An Empirical Configuration Study of a Common Document Clustering Pipeline

Anton Eklund, Dept. of Computing Science, Umeå University & Adlede, Sweden antone@cs.umu.se
Mona Forsman, Adlede, Umeå, Sweden mona.forsman@adlede.com
Frank Drewes, Dept. of Computing Science, Umeå University, Sweden drewes@cs.umu.se

Abstract Document clustering is frequently used in applications of natural language processing, e.g. to classify news articles or create topic models. In this paper, we study document clustering with the common clustering pipeline that includes vectorization with BERT or Doc2Vec, dimension reduction with PCA or UMAP, and clustering with K-Means or HDBSCAN. We discuss the interactions of the different components in the pipeline, parameter settings, and how to determine an appropriate number of dimensions. The results suggest that BERT embeddings combined with UMAP dimension reduction to no less than 15 dimensions provides a good basis for clustering, regardless of the specific clustering algorithm used. Moreover, while UMAP performed better than PCA in our experiments, tuning the UMAP settings showed little impact on the overall performance. Hence, we recommend configuring UMAP so as to optimize its time efficiency. According to our topic model evaluation, the combination of BERT and UMAP, also used in BERTopic, performs best. A topic model based on this pipeline typically benefits from a large number of clusters.

1 Introduction

Clustering is an important technique for mining, classifying, and structuring unlabeled text data in an unsupervised manner. Some use cases are the classification of news articles (Iulia-Maria et al., 2020; Radu et al., 2020), social media analysis (Curiskis et al., 2020; Asyaky and Mandala, 2021), and topic modeling (Sia et al., 2020; Churchill and Singh, 2022; Zhang et al., 2022; Zhao et al., 2021). For topic modeling and document classification, practitioners typically use a defacto standard document clustering pipeline: document vectorization → dimension reduction → clustering; see Figure 1. This pipeline is attractive since it is straightforward to understand and provides flexibility due to its modularity. A popular application of this pipeline is BERTopic (Grootendorst, 2022), which converts the pipeline into a topic model by adding a topic keyword extractor.

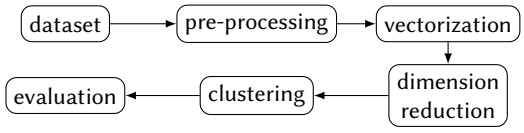

Figure 1: Clustering pipeline overview. The main parts are vectorization, dimension reduction, and clustering.

Since the pipeline components can be chosen from among many algorithms, and those usually depend on multiple parameter settings, it is challenging to analytically determine the best choice of components and their parameters, and the result depends on the concrete application. Further, the effect of the number of dimensions to reduce the vector space to is understudied in research on document clustering. In this paper, we conduct a systematic empirical study of how common embedding techniques, dimension reduction techniques, and clustering algorithms interact. From this, we derive recommendations that, as we hope, can guide practitioners who need to find a suitable configuration for clustering collections of unlabeled documents.

The first component of modern document clustering pipelines usually turns documents into numeric representations, called embeddings. Statistical methods such as Bag-of-Words or TF-IDF (Sammut and Webb, 2010) have been studied as part of topic models created with such a pipeline (Truică et al., 2016), but have nowadays become replaced by neural methods such as Doc2Vec (Le and Mikolov, 2014) and Google's Transformer-based BERT (Devlin et al., 2019), which outperform the older methods; see Curiskis et al. (2020) and Radu et al. (2020) for the former, and Subakti et al. (2022) for the latter.

The next step, dimension reduction, is added to

avoid degrading performance of clustering algorithms in high-dimensional vector spaces (Steinbach et al., 2004; Zimek, 2014)[1]. We study how the reduction to a range of different dimensions affects the quality of the resulting clusterings. There are two major classes of dimension reduction algorithms, those based on matrix factorization, and those based on neighbor graphs. Principle Component Analysis (PCA, by Pearson (1901); Hotelling (1933)) is a well-known and widely used example of the former. The latter, graph-based methods such as UMAP, calculate neighbor relations between points in the vector space, and then project them to a lower dimension, trying to preserve the neighbor relation. UMAP, invented by McInnes et al. (2018), is based on differential geometry and benefits from a solid mathematical foundation. UMAP has many applications, such as bioinformatics (Becht et al., 2019), material sciences (Li et al., 2019) and machine learning (Ordun et al., 2020; Sainburg et al., 2021).

The final step is clustering in the dimension-reduced vector space. In our work, we focus on distance-based clustering algorithms, where the similarity of objects is determined by their distance in the vector space. The clustering literature is extensive (Aggarwal and Zhai, 2012). Among the most popular approaches are algorithms based on determining cluster centroids (K-Means (Lloyd, 1982), K-Medoids (Kaufman and Rousseeuw, 1990)), calculating local density (DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999), HDBSCAN (Campello et al., 2013; McInnes and Healy, 2017)), computing spectral distributions (SPECTRAL (Ng et al., 2001)), or performing a hierarchical analysis (BIRCH (Zhang et al., 1996), Affinity Propagation (Frey and Dueck, 2007), Mean-Shift (Fukunaga and Hostetler, 1975; Cheng, 1995)). Centroid-based algorithms calculate the distances to cluster centroids to determine which point a cluster should be assigned to. In this work, we use K-means as a representative of this family since it is a widely used algorithm in multivariate data analysis. Density-based algorithms group data points that are in high-concentration areas of the vector space into clusters, with sparser regions in between. DBSCAN is one widely used density-based algorithm, with HDBSCAN being a hierarchical extension. In a comparison of DBSCAN and HDBSCAN for clustering news articles represented by Doc2Vec vectors, Radu et al. (2020) found both to be viable. We use HDBSCAN in this work due to its popularity in text clustering and because it is the default clustering algorithm in BERTopic (Grootendorst, 2022), the popular topic model based on the instance BERT → UMAP → HDBSCAN of the pipeline studied here.

[1]The term *curse of dimensionality* was coined by Bellman (2003), originally published as (Bellman, 1957), to refer to the algorithmic disadvantages of a high-dimensional vector space.

Looking at the literature on document clustering, we observe the following in particular:

(a) Doc2Vec and BERT have been extensively compared with TF-IDF as document representations for clustering, but not with each other;

(b) the use of dimension reduction in combination with document embeddings and clustering is an understudied method despite its popularity in practice;

(c) in particular, it is largely unknown how the performance of a clustering system for documents is affected by the number of dimensions of the embedding space.

To shed some light onto these questions, we have studied combinations of Doc2Vec, BERT, PCA, UMAP, K-Means, and HDBSCAN. The choice of these specific methods is motivated in Section 2.

We performed our experiments on collections of news articles because we expect news articles to belong to comparatively distinct topics and be grammatically correct. Moreover, there is a considerable practical demand for systems that can cluster collections of unlabeled news articles because maintaining a consistent tagging of articles even internally in a single publishing house is a significant problem, not to speak of multiple publishers.

We report on an extensive, systematic set of experiments with the task to broadly cluster three different labeled datasets of news articles using combinations of the above-mentioned embeddings and techniques with varying parameter settings, where the datasets are treated as unlabeled datasets and the gold labels are used for performance evaluation. To not only rely on a single quality measure, the quality of a clustering is assessed using both the Adjusted Rand Index (ARI, Hubert and Arabie (1985)) and the Adjusted Mutual Information Index (AMI, Vinh et al. (2010)). Additionally, to assess the intrinsic quality of the resulting topic models independently of the ground truth, a common measurement for topic coherence, c_v (Röder et al., 2015), is used. The performance of the pipeline as a topic model is evaluated by adding a topic keyword extractor to each pipeline setup, hence, converting them into BERTopic-style topic models.

The structure of the paper is as follows. Section 2 explains our method. Sections 3 and 4 present and discuss the results of our experiments, respectively. Finally, conclusions are presented in Section 5.

2 Method

To enable experiments with different configurations of the clustering pipeline while keeping the components

separate and individually adjustable, a test suite was designed and implemented. We use the following terminology to separate individual instances of the pipeline architecture of Figure 1 from its parameter settings:

Definition 1 A pipeline with specified vectorization, dimension reduction, and clustering components, but with unspecified parameter settings, is a *setup*. A setup with specific parameter settings is a *configuration*.

The datasets used for training and evaluation are presented in Section 2.1. In Section 2.2, the structure of our test suite is discussed. Section 2.3 explains how we compare and evaluate different configurations.

2.1 Datasets

Three datasets with different characteristics were used as test data. All three consist of news articles written in English. The datasets are fully labeled, meaning that there are no unlabeled articles.

SNACK – *Scraped News Articles Classified with Keywords* consists of publicly available news articles scraped from the Internet in 2021. Topic-related keyword lists were used for classifying the articles, using keywords extracted by a term-based method. Articles classified as Technology (3156), Food/Drink (2246), Sports (2836), Stocks (2208), Conflicts (3086) and Movies/TV-series (2859) of more than 500 characters were used for our experiments. The classes were chosen because they are largely unrelated. Articles occurring in multiple classes were removed. Unfortunately, the corpus cannot be made available as we do not own the publication rights of the individual articles it consists of. However, the URLs can be provided upon request.

AG News by Zhang et al. (2015) contains 1 000 000 categorized articles. For our study, a subset consisting of 15 000 articles from each of the four categories Sports, Business, Science/Technology and World was used. This dataset was included to get a perspective on how configurations perform on a dataset consisting of a large number of comparatively short documents.

Reuters is based on the Thomson Reuters Text Research Collection (TRC2)[2] of 1 800 370 articles from 2008 and 2009. 578 712 of the articles are tagged with keywords. Using the keywords Market Bonds (2738), Environment (515), Natural disaster (777), Soccer England (1974), Film (844), USA Politics (2559) and Auto (1678) as selectors, a dataset of 11 085 articles was extracted for our experiments.

Name	Articles	Classes	Words
SNACK	16 391	6	7 509 853
AG News	60 000	4	4 520 259
REUTERS	11 085	7	3 148 736

Table 1: The datasets used in this study along with their size statistics.

2.2 The Clustering Pipeline

The version of the clustering pipeline used in our test suite is shown in Figure 1. Recall from Definition 1 that every choice of specific components results in a *setup*, and additionally fixing the parameters of these components yields a *configuration*.

The documents to be clustered are loaded and enter pre-processing. The pre-processing depends on the embedding to be used. For Doc2vec, stopwords, punctuation, and special characters are removed. The WordNet Lemmatizer is used for lemmatizing as it has been shown to be superior to stemming in clustering tasks (Iulia-Maria et al., 2020). For BERT, we follow the cleaning and tokenization steps described by Devlin et al. (2019), using the HuggingFace[3] implementation.

Doc2Vec, in the Gensim implementation by Radim Řehůřek[4], was selected as a typical representative of classical prediction-based neural embeddings. The Doc2Vec training process was run for 15 epochs. BERT, as implemented by HuggingFace[3], was chosen as a representative of the Transformer-based class of embeddings. The BERT model was fine-tuned twice on all the sentences of the chosen dataset, using masked language modeling. The texts went through the preprocessing and vectorization only once to save time and to fairly compare the configuration parameters of the other modules.

For the vectorization phase of the pipeline, PCA was chosen to represent the class of matrix factorization techniques since it is widely applied in cases where dimension reduction is needed. UMAP (McInnes et al., 2018) represents the techniques based on neighbor graphs. It was chosen because it outperforms the popular t-SNE with respect to both efficiency and quality (see the original article). The numbers of dimensions tested are shown in Table 2. It is valuable to include reductions to as few as two or three dimensions since those are easy to visualize. As we will see, the effect of an increasing number of dimensions on the scores is not large. Hence, we opted to include comparatively few higher dimensions to reduce the computational resources needed.

[2] https://trec.nist.gov/data/reuters/reuters.html

[3] https://huggingface.co/

[4] https://radimrehurek.com/gensim/

Component	Technique	Settings	Value
Vectorization	Doc2Vec	dimensions	300
	BERT	dimensions	768
		layers	12
		attention heads	12
Dim. reduction	both	dimensions	[2, 3, 5, 7, 10, 15, 25, 50]
	PCA	principal components	equal to number of dimensions
	UMAP	*n_neighbors*	[5, 20, 80, 320, 1280, 2560]
Clustering	K-Means	k	$[6, 12, 24, 48, 96, 192, 384]^{\text{SNACK}}$
			$[4, 8, 16, 32, 64, 128, 256]^{\text{AG}}$
			$[7, 14, 28, 56, 112, 224, 448]^{\text{REUT}}$
	HDBSCAN	*min_cluster_size*	$[5, 10, 20, 40, 80, 160, 320, 640, 1280]^{\text{SNACK+REUT}}$
			$[10, 20, 40, 80, 160, 320, 640, 1280, 2560]^{\text{AG}}$

Table 2: Major pipeline components and their explored setting configurations.

To investigate the effect of dimension reduction with UMAP, the main setting we manipulate is the variable *n_neighbors*, which determines how many points in the vicinity of a given point should be used to measure local density. A low value makes UMAP focus on the local structure of the vector space whereas a high value emphasizes its global structure. The settings we explored can be found in Table 2.

After dimension reduction, the vectors are L2-normalized, a step which, for simplicity, is not shown in Figure 1. For count-based methods, this normalization is common practice, whereas for neural methods there does not appear to be a clear recommendation as to which approach to use. Initial experiments revealed it to be advantageous for the overall scores to normalize vectors and center them around the origin after dimension reduction so that the clustering algorithm works on normalized vectors. This step could also be performed prior to dimension reduction. Since our initial experiments revealed no significant difference between these options, we chose the former as it will ensure that the norm of all vectors is 1 when the actual clustering algorithm is invoked.

As clustering components, we selected the common K-Means and the more recent HDBSCAN. K-Means is mainly parameterized by the number k of clusters to partition the dataset into. HDBSCAN transforms the vector space based on the local density of the set of points to be clustered and then creates a minimum spanning tree over these points. From that tree, a hierarchy can be created and then converted to a flat structure depending on a parameter *min_cluster_size*. In the respective configurations, we consider a range of settings for the parameters k and *min_cluster_size* of K-Means and HDBSCAN, respectively, as specified in Table 2. Using a number k different from the number of distinct labels of the dataset allows K-Means to identify a high-quality clustering with a number of clusters that differs from that of the ground truth. In fact, considering larger values of k is essential when evaluating the system as a topic modeling system.

In contrast to K-Means, which labels all points in the vector space, HDBSCAN detects apparent noise which it then leaves unlabeled. Since our datasets are fully labeled and are thus considered to not contain any noise, this difference makes HDBSCAN suffer in the comparison. To avoid this effect, we use soft clustering for HDBSCAN, meaning that all points get a similarity score with respect to each cluster and are then assigned to the cluster that results in the highest score.

This pipeline is often used to build topic models. To be able to evaluate the pipeline as such, we need to convert each configuration to a topic model. This is done by adapting the c-TF-IDF of BERTopic (Grootendorst, 2022) and assigning top keywords for all clusters. The top 10 keywords are used to represent a cluster as a topic. Further mentions of topic modeling refer to configurations that have the addition of c-TF-IDF, hence, which are topic models in the style of BERTopic.

2.3 Evaluation

In our test suite, we ran the configurations shown in Table 2 to cluster the datasets, and evaluated the quality of the resulting clusterings. For evaluation, we used the gold labels of the datasets together with two different methods of measuring clustering quality: the pair-based measurement Adjusted Rand Index (ARI, Hubert and Arabie (1985)) and the Shannon-based Adjusted Mutual Information Index (AMI, Vinh et al. (2010)). These were chosen because they are widely used in practice and have complementary strengths (Romano et al., 2016): ARI is considered to be advantageous if the ground truth consists of big equal-sized clusters

whereas AMI is preferable when the dataset is unbalanced, containing both large and small clusters.

Scores range from 0 to 1 where 0 marks a random clustering and 1 a clustering that agrees perfectly with the ground truth. We consider a higher score to indicate better clustering even though scores are not comprehensive for all aspects of clustering quality.

In addition to measuring quality by means of comparison with the ground truth, we use the topic coherence measure c_v by Röder et al. (2015) to estimate the intrinsic clustering quality by calculating a score between 0 and 1. The conclusion of Röder et al. (2015) was that c_v is the topic coherence measure most correlated to human judgment. Our coherence calculations employ the window size of 110 also used in Röder et al. (2015). Since c_v is computed by calculating a coherence score for each individual cluster and aggregating the scores, it may favor large numbers of small clusters (one cluster with a low score does not impact the aggregated score as much). However, we found that clusterings with a large number of clusters are not assigned a much higher score than those with a smaller number of clusters. Thus, one can also use c_v to determine an appropriate number of clusters. Hence, we find c_v adequate for comparing the quality of clusterings resulting from different configurations.

2.4 Limitations

While the components whose interactions we study have been chosen to be both typical and representative of a wide range of components that practical clustering pipelines may be composed of, they can only be example instances as there are many other options. We have therefore made our test suite available for download[5]. Some design choices, explained above, were made to keep the project and in particular the number of experiments manageable.

Another limitation of this study lies in the choice of datasets used in the experiments. They all have relatively few categories (at most seven) and are all reasonably balanced. The largest imbalance is found in the Reuters dataset where the largest category, USA Politics (2559) is five times larger than the smallest category, Environment (515). There could be many situations where the datasets contain many more categories or have a more unbalanced ground truth. Thus, the results of this work provide approximate parameter values for practitioners to initiate configuring their own system but should not be taken as universal truths.

3 Results

The results are presented as a qualitative analysis with the 2D plots (Figures 2-5) in Section 3.1, and a quantitative analysis presented in Sections 3.2 and 3.3. Details on how the scores were attained and processed are described below.

Combining all the possible configurations that can be constructed from Table 2 yielded 1792 total combinations distributed over 8 setups. The experiments, described in Section 2, were conducted by running all configurations on each dataset. Each configuration was run three times to account for non-deterministic components such as K-Means, UMAP, and the coherence measure c_v. The mean ARI, AMI, and c_v of the three runs on each configuration are considered to be the final scores of the configuration in question. In the text, performance refers to these scores and a better performance is a higher score. Figure 6 shows aggregated results obtained by averaging the performance figures for all configurations of each setup. Since the number of dimensions of the clustering space has a major influence on the results, it is kept as the X-axis, thus giving rise to *trend plots* that describe trends depending on the number of dimensions.

3.1 Visualization in 2D

The 2D plots in Figure 2–5 visualize the vectorized document spaces reduced to two dimensions. While the plots cannot be translated directly into higher dimensions, one can qualitatively compare the vector spaces with the corresponding results in the trend plots. There are clear differences between the 2D vector spaces created with UMAP and PCA. By adding the label color, it is clearly visible that the UMAP reductions keep a more defined geometry of the data corresponding to the original labeling.

3.2 Aggregated Trends per Setup

The trend plots shown in Figure 6 are an attempt to provide a general view of how well the setups work and how this depends on the number of dimensions. In order to obtain a compact comparison of all setups we illustrate their trends for each dataset-metric pair, i.e., there is one figure for each pair. Each figure contains 8 trend lines, one per setup. Each aggregated trend line shows the mean score (vertical axis) over all parameter settings for the given dataset and metric, depending on the dimension setting (horizontal axis).

The aggregated trends in Figure 6 show that the mean score is less in 2D than in 5D and higher. For most setups, the score increases until somewhere between

[5] https://github.com/antoneklund/Systematic_Parameter_Search_News_Article_Clustering

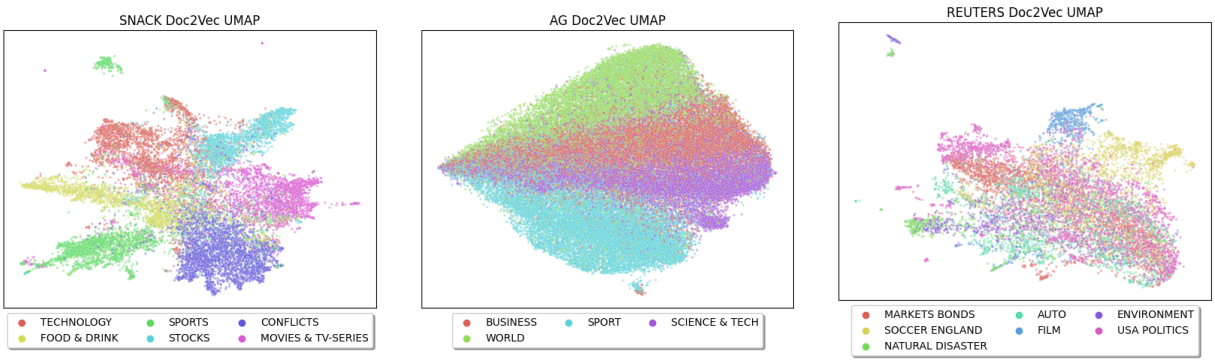

Figure 2: 2D UMAP reductions of Doc2Vec vectors.

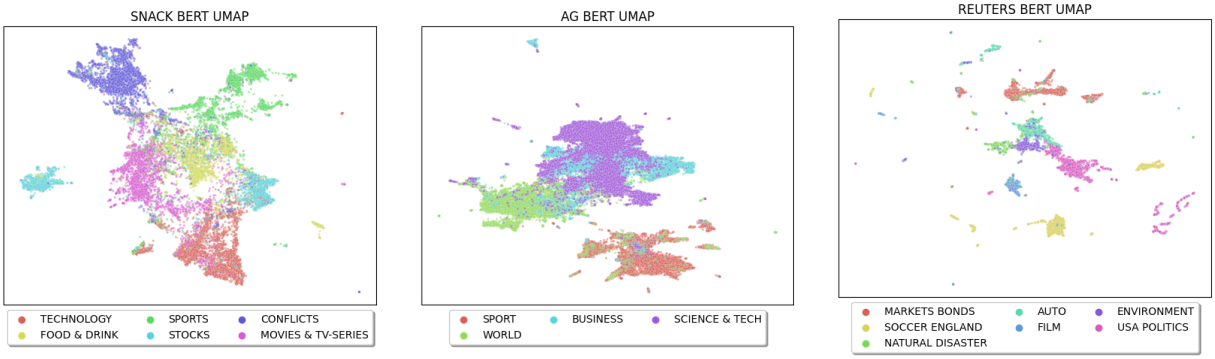

Figure 3: 2D UMAP reductions of BERT vectors.

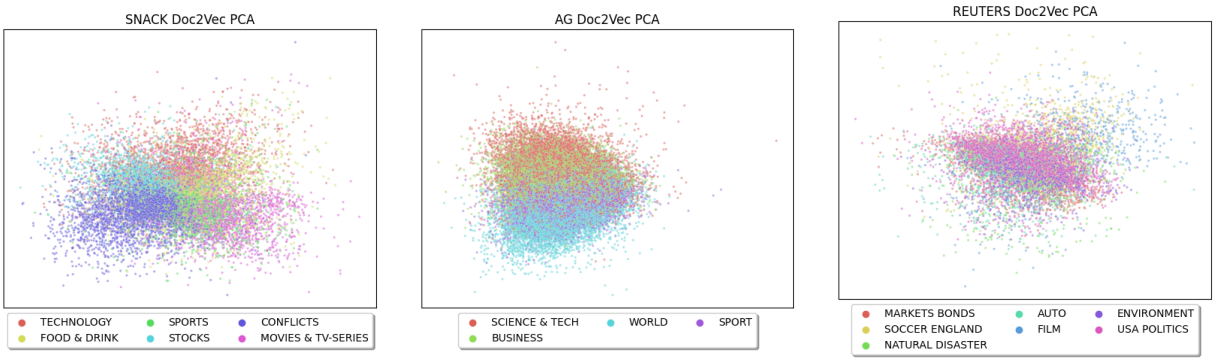

Figure 4: 2D PCA reductions of Doc2Vec vectors.

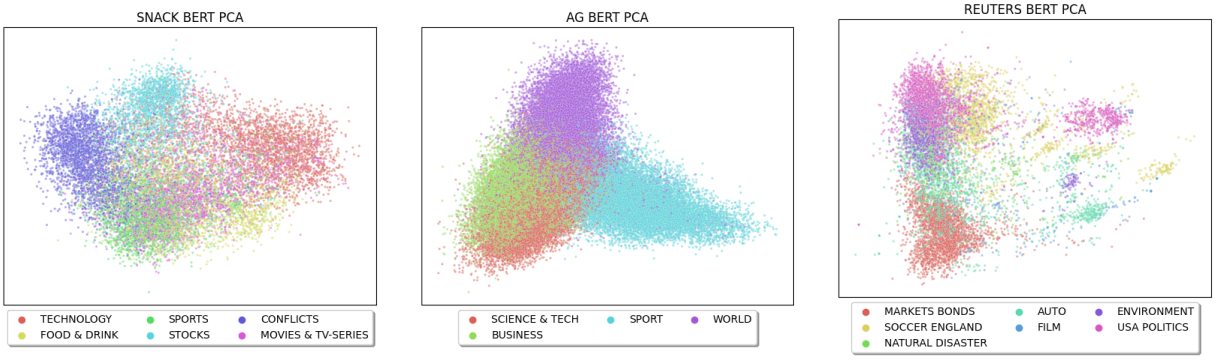

Figure 5: 2D PCA reductions of BERT vectors.

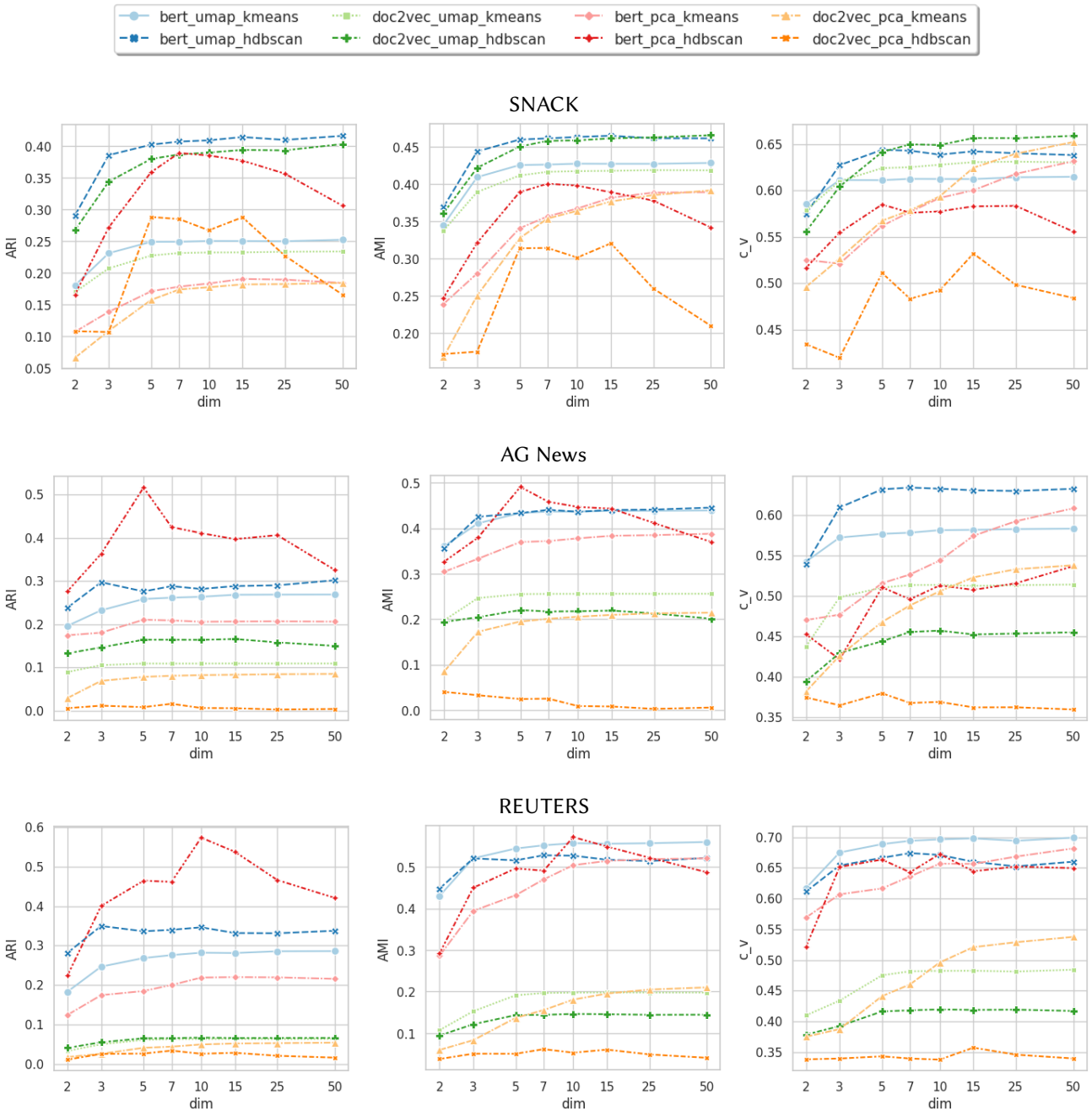

Figure 6: Aggregated trends for the SNACK (top), AG News (middle), and Reuters datasets (bottom). The evaluation metrics are ARI (left), AMI (middle), and c_v (right).

10D and 15D, after which there is no significant change. The exception is the setup bert_pca_hdbscan, which decreases after a peak in performance. (On SNACK, the setup bert_pca_kmeans is another exception, showing a similar behavior.)

The ARI and AMI scores for the setups does not indicate that more than 50D are needed. However, when looking at the c_v scores in Figure 6, there are setups (bert_pca_kmeans and doc2vec_pca_kmeans) that are still on a rising trend at 50D.

Some patterns re-occur across the different datasets. Setups that include BERT tend to perform better than those using Doc2Vec. This is true for AG News and Reuters but not for SNACK where the trends look similar for both vectorization methods. Also, more often than not setups that use UMAP seem to give rise to higher scores than the ones using PCA when the other two components are kept unchanged.

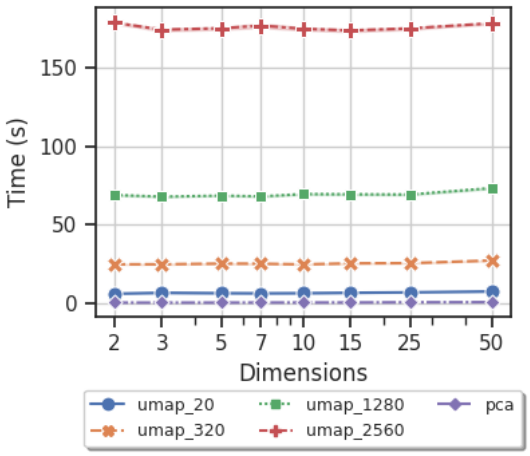

Figure 7: Dimension reduction mean time comparison over dimension for the Reuters dataset. UMAP with *n_neighbors* = 20 is around 6*s* and PCA is around 0.5*s*.

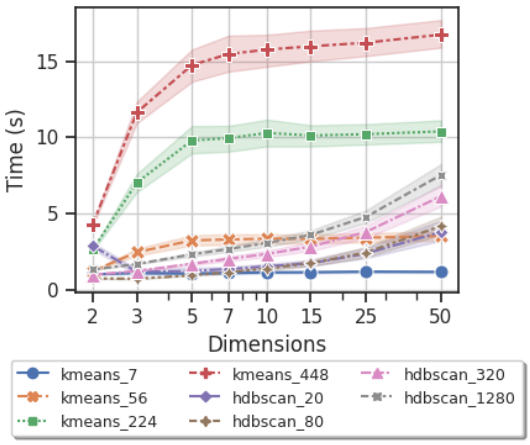

Figure 8: Clustering time comparison over dimension for the Reuters dataset.

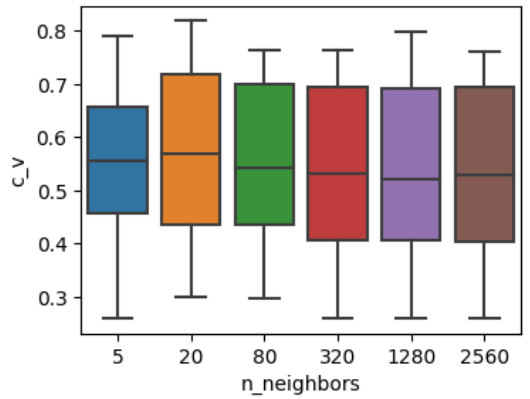

Figure 9: The different scores of c_v on the Reuters dataset depending on the UMAP variable *n_neighbors*.

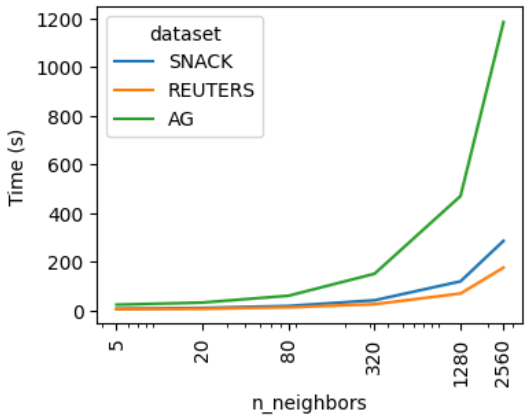

Figure 10: Comparison of time required to perform the UMAP dimension reduction depending on the variable *n_neighbors*.

3.3 Configurations

For the various configurations, there are large quantities of data that mostly tie into the individual datasets. Hence, showing all of them is not meaningful. Therefore, we present a sample of, as we hope, informative configurations in Tables 3–5. We chose the best configuration for each setup. Three tables are presented, one for each of the evaluation metrics ARI, AMI, and c_v. The best-performing configuration for a dataset is highlighted. This is most often bert_umap_hdbscan or bert_umap_kmeans but on the SNACK dataset, doc2vec_umap_hdbscan also achieves a high score.

In addition to these, we chose the Reuters dataset to show the relation between the number of clusters and the final score in Figure 11. The plots report all scores for ARI, AMI, and c_v divided into the different setups. For the Reuters dataset, the ground truth number of clusters is seven, and this is also where we find the highest scores for ARI and AMI. The topic modeling score c_v attains a higher value for a number of clusters larger than the ground truth.

The mean dimension reduction wall times for PCA and UMAP with different settings of the parameter *n_neighbors* are shown in Figure 7. PCA is faster than UMAP by a large margin. UMAP computation time increases significantly along with *n_neighbors*. However, the computation time is rather unaffected by increasing the number of dimensions from 2D to 50D.

The average c_v score per UMAP *n_neighbors* setting for the Reuters dataset is plotted in Figure 9. The boxes are similar, which means that the parameter has only a small impact on the score. The best-performing setting is *n_neighbors* = 20 where the mean score is slightly higher than for the other settings. Related to this is the dimension reduction time reported for different *n_neighbors* that are shown in Figure 10. It can be

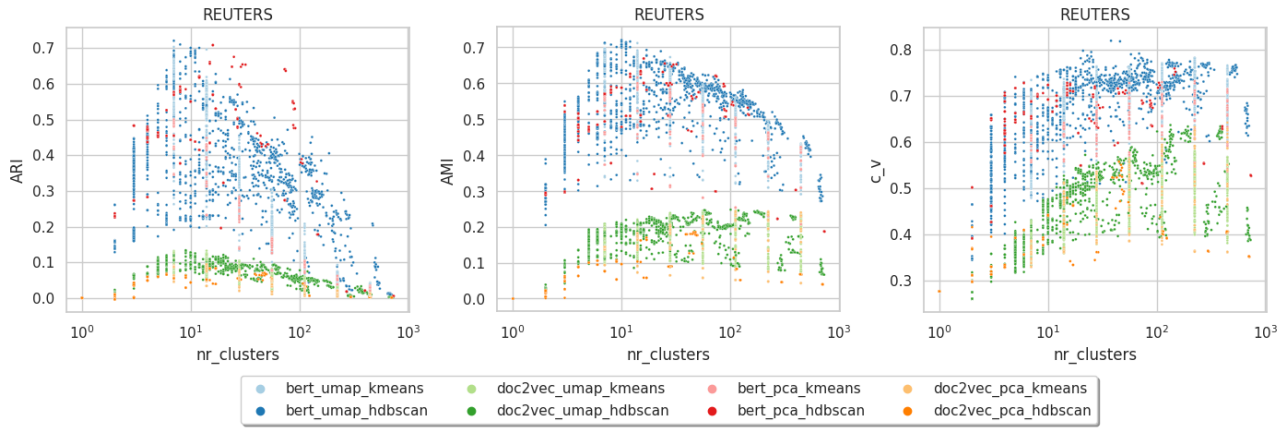

Figure 11: Relation between the number of clusters and the score for different metrics in the Reuters dataset.

Data	Setup	Dim	Alg. setting	Nr of clusters	Time	*ARI*
SNACK	bert_umap_kmeans	25	6	6	10.82	0.56
	bert_umap_hdbscan	15	320	7	20.27	**0.58**
	doc2vec_umap_kmeans	15	6	6	12.5	0.55
	doc2vec_umap_hdbscan	50	640	6	21.05	**0.58**
	bert_pca_kmeans	25	6	6	2.62	0.51
	bert_pca_hdbscan	15	160	6	6.37	0.50
	doc2vec_pca_kmeans	50	6	6	2.02	0.50
	doc2vec_pca_hdbscan	15	160	6	7.26	0.45
AG NEWS	**bert_umap_kmeans**	10	4	4	31.06	**0.67**
	bert_umap_hdbscan	50	2560	4	1288.69	0.59
	doc2vec_umap_kmeans	5	4	4	30.19	0.26
	doc2vec_umap_hdbscan	15	160	5	75.11	0.28
	bert_pca_kmeans	50	4	4	5.82	0.64
	bert_pca_hdbscan	5	2560	4	16.79	0.57
	doc2vec_pca_kmeans	50	4	4	3.73	0.17
	doc2vec_pca_hdbscan	3	80	5	4.29	0.09
REUTERS	bert_umap_kmeans	50	7	7	14.62	0.69
	bert_umap_hdbscan	25	160	10	14.47	**0.70**
	doc2vec_umap_kmeans	15	7	7	5.09	0.12
	doc2vec_umap_hdbscan	25	80	8	6.27	0.13
	bert_pca_kmeans	15	7	7	1.61	0.50
	bert_pca_hdbscan	15	20	16	2.33	0.69
	doc2vec_pca_kmeans	50	14	14	2.68	0.09
	doc2vec_pca_hdbscan	25	10	17	5.66	0.08

Table 3: A table of the best configuration according to ARI for each setup on each dataset. The column 'Alg. setting' reports the number k of clusters in K-Means and *min_cluster_size* in HDBSCAN.

Data	Setup	Dim	Alg. setting	Nr of clusters	Time	*AMI*
SNACK	bert_umap_kmeans	25	6	6	10.81	0.54
	bert_umap_hdbscan	15	320	7	20.27	**0.55**
	doc2vec_umap_kmeans	15	6	6	11.33	0.54
	doc2vec_umap_hdbscan	25	640	6	16.55	**0.55**
	bert_pca_kmeans	25	6	6	2.62	0.51
	bert_pca_hdbscan	15	160	6	6.37	0.49
	doc2vec_pca_kmeans	50	6	6	2.02	0.49
	doc2vec_pca_hdbscan	7	160	6	3.23	0.45
AG NEWS	**bert_umap_kmeans**	10	4	4	31.06	**0.64**
	bert_umap_hdbscan	25	2560	3	68.6	0.63
	doc2vec_umap_kmeans	3	8	8	30.26	0.31
	doc2vec_umap_hdbscan	15	160	5	75.11	0.31
	bert_pca_kmeans	50	4	4	5.82	0.6
	bert_pca_hdbscan	5	2560	4	16.79	0.54
	doc2vec_pca_kmeans	50	16	16	11.09	0.24
	doc2vec_pca_hdbscan	3	80	5	4.29	0.15
REUTERS	bert_umap_kmeans	50	7	7	14.62	0.69
	bert_umap_hdbscan	25	160	10	14.47	**0.71**
	doc2vec_umap_kmeans	10	56	56	8.1	0.24
	doc2vec_umap_hdbscan	10	10	62	8.1	0.24
	bert_pca_kmeans	25	14	14	2.18	0.6
	bert_pca_hdbscan	15	20	16	2.33	0.66
	doc2vec_pca_kmeans	50	224	224	10.27	0.24
	doc2vec_pca_hdbscan	25	5	47	6.29	0.18

Table 4: A table of the best configuration according to AMI for each setup on each dataset. The column 'Alg. setting' reports the number k of clusters in K-Means and *min_cluster_size* in HDBSCAN.

seen that the computation time increases with larger *n_neighbors* as well as with the size of the dataset.

4 Discussion

The purpose of this study has been to help practitioners limit the time spent on building a clustering system and tuning its hyperparameters. The following discussion is structured according to the three main degrees of freedom, namely the number of dimensions, the choice of components, and the parameter tuning.

4.1 Dimension

The number of dimensions of the clustering vector space is relevant for the clustering result. Too few dimensions will remove relevant information from the vector space, and too many dimensions may make the clustering drop in performance and become computationally inefficient. The difficulty lies in quantifying *too few* and *too many*. The results of this study show that performance typically increases from 2D to somewhere between 10D and 15D, where the increase stagnates. The expected performance drop in higher dimensions due to the curse of dimensionality does not seem

Data	Setup	Dim	Alg. setting	Nr of clusters	Time	c_v
SNACK	bert_umap_kmeans	50	96	96	13.01	0.67
	bert_umap_hdbscan	5	20	41	19.56	**0.73**
	doc2vec_umap_kmeans	15	12	12	7.00	0.68
	doc2vec_umap_hdbscan	15	40	17	20.98	0.71
	bert_pca_kmeans	50	96	96	11.31	0.65
	bert_pca_hdbscan	50	20	12	16.98	0.64
	doc2vec_pca_kmeans	50	96	96	9.63	0.68
	doc2vec_pca_hdbscan	50	5	105	26.69	0.67
AG NEWS	bert_umap_kmeans	50	246	256	82.13	0.67
	bert_umap_hdbscan	50	20	140	116.43	**0.73**
	doc2vec_umap_kmeans	25	128	128	87.21	0.57
	doc2vec_umap_hdbscan	5	10	107	52.09	0.60
	bert_pca_kmeans	50	64	64	31.78	0.67
	bert_pca_hdbscan	50	40	14	369.37	0.63
	doc2vec_pca_kmeans	50	128	128	62.03	0.61
	doc2vec_pca_hdbscan	5	10	38	9.59	0.5
REUTERS	bert_umap_kmeans	25	224	224	11.21	0.78
	bert_umap_hdbscan	15	20	63	8.74	**0.79**
	doc2vec_umap_kmeans	50	112	112	10.81	0.58
	doc2vec_umap_hdbscan	25	5	187	17.53	0.62
	bert_pca_kmeans	50	224	224	11.08	0.72
	bert_pca_hdbscan	50	5	147	12.44	0.75
	doc2vec_pca_kmeans	50	448	448	16.02	0.63
	doc2vec_pca_hdbscan	15	5	47	4.20	0.55

Table 5: A table of the best configuration according to c_v for each setup on each dataset. The column 'Alg. setting' reports the number k of clusters in K-Means and *min_cluster_size* in HDBSCAN.

to pose a significant problem for the range of dimensions tested in this article. Hence, for a system that has to perform well on unknown data, a reasonable initial guess would be to use 15D or (moderately) higher.

While a higher-dimensional vector space (within the range in this study) seems to ensure better performance, it has to be weighed against the resulting increase in computation time. As seen in Figure 7, the cost of performing the dimension reduction itself does not significantly depend on the number of dimensions. Instead, the most significant factor affecting the efficiency of the dimension reduction is the size of the dataset and (in the case of UMAP) the *n_neighbors* parameter as shown in Figure 10. As seen in Figure 8, the clustering times increase very slowly in higher dimensions. Still, the increase in clustering time indicates that the number of dimensions should be kept down if there is no significant performance gain.

We recommend attempting to find a balance between efficiency and desired performance. As previously mentioned, the performance increase tends to stagnate around 15D. Hence, as a rule of thumb, we recommend a reduction to a range from around 15D to 25D. Future work will need to be conducted to study the impact of a number of dimensions higher than 50D, which was the limit in this study.

4.2 Choice of Components

The trend plots in Figures 6 give insights into the component performance, and the 2D plots of the vector spaces in Figures 2–5 add a geometrical view of the results. From this, we draw the following observations.

4.2.1 Vectorization Method

The performance of setups that include BERT is better or similar to that of setups that include Doc2Vec when all other components are the same. We can also see that the highlighted best-performing configurations always include BERT in Tables 3–5. At best, Doc2Vec achieves on-par results with BERT on SNACK. This makes us conclude that BERT as a vectorization method is preferable over Doc2Vec, and we recommend using it in a clustering pipeline.

4.2.2 Dimension Reduction

For dimension reduction, the setups using UMAP yield more stable results, with the scores increasing until they stabilize at around 10D. PCA sometimes shows peaks in scores for configurations around 5D to 15D. However, the performance decreases in higher dimensions as seen in Figure 6. We note in Tables 3–5 that setups with UMAP achieve the top scores. Therefore, UMAP generally seems like a more stable recommendation. However, in this context, it is worth recalling a major advantage of PCA that is not highlighted in the experiments of this article. Namely, that the axes of its coordinate system correspond to the Eigenvectors computed in the course of matrix decomposition. As such, these axes carry a distinct mathematical meaning, which is important for explainability. A common application of this fact is to use the explained variance of the axes for analysis (Raunak et al., 2019).

For visualization purposes, the choice of UMAP is evident when comparing the 2D plots of UMAP in Figures 2 and 3, with PCA in Figures 4 and 5. The vector spaces for the UMAP-reduced datasets form clear clusters without mixing the categories. This result is expected as preserving cluster structure in lower dimensions is something that neighbor graph methods were designed to do.

Nevertheless, if explainability is not a major concern, it seems safe to conclude that UMAP as a component in the document clustering pipeline is preferable over PCA because of both performance stability and visualization properties. This is also supported in the literature by Allaoui et al. (2020). However, PCA performs well in certain configurations and is more efficient. PCA could therefore be preferable in situations where strict time constraints must be obeyed or it is important to be able to interpret the vector space axes.

4.2.3 Clustering Algorithm

Our results show that HDBSCAN generally performs well in combination with UMAP. K-Means also displays good performance but achieves slightly lower scores than HDBSCAN. First and foremost, performance tends to be determined by the other components and particularly the vectorization. It is therefore sensible to leave the choice of clustering algorithm to the practitioner who can visualize the vector space (preferably with UMAP) to obtain information about its shape and make an informed decision (Eklund and Forsman, 2022).

HDBSCAN combined with PCA is the only setup that sometimes exhibits a downward trend after a peak around 5D to 15D. This could signal that HDBSCAN is inferior to K-Means at handling the shrinking variance in distance that occurs in higher dimensions. However, this phenomenon does not occur in all setups involving HDBSCAN, meaning that the behavior cannot be caused by HDBSCAN alone. In fact, the peaks sometimes occur in the best-performing configuration for a dataset. Therefore, we cannot discourage combining PCA with HDBSCAN, but we do advise caution when using this combination.

The rightmost plots (with the metric c_v) in Figure 6 show that a topic model could be successfully created with any combination of UMAP or PCA, and HDBSCAN or K-Means. The performance is again mostly dependent on the vectorization. Furthermore, while the performance increase often seems to stagnate in higher dimensions, setups with PCA and K-Means keep improving. This indicates that increasing the number of dimensions beyond what was done in this study may eventually turn PCA and K-Means into the best-performing combination.

4.3 Parameter Tuning

Parameter tuning is the task most dependent on the dataset. However, being able to trust that the system is well configured is especially important when facing unseen data, and thus when tuning is most difficult. Tables 3–5 contain the best-performing configurations for each setup. These tables give some ideas of what is important when choosing a parameter setting.

One central aspect appears to be the number of clusters. For ARI (Table 3) and AMI (Table 4), it is clear that if the clustering produces a number of clusters closer to the number of gold labels, then the score will be higher. Where this fails, such as setups involving Doc2Vec for the Reuters dataset in Table 4, is when the score is so low that the setup should be discarded no matter the configuration. The recommendation that the number of clusters should stay close to the number of gold labels is also supported by Figure 11, where the highest scores are obtained by values around seven for *nr_clusters*. In a real-world environment, it could of course be difficult to make practical use of this observation because the "real" number of clusters may not a priori be known. Strategies exist for finding an optimal number of clusters for a dataset that can be used to set the parameter k for K-Means (Kodinariya et al., 2013). In this regard, an advantage of HDBSCAN is that *min_cluster_size* is related to the dataset size, which is usually known.

The metric c_v used to evaluate topic modeling systems favors pipeline configurations that result in a larger number of clusters than the coarse categorization of the annotated data; see Table 5. This is also indicated by the large values for both c_v and *nr_clusters* in the rightmost plot in Figure 11. Some benefits of using smaller values of *min_cluster_size*, which yield a larger number of clusters, have been suggested for topic modeling of short social media texts (Asyaky and Mandala, 2021). Our results let us agree with this recommendation for longer news article texts as well. Overall, the clustering algorithms show comparative performances when applied to the same vector space. Hence, there does not seem to be any harm in choosing the algorithm based on domain and application knowledge.

Computational efficiency is an aspect practitioners may have to take into account. UMAP takes considerably longer time to compute than PCA as shown in Figure 7 and supported by the benchmarking comparison found in the UMAP documentation[6]. UMAP complexity is bound by the calculation of *n_neighbors* and has empirically been shown to be $O(N^{1.14})$ (McInnes et al., 2018). Our results support this by showing a wall time that essentially increases linearly, as shown in Figure 10. From Figure 9 we can also see that *n_neighbors* in general has a low impact on the overall scores. If there are any patterns, it is that smaller values of *n_neighbors* are more frequently present in the best-scoring configurations. In UMAP, the parameter *n_neighbors* is supposed to weigh retaining the local structure against retaining the global structure of the data (smaller vs. larger *n_neighbors*, respectively). Judging from the results in this study, we presume that it is better to focus on preserving the local structure, as also supported in Asyaky and Mandala (2021).

In conclusion, the choice of parameters should be based on how many clusters one expects to find, weighed against any efficiency constraints the system may have. There are indications that the UMAP parameter *n_neighbors* should be chosen with a lower value to preserve the local structure of the data when working with document embeddings.

[6] https://umap-learn.readthedocs.io/en/latest/benchmarking.html

5 Conclusions

After systematically studying different setups of vectorization, dimension reduction, and clustering together with a large number of parameter settings, we conclude that the vectorization component has the most significant impact on the performance of the system and that BERT usually results in a better embedding space for clustering than Doc2Vec. When reducing the vector space, vectors should not be reduced to less than 15D. UMAP most frequently exhibits better performance and visualization capabilities than PCA. However, PCA can be favored if computational efficiency or explainability is required. The clustering algorithms perform roughly on par with each other but with a slight advantage to HDBSCAN over K-Means. The choice of a clustering algorithm ultimately comes down to knowledge about the dataset and application domain. Influencing that choice, and all the parameters of the setup, are mainly the computation time and the number of clusters that the data shall be divided into.

The popularity of the practical pipeline for document clustering and topic modeling studied in this paper is unlikely to decrease in the near future. With this in mind, we think that additional work aiming to evaluate and improve such systems is required. This study used labeled data to assess the performance of different setups and configurations. While we were able to draw a number of general rule-of-thumb conclusions that will hopefully benefit the practitioner, there is no getting around the fact that, ultimately, a lot of domain knowledge is required in concrete practical scenarios. The use of automatic measurements, as done in this study, can be one way of coming up with reasonable settings. However, we believe that such methods have intrinsic limitations in contexts whose end users are humans, e.g., consumers of news articles or readers of online advertisements. In such cases, we believe it to be necessary to complement automatic assessments of the quality of clusterings or topic models by systematic methods based on human judgment. How this can be done in a qualified manner with reasonable budgets appears to be an open question that deserves the focus of future research.

Acknowledgment

We thank the reviewers for their thorough reading of the initial manuscript and their insightful comments which have been useful in revising this paper.

References

Charu C. Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 77–128. Springer US, Boston, MA.

Mebarka Allaoui, Mohammed Lamine Kherfi, and Abdelhakim Cheriet. 2020. Considerably improving clustering algorithms using umap dimensionality reduction technique: A comparative study. In Abderrahim El Moataz, Driss Mammass, Alamin Mansouri, and Fathallah Nouboud, editors, *Image and Signal Processing*, pages 317–325. Springer International Publishing, Cham.

Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. Optics: Ordering points to identify the clustering structure. *ACM Sigmod record*, 28(2):49–60.

Muhammad Sidik Asyaky and Rila Mandala. 2021. Improving the performance of hdbscan on short text clustering by using word embedding and umap. In *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–6.

Etienne Becht, Leland McInnes, John Healy, Charles Antoine Dutertre, Immanuel W.H. Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W. Newell. 2019. Dimensionality reduction for visualizing single-cell data using umap. *Nature Biotechnology*, 37:38–44.

Richard Ernest Bellman. 1957. *Dynamic Programming*. Princeton University Press.

Richard Ernest Bellman. 2003. *Dynamic Programming*. Dover Publications.

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yizong Cheng. 1995. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799.

Rob Churchill and Lisa Singh. 2022. The evolution of topic modeling. *ACM Comput. Surv.*, 54(10s).

Stephan A. Curiskis, Barry Drake, Thomas R. Osborn, and Paul J. Kennedy. 2020. An evaluation of document clustering and topic modelling in two online social networks: Twitter and reddit. *Information Processing & Management*, 57(2):102034.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Anton Eklund and Mona Forsman. 2022. Topic modeling by clustering language model embeddings: Human validation on an industry dataset. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 635–643, Abu Dhabi, UAE. Association for Computational Linguistics.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. 2nd Intl. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science*, 315(5814):972–976.

K. Fukunaga and L. Hostetler. 1975. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Harold Hotelling. 1933. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2:193–218.

Rădulescu Iulia-Maria, Ciprian-Octavian Truică, Elena Simona Apostol, Alexandru Boicea, Mariana Mocanu, Daniel-Călin Popeangă, and Florin Rădulescu. 2020. Density-based text clustering using document embeddings. In *Proceedings of the 36th International Business Information Management Association Conference (IBIMA)*.

Leonard Kaufman and Peter J. Rousseeuw. 1990. *Partitioning Around Medoids (Program PAM)*, chapter 2. John Wiley & Sons, Ltd.

Trupti M Kodinariya, Prashant R Makwana, et al. 2013. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China. PMLR.

Xin Li, Ondrej E. Dyck, Mark P. Oxley, Andrew R. Lupini, Leland McInnes, John Healy, Stephen Jesse, and Sergei V. Kalinin. 2019. Manifold learning of four-dimensional scanning transmission electron microscopy. *npj Computational Materials*, 5:5.

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Leland McInnes and John Healy. 2017. Accelerated hierarchical density based clustering. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

Andrew Ng, Michael Jordan, and Yair Weiss. 2001. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Catherine Ordun, Sanjay Purushotham, and Edward Raff. 2020. Exploratory analysis of covid-19 tweets using topic modeling, umap, and digraphs. *arXiv preprint arXiv:2005.03082*.

Karl Pearson. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2.

Robert George Radu, Iulia Maria Rădulescu, Ciprian Octavian Truică, Elena Simona Apostol, and Mariana Mocanu. 2020. Clustering documents using the document to vector model for dimensionality reduction. In *Proceedings of the 22nd IEEE International Conference on Automation, Quality and Testing, Robotics - THETA, AQTR 2020*.

Vikas Raunak, Vivek Gupta, and Florian Metze. 2019. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, page 399–408, New York, NY, USA. Association for Computing Machinery.

Simone Romano, Nguyen Xuan Vinh, James Bailey, and Karin Verspoor. 2016. Adjusting for chance clustering

comparison measures. *The Journal of Machine Learning Research*, 17(1):4635–4666.

Tim Sainburg, Leland McInnes, and Timothy Q. Gentner. 2021. Parametric umap embeddings for representation and semisupervised learning. *Neural Computation*, 33(11):2881–2907.

Claude Sammut and Geoffrey I. Webb, editors. 2010. *TF-IDF*. Springer US, Boston, MA.

Suzanna Sia, Ayush Dalmia, and Sabrina J. Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1728–1736, Online. Association for Computational Linguistics.

Michael Steinbach, Levent Ertöz, and Vipin Kumar. 2004. The challenges of clustering high dimensional data. In Luc T. Wille, editor, *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition*, pages 273–309. Springer Berlin Heidelberg, Berlin, Heidelberg.

Alvin Subakti, Hendri Murfi, and Nora Hariadi. 2022. The performance of bert as data representation of text clustering. *Journal of Big Data*, 9:15.

Ciprian-Octavian Truică, Florin Rădulescu, and Alexandru Boicea. 2016. Comparing different term weighting schemas for topic modeling. In *2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 307–310.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(95):2837–2854.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. Birch: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, page 103–114, New York, NY, USA. Association for Computing Machinery.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, volume 28, pages 649–657.

Zihan Zhang, Meng Fang, Ling Chen, and Mohammad Reza Namazi Rad. 2022. Is neural topic modelling better than clustering? an empirical study on clustering with contextual embeddings for topics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3886–3893, Seattle, United States. Association for Computational Linguistics.

He Zhao, Dinh Phung, Viet Huynh, Yuan Jin, Lan Du, and Wray Buntine. 2021. Topic modelling meets deep neural networks: A survey. *arXiv preprint arXiv:2103.00498*.

Arthur Zimek. 2014. Clustering high-dimensional data. In Charu C Aggarwal and Chandan K Reddy, editors, *Data clustering*, chapter 9, pages 202–229. Citeseer.