# Clustering Documents using the Document to Vector Model for Dimensionality Reduction

Robert-George Radu[1], Iulia-Maria Rădulescu[2], Ciprian-Octavian Truică[3],
Elena-Simona Apostol[4], Mariana Mocanu[5]
Computer Science and Engineering Department, Faculty of Automatic Control and Computers
University Politehnica of Bucharest, Bucharest, Romania
Email: [1]robert_george.radu@stud.acs.upb.ro, [2]iulia.radulescu@cs.pub.ro, [3]ciprian.truica@cs.pub.ro,
[4]elena.apostol@cs.pub.ro, [5]mariana.mocanu@cs.pub.ro

***Abstract*—The TF-IDF model is the most common way of representing documents in the vector space. However, its results are highly dimensional, posing problems to the classic clustering algorithms due to the curse of dimensionality. Recent word embeddings based techniques can reduce the documents representations dimensionality while also preserving the semantic relationships between words. In this paper, we analyze the accuracy of four different classical clustering algorithms (K-Means, Spherical K-Means, LDA, and DBSCAN) in combination with the Document to Vector model.**

***Index Terms*—text clustering, document embeddings, text preprocessing, clustering evaluation**

## I. Introduction

Clustering is a data mining task which aims to group similar objects based on a dissimilarity measure. One of the many uses of clustering is textual data analysis using document clustering, which plays an important role in document retrieval, web search and spam filtering [9]. Many methods for clustering documents have been proposed, most of them using a term-frequency inverse-document-frequency matrix (TF-IDF matrix) to represent a corpus on which a chosen clustering method is applied [3]. However, the TF-IDF model has several drawbacks: it does not consider the semantic similarities between words, neither the word order and produces a high dimensional representation which often must be reduced using Principal Component Analysis or similar techniques [9]. The Paragraph Vector or Document to Vector (Doc2Vec) model [9] overcomes these disadvantages by representing words as n-dimensional vectors learnt using the word context.

In this paper we assess the Doc2Vec model's accuracy by combining it with i) two distance based algorithms used in text clustering, i.e., K-Means [10] and Spherical K-Means [5], ii) a density based algorithm inclined to be affected by the curse of dimensionality [1], i.e., DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [4], iii) and a topic modeling algorithm also used for clustering, i.e., Latent Dirichlet Allocation (LDA) [2]. As shown in the results section, all these algorithms obtain improved accuracy when paired with the Doc2Vec model.

We also preprocessed our corpus with i) two types of stemmers: the Porter stemmer and the Lancaster Stemmer and with ii) one lemmatizer: the WordNet lemmatizer, in order to demonstrate the independence of our results to the applied preprocessing technique.

The paper is organized as follows. Section II presents other evaluations of the Doc2Vec model used for clustering. Section III describes the pipeline for preprocessing, embedding, and clustering documents as well as the evaluation methods. In Section IV we present our experimental setup and discuss the results. In Section V we summarize our findings and hint at future work.

978-1-7281-7166-1/20/$31.00 

## II. Related Work

A number of authors have evaluated the performance of the Doc2Vec model in combination with different clustering algorithms.

In their paper [3] Curiskis, Stephan A., et al. evaluate $4$ feature representation methods derived from TF-IDF and embedding matrices (the TF-IDF matrix, the mean Word2Vec matrix, the mean Word2Vec matrix weighted by the TF-IDF scores and the Doc2Vec matrix for each document) combined with $4$ clustering techniques (the K-Means, K-Medoids, Hierarchical Agglomerative and Non Negative Matrix Factorization algorithms) on Twitter data and Reddit comments. The K-Means algorithm combined with the Doc2Vec model obtains the best Adjusted Rand Index scores.

Israel Mendonça et al. [11] evaluate the results of K-Means, Spectral Clustering, Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM), Mean Shift and DBSCAN in combination with the Doc2Vec model, but since they are using a vector size of $300$ real numbers, they obtain low accuracy for DBSCAN.

Víctor Mijangos, Gerardo Sierra and Azucena Montes [12] apply a spectral relaxation procedure on the Doc2Vec model's document matrix representation in order to reduce the dimensionality of the dataset, then they run K-Means on the preprocessed documents. In order to employ spectral clustering on the document vectors, the vectors are transformed into a graph using a K-Nearest Neighbors approach, defining a kernel function which computes the weighted adjacency matrix of the graph. The dataset used for the experiments is Sexualities of Mexico. The results obtained a Rand Index value of $0.538$

when using Doc2Vec model compared to a Rand Index value of 0.390 obtained with the Bag of Words model.

Extensive experiments were performed using the Doc2Vec model to cluster a corpus of 82 793 Lithuanian news articles [20] such as: i) which is the best number of articles - document model combination (comparing Doc2Vec and Bag Of Words), ii) whether lemmatization improves the results, iii) what is the most advantageous number of training epochs and vector size respectively.

## III. Methodology

The pipeline for processing, embedding, and clustering documents (Figure 1) consists of the following modules: i) the text preprocessing module that employees stemming and lemmatization to minimize the vocabulary and remove duplicate concepts, ii) the document embeddings module used for embedding the documents using either TF-IDF and Doc2Vec models, and iii) the clustering module for detecting the documents that have similar hidden structures.

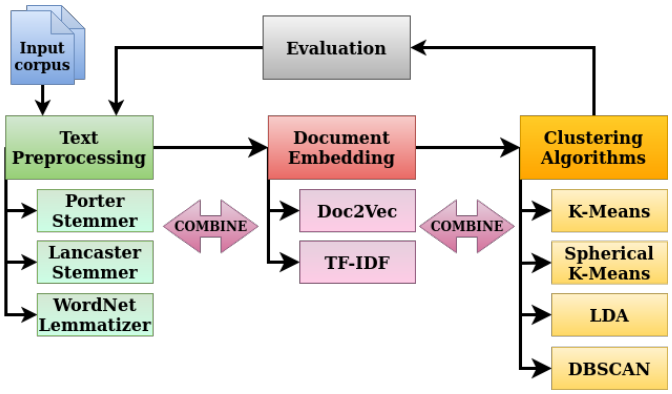


Fig. 1. Architecture pipeline

### A. Text preprocessing

We use preprocessing techniques on the textual data before applying the clustering algorithms to achieve meaningful results and reduce noise. One of the problems which occur when dealing with a large amount of textual data is language inflections: treating words with similar meaning differently (for example *play*, *players*, *payer*, *playing* should be normalized to their root form). We use the Porter Stemmer, Lancaster Stemmer, and the WordNet Lemmatizer for structuring, cleaning, and minimizing the vocabulary of the input corpus.

*1) Porter Stemmer:* Porter's algorithm [16] is the best known stemming algorithm for English. It is mostly used for word normalization of inflected forms for Information Retrieval [7]. Over time it has proven to be very efficacious in practice. The algorithm is based on several word truncation rules. The rules are grouped into five sequentially applied phases. All five phases consist of rules and rules selection conventions. One of the conventions specifies the application of the rules on the longest suffix. Also, for a rule to be applied to a word, certain conditions must be met. An example would be the condition that the number of syllables after truncation would be greater than 1. Thus for the word *understatement*, the rule is valid (*understatement* $\rightarrow$ *understat*), while for the word *element* the rule it is not valid (*element* $\rightarrow$ *element*).

*2) Lancaster Stemmer:* The Lancaster stemmer[15], also known as Paice/Husk stemmer, belongs to the category of algorithmic stemmers ordered by the last letter of a suffix (suffixes are detached gradually in an undetermined number of phases), trying to find a suitable rule for the current letter on each step. It headlines an independent cached set of 129 stemming rules, disposed of in a rule file text, any of which may define the elimination or substitution of a suffix. Due to the predefined set of rules, the stemmer is considered *strong*, making it beneficial for index compression.

The efficiency of the algorithm is maintained by the substitution approach which bypasses the need for a separate step in the mechanism to rewrite or add partial matching. Moreover, indexing the rules based on the last letter helps to contribute to an efficient search. The algorithm ends if there is no rule available, if there are only two letters left and a word starts with a vowel, or if there are only three letters left and a word starts with a consonant. In other circumstances, the rule is applied and the process reiterates.

*3) WordNet lemmatizer:* Unlike stemmers who use language-based rules, lemmatizers require an entire vocabulary and morphological analysis of words to accurately lemmatize the texts. Lemmatizers are an NLP (Natural Language Processing) tool that offers a more complex alternative to stemming. WordNet [14] is a wide, openly, and free linguistic database for the English language proposing to provide systematic grammatical associations between words. It allows lemmatization facilities along and it is one of the most frequently used and most original lemmatizers.

All parts of speech are arranged into collections of experimental synonyms, each signifying a definite notion. The sets are interlaced by instruments of theoretical – morphological and linguistic lexical relations. Therefore, WordNet is a valuable mechanism for computational syntax and natural language processing.

WordNet gathers words based on their meaning, being the equivalent of a vocabulary. However, there are some essential features. Firstly, it manages to find the meaning of words and offers disambiguation for them at the phrase level by adding context. Secondarily, WordNet classifies the syntactic relations between words organized in a certain order as a vocabulary instead of only comprising the words of a language.

### B. Document embeddings

The preprocessed corpus must be represented numerically in order to be understood by the clustering algorithms. The most popular model that accomplishes this task is TF-IDF [8], [22], which represents each word from a given document using the number of its occurrences in that given document and the number of its occurrences in the whole corpus. However, the more recent Doc2Vec model

computes context-dependent representations which preserve the semantic similarities between words while reducing the dataset's dimensionality. For a corpus of documents $D = \{d_1, d_2, ..., d_M\}$, where $M = ||D||$ is the total number of documents in the dataset, a document $d_i$ is defined as a sequence of terms $w_{ij}$, $d_i = \{w_{i1}, w_{i2}, ..., w_{iV}\}$ where $V$ is the length of the vocabulary. The vocabulary is the set of distinct words that appear in the corpus of documents.

*1) Vector Space Model:* The Vector Space Model is a vectorized representation of text documents that assigns a weight, e.g., TF-IDF, to each document-term pair. TF-IDF ($TFIDF(w_{ij}, d_i, D) = TF(w_{ij}, d_i) \cdot IDF(w_{ij}, D)$) is computed as the product between the term frequency of a term $w_{ij}$ in a document $d_i$ ($TF(w_{ij}, d_i) = f_{w_{ij}, d_i}$) and the inverse-term frequency ($IDF(w_{ij}, D) = \log_2 \frac{M}{M_{ij}}$). *IDF* is a statistical measure that computes the importance of a term $w_{ij}$ in a corpus of documents $D$ by counting the number of documents $M_{ij}$ where the term appears.

*2) Document to Vector:* The Document to Vector (Doc2Vec) model, also called Paragraph Vector, learns continuous distributed vector representations for pieces of text [9]. The text dimension may vary from phrases and sentences to large documents. The Doc2Vec model is similar to the Word2Vec model [13] which maps words into the vector space maintaining the semantic similarities by using a given word's context. Both approaches rely on the distributional hypothesis [18] which states that words with similar meaning appear in similar contexts. The word-vector space mapping is obtained using a shallow neural network with one hidden layer, which represents the word's embeddings. Two neural network architectures are proposed for word embeddings [13]: the Continuous Bag of Words architecture, which predicts a word given the context, and the Skip-Gram Model predicts the context given a word. The Paragraph Vectors Distributed Memory (PVDM) model [9] extends the Continuous Bag of Words architecture by mapping each document to a vector via an additional document-to-vector matrix and concatenating this vector to the word vectors in order to predict the central word. Given a sequence of training words $w_i = \{w_1, w_2, ..., w_T\}$, the objective of the PVDM model is to minimize the log probability of the prediction function $\pi = p(w_t | w_{t-k}, ..., w_{t+k})$. The prediction uses the softmax multiclass classifier $\pi = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}$, where $y_i = b + Uh(w_{t-k}, ..., w_{t+k})$ is an un-normalized log-probability for each output word $i$ with $U$ and $b$ the softmax parameters and $h$ the function that averages the word vectors.

### C. Text clustering

Text clustering is the task of grouping similar documents together by analysing the textual data from a structural (i.e., K-Means, Spherical K-Means, and DBSCAN) or semantic (i.e., LDA) perspective.

*1) K-Means Algorithm:* K-means [10] is an iterative algorithm that aims to find optimal spherical cluster structures and centroids. The K-Means algorithm works as follows: 1) the number of clusters $K$ is chosen and the centroids $c_k$ are initialized, 2) every document $d_i$ is assigned to the closest cluster using the Euclidean distance, 3) the new coordinates of the centroids are calculated as the average of the term weights of each document in each cluster. Steps 2) and 3) are repeated until the algorithm converges. The quality of a cluster $C_k$ is measured using the squared-error sum (Equation (1)). A low-value corresponds to a *well-formed* cluster.

$$\sum_{k=1}^{K} \sum_{d_i \in C_k} (d_i - c_k)^2 \tag{1}$$

*2) Spherical K-Means Algorithm:* Spherical K-Means [5] is an adaptation of the K-Means algorithm for text clustering that uses cosine distance as a measure of dissimilarity between objects. The algorithm is based on the observation that the document size is irrelevant in the clustering process, therefore normalizing each of the feature vectors to be of unit length. The objective of the spherical K-Means algorithm is to minimize the sum of distances $\delta$ for every document $d_i$ ($i = \overline{1, M}$) and the corresponding cluster centroid $c_k$ ($k = \overline{1, K}$) (Equation (2)).

$$\delta(d_i, c_k) = 1 - cos(d_i, c_k) = 1 - \frac{< d_i, c_k >}{||d_i|| \cdot ||c_k||} \tag{2}$$

The solution for spherical K-Means can also be extended by performing soft clustering by moving points between clusters for finding optimal group memberships.

*3) DBSCAN (Density-Based Spatial Clustering of Applications with Noise) Algorithm:* DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [4] is a density based clustering algorithm suitable for arbitrary shaped clusters with noise. It is based on the concepts of density reachability and density connectivity given the input parameters $\epsilon$, the nearest neighbor radius, and the minimum number of neighbors of an object necessary to expand a new cluster (*minPts*). DBSCAN defines clusters in terms of density as subsets of the input dataset which satisfy the maximality and connectivity conditions.

Because it relies on nearest neighbor queries, DBSCAN is inclined to be affected by the curse of dimensionality: the concept of nearest neighbors is less meaningful in high dimensions due to the fact that the distances between a given object and its nearest and farthest neighbours become indistinguishable [1]. However, by reducing the dimensionality of the input dataset using the Doc2Vec model we obtain accurate results.

DBSCAN can be paired with any distance function adequate to the dataset to be analyzed [19]. Thus, we used the cosine dissimilarity measure in our experiments. We preferred the cosine distance function over the euclidean distance function since it is more appropriate for clustering textual high-dimensional data.

*4) Latent Dirichlet Allocation:* Latent Dirichlet Allocation (LDA) [2] is a generative probabilistic model of a document collection. Documents are represented as random mixtures over latent (hidden) topics, where each topic is characterized

by a distribution over words. LDA assumes the following generative process for each document: i) the number of words comprising the document is sampled from a Poisson distribution, ii) the document's topic distribution $\theta$ is sampled from a Dirichlet distribution $Dir(\alpha)$, iii) each word in the document is generated by a topic $z_n$ sampled from the multinomial distribution $Multinomial(\theta)$. A word $w_n$ is chosen from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic $z_n$, i.e., the probability that the topic $z_n$ contains the word $w_n$.

Even though LDA is generally effective for topic modelling, it is also appropriate for document clustering by using a hard-clustering approach, i.e., assigning each document to the topic with the highest probability.

### D. Evaluation measures

Multiple evaluation methods have been proposed in the literature [24]. These methods are grouped into three categories: i) evaluation measures based on counting of pairs of elements, ii) evaluation methods that use the summation of set overlaps, and iii) evaluation that uses the information-theoretical mutual information. We use five evaluation measures, four from the first category (Homogeneity, Completeness, V-Score, Adjusted Rand Index) and one from the last category (Adjusted Mutual Information score). We further describe them in detail.

*1) Homogeneity score:* A homogeneous clustering is obtained when the objects belonging to a single class are assigned to a single cluster. Thus, each cluster contains objects from the same, single class. The homogeneity score (Equation (3)) [17] determines how close is a given clustering to the ideal result by examining the conditional entropy of the ground truth class distribution given the clustering results [17].

$$h = \begin{cases} 1 \text{ if } H(K,C) = 0 \\ 1 - \frac{H(C|K)}{H(C)} \text{ otherwise} \end{cases} \tag{3}$$

In 3, $C$ represents the ground truth clustering, $K$ represents the result obtained by the employed clustering method and $H(C|K)$ represents the conditional entropy of the class distribution given the clustering results.

The homogeneity score's values range from 0 (none of the clusters contains points from the same class) to 1 (perfectly homogeneous labeling).

*2) Completeness score:* Completeness is symmetric to homogeneity in that it relies on calculating the conditional entropy of the clustering results distribution given the ground truth classes ($H(K|C)$). The completeness score [17] is formally defined in Equation (4).

$$c = \begin{cases} 1 \text{ if } H(K,C) = 0 \\ 1 - \frac{H(K|C)}{H(K)} \text{ otherwise} \end{cases} \tag{4}$$

The completeness score values range from 0 (none of the points from a class belong to the same cluster) to 1 (perfectly complete labeling).

*3) V-measure score:* The V-measure score (Equation (5)) [17] is equal to the harmonic mean between completeness and homogeneity.

$$\frac{(1+\beta) \times h \times c}{\beta \times h \times c} \tag{5}$$

The $\beta$ parameter in Equation 5 represents the ratio of weight between homogeneity and completeness ($\beta > 1$ indicates that completeness is weighted more than homogeneity while $\beta < 1$ indicates that completeness is weighted less than homogeneity). The result is independent of the equivalence between the values obtained by the algorithm and those defined as ground truth (if we change the classes or clusters between them, we will obtain the same completeness score). If we invert the points in the clusters with the labels in the classes we will obtain the same V-measure score, meaning that V-measure is a symmetric metric. The range of values is from 0 (none of the points from a class belong to the same cluster or none of the clusters contains points from the same class) to 1 (perfectly complete labeling).

*4) Adjusted Rand Index score:* The Adjusted Rand Index score measures the similarity between the obtained clustering results and the ground truth and can be formally defined as follows [21]: Given a set $S = \{s_1, s_2, ..., s_n\}$ of n elements and two groupings $X = \{X_1, X_2, ..., X_r\}$ and $Y = \{Y_1, Y_2, ..., Y_t\}$ the overlap between X and Y can be summarized in contingency table $[n_{ij}]$ where $n_ij = |X_i \cap Y_i|, a_i = \sum_{j=1}^{s} n_{ij}$ and $b_j = \sum_{i=1}^{r} n_{ij}$. Using the contingency table, the Adjusted Rand Index is defined in Equation (6).

$$ARI = \frac{\sum_{i,j} \binom{n_{ij}}{2} - \frac{\left(\sum_i \binom{a_i}{2}\right)\left(\sum_j \binom{b_j}{2}\right)}{\binom{n}{2}}}{\frac{1}{2}\left(\sum_i \binom{a_i}{2} + \sum_b \binom{b_j}{2}\right) - \frac{\left(\sum_i \binom{a_i}{2}\right)\left(\sum_j \binom{b_j}{2}\right)}{\binom{n}{2}}} \tag{6}$$

The Adjusted Rand Index score ranges from 0 (complete disagreement between the obtained clustering results and the ground truth) and 1 (maximum level of agreement between the clustering results and the ground truth).

*5) Adjusted Mutual Information score:* The Adjusted Mutual Information score (Equation (7)) [23] is also determined by the level of agreement between the obtained clustering results ($U$) and the ground truth ($V$).

$$\frac{MI(U,V) - E(C,K)}{avg(H(C), H(K)) - E(MI(U,V))} \tag{7}$$

In (7), $MI$, $E$ and $H$ represent the mutual information between clusterings, the expected mutual information and the entropy, respectively. The values range from $-\infty$ to 1 (perfect match). Random cluster labeling will have an Adjusted Mutual Information approximate to 0.

## IV. Experimental Results

To assess the accuracy of the algorithms w.r.t. the Doc2Vec model, we have collected 12 263 articles from Arxiv[1] labeled

[1] Arxiv https://arxiv.org/

with the following tags: Databases, Geophysics, Metric Geometry, Genomics, and Economics. The dataset is presented in Table I. The source code is available online[2].

TABLE I
TEST CORPORA ATTRIBUTES

| | |
|---|---|
| Number of documents | 12 263 |
| Number of clusters | 5 |
| Mean number of words per document with lemmatizer | 151 |
| Mean number of documents per category | 2 453 |

For the Doc2Vec embedding, we use the PVDM model with the hierarchical softmax activation function, a vector size of 16 real numbers and a window size of 3 words. The model was trained for 200 epochs. Since the training corpus is small, a small vector size does not impact the accuracy of the concepts' representation. We also ran K-Means and Spherical K-Means with a Doc2Vec model with a vector size of 128 real numbers and we obtained similar results, thus verifying that the small dimensionality does not affect the model's accuracy.

For the DBSCAN algorithm, the *minPts* parameter varies according to the preprocessing method for a constant $\epsilon = 0.275$. In our experiments, we set $minPts = 340$ for the tests that use the Porter stemmer, $minPts = 250$ for the experiments that use the Lancaster stemmer and $minPts = 404$ for the tests that use WordNet Lemmatizer. We have chosen these values after we performed hyperparameter tuning.

For the K-Means, Spherical K-Means, and LDA algorithms we set the number of clusters, respectively the number of topics to 5.

The results for each algorithm w.r.t. the document embedding are shown in Tables II, III, IV, V, and VI. The DBSCAN algorithm displays the most important improvement; for example, the Adjusted Mutual Information score is 0.6 when using Doc2Vec model as compared to 0.001 when using TF-IDF. This happens because DBSCAN is not accurate in high dimensions. However, even for the Doc2Vec model, DBSCAN considers many documents to be noise: 7 145, 7 783, and 9 410 documents preprocessed by the Porter and Lancaster stemmers and the WordNet lemmatizer respectively.

We note that when using the highly dimensional representation computed by TF-IDF, DBSCAN assigns all the documents to the noise cluster, thus explaining the perfect Completeness score.

K-Means and Spherical K-Means also obtain improved results for the Doc2Vec model. The Adjusted Rand Index for the K-Means algorithm increases from 0.62 when used with the TF-IDF embedding to 0.83 when used with Doc2Vec embedding. The lowest Adjusted Rand Index values are obtained for the Lancaster stemmed corpus for both TF-IDF and Doc2Vec models, i.e., for K-Means the results are 0.63 and 0.78, while for Spherical K-Means the values are 0.71 and 0.78. This happens because the Lancaster stemmer is aggressive and frequently over-stems words [6]. These results show that, similar to TF-IDF, Doc2Vec is affected by the incorrect assignment of words with different meaning to the same root.

The evaluation measures for LDA are generally better than for the Doc2Vec model, thus demonstrating that this representation preserves the document-topics and topic-words relationships. The biggest difference of the Adjusted Multual Information w.r.t. the two embeddings models is obtained by LDA with the text lemmatized using the WordNet: 0.19. Thus, when each word is reduced to its correct root, Doc2Vec model yields a significantly better representation for LDA.

TABLE II
K-MEANS TF-IDF VS. DOC2VEC HOMOGENEITY COMPARISON

| **Algorithm** | **Porter Stemmer** | **Lancaster Stemmer** | **WordNet Lemmatizer** |
|---|---|---|---|
| DBSCAN TF-IDF | 8.03e-16 | 8.03e-16 | 8.03e-16 |
| DBSCAN Doc2Vec | 0.54 | 0.47 | 0.56 |
| K-Means TF-IDF | 0.67 | 0.67 | 0.69 |
| K-Means Doc2Vec | 0.79 | 0.76 | 0.79 |
| Spherical K-Means TF-IDF | 0.75 | 0.74 | 0.75 |
| Spherical K-Means Doc2Vec | 0.79 | 0.76 | 0.79 |
| LDA TF-IDF | 0.70 | 0.57 | 0.56 |
| LDA Doc2Vec | 0.70 | 0.66 | 0.75 |

TABLE III
K-MEANS TF-IDF VS. DOC2VEC COMPLETENESS COMPARISON

| **Algorithm** | **Porter Stemmer** | **Lancaster Stemmer** | **WordNet Lemmatizer** |
|---|---|---|---|
| DBSCAN TF-IDF | 1.00 | 1.00 | 1.00 |
| DBSCAN Doc2Vec | 0.56 | 0.56 | 0.65 |
| K-Means TF-IDF | 0.71 | 0.71 | 0.74 |
| K-Means Doc2Vec | 0.78 | 0.74 | 0.78 |
| Spherical K-Means TF-IDF | 0.77 | 0.76 | 0.76 |
| Spherical K-Means Doc2Vec | 0.77 | 0.74 | 0.78 |
| LDA TF-IDF | 0.71 | 0.60 | 0.55 |
| LDA Doc2Vec | 0.68 | 0.66 | 0.73 |

TABLE IV
TF-IDF VS. DOC2VEC V-SCORE COMPARISON

| **Algorithm** | **Porter Stemmer** | **Lancaster Stemmer** | **WordNet Lemmatizer** |
|---|---|---|---|
| DBSCAN TF-IDF | 1.6e-15 | 1.6e-15 | 1.6e-15 |
| DBSCAN Doc2Vec | 0.55 | 0.51 | 0.60 |
| K-Means TF-IDF | 0.69 | 0.69 | 0.71 |
| K-Means Doc2Vec | 0.79 | 0.75 | 0.79 |
| Spherical K-Means TF-IDF | 0.76 | 0.75 | 0.76 |
| Spherical K-Means Doc2Vec | 0.78 | 0.78 | 0.79 |
| LDA TF-IDF | 0.70 | 0.58 | 0.55 |
| LDA Doc2Vec | 0.69 | 0.66 | 0.74 |

## V. CONCLUSIONS

In this paper, we evaluated the Doc2Vec model's accuracy by combining it with four different clustering algorithms: K-Means, Spherical K-Means, LDA, DBSCAN and comparing the results with the ones obtained when using the TF-IDF model. We preprocessed the input corpus with two stemmers and a lemmatizer in order to verify that the Doc2Vec

[2]Github https://github.com/IuliaRadulescu/DBSCANDoc2Vec

TABLE V
TF-IDF VS. DOC2VEC ADJUSTED RAND INDEX COMPARISON

| Algorithm | Porter Stemmer | Lancaster Stemmer | WordNet Lemmatizer |
|---|---|---|---|
| DBSCAN TF-IDF | 0 | 0 | 0 |
| DBSCAN Doc2Vec | 0.40 | 0.40 | 0.54 |
| K-Means TF-IDF | 0.62 | 0.63 | 0.64 |
| K-Means Doc2Vec | 0.83 | 0.78 | 0.83 |
| Spherical K-Means TF-IDF | 0.72 | 0.71 | 0.72 |
| Spherical K-Means Doc2Vec | 0.82 | 0.78 | 0.83 |
| LDA TF-IDF | 0.75 | 0.58 | 0.52 |
| LDA Doc2Vec | 0.72 | 0.65 | 0.78 |

TABLE VI
TF-IDF VS. DOC2VEC ADJUSTED MUTUAL INFORMATION COMPARISON

| Algorithm | Porter Stemmer | Lancaster Stemmer | WordNet Lemmatizer |
|---|---|---|---|
| DBSCAN TF-IDF | 8.03e-16 | 8.03e-16 | 8.03e-16 |
| DBSCAN Doc2Vec | 0.55 | 0.51 | 0.60 |
| K-Means TF-IDF | 0.69 | 0.69 | 0.71 |
| K-Means Doc2Vec | 0.79 | 0.75 | 0.79 |
| Spherical K-Means TF-IDF | 0.76 | 0.75 | 0.76 |
| Spherical K-Means Doc2Vec | 0.78 | 0.75 | 0.79 |
| LDA TF-IDF | 0.70 | 0.58 | 0.55 |
| LDA Doc2Vec | 0.69 | 0.66 | 0.74 |

embeddings are independent of the preprocessing method. The experimental evaluation proves empirically that reducing the dimensions improves the quality of clusters while maintaining the cluster inner structures and the latent semantic space of terms by preserving the document-topics relations.

The DBSCAN with the Doc2Vec embedding improved the most: the Adjusted Mutual Information increased from $0.001$ to $0.6$. Thus, the experiments show that density-based algorithms, which are greatly affected by the curse of dimensionality, can improve the documents clustering when the dimensions are reduced and the corpus is modeled using Doc2Vec.

The centroid-based algorithms obtain slightly better results when using Doc2Vec instead of TF-IDF. The Adjusted Mutual Information values for K-Means increase from $0.69$ to $0.79$, while for Spherical K-Means increase from $0.76$ to $0.79$. The evaluation shows that the Doc2Vec model improves cluster compactness for centroid-based clustering algorithms while significantly reducing dimensionality.

When using topic modeling, the results show that the Doc2Vec model preserves the document-topics relationship. For the corpus preprocessed with the Porter Stemmer, LDA obtains similar Adjusted Mutual Information results ($\sim 0.70$) regardless of the document embedding, i.e., TF-IDF and Doc2Vec models.

As future directions, we plan to run the aforementioned experiments on a larger corpus with more similar document categories. We will use the Arxiv API to extract sub-categories of the main categories and try to improve the accuracy of clustering with Doc2Vec model.


## ACKNOWLEDGEMENT

This research was funded through the subsidiary contract IntAli 20175/30.10.2019 for the grant 53/05.09.2016, ID 40270, MySMIS code: 105976


## REFERENCES


[1] I. Assent, "Clustering high dimensional data," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, no. 4, pp. 340–350, 2012.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," Journal of machine Learning research, vol. 3, pp. 993–1022, 2003.

[3] S. A. Curiskis, B. Drake, T. R. Osborn, and P. J. Kennedy, "An evaluation of document clustering and topic modelling in two online social networks: Twitter and reddit," Information Processing & Management, p. 102034, 2019.

[4] M. Ester, H.-P. Kriegel, J. Sander et al., "A density-based algorithm for discovering clusters in large spatial databases with noise." in KDD, vol. 96, no. 34, 1996, pp. 226–231.

[5] K. Hornik, I. Feinerer, M. Kober, and C. Buchta, "Spherical k-means clustering," Journal of Statistical Software, vol. 50, no. 10, pp. 1–22, 2012.

[6] A. G. Jivani, "A comparative study of stemming algorithms," Int. J. Comp. Tech. Appl., vol. 2, no. 6, pp. 1930–1938, 2011.

[7] K. Kettunen, T. Kunttu, and K. Järvelin, "To stem or lemmatize a highly inflectional language in a probabilistic ir environment?" Journal of Documentation, vol. 61, no. 4, pp. 476–496, 2005.

[8] D. Kim, D. Seo, S. Cho, and P. Kang, "Multi-co-training for document classification using various document representations: TF–IDF, LDA, and doc2vec," Information Sciences, vol. 477, pp. 15–29, 2019.

[9] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in ICML, 2014, pp. 1188–1196.

[10] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Berkeley symposium on mathematical, statistics, and probability, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[11] I. Mendonça, A. Trouvé, A. Fukuda et al., "On clustering algorithms: applications in Word-Embedding documents," Journal of Computers, vol. 14, no. 2, pp. 88–92, 2019.

[12] V. Mijangos, G. Sierra, and A. Montes, "Sentence level matrix representation for document spectral clustering," Pattern Recognition Letters, vol. 85, pp. 29–34, 2017.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in ICLR, 2013.

[14] G. A. Miller, "WordNet: a lexical database for english," Communications of the ACM, vol. 38, no. 11, pp. 39–41, 1995.

[15] C. D. Paice, "Another stemmer," ACM SIGIR, vol. 24, no. 3, pp. 56–61, 1990.

[16] M. Porter, "An algorithm for suffix stripping," Program, vol. 14, no. 3, pp. 130–137, 1980.

[17] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in EMNLP-CoNLL, 2007, pp. 410–420.

[18] M. Sahlgren, "The distributional hypothesis," Italian Journal of Disability Studies, vol. 20, pp. 33–53, 2008.

[19] E. Schubert, J. Sander, M. Ester et al., "DBSCAN revisited," ACM TODS, vol. 42, no. 3, pp. 1–21, 2017.

[20] L. Stankevičius and M. Lukoševičius, "Lithuanian news clustering using document embeddings," in International Conference on Information Technologies, 2019, pp. 104–109.

[21] C.-O. Truică, O. Novović, S. Brdar, and A. N. Papadopoulos, "Community detection in who-calls-whom social networks," in DaWaK. Springer, 2018, pp. 19–33.

[22] C.-O. Truică, F. Rădulescu, and A. Boicea, "Comparing different term weighting schemas for topic modeling," in SYNASC, 2016, pp. 307–310.

[23] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," JMLR, vol. 11, pp. 2837–2854, 2010.

[24] S. Wagner and D. Wagner, Comparing clusterings: an overview. Universität Karlsruhe, Fakultät für Informatik Karlsruhe, 2007.