

Evaluation of different text representation techniques and distance metrics using KNN for documents classification


Evaluación de distintas técnicas de representación de texto y medidas de distancia de texto usando KNN para clasificación de documentos

Luis Alexander Calvo-Valverde¹, José Andrés Mena-Arias²

Calvo-Valverde, L; Mena-Arias, J. Evaluation of different text representation techniques and distance metrics using KNN for documents classification. *Tecnología en Marcha*. Vol. 33-1. Enero-Marzo. Pág 64-79.

 <https://doi.org/10.18845/tm.v33i1.5022>

¹ DOCINADE, Instituto Tecnológico de Costa Rica, Cartago, Costa Rica. Correo electrónico: lcalvo@tec.ac.cr.

 <https://orcid.org/0000-0003-3802-9944>

² Maestría en Computación, Instituto Tecnológico de Costa Rica. Cartago, Costa Rica. Correo electrónico: andres.menaarias@gmail.com.



Keywords

Text similarity; text classification; KNN; topic modeling.

Abstract

Nowadays, text data is a fundamental part in databases around the world and one of the biggest challenges has been the extraction of meaningful information from large sets of text. Existing literature about text classification is extensive, however, during the last 25 years the statistical methods (where similarity functions are applied over vectors of words) have achieved good results in many areas of text mining. Additionally, several models have been proposed to achieve dimensional reduction and incorporate the semantic factor, such as the topic modelling. In this paper we evaluate different text representation techniques including traditional bag of words and topics modelling. The evaluation is done by testing different combinations of text representations and text distance metrics (Cosine, Jaccard and Kullback-Leibler Divergence) using K-Nearest-Neighbors in order to determine the effectiveness of using topic modelling representations for dimensional reduction when classifying text. The results show that the simplest version of bag of words and the Jaccard similarity outperformed the rest of combinations in most of the cases. A statistical test showed that the accuracy values obtained when using supervised Latent Dirichlet Allocation representations, combined with the relative entropy metric, were no significantly different to the ones obtained by using traditional text classification techniques. LDA managed to abstract thousands of words in less than 60 topics for the main set of experiments. Additional experiments suggest that topic modelling can perform better when used for short text documents or when increasing the parameter of number of topics (dimensions) at the moment of generating the model.

Palabras clave

Similitud de texto; clasificación de texto; KNN; modelado de temas.

Resumen

Actualmente, los datos textuales constituyen una parte fundamental de las bases de datos de todo el mundo y uno de los mayores desafíos ha sido la extracción de información útil a partir de conjuntos grandes de documentos de texto. La literatura existente sobre métodos para resolver este problema es muy extensa, sin embargo, los métodos estadísticos (que utilizan métricas de similitud sobre vectores de palabras) han mostrado resultados muy favorables en el campo de la minería de texto durante los últimos 25 años. Adicionalmente, otros modelos han surgido como una prometedora alternativa para lograr reducción dimensional e incorporación de la semántica en la clasificación de documentos, tal como el modelado de temas. Este proyecto se enfoca en la evaluación de técnicas de representación y medidas de similitud de texto (Coseno, Jaccard y Kullback-Leibler) usando el algoritmo de Vecinos más Cercanos (KNN por sus siglas en inglés), con el fin de medir la efectividad del modelado de temas para reducción dimensional al clasificar texto. Los resultados muestran que la versión más tradicional del vector de palabras y la similitud Jaccard superaron al resto de las combinaciones en la mayoría de los casos de uso. Sin embargo, el análisis estadístico mostró que no hubo una diferencia significativa entre la exactitud obtenida al usar representaciones generadas por la Asignación de Dirichlet Latente (técnica de modelado de temas más conocida como LDA por sus siglas en inglés), y la obtenida usando técnicas tradicionales de clasificación de texto. LDA logró abstraer miles de palabras en menos de 60 temas para el primer conjunto de pruebas. Experimentos adicionales sugieren que el modelado de temas puede llegar a lograr un mejor rendimiento al ser usado para clasificar textos cortos y al incrementar el número de temas permitidos al momento de generar el modelo.

Introduction

Nowadays, large volumes of text information are available on the Internet and in institutional databases with a clear trend of continuous growing, hence applying data mining and machine learning techniques over text data is becoming extremely relevant [1]. One of the biggest challenges in this area has been the ability to extract meaningful information from large sets of text.

For example, consider those systems used in most of the companies to track application failures reported by users. Every time a user reports an issue, the support person receives certain information, including a free text description of the problem observed by the user. Once the support specialist takes action and resolves the issue, the system generally allows to take notes about the investigation done, the explanation of what caused the failure and select the root cause category from a dropdown. Suppose that someone wants to analyze the roots causes of issues received in the last year, but it is found that, in many cases, the category was not selected when the problem was resolved, causing the report to have missing values in multiple records. In this situation, the root cause categories could be estimated by analyzing the free text in the respective descriptions and notes taken for the issue. Even though this task could be done by people, when there are thousands of records in the database then there is a need of evaluating automated solutions where a program is able to find those relations between the text and the missing category.

Under this context, this paper analyses the problem of text classification, where an algorithm classifies the observed text values within different categories. Existing literature proposes different machine learning and data mining algorithms to solve classification problems in a supervised way [2, 3, 4]. Algorithms such as K-Nearest-Neighbors (KNN), where text distance metrics can be used to classify elements, are relevant for documents classification problems [3, 5].

Current literature is very extensive about metrics to determine how similar a text document is with respect to another. Statistical methods intend to create mathematical data representations of text without considering semantic nor linguistic properties and have shown very good results in the last 25 years in text mining areas [1]. Statistic methods include the representation of documents through bags of words and the use of distance metrics such as Cosine and Jaccard [3, 6].

Additionally, topic modelling has arisen as a promising alternative to the existing methods, by achieving good results in documents classification using probability distributions similarity, dimensional reduction and incorporating topics semantic [7, 8, 9]. Latent Dirichlet Allocation (LDA) is one of the most popular techniques in this area, using both unsupervised and supervised learning [10].

The goal of this investigation is to evaluate the use of methods involving topic modelling and probability distributions comparison such as LDA and Kullback-Leibler divergence (KLD, also known as relative entropy), against traditional data representation techniques and text distance metrics (bag of words, Cosine, Jaccard). For that purpose, we analyze the results obtained after running several experiments using the techniques already mentioned to classify datasets of text documents. The results show that the accuracy scores achieved by using document representations obtained by LDA, combined with the relative entropy metric, were always outperformed by the ones obtained by using traditional text classification techniques. However, differences in the results are not statistically significant as shown in a posterior statistical test. The topics modeling managed to abstract thousands of words in less than 60 topics for the main set of experiments. An additional analysis highlights cons, improvement areas and scenarios where such models could potentially achieve a better performance.

The rest of the document is organized as follows. Section 2 summarizes the previous work done around text classification. In section 3 we describe the models theoretically. The details about the experiments are shown in section 4 and the obtained results are analyzed in section 5. Finally, in sections 6 and 7 we propose the future work and present the conclusions.

Related work

There has been a lot of work around algorithms to solve the problem of text classification in the areas of machine learning and data mining. This paper was focused on those methods aimed to find relationships among text and categorical attributes. In this section we summarize the work done by different authors about these topics.

In [2, 4] several algorithms of supervised classification are considered, such as Hotdeck, KNN, and Decision Trees. These methods require a labeled training set of data and they use similarity metrics to define the relations among the elements to classify. Other popular techniques include mathematical models to approximate values and reduce errors based on training data, for example, Artificial Neural Networks and Support Vector Machines [4]. From all these algorithms, we considered the first group for this work since they were the best match for our problem by allowing the definition of similarity metrics.

When studying the classification of text documents, current literature is focused on two types of methods: linguistic and statistical [1]. In linguistic methods, text is handled based on the processing of natural language, considering semantic representations and relationships of words within a linguistic model [1]. Different authors have incorporated semantics when comparing text documents, requiring the integration of external sources of knowledge and lexical data bases such as Wikipedia or WordNet [11, 12]. In such methods, words similarity is determined by the degree of overlapping in their meanings or by the distance between two terms if represented using a graph of hypernyms [12]. Even though linguistic methods can achieve more expressive representations of text, they can also add more complexity due to the construction of semantic models and context-specific dependency in the data [1]. In this work we use statistical methods which involve the mathematical representation of text without considering semantics or linguistic properties [1]. The general process in statistic methods consist on using some technique to create a representation of documents and then to apply a function on these representations to calculate how similar are to each other.

The bag of words is the data representation technique used in most of the consulted literature [13, 6, 14, 15, 1, 16]. It consists on representing each text document as a vector of frequencies [13]. A variant of this representation uses Term Frequency-Inverse Document Frequency (TF-IDF) weighting [16, 6] where each word in a document is assigned a weight depending on their frequencies within a specific document and throughout all the documents.

The topic modelling technique known as LDA is used in [9, 7] and considers two main concepts: 1) a single document can have several latent topics and 2) each topic can be drawn as a probability distribution of words (documents are represented as vectors of topics instead of bags of words). A supervised variant of LDA (sLDA) is proposed in [10], which incorporates a response variable (or class) when calculating the model of topics. That work uses sLDA to predict movies and web sites ratings based on the text of user reviews.

Regarding text distance metrics the literature is very extensive, being the Cosine similarity one of the most used techniques, like in [17, 6]. Authors in [7] combine LDA representation with the Cosine function to calculate similarity among publications using the text of the title, abstract and author names. Other metrics used for both classification and grouping of text documents include simpler vector functions (Manhattan and Euclidean), set-based metrics (Jaccard) and entropy measure in probability distributions (KLD) [17, 6, 3].

Other techniques for text classification that have gained popularity and use the concept of word vectors representation, are fasttext [18] and GloVe [19]. These algorithms have different ways to create vectors of words in an efficient way from a big corpus, and then use those vectors to obtain the nearest neighbors of a single word. However, since fasttext and GloVe are complete classification algorithms by themselves and implementations do not have the flexibility to configure distance metrics, they were not considered as part of the experiments on this paper.

Recent studies have evaluated text classification techniques mostly focused on classification algorithms [4] and distance measures [3]. There have also been works related to the comparison of preprocessing methods and document representations [15, 14]. This work intends to achieve a similar evaluation but combining the representations with the distance metrics.

Model descriptions

KNN classification algorithm

In this paper, we use KNN because it is a supervised classifier and can be configured to use text distance metrics as explained in a later section. KNN is an algorithm that selects the nearest k observations to a certain value according to a distance metric. Even though it is categorized as a machine learning algorithm, the learning process in KNN simply consists on storing all the training data and comparing against it at the arrival of test data [4].

Algorithm 1 shows the general idea of KNN used for text classification. The algorithm receives a training dataset E and a test dataset X . The *FindNeighbors* function returns a list of k rows taken from E where the values of the textual attributes are the nearest ones respect to the values of the same attributes in X according to a distance function. The function *GetCategoryByVoting* obtains the most frequent value in a categorical attribute from the rows stored in *neighbors*.

Documents representation using bag of words

Let D be a dataset composed by m text documents where there is a total of n different words. The main idea of a bag of words is to represent each document as a vector $d = p_1, p_2, \dots, p_n$, where the element p_i corresponds to the frequency of the i -th word in that document [14]. In this case, a set of documents can be seen as the matrix in equation 1, being each p_{ij} the value of the frequency of term j in document i .

Algorithm 1. KNN algorithm for text classification

Input: $k; E; X$

Let E be a set of text documents labeled with a categorical attribute (training dataset);

Let X be a set of text documents with no category associated; **foreach** row x in X **do**

$neighbors \leftarrow \text{FindNeighbors}(k, x, E)$;

$category \leftarrow \text{GetcategoryByVoting}(neighbors)$;

$x[category] \leftarrow category$;

$$D = \begin{pmatrix} P_{11} & \dots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{m1} & \dots & P_{mn} \end{pmatrix} \quad (1)$$

The vector representation of a document can also include a weight value instead of the frequency value. The technique Term Frequency - Inverse Document Frequency (TFIDF) consists on calculating a weight for each word, combining the frequency of a term in a document and the number of documents containing such term respect to the total of documents [1]. TF-IDF intends to highlight those terms that represent better specific documents and to lower the weight to irrelevant terms [1].

Other techniques commonly applied at the moment of creating bags of words include the removal of stop words (non-relevant terms such as articles or pronouns), and word stemming (keeping only the root of words in order to create semantic grouping) [1, 14].

Text distance metrics

In this paper we focused on evaluating term-based distance metrics, on which a text string is divided in terms, like in a bag of words, and this representation is used to do comparisons among vectors [3]. A classification algorithm can use these metrics to calculate the distance between two elements.

Among the most common metrics used for words vector comparison are the Cosine and Jaccard similarities [6, 3]. However, there are some approaches based on probabilistic concepts that have also been used for text documents comparison, such as KLD [7, 3]. All these methods are explained in the below sections.

Cosine distance

This distance measures the degree of similarity between two documents d_1 and d_2 using the cosine of the angle formed by their vector representations [6], as shown in equation 2.

$$\cos(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 \cdot \vec{d}_2}{\|\vec{d}_1\| \|\vec{d}_2\|} \quad (2)$$

Jaccard similarity

Having two bags of words d_1 and d_2 as two sets of elements (without considering terms frequency), the Jaccard similarity is defined in equation 3 as the size of the intersection of d_1 and d_2 divided by the size of the union of the same sets [13].

$$jaccard(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 \cap \vec{d}_2}{\vec{d}_1 \cup \vec{d}_2} \quad (3)$$

Kullback-Leibler divergence

Considering a document as a probability distribution of words represented in a vector, we can use KLD to measure the level of entropy between two probability distributions [3]. So if n is the total of words in a collection of documents, two documents d_1 and d_2 can be represented as probability distributions $\vec{d}_i = p_{i,1}, p_{i,2}, \dots, p_{i,n}$ where the value of $p_{i,j}$ represents the probability of word j belonging to document d_i for $i \in \{1,2\}$ and $1 \leq j \leq n$, then KLD is calculated as shown in equation 4.

$$D_{KL}(\vec{d}_1 \parallel \vec{d}_2) = \sum_{j=1}^n p_{1,j} \times \log \left(\frac{p_{1,j}}{p_{2,j}} \right) \quad (4)$$

KLD has been used in documents clustering and it is not symmetric, hence it should be combined with other methods to get a single value, such a weighted average [3].

Topics modelling

Topic modelling has arisen as a robust technique to structure collections of documents by using probabilistic models to find hidden semantic patterns [1]. It represents an alternative method to achieve dimensional reduction and it has several advantages compared to other models of semantic analysis [13].

This work used the documents representations with topics obtained through LDA, which is explained in the next section.

Latent Dirichlet Allocation

LDA is a probabilistic model used to classify documents in topics considering two aspects: 1) the same document can have several latent topics and 2) each topic can be represented by a distribution of words [7]. In this case, a documents will be assigned to probabilities of latent topics where, at the same time, each topic is a probability distribution of words.

The idea of LDA is to achieve dimensional reduction (compared to traditional representations using bag of words) and easily assign probabilities to new documents that were not part of the training dataset [13].

Even though LDA was originally created to discover latent topics in an unsupervised way [20], this paper explores the supervised variant proposed in [10]. In supervised LDA, if we have K topics, $\beta_{1:K}$ (where each β_k is a vector of probabilities distribution per topic), a Dirichlet parameter α , and parameters η and σ^2 of the response variable [10]. It is assumed that each text document is generated as follows:

1. Get topic proportions $\theta | \alpha \sim Dir(\alpha)$.
2. For each word:
 - Assign a topic $z_n | \theta \sim Mult(\theta)$.
 - Get a word $w_n | z_n, \beta_{1:k} \sim Mult(\beta_{z_n})$.
3. Get response variable (class) $y | z_{1:N}, \eta, \sigma^2 \sim N(\eta^T z, \sigma^2)$.

This generative model can be observed in figure 1. Each node denotes a random variable, while edges represent dependencies. Boxes denote repetition for D documents, K topics and N words [1].

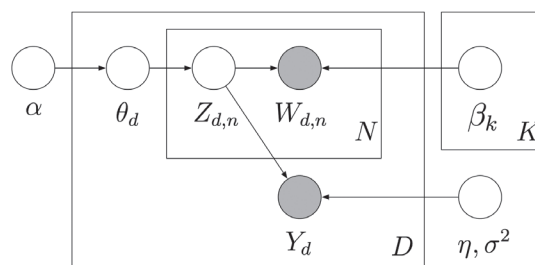


Figure 1. Graphical representation of supervised LDA [10].

In LDA, the final representation of a document d_i is not a bag of words, but a vector of k latent topics $\vec{d}_i = p_{i,1}, p_{i,2}, \dots, p_{i,k}$ where the value of $p_{i,j}$ represents the probability that d_i belongs to topic j for $1 \leq j \leq n$ [7, 13]. Subsequently, the similarity between two probability distributions can

be calculated using a similarity metric. This paper was focused on the vector representation of documents only and not the complete LDA algorithm nor its prediction model.

Experimental setup

Development environment

Experiments were executed on a virtual server with two processors of 2GHz each and 4GB of RAM running Ubuntu 16.04 distribution.

Documents representation as vectors of words were done using package RTextTools 1.4.2 in R version 3.2.3, while the main algorithms (KNN and text distance metrics) were implemented in Python version 3.5.2 using scikit-learn 0.18 libraries.

For supervised LDA, we used the C++ implementation based on the work of [10] and published in the author's website ².

Datasets

The data used in this experiments consisted on four datasets. Two of them extracted from the Reuters collection for text classification obtained from the UCI public repository ³, the WebKB ⁴ and the Enron Email ⁵ datasets from the Carnegie Mellon University repository.

To simplify the execution of experiments, we created two datasets by transforming and compiling a subset of the XML files taken from Reuters collection, reuters-1 and reuters2. We also transformed the other two datasets according to the format required by the implemented algorithms. All the final datasets contain an attribute of type text and a categorical attribute. The content of each dataset is summarized below:

- **reuters-1**. Each category value can be one of the 44 topics defined in the original Reuters dataset.
- **reuters-2**. In this case, each category value can be one of the 59 places (country codes) defined in the original Reuters dataset.
- **enron-email**. This dataset consists on documents representing email contents, each categorized as 'YES' (span email) or 'No' (not span email).
- **webkb**. Each document contains the text of web pages of various universities, manually categorized into 4 classes: student, faculty, course or project.

Implementation of data representation techniques

The bags of words were generated from the original text documents using RTextTools package. We also applied removal of stop words and words stemming. Each file was transformed into two representations, one containing term frequencies and another using TF-IDF representation. The resulting matrices per document are summarized below:

- **reuters-1**: 1063 documents x 6444 words
- **reuters-2**: 1641 documents x 8666 words
- **enron-email**: 3657 documents x 6001 words
- **webkb**: 4199 documents x 7678 words

2 <https://github.com/blei-lab/class-slda>

3 <http://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection>

4 <http://www.cs.cmu.edu/TextLearning/datasets.html>

5 <https://www.cs.cmu.edu/~enron/>

To generate the vectors of topics, we used the C++ implementation of supervised LDA. The code was modified to allow us the extraction of the topic models representations. Even though the complete LDA algorithm does both estimation and inference operations, for this work we just needed the vectors of topics representations generated in the estimation step. The implementation didn't allow to calculate the vector of topics for the webkb dataset because of memory errors when processing more than 4000 documents. The resulted LDA representations consisted on the following matrices:

- **reuters-1**: 1063 documents x 44 topics
- **reuters-2**: 1641 documents x 59 topics
- **enron-email**: 3657 documents x 100 topics

At this point we also partitioned the data representations in a set of 25% test data and a set of 75% training data. This allowed to execute all the experiments under the same conditions (same training and test data for all the experiments).

Implementation of KNN and the text distance metrics

KNN algorithm was implemented using the module KNeighborsClassifier from sklearn Python package. This module can be configured with customized similarity functions, which allowed us to define and incorporate the text distance metrics desired. We used Jaccard and Cosine implementations included within the sklearn package and for KLD we used the entropy metric from scipy package.

Parameters selection

The main parameter for KNN is the value of k which determines the number of neighbors to consider when labeling a document. To select this parameter, we executed a subset of the experiments using multiple k values and calculated the accuracy obtained on each case. Figures 2 and 3 show the accuracy trend graphs obtained by running experiments using cosine distance and word vectors for reuters-1 and reuters-2 datasets respectively. We observed that the accuracy tended to decrease as we increased the value of k . For that reason we used $k = 1$ and $k = 11$ to run the full set of experiments and analyze the results.

In the case of supervised LDA, we fixed the number of iterations and the error convergence thresholds with the library default values to simplify the selection and avoid performance issues (execution time increased considerably when slightly increasing the default values). Given the variety of text documents, we considered using three different values of α : 0.2 (assumes text documents very different from each other and with few topics), 0.8 (assumes similar text documents with many topics each) and an intermediate value of 0.5. And finally, the number of topics was defined based on each dataset, assuming that the number of topics can be similar to the number of different classes in the Reuters documents (44 topics for reuters-1 and 59 topics for reuters-2). For the enron-email dataset the number of topics was set to 100, considering that the number of classes was too short and more than 100 topics caused the algorithm implementation to throw memory errors.

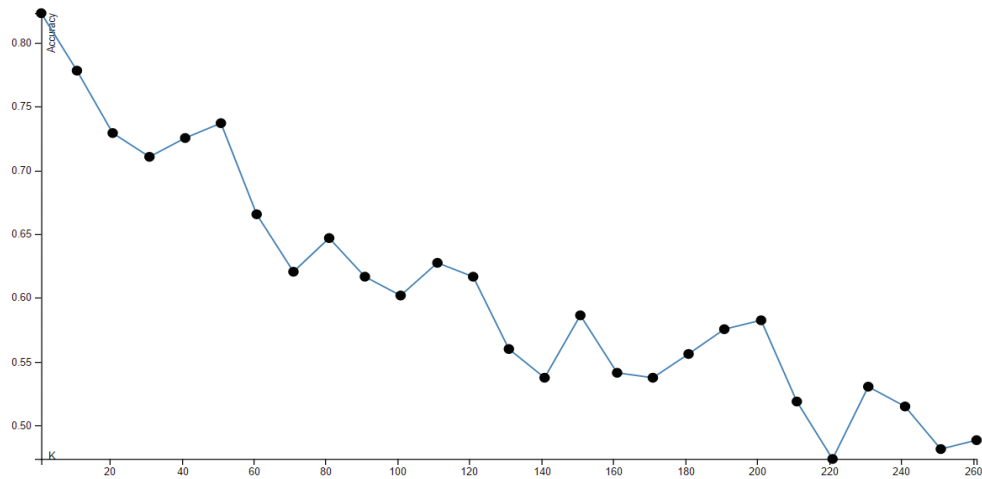


Figure 2. Accuracy trend for different values of k using KNN, word vectors and cosine similarity with reuters-1 dataset.

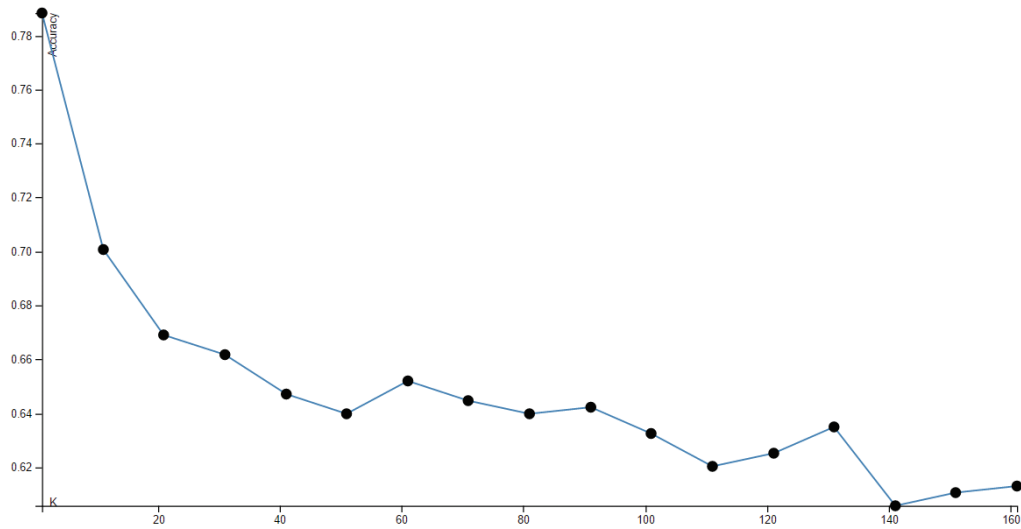


Figure 3. Accuracy trend for different values of k using KNN, word vectors and cosine similarity with reuters-2 dataset.

Results and analysis

The results are shown in table 1, which contains the accuracy, precision, recall and F1 score obtained on each experiment for the four datasets. Each experiment involved a data representation technique, a text distance metric and the KNN algorithm. Data represented as bags of words was divided in two groups: with TF-IDF and without TF-IDF, while LDA representations were divided in three groups based on the different values of α . All the scores in the table were calculated using the macro-weighted metrics defined in the sklearn package. In this work we focused on the accuracy and the F1 score for the analysis.

In terms of datasets, the experiments show better results when classifying data in reuters1 compared to the results obtained for the other datasets. Classification in reuters-2 represented a more complex problem, since each text had to be classified into one geographical location out of 59, while for reuters-1 the classification was done using 44 categories labeled from the original datasets. In the case of the webkb and enron-email datasets, which obtained the lowest overall results, we can assume it is expected considering that KNN implies over-fitting by definition. This means that the accuracy of the classification could decrease as we use bigger datasets, causing the model to fail at finding patterns to generalize when new data is presented.

Table 1. Results for each experiment per dataset, value of k and representation-metric combination. The best scores are highlighted for each set of experiments using the same dataset and value of k .

Dataset	k	Representation/Metric	Accuracy	Precision	Recall	f1
enron-email	1	LDA $\alpha=0.2$ / Cosine	0.616648412	0.618932375	0.60757046	0.602711829
		LDA $\alpha=0.5$ / Cosine	0.659364732	0.659520842	0.654084552	0.65377576
		LDA $\alpha=0.8$ / Cosine	0.641840088	0.640540355	0.63768807	0.637665435
		LDA $\alpha=0.2$ / KLD	0.592552026	0.591153931	0.591200466	0.591174346
		LDA $\alpha=0.5$ / KLD	0.644030668	0.647032856	0.646773681	0.644015294
		LDA $\alpha=0.8$ / KLD	0.647316539	0.646753272	0.647223988	0.646736361
		Words Vector w TF-IDF / Cosine	0.644030668	0.64363091	0.644124815	0.643536322
		Words Vector w/o TF-IDF / Cosine	0.663745893	0.662953569	0.663382072	0.663032802
		Words Vector w/o TF-IDF / Jaccard	0.69550931	0.695412448	0.69612206	0.695201795
	11	LDA $\alpha=0.2$ / Cosine	0.605695509	0.611624313	0.593928798	0.583447395
		LDA $\alpha=0.5$ / Cosine	0.680175246	0.682739781	0.673977538	0.673452489
		LDA $\alpha=0.8$ / Cosine	0.669222344	0.670977838	0.663117186	0.662484332
		LDA $\alpha=0.2$ / KLD	0.635268346	0.636287523	0.62817864	0.626226787
		LDA $\alpha=0.5$ / KLD	0.69441402	0.694961504	0.695618775	0.69426731
		LDA $\alpha=0.8$ / KLD	0.700985761	0.700231048	0.700757576	0.700351645
		Words Vector w TF-IDF / Cosine	0.665936473	0.672142885	0.670216147	0.665621979
		Words Vector w/o TF-IDF / Cosine	0.706462212	0.706933045	0.707644628	0.706306834
		Words Vector w/o TF-IDF / Jaccard	0.705366922	0.707697529	0.70780356	0.705365508
reuters-1	1	LDA $\alpha=0.2$ / Cosine	0.8	0.804452	0.8	0.792715
		LDA $\alpha=0.5$ / Cosine	0.8	0.821869	0.8	0.803953
		LDA $\alpha=0.8$ / Cosine	0.816	0.815385	0.816	0.811214
		LDA $\alpha=0.2$ / KLD	0.8	0.797469	0.8	0.789491
		LDA $\alpha=0.5$ / KLD	0.768	0.776959	0.768	0.759982
		LDA $\alpha=0.8$ / KLD	0.8	0.797469	0.8	0.789491
		Words Vector w TF-IDF / Cosine	0.808	0.834513	0.808	0.808946
		Words Vector w/o TF-IDF / Cosine	0.856	0.860481	0.856	0.852386
		Words Vector w/o TF-IDF / Jaccard	0.9	0.900628	0.9	0.893534
	11	LDA $\alpha=0.2$ / Cosine	0.796	0.779623	0.796	0.780964
		LDA $\alpha=0.5$ / Cosine	0.788	0.754407	0.788	0.762225
		LDA $\alpha=0.8$ / Cosine	0.78	0.758448	0.78	0.757489
		LDA $\alpha=0.2$ / KLD	0.828	0.796299	0.828	0.808493
		LDA $\alpha=0.5$ / KLD	0.824	0.797579	0.824	0.798982
		LDA $\alpha=0.8$ / KLD	0.804	0.760077	0.804	0.773005

Continúa...

Continuación

		Words Vector w TF-IDF / Cosine	0.816	0.803179	0.816	0.797044
		Words Vector w/o TF-IDF / Cosine	0.844	0.807863	0.844	0.821831
		Words Vector w/o TF-IDF / Jaccard	0.868	0.836532	0.868	0.845142
reuters-2	1	LDA $\alpha=0.2$ / Cosine	0.70437	0.719479	0.70437	0.707669
		LDA $\alpha=0.5$ / Cosine	0.714653	0.750129	0.714653	0.730143
		LDA $\alpha=0.8$ / Cosine	0.719794	0.748855	0.719794	0.729776
		LDA $\alpha=0.2$ / KLD	0.709512	0.741322	0.709512	0.716883
		LDA $\alpha=0.5$ / KLD	0.748072	0.765898	0.748072	0.751746
		LDA $\alpha=0.8$ / KLD	0.750643	0.779055	0.750643	0.76193
	11	Words Vector w TF-IDF / Cosine	0.830334	0.847533	0.830334	0.835109
		Words Vector w/o TF-IDF / Cosine	0.812339	0.822891	0.812339	0.80835
		Words Vector w/o TF-IDF / Jaccard	0.845758	0.851068	0.845758	0.845541
		LDA $\alpha=0.2$ / Cosine	0.74036	0.638793	0.74036	0.682275
		LDA $\alpha=0.5$ / Cosine	0.701799	0.629419	0.701799	0.662099
		LDA $\alpha=0.8$ / Cosine	0.706941	0.646742	0.706941	0.664348
		LDA $\alpha=0.2$ / KLD	0.745501	0.67593	0.745501	0.700212
		LDA $\alpha=0.5$ / KLD	0.74036	0.665861	0.74036	0.68856
		LDA $\alpha=0.8$ / KLD	0.755784	0.705765	0.755784	0.716822
		Words Vector w TF-IDF / Cosine	0.820051	0.791033	0.820051	0.787891
		Words Vector w/o TF-IDF / Cosine	0.760925	0.688958	0.760925	0.713346
		Words Vector w/o TF-IDF / Jaccard	0.807198	0.775127	0.807198	0.77375
webkb	1	Words Vector w TF-IDF / Cosine	0.695781343	0.647058199	0.642711206	0.64227423
		Words Vector w/o TF-IDF / Cosine	0.720686368	0.679451312	0.68245294	0.677963291
		Words Vector w/o TF-IDF / Jaccard	0.786463298	0.778443989	0.752974877	0.757970349
	11	Words Vector w TF-IDF / Cosine	0.757864633	0.736808453	0.701032144	0.708486955
		Words Vector w/o TF-IDF / Cosine	0.758817922	0.765218216	0.700841447	0.70733323
		Words Vector w/o TF-IDF / Jaccard	0.832221163	0.859234082	0.766482197	0.779513748

For all the scenarios, the simplest method of bag of words (based on terms frequencies only) achieved better accuracy and F1 score than the rest of the data representation techniques. Jaccard similarity, which could also be considered the simplest function used on these experiments, represented the better metric to be combined with bag of words without TF-IDF in most of the cases. For instance, for $k=1$, this combination achieved the highest accuracy of 0.9 in reuters-1.

Conventional methods outperformed supervised LDA in terms of accuracy and F1 score in all the experiments. However, a posterior statistical analysis threw that the accuracy and F1 scores in table 1 do not show significant differences regardless of the representation-metric combination used (see results in table 2). For this statistical test, we applied a Lilliefors (Kolmogorov-Smirnov)

normality test on the accuracy and the F1 values using the *nortest* package in R. The accuracy values did not follow a normal distribution but the F1 score did, and for that reason we applied different statistical tests (Kruskal-Wallis and Anova) to each variable. Considering a significance level of 0.05 both tests show no significant differences among the means.

LDA achieved a huge dimensional reduction as a text data representation method, and the accuracy and F1 scores results are not significantly different to the ones obtained when using bag of words. From this perspective, the use of vectors of topics for text classification seems to have a big potential.

In the LDA experiments using the reuters-1 dataset, we also observed that the average size of text documents was 635 bytes for those correctly classified and 963 bytes for those

Table 2. Results of the statistical tests applied on the accuracy and F1 scores using a significance level of 0.05

Quantitative variable	accuracy	F1 score
Qualitative variable	Representation/Metric	Representation/Metric
Statistical test	Kruskal-Wallis	Anova
Result	chi-squared = 7.9864, df = 8, p-value = 0.4348	Df=8 Sum Sq=0.05013 Mean sq=0.006266 F value=1.165 Pr(>F)=0.341
Hypothesis	Kruskal-Wallis test shows no significant differences among the means (p >0.05)	Anova test shows no significant differences among the means (p >0.05)

Table 3. Results obtained after executing experiments using short text documents extracted from reuters-1

Metric/Representation	Accuracy	F1
Cosine/LDA $\alpha=0.8$	0.818181818	0.804746654
Cosine/Bag of words without TF-IDF	0.787878788	0.769054178
Jaccard/Bag of words without TF-IDF	0.787878788	0.777777778
KLD/LDA $\alpha=0.5$	0.787878788	0.774104683
Cosine/LDA $\alpha=0.5$	0.787878788	0.759322717
KLD/LDA $\alpha=0.8$	0.757575758	0.728760473
KLD/LDA $\alpha=0.2$	0.727272727	0.683261183
Cosine/LDA $\alpha=0.2$	0.727272727	0.683261183
Cosine/Bag of words with TF-IDF	0.696969696	0.624756885

where the categorization was wrong. 50% of the documents incorrectly classified had a size 900 bytes or more. Based on this, we created an additional dataset consisting of 129 training documents and 34 test documents, whose size were less than 150 bytes, taken from reuters-1.

Results in table 3 show how LDA, using $\alpha = 0.8$ and combined with Cosine achieved the best scores.

We also explored the optimization of the LDA model (vectors of topics) for the reuters-1 and reuters-2 datasets by trying different values for the parameter *number of topics*, which originally was set as the number of different categories in each dataset. We generated two additional LDA models, using 100 and 200 topics respectively. Then we executed experiments using these models, $k = 1$, α values of 0.2 and 0.8, and both Cosine and KLD metrics. As shown in the graph of figure 4, the results of the experiments applied to LDA models using $\alpha = 0.2$ seem to improve in terms of accuracy and F1 as the number of topics is increased. This trend is more evident when using KLD (it doesn't happen for Cosine). The result suggests that by increasing the number of dimensions in the vectors of topics, LDA and KLD can obtain better results as long as the number of topics per document is low (α value near to zero in the Dirichlet function). It makes sense, given that the level of abstraction (hence, the complexity) is reduced when increasing the number of dimensions, and a low α value means documents represented by a few topics. This approach looks promising, since the number of topics is still much smaller than the dimensions of the bags of words, however, the time complexity of adding more topics is big. Execution time measures were not part of the scope of this work but we consider important to mention that the time to generate a model using 200 and 100 topics was more than eight times greater than the required time to generate the bags of words models.

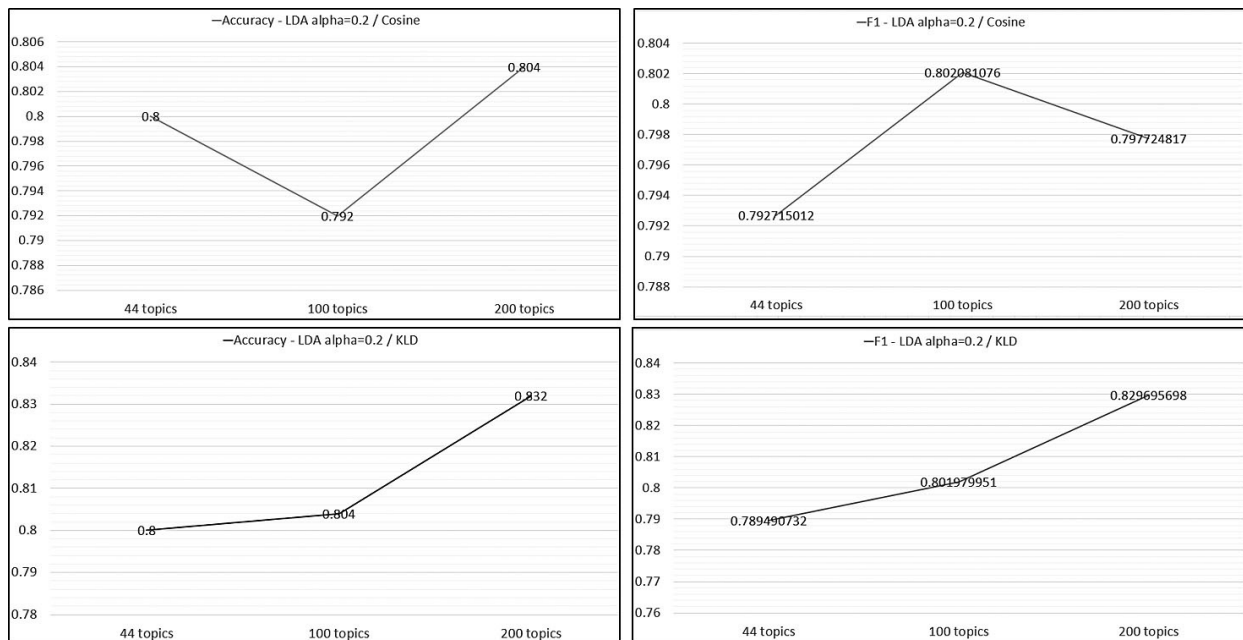


Figure 4. Trend plots for accuracy and F1 scores obtained by using LDA with $\alpha = 0.2$, $k = 1$ and metrics Cosine and KLD for different number of topics

Future work

This work was focused on the analysis of different algorithms intended to work on any text documents dataset. However, as observed in the results of this investigation and other similar works, the solution is strongly dependent on the specific context of the datasets. Based on this, it

could be useful to explore the incorporation of more datasets from different sources and analyze the results based on the specific characteristics of each one. A big contribution could consist on applying the studied algorithms to enterprise databases allowing a better understanding of practical applications.

Given the importance of data preprocessing on this work, something interesting to evaluate are the different implementations of a same data representation technique. For example, the evaluation of the behavior and the performance of using R versus Python when removing stop words, doing words stemming or TF-IDF weighting.

The scope of this investigation was limited to statistical techniques mostly based on lexical analysis of text. A potential good contribution to the experiments could be the incorporation of semantic analysis techniques [11], in order to compare the results to the ones obtained in this work.

KNN facilitated the design of experiments because of its simplicity and flexibility to combine data representations and text distance metrics. A future work can consider the use of other supervised classification algorithms and ensemble methods to study their behavior and results respect to KNN.

Regarding LDA, there were many aspects left out of the scope of this work but that could represent valuable contributions to the investigation in the future. Firstly, we only used the documents representation generated as part of the model but we didn't apply the complete supervised LDA algorithm. The study and implementation of this algorithm, which doesn't need KNN or text distance metrics, could improve the results by taking advantage of the probabilistic properties of the Dirichlet function and its optimization methods. Second, we can dive deep in optimizing many of the parameters required by LDA to generate the models of topics, looking for a balance between the accuracy of the models and the time it takes to build it. And the third aspect to consider, is the optimization of the existing supervised LDA implementations or to explore alternative topic modelling techniques, looking to improve execution time and lower the complexity.

The execution time measurement was not in the scope of this work, but this variable could add an important value to the results, by allowing the identification of pros and cons (in terms of performance and efficiency) when using a text distance metric respect to the others.

Conclusion

In this paper we studied different text distance metrics applied to documents classification. These metrics involved text documents representation methods and mathematical functions applied to those representations. We evaluated the accuracy and the F1 score obtained after classifying values using KNN algorithm.

The results evidence how the quality of the datasets affected the accuracy and the F1 obtained in all the experiments. For instance, the four datasets were very different in terms of number of classes, number of records and semantic.

LDA achieves an important dimensional reduction as text representation technique, and the experimental results are statistically comparable to the ones achieved by using bag of words, even though they were always outperformed by the rest of the studied methods. For some datasets, the number of dimensions in the vectors of topics was up to 150 times smaller than the vectors of words, abstracting thousands of words in less than 60 topics. Some additional

experiments suggest that LDA combined with KLD can perform better when used for short text documents, increasing the parameter of number of topics when generating the model and keeping a low value of α .

Acknowledgements

The authors would like to thank the *Maestría en Computación* program at *Instituto Tecnológico de Costa Rica* for providing the occasion for this research.

References

- [1] A. N. Srivastava, M. Sahami, Text mining: Classification, clustering, and applications, CRC Press, 2009.
- [2] C. T. Tran, M. Zhang, P. Andrae, Multiple imputation for missing data using genetic programming, in: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, ACM, 2015, pp. 583–590.
- [3] M. Kocher, J. Savoy, Distance measures in author profiling, *Information Processing & Management* 53 (5) (2017) 1103–1119.
- [4] V. K. Vijayan, K. R. Bindu, L. Parameswaran, A comprehensive study of text classification algorithms, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 1109–1113. doi:10.1109/ICACCI.2017.8125990.
- [5] Y. Zhao, Y. Qian, C. Li, Improved knn text classification algorithm with mapreduce implementation, in: Conference: Conference: 2017 4th International Conference on Systems and Informatics (ICSAI), 2017, pp. 1417–1422.
- [6] A. J. Soto, A. Mohammad, A. Albert, A. Islam, E. Milios, M. Doyle, R. Minghim, M. C. Ferreira de Oliveira, Similarity-based support for text reuse in technical writing, in: Proceedings of the 2015 ACM Symposium on Document Engineering, ACM, 2015, pp. 97–106.
- [7] D.-H. Bae, S.-H. Yoon, T.-H. Eom, J. Ha, Y.-S. Hwang, S.-W. Kim, Computing paper similarity based on latent dirichlet allocation, in: Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication, ACM, 2014, p. 77.
- [8] K. Bougiatiotis, T. Giannakopoulos, Content representation and similarity of movies based on topic extraction from subtitles, in: Proceedings of the 9th Hellenic Conference on Artificial Intelligence, ACM, 2016, p. 17.
- [9] M. Pavlinek, V. Podgorelec, Text classification method based on self-training and lda topic models, *Expert Systems with Applications* 80 (2017) 83–93.
- [10] J. D. McAuliffe, D. M. Blei, Supervised topic models, in: Advances in neural information processing systems, 2008, pp. 121–128.
- [11] S. Seifzadeh, A. K. Farahat, M. S. Kamel, F. Karray, Short-text clustering using statistical semantics, in: Proceedings of the 24th International Conference on World Wide Web, ACM, 2015, pp. 805–810.
- [12] N. Devraj, M. Chary, How do twitter, wikipedia, and harrison's principles of medicine describe heart attacks?, in: Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics, ACM, 2015, pp. 610–614.
- [13] C. C. Aggarwal, Data mining: the textbook, Springer, 2015.
- [14] A. K. Uysal, S. Gunal, The impact of preprocessing on text classification, *Information Processing & Management* 50 (1) (2014) 104–112.
- [15] H. K. Kim, H. Kim, S. Cho, Bag-of-concepts: Comprehending document representation through clustering words in distributed representation, *Neurocomputing* 266 (2017) 336–352.
- [16] A. Onan, S. Korukoğlu, H. Bulut, Ensemble of keyword extraction methods and classifiers in text classification, *Expert Systems with Applications* 57 (2016) 232–247.
- [17] N. Liebman, D. Gergle, Capturing turn-by-turn lexical similarity in text-based communication, in: Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, ACM, 2016, pp. 553–559.
- [18] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of tricks for efficient text classification, arXiv preprint arXiv:1607.01759.
- [19] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.
- [20] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, *Journal of machine Learning research* 3 (Jan) (2003) 993–1022.