



Word Embedding-based Topic Modeling

Slimane Bellaouar^{1,2,*} , Ahmed Itbirene^{1,2}, and Brahim Chihani^{1,2}

¹ Department of Mathematics and Computer Science, Université de Ghardaia, Algeria

² Laboratoire des Mathématiques et Sciences Appliquées (LMSA), Université de Ghardaia, Algeria

{bellaouar.slimane, itbirene.ahmed, chihani.brahim}@univ-ghardaia.dz

* Corresponding author: Slimane Bellaouar bellaouar.slimane@univ-ghardaia.dz

Abstract. The extraction of topics from information that is in the form of unmarked texts has become a challenging task due to the significant advancements in the field of digitization. Therefore, we need a topic modeling technique, which is based on unsupervised algorithms. Our paper delineates the topic modeling concept and the inherent approaches including Latent Dirichlet Allocation (LDA), Embedded Topic Model (ETM), Gaussian LDA (G-LDA), and LDA with Word2Vec (LDA2Vec). In the experimental work, we make an empirical comparison between both LDA and ETM methods on the 20 newsgroups dataset, in terms of topic coherence and runtime. The results are absolutely in favor of the ETM approach.

Keywords: Topic modeling · Word embeddings · Latent Dirichlet Allocation (LDA) · Word2Vec · Topic coherence.

1 Introduction

Since the explosion of information with the advent of the third generation of web sites, classic data analysis approaches have become helpless in front of the massive flow of information. Then it is necessary to find more effective and smarter ways to confront this phenomenon, especially when the information is unlabeled text type. In fact it is not easy to classify this information in specific topics. Thus it is advantageous to make reference to unsupervised algorithms, namely topic modeling.

Topic modeling is a field of text mining that uses unsupervised machine learning techniques for extracting the meanings of words according to their context. The system is capable of processing a large corpus of documents by organizing words into a bag of words and identifying topics using a similarity analysis.

In our work, we present different approaches of topic modeling starting with the classical method named Latent Dirichlet Allocation (LDA) ([9]), and passing through several new methods that are devised from the combination of LDA model and Word Embedding technique. Those methods are Embedded

© The Author(s) 2024

C. A. Kerrache et al. (eds.), *Proceedings of the International Conference on Emerging Intelligent Systems for Sustainable Development (ICEIS 2024)*, Advances in Intelligent Systems Research 184,

https://doi.org/10.2991/978-94-6463-496-9_8

Topic Model (ETM) [12], Gaussian LDA (G-LDA) [11], and finally LDA with Word2Vec (LDA2Vec) [18].

As a contribution, we make an experimental comparison between both LDA and ETM methods, using the corpus of 20 Newsgroups. The results we obtained in terms of topic coherence and runtime, are in favor of the ETM approach.

The subsequent sections of the paper are organized in the following manner: Section 2 focuses on text representation, both traditional and *Word embeddings*. Section 3 is dedicated to related works. The experimental process is delineated in Section 4. Finally, Section 5 concludes our research work.

2 Text Representation

The usual method for text representation involves the mapping of words or documents from a vocabulary into vectors of real numbers, which are known as *word vectors*.

2.1 Traditional Text Representation

It is advisable to conduct a concise overview of some of the traditional methods that predate word embeddings before focusing on *Word2Vec*.

Bag-Of-Words (BOW) Model [15] is a text representation that delineates the word occurrence frequencies in a document. The term "bag of words" is coined since it involves disregarding any information pertaining to the arrangement or organization of words in the document. The approach exclusively concentrates on the existence of acknowledged terms in the document. This process is frequently referred to as *vectorization*.

The bag-of-words model has been implemented with considerable success in the context of prediction problems, including document classification and language modeling. Still, it is plagued by certain deficiencies [15], including meaninglessness and sparsity.

Term Frequency - Inverse Document Frequency (TF-IDF) is a technique used to measure the importance of a word inside a particular document in a collection of documents, often known as a corpus [21]. It assigns a weight to a specific term based on how often it appears in documents. The value of TF-IDF is directly correlated with the word's frequency in a document, but it is adjusted by the number of documents in the corpus that include the term. This adjustment takes into consideration the fact that certain words have a higher frequency of occurrence compared to others.

TF-IDF is given by Equation (1).

$$W_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right). \quad (1)$$

In this context, $tf_{i,j}$ denotes the frequency of term i in document j , df_i represents the number of documents that contain term i , and N indicates the overall number of documents.

Nevertheless, despite the fact that *BOW* and *TF-IDF* representations assign weights to various words, they fail to capture the semantic meaning of the words.

2.2 Word Embeddings

Word embedding tries to teach computers or machines the meaning of words. In recent times, the area of NLP has made advancements in linguistic models by adopting neural network design as opposed to the conventional n-gram models. These novel methodologies encompass language modeling and feature learning algorithms that convert words into real number vectors, thereby earning the name *word embeddings*.

Word embedding models have rapidly gained popularity due to their ability to convert strings into real numbers, enabling us to do calculations. For instance, we may identify words that share a similar meaning (synonyms) by measuring the distance between vectors.

Word2Vec Word2Vec is the standard method for training word embeddings, which gained its popularity since 2013 [17]. It is the first method that demonstrated vector arithmetics to solve word analogies [4, 14].

Word2Vec converts words into vectors, enabling various operations such as addition, subtraction, and distance calculation between vectors. These operations facilitate the execution of linear word analogies, thereby preserving the relationships among the words. Thus, an illustration of this correlation may be seen in a widely renowned outcome of Word2Vec, which states: $\vec{king} - \vec{man} + \vec{woman} = \vec{queen}$.

Word2Vec has demonstrated its efficacy across a range of subsequent NLP tasks. The system offers two primary architectures, Continuous Bag-Of-Words (CBOW) and Skip-Gram (SG), which are utilized to generate a distributed representation of words. The Skip-Gram model, for example, utilizes a word to predict the target context. This refers to the ability to predict words that occur within a defined range both before and after the current word (see Fig. 1).

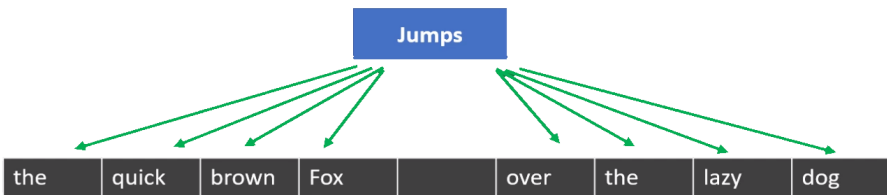


Fig. 1. Skip-gram example.

The skip-gram neural network model architecture is depicted in Fig. 2.

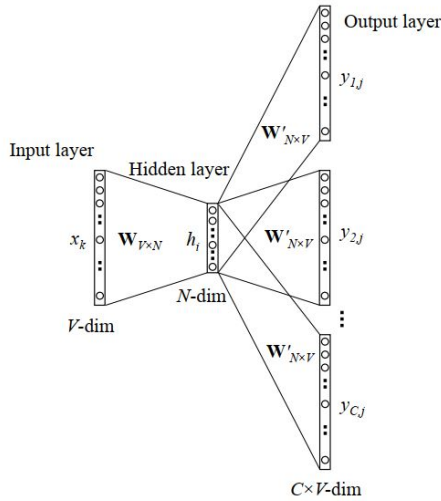


Fig. 2. Skip-gram architecture [23].

3 Related Works

This section focuses on different topic modeling approaches. We start with Latent Dirichlet Allocation (LDA) and pass through three other approaches, labeled Embedded Topic Model (ETM), Gaussian LDA (G-LDA), and finally LDA with Word2Vec (LDA2Vec). These methods are the combination of word embeddings and LDA.

3.1 Latent Dirichlet Allocation (LDA)

LDA is a Bayesian generative probabilistic model that is used to identify the abstract topics that occur in a corpus. It was originally proposed by [9]. The fundamental concept underlying LDA is that every document can be characterized by a distribution of topics, and each topic can be characterized by a distribution of words (Fig. 3).

Formally, in the context of text analysis, a collection of text documents can be represented as a bag-of-words. In this representation, n_{wd} denotes the count of the word w in the document d . The model trains to determine the probability distribution of words for each topic ($\phi_{wt} = p(w | t)$) and the probability distribution of topics for each document ($\theta_{td} = p(t | d)$).

Before diving into the details, we present some notations.

- K : The overall topic count.
- D : The overall document count in the corpus.
- N : The overall word count in a given document.

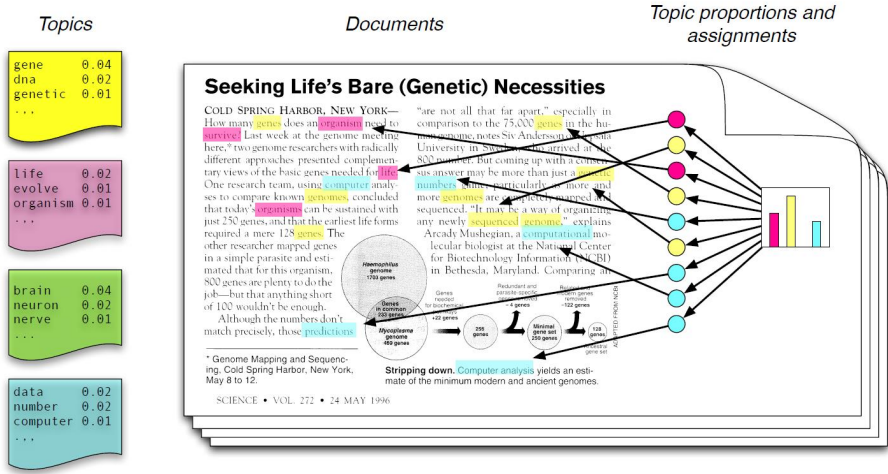


Fig. 3. (Right) Each document represents a probability distribution across topics. (Left) Each topic is represented by a probability distribution across words [7].

- α : hyper parameter for θ ; encouraging sparseness of document-topics distribution
- β : hyper parameter for ϕ ; encouraging sparseness of topic-words distribution .
- ϕ_k : per-topic distribution over words, $K \times V$ matrix (V is the size of the vocabulary).
- θ_d : per-document distribution over topics, $D \times K$ matrix (one row for each document).
- $z_{d,n}$: per-word topic assignment, $N \times K$ matrix (one row for each word from the document).
- $w_{d,n}$: observed word drawn from topic's word distribution (the n^{th} word in the d^{th} document).

To recap the aforementioned, α and β are Dirichlet-prior parameters, ϕ and θ are Dirichlet distributions, z and w are multinomials. The mathematical formula of LDA can be presented by Equation 2.

$$p(\phi, \theta, z, w \mid \alpha, \beta) = \prod_{k=1}^K p(\phi_k \mid \beta) \prod_{d=1}^D p(\theta_d \mid \alpha) \prod_{n=1}^N p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \phi_{1:k}, z_{d,n}). \quad (2)$$

The generative process of LDA model (Algorithm 1):

Algorithm 1: Generative process of LDA model [8]

```

1 for each topic  $k \in \{1, \dots, K\}$  do
2    $\lfloor$  Choose a distribution over words:  $\phi_k \sim Dir(\beta)$ 
3 for each document  $d \in \{1, \dots, D\}$  do
4    $\lfloor$  Choose a distribution over topics:  $\theta_d \sim Dir(\alpha)$ 
5     for each word  $n \in \{1, \dots, N\}$  inside document  $d$  do
6        $\lfloor$  Choose a topic from document's distribution:  $z_{d,n} \sim Mult(\theta_d)$ 
7        $\lfloor$  Choose a word from topic's distribution:  $w_{d,n} \sim Mult(\phi_{z_{d,n}})$ 

```

It is worth noting that LDA performs better than LSA (Latent Semantic Analysis), a conventional method for topic modelling, in terms of both runtime and topic coherence [6].

3.2 Topic Modeling with LDA and Word Embeddings

LDA is highly influential and it is used very widely in topic modeling. However, it suffers from some limitations, but we focus on a particular problem and that is when we treat every word type (term) as a distinct entity. To clarify a little bit: each topic is a categorical distribution over the terms of our vocabulary. Therefore, the model learns a separate probability for each term, then we have a separate parameter for the probability of the word “cat” and the word “kitten” given a particular topic ($p(\text{cat} | t) \neq p(\text{kitten} | t)$), i.e., there is no sharing between these distributions, although they are synonyms.

An alternative for this problem is word embeddings that becomes crucial in many applications of NLP [2, 5, 13, 16, 25]. The idea is instead of modeling words we model continuous vector-space embeddings. These are feature representations for text where we can think of words as vectors in a vector space, and the words that share the same meaning have a comparable representation.

The idea of training these embeddings is that, the features are based on the context in which a word gets used in the training corpus, and it exploits what is called the distributional hypothesis, which is essentially the words that frequently occur in similar settings are likely to possess a comparable semantic significance.

Mixing LDA and word embeddings to gain powerful topic models What we do is instead of modeling the probability of words as we do in basic LDA, we model the probability of the features associated with words that is the dimension of these embeddings. We can use pre-trained model of word embeddings that is trained on a very large corpus, and that is easily available for English and some other languages.

The basic idea is that we train our embedding model on a large, general-domain corpus, and then we train our topic model on typically much smaller target corpus.

There are several topic models proposed by the researchers that make use of this idea. In this paper, we are going to talk through a few of them.

Embedded Topic Model (ETM) ETM is a topic model that marries the probabilistic topic modeling of Latent Dirichlet Allocation (LDA) with the contextual information brought by word embeddings, most specifically, Word2Vec using CBOW model [17]. It was originally published by [12]. ETM benefits from the good properties of both

topic models and word embeddings. As a topic model, it discovers the hidden topics inside documents; as a word embedding model it provides the meaning of words.

We can describe the formal task of ETM as: given a corpus of D text documents containing V distinct words, and K latent topics. The model trains to:

- Find the probability distribution of topics for each document θ_d .
- Embed each word in an L -dimensional space $E_v \in R^L$.
- Embed each topic in an L -dimensional space $\alpha_k \in R^L$.

In what follow, we present some notations inherent to the ETM model:

- K : overall topic count.
- D : overall document count in the corpus.
- N : overall word count in a given document.
- E : embedding matrix of size $L \times V$. E_v is a vector of size L that contains the word embedding v (V is the vocabulary size and L is the dimensional space).
- α_k : topic embedding vector of size L .
- θ_d : per-document distribution over topics, $D \times K$ matrix.
- $z_{d,n}$: per-word topic assignment, $N \times K$ matrix.
- $w_{d,n}$: observed word drawn from a distribution defined by the softmax of the dot product between E (word embedding matrix) and α_k (topic embedding vector).

The Algorithm 2 presents the generative process of the ETM model:

Algorithm 2: Generative process of the ETM model [12]

```

1 for each document  $d \in \{1, \dots, D\}$  do
2   Choose a distribution over topics:  $\theta_d \sim LN(0, I)$ 
3   for each word  $n \in \{1, \dots, N\}$  inside document  $d$  do
4     Choose a topic from document's distribution:  $z_{d,n} \sim Cat(\theta_d)$ 
5     Choose a word:  $w_{d,n} \sim softmax(E^T \alpha_{z_{d,n}})$ 

```

Gaussian LDA (G-LDA) Topics in LDA are semantically incoherent and this is one of the Limitations of LDA model, according to [10, 19]. In order to solve this problem, the G-LDA method that was originally proposed by [11] can find the semantic coherence between topics using word embedding technique that has shown its efficacy in capturing the lexico-semantic regularities in a language [1, 17].

G-LDA replaces categorical distributions over word in LDA with multivariate gaussian distributions over the word embedding space.

G-LDA considers topics K as multivariate gaussian distributions with mean μ_k and covariance Σ_k on the continuous embedding space distributions over words. It is based on the following values: a gaussian distribution μ_k centered at zero for the mean and an inverse wishart distribution for the covariance matrix Σ_k , where the inverse wishart is used as the conjugate prior for the covariance matrix of a multivariate normal distribution and it ensures positive definiteness of this matrix. The inverse wishart function was discovered for the first time by John Wishart in 1928, it defined with a matrix of $r \times r$ dimension and a number of degrees of freedom df , with $df > (r-1)$ according to [24].

LDA with Word2Vec (LDA2Vec) is a marriage of LDA and Word2Vec model. It was originally published by [18]. It is understood that the text contains a mixture of various topics, and each topic consists of a mixture of words. Word2Vec is employed in the lexical domain to acquire vectorized word representations.

LDA2Vec is based specifically on the Skip-Gram model of Word2Vec to generate word vectors, who trains the word embeddings using the input word to predict the words context. LDA2Vec utilizes a modified version of Skip-Gram named Skip-Gram Negative-Sampling (SGNS) to achieve two objectives: generating feature vectors that represent entire documents and simultaneously learning continuous document weights that are applied to topic vectors.

4 Experiments

In this section, we carry out an experimental study that represents a comparison between two methods of topic modeling; they are known as LDA and ETM. The evaluation is carried out according to runtime and different topic coherence measures (Cv, UCI, NPMI, UMass).

The reason for selecting LDA is its widespread usage as a topic model and its similarity in generating process to the ETM. We also considered ETM because it is a new document model technique that benefits from LDA properties and Word Embeddings, which makes it a powerful and an interesting model to work with in topic modeling. We apply this study on the 20Newsgroups dataset.

4.1 Environment

For implementation purpose, we choose Python as a programming language due to its efficient, robust and powerful performances in the field of natural language processing (NLP), most specifically, topic modeling.

Concerning needed packages, we use *NLTK* (Natural Language Toolkit) library for data preprocessing. As regards topic modeling, we use *Gensim*, an open-source library, specialized major in topic modeling, which has shown its efficiency and flexibility for constructing multiple text representations (e.g., Bag-of-Words and Word2Vec), and training multiple topic models, like LDA, and also evaluating topic models by calculating their topic coherence scores. In connection with topic modeling, we use also embedded-topic-model package. To train our models, we use Google Colab on 2.2 Ghz CPU with 12.72 GB of RAM and 107.77 GB of disk.

4.2 Data Preprocessing

The dataset we used in our experimental study is the *20 Newsgroups*. This dataset comprises around 18,000 newsgroup documents that have been divided into 20 distinct newsgroups. We can upload the dataset from the scikit-learn library by importing the *fetch_20newsgroups* function from *sklearn.datasets* module. Before diving into experiments, we accomplish the some preprocessing techniques on the corpus to prepare it for the topic models.

First of all, we clean the dataset from any non-alphabetic characters (numbers, punctuations, line breaks, etc.) using regular expressions. We also proceed lower casing, deletion of words that have at most three characters, stop words removal and lemmatization.

Secondly, we construct a dictionary that contains the vocabulary corresponding to all newsgroup documents in the dataset, in order to encapsulate the mapping between each word from the vocabulary and its unique integer id. At this phase, the vocabulary size achieves 74,404 unique words.

Thirdly, we filter out the words in the dictionary by their frequency to remove the words that they are not valuable in identifying the topics from documents. We apply this idea by eliminating all the words that rarely appear in the corpus (words that are contained in less than 50 documents). In the other hand, we also exclude all the words that appear very frequently in the corpus (words that are contained in more than 75% of documents). On the fulfillment of this task, the vocabulary size reduces to become 3,224 unique words.

Finally, we aim to create a document-term-matrix by converting each document from the corpus into a bag-of-words format.

4.3 Topic Coherence Measures

Topic coherence measurements are utilized to assess the quality of topic models. We use four measures:

The *UCI* coherence (Equation (3)) is an extrinsic measure introduced by [20]. It uses the Pointwise Mutual Information (PMI) as a pairwise score function of all the n -top words pairs.

$$UCI(w_i, w_j) = \log \frac{P(w_i, w_j) + 1}{P(w_i).P(w_j)}. \quad (3)$$

Where $P(w_i)$ is the probability of seeing w_i in a random document and $P(w_i, w_j)$ is the probability of seeing both terms w_i and w_j co-occurring in a random document.

The *NPMI* coherence measure is an enhanced version of the *UCI* coherence using the Normalized Pointwise Mutual Information [3]. It is calculated as in Equation (4).

$$NPMI(w_i, w_j) = \frac{\log \frac{P(w_i, w_j) + 1}{P(w_i).P(w_j)}}{-\log(P(w_i, w_j) + 1)}. \quad (4)$$

The *Cv* coherence measure combines the indirect cosine measure with the Normalized Pointwise Mutual Information (NPMI) and the boolean sliding window [22].

The *UMass* coherence measure (Equation (5)) is an intrinsic measure based on document co-occurrence.

$$UMass(w_i, w_j) = \log \frac{D(w_i, w_j) + 1}{D(w_i)}. \quad (5)$$

Where $D(w_i)$ stands for the number of documents containing the term w_i and $D(w_i, w_j)$ is the number of documents containing both terms w_i and w_j .

4.4 Results and Discussion

After the implementation of both LDA and ETM models and data preprocessing, we finally get to the comparison between the two mentioned topic models using multiple topic coherence measures (*Cv*, *UCI*, *NPMI* and *UMass*) and the runtime factor.

For each approach, we consider different number of topics (100, 150, 200, 250, 300), then we calculate the *Cv*, *UCI*, *NPMI*, *UMass* coherence scores and the runtime score.

Table 1. LDA performance metrics.

Number of Topics	Cv Score	UCI Score	NPMI Score	UMass Score	Runtime Score
100	0.4892	-2.5879	-0.0416	-3.3551	1741
150	0.4447	-3.3807	-0.0943	-3.6061	1859
200	0.4173	-4.7760	-0.1333	-3.8150	2015
250	0.3982	-5.4577	-0.1637	-3.9794	2277
300	0.3900	-6.0864	-0.1898	-4.2436	2520

After training the LDA model, we calculate the desired topic coherence scores and the runtime (Table 1).

Before we run ETM model, we need to pass through two important operations.

1. Pretrain the embeddings on the corpus to create a Word2Vec model or load a pretrained one. In our experiments, we choose to work with Google’s Word2Vec pre-trained model (*Word2Vec-GoogleNews-vectors*). The model was trained on a corpus of 3 billion words, and contains 3 million English word vectors of size 300-dimension.
2. Transform the corpus into a format understandable by the model

Thereafter, we calculate the desired topic coherence scores and the runtime (Table 2).

Table 2. ETM performance metrics.

Number of Topics	Cv Score	UCI Score	NPMI Score	UMass Score	Runtime Score
100	0.5049	-0.0376	0.0308	-2.1721	1366
150	0.4924	-0.1281	0.0250	-2.2050	1523
200	0.4854	-0.1522	0.0224	-2.2273	1686
250	0.4729	-0.3281	0.0147	-2.3120	1961
300	0.4799	-0.3630	0.0139	-2.3042	2204

Comparison between LDA and ETM: To compare LDA and ETM approaches, we use the performance measures calculated earlier. Fig. 4 depicts these results by considering *Cv*, *UCI*, *NPMI* and *UMass* and Fig. 5 displays the runtime results.

We can clearly observe that *Cv*, *UCI*, *NPMI* and *NPMI* scores decrease with every increase in the number of topics for both LDA and ETM models (Fig. 4). Regarding the runtime (Fig. 5), we can see that it is proportional to the number of topics, and it is obvious that ETM is better than LDA.

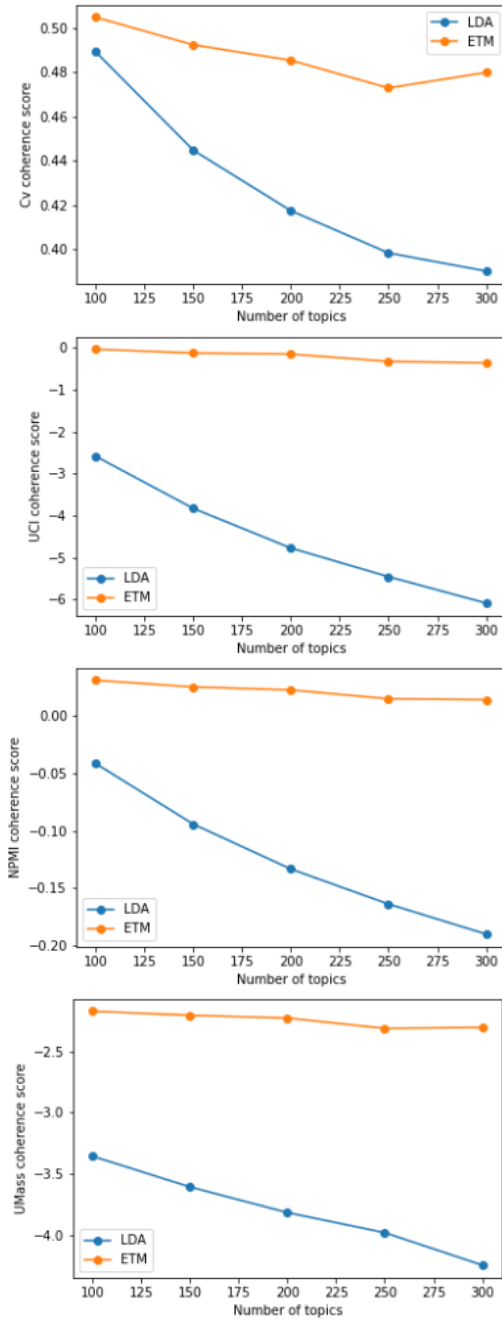


Fig. 4. Comparison results between LDA and ETM using *Cv*, *UCI*, *NPMI* and *UMass* scores.

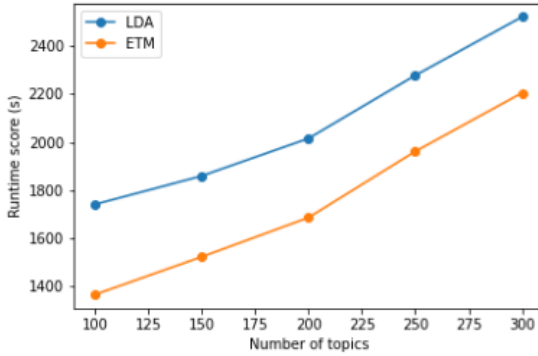


Fig. 5. Comparison results between LDA and ETM - runtime measure.

Considering all five measures we find that ETM absolutely outperforms LDA under the circumstances we performed our experimental study. We clarify the obtained result because of the semantic of words that are provided in the ETM model thanks to the word embedding technique.

5 Conclusions

Our paper focused on an unsupervised machine learning technique called Topic Modeling, also classified as a Natural Language Processing (NLP) technique. This technique is meant for extracting the hidden topics discussed in a collection of documents (corpus).

In our study, we presented a standard approach in the domain named Latent Dirichlet Allocation (LDA), and three other approaches referenced as Embedded Topic Model (ETM), Gaussian LDA (G-LDA), and LDA with Word2Vec (LDA2Vec). These approaches are the combination between LDA and the word embedding technique.

In this paper, we performed a comparative study between LDA and ETM approaches. We used the 20 newsgroups dataset as a corpus, and Google's pre-trained Word2Vec as a word embedding model. The results we got in terms of runtime and topic coherence measures were all in favor of the ETM approach.

As further work, we would like to work on an Arabic corpus, and also perform the study in larger corpora in order to get more efficient and accurate results.

Acknowledgments. This research is supported by the DGRSDT-Algeria, under the PRFU Project: C00L07UN470120230001.

References

1. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., Soroa, A.: A study on similarity and relatedness using distributional and wordnet-based approaches (2009)
2. Ahmad, A., Amin, M.R.: Bengali word embeddings and its application in solving document classification problem. In: 2016 19th International Conference on Computer and Information Technology (ICCIT). pp. 425–430. IEEE (2016)

3. Aletras, N., Stevenson, M.: Evaluating topic coherence using distributional semantics. In: Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers. pp. 13–22 (2013)
4. Allen, C., Hospedales, T.: Analogies explained: Towards understanding word embeddings. In: International Conference on Machine Learning. pp. 223–231. PMLR (2019)
5. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
6. Bellaouar, S., Bellaouar, M.M., Ghada, I.E.: Topic modeling: Comparison of LSA and LDA on scientific publications. In: DSDE '21: 2021 4th International Conference on Data Storage and Data Engineering, Barcelona, Spain, February 18–20, 2021. pp. 59–64. ACM (2021), <https://doi.org/10.1145/3456146.3456156>
7. Blei, D., Carin, L., Dunson, D.: Probabilistic topic models. *IEEE signal processing magazine* **27**(6), 55–65 (2010)
8. Blei, D.M., Lafferty, J.D.: Topic models. In: Text mining, pp. 101–124. Chapman and Hall/CRC (2009)
9. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *the Journal of machine Learning research* **3**, 993–1022 (2003)
10. Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J.L., Blei, D.M.: Reading tea leaves: How humans interpret topic models. In: Advances in neural information processing systems. pp. 288–296 (2009)
11. Das, R., Zaheer, M., Dyer, C.: Gaussian lda for topic models with word embeddings. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 795–804 (2015)
12. Dieng, A.B., Ruiz, F.J., Blei, D.M.: Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics* **8**, 439–453 (2020)
13. Dos Santos, C., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 69–78 (2014)
14. Ethayarajh, K., Duvenaud, D., Hirst, G.: Towards understanding linear word analogies. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 3253–3262. Association for Computational Linguistics, Florence, Italy (Jul 2019). <https://doi.org/10.18653/v1/P19-1315>, <https://www.aclweb.org/anthology/P19-1315>
15. Kim, H.K., Kim, H., Cho, S.: Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing* **266**, 336–352 (2017)
16. Maas, A., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies. pp. 142–150 (2011)
17. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
18. Moody, C.E.: Mixing dirichlet topic models and word embeddings to make lda2vec. arXiv preprint arXiv:1605.02019 (2016)
19. Newman, D., Karimi, S., Cavedon, L.: External evaluation of topic models. In: in Australasian Doc. Comp. Symp., 2009. Citeseer (2009)
20. Newman, D., Lau, J.H., Grieser, K., Baldwin, T.: Automatic evaluation of topic coherence. In: Human language technologies: The 2010 annual conference of the

- North American chapter of the association for computational linguistics. pp. 100–108 (2010)
21. Ramos, J., et al.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning. vol. 242, pp. 29–48. Citeseer (2003)
 22. Röder, M., Both, A., Hinneburg, A.: Exploring the space of topic coherence measures. In: Proceedings of the eighth ACM international conference on Web search and data mining. pp. 399–408 (2015)
 23. Rong, X.: word2vec parameter learning explained. arXiv preprint arXiv:1411.2738 (2014)
 24. Sawyer, S.: Wishart distributions and inverse wishart sampling. URL: www.math.wustl.edu/sawyer/hmhandouts/Whishart.pdf (2007)
 25. Zeng, D., Liu, K., Chen, Y., Zhao, J.: Distant supervision for relation extraction via piecewise convolutional neural networks. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 1753–1762 (2015)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

