Lifting the Curse: Exploring Dimensionality Reduction on Text Clustering Applications

Leonidas Akritidis
School of Science and Technology
International Hellenic University
Thessaloniki, Greece
Email: lakritidis@ihu.gr

Panayiotis Bozanis
School of Science and Technology
International Hellenic University
Thermi, Thessaloniki, Greece
Email: pbozanis@ihu.gr

***Abstract*—Nowadays, huge amounts of text are being generated on the Web by a vast number of applications. Examples of such applications include instant messengers, social networks, e-mail clients, news portals, blog communities, commercial platforms, and so forth. The requirement for effectively identifying documents of similar content in these services rendered text clustering one of the most emerging problems of the machine learning discipline. Nevertheless, the high dimensionality and the natural sparseness of text introduce significant challenges that threat the feasibility of even the most successful algorithms. Consequently, the role of dimensionality reduction techniques becomes crucial for this particular problem. Motivated by these challenges, in this article we investigate the impact of dimensionality reduction on the performance of text clustering algorithms. More specifically, we experimentally analyze its effects in the effectiveness and running times of eight clustering algorithms by employing six high-dimensional text datasets. The results indicate that, in most cases, dimensionality reduction may significantly improve the algorithm execution times, by sacrificing only small amounts of clustering quality.**

***Index Terms*—clustering, text clustering, dimensionality reduction, data reduction, unsupervised learning**

I. Introduction

Text clustering refers to the unsupervised problem of identifying and grouping together semantically similar documents in previously unexplored text collections. The explosive growth of numerous online applications that generate and manage huge volumes of text rendered the problem of text clustering particularly important. Examples of such applications include microblogs, shopping platforms, news portals, e-mail clients, digital libraries, and so on.

Motivated by the significance of the problem for both the scientific community and the industry, numerous researchers introduced several state-of-the-art solutions with the aim of confronting it. Nevertheless, text is generally characterized by high degrees of diversity. Therefore, two or more documents may express similar or identical meanings, despite the fact that they consist of completely different words. Text diversity is usually an undesired property, because it has several severe side-effects. First, it blurs the semantic similarities between two documents, making it hard to identify their thematic affinity. And second, it renders the data both sparse and highly dimensional, leading to a significant degradation of the performance of the relevant algorithms [1], [2].

In brief, high dimensionality and sparseness lead to very long vector representations that have the vast majority of their elements equal to zero. The algorithms that operate on such data –like text clustering algorithms– require large amounts of memory to accommodate and process all the vector components, whereas the execution times get drastically increased. This situation, where the high dimensionality of the input data may render an algorithm even infeasible, is broadly known as the *"curse of dimensionality"*.

The dimensionality reduction techniques have been developed with the aim of confronting the issues caused by the curse of dimensionality. By exploiting the statistical properties of the data (such as the variance of the input variables), or by applying matrix decompositions, these methods essentially transform the original data by projecting it onto spaces of lower dimensionality. Consequently, they decrease the memory requirements, allowing the machine learning algorithms to work more efficiently. The two most popular dimensionality reduction algorithms are Singular Value Decomposition (SVD) [3] and Principal Component Analysis (PCA) [4]. Notice that other feature engineering techniques (like feature selection methods) also lead to a reduction of the input space size. However, these techniques attempt to generate a representative subset of the input variables without transforming them, contrary to the dimensionality reduction algorithms.

In this paper, we conduct a study with the aim of evaluating the impact of dimensionality reduction techniques in text clustering applications. More specifically, we perform clustering by executing 8 major algorithms on 6 well-established text datasets. Simultaneously, we apply dimensionality reduction several times on these datasets, and we record the clustering performance and the running times of the algorithms for each dimensional space. With this strategy, we are able to compare the algorithm accelerations against the changes of their performance for multiple input space sizes.

The rest of the paper is organized as follows: Section II contains a brief reference to the most important clustering and dimensionality reduction algorithms of the relevant literature. Next, Section III describes the experimental setup (datasets, algorithms, etc.), and discusses the results of our study. Finally, Section IV summarizes the findings of this work and highlights its conclusions.

II. Related Work

The problems of clustering and dimensionality reduction are considered as the most representative topics in the unsupervised learning research field. Therefore, a large number of researchers are still working on the development of enhanced algorithms. According to their adopted logic, the current state-of-the-art solutions can be categorized into several classes.

Specifically, the space partitioning methods attempt to solve the problem by separating the space into distinct, non-overlapping subspaces. All the data points that fall into the same subspace are considered proximal and they are subsequently grouped together into the same cluster. k-means is the most popular space partitioning method, and numerous variants have been introduced to improve it.

Among others, Spherical k-means founded a decomposition method for constructing concept vectors from the cluster centroids to perform text clustering [5]. In addition, [6] constructed a co-occurrence graph to identify groups of similar words. From the perspective of execution times, MiniBatch k-Means partitions the input data into subsets (called mini-batches) with the aim of decreasing the clustering duration without sacrificing the quality of the generated clusters [7].

Another category of clustering algorithms includes the hierarchical methods, with agglomerative clustering being the most popular one. In this scheme, the data elements are initially placed into an equal number of singleton clusters. Then, the two most similar clusters are progressively merged until a single cluster is generated, or no more similar clusters exist. Variants of agglomerative clustering may derive by adopting different cluster linkage methods; e.g., Ward, complete, average, etc [8], [9]. Another hierarchical algorithm is BIRCH, a memory efficient algorithm for large-scale text clustering [10], [11]. This method constructs a feature tree that allows the input data to be compressed (with losses) to a set of clustering feature nodes.

The spectral analysis methods are based on the interesting approach of extracting and examining the spectrum of a graph that reflects the affinity of the input samples. The original Spectral clustering algorithm initially constructs the similarity and the Laplacian matrix of the affinity graph. In the sequel, it performs eigen-decomposition of the Laplacian, and applies k-means to cluster the generated eigenvectors [12]. Its solid theoretical background and its high performance rendered Spectral clustering one of the most popular and well-studied methods in the area [13].

In case the clusters are not convex shaped, then the density-based methods offer an attractive alternative. In this context, DBSCAN considers that the clusters are high-density areas delimited by low-density areas [14], [15]. Therefore, the identified clusters may possess any shape, for example, ring, circle, sigmoid, etc. OPTICS is quite similar to DBSCAN, but it also constructs a graph that determines the reachability of a data point from other points [16]. Notice that, in text clustering applications, the clusters are usually convex-shaped. In such cases, the density-based methods are frequently outperformed by other techniques [17], [18]. Moreover, the numerous nearest neighbor queries that are required, render them considerably more expensive than other competitive approaches.

Regarding the dimensionality reduction algorithms, the two baseline methods are Singular Value Decomposition (SVD) [3] and Principal Component Analysis (PCA) [4]. Apart from those, Random Projection is a computationally efficient algorithm that projects the original high-dimensional data onto a lower-dimensional space by using a random matrix whose columns have unit lengths. This method has been proved quite successful in both image and text mining tasks [19]. In addition, the traditional SVD has been utilized effectively in sentiment classification applications to perform dimensionality reduction [20]. Finally, the method of [21] considered the label and structural information of text, by adopting a semi-supervised approach for feature weighting and extraction.

TABLE I
Text clustering datasets

Dataset	Samples	Dimensions	Clusters
Tweet	2472	5076	89
PriceRunner TVs	3564	2720	1280
TitleSet	11109	8079	152
SnippetSet	11109	18436	152
20 Newsgroups	20000	9887	20
Wines	11258	7173	88

III. Clustering and Dimensionality Reduction

This section presents the experimental study of the effects of dimensionality reduction on text clustering. The presentation is organized in 5 subsections that describe: i) the utilized datasets (III-A), ii) the clustering algorithms that participated in the study (III-B), iii) the performance evaluation measures (III-C), and iv) the results of the evaluation in terms of clustering effectiveness (III-D) and execution times (III-E).

All the experiments were conducted on a system equipped with 32GB of RAM and an Intel CoreI7 7700 processor running at 3.6GHz.

A. Datasets

Table I contains the basic attributes of the datasets that were utilized in this work. More specifically, the number of samples (i.e., documents), dimensions, and clusters are presented in columns 2, 3 and 4, respectively.

The first dataset is a collection of 2472 Tweets that were considered highly relevant to 89 queries of the TREC microblog tracks of 2011 and 2012[1]. After the application of several text cleaning and preprocessing filters (see the next subsection), about 5 thousand distinct words were included in the corpus. The second dataset draws its origin from PriceRunner, a popular online product comparison platform. It includes 3564 product titles that correspond to 1280 different TV models [17], [18], and it is publicly available on Kaggle[2].

[1] http://trec.nist.gov/data/microblog.html

[2] https://www.kaggle.com/datasets/lakritidis/product-clustering-matching-classification

TABLE II
CLUSTERING ALGORITHMS AND HYPER-PARAMETER SETTING

Clustering Algorithm	Hyper-parameters
k-means	Number of clusters: Actual. Max iterations: 200. Centroid initialization: k-means++.
MiniBatch k-means	Number of clusters: Actual. Max iterations: 200. Centroid initialization: k-means++. Batch size: 1024.
BIRCH	Number of clusters: Actual. Cluster radius threshold: 0.5. Max number of clusters in a node: 50.
Agglomerative Clustering	Number of clusters: Actual. Linkage: Complete. Distance measure: Euclidean.
Agglomerative (Ward)	Number of clusters: Actual. Linkage: Ward. Distance measure: Euclidean.
Spectral Clustering	Number of clusters: Actual. Affinity: RBF. γ: 1.0.
DBSCAN	ϵ: 0.5. Nearest Neighbors: 5. Distance measure: Euclidean.
OPTICS	ϵ: 0.5. Nearest Neighbors: 5. Distance measure: Euclidean.

Furthermore, TitleSet and SnippetSet comprise 11109 news headlines from 152 stories that were published on Google News. They have been used in numerous articles of the relevant literature for evaluating text clustering algorithms [22], [23]. The former includes only the news titles, whereas the latter includes excerpts of the main articles.

The largest dataset that we used is the well-known 20 Newsgroups, a traditional benchmark for text machine learning algorithms and NLP techniques. It includes 20 thousand news stories that can be categorized in 20 classes. The input vector space consists of approximately 10 thousand distinct words. Finally, the sixth dataset utilized in this study is Wines[3], a collection of 11258 descriptions for 88 wine varieties produced by 995 different wineries.

B. Clustering Algorithms

In this study, we examined the performance of eight clustering algorithms on the six aforementioned datasets. The algorithms were selected with the aim of representing the four most popular approaches to clustering; namely:

- *Space partitioning methods:* This category includes the popular *k-means* algorithm and another faster variant of it, *MiniBatch k-means* [7].
- *Hierarchical methods:* Here, we implemented three techniques: i) the traditional *Agglomerative* clustering with complete linkage, ii) an effective variant of Agglomerative clustering that applies the *Ward* method for record linkage, and iii) *BIRCH*, a memory efficient, online-learning algorithm for large-scale clustering [10].
- *Spectrum analysis methods*: The well-established *Spectral* clustering method was selected to represent this family. Initially, this approach employs a Gaussian (RBF) kernel function to construct the Laplacian matrix. Then, it performs eigen-decomposition of this matrix and it executes k-means to cluster the generated eigenvectors.
- *Density-based methods*: These algorithms are useful when the clusters are not convex shaped. *DBSCAN* [14] and *OPTICS* [16] have been examined from this category.

Table II enlists the eight algorithms that have been examined in this study. The second column reports the tuning of their respective hyper-parameters. Notice that these values were not the absolute optimal, but they have been set with the aim of delivering decent performance across all datasets.

[3]https://github.com/broepke/TextClustering

C. Performance Evaluation Measures

Three performance evaluation metrics were used to measure the effectiveness of the 8 aforementioned algorithms; specifically:

- *Adjusted Mutual Information (AMI):* The plain Mutual Information (MI) determines the similarity between two clusterings U and V as follows:

$$MI(U,V) = \sum_{i=1}^{|V|}\sum_{j=1}^{|U|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|}$$

  A weakness of MI is that it increases when the number of clusters is large, regardless of whether there is actually more information shared. AMI confronts this problem as indicated by the following formula:

$$AMI(U,V) = \frac{MI(U,V) - \mathbb{E}[MI(U,V)]}{(H(U)+H(V))/2 - \mathbb{E}[MI(U,V)]}$$

- *Adjusted Random Index (ARI):* The Random Index (RI) is defined as the ratio between the correctly grouped pairs divided by the total number of pairs, ignoring permutations:

$$RI(U,V) = \frac{|\text{Common pairs between } U \text{ and } V|}{|\text{Pairs in } U \text{ and } V|}$$

  ARI corrects RI for chance, by ensuring that its value will be equal to 0 if U and V are formed randomly, and equal to 1 if they are identical:

$$ARI(U,V) = \frac{RI(U,V) - \mathbb{E}[RI(U,V)]}{\max RI(U,V) - \mathbb{E}[RI(U,V)]}$$

  where $\mathbb{E}[RI(U,V)]$ denotes the Expected Random Index between U and V.
- *V-measure*, or *Normalized Mutual Information (NMI):* It combines the metrics of cluster completeness C and homogeneity G into a single formula:

$$NMI(U,V) = \mathcal{V} = \frac{2CG}{1+C+G}$$

  The completeness C of a cluster U is defined as the ability of an algorithm to place all the members of a class into same cluster. On the other hand, the homogeneity G of a cluster U indicates the purity of U; namely, the ability of algorithm to avoid placing elements from different classes into the same cluster.

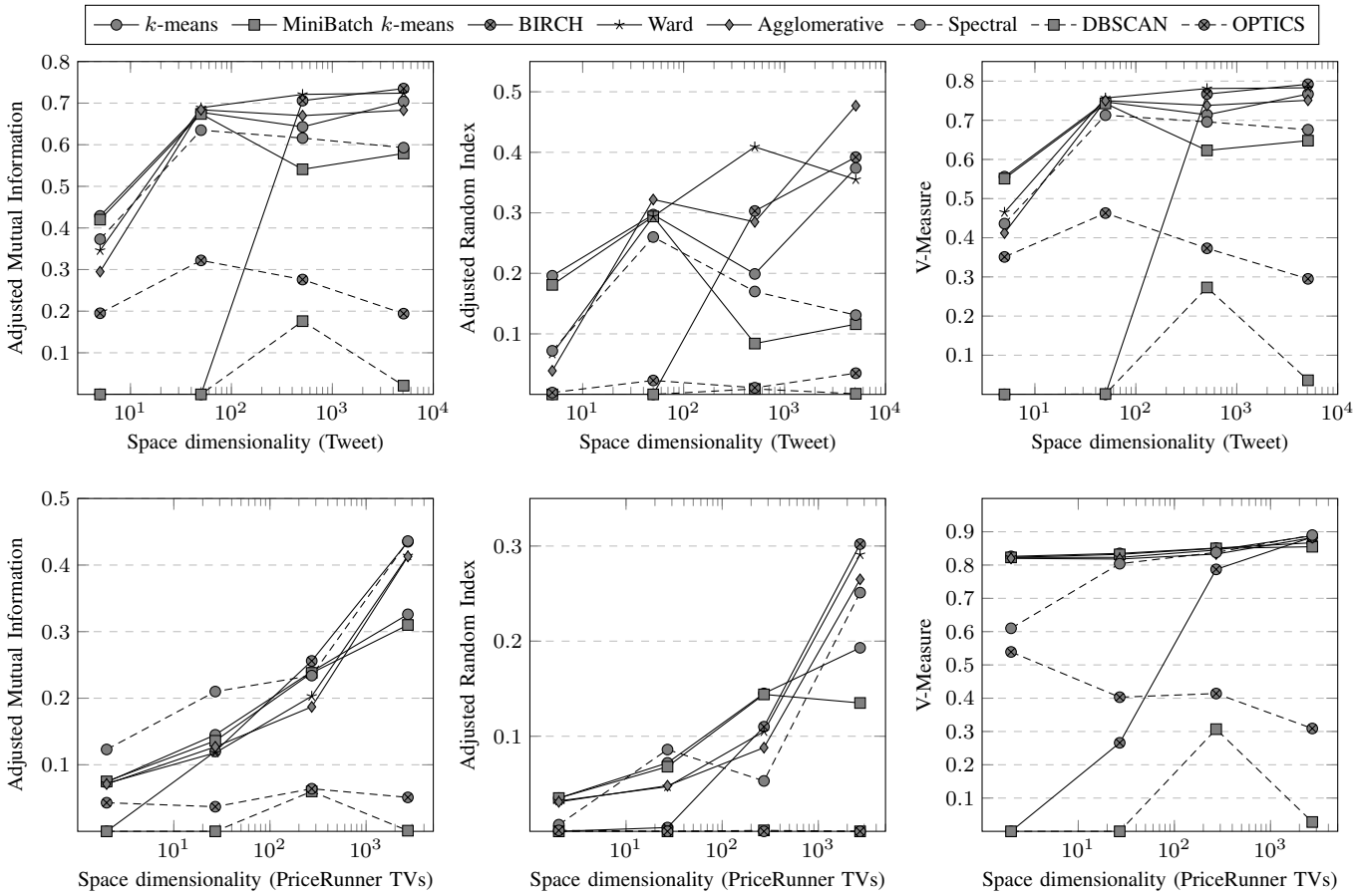

Fig. 1. Adjusted Mutual Info (left column), Adjusted Random Index (central column), and V-measure (right column) of the 8 clustering methods of Table II against input spaces of variable dimensionality (logarithmic scaling). The 3 diagrams at the top concern the Tweet dataset, whereas the 3 diagrams at the bottom concern PriceRunner TVs. The rightmost markers represent the original input spaces with all features included, i.e., without dimensionality reduction.

D. Results on Clustering Effectiveness

This subsection presents the results of our study on the impact of dimensionality reduction on text clustering performance. At first, we briefly describe the text preprocessing filters that were applied, and then we refer to the procedure for converting the raw text into numerical vectors.

Initially, a case folding filter was applied at the raw input text with the aim of transforming all the uppercase characters to lowercase. Then, a simple regular expression was formed to remove the punctuation symbols (dots, commas, etc.). After the documents were split to their component words, the popular tf-idf vectorization method was invoked to generate a collection of L2-normalized input vectors. Notice that tf-idf produces very long, high-dimensional and sparse vectors. The third column of Table I reports the size of the input dimensional space for each dataset.

The experiments were conducted by projecting the original input spaces onto other vector spaces of lower dimensionality. More specifically, for each dataset, we formed multiple target spaces that were one, two, three, etc. orders of magnitude smaller than the original ones. In all cases, the space reductions were carried out by employing the *Truncated Singular Value Decomposition (TSVD)* algorithm, a variant of PCA that can operate in the presence of sparse matrices.

The values of AMI, ARI, and V-measure of the eight clustering methods on the six datasets of Table I are illustrated in Figures 1, 2 and 3. In all 18 diagrams, the horizontal axes are plotted in logarithmic scale to conveniently illustrate the vector space sizes that differ by multiple orders of magnitude. Moreover, the rightmost markers represent the performance of the algorithms in the original input spaces, that is, without the application of dimensionality reduction. In all 3 figures the left, center, and right diagrams depict the values of AMI, ARI, and V-Measure, respectively.

The top diagrams of Figure 1 depict three comparative charts on the Tweet dataset. In the original input space with the 5076 dimensions, the most effective method in terms of AMI was BIRCH, with a score of 0.735. Ward and k-means followed, with values that were equal to 0.724 and 0.704, respectively. The top left diagram also illustrates that these three methods were affected by slight margins of 1–9% when the number of dimensions was decreased by one order of magnitude, i.e., 507. On the other hand, Spectral clustering and MiniBatch k-means were outperformed by the three aforementioned algorithms in both feature spaces.

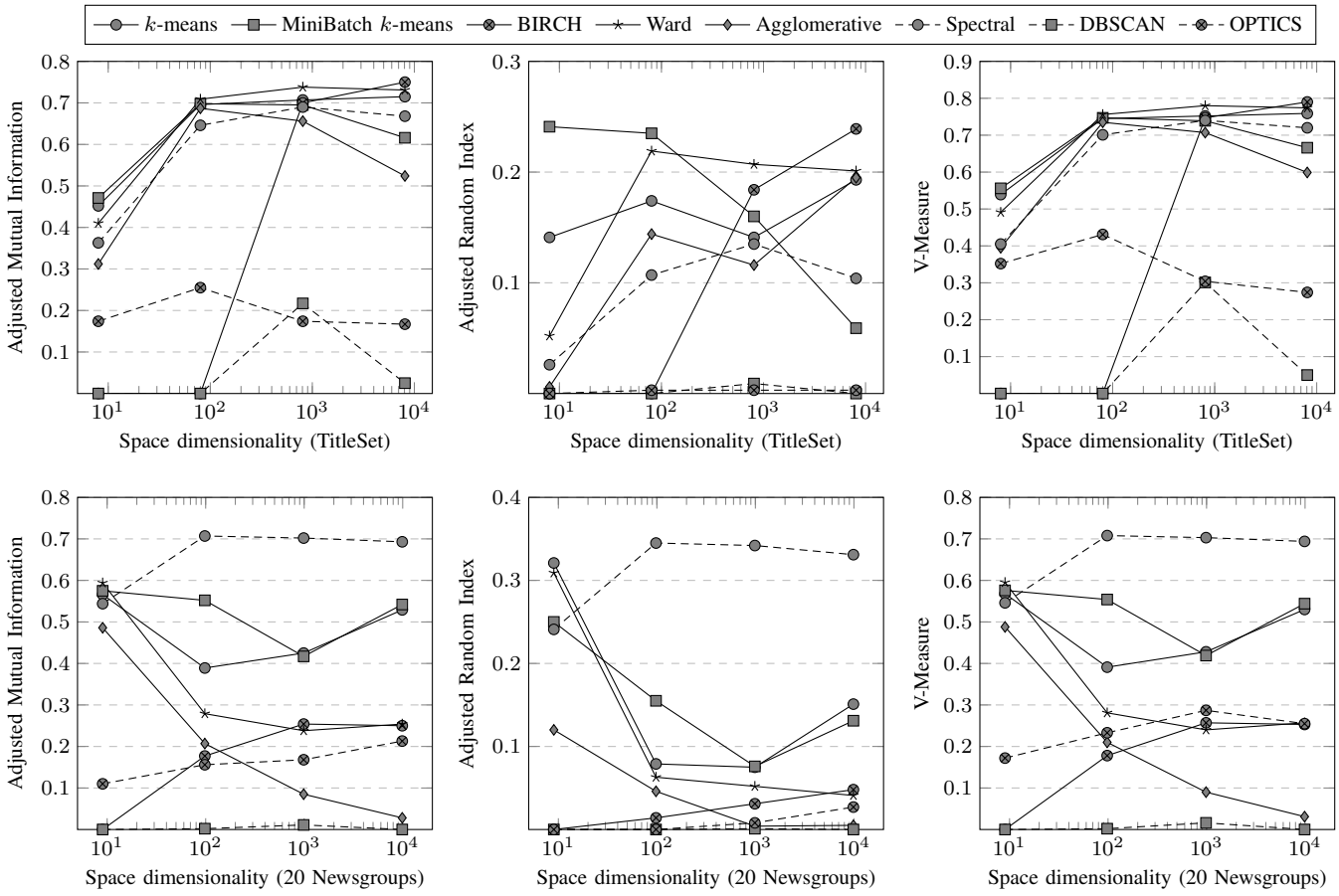

Fig. 2. Adjusted Mutual Info (left column), Adjusted Random Index (central column) and V-measure (right column) of the 8 clustering methods of Table II against input spaces of variable dimensionality (logarithmic scaling). The 3 diagrams at the top concern the TitleSet dataset, and the 3 diagrams at the bottom concern 20 Newsgroups. The rightmost markers represent the original input spaces with all features included, i.e., without dimensionality reduction.

Another interesting observation is that the density based methods, namely, DBSCAN and OPTICS, were benefited by dimensionality reduction, since their AMI increased when the size of the feature space was decreased by one order of magnitude. However, their performance was still very weak, and not comparable to the effectiveness of the other methods. All diagrams reveal that this observation does not hold for this case only and can be extended into the general conclusion that *the density based methods do not perform well in text clustering applications*. For this reason, and due to space restrictions, we shall not comment further on them.

For more aggressive reductions, that is, by two orders of magnitude, the AMIs of most methods degraded further. Therefore, k-means and Ward dropped by roughly 4% and 5% to 0.68 and 0.69, respectively. Interestingly, the performance of BIRCH, which was the most effective method of the two previous cases, dropped to absolute 0 here. A careful inspection of all 18 diagrams leads to a second conclusion: *BIRCH does not tolerate an excessive dimensionality reduction of the feature space by more than one orders of magnitude*. For a reduction by 3 orders of magnitude, only 5 features are left in the dataset, Therefore, the accuracy of all methods drops below acceptable levels.

Regarding the other two evaluation measures, the top right diagram of Figure 1 that illustrates the V-Measure values is very similar to the diagram of AMI. Regarding ARI, the ranking of the algorithms was different. Agglomerative was the most successful technique, as it achieved $ARI = 0.477$ and outperformed the other methods by a significant margin. It was followed by the 3 aforementioned methods, namely, BIRCH, k-means, and Ward. In the reduced feature space with the 507 dimensions, the accuracy of Ward was increased, making Ward the best performing approach.

The three bottom diagrams of Figure 1 illustrate the clustering effectiveness of the algorithms on the PriceRunner TVs dataset. The performance of the three hierarchical methods was also among the best. In the original feature space (2720 features), the value of AMI for BIRCH and Spectral clustering was 0.435. Ward and Agglomerative clustering were the other two methods with AMIs greater than 0.4. In this dataset, even the smallest reduction in number of features led to significant degradation of the accuracy of almost all algorithms. Contrary to the Tweets dataset, Ward and Agglomerative were among the most affected methods, and they were outperformed by the space partitioning approaches –k-means and MiniBatch k-means– in terms of both AMI and ARI.

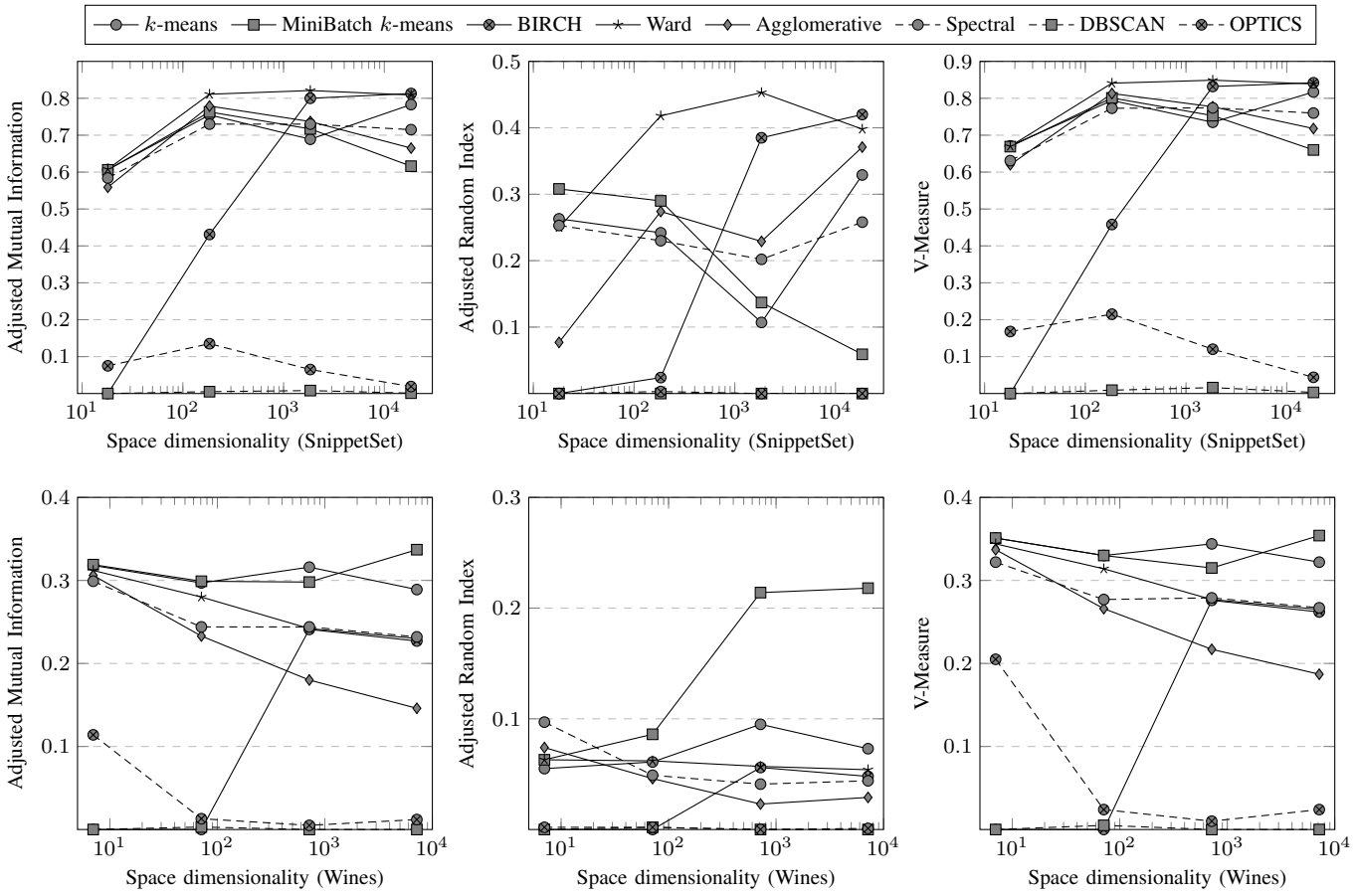

Fig. 3. Adjusted Mutual Info (left column), Adjusted Random Index (central column) and V-measure (right column) of the 8 clustering methods of Table II against input spaces of variable dimensionality (logarithmic scaling). The 3 diagrams at the top concern the SnippetSet dataset, and the 3 diagrams at the bottom concern Wines. The rightmost markers represent the original input spaces with all features included, i.e., without dimensionality reduction.

The 6 diagrams of Figure 2 depict the clustering performance of the algorithms in the TitleSet (top) and 20 Newsgroups (bottom) datasets. For TitleSet, the shapes of the curves are very similar to those that were observed in the Tweets dataset. According to all 3 evaluation measures, the hierarchical methods BIRCH and Ward were again the most accurate on the original feature space. Regarding the Agglomerative approach, the values of AMI and V-measure were considerably lower, rendering this method less accurate than k-means and Spectral clustering.

Interestingly, for most methods, the reduction of the dimensional space by one or two orders of magnitude resulted in no, or very slight degradations of AMI and V-measure. As mentioned earlier, BIRCH is the most significant exception, since its accuracy is quickly nullified as the number of features decreases. Similarly to the Tweets dataset, the ARI measure provides a diverse impression. Thus, it seems that MiniBatch and Spectral clustering work better with fewer features.

Moreover, in the 20 Newsgroups dataset, Spectral clustering outperformed all its adversaries in all three measurements and all dimensional spaces, apart from the smallest one (i.e., reduction by three orders of magnitude). Remarkably, the method's accuracy in this test was not affected by dimensionality reduction. Overall, apart from the second dataset, it was revealed that *Spectral clustering is robust to reductions of the feature space by one or two orders of magnitude*.

The performance of the hierarchical methods was rather problematic on this test. Furthermore, Ward and Agglomerative achieved far better results on the smallest input spaces which included just 9 features. For example, on the original feature space, the ARI of Ward was 0.254, whereas on the smallest space it was boosted to 0.594 which is the highest among the other algorithms. These results indicate that *the impact of dimensionality reduction cannot be always predicted, since there are datasets where a portion of the algorithms perform better on reduced dimensional spaces*.

We continued the experiments by examining the performance of the 8 clustering algorithms on the last two datasets, namely, SnippetSet and Wines. The results are respectively depicted in the top and bottom diagrams of Figure 3. For SnippetSet, the ranking of the algorithms was very similar to the one of TitleSet. In addition, the larger corpus (compared to TitleSet where only the titles were available) benefited the accuracy of all algorithms for all three measures. Therefore, the hierarchical methods (i.e., BIRCH, Ward, Agglomerative) and Spectral clustering were particularly effective in all spaces.

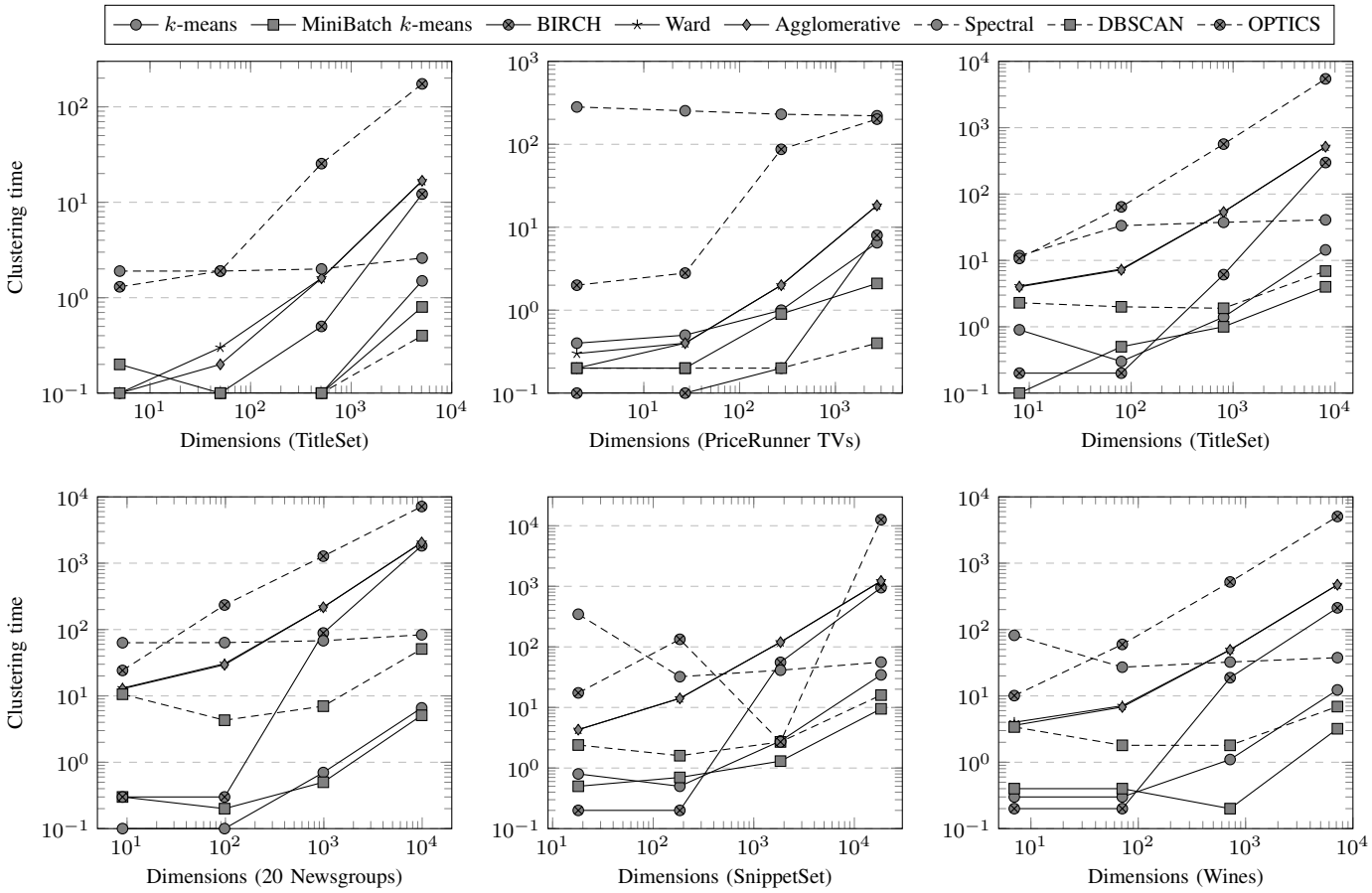

Fig. 4. Execution times of the 8 clustering methods of Table II against input spaces of variable dimensionality (logarithmic scaling). The rightmost markers represent the original input spaces with all features included, i.e., without dimensionality reduction.

With only a few exceptions (with BIRCH being the most obvious), the dimensionality reduction by two orders of magnitude led to small losses in performance. This was valid for the space partitioning techniques too. For a reduction by one order of magnitude, we recorded an interesting diversity in the behavior of k-Means vs. Minibatch k-Means: although the AMI of the former was decreased (from 0.783 in the original space, to 0.689 in the reduced space), the AMI of the latter was increased (from 0.616 in the original space, to 0.717 in the reduced space). Nevertheless, the conclusions that we derived from the four previous datasets were verified again in this test.

In the Wines dataset, the situation was reversed and the space partitioning methods were the most powerful. In contrast, the hierarchical methods, were clearly ineffective, since their performance was poor in this dataset. Interestingly, the aforementioned diversity in the behavior of k-Means and MiniBatch k-Means was also reversed. More specifically, for a reduction by one order of magnitude, the AMI of the former was increased (from 0.289 in the original space, to 0.316 in the reduced one), whereas the AMI of the latter was decreased (from 0.337 in the original space, to 0.298 in the reduced one).

For more aggressive reductions (namely, by two or more orders of magnitude), the rates at which the performance degraded were rather mixed. As expected, BIRCH was hugely affected; in the other hand, k-Means and Spectral clustering were particularly stable. Regarding the winner of this test, MiniBatch k-Means there was a moderate drop on its accuracy. For example, its AMI decreased from 0.337 in the original space to 0.299 in the reduced one. On the other hand, its ARI was almost vanquished from 0.218 in the original space to 0.086 in the reduced one.

E. Results on Clustering Duration

Finally, we also conducted a study on the running times of all algorithms in all datasets and all dimensional spaces. The results are illustrated in the six diagrams of Figure 4. Each diagram represents a dataset. Both the vertical and the horizontal axes are plotted in logarithmic scale to conveniently illustrate the large fluctuations of the durations on the various dimensional spaces.

The conclusion that derives from the inspection of these diagrams complies with the anticipated behavior: *dimensionality reduction has a positive impact on the running times of clustering algorithms*. However, *the execution acceleration is occasionally sublinear to the size of the input vector space*. In other words, a dimensionality reduction by one, two, etc. orders of magnitude is not always translated to a speed-up by one, two, etc. orders of magnitude. In some cases, the gains

are infinitesimal; for example, Spectral clustering on TitleSet. However, the aforementioned conclusion remains quite solid.

In all datasets, and in the original vector spaces the slowest algorithm was always OPTICS. Once again, its increased execution times combined with its ineffectiveness renders this algorithm inappropriate. On the other hand, the other density-based method, DBSCAN was substantially faster.

As expected, the space partitioning k-Means and MiniBatch k-Means were the fastest methods, with the latter outperforming the former. These two methods were among the most benefited ones, since the acceleration rate was approximately linear to the size of the input vector space. Indicatively, in the original space of the 20 Newsgroups dataset with the 9887 features, k-Means and MiniBatch k-Means consumed roughly 6.6 and 5.1 seconds respectively to complete their task. This time dropped by about 10 times (0.7 and 0.5 seconds, respectively) for a dimensionality reduction by 10X.

The hierarchical methods were slower than the space partitioning ones; Ward and Agglomerative clustering had almost equal running times and they were both slightly slower than BIRCH. Their speed-up was almost linear to the reduction in the size of the input space. In contrast, *Spectral clustering was not particularly benefited from dimensionality reduction*.

IV. Conclusions and Future Work

In this paper, we presented an experimental study on the impact of dimensionality reduction in text text clustering applications. By employing 6 well-established datasets, we created input vector spaces of various dimensionalities, and we measured the performance and running times of 8 dominant clustering algorithms. Throughout this investigation, we derived several useful conclusions that we summarize in the following list:

1) Regardless of the input vector space size, the density based methods (i.e. DBSCAN and OPTICS) did not perform well on text clustering tasks.
2) BIRCH does not tolerate excessive dimensionality reductions, that is, by more than one orders of magnitude.
3) In contrast, Spectral clustering is robust to reductions of the feature space by one or two orders of magnitude.
4) The impact of dimensionality reduction is sometimes unpredictable, since there are datasets where some algorithms perform better on reduced dimensional spaces.
5) Dimensionality reduction benefits the running times of all text clustering algorithms. In most cases, the smaller the dimensionality, the faster the clustering procedure is.

Our future plans for this work include the examination of the impact of several word and sentence embedding techniques in the effectiveness of the NLP clustering applications. More specifically, we intend to evaluate the performance of the recent state-of-the-art text clustering algorithms by employing both pre-trained and variable-size word embeddings.

References

[1] L. Akritidis, M. Alamaniotis, A. Fevgas, and P. Bozanis, "Confronting sparseness and high dimensionality in short text clustering via feature vector projections," in *Proceedings of the 32nd IEEE International Conference on Tools with Artificial Intelligence*, 2020, pp. 813–820.

[2] L. Akritidis, M. Alamaniotis, A. Fevgas, P. Tsompanopoulou, and P. Bozanis, "Improving hierarchical short text clustering through dominant feature learning," *International Journal on Artificial Intelligence Tools*, pp. 1–24, 2022.

[3] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to Information Retrieval*, 2008, vol. 39.

[4] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International conference on artificial neural networks*, 1997, pp. 583–588.

[5] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, no. 1-2, pp. 143–175, 2001.

[6] C. Jia, M. B. Carson, X. Wang, and J. Yu, "Concept decompositions for short text clustering by identifying word communities," *Pattern Recognition*, vol. 76, pp. 691–703, 2018.

[7] D. Sculley, "Web-scale k-means clustering," in *Proceedings of the 19th International Conference on World Wide Web*, 2010, pp. 1177–1178.

[8] S. Miyamoto, R. Abe, Y. Endo, and J.-I. Takeshita, "Ward method of hierarchical clustering for non-Euclidean similarity measures," in *Proceedings of the 7th International Conference of Soft Computing and Pattern Recognition*, 2015, pp. 60–63.

[9] S. Sharma, N. Batra *et al.*, "Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering," in *Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing*, 2019, pp. 568–573.

[10] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," *ACM SIGMOD record*, vol. 25, no. 2, pp. 103–114, 1996.

[11] B. Lorbeer, A. Kosareva, B. Deva, D. Softić, P. Ruppel, and A. Küpper, "Variations on the clustering algorithm BIRCH," *Big Data Research*, vol. 11, pp. 44–53, 2018.

[12] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Adv. in Neural Information Process Systems*, vol. 14, 2001.

[13] H. Jia, S. Ding, X. Xu, and R. Nie, "The latest research progress on spectral clustering," *Neural Computing and Applications*, vol. 24, no. 7, pp. 1477–1486, 2014.

[14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, vol. 96, no. 34, 1996, pp. 226–231.

[15] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems*, vol. 42, no. 3, pp. 1–21, 2017.

[16] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering points to identify the clustering structure," *ACM SIGMOD Record*, vol. 28, no. 2, pp. 49–60, 1999.

[17] L. Akritidis and P. Bozanis, "Effective unsupervised matching of product titles with k-combinations and permutations," in *Proceedings of the 14th IEEE International Conference on Innovations in Intelligent Systems and Applications*, 2018, pp. 1–10.

[18] L. Akritidis, A. Fevgas, P. Bozanis, and C. Makris, "A self-verifying clustering approach to unsupervised matching of product titles," *Artificial Intelligence Review*, vol. 53, no. 7, pp. 4777–4820, 2020.

[19] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 245–250.

[20] L. Shyamasundar and P. J. Rani, "Twitter sentiment analysis with different feature extractors and dimensionality reduction using supervised learning algorithms," in *Proceedings of the 2016 IEEE Annual India Conference*, 2016, pp. 1–6.

[21] K. Kim, "An improved semi-supervised dimensionality reduction using feature weighting: Application to sentiment analysis," *Expert Systems with Applications*, vol. 109, pp. 49–65, 2018.

[22] J. Yin and J. Wang, "A Dirichlet Multinomial Mixture model-based approach for short text clustering," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 233–242.

[23] ——, "A model-based approach for text clustering with outlier detection," in *Proceedings of the 32nd IEEE International Conference on Data Engineering*, 2016, pp. 625–636.