# On the Importance of Data Representation for the Success of Text Classification

Carolina Y. Cuello [1,2], Vanessa Jofre Caradonna [2],
Ma. José Garciarena Ucelay [1,2], and Leticia C. Cagnina [1,2,3]

[1] LIDIC, Universidad Nacional de San Luis, San Luis, Argentina,
[2] Universidad Nacional de San Luis, San Luis, Argentina,
[3] Consejo Nacional de Investigaciones Científicas y Técnica (CONICET), Argentina
{carolina.yamile.cuello,vane.jofre.caradonna,
mjgarciarenaucelay,lcagnina}@gmail.com

**Abstract.** Text mining approaches use natural language processing to automatically extract patterns from texts. Tasks as topic labeling, news classification, question answering, named entity recognition and sentiment analysis, usually require elaborate and effective document representations. In this context, word representation models in general, and vector-based word representations in particular, have gained increasing interest to alleviate some of the limitations that Bag of Words exhibits. In this article, we analyze the use of several vector-based word representations besides the classical ones, in a polarity analysis task on movie reviews. Experimental results show the effectiveness of more elaborate representations in comparison to Bag of Words. In particular, Concise Semantic Analysis representation seems to be very robust and effective because independently the classifier used with, the results are really good. Dimension and time of getting the representations are also showed, concluding in the efficiency of the classifiers when Concise Semantic Analysis is considered.

**Keywords:** text mining, text representations, text classification, movie reviews, sentiment analysis, polarity analysis

## 1 Introduction

Since the creation of the World Wide Web in 1989, the history of the communication has being evolved year by year. In 1990, with few computers connected around the world, the first web browser was presented. By the early year 2000, only some countries had access to the information through Internet. But the use of this technology grew rapidly and, in 2016, more than 80% of people in USA, UK, Australia, Spain, Israel and Germany between others, had Internet access[1]. This massive use of Internet can be translated in communication, re-

[1] Pew Research Center. Last access: July 4th, 2022. Available on https://www.pewresearch.org/global/2016/02/22/internet-access-growing-worldwide-but-remains-higher-in-advanced-economies/

sources, information and *data*. Thus, we started to live the exploitation of digital technologies.

Nowadays, 2.5 quintillion bytes of data are uploaded at day through Internet[2], and machine learning (ML) methods allow to use them for different purposes. Particularly, if the online resource is text, tasks as topic labeling, news classification, question answering, named entity recognition and sentiment analysis [1] are frequently solved with ML.

There are some interesting papers reviewing text classification algorithms [2–5]. They range from traditional algorithms such as Naïve Bayes, Decision Trees and Support Vector Machines to Deep Learning methods such as Recurrent and Convolutional Neural Networks, and Transformers. Besides the classifiers and the proposed systems, we believe that it is equally important to discuss the way in which data is represented for the understandability and efficacy of ML methods. For that reason, we analyze different representations for texts in a particular case of study, aiming to show the adequacy of each one with standard ML classifiers.

**Contributions.** We review traditional *document-level representations* such as those that rely on term frequencies (Bag of Words (BoW) with different weighting schemes [6, 7]), and more elaborate and compact ones (Concise Semantic Analysis (CSA) [8] and Latent Semantic Analysis (LSA) [9]). Bags of tokens[3] condense the document into a single count vector. Each position of that vector has associated a particular token and its value represents the frequency in the document. CSA extracts concepts from the texts and relates each document with the words in the corresponding concept space. LSA discovers latent topics of the texts and represents those as a mixture of topics which are probability distributions over words. Finally, we move to *featurized word-level representations* which capture the semantic and syntactic correlation between words. Common used features are aspects, relations and words, and are represented by dense vectors of fixed size named *embeddings*. Some examples of words embeddings are Word2vec [10] and GloVe [11]. Thus, we analyze several dimensions of each representation like the performance obtained with state-of-the-art classifiers on a particular problem to solve: polarity analysis of IMDB reviews related to movies. We also discuss the size of the obtained representations and the time required to get them. We conclude the work with some highlights about the adequacy of the text representations for this particular problem.

**Organization.** In Section 2 we describe the polarity analysis task and the corresponding dataset used in this paper. In Section 3 we briefly define the document-level and featurized word-level representations for the texts analyzed. Then, the experimental study carried out is in Section 4. Finally, Section 5 summarizes the main conclusions obtained and future work.

[2] Earthweb web site. Last access: July 4th, 2022. Available on https://earthweb.com/how-much-data-is-created-every-day/

[3] We assume that tokens would be words, n-grams, concepts, classes, topics, etc.

## 2 Case Study: Polarity Analysis on IMDB Movie Reviews

*Sentiment Analysis* is one of the most popular applications of ML in Natural Language Processing since it is relevant in many domains such as marketing, politics, research, affective computing and so on [12–15]. However, it is a challenging field because when people express their feelings, emotions and opinions, language can be used to persuade, encourage and communicate a positive or negative form of thinking. Especially when analyzing written opinions, due to the use of non-verbal language and the missing context, it is even more difficult to detect ambiguity and sarcasm.

In general, Sentiment Analysis encompasses several aspects like polarity, subjectivity, intensity, intentions and aspect-based analysis, as well as specific emotions identification. Depending on the task selected, regressors or classifiers are trained [16–18]. Particularly, *Polarity Analysis* task can consider continuous or discrete values for documents labels. When the labels are continuous numbers, usually they vary between -1 (negative) and 1 (positive). Although when the labels are categorical they are mostly classified as positive, negative or neutral.

In this work, we used IMDB Movie Review Dataset [19] which was built in view of the latter approach and taking into account only positive and negative classes. Therefore, it is a task review-level polarity classification, where a classifier must predict if a review is *positive* or *negative* given its text. IMDB Movie Review Dataset consists of 50,000 reviews about movies collected from Internet Movie Database (IMDb[4]). This website contains information related to films, television series, videogames and streaming content, information about cast, production crew, plot summaries, ratings, also fan and critical reviews.

When the authors built this dataset they extracted only ratings and reviews. Hence, they considered only highly polarized reviews, that is, a negative review has a score less than or equal to 4, and a positive review has a score greater than or equal to 7 (considering a scale from 1 to 10). Neutral reviews were not included in the collection. Additionally, there are not more than 30 reviews assigned per movie, this is to avoid correlation of common terms (for example, characters, actors, events, etc.). The dataset is divided into two sets of 25,000 reviews. One set is used to train and the other one to test the different proposed models. Also each collection is balanced in their amount of positive and negative labeled reviews and there are no movies in common between the two sets. We choose this well-known corpus because it is the most utilized in polarity analysis task.

## 3 Evaluated Models

A *document* is a unit of textual data that belongs to a collection and constitutes one of the main lexical resources of text mining. To extract relevant information from a document, ML approaches should receive the data in an adequate form, that is, a proper representation of the text.

[4] www.imdb.com

As mentioned in Section 1, the objective of this work focuses on studying different types of representations, such as the simple *BoW*, more elaborate and compact ones such as *CSA* and *LSA*, and the featurized word-level vectors as *Word2vec* and *GloVe*. These text representations were combined with standard classifiers to obtain the models. We considered *Support Vector Machines* with Linear and RBF kernels, the *Multinomial* version[5] of the probabilistic model *Naïve Bayes* and the decision trees classifier named *Random Forest*. Next, we describe briefly these representation techniques considered in our experimental study.

The *BoW* strategy builds a set with the words of the documents, that is, the *vocabulary* of the dataset. Formally, a document $d$ is represented by a vector of weights $d_{BoW} = (w_1, w_2, ..., w_n)$ where $w_i$ depends on the selected weighting scheme (boolean, term frequency, term frequency-inverse document frequency, etc.) and $n$ is the size of the vocabulary of the collection. This representation is one of the most used in text categorization tasks because it is simple to implement, the classifiers perform relatively well and it is possible to consider different weighting schemes for the terms. However, *BoW* have known limitations such as the order of the words in the document is lost and, context and grammar are not considered, causing the loss of semantic and conceptual information.

The *CSA* technique proposes a vector representation of low dimensionality with a high level of representativeness. A space of concepts is built according to the categories used for the classification task. Then, for each word of the vocabulary, a value that indicates the relationship between the word and each concept is calculated and stored in a vector. Finally, the representation of the document is obtained by adding the vectors corresponding to all the words included in the document, weighted by the relative frequency of each word in the document.

*LSA* is a method that uses the contexts in which a word appears and associates it with a meaning. In order to find a small subset of concepts, *LSA* analyzes the relationship between the meanings of the documents and the words in those (the latent space). The matrix word-document is created, where the rows correspond to the words and the columns to the documents. The components $f_{ij}$ of that matrix indicate how many times the word $i$ is in the document $j$. By applying a weighting scheme, the model is improved to give relevance to the word that appears in the context of the documents. This representation faces polysemy and synonymy problems, which are alleviated by applying the Singular Value Decomposition technique. The new matrix corresponds to a concept-document one which is the representation of each document.

*Word2vec* is a neural model that computes feature vectors, usually named word embeddings, by processing a large collection of documents. Formally, given a set of $n$ words and a context window of size $r$ (the $r$ words before and after the target word), the model tries to predict which is the target word based on their neighbors. This is the Continuous Bag-of-Words version. On the other hand, Skip-Gram approach consists in predict the neighboring words from the target word. The resultant embeddings are vectors of dimensions $1 \times W$ with

[5] The multinomial variant of Naïve Bayes is usually employed with textual data.

$W$ denoting the number of neurons in the hidden layer of the neural network. Finally, the document representation can be obtained averaging or summing the embeddings of each word in the text. *Word2vec* efficiently models related words, which appear in similar contexts, and the representation captures the semantic relationship between them. An advantage of considering *Word2vec* embeddings is the possibility to use pre-trained vectors to represent the data for a particular task. In this way, the hard process to obtain the embeddings is avoided. However, this model has some problems representing out-of-vocabulary words, even there are no shared representations at sub-word levels.

*GloVe*, stands for Global Vectors, is an unsupervised learning algorithm for obtaining vector representations of words. The technique generates a continuous vector space using matrix factorization and local context windows. The training is performed on global statistics of word-word co-occurrence of texts. The computed representations are based on the proportion of the co-occurrence probabilities that are obtained from the semantic relationship between words. When that relationship exists, the probability is large, whereas that in the case of words that are related to only few words, the value obtained is small. After computing the word embeddings, the representation of the document is obtained averaging or summing the word vectors of the text. *GloVe* captures syntactic and semantic information with good results, but it provides limitations related to the detection of word analogies. The model is straightforward to obtain and reproduces correctly sub-linear relationships in the vector space. There are evidence that *GloVe* performs better than *Word2vec* in word analogy tasks, although the problem of words out-of-vocabulary is still present.

## 4 Experimental Study

The execution of the experiments was carried out with *Google Colaboratory*[6] platform. We employed *Jupyter notebooks* with Python language (version 3.7.13) to code the routines to generate, train, test and evaluate the models. In particular, we utilized the following main libraries for Natural Language Processing[7]: *sklearn* (version 1.0.2), *gensim* (version 3.6.0), *nltk* (version 3.7), *spacy* (version 3.3.1), *numpy* (version 1.21.6) and *pandas* (version 1.3.5).

The corpus used to generate and evaluate the models was described in Section 2. It consists of 50,000 movie reviews extracted from IMDd site. We consider the original training and testing sets, that is, 25,000 samples to train and 25,000 more to evaluate. In each subset there is the same amount of reviews with positive and negative polarities. The reviews were pre-processed by converting all the characters to lowercase and removing stop words, numbers, punctuation marks and any special character. The hyperparameters for each representation are described below.

In order to obtain the vocabulary of the BoW representation, we selected several subsets of different amount of terms. We tested with the 100, 500, 1,000,

[6] https://colab.research.google.com/
[7] This information is important for reproducibility isuues.

2,500, 5,000, 7,500 and 10,000 most frequent terms. Then, we evaluated *unigrams* and *bigrams* of words, *unigrams + bigrams* of words and *trigrams* of characters. We considered those proposals with *Boolean*, *Term Frequency* and *Term Frequency–Inverse Document Frequency* (TF-IDF) weighing schemes.

Since reviews are classified into two classes, i.e. positive and negative, the resulting vectors of CSA representation are two-dimensional. Meanwhile, in LSA, we considered 100 and 300 concepts. In addition, this last representation was calculated considering TF-IDF weights.

Regarding word embeddings representations, for Word2vec and GloVe approaches, we made use of pre-trained vectors provided by Google[8] and Stanford University[9], respectively. Word2vec pre-trained vectors were generated from *Google English News* corpus, with Skip-Gram architecture, a context windows size of 5 words and 300 dimensions. On the other hand, for GloVe pre-trained embeddings, we considered the ones learned from *Twitter* with vector's sizes of 25, 50, 100 and 200. We also used the GloVe embeddings trained with *Common Crawl* data which have 300 components. For both methods, the document embeddings were obtained by averaging the word vectors present in the corresponding text.

The different representations were evaluated using *Support Vector Machine* classifiers with linear (SVM-Linear) and RBF (SVM-RBF) kernels, *Multinomial Naïve Bayes* (MNB) and *Random Forest* (RF). It should be noted that we employed the default parameters of each classifier. Finally, we used *precision*, *recall* and *F-Measure* to evaluate the performance of the classifiers in a single run.

In Table 1 we present a summary of the best F-Measure values obtained for each model (representation and classifier). The highest values are highlighted in bold.

**Table 1.** Summary of the best F-Measure values obtained with different models and the dimension of each document representations.

| Document Representation | Size | SVM-Linear | SVM-RBF | MNB | RF |
|---|---|---|---|---|---|
| **BoW**$_{(Unigram+Bigram)}$ | 10,000 | 0.86 | 0.88 | 0.84 | 0.84 |
| **CSA** | 2 | **0.93** | **0.92** | **0.92** | **0.88** |
| **LSA** | 300 | 0.86 | 0.87 | 0.82 | 0.80 |
| **Word2vec** | 300 | 0.85 | 0.86 | 0.74 | 0.81 |
| **GloVe** | 300 | 0.85 | 0.83 | 0.83 | 0.83 |

Based on all the experiments executed for BoW representation, the combination of *unigrams + bigrams* of words with TF-IDF weighting scheme and

[8] Word2vec pre-trained embeddings downloaded from https://code.google.com/archive/p/word2vec/

[9] GloVe pre-trained embeddings available at https://nlp.stanford.edu/projects/glove/

considering the 10,000 most frequent words, was the best. The models using BoW obtained good performance (between 0.86 and 0.88) with SVM classifiers.

Slightly lower performance was achieved with models that include embeddings as text representations. The best metrics were reached with vectors of 300 dimensions and, in the case of GloVe, using the pre-trained embeddings from *Common Crawl*.

The highest result obtained with models considering LSA utilized 300 concepts when building the representation. The performance of SVM classifier with LSA is similar to that of BoW, but with the other classifiers are barely lower.

The best F-Measure values in our experimental study were achieved with CSA representation which has only 2 dimensions. All those values are equal or higher than 0.92 with the exception of RF classifier which is 0.88.

Table 2 presents a summary of the time taken to generate the different document representations (see *Generation* column) and the execution time of the classifiers, expressed in seconds. The columns corresponding to the classifiers indicate the sum of the training and the testing stages of the best models stated in Table 1. This allows us to take into account the temporal complexity of each model. A value of 0 (second) in Table 2 means a period of time less than 1 second.

**Table 2.** Time consumed by the different representations and classifiers expressed in seconds.

| **Representation** | **Generation** | **SVM-Linear** | **SVM-RBF** | **MNB** | **RF** |
|---|---|---|---|---|---|
| **BoW** | 25 | 0 | 1,397 | 3 | 63 |
| **CSA** | 1,257 | 0 | 18 | 0 | 3 |
| **LSA** | 29 | 60 | 372 | 0 | 60 |
| **Word2vec** | 4,551 | 8 | 211 | 0 | 200 |
| **GloVe** | 62 | 124 | 1,271 | 0 | 221 |

The representation that took the longest time was Word2vec (4,551 seconds). The reason, possibly, is that it includes the time considered to load the full set of pre-trained vectors into memory. Meanwhile, despite of CSA is the second slowest in computing the representation, it is only a quarter of the time Word2vec took. On the other hand, if we consider the time spent for the models using CSA, all classifiers are very fast.

Analyzing the execution time involved in generating the embeddings of documents, GloVe has minimal execution time compared to Word2vec, and their performance (see Table 1) is similar.

Finally, it is worth mentioning that the times to obtain BoW, LSA and GloVe representations are really good (62 seconds or less). Regarding the classifiers, MNB always was the fastest, followed by SVM-Linear. Whereas SVM-RBF generally took the longest time to train and test the models.

## 5 Conclusions and Future Work

In this article, we analyzed different text representations for Polarity Analysis task. From the experimental study, we can conclude that models including CSA representation obtained the best F-Measure values. The CSA vectors representing the documents are really small (just 2 dimensions) and CSA-based models are the fastest to execute. CSA preserves only the necessary information for the classifiers extracting few concepts from categories and then interprets those concisely on the words of the documents. Thus, this representation can be considered robust and efficient demonstrating to be adequate for this task.

Regarding BoW with unigrams+bigrams of words, TF-IDF weighting scheme and vocabulary of 10,000, we found that it is fast to obtain and the models generally run rapidly (except when SVM classifier is used with RBF kernel). Nevertheless, the size of the representation is large, more precisely 10,000 dimensions.

Another text representation that can be obtained quickly is LSA. With just 300 dimensions, the analyzed models including LSA as representation are fast to execute beyond the good performance they showed.

Finally, the models including embeddings-based representations perform similarly well although Word2vec is slower to obtain compared to GloVe.

In relation to the time complexity of the classifiers, MNB and SVM demonstrated to be very efficient in comparison to RF when evaluating the training and testing stages. Also, if we focus in their performance, we observed that they were really good.

As future work we plan to continue the analysis of these text representations combining them with neural networks-based classifiers. We are also interested in deeply study the models which use embeddings to fine-tune their pre-trained word vectors for this particular tasks. Finally, we want to include more advanced models such as BERT in our study.

**Acknowledgments.** Authors thank project PROICO 03-0620 of the Universidad Nacional de San Luis and Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina for the continuous support to the research.

## References

1. Gasparetto, A., Marcuzzo, M., Zangari, A., Albarelli, A.: A Survey on Text Classification Algorithms: From Text to Predictions. Information. 13(2), 83 (2022)
2. Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., He, L.: A Survey on Text Classification: From Traditional to Deep Learning. ACM Transactions on Intelligent Systems and Technology (TIST). 13(2), 1–41 (2022)
3. Li, R., Liu, M., Xu, D., Gao, J., Wu, F., Zhu, L.: A Review of Machine Learning Algorithms for Text Classification. In: Lu, W., Zhang, Y., Wen, W., Yan, H., Li, C. (eds) Cyber Security. CNCERT 2021. Communications in Computer and Information Science, vol 1506, pp. 226-–234. Springer, Singapore (2022)

4. Riduan, G. M., Soesanti, I., Adji, T. B.: A Systematic Literature Review of Text Classification: Datasets and Methods. In: 2021 IEEE 5th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), pp. 71–77. IEEE, Purwokerto, Indonesia (2021)
5. Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D.: Text Classification Algorithms: A Survey. Information. 10(4), 150 (2019)
6. Salton, G., McGill, M. J.: Introduction to Modern Information Retrieval. McGraw-Hill Book Co., New York (1983)
7. Feldman, R., Sanger, J.: The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press, (2007)
8. Li, Z., Xiong, Z., Zhang, Y., Liu, C., Li, K.: Fast Text Categorization Using Concise Semantic Analysis. Pattern Recognition Letters. 32(3), 441–448 (2011)
9. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., Harshman, R.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science. 41(6), 391–407 (1990)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. In: 1st International Conference on Learning Representations (ICLR) 2013. Workshop Track Proceedings, Scottsdale, Arizona, USA (2013)
11. Pennington, J., Socher, R., Manning, C. D.: GloVe: Global Vectors for Word Representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543. Association for Computational Linguistics, Doha (2014)
12. Rambocas, M., Pacheco, B. Online Sentiment Analysis in Marketing Research: A Review. Journal of Research in Interactive Marketing. 12(2), 146–163 (2018)
13. Bose, R., Dey, R. K., Roy, S., Sarddar, D.: Analyzing Political Sentiment Using Twitter Data. In: Satapathy, S., Joshi, A. (eds) Information and Communication Technology for Intelligent Systems. Smart Innovation, Systems and Technologies, vol 107, pp. 427–436. Springer, Singapore (2019)
14. Yousif, A., Niu, Z., Tarus, J. K., Ahmad, A.: A Survey on Sentiment Analysis of Scientific Citations. Artificial Intelligence Review. 52(3), 1805—1838 (2019)
15. Cambria, E., Das, D., Bandyopadhyay, S., Feraco, A.: Affective Computing and Sentiment Analysis. In: Cambria, E., Das, D., Bandyopadhyay, S., Feraco, A. (eds) A Practical Guide to Sentiment Analysis, vol. 5, pp. 1–10. Springer, Cham (2017)
16. Yadav, A., Vishwakarma, D. K.: Sentiment Analysis Using Deep Learning Architectures: A Review. Artificial Intelligence Review. 53(6), 4335-4385 (2020)
17. Birjali, M., Kasri, M., Beni-Hssane, A.: A Comprehensive Survey on Sentiment Analysis: Approaches, Challenges and Trends. Knowledge-Based Systems. 226, 107–134 (2021)
18. Wankhade, M., Rao, A. C. S., Kulkarni, C.: A Survey on Sentiment Analysis Methods, Applications, and Challenges. Artificial Intelligence Review. 55, 1–10 (2022).
19. Maas, A. L., 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. Association for Computational Linguistics, Portland, Oregon, USA (2011)