How Dimensionality Reduction affects Sentiment Analysis NLP Tasks: An Experimental Study

Leonidas Akritidis and Panayiotis Bozanis

School of Science and Technology, International Hellenic University, 14th km Thessaloniki, Nea Moudania 570 01, Thessaloniki, Greece
{lakritidis,pbozanis}@ihu.gr

Abstract. Dimensionality reduction is a well-known technique for limiting the size of the feature space and for discovering latent meaningful variables in the input data. It is particularly valuable when the raw data is sparse and its processing by machine learning algorithms becomes computationally very expensive. On the other hand, sentiment analysis refers to a collection of text classification methods that identify the polarity of the user opinions in blog posts, reviews, tweets, etc. However, since text is naturally very sparse, training classification models is often intractable, rendering the importance of dimensionality reduction even greater. In this paper we study the impact of dimensionality reduction in sentiment analysis classification tasks. Through extensive experimentation with traditional algorithms and benchmark datasets, we verify the general intuition that the dimensionality reduction methods significantly improve the data preprocessing times and the model training durations, while they sacrifice only small amounts of accuracy. Simultaneously, we highlight several exceptions to this rule, where the training times actually increase and the accuracy losses are significant.

Keywords: sentiment analysis · sentiment classification · text classification · dimensionality reduction · PCA · SVD

1 Introduction

Sentiment analysis refers to the popular problem of recognizing the polarity of user opinions in standard, usually unstructured excerpts of text. The opinion polarities may be binary (positive or negative), ternary (e.g., neutral), or fall into a specific range (for example, ratings within 1–5 or 1–10 scale). Nowadays, sentiment analysis techniques are widely applied in numerous applications, with the aim of automatically evaluating the submitted user opinions. Indicative examples include blog communities [2,4], customer reviews [13,16], social networks [6,18], microblogs [22, 23], forums, messengers, and so on.

From the perspective of machine learning, the sentiment analysis algorithms fall into the broader category of text classifiers. Text classification is one of the most well studied machine learning problems, and a vast amount of research is presently conducted towards the improvement of the existing models [1, 3, 10].

The most common approaches train either binary, or multi-class text classifiers by utilizing manually, or artificially labeled training sets. The learned models are subsequently employed to determine the polarity of user opinions in previously unseen text corpora.

The most recent advances in the area include sophisticated NLP methods based on deep learning architectures such as the LSTM [5, 12], Convolutional Neural Networks [19], attention-based Transformers [24], etc. These methods have been proved effective at capturing the text semantics either at word level [14, 20], or at sentence level [9, 24]. Despite their success in pure NLP tasks (e.g., in machine translation), the traditional classification algorithms are still of great usefulness because they combine simplicity, effectiveness, and fast training rates.

Since the vast majority of machine learning algorithms work with numerical vectors, the typical raw text representation is not applicable, and the input documents must be appropriately transformed (namely, *vectorized*) to satisfy this requirement. Nevertheless, the traditional text vectorization methods such as tf-idf, convert each word (or n-gram) of the corpus to a distinct feature with its own weight. Hence, they lead to representations of very long and sparse vectors, with hundreds of thousands, or even millions of features, that negatively affect both the model training durations and memory consumption. These side effects are broadly known in the literature as the *curse of dimensionality*.

To limit its consequences, the dimensionality reduction methods construct latent spaces of lower dimensionality, and project the original feature vectors onto that space. This is primarily performed by applying matrix decomposition (or factorization) techniques, so that the least significant features are discarded, and the most significant ones are used to build another space of lower dimensionality. In this way, they partially confront the problems of the curse of dimensionality, usually with a small sacrifice in the generated model accuracy.

This paper presents an extensive study of the effectiveness of dimensionality reduction in sentiment analysis NLP tasks. Our motivations derive from i) the importance of sentiment analysis and its wide adoption by numerous systems, and ii) the limited number of similar studies in the relevant literature. In general, our experimental evaluation confirmed that the dimensionality reduction methods greatly improve the model training durations, while they introduce small accuracy losses. Nonetheless, the main contribution of this work lies in the detection of several cases where this observation does not hold. In these cases, the training durations on lower dimensional spaces were actually increased, whereas the accuracy losses were significantly larger than normal. This indicates that an in-depth research is required to accurately evaluate the impact of dimensionality reduction in sentiment analysis classification.

The rest of this paper is organized as follows: Section 2 presents the current advances in the fields of text classification, sentiment analysis, and dimensionality reduction algorithms. Subsequently, Sections 3 and 4 present the results of our experimental study and discuss the significance of our findings, respectively. Finally, Section 5 summarizes the conclusions of this research and outlines the most important elements of our future work.

2 Related Work

The rapid growth of the (micro)blogging communities, social networks and commercial platforms has rendered sentiment analysis one of the hottest topics in the Natural Language Processing (NLP) research field. As a result, the relevant literature includes numerous works that deal with this interesting problem. The earliest articles tested the performance of several text classifiers, such as Naive Bayes, SVMs, Neural Networks, etc., in small datasets containing user reviews for products and events [11, 15]. A survey on multiple sentiment analysis algorithms and applications was conducted in [13].

During the past few years, the deep learning methods have been proved very successful in numerous classification tasks. Therefore, a significant amount of research has focused on the exploitation of these methods in the problem of sentiment analysis. More specifically, [19] proposed a framework based on the Convolutional Neural Networks (CNN) and the Word2Vec model [14] to improve the accuracy and generalizability of their approach.

In addition, the Long Short-Term Memory (LSTM) networks are presently among the most effective deep learning methods for modeling sequential data. They have been applied extensively in numerous text classification problems, including sentiment analysis. For instance, in [25], the authors introduced an attention-based network for aspect-level sentiment classification, whereas [12] augmented the LSTM framework with a stacked attention mechanism consisting of attention models for both the target and sentence levels. Furthermore, in [5], the LSTM Recurrent Neural Network (RNN) has been utilized to perform sentiment classification and topic discovery in COVID-19 online discussions.

The transformers constitute another recent, yet powerful deep learning architecture for NLP tasks. Similarly to RNNs, they also process sequential data, but not necessarily in the provided order. In contrast, they implement an attention mechanism that identifies the context at any position in the input sequence [24]. The work of [17] introduced DICET, a transformer-based method for Twitter sentiment analysis. DICET encodes the transformer representations, and applies a contextual embedding technique to enhance the quality of tweets. Moreover, [26] presented an experimental comparison between 5 sentiment analysis tools and 4 pre-trained Transformer-based models on six datasets, demonstrating that the 4 fine-tuned models were superior to the 5 tools by a margin of 6.5-35.6%.

On the other hand, dimensionality reduction is a well-established technique for reducing the size of the input space and limiting the side effects of the curse of dimensionality. In the context of sentiment analysis, it has been applied to project the initial feature vectors onto a lower dimensional space that retains the most important input variables. Indicatively, [8] introduced a semi-supervised Laplacian eigenmap, called SS-LE, that discards the redundant features by decreasing the detection errors of the sentiments. In [21], the authors employed the traditional Singular Value Decomposition (SVD) to perform dimensionality reduction in sentiment classification. Finally, the method of [7] considered the label and structural information of text by adopting a semi-supervised approach for feature weighting and extraction.

Although the effectiveness of the deep learning architectures in sentiment classification is unquestionable, the traditional machine learning algorithms are still of great usefulness, mainly because they are able to quickly train simple, yet powerful models. In the following experiments, we demonstrate multiple cases where simple algorithms, such as Logistic Regression, combine both decent classification quality and fast training procedures.

This preliminary work is among the first to consider the implications of dimensionality reduction in the performance of these algorithms in sentiment analysis tasks. In future versions, we intend to further extend this study with the aim of covering the majority of the aforementioned deep learning architectures.

3 Sentiment Analysis and Dimensionality Reduction

This section presents the experimental evaluation of the effectiveness of various machine learning algorithms in sentiment analysis NLP tasks, against the dimensionality of the underlying feature space. The analysis is organized in four subsections: i) Subsection 3.1 describes the characteristics of the utilized datasets, ii) Subsection 3.2 briefly presents the six classification algorithms that participated in our tests along with the selected values of their hyper parameters, iii) Subsection 3.3 outlines our text preprocessing methodology, and iv) Subsection 3.4 discusses the accuracy results of the classifiers and the model training durations.

All the experiments were conducted on a single workstation equipped with 32GB of RAM, and an Intel Core i7-7700 processor running at 3.6GHz. The code was developed by using several Python libraries, whereas all the algorithms were executed without CPU or GPU parallelization. To facilitate the reproducibility of the presented results, we released the code on Github[1].

3.1 Datasets

Four popular sentiment analysis datasets were used in this study. All of them are publicly available, and have been utilized extensively in the relevant literature for evaluating NLP algorithms. In particular, IMDb[2] consists of 50,000 movie reviews for binary sentiment classification. The second dataset, Twitter US Airline[3], includes more than 14 thousand tweets with ternary user opinions (namely, positive, negative, and neutral) on five major US airlines.

The third dataset also originates from Twitter, and includes a collection of roughly 28 thousand tweets[4]. The tweets have been authored by a set of influential users, and contain positive, negative, and neutral opinions on publicly traded companies. The dimensionality of the input vector space of this dataset, after text cleaning and data preprocessing (full details are provided in Subsection 3.4), was approximately 12 thousand features.

[1] https://github.com/lakritidis/SADR
[2] https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
[3] https://www.kaggle.com/crowdflower/twitter-airline-sentiment
[4] https://www.kaggle.com/vivekrathi055/sentiment-analysis-on-financial-tweets

Table 1. Datasets for sentiment analysis accompanied by their characteristics

Dataset	**Instances**	**Dimensionality**	**Classes**
IMDb Movie Reviews	50,000	77,026	2
Twitter US Airline Sentiment	14,640	9,849	3
Financial Tweets Sentiment	28,437	12,138	3
Amazon Reviews (office products)	53,258	35,229	5

A subset of the Amazon Reviews[5] collection was also included in our analysis. This dataset contains about 53 thousand reviews on office products, accompanied by user ratings in a 1–5 scale. In this case, the dimensionality of the input space, after cleaning and preprocessing, was considerably lower than that of IMDb; specifically, 35,229. The most important characteristics of these four datasets are summarized in Table 1.

3.2 Text Classifiers

Six classification methods were used in this work: k-Nearest Neighbors (kNN), Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), Support Vector Machines (SVM), and the feed-forward Artificial Neural Network (ANN). Each classifier may be parameterized by fine-tuning several hyper parameters. Nevertheless, notice that a head-to-head comparison between these classifiers is out of the scope of this article. In contrast, we are primarily interested in evaluating the impact of dimensionality reduction in the performance of these algorithms. For this reason, we did not attempt to individually fine-tune the hyper parameters to improve the performances in terms of both effectiveness and efficiency; rather, we used typical values that yielded decent classification accuracy values.

In Table 2 we refer to these 6 algorithms, and we report some indicative values for their respective hyper parameters. In brief, the nearest neighbor queries of kNN were executed with $k = 10$, whereas the distances were measured by using the Minkowski metric. Regarding Logistic Regression, the optimization of the cost function was performed by using the Limited-memory BFGS algorithm, and the maximum number of iterations was set to 300. The Decision Trees that we trained were programmed to expand until all their leaves are pure. The same setting was also applied to the 100 estimators that were included in our Random Forest classifier. Concerning SVM, we employed the RBF kernel, since, in general, it performed better than the linear kernel on our data. Similarly to Logistic Regression, we also applied L2 regularization in the SVM classifier. The architecture of the ANN included two hidden layers, with 50 and 300 neurons, respectively, whereas ReLU was used as the network activation function.

Finally, for all binary classifiers that do not natively support multi-label classification, the well-established One-vs-Rest (OvR) technique was applied.

[5] https://jmcauley.ucsd.edu/data/amazon/

Table 2. Classifiers and hyper-parameters

Classifier	**Hyper-parameters**
k-Nearest Neighbors	$k = 10$, Minkowski distance
Logistic Regression	LBGFS, L2 regularization, Max iterations: 300
Decision Tree	Expand the tree until all leaves are pure
Random Forest	Estimators: 100, Expand the trees until all leaves are pure
SVM	RBF kernel, L2 regularization
Feed-Forward Neural Net	Architecture: (50,300), Activation function: ReLU

3.3 Text Preprocessing and Dimensionality Reduction

Before we proceed to the presentation of the results of our experimental study, we briefly describe the text preprocessing methodology. Initially, the input raw text was converted to lowercase, and a simple word-level tokenization process was executed to split each input document to a bag of words. In the sequel, the stop words were removed, and the WordNet lemmatizer was employed to convert each word to its meaningful base form. The collection was subsequently split to training and test sets by applying a constant ratio of 70%/30% and stratification by class. Finally, the two sets were transformed to numerical vectors by applying tf-idf vectorization, a well-established technique that leads to sparse, high-dimensional vectors.

Eventually, the dimensionality of both the training and test sets was reduced by applying a variant of Principal Component Analysis (PCA), called Truncated Singular Value Decomposition (TSVD). Notice that, although PCA identifies the principal components by maximizing the variance of the projected data, it is not feasible to sparse matrices. In contrast, TSVD does not center the data before computing the singular value decomposition; therefore, it operates efficiently on sparse matrices. In this context, TSVD is often known as Latent Semantic Analysis (LSA).

3.4 Results

Figures 1 and 2 illustrate the accuracy values (left diagrams) and the training durations (right diagrams) of the six classifiers on the 4 datasets of Table 1. In both types of diagrams, the horizontal axes are in logarithmic scale, and denote the various dimensionalities of the respective input vector spaces. Since the measured time differences among the different algorithms were frequently large (i.e., many orders of magnitude), we also adopted the logarithmic scale for the vertical axes of the right diagrams of all figures. Furthermore, notice that, in all figures, the rightmost markers represent the performances of the algorithms in the original feature space. That is, without applying dimensionality reduction.

Regarding the IMDb dataset, SVM and Logistic Regression were the most effective classifiers, since they respectively achieved accuracy values of 0.9 and 0.89 in the original feature space of the 77 thousand features. Nevertheless, the latter was much faster than the former, since its training duration was smaller

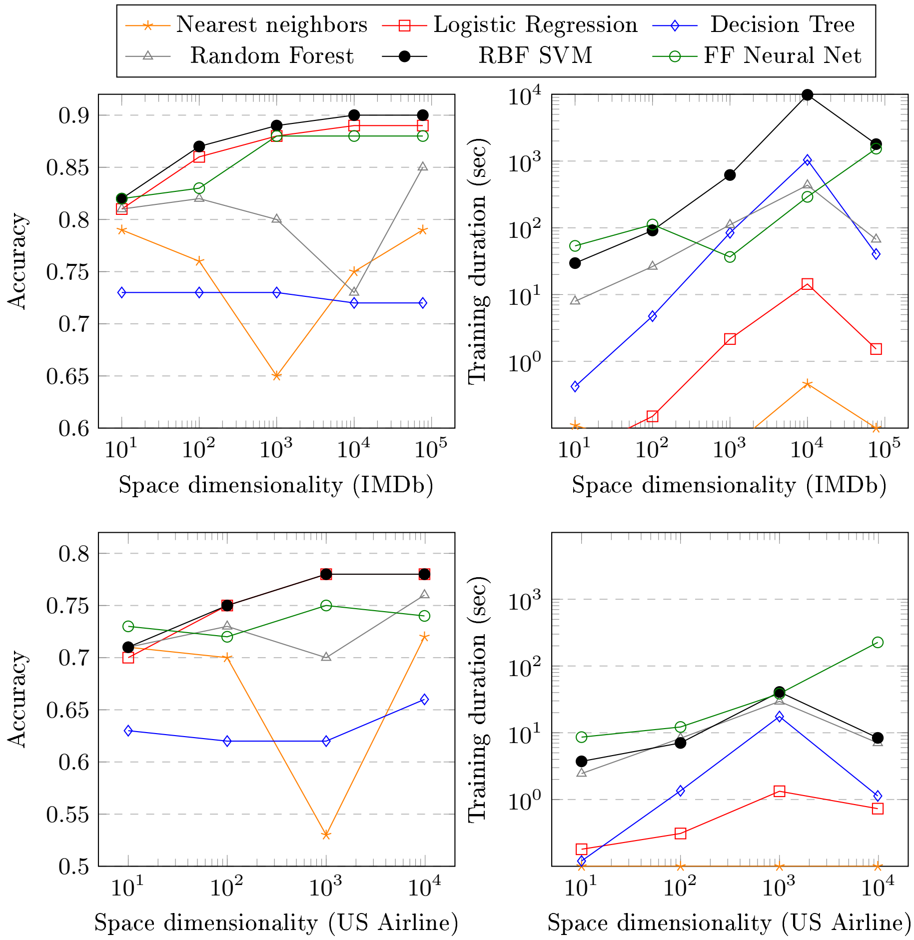

Fig. 1. Accuracy values (left) and training durations (right) of the six classifiers of Table 2 for the IMDb (top) and the Twitter US Airline (bottom) datasets. The horizontal axes are plotted in logarithmic scale, and represent various input spaces of different dimensionalities. In all diagrams, the rightmost markers denote the performance of the algorithms in the original feature spaces; namely, without dimensionality reduction.

than 2 seconds, compared to the roughly 30 minutes of SVM. The performance of ANN was competitive, since it achieved an accuracy of 0.88, by consuming about 25 minutes to learn the required weights and biases.

Next, we applied TSVD to project the data into vector spaces of 10, 10^2, 10^3, and 10^4 dimensions. Remarkably, the accuracy losses for the smallest vector space (i.e., with 10 dimensions) were not very large. More specifically, the performances of SVM and ANN have dropped to 0.82 from 0.9 and 0.88, respectively, whereas the training durations were drastically reduced to 30 and 54 seconds, respectively. LR and RF were almost equivalent to SVM and ANN, since their accuracy was slightly smaller; namely, 0.81. However, they were both much faster than their counterparts, since model training lasted for just 8 and 0.1 seconds, respectively.

Counter-intuitively, the accuracy of Random Forests was dropping as the input vector space was growing larger, from 10 to 10^4 components. On the other hand, the performance of the Decision Tree was almost unchanged and independent of the number of features. Another interesting conclusion derives by comparing the model training times of all classifiers in the original and the reduced 10^4-dimensional space. The model training was faster in the former case, with one exception for ANN. This is the second counter-intuitive observation, since one would expect that the utilization of shorter vectors would led to smaller execution times.

The performance of all classifiers was degraded in the Twitter US Airline Sentiment dataset (bottom diagrams of Fig. 1). In the original feature space, SVM and Logistic Regression were again the most effective algorithms, with accuracy values equal to 0.78, followed by Random Forests (0.76) and ANN (0.74). Logistic Regression was the fastest among these methods, with training times that were lower than one second. On the contrary, Random Forests and SVM consumed roughly 7 and 8.5 seconds, respectively, whereas ANN was significantly slower, since it took the backpropagation algorithm almost 4 minutes to learn the network parameters. The fastest algorithm among all six classifiers was kNN, with a rapid training time of 0.07 seconds.

Regarding the reduced input spaces, one may observe a pattern that is similar to the one of IMDb. More specifically, when the dimensions are reduced by one order of magnitude, i.e., they become equal to 10^3, the training times are counter-intuitively increased, again, with an exception for ANN. In some classifiers, namely, Random Forest, Decision Tree, and kNN, the accuracies are worse compared to those that were measured for the 100-dimensional space. Therefore, dimensionality reduction is actually meaningless in this case. In contrast, a reduction by two orders of magnitude (e.g., the vector space includes 100 components) leads to improved model training durations, accompanied by a tolerable decrease in the effectiveness.

ANN and the two tree classifiers were the most effective methods in the Financial Tweets dataset, since, in the original input space, all of them scored a very high accuracy of 0.97. LR and SVM were slightly outperformed in this case, with their accuracy values measured at 0.92 and 0.94, respectively. In terms of efficiency, kNN was the fastest method, followed by LR: they both consumed less than 1 second to train their models. Decision Tree and Random Forest were significantly slower, with roughly 3 and 9 seconds, respectively.

Similarly to the two previous datasets, the reduction of the dimensionality of the input vector space by one order magnitude rendered the algorithms slower, except for ANN. Indicatively, the model training procedure for SVM consumed 68 seconds in the original vector space with the 12 thousand dimensions, and 228 seconds in the reduced space of the 1000 dimensions. At this point, it is becoming solid that a limited reduction by one order of magnitude is only beneficial for Neural Networks, since the rest of the classifiers are rendered both slower and

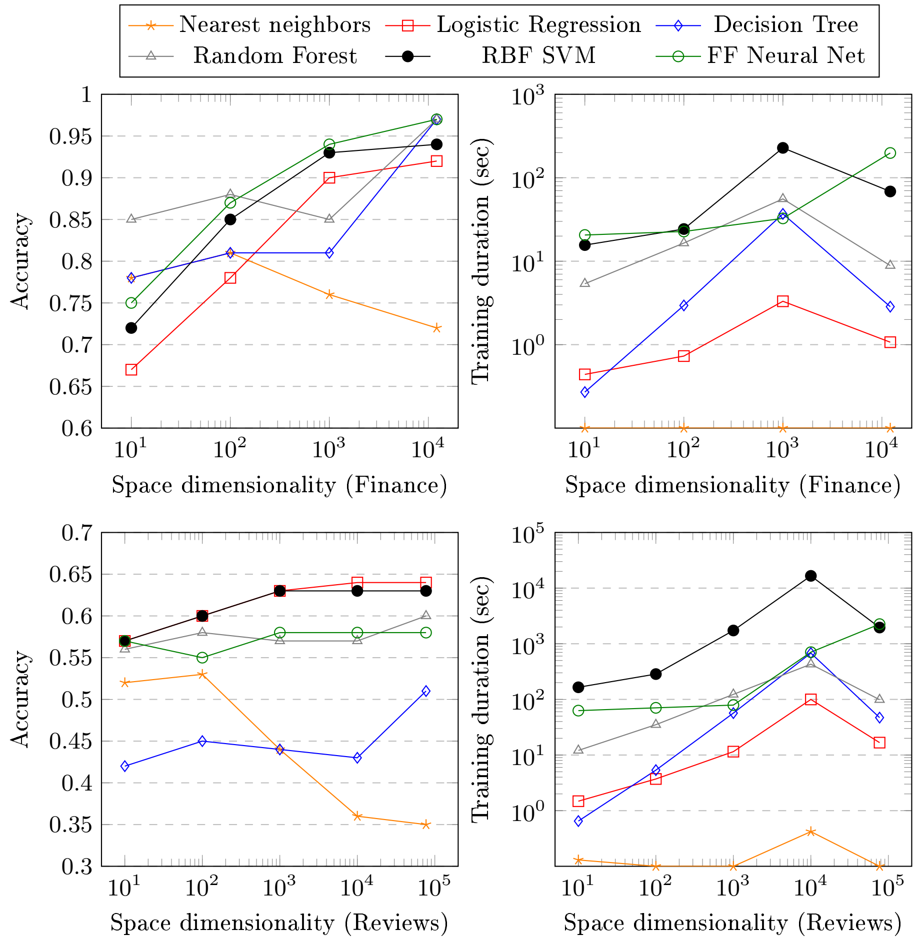

Fig. 2. Accuracy values (left) and training durations (right) of the six classifiers of Table 2 for Financial Tweets (top) and Amazon Product Reviews (bottom). The horizontal axes are plotted in logarithmic scale, and represent various input spaces of different dimensionalities. In all diagrams, the rightmost markers denote the performance of the algorithms in the original feature spaces; namely, without dimensionality reduction.

less accurate. The genuine gains in execution speeds are obtained by performing a more aggressive reduction, namely, by at least two orders of magnitude.

Finally, in the fourth dataset with the 53 thousand Amazon product reviews (bottom diagrams of Fig. 2), Logistic Regression was again the most accurate and the second fastest algorithm. In the original feature space, its accuracy was equal to 0.64, whereas model training consumed roughly 17 seconds. On the other hand, ANN and SVM were substantially slower (approximately 38 and 33 minutes, respectively) and slightly less effective; their accuracy values were 0.58 and 0.63, respectively. The aforementioned behavior in the reduced input spaces was repeated in this dataset. For 10^4 dimensions, SVM training lasted for 4.5 hours, whereas for 10^3 dimensions the respective time was 28 minutes.

4 Discussion

In the previous experimental analysis, some common behaviors of the different algorithms were identified. In this section, we attempt to carefully examine all these similar behaviors, with the aim of facilitating the formulation of several generalized conclusions. The following list summarizes our observations:

1. The experiments in all four datasets demonstrated that conducting a limited reduction in the dimensionality of the input vector space is not beneficial. More specifically, reducing the dimensions by one order of magnitude renders the classifiers both slower and less effective. The feed-forward ANNs are the only exception to this rule.
2. On the other hand, reducing the dimensions of the vector space by two orders of magnitude has a small to moderate impact in both the model training durations and accuracies.
3. The aggressive dimensionality reduction (e.g., input spaces of just 10 features, or reduced by three orders of magnitude) leads to significant, but not fatal accuracy losses. The model training times are substantially lowered, especially for the deep ANN and the non-linear SVM classifiers. In particular, a decrease by at least one order of magnitude in model training durations is achieved.

An additional, albeit not novel conclusion is that there is no golden classification method that outperforms all its adversaries in all tests. In terms of accuracy, the experimentation with four different data sets revealed five winning methods: SVM (IMDb and US Airline), Logistic Regression (US Airline and Amazon Reviews), and Decision Tree, Random Forest and ANN (in the Twitter Finance Sentiment dataset). On the other hand, in terms of model training durations, kNN and Logistic Regression were clearly the most efficient sentiment classification methods.

5 Conclusions and Future Work

This paper investigated the impact of dimensionality reduction in the performance of sentiment classification methods. Sentiment analysis is presently one of the hottest topics in NLP research, due to the explosive growth rates of social networks, blogging communities, commercial platforms, and so on. For this reason, a huge amount of research is conducted today with the aim of improving the performance of the current state-of-the-art models.

Nevertheless, text is a particularly sparse and high-dimensional form of data that occasionally triggers the notorious curse of dimensionality. This a condition where the majority of algorithms are rendered both inefficient and memory demanding. Consequently, dimensionality reduction plays a crucial role in the feasibility of the applied machine learning models, especially in NLP tasks such as sentiment analysis.

The experimental study that we conducted on four popular datasets with six major classification algorithms yielded several interesting conclusions. Firstly, reducing the dimensional space by one order of magnitude is rather meaningless, since it may be harmful for both model training durations and achieved accuracies. Secondly, reducing by two orders of magnitude leads to only small accuracy losses, but with small improvements in training times. The results of our study showed that significant benefits in the efficiency derive by reductions of three orders of magnitude, or more. Remarkably, the effectiveness degradation ranges from small to significant, albeit, not fatal.

We intend to further extend this research in the future with the aim of studying the implications of dimensionality reduction in the performance of the deep learning models. The current dominant NLP techniques, such as the RNNs, the LSTMs and the Transformers, are the objectives of our future work. We also intend to conduct experiments with additional state-of-the-art dimensionality reduction algorithms by employing large-scale training sets, with the aim of strengthening our conclusions.

References

1. Akritidis, L., Bozanis, P.: A supervised machine learning classification algorithm for research articles. In: Proceedings of the 28th ACM Symposium on Applied Computing. pp. 115–120 (2013)
2. Akritidis, L., Bozanis, P.: Improving opinionated blog retrieval effectiveness with quality measures and temporal features. World Wide Web **17**(4), 777–798 (2014)
3. Akritidis, L., Fevgas, A., Bozanis, P.: Effective products categorization with importance scores and morphological analysis of the titles. In: Proceedings of the 30th IEEE International Conference on Tools with Artificial Intelligence. pp. 213–220 (2018)
4. Boldrini, E., Balahur, A., Martínez-Barco, P., Montoyo, A.: Using EmotiBlog to annotate and analyse subjectivity in the new textual genres. Data Mining and Knowledge Discovery **25**(3), 603–634 (2012)
5. Jelodar, H., Wang, Y., Orji, R., Huang, S.: Deep sentiment classification and topic discovery on novel coronavirus or COVID-19 online discussions: NLP using LSTM Recurrent Neural Network approach. IEEE Journal of Biomedical and Health Informatics **24**(10), 2733–2742 (2020)
6. Kaya, T., Bicen, H.: The effects of social media on students' behaviors; Facebook as a case study. Computers in Human Behavior **59**, 374–379 (2016)
7. Kim, K.: An improved semi-supervised dimensionality reduction using feature weighting: Application to sentiment analysis. Expert Systems with Applications **109**, 49–65 (2018)
8. Kim, K., Lee, J.: Sentiment visualization and classification via semi-supervised nonlinear dimensionality reduction. Pattern Recognition **47**(2), 758–768 (2014)
9. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: Proceedings of the 2015 International Conference on Machine Learning. pp. 957–966 (2015)

10. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent Convolutional Neural Networks for text classification. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence. pp. 2267–2273 (2015)
11. Lane, P.C., Clarke, D., Hender, P.: On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data. Decision Support Systems **53**(4), 712–718 (2012)
12. Ma, Y., Peng, H., Khan, T., Cambria, E., Hussain, A.: Sentic LSTM: a hybrid network for targeted aspect-based sentiment analysis. Cognitive Computation **10**(4), 639–650 (2018)
13. Medhat, W., Hassan, A., Korashy, H.: Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal **5**(4), 1093–1113 (2014)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
15. Moraes, R., Valiati, J.F., Neto, W.P.G.: Document-level sentiment classification: An empirical comparison between svm and ann. Expert Systems with Applications **40**(2), 621–633 (2013)
16. Mukherjee, S., Bhattacharyya, P.: Feature specific sentiment analysis for product reviews. In: Proceedings of the 13th International Conference on Intelligent Text Processing and Computational Linguistics. pp. 475–487 (2012)
17. Naseem, U., Razzak, I., Musial, K., Imran, M.: Transformer based deep intelligent contextual embedding for twitter sentiment analysis. Future Generation Computer Systems **113**, 58–69 (2020)
18. Ortigosa, A., Martín, J.M., Carro, R.M.: Sentiment analysis in Facebook and its application to e-learning. Computers in Human Behavior **31**, 527–541 (2014)
19. Ouyang, X., Zhou, P., Li, C.H., Liu, L.: Sentiment Analysis using Convolutional Neural Network. In: Proceedings of the 2015 IEEE International Conference on Computer and Information Technology. pp. 2359–2364 (2015)
20. 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. pp. 1532–1543 (2014)
21. Shyamasundar, L., Rani, P.J.: Twitter sentiment analysis with different feature extractors and dimensionality reduction using supervised learning algorithms. In: Proceedings of the 2016 IEEE Annual India Conference. pp. 1–6 (2016)
22. Stieglitz, S., Dang-Xuan, L.: Emotions and information diffusion in social media—sentiment of microblogs and sharing behavior. Journal of Management Information Systems **29**(4), 217–248 (2013)
23. Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment in Twitter events. Journal of the American Society for Information Science and Technology **62**(2), 406–418 (2011)
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems **30** (2017)
25. Wang, Y., Huang, M., Zhu, X., Zhao, L.: Attention-based lstm for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 606–615 (2016)
26. Zhang, T., Xu, B., Thung, F., Haryono, S.A., Lo, D., Jiang, L.: Sentiment analysis for software engineering: How far can pre-trained transformer models go? In: Proceedings of the 2020 IEEE International Conference on Software Maintenance and Evolution. pp. 70–80 (2020)