An embedding-based topic model for document classification

Abstract

Topic modeling is an unsupervised learning task that discovers the hidden topics in a collection of documents. In turn, the discovered topics can be used for summarizing, organizing and understating the documents in the collection. Most of the existing techniques for topic modeling are derivatives of the Latent Dirichlet Allocation which uses a bag-of-word assumption for the documents. However, bag-of-word models completely dismiss the relationships between the words. For this reason, this paper presents a two-stage algorithm for topic modelling that leverages word embeddings and word co-occurrence. In the first stage, we determine the topic-word distributions by soft-clustering a random set of embedded n-grams from the documents. In the second stage, we determine the document-topic word distributions by sampling the topics of each document from the topic-word distributions. This approach leverages the distributional properties of word embeddings instead of using the bag-of-word assumption. Experimental results on various data sets from an Australian compensation organization show the remarkable comparative effectiveness of the proposed algorithm in a task of document classification.

1 Introduction

Advances in new technologies have led to a rapid increase in the amount of data, particularly unstructured text in the form of news, blogs, web pages, articles, books, image and video captions, speech-to-text conversions, and postings on social networks. Conversely, the possibility to manually annotate such data remains inherently limited and calls for automated tools that do not rely on human supervision. Among the approaches offered by natural language processing, *topic modelling* stands out as an unsupervised approach that can help organize, summarize and understand such vast collections of textual information.

Topic models such as Latent Dirichlet Allocation (LDA) [4] have been successfully used for discovering hidden topics in text collections for years. Their most attractive feature is that they can perform their analysis from modelling assumptions without the need for manual supervision. The first model credited as a proper topic model is due to Hoffman [11], who overlaid a probabilistic model to the existing latent semantic indexing of Deerwester et al. [8], and is known as pLSA or pLSI. Blei et al. [4] later introduced the contemporary Bayesian formulation of topic modeling (i.e. LDA) by imposing conjugate Dirichlet priors on both the topics and the document-topic weights. LDA has proven hugely popular, and extensions include, among others, hierarchical versions using Dirichlet processes to address an unknown number of topics [12, 30, 33], topic evolution for topics that change over time [2, 9, 34], approaches based on correlation [3, 10], approaches based on variational autoencoders [29], and sentiment-driven topic discovery [20].

Conventional LDA uses a bag-of-word (BOW) model which models each unique word independently of the others. Anecdotally, the hidden topics learned under such simple assumptions do not correlate well with the human judgment of topic quality, especially for low-resource collections (i.e. few or short documents) [6, 18, 23]. Adding prior knowledge into topic models is often necessary to improve the models' performance [7, 26]. Recent work has also shown that interactive human feedback can improve the quality and stability of the topics [13, 37]. Additional information about the documents [25] or their words [26] can also improve LDA's topics.

To improve the quality of the extracted topics, in this paper we propose an algorithm which incorporates prior knowledge on words in the form of word embeddings The algorithm consists of two main steps. First, the topic-word distributions are determined using an incremental soft-clustering algorithm over word embeddings. Word embeddings, such as for instance Word2Vec [22], capture the correlation between words and can overcome the typical flaws of bag-of-word models, which do not account for word co-occurence and generate very high-dimensional and sparse representations. Second, the document-topic distributions are computed using matrix factorization and a sampling procedure. These two steps can be applied only once, or iteratively applied until a stability criterion for the topic-word and document-topic distributions is met. The number of topics is also allowed to increase during this process, depending on how well the existing topics fit the collection. Such a two-step learning has some analogies with the Expectation-Maximization (EM) approach to topic modeling, where the computation of the topic-word matrix can be seen as a maximization problem (i.e. an optimization of parameters), and the topic-document assignments as an expectation (a soft data partitioning).

Throughout this paper we use the following notations and symbols. The symbols V, M, N and K are used to denote the vocabulary size (or the total number of unique words in the collection, or corpus), the number of documents, the dimensionality of the word embedding space, and the number of topics (as well as clusters) in the word-vector space, respectively. The notations w_ν and y_ν are used for representing the ν-th word in the vocabulary and its word vector in the word-vector space, respectively. Given the notation for the word vectors, the word embedding matrix is denoted as Y, with $Y = [y_{\nu n}]$; $\nu = 1, \cdots, V$ and $n = 1, \cdots, N$. The notation D is used to denote the document-word matrix, i.e. $D = [d_1, \cdots, d_M] \in \mathcal{R}^{M \times V}$ where d_m is the m-th document. We use Φ and Θ for the topic-word matrix (distributions) and the document-topic matrix, respectively, i.e. $\Phi = [\phi_1, \cdots, \phi_K] \in \mathcal{R}^{K \times V}$ and $\Theta = [\theta_1, \cdots, \theta_M] \in \mathcal{R}^{M \times K}$, where ϕ_k and θ_m are, respectively, the k-th topic and the topic distribution for the m-th document.

2 Related work and motivation

Many topic modeling approaches have been proposed in the literature to amend the recognized limitations of LDA. Those most relevant to our work make use of additional, external information to improve the quality of the extracted topics. In particular, combining word embeddings [22, 24] with topic modeling is a promising solution that has been well studied in recent years; for this reason, we provide a brief overview of the main approaches hereafter.

Among the early approaches, Das et al. [7] have proposed representing the words in the vocabulary by pre-trained word embeddings, and modelling the topics as Gaussian distributions in the word embedding space rather than by the usual multinomial distributions. Other works have proposed to jointly learn the word embeddings and the topics [27, 35] or to embed the topics by a generative model [16]. Xun et al. [36] have proposed a unified language model based on matrix factorization that simultaneously takes into account global and local context information, and to model the topics and the word embeddings collaboratively. Jiang et al. [14] have integrated topic generation and embedding learning in a unified framework, and proposed a Monte Carlo EM algorithm to estimate the parameters of interest. Their assumption is that the words in a document are generated by two modalities: one based on the usual multinomial distributions, and the other based on topic embeddings as well as word embeddings. Mekala et al. [21] have proposed a representation called the sparse composite document vector (SCDV), where word embeddings are clustered to capture the semantic contexts in which words occur. After extraction, multiple SCDVs are chained together to form document-topic vectors that can express complex, multi-topic documents. In turn, both Liu et al. [17] and Xue et al. [35] have mapped word embeddings to a latent topic space that captures the multiple senses in which words may occur (polysemy). However, whole documents are mapped in the same embedding space as individual words, potentially reducing their expressive power. Word embeddings have even been used as auxiliary features to improve low-resource topic modeling of short texts [16].

Among this substantial literature, the approach we propose in this paper is most similar to the Gaussian-LDA of Das et al. [7]. However, it uses a simpler, more efficient clustering algorithm for estimating the topics instead of full posterior inference. In addition, rather than measuring the effectiveness of the inferred topic models by the typical intrinsic measures, we prefer to evaluate them as features in document classification tasks. In this way, we gain a more direct idea of their usefulness.

3 Topic-based word embedding

In this section, we briefly recapitulate the Word2Vec word embedding technique, and then we describe a topic-based word embedding. Either embedding has its own rationale: while Word2Vec captures local co-occurrences and semantics, topic-based embeddings can learn global semantics from entire collections of documents [18].

A ubiquitous trend in contemporary natural language processing (NLP) is the use of word embeddings for converting words to vectors whose relative similarities correlate with semantic similarity. Simply put, word embeddings are a type of word representation that allows words with similar meaning to have similar real-valued vector values in a predefined vector space. In traditional word embeddings, each unique word is mapped to one vector, and the vector values are learned from large, unsupervised amounts of text by using neural networks. For this reason, word embeddings have become an integral part of the deep learning machinery. The learned vectors have been used as features in a great variety of applications including, among many others, information retrieval [19], document classification [15–17], question answering [31], named entity recognition [32], and semantic parsing [28].

Word2Vec (Mikolov et al. [22]) is a popular word embedding algorithm which learns a vector representation for each word using a neural network language model. In the skip-gram variant of Word2vec, the neural network architecture uses an input, a projection, and an output layers to predict the nearby words of any given word within a text. Each word vector is learned by maximizing the log probability of the neighboring words across typically large amounts of text, i.e. given a generic sequence of words $w_1, \cdots, w_T$, the objective is to maximize:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j\in nb(t)} \log p(w_j|w_t) \tag{1}$$

where $nb(t)$ is the set of the neighboring words of word w_t, and $p(w_j|w_t)$ is obtained as the (hierarchical) softmax of the associated word vectors v_{w_j} and v_{w_t}. The reader can refer to [22] for further details. Learning these word embeddings is entirely unsupervised and can be performed over any collection of documents.

Although word embeddings and topics are two very different concepts, it is legitimate to speculate that words with similar semantics may possibly appear within the same topics. Based on this assumption, one could then create a word embedding by simply fitting a topic model with a large number of topics on a collection of documents. To do so, we could, for instance, apply LDA topic modelling to a collection of documents with topics $\Phi^0 = [\phi_1^0, \cdots, \phi_{K_0}^0] \in \mathcal{R}^{K_0 \times V}$, where $\phi_k^0 = [\phi_{k1}, \cdots, \phi_{kV}]$ is the k-th topic and $\phi_{k\nu}$ is the weight of the ν-th word in the k-th topic. Typically, the topics are sparse vectors, in the sense that most of their entries are usually very close to zero. By introducing a threshold, the topic-word matrix Φ^0 can therefore be made into a formally sparse matrix. Such a topic model can also be further modified with the use of alternative objectives.

4 The proposed method

In this section we present a novel topic learning approach that is heavily reliant on clustering. Figure 1 shows a synopsis of the proposed approach which consists of the following steps:

1. (Preprocessing) Includes removal of stop-words, very rare terms and common names.
2. (topic-word matrix) Using a sample set of n-grams from the text corpus, a word embedding and a clustering approach, computes the topic-word matrix.
3. (Document-topic matrix) Determines the document-topic matrix by scanning the text corpus and sampling from the topic-word matrix.

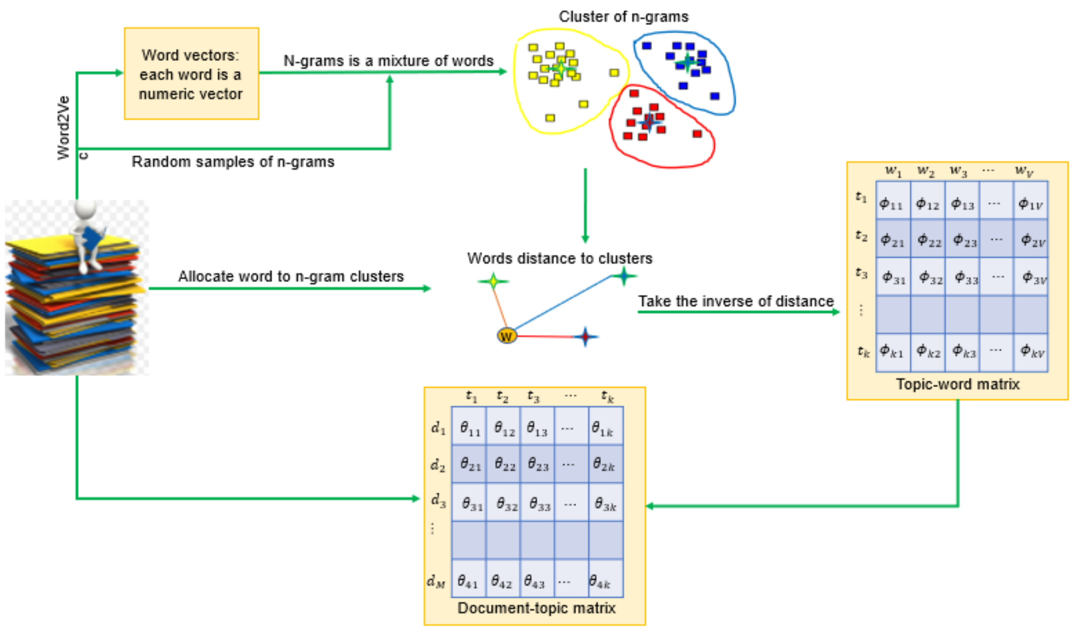
Cluster of n-grams
Word vectors: each word is a numeric vector
N-grams is a mixture of words
Word2Vec
Random samples of n-grams
Allocate word to n-gram clusters
Words distance to clusters
Take the inverse of distance
Topic-word matrix
Document-topic matrix

Figure 1: Overall process of the cluster-based topic learning.

Our assumptions for the topic-word and document-topic relationships are the same as in the LDA model, i.e. each topic is a mixture of words, and each document is a mixture of topics. However, the topic-word distributions are not modelled as multinomial distributions with Dirichlet priors; rather, they reflect the memberships of the words in word embeddings clusters. More details are provided in the following subsection.

4.1 Topic-word learning

In LDA, the topic-word distributions follow a Dirichlet distribution, and the ensuing problem is that the word co-occurrences are not taken into account. To amend this issue, we use n-grams and word embeddings to incorporate some form of word semantics into the model. To do so, we first learn the word vectors from a collection of documents using a word embedding technique. From an initial evaluation of Word2Vec and the equally popular GloVe [24], we have decided to adopt the latter. Then, a random sample set of n-grams form the text corpus is selected and is partitioned into K clusters using, for example, the k-means algorithm. As each n-gram consists of a (small) set of words, it can be represented by the average vector of the words.The inverse exponentiated Euclidean distance of the vectors to the clusters' centers is then computed, and used as the cluster-word (or topic-word) matrix. Therefore, each topic can be seen as a mixture of words, where the proportions are given by the inverse exponentiated distances of the words from that cluster/topic. The following Algorithm 1 lists the main steps for computing the topic-word matrix:

Algorithm 1 The topic-word matrix algorithm.

```
Input: External text corpus, text documents $X$ and number of topics $K$.
Output: Topic-word matrix $\Phi$.

1. Compute word vectors $W$ by using a word embedding technique on $D$, a large
   collection of documents.
2. For each document $d_i$ in $D$ :
    2.1. Select $r_i$ random samples of $n$-grams from $d_i$.
    2.2. Convert the $n$-grams to vectors by averaging the corresponding word vectors.
    2.3. Add the $n$-gram vectors to a set of $n$-grams:
         $\mathcal{G} = \mathcal{G} \cup \{g_{i,1}, \cdots, g_{i,r_i}\}$
3. Apply a clustering algorithm, e.g. $k$-means, to $\mathcal{G}$.
4. For the $k$-th topic, $k = 1 : K$, and word $w_\nu$, $\nu = 1 : V$:
    4.1. Compute the distance of word $w_\nu$ to the $k$-th center, $c_k$, as:
         $d(w_\nu, c_k) = \|w_\nu - c_k\|,$
         where $\|.\|$ is the Euclidean norm.
    4.2. The corresponding element of topic-word matrix is:
         $\phi_{\nu,k} = \exp^{-d(w_\nu, c_k)}.$
```

In the above algorithm, parameter r_i is a random integer that is scaled proportionally to the length of the document. In Step 4.1., the Euclidean norm is used to compute the distance between word w_ν and cluster k, which is then exponentiated and inverted in Step 4.2. to represent the word's membership to the cluster.

4.2 The document-topic matrix

To determine the document-topic matrix (or document-topic distributions), one can use matrix factorization by considering the fact that $X = \Theta\Phi$, where X is the observed document-term matrix, Φ is the topic-word matrix described in Section 4.1, and Θ is the desired document-topic matrix. In a real situation, the equality condition is mitigated to $X \approx \Theta\Phi$ and Θ is computed as follows:

$$\min_{\Theta} \|X - \Theta\Phi\|_2^2 \quad \text{s.t.} \quad \Theta \geq 0. \tag{2}$$

Adding a regularizer, Eq. (2) is equivalent to minimizing the following:

$$f(\Theta) = \|X - \Theta\Phi\|_2^2 + \lambda\|\Theta\|_2^2. \tag{3}$$

where λ is a scalar parameter. Although the minimization of (2) is equivalent to non-negative matrix factorization (NMF), [1], in our approach we only solve for the document-term matrix, Θ, for a given topic-word matrix, Φ.

The optimization problem in (3) can be solved by a standard pseudoinverse approach. An alternative is to use a heuristic approach where sampling from the topic-word matrix based on the observed words is used for determining the document-topic matrix. The quality of the solution can be examined using the objective function (3), and the process can be incrementally repeated by iteratively increasing the number of topics and calculating the document-topic matrix each time, until no progress is achieved. Using sampling allows us to easily control the size of the sample set and the computational complexity; however, in the future, we plan to also explore incremental optimization algorithms [5].

The heuristic approach consists of simply scanning all the documents and assigning topics to words by sampling from the topic-word matrix conditional to the given word. The main steps of the algorithm for determining the document-topic matrix are as follows:

Algorithm 2 Document-topic matrix.

Input: Text documents D and topic-word matrix Φ.
Output: Document-topic matrix $\Theta = [\theta_{i,j}],\ i = 1 : M,\ j = 1 : K$.

```
1. For each document $d_i$ in $D$:
    1.1. Select a random number, $n_i \in [l_i, u_i]$, where $l_i$ and $u_i$ are lower and upper
         bounds to number of samples for document $d_i$.
    1.2. Sample $n_i$ words from $d_i$, by replacement.
             $W_i = \{w \in d_i | w \sim U[W_d]\}.$
    1.3. For each word $w$ in $W_i$:
        1.3.1. Sample $\tau$ topics from $\Phi$, with replacement.
                   $T_\tau = \{t_1, \cdots, t_\tau\} \sim \Phi_w.$
        1.3.2. Assign topic $t_j$ for $w$ as:
                   $t_j = \underset{l}{\operatorname{argmax}} |T_\tau|.$
        1.3.3. $\theta_{i,j} = \theta_{i,j} + 1$
    1.4. Normalize rows of $\Theta$ to add up to one, $\|\Theta_i\| = 1$, $i = 1, \cdots, M$.
```

In Algorithm 2, Θ_i corresponds to the weight vector of document d_i, $\theta_{i,j}$ to the weight of topic j in document i, and Φ_w to the topic vector associated with word w, i.e. $\Phi = [\Phi_1, \cdots, \Phi_V]^T$.

5 Experiments

5.1 Data sets

The data sets used in our experiments have been obtained from an accident compensation agency of the Victorian Government in Australia. They consist of phone calls between clients and claim managers of the organization that address challenges and issues of the clients. Each observation might contain several phone calls related to the same client. The data have been annotated by experts with a class label of "yes" or "no" depending on whether they contained a specific "challenge" or not. Where labeling was ambiguous, the data have been left undecided. A brief description of the data sets is available in Table 1, where M_1 and M_2 are the number of samples from classes "yes" or "no", respectively, and N is the total number of samples in the data set. To train the classifiers, we have used 10 different random and independent subsets of each data set.

Table 1: Data sets summary.

Datasets	M_1	M_2	N
D1	4,704	3,779	11,591
D2	3,896	2,267	11,553
D3	3,186	1,058	11,220

The data have required many pre-processing steps including: transforming to lowercase letters, removing stop words and common words, and replacing synonyms. For example, "return to work" and "back to work" are treated as the same words and need to be converted to a single word, say for instance "rtw". The vocabulary (i.e. set of unique words) of these data sets is very different from a general-purpose vocabulary as it contains many specialized terms and acronyms. Since the size of the data sets is limited, our topic modelling and document classification tasks can then be regarded as low-resource cases, and expect to benefit from the proposed approach.

5.1.1 Implementation and models

For the classification task, we use the rows of our document-topic matrix, Φ, as the features of each document. We then compare the proposed model, that we named "NgTC" from "n-gram topic clustering", with three baselines:

- the tf-idf features,
- the weighted averages of word vectors from Word2Vec;
- using the rows of the document-topic matrix obtained from an LDA model as features.

To evaluate and compare the effectiveness of the proposed model, we have applied different classifiers with the various features. The classifiers we have used are:

- Extreme Learning Machine (elm),

- Support Vector Machines with Radial Basis Function Kernel (svm),
- Stochastic Gradient Boosting (gbm),
- Boosted C5.0 (c50),
- Random Forest (rf)

The implementation of all models and classifiers was performed in R, a popular language and environment for data science, statistical computing and graphics (see: https://www.r-project.org/). All experiments have been carried out on a 1.90 GHz Intel(R) Core(TM) i7 machine with 32 GB of RAM.

5.1.2 Document classification

To evaluate the proposed model and compare it with the baselines introduced in the previous subsection, we have carried out experiments on the three data sets. As performance measure, we report the average of the accuracies over 10 random and independent samples from the data. In addition, we report the accuracies by class, and the highest and lowest values on the 10 runs; see Tables 2–4.

Table 2 shows the accuracy results for data set D1 for all the models and the different classifiers. On this data set, the proposed model has outperformed the other models with all classifiers except for the case of "svm", for which the results are almost the same for all the models. The best overall accuracy (92.53%) has been obtained by "NgTC" with "c50", with an impressive improvement of 25.71 percentage points over the second-best model ("dtm" with "rf"). The results for data set D2 (Table 3) are very similar to those for D1, and "NgTC" has also proved the most accurate on data set D3 (Table 4), although by smaller margins. In general, these results demonstrate the effectiveness of the proposed model, "NgTC", in classifying the text documents.

Figures 2 – 4 plot the overall accuracies for data sets D1 – D3 with the different classifiers. These results, in combination with those in Tables 2–4, show that the proposed model has achieved the best performance, followed by the tf-idf features ("dtm" model). It is important to note that the number of the features in all the models has been set to the same value (200), with the exception of the "dtm" model where the number of the features has to reflect the size of the vocabulary, and it has therefore been set to over 11,000.

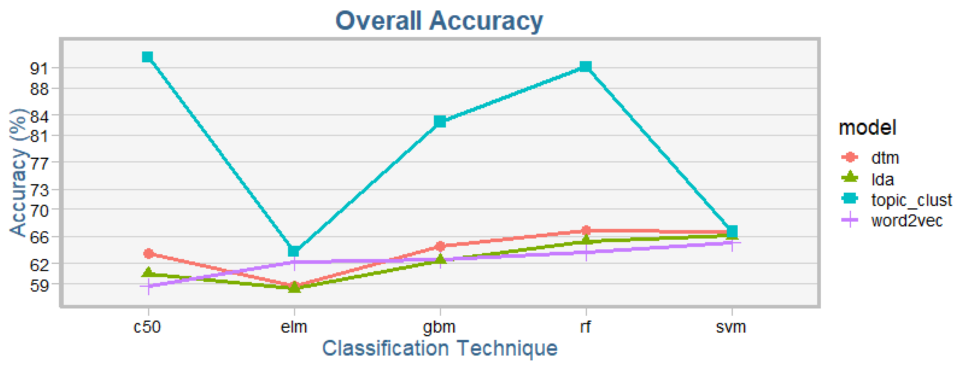

Figure 2: Accuracy (%) of models for D1 using the different classifiers.

Table 2: Accuracy results for D1: average of 10 independent random runs for the 4 models with 5 classification algorithms; results include accuracies, lower and upper values for the accuracies, and by-class accuracies

model	method	Accuracy	AccuracyLower	AccuracyUpper	Pos Pred Value	Neg Pred Value
	elm	58.61	56.24	60.98	60.70	54.33
	svm	66.78	64.46	68.99	67.84	64.64
dtm	gbm	64.61	62.28	66.89	64.14	63.95
	c50	63.41	61.06	65.69	61.53	64.89
	rf	66.82	64.52	69.07	68.60	64.09
	elm	62.24	59.88	64.55	64.14	58.91
	svm	65.08	62.76	67.34	65.78	63.27
word2vec	gbm	62.52	60.17	64.82	62.75	62.08
	c50	58.62	56.24	60.98	58.50	57.05
	rf	63.64	61.33	65.94	63.75	63.19
	elm	58.27	55.87	60.62	60.50	53.74
	svm	66.16	63.85	68.41	65.68	65.63
lda	gbm	62.45	60.07	64.75	62.01	61.84
	c50	60.48	58.10	62.80	62.20	57.53
	rf	65.29	62.97	67.57	65.13	64.76
	elm	63.75	61.41	66.04	65.41	60.88
	svm	66.68	64.37	68.92	68.21	64.11
NgTC	gbm	82.94	81.04	84.70	81.26	86.70
	c50	92.53	91.17	93.75	94.10	90.67
	rf	91.10	89.64	92.42	89.83	92.97

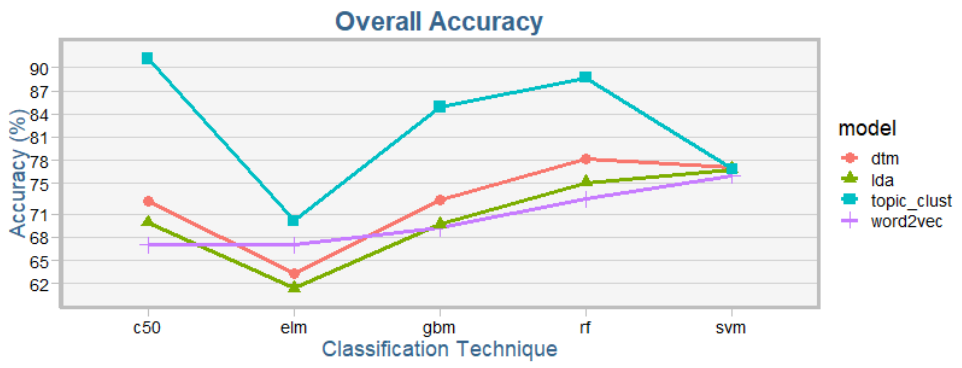

Figure 3: Accuracy (%) of models for D2 using the different classifiers.

Table 3: Accuracy results for D2: average of 10 independent random runs for the 4 models with 5 classification algorithms; results include accuracies, lower and upper values for the accuracies, and by-class accuracies

model	method	Accuracy	AccuracyLower	AccuracyUpper	Pos Pred Value	Neg Pred Value
	elm	63.25	60.47	65.94	51.75	65.58
	svm	77.03	74.59	79.34	78.71	73.29
dtm	gbm	72.84	70.26	75.31	75.40	67.08
	c50	72.74	70.16	75.20	76.18	65.60
	rf	78.24	75.85	80.50	78.15	78.58
	elm	67.01	64.31	69.62	57.53	70.89
	svm	76.02	73.52	78.39	77.65	72.18
word2vec	gbm	69.22	66.55	71.79	69.41	68.51
	c50	67.00	64.30	69.61	67.09	61.49
	rf	72.98	70.41	75.45	72.13	76.58
	elm	61.35	58.57	64.1	45.43	64.02
	svm	76.81	74.36	79.13	76.38	75.43
lda	gbm	69.70	67.06	72.27	69.49	71.30
	c50	69.9	67.25	72.45	68.08	67.37
	rf	75.10	72.58	77.48	76.82	75.39
	elm	70.16	67.54	72.72	64.72	72.04
	svm	76.86	74.39	79.16	78.52	73.11
NgTC	gbm	84.89	82.76	86.82	82.17	92.58
	c50	91.16	89.44	92.68	92.09	89.53
	rf	88.69	86.79	90.42	86.26	94.62

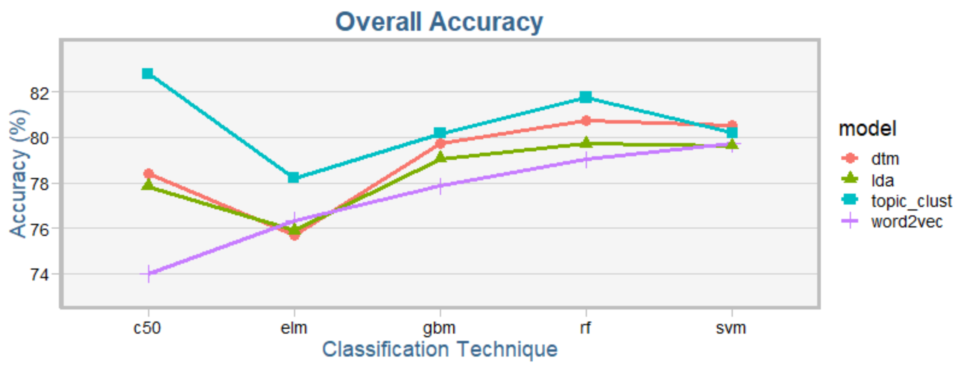

Figure 4: Accuracy (%) of models for D3 using the different classifiers.

Table 4: Accuracy results for D3: average of 10 independent random runs for the 4 models with 5 classification algorithms; results include accuracies, lower and upper values for the accuracies, and by-class accuracies

model	method	Accuracy	AccuracyLower	AccuracyUpper	Pos Pred Value	Neg Pred Value
	elm	75.68	72.63	78.51	52.93	76.98
	svm	80.52	77.70	83.15	81.02	76.24
dtm	gbm	79.72	76.85	82.38	80.2	68.94
	c50	78.39	75.44	81.11	79.85	62.24
	rf	80.72	77.90	83.33	81.40	75.35
	elm	76.35	73.34	79.18	54.37	79.10
	svm	79.72	76.87	82.39	79.80	78.90
word2vec	gbm	77.85	74.91	80.62	78.09	73.76
	c50	73.99	70.88	76.90	75.17	50.19
	rf	79.03	76.16	81.73	79.08	78.42
	elm	75.88	72.89	78.75	55.83	76.91
	svm	79.64	76.79	82.29	80.07	75.20
lda	gbm	79.06	76.17	81.77	80.32	64.29
	c50	77.84	74.90	80.60	81.82	58.31
	rf	79.72	76.85	82.37	80.11	75.57
	elm	78.21	75.28	80.93	64.69	79.66
	svm	80.19	77.34	82.81	80.30	79.08
ngtc	gbm	80.18	77.33	82.82	81.31	71.38
	c50	82.79	80.07	85.27	87.48	66.56
	rf	81.75	78.96	84.27	81.45	84.82

6 Conclusion

In this paper, we have proposed a novel topic model based on n-gram embeddings and a two-stage learning approach. In the first stage, the topic-word distributions are computed using clusters of n-gram word embeddings. In the second stage, the document-topic matrix is computed using sampling from the topic-word conditional distributions. In a set of experiments over textual data sets from an Australian compensation agency, we have compared the proposed model against relevant baselines using different classifiers. The results show that the proposed model has outperformed all the baselines on all data sets, with improvements over the runner-up of up to 25.71 accuracy percentage points. As future work, we note that in this paper we have only used a single iteration of the model, i.e. the topic-word and document-topic matrix have been computed only once. However, they could be alternatively learned to attain even more refined parameters. For the second stage, we plan to explore incremental optimization algorithms as an alternative to the current sampling procedure.

References

[1] Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models - Going beyond SVD. In *Proceedings - Annual IEEE Symposium on Foundations of Computer Science,*

FOCS, 2012.

[2] D.M. Blei and J.D. Lafferty. Dynamic topic models. In *International Conference on International Conference on Machine Learning (ICML)*, 2006.

[3] D.M. Blei and J.D. Lafferty. A correlated topic model of science. *Statistics*, 1(1):17–35, 2007.

[4] D.M. Blei, A.Y. Ng, and Jordan MI. Latent dirichlet allocation. *Machine Learning Research*, 3:1107–1135, 2003.

[5] Andrea Cassioli, A. Chiavaioli, Costanzo Manes, and Marco Sciandrone. An incremental least squares algorithm for large scale linear classification. *Eur. J. Oper. Res.*, 224(3):560–565, 2013.

[6] Jonathan Chang. Not-So-Latent Dirichlet Allocation : Collapsed Gibbs Sampling Using Human Judgments. *Computational Linguistics*, 2010.

[7] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian LDA for Topic Models with Word Embeddings. *Proceedings ACL 2015*, pages 795–804, 2015.

[8] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *J Am Soc Inf Sci*, 41(6):391, 1990.

[9] Mohamed Dermouche, Julien Velcin, Leila Khouas, and Sabine Loudcher. A Joint Model for Topic-Sentiment Evolution over Time. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2014.

[10] Junxian He, Zhiting Hu, Taylor Berg-Kirkpatrick, Ying Huang, and Eric P. Xing. Efficient correlated topic modeling with topic embedding. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[11] T. Hofmann. Probabilistic latent semantic indexing. In *Proc. of the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval*, 1999.

[12] Wei Shou Hsu and Pascal Poupart. Online Bayesian moment matching for topic modeling with unknown number of topics. In *Advances in Neural Information Processing Systems*, 2016.

[13] Yuening Hu and Jordan Boyd-Graber. Efficient tree-based topic modeling. In *50th Annual Meeting of the Association for Computational Linguistics, ACL 2012 - Proceedings of the Conference*, 2012.

[14] Di Jiang, Lei Shi, Rongzhong Lian, and Hua Wu. Latent Topic Embedding. In *Coling*, pages 2689–2698, 2016.

[15] M. J. Kunser, Y. Sun, N. I. Kolkin, and K. Q. Weinberger. From word embeddings to document distances. In *Proc. of the 32nd International Conference on International Conference on Machine Learning (ICML'15)*, volume 37, pages 957–966, 2015.

[16] Shaohua Li, Tat Seng Chua, Jun Zhu, and Chunyan Miao. Generative topic embedding: A continuous representation of documents. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 2016.

[17] Yang Liu, Zhiyuan Liu, Tat Seng Chua, and Maosong Sun. Topical word embeddings. In *Proceedings of the National Conference on Artificial Intelligence*, 2015.

[18] Jeffrey Lund, Piper Armstrong, Wilson Fearn, Stephen Cowley, Courtni Byun, Jordan Boyd-Graber, and Kevin Seppi. Automatic and human evaluation of local topic quality. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2020.

[19] C. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[20] Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and Chengxiang Zhai. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *16th International World Wide Web Conference, WWW2007*, 2007.

[21] Dheeraj Mekala, Vivek Gupta, Bhargavi Paranjape, and Harish Karnick. SCDV : Sparse Composite Document Vectors using soft clustering over distributional representations. In *EMNLP*, 2017.

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.

[23] D. Newman, E. V. Bonilla, and W. Buntine. Improving topic coherence with regularized topic models. In *Proc. of the 24th International Conference on Neural Inform. Processing Syst.*, pages 496–504, 2011.

[24] Socher R. Manning C. D. Pennington, J. Glove: Global vectors for word representation. In *Proc. of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, volume 12, pages 1532–1543, 2014.

[25] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP 2009 - Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: A Meeting of SIGDAT, a Special Interest Group of ACL, Held in Conjunction with ACL-IJCNLP 2009*, 2009.

[26] S. Seifollahi, M. Piccardi, and E. Zare-Borzeshi. A semi-supervised hidden markov topic model based on prior knowledge. In *the 15th Australasian Data Mining Conference (AusDm 2017)*, 2017.

[27] Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. Jointly learningword embeddings and latent topics. In *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017.

[28] R. Socher, J. Bauer, C. D. Manning, and A. Y. NG. Parsing with compositional vector grammars. In *Annual Meeting of the Association of Computational Linguistics*, 2013.

[29] Akash Srivastava and Charles A. Sutton. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[30] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 2006.

[31] S. Tellex, B. Katz, J. Lin, A. Fernandes, and G. Marton. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proc. of the 26th Annual International ACM SIGIR Conference on Search and Development in Information Retrieval*, pages 41–47, 2003.

[32] J. Turian, L. Ratinov, Y. Bengio, and D. Roth. A preliminary evaluation of word representations for named-entity recognition. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*, 2009.

[33] Chong Wang, John Paisley, and David M. Blei. Online variational inference for the hierarchical Dirichlet process. In *Journal of Machine Learning Research*, 2011.

[34] Xuerui Wang and Andrew McCallum. Topics over Time: A non-markov continuous-time model of topical trends. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[35] Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. Distilled Wasserstein learning for word embedding and topic modeling. In *Advances in Neural Information Processing Systems*, 2018.

[36] Guangxu Xun, Yaliang Li, Jing Gao, and Aidong Zhang. Collaboratively improving topic discovery and word embeddings by coordinating global and local contexts. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[37] Yi Yang, Doug Downey, and Jordan Boyd-Graber. Efficient methods for incorporating knowledge into topic models. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 2015.