: Architecture of TSANL: In this figure, TSANL is a neural language model. The i-th topic, z_{ji}, and word w_{ji} in the j-th document are fed to LSTMs together with their embeddings c_{ji} and a_{ji}, respectively

coordinate updates by iteratively setting the gradient to zero:

$$
\begin{aligned}
a_v &= (\omega_w \sum_{\dot{v}=1}^{V} b_{\dot{v}} b_{\dot{v}}^T + \omega_r \sum_{\dot{o}=1}^{O} d_{\dot{o}} d_{\dot{o}}^T + (\omega_w \lambda_w + \omega_r \lambda_r) I)^{-1} \\
&\times (\omega_w \sum_{\dot{v}=1}^{V} w_{v,\dot{v}} b_{\dot{v}} + \omega_r \sum_{\dot{o}=1}^{O} r_{v,\dot{o}} d_{\dot{o}}) \\
b_{\dot{v}} &= (\omega_w \sum_{v=1}^{V} a_v a_v^T + \omega_s \sum_{o=1}^{O} c_o c_o^T + (\omega_w \lambda_w + \omega_s \lambda_s) I)^{-1} \\
&\times (\omega_w \sum_{v=1}^{V} w_{v,\dot{v}} a_v + \omega_s \sum_{o=1}^{O} s_{o,\dot{v}} c_o) \\
c_o &= (\omega_u \sum_{\dot{o}=1}^{O} d_{\dot{o}} d_{\dot{o}}^T + \omega_s \sum_{\dot{v}=1}^{V} b_{\dot{v}} b_{\dot{v}}^T + (\omega_u \lambda_u + \omega_s \lambda_s) I)^{-1} \\
&\times (\omega_u \sum_{\dot{o}=1}^{O} u_{o,\dot{o}} d_{\dot{o}} + \omega_s \sum_{\dot{v}=1}^{V} s_{o,\dot{v}} b_{\dot{v}}) \\
d_{\dot{o}} &= (\omega_r \sum_{v=1}^{V} a_v a_v^T + \omega_u \sum_{o=1}^{O} c_o c_o^T + (\omega_u \lambda_u + \omega_r \lambda_r) I)^{-1} \\
&\times (\omega_u \sum_{o=1}^{O} u_{o,\dot{o}} c_o + \omega_r \sum_{v=1}^{V} r_{v,\dot{o}} a_v)
\end{aligned}
\tag{6}
$$

where $\mathbf{a_v}$, $\mathbf{b_{\dot{v}}}$, $\mathbf{c_o}$, and $\mathbf{d_{\dot{o}}}$ represent word embedding of the v-th word, context word embedding of the $\dot{v}$-th word, topic embedding of the o-th topic, and context topic embedding of the $\dot{o}$-th topic, respectively.

4 TOPIC STRUCTURE-AWARE NEURAL LANGUAGE MODEL

4.1 Motivation and Methodology

Here, we explain the motivation underlying Topic Structure-Aware Neural Language Model (TSANL); its concept is described using Figure 2. Because TSANL aims to explain the generative process of documents precisely in view of their global semantic and local semantic information and structures such as segments and phrases, it trains LSTMs by using embedding representations of topics and words. Like other topic models, TSANL represents each document as a mixture of underlying topics, $d_j = \{z_{j,1}, ..., z_{j,i}, ..., z_{j,N_j}\}$, and generates each word $w_{j,i}$ from one topic.

Unlike other topic models, TSANL preserves the sequence of ordered topics, instead of topic-collocations. As finding better ways to model long-range dependencies in language modeling is an open research issue [4], our model uses both global semantic and local semantic information to predict succeeding topics and words. That is, TSANL preserves the unit of topic as the document's structure, and segments a document into coherent topic sequences instead of topic-collocations. This allows TSANL, unlike other N-gram topic models [10, 11, 16, 27], to maintain correctly both an observed sequence of words and an observed sequence of topics. Since we assume that the sequence of topics conveys more meaning than the sum of the individual topics, TSANL forms these sequences, segments, by training LSTM, to directly predict the next topic conditioned on the sequence of topics, as does LLA [30]. LSTM keeps the internal state, and at each time step, takes an input and updates its state by applying the transition function consisting of the neural network of the previous time step's state and input. In order to incorporate the local context information, TSANL adds to LSTM the embedding of previous topic, $c_{j,i-1}$, instead of z_{ji}.

In forming a sequence of words over the same topic (phrases), TSANL trains LSTM to predict the embedding representation of the next word conditioned on the sequence of word embeddings on the same topic, unlike other neural language models, topicRNN [4], TDLM [13] and LLA, and to predict the next word that has the embedding representation vector closest to the embedding vector identified by LSTM.

4.2 Generative process

TSANL requires pre-trained topic models to gain embedding representations. With reference in the graphical model shown in Figure 2, we overview the procedure of TSANL as follows:
1. Initialize

- Initialize $LSTM_z$, and $LSTM_{z,w}$
- Initialize topic embedding representations, $\mathbf{C}$
- Initialize word embedding representations, $\mathbf{A}$

2. For each document d in Corpus D:

- For i-th token in j from 1 to N_j:
  - Get topic embedding representation, $a_{j,i}$ from the state of $LSTM_z$
  - Choose topic $z_{j,i}$ that has the topic embedding representation vector closest to the embedding vector, $c_{j,i}$, as found in the topic embedding look-up table, $\mathbf{C}$.
  - if $i \leq 1$ and $z_{j,i}! = z_{j,i-1}$, Chose $LSTM_{z,w}(c_{j,i})$
  - Get word embedding representation $a_{j,i}$ from the state of $LSTM_{z,w}(\mathbf{w})$, where $\mathbf{w}$ represents the sequence of preceding words over the same topic

Table 2: Basic statistics of the datasets in this paper

category		#documents (D)	#vocabulary (V)
20Newsgropus		11,296	10,785
Amazon	Movies and TV	20,197	45,155
	CDs and Vinyl	17,766	37,170

– Choose word $w_{j,i}$ that has the word embedding representation vector closest to embedding vector $a_{j,i}$, as found in the word embedding look-up table, $\mathbf{A}$.

In this process, topic and word embedding vectors are held in the lookup tables, $\mathbf{A} \in \mathbb{K}^{\mathbf{M}\times\mathbf{V}}$(word embedding vector), $\mathbf{C} \in \mathbb{K}^{\mathbf{M}\times\mathbf{K}}$(topic embedding vector), where M is the dimension of the vector, V is the size of vocabulary, and K is the number of topics.

4.3 Phrase detection

This task is optionally performed only once before the training step. Since conventional N-gram topic models obtain N-gram after inference terminates, they require both large memory and much time for their convergence, and yield low quality high-order $N(\geq 3)$-grams [5]. To conquer these problems, our approach mines candidates for these N-grams before the inference arep. By detecting candidates, Wat2vec restricts all constituent terms within each N-gram to those that share the same latent topic, and constructs a trie by inserting them. As this phase inherently makes N-grams of unlimited length, and to avoid phrase detection error, we use a trie [3] to highlight candidate phrases. A trie is an ordered tree structure that is used to store a dynamic set or associative array where the keys are usually strings, and all the descendants of a node have the common prefix of the string associated with that node. As this structure is composed of words instead of characters, we call this tree structure the candidate phrase trie.

4.4 Training

We outline here the execution of the whole learning process and inference step. As with language model learning, TSANL trains LSTMs to learn sequences of topics and words from a given corpus with the embeddings of observed words and topics learnt from this corpus. Each training pattern of these networks is comprised of N time steps in the embeddings of words, $(w_{t-N}...w_{t-1})$, and those of topics, $(z_{t-N}...z_{t-1})$, followed by one word, w_t, and topic, z_t, respectively. To create these sequences, we slide the window along the word and its topic assignment in the given corpus so that each word, and topic is learned from the preceding N words, and topics, respectively. That is, we train $LSTM_z$ to predict the embedding of topic, z_t, from the preceding N words' embedding representations, and $LSTM_{z,s}$ to predict the embedding of word, w_t, associated with topic z from the preceding N words' embedding representations over the same topic.

5 EXPERIMENTS

5.1 Datasets and Experiment design

We conducted evaluations using 20 Newsgroups data sets[1] and the Amazon review data sets[2], as they are well-known and publicly available [6, 17, 18, 28]. First, we selected two subsets from the Amazon review data sets, counted the number of reviews per product, and used the reviews of the top 137/115 products (object ID) in each subset, where each product had at least 100 reviews. Then, we removed the stop words and rare words (frequency lower than 5 in each collection) using NLTK[3], and converted all words to lowercase. Statistics about the data set are shown in Table 2.

To illustrate the utility of semantic relationships between words, we visualize the two-dimensional PCA projection of word embeddings, where we set the number of iterations to 5000. We calculated this projection from all words' embeddings, but show only the top 500 frequent words in Fig 4.

5.2 Document Classification

We evaluate embedding quality by using the quality of document classification results, a popular way of evaluating the effectiveness of a learned document representation yielded by a topic model [24, 25]. Generally speaking, more accurate topic modeling enables better document representation, resulting in improved document classification accuracy. As this experiment aims to evaluate the document-level representation, we use Doc2vec (PD-DBOW) as the document level representation benchmark, TWE(TWE-1) [4] with, GibbsLDA++[5] and CLM as the topic awareness embedding representation benchmark. Unless otherwise stated, documents are classified by Support Vector Machine (SVM) over each representation vector. As in [17], we represent the semantics of a document by aggregating over all topical word embeddings of each word in the document, i.e., $d = \sum_{w\in d} Pr(w|d)a_w$, where $Pr(w|d)$ can be weighted with TFIDF scores of the words in d. Since topic embeddings can be used as the semantic representation of a document by aggregating over all topic embeddings like word embeddings, i.e., $d = \sum_{w\in d} Pr(t|d)c_t$, where $Pr(t|d)$ can be weighted with TFIDF scores of the topics in d, we call these variations of TWE and Wat2vec as TWE(z) and Wat2vec(z), respectively.

Using their document-topic distribution or document embedding, each model was tasked with classifying 20Newsgroup documents, and Amazon reviews into 20 classes, where each corpus was randomly split into training and testing data sets at the ratio of 4:1. To compare the overall performance over all document classes or labels, we compared them using the macro-averaged precision, recall, and F1 coherence measure using gensim[6] over all models. The classifiers of TWE, CLM,

[1] http://qwone.com/ jason/20Newsgroups/
[2] http://jmcauley.ucsd.edu/data/amazon/qa/
[3] NLTK 3.2.4: http://www.nltk.org
[4] https://github.com/largelymfs/topical_word_embeddings/tree/master/TWE-1
[5] http://gibbslda.sourceforge.net
[6] https://radimrehurek.com/gensim/

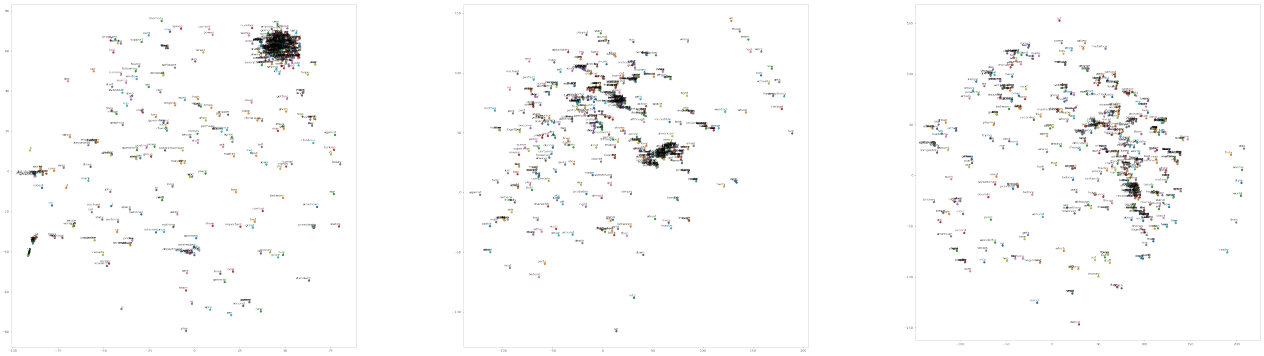

Figure 4: Visualization of the word embeddings from (left) 20News, (center) Movie, (right) CDs, using Wat2vec

Table 3: Performance of various models in document classification task: The skip length (context window size) of all embedding models is set to 5, and the negative sampling number is set to 5.

			20News			Movie			CD		
	K	D	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Doc2Vec	-	100	0.67	0.67	0.67	0.52	0.48	0.49	0.48	0.48	0.47
	-	200	0.67	0.67	0.67	0.56	0.50	0.52	0.51	0.50	0.50
	-	300	0.67	0.67	0.67	0.53	0.50	0.50	0.48	0.47	0.46
TWE	100	100	0.60	0.58	0.56	0.31	0.30	0.29	0.42	0.40	0.40
	100	300	0.63	0.62	0.60	0.32	0.31	0.31	0.49	0.48	0.47
	150	100	0.56	0.55	0.53	0.35	0.32	0.32	0.43	0.41	0.41
	150	300	0.61	0.61	0.59	0.37	0.37	0.36	0.48	0.47	0.46
CLM	100	100	0.72	0.73	0.71	0.43	0.43	0.42	0.49	0.47	0.46
	100	300	0.79	0.78	0.77	0.55	0.56	0.55	0.64	0.63	0.62
	150	100	0.74	0.74	0.72	0.43	0.42	0.41	0.47	0.47	0.46
	150	300	0.77	0.78	0.76	0.58	0.57	0.56	0.62	0.62	0.61
Wat2vec2	100	100	0.74	0.69	0.67	0.46	0.44	0.43	0.50	0.48	0.47
	100	300	0.77	0.74	0.73	0.58	0.59	0.58	0.64	0.64	0.63
	150	100	0.73	0.70	0.68	0.47	0.45	0.45	0.48	0.48	0.46
	150	300	0.77	0.76	0.74	0.60	0.60	0.59	0.63	0.63	0.62
CLM(z)	100	100	0.67	0.39	0.41	0.39	0.38	0.38	0.38	0.39	0.38
	100	300	0.72	0.43	0.45	0.40	0.39	0.39	0.42	0.44	0.42
	150	100	0.61	0.39	0.37	0.36	0.35	0.35	0.41	0.43	0.41
	150	300	0.65	0.42	0.42	0.42	0.39	0.40	0.42	0.45	0.43
Wat2vec(z)	100	100	0.68	0.70	0.69	0.44	0.46	0.44	0.43	0.41	0.39
	100	300	0.64	0.66	0.64	0.36	0.37	0.36	0.44	0.45	0.44
	150	100	0.58	0.59	0.57	0.43	0.44	0.43	0.42	0.44	0.42
	150	300	0.59	0.62	0.60	0.45	0.45	0.44	0.44	0.46	0.44

and Wat2vec were trained for each by applying the topic distribution of the training data set; the SVM classifier received the document embedding vectors as input.

Table 3 shows that our model yielded word representations effective for document classification; document-topic distributions yield better performance in this task than embedding models. While CLM and Wat2vec offer similar performance for word embeddings, Wat2vec(z) outperforms CLM for topic embeddings. This comparison demonstrates that the word embedding obtained from skip-grams with topics play an important role in allowing TSANL to learn LSTMs over topics.

6 DISCUSSION

The problem of employing deep learning for natural language processing is that it is difficult to attach a reasonable interpretation to each dimension of the generated distributions [1, 17, 22, 23]. Topic models and word embedding represent the global and local semantics, as topics encode word co-occurrence patterns in each document and the embedding representations elucidate context information. TSANL incorporates LSTM [8] as it supports both the sequential dependency of word and topic coherence.

7 CONCLUSION

This paper proposes Wat2vec as embedding models and TSANL as neural language models. As TSANL aims to capture long-range dependencies of topics and words, as segments and phrases, respectively, Wat2vec uses topics as global semantic information and local semantic information as embedding representations of topics and words, and embeds both words and topics in the same space. Experiments showed that Wat2vec learns these embedding representations for TSANL to maintain the ordering of both segments and topical phrases, and enhances previous models. Future work is to extend TSANL to deal with streaming data by developing an online algorithm.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3 (Mar 2003), 1137–1155.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (2003), 993–1022.

[3] Rene De La Briandais. 1959. File Searching Using Variable Length Keys. In *Papers Presented at the the March 3-5, 1959, Western Joint Computer Conference*. 295–298.

[4] Adji B. Dieng, Chong Wang, Jianfeng Gao, and John William Paisley. 2016. TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency. *CoRR* abs/1611.01702 (2016).

[5] Ahmed El-Kishky, Yanglei Song, Chi Wang, Clare R. Voss, and Jiawei Han. 2014. Scalable Topical Phrase Mining from Text Corpora. *Proc. VLDB Endow.* 8, 3 (Nov 2014), 305–316.

[6] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. 507–517.

[7] Aurélie Herbelot and Eva Maria Vecchi. 2015. Building a shared world: mapping distributional to model-theoretic semantic spaces. In *EMNLP*. 22–32.

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[9] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. 263–272.

[10] Shoaib Jameel and Wai Lam. 2013. An Unsupervised Topic Segmentation Model Incorporating Word Order. In *SIGIR*. 203–312.

[11] Noriaki Kawamae. 2014. Supervised *N*-gram topic model. In *WSDM*. 473–482.

[12] Noriaki Kawamae. 2016. *N*-gram over Context. In *WWW*. 1045–1055.

[13] Jey Han Lau, Timothy Baldwin, and Trevor Cohn. 2017. Topically Driven Neural Language Model. In *ACL*. 355–365.

[14] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding As Implicit Matrix Factorization. In *NIPS*. 2177–2185.

[15] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M Blei. 2016. Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-occurrence. In *RecSys*. 59–66.

[16] Robert V. Lindsey, William P. Headden, III, and Michael J Stipicevic. 2012. A phrase-discovering topic model using hierarchical Pitman-Yor processes. In *EMNLP-CoNLL*. 214–222.

[17] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical Word Embeddings. In *AAAI*. 2418–2424.

[18] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR*. 43–52.

[19] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* (2013).

[20] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*. 1045–1048.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.

[22] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving Topic Models with Latent Feature Word Representations. *TACL* 3, 4 (2015), 299–313.

[23] Liqiang Niu, Xinyu Dai, Jianbing Zhang, and Jiajun Chen. 2015. Topic2Vec: Learning distributed representations of topics. In *IALP*. 193–196.

[24] Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. 2017. Jointly Learning Word Embeddings and Latent Topics. In *SIGIR*. 375–384.

[25] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K. Reddy. 2018. Short-Text Topic Modeling via Non-negative Matrix Factorization Enriched with Local Word-Context Correlations. In *WWW*. 1105–1114.

[26] H. Wallach. 2006. Topic Modeling: Beyond Bag-of-Words. In *ICML*. 977–984.

[27] X. Wang, A. McCallum, and X. Wei. 2007. Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval.. In *ICDM*. 697–702.

[28] Guangxu Xun, Yaliang Li, Jing Gao, and Aidong Zhang. 2017. Collaboratively Improving Topic Discovery and Word Embeddings by Coordinating Global and Local Contexts. In *KDD*. 535–543.

[29] Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic Word Embeddings for Evolving Semantic Discovery. In *WSDM*. 673–681.

[30] Manzil Zaheer, Amr Ahmed, and Alexander J. Smola. 2017. Latent LSTM Allocation: Joint Clustering and Non-Linear Dynamic Modeling of Sequence Datay. In *ICML*. 3967–3976.

[31] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. 2015. SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity. In *KDD*. 1513–1522.