applied sciences

MDPI

Article

An Empirical Evaluation of Document Embeddings and Similarity Metrics for Scientific Articles

Joaquin Gómez [1] and Pere-Pau Vázquez [2,*]

[1] Department of Computer Science, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain; joaquin.gomez@estudiantat.upc.edu
[2] ViRVIG Group, Department of Computer Science, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain
* Correspondence: pere.pau.vazquez@upc.edu

Abstract: The comparison of documents—such as articles or patents search, bibliography recommendations systems, visualization of document collections, etc.—has a wide range of applications in several fields. One of the key tasks that such problems have in common is the evaluation of a similarity metric. Many such metrics have been proposed in the literature. Lately, deep learning techniques have gained a lot of popularity. However, it is difficult to analyze how those metrics perform against each other. In this paper, we present a systematic empirical evaluation of several of the most popular similarity metrics when applied to research articles. We analyze the results of those metrics in two ways, with a synthetic test that uses scientific papers and Ph.D. theses, and in a real-world scenario where we evaluate their ability to cluster papers from different areas of research.

Keywords: document similarity; similarity measures; word embeddings; natural language processing

Citation: Gómez, J.; Vázquez, P.-P. An Empirical Evaluation of Document Embeddings and Similarity Metrics for Scientific Articles. *Appl. Sci.* **2022**, *12*, 5664. https://doi.org/10.3390/app12115664

Academic Editors: Valentino Santucci and Julian Szymanski

Received: 6 April 2022
Accepted: 30 May 2022
Published: 2 June 2022

1. Introduction

Currently, all sorts of textual documents are generated at a fast pace. From blog posts on social platforms to more traditional news, articles, and reports, the number of documents that are published every day is continuously increasing. Reinsel et al. estimated that the sum of data from data centers, enterprise hardened infrastructures, and devices in the world would grow up from 33 zettabytes in 2018 to 175 zettabytes in 2025 [1]. These documents may appear on the Internet as publicly accessible or behind paywalls. Surprisingly enough, and although the amount may be negligible by now, not all the authors or consumers are human. Currently, in certain niches, documents are generated, and read, by artificial intelligence software (https://www.theguardian.com/commentisfree/2020/dec/05/companies-are-now-writing-reports-tailored-for-ai-readers-and-it-should-worry-us, accessed on 15 May 2022). Independent of the final consumer of the document, these data undergo several automatic processing tasks for solving problems such as summarization, indexing, or searches [2]. In all these processes, one key basic level task that may be required is the evaluation of document similarities.

Many ways exist to evaluate document similarity. In natural language processing, the state of the art is the use of word or document embeddings followed by some similarity measure (e.g., cosine similarity). However, to our knowledge, there is no previous systematic evaluation that analyzes the performance of the most commonly used measures. More concretely, no previous research exhaustively compares the outcomes of embeddings + distances in unsupervised clustering as well as performance times on a regular PC as we do. Thus, our goals are:

- Evaluate and compare a number of document embedding techniques for text similarity evaluation.
- Evaluate and compare a set of similarity functions for document comparison.

- Measure the performance of a set of word embeddings and similarity metrics on a regular computer.

Note that our interest is to see how practical these algorithms are; that is why we include the measurement of the performance on a regular computer. Moreover, as discussed later (see Section 4), we also discarded some of the most recent models that, not only require highly capable servers, but also have huge CO_2 consumption during the training phase. We expect that the insights we have obtained with the experiments will be useful for a wide range of researchers.

After introducing some background concepts in Section 2, and reviewing previous work in Section 3, we proceed with our analysis. First, we present the different popular embeddings and similarity measures from the literature that will be used in our experiments in Section 4. We describe the implementation details of the experiments in Section 5. Then, the validation experiment is described in Section 6, and the clustering experiment is described in Section 7. The results are analyzed in Section 8. We conclude the paper with some potential lines for future research in Section 9.

2. Basic Notions of Similarity Measures

Mathematically, a similarity measure is a real-valued function that quantifies the notion of how two entities differ. The similarity measures can be defined in many ways. Two approaches are very common: formulation as a similarity value or as a distance. In this context, a distance is the inverse of a similarity value: the larger the distance between two entities, the more dissimilar they are. A distance can be a metric or not, depending on whether it fulfills certain axioms. Given two elements x and y of a set X, a distance function d is a metric on the set X if the following conditions are fulfilled:

1. **Non-negativity:** $d(x, y) \geq 0$;
2. **Identity of indiscernibles:** $d(x, y) = 0 \Leftrightarrow x = y$;
3. **Symmetry:** $d(x, y) = d(y, x)$;
4. **Triangle inequality:** $d(x, y) \leq d(x, z) + d(z, y)$.

The accomplishment of these axioms is relevant in some cases, such as when may use the metric as a distance to position documents' representations in a 2D or 3D layout. However, in many contexts, it is not imperative to work with distances that are metrics. Some other factors need to be considered. For example, large objects that differ only in a small portion are intuitively more similar than smaller objects that differ in the same portion. Therefore, some authors also use the notion of *normalized admissible distance* [3], which takes into account this fact. Thus, the absolute difference between two objects does not govern similarity. However, the relative difference seems to be a more intuitive indicator of similarity. As a result, some authors have proposed measures based on this notion of normalized distance, as we will see later.

3. Related Work

Although specific methods tailoring a concrete domain may yield better results than unsupervised methods (e.g., [4,5]), we are interested in general models widely applicable to a range of fields. One of the key factors is that training learning models for specific domains or tasks is very costly regarding energy and environmental effects [6].

There is a significant amount of research involving texts comparison. However, the number of ways that text can be processed and compared varies greatly. Most typical forms of text similarity involve small documents, ranging from single words to one sentence or a paragraph. These are simpler to evaluate. One of the challenges we find when addressing evaluation problems is the difficulty to find baselines to compare. For example, some datasets have been built to represent word similarity and relatedness. However, they are fairly small, with some hundreds of words [7]. Furthermore, there are no equivalents for different languages. When addressing longer documents, researchers evaluate their success by custom datasets, and by manually analyzing some results. For example, Dai et al. [8] compare Wikipedia articles, and for a selection of results, they check whether the nearest

neighbors make sense. Shahmirzadi et al. [9] compare patents, and they use the hierarchy of classes created by USPTO, patent citations, and patents rejections list (as they expect to be more similar than patent citation) to assess their results. Alvarez [7] use academic articles from PubMed, and they create a custom dataset for testing, where the similarity is obtained from common references. Finally, Vázquez created a synthetic testing dataset by taking research articles and iteratively adding pseudo-random sentences from one document to another [10]. This way, the documents become more and more similar. Though we might not find this kind of scenario in the real world often, it may be suitable for evaluating measuring similarities, for example when the problem to solve involves ranking the similarity among a set of texts.

We can find in the literature several papers that compare the behavior of different vector space models when used for similarity detection between texts. Most of those comparisons are restricted to a few vector representations. Furthermore, they usually do not compare similarity measures: commonly cosine similarity is the one used.

For short texts, several papers focus on word or sentence embeddings. For example, Lau and Baldwin compare *word2vec* and *doc2vec* in two tasks (duplicate questions and sentences' similarity) that involve only short paragraphs [11]. Arora et al. [12] presented a method for sentence embedding based on a weighted averaging of word vectors. Their method outperforms other popular methods to detect sentence to sentence similarity. Other approaches that create sentences embeddings are due to Kiros et al. [13] or Wieting et al. [14].

For longer texts, Dai et al. [8] compare *doc2vec* [15], also known as paragraph vectors, with Latent Dirichlet Allocation (LDA) [16] for Wikipedia articles search and *arXiv* papers. They found that *doc2vec* works better than LDA for finding nearest neighbors on Wikipedia. It also generates a lower number of false positives. In both cases, the similarity measure used is cosine similarity. On the other hand, given the large space of parameterization possibilities, as well as the problems to be addressed, previous research has shown contradictory results when analyzing the performance of different embeddings, such as *GloVe* versus *word2vec* (cf. [17,18]). It is, however, widely accepted, that these systems outperform other classical approaches, such as Pointwise Mutual Information, for some tasks [19]. Shahmirzadi et al. [9] compare patents using vector spaces built using TF-IDF and *doc2vec*. Though the latter works better, to achieve the best results, fine-tuning is necessary. Given the costs involved, TF-IDF may be a sensible choice in many scenarios due to its performance and lower costs. Again, cosine similarity is used to compare embeddings.

Our work may be related to the work by Naili et al. [20]. They compare word embeddings for topic segmentation in English and Arabic languages. They found that for most of the tasks, GloVe and *word2vec* yield similar results and both seem to outperform LSA [21]. They also highlighted the fact that these models work better when the training and testing data sets are from the same domain. We have in common that we also analyze several word embeddings, but we extend the analysis to study how different similarity metrics behave. Alvarez also had a similar goal: comparing the effect of different embeddings for article similarity calculation [7]. Their evaluation uses three levels: title, abstract, and full body. Moreover, they compare *doc2vec* with *doc2vecC* [22], an extension of *doc2vec* that creates an embedding by averaging all context windows in a document. It costs around 12 times more to train than *doc2vec*. Furthermore, the authors report that despite it performing slightly better than *doc2vec* for abstracts, it is less effective for full-body articles.

Concerning similarity measures, as denoted, most of the evaluations use cosine similarity. Alvarez used Word Mover's Distance (WMD) [23], but only for titles and abstracts, due to its cost. We evaluate both cosine similarity, WMD, and other versions such as the Soft Cosine Similarity and the Relaxed Word Mover's Distance, and we also test a measure from Information Theory, Normalized Relative Compression [10].

4. Embeddings and Measures

To perform an extensive evaluation of the performance of different embeddings and similarity measures, we selected a set of the most popular ones. In this section, we give a brief description of them.

4.1. Embeddings

When talking about embeddings, one of the first ones that may come to mind is *word2vec*. **word2vec** (w2v), introduced by Mikolov et al. [24], trains a network language model in two steps: (i) word learning, and (ii) training $n-gram$ Neural Network Language Models (NNLM) on the top of the representation of words. They introduced two log-linear model architectures: the Continuous Bag-Of-Words Model (CBOW) (that predicts a current word based on the context and a window of words around the one to be predicted) and the Continuous Skip-gram Model (CSG) (that predicts a window of words around a known word). The authors found that low-dimension vectors, e.g., 300, trained in more data are capable to catch more accuracy than higher-dimension vectors trained in less amount of data. They also found that the training on twice the data during one epoch achieves in many cases better results than iterating over the same data during three epochs, which makes these models clearly interesting to be trained on large amounts of data.

fastText (fT) model, introduced by Bajonowski et al. [25], is presented as an extension of the CSG model, but instead of taking into account entire words, fT runs over subword information. The idea of this model lies in the fact that languages such as Spanish, which have a lot of non-commonly used conjugations, are hard for learning good word representations because these words are rare in the training sets. To catch these words, the model works with subword units. It was shown that working with subword information, the modeling of the morphologically rich language is improved. They compared the fT model to the basic CSG model for several languages analyzed (Czech, Russian, Spanish, and French). The results found that the prediction rate improves, albeit with different rates depending on the language.

GloVe (GV), presented by Pennington et al. [17], tries to combine two approaches to learn word vectors: the global matrix factorization and the local context window. The global matrix factorization approach is a set of techniques that decompose large matrices containing captured statistical information. The most commonly used is Latent Semantic Analysis (LSA) whose matrix captures the appearances of words in documents. In the end, these types of techniques are statistics-based methods to learn word representations. On the other hand, the local context window-based techniques, such as CSG and CBOW, are methods focused on the context of the words, without taking into consideration the associated corpus statistics. The authors found that in almost all tasks with different corpus, GV outperforms w2v models. Relative to itself, GV seems to be the best one in semantics but not in syntax, but in the end, good in general. Moreover, the authors concrete that the syntactic subtask performance improves when the corpus size increases, but it does not necessarily occur in a semantic task.

Introduced by Le and Mikolov [15], **doc2vec** (d2v), also known as Paragraph Vector, extends word2vec models to use the same base algorithm to learn document vectors and to make the predictions more accurate using their information. As in w2v models, the proposed d2v models map every word to a unique vector (column in a matrix W), but they also map every sentence, paragraph, or large text to a unique vector (column in a matrix D). Essentially, the only change from w2v is that the hidden layer is computed by averaging the matrices W and D, instead of just the matrix W. Using the proposed models in Sentiment Analysis and Information Retrieval tasks, the authors found that both outperform in terms of error rate all the baselines compared. Regarding the window size, they propose a range from 5 to 12. Since doc2vec has a representation for whole sentences, we cannot use similarity metrics that rely on the use of word representations, such as the SCS or WMD and its derivatives. Consequently, only CS will be tested with this representation.

ELMo, the acronym of Embeddings from Language Models, was introduced by Peters et al. [26]. It uses a new type of networks, Recurrent Neural Network (RNN), i.e., a neural network that has internal connection with itself, allowing the use of an internal state (memory). Although we did several experiments with it, some others did not finish in acceptable time, as explained later.

The last method we consider is named Bidirectional Encoder Representations from Transformer, or **BERT**, due to Devlin et al. [2]. In this case, BERT uses another type of architecture, the transformer [27]. A transformer involves an encoder and a decoder, containing both an attention layer, being this layer the one that allows dispensing with the recurrence of RNNs. Because there is no memory, they need to consume an entire set of tokens. Both the encoder and the decoder have three Multi-Head Attention layers. Unfortunately, BERT does not work with texts containing more than 512 tokens, which is an arbitrary selection carried out by the model's authors. This is not enough for scientific articles, since they commonly have from 1 k+ to almost 10 k different tokens (see Section 7.1 for concrete numbers). Thus, using some technique such as splitting the text into multiple parts might be unfair, and the results difficult to interpret, if compared directly to the other techniques. Other possibilities might be the evolutions of BERT, such as RoBERTa [28] or ALBERT [29], which outperform BERT training during more time and/or using more data, or the recent Transformer-based models which can work with more than 512 tokens. Namely, XL-Transformer, which has no input length limit [30], or its evolution, XLNet, also with no input limit and outperforming BERT in many tasks [31]. There have also been published document-specific transformer-based models such as Longformer [32] (which, in the published version, supports up to 4096 tokens, still smaller than required). Unfortunately, these systems have higher training/inference computation requirements. For example, the Longformer model pretrains starting on the RoBERTa released checkpoint [32]. As a result, we pulled BERT and other aforementioned models from the analysis since they are extremely hard to train if we want remarkable results, and they have unjustified ecological impact [33]. e.g., Strubell et al. [6] estimated the emitted CO_2 for the training of the BERT-base model published by Google in 652 kg, with 79 hours of training in a 64 Nvidia Tesla V100 GPUs. Another important point in this decision is the prediction time of big bunches of large documents such as a thesis, due to the number of computations required by these big models. With this paper, we show that simpler and less complex models, that can be executed on a regular desktop, we achieve remarkable results. Therefore, we will focus on the already-established models in the literature.

4.2. Similarity Measures

In NLP, the most common measures dealing with text are cosine similarity and soft cosine similarity. Both of them work directly over word or document embeddings. Although other measures have also been used, such as the Jacquard similarity, or the Euclidean distance, they are far less popular than the ones described next.

Cosine Similarity is a measure of similarity between two non-zero vectors. CS makes a judgment of orientation, not magnitude, over the vectors. This is because it computes the angle of both and uses this angle as a measure of similarity. This measure is probably one of the most common similarity measures used and studied in the literature for areas such as data mining, information retrieval, or text mining. It can be calculated as follows:

$$\cos\theta = \frac{A \cdot B}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

where A_i and B_i are components of vectors A and B. The higher the absolute value, the more similar the elements should be. One of the problems of this measure is the fact that only angles between vectors are used, not magnitudes. If magnitudes are important, the cosine similarity maybe not completely useful. As for complexity, cosine similarity can be calculated in $\mathcal{O}(n)$, where n is the dimensionality of the vector. It is important to remark

that, although it is a similarity measure, it is not a proper metric because the triangle inequality does not hold, and it is defined for positive values only.

Soft Cosine Similarity is a generalization of the CS [34]. When objects are represented in a Vector Space Model (VSM), they are represented as vectors of values of features, corresponding each feature to a dimension in the VSM. The traditional CS considers VSM features as independent or in an orthonormal basis, i.e., there is no relation between the different features. In many contexts, this is not realistic, and SCS tries to take this into account to strengthen the measure. To consider the similarity or relation between features the authors propose a free-context formulation, due to the feature-nature dependency, to compute the relation between the features. This relation is represented as a matrix that can be precalculated or evaluated in real-time following whichever measure or system. For example, in NLP context the authors use the Lavenshtein distance (String metric for measuring the distance/similarity between two strings by counting the minimum number of operations needed to transform a string into another one.) and hint at the possibility of using the measures available through WordNet (Lexical database of semantic relations between words.). Sidorov et al. [34] found that SCS works better than CS for many tasks using the Lavenshtein distance as a measure of similarity between features.

Word Mover's Distance [23] is a distance measure introduced that works over text documents. It is based on the well-known Earth Mover's Distance (or EMD), introduced by Rubner et al. [35]. WMD starts from the same point as SCS: the not real orthogonality of the feature vectors. Furthermore, as well as SCS, for the specific use in NLP, WMD tries to supply the necessity of considering the relation or distance between individual words when comparing two text documents. This measure takes advantage of the fact that word embeddings preserve the semantic relations in vector operations. Using it as a base, WMD, in general terms, represents text documents as a weighted point cloud of embedded words. It measures the distance between two documents as the minimum cumulative distance that words from document A need to travel to match exactly the point cloud of document B. Specifically, WMD, such as EMD, reduces the problem of the distance to a transportation problem. To calculate the distances of two documents, a flow matrix $T \in \mathbb{R}^{(n \times n)}$ represents the cost of traveling from each word to the rest. Apart from WMD, the authors propose a relaxed version of the Word Mover's Distance (RWMD), which we also tested. The difference from WMD lies in the elimination of the second constraint. Thus, the problem becomes reduced to the search of the minimum cost of travel for each word. This approach can be performed as a Nearest Neighbour Search (NNS) in the Euclidean word2vec space. Rubner et al. [35] assess that EMD is a metric because it uses a metric for the calculation of the cost, i.e., the Euclidean distance. WMD, as a special case of EMD, is also a metric itself, unlike CS and SCS. The comparison with other relevant methods performed by the authors indicates the general superiority of WMD with many datasets for a k-Nearest Neighbour (kNN) task.

Normalized Relative Compression (NRC) is a measure of distance between two files based on a model constructed for a reference file and the compression of the target file. It works trying to reconstruct the target file using the reference, and this "possibility of reconstruction" is the degree of similarity or the distance. Like CS and SCS, but unlike WMD, NRC is a general-purpose measure, as it is based in the Normalized Compression Distance [3], which can be applied to any sequence. NRC can be computed using the following expression:

$$NRC(x, y) = \frac{C(x\|y)}{|x|} \tag{2}$$

where $C(x\|y)$ is the compression of x relative to y.

The relative compression is performed in two steps. First, given a reference file y, the compressor builds a representation of it using finite context models. Second, it builds a representation of x, i.e., the target, based on the representation of y. With these two steps,

NRC measures how much the target can be built using the reference. For this project, we use the relative compressor developed by Pinho et al. [36].

NRC, unlike the previously presented measures, works with the size of the documents and its compression, performing a relative compression. Clearly, this is an entirely different distance calculation approach. As a normalized measure, it ranges from 0 to 1, and the result depends on the $C(x\|y)$ factor, which can be:

- $C(x\|y) \approx 0$ if, and only if, x is very similar to y;
- $C(x\|y) \approx |x|$ if, and only if, $C(x|y) \approx C(x)$;

where $C(x)$ is the compressed file size and $C(x|y)$ is the conditional compression of x given y.

NRC is asymmetric and does not preserve the triangle inequality [10]. Thus, it is not a metric. Recent results obtained in its use for authorship attribution [36] and in the visual analysis of research papers [10] suggest that NRC is a good measure for many NLP tasks.

4.3. Selected Embeddings and Similarity Measures

After analyzing the different possibilities to deal with text available in the literature, we ended up with 5 different embeddings, plus the NRC measure that, as explained, works using a wholly different philosophy. These textual representations are evaluated using four similarity measures, in addition to NRC. Table 1 summarizes the general characteristics of each one of the measures selected for this project.

Table 1. Summary of the evaluated similarity measures with their complexity. Most of the techniques have an acceptable cost except, probably, WMD, which is very slow. Regarding memory, again, the cost is affordable. Here, the Finite Context Models used in the relative compression grow exponentially with the number of orders. However, in our case, we used a bounded number of orders which limits the required memory and keeps it at an acceptable rate.

Name	Purpose	Metric	Data Format	Time Complexity	Memory Complexity
CS	General (features)	No	Vector	$\mathcal{O}(n)$	$\mathcal{O}(1)$
SCS	General (features)	No	Vector	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
WMD	Text	Yes	Text	$\mathcal{O}((n+m)n)$	$\mathcal{O}(n^2)$
RWMD	Text	Yes	Text	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
NRC	General (files)	No	Bytes	$\mathcal{O}(n_{ref} + n_{target})$	$\exp(orders)$

As mentioned earlier, not all combinations are possible. For example, NRC is a distance itself and does not use any word embedding. ELMo does not have unique representations for words, and thus, measures that depend on this, such as WMD or SCS, cannot be used with the embedding. The result is a set of 12 combinations, as depicted in Table 2.

Table 2. The different combinations of measures (columns) and embedding techniques (rows) used in the experiments. Being imprecise, NRC would be the embedding and the similarity measure at the same time.

	SCS	WMD	RWMD	CS	NRC
w2v	✓	✓	✓		
fT	✓	✓	✓		
GV	✓	✓	✓		
d2v				✓	
ELMo				✓	
NRC					✓

5. Empirical Evaluation

After reviewing the different embeddings and measures, we proceed to describe how the experiments were carried out. To guarantee the same conditions of evaluation for all the combinations, we established the same training method for all the models. Even so, some techniques, due to their implementations or characteristics, require specific procedures, as explained next.

5.1. Implementations

To keep the major equality between distances and embeddings, we selected the implementations of the distances in the *Gensim* (Available at https://radimrehurek.com/gensim/ accessed on 15 May 2022) and *SciPy* (Available at https://www.scipy.org accessed on 15 May 2022) libraries. For CS, the implementation used is the one from *SciPy*, which consumes two vectors and returns the distance. SCS and WMD are computed using the implementation from *Gensim*. For the first one, a similarity matrix and the documents as BoW are needed. The matrix can be obtained directly from the *Gensim* embeddings, passing a dictionary of the words in documents to the similarity_matrix() function; and the BoWs can be easily obtained using the function doc2bow() in the class Dictionary from the corpora module of *Gensim*. Soft-Cosine is calculated with softcossim() from the *matutils* module. The second one, WMD, just requires the documents to be tokenized by sentences, and these sentences are tokenized by words. The distance is obtained by calling the wmdistance() function from embeddings, passing the documents. For RWMD, we found that *Gensim* has no implementation. Thus, after inspecting some other possibilities, including the one provided by R, we ended up choosing the implementation available in the *spaCy* (Available at https://spacy.io accessed on 15 May 2022) library. More concretely, we used *wmd-relax* (Available at https://spacy.io/universe/project/wmd-relax accessed on 15 May 2022), which has no changes with respect to the one proposed by Kusner et al. [23].

5.2. Selected Data

This project is focused on large texts, more concretely research papers. This allows for the possibility of having a bounded, but large vocabulary set, which is important if we look for good results. It also represents a real use scenario that can be achieved with a single computer. The training dataset used, provided generously by the library of our university and initially processed by the inLab FIB laboratory (inLab FIB UPC is an innovation and research laboratory of the Facultat d'Informàtica de Barcelona of the Universitat Politècnica de Catalunya · BarcelonaTech.), is formed by 1.072 papers in English from 18 research groups of the *Universitat Politècnica de Catalunya*. The set of articles was hand-picked by the library technicians to generate a representative set of the research that is carried out in our university, big enough to be used in the context of natural language processing. Thus, the research groups include different areas, such as architecture, maritime engineering, computational biology, and artificial intelligence, to name a few.

5.3. Cleaning

First, cleaning, and data preparation is necessary to facilitate the posterior automatic processing. The first step is to convert the PDF files to plain text. To accomplish this, we used Apache Tika™ (Available at https://tika.apache.org accessed on 15 May 2022) from The Apache Software Foundation. After this conversion, a set of common processing steps are applied: (i) lowercasing, (ii) removal of emails and web addresses, (iii) erasing non-printable characters, (iv) stopword removal, and (v) stemming. Email and address removal was carried out using the *re* Python library for regular expressions. For stopword removal, we used the stopwords set from the *Gensim* library. Stemming was carried out using the Porter Stemmer, also available in the *Gensim* library. Finally, to guarantee that the result is correctly formatted, all the papers used as training data were manually checked by one of the authors.

5.4. Training

For all the analyzed models, i.e., w2v, fT, GV, and d2v, we perform the training. For ELMo, we used a pretrained model, since training or hyper-tuning them implies the huge computational costs required and their environmental impact (e.g., Ref. [6] estimates that the training impact of those models generates emissions of 120 kg of CO_2 with 336 h of training for ELMo). The pretrained option for ELMo is the one available in TensorFlow Hub (TFHub) (Available at https://tfhub.dev/google/ELMo/3 accessed on 15 May 2022). More specifically, we used the third version trained on the 1 Billion Word Benchmark (benchmark corpus for measuring progress in statistical language modeling. Available at https://www.statmt.org/lm-benchmark/ accessed on 1 March 2022).

Table 3 presents the training parameters selected and models characteristics. As mentioned previously, this way guarantees equal conditions for all the models. The parameters which do not appear in the table have not been modified, leaving the default values. w2v, fT, and d2v have been trained using the implementations available in the *Gensim* library.

Table 3. Summary of the models' training parameters, where applicable.

	w2v	fT	GV	d2v	ELMo
Vector size	300				1024
Window size	7			5	-
Min vocabulary count	4				-
Corpus count	398,387				793471
Corpus total words	4,366,988				1B
Training epochs	10				-
Architecture	CSG	CSG	-	PV-DM	-

For w2v and fT the training processes are the same, just changing the training class. It consists of reading the corpus using an iterator class that processes the corpus of documents to convert it into a list of tokenized sentences. The details of the implementations can be found in Appendix A.

6. Task I–Validation

The aim of this experiment is twofold: First, it will help us validate all the combinations of measures and embeddings (exposed in Table 2) as a good distance/similarity. Second, it will give us an idea of the outcomes of all algorithms with a synthetic dataset, that works as a ground truth.

Our notion of similarity dictates that, given two documents, if we add some sentences from one to the other, their similarity must increase. From a formal point, this behavior is what we would expect from a similarity measure that fulfills the *Identity of indiscernibles* and *Symmetry* axioms. Therefore, by iteratively repeating this process, similarity should increase monotonically in each iteration [10]. Note that, in most NLP tasks, this notion of fine-grain similarity is not needed. However, algorithms failing this test up to a certain (high) degree will certainly be dubious candidates for document similarity measurement, particularly in those tasks in which it is mandatory to have rigorous results, for example if a ranking is needed.

Besides measuring the behavior of the different algorithms in this test, we also measure and compare the time costs.

6.1. Data

To perform the validation experiment, two accumulative-incremental-by-addition datasets will be created; one for research papers, which is the main focus of this project,

and one for Ph.D. theses, to explore the performance of the combinations beyond the length of 10 to 20 pages that most papers usually have.

An accumulative-incremental-by-addition dataset consists of taking two source documents, which will be called *base document* (d_b) and *destination document* (d_{d0}), respectively. Then, in step one, we add several words or sentences, randomly picked, from d_b to d_{d0}. This results in a modified destination document (d_{d1}). We perform this step several times, but each step we take as the destination, the result of the previous step. That is, the step i of this procedure uses d_{di} document as destination one, and generates d_{di+1}. The result is a set of document pairs that, with the intuitive definition of similarity, should be increasingly similar. In our case, we add sentences, to preserve the context of the words, which is an important aspect for the embeddings that capture words' context. This creation process is based on the premise that the more sentences the destination document receives, the more similar to the base document should be. Furthermore, the decrease/increase should be monotonous, since the previous sentences are accumulated at each step.

For the papers' dataset, the same previously listed research groups from our University (UPC) will be used. To systematically evaluate the combinations for different scenarios, one base paper from one group (with acronym CDIF) will be tested with other five papers from the same research group (named CDIF1 to CDIF5 in the figures), and the same base paper will be tested with other five papers from five different research groups (with acronyms ANT, ADR&M, UMA, AIEM, and IDEAI). The list of selected papers can be found at Appendix B. At each step, 10 new sentences are copied from the base paper to the destination one. The result is a set of 11 documents ranging from 0 to 100 transferred sentences.

Following the same idea, the Ph.D. theses set has been created, taking as a base the thesis "Influence of agricultural practices on the Microbiome and the Antibiotic Resistance Gene complement in soils, plants, and crops" by Francisco Diogo de Almeida Cerqueira, and as a destination, the thesis "Deep Learning architectures applied to wind time series multistep forecasting" by Jaume Manero Font. Unlike the set of papers, for the theses, the steps go from 0 to 1000 transferred sentences, with 10 sentences copied each step. We proceeded this way because both theses have more than 150 pages with very diverse and differentiated information, which appears to be hard to break just by passing a few sentences.

6.2. Performance Analysis

First, we analyze the performance of the different combinations. We measured the average time of execution as well as the standard deviation. For papers, the average is the result of 11 executions, while the Ph.D. theses data used 101 executions.

In Figure 1, we depict the time cost for the experiment that compares papers from the same research group (third row) and papers from different groups (fourth row). In Table 4, we can see the mean and the standard deviation of the computation times for the different sets of documents previously described. Each one corresponds to papers from the same groups, papers from different groups, and theses, respectively. If we analyze the standard deviations, we can see that it does not explode despite the increase in the number of sentences. On the contrary, the values keep low, which adds confidence to the samples. Average calculation times, which include the embeddings computing when necessary, together with the distance evaluation, show that the embedding used for the same measure does not change the order of magnitude of the times. This happens for the three scenarios. As an example, for SCS, the order of magnitude remains 10^0 with all the embeddings for papers from the same group and for the papers from different groups; and 10^1 for the theses. This shows that, in these cases, the impact of the embeddings' computation is not relevant. Average times grow with the increased dissimilarity between the groups. Thus, papers in the same group require a low cost in evaluating their similarity, which is smaller than papers of different groups. Furthermore, in turn, these values are also smaller than for the Ph.D. theses operations.

w2v	w2v + SCS				w2v + WMD				w2v + RWMD			
Documents	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**
Same group	3.20	5.15	1.87	0.77	30.10	46.18	13.30	8.65	0.33	0.57	0.19	0.08
Different group	4.30	9.71	2.42	1.82	97.69	195.78	97.67	54.51	0.84	1.78	0.31	0.42

fT	fT + SCS				fT + WMD				fT + RWMD			
Documents	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**
Same group	2.39	3.54	1.71	0.43	29.99	51.27	13.13	11.36	0.29	0.53	0.13	0.10
Different group	4.05	7.52	2.17	1.38	104.28	222.85	104.28	63.02	0.70	1.35	0.70	0.36

GloVe	GloVe + SCS				GloVe + WMD				GloVe + RWMD			
Documents	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**	**avg**	**max**	**min**	**stdev**
Same group	4.54	7.65	3.22	1.09	25.02	39.82	10.94	9.60	0.33	0.59	0.14	0.11
Different group	6.21	9.37	4.03	1.61	75.43	132.73	27.56	39.20	0.69	1.25	0.28	0.33

ELMo	ELMo + CS			
Documents	**avg**	**max**	**min**	**stdev**
Same group	166.12	444.97	54.06	83.91
Different group	174.24	368.11	77.63	76.60

NRC	NRC			
Documents	**avg**	**max**	**min**	**stdev**
Same group	0.06	0.15	0.03	0.02
Different group	0.07	0.29	0.03	0.04

Figure 1. Time performance (in seconds) of the different similarity measures when evaluating papers belonging to the same (third row of each table) research group and different ones (fourth row) with the different strategies. The timings include the calculation of the embeddings when necessary and the similarity measurement.

Table 4. Mean and standard deviation for computation times for the different combinations with Ph.D. theses.

	SCS			WMD			RWMD			CS		NRC
	w2v	**fT**	**GV**	**w2v**	**fT**	**GV**	**w2v**	**fT**	**GV**	**d2v**	**ELMo**	
$\bar{t}$	17,986	19,680	32,810	>1 h	>1 h	>1 h	59,137	50,750	50,653	0.648	>1 h	0.8489
σ_t	1598	0.969	0.695	-	-	-	4787	3288	3551	0.009	-	0.131

SCS is the measure with a lower increment in its order of magnitude for each scenario, i.e., it seems to scale well with the size and the complexity. NRC, instead, increases, but not up to remarkable values. Finally, the cost of WMD and CS + ELMo is so huge, that they could not be analyzed due to the executions that did not finish. Due to the characteristics of WMD, i.e., that is based on a Linear Programming Problem, it seems logical that the theses scenario will become untreatable for that distance because in the experiments carried out, the execution of the measure exceeds 8 hours. RWMD, which has one less constraint, relaxes the problem and becomes an option for all the scenarios, being between 10^{-1} and 10^{1}. All the experiments were carried out on the same computer. On the other hand, considering that CS is a low-complex measure ($\mathcal{O}(n)$), the problem which seems to be in the embeddings' computation with ELMo, is that it becomes a hard problem for large texts.

The measure with the best scalability is CS combined with d2v, which continues in the 10^{-1} order of magnitude for all the scenarios. The second is NRC, whose order of magnitude is 10^{-2} for papers and 10^{-1} for Ph.D. theses. SCS and RWMD have roughly the same performance, being the inflection point for RWMD (and WMD), the scenario of papers from different groups because the times observed are relatively higher in some cases.

6.3. Analysis of Results

Next, we analyze the behavior of the different similarity measures concerning our synthetic test. The expected result should be that, as long as we introduce a larger number of sentences from one document to the other, the similarity should increase. Since we

measured distances, the charts, ideally, should show a monotonically decreasing distance. First, we analyze the results relative to papers, and afterward, the Ph.D. thesis.

Figure 2 shows the results of soft cosine similarity for the different embeddings. There are two notable observations here: First, the distance decreases in all scenarios. Second, most of the papers belonging to the same group (the ones labeled CDIF*i*) have smaller distances both at the start and the end than papers from different groups. This follows our intuition that articles written by people of the same research group will likely describe more similar problems and techniques than papers of other research groups. Moreover, the relative position of the papers is mostly preserved (e.g., CDIF2 is the one with a smaller distance at the beginning and the end, and IDEAI is the group with both larger distances), although some differences appear within the papers in the middle range.

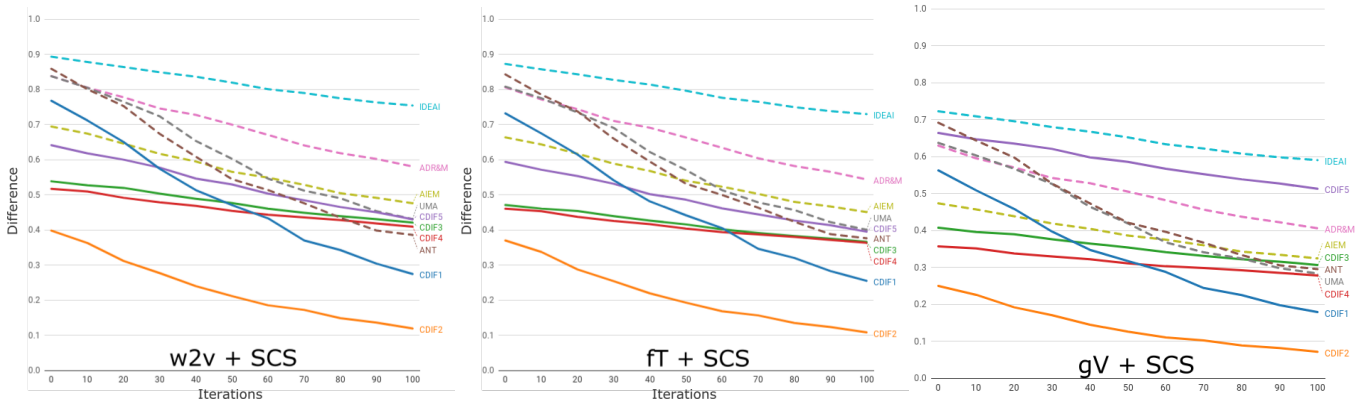


Figure 2. Behavior of SCS in the validation experiment. Some papers within the same group as the base paper (CDIF) exhibit smaller distances than papers from different research groups (labeled after the group name). More concretely, papers CDIF1 through CDIF4 show higher similarity than the rest of the papers in fT + SCS (**center**) and good performance in w2v + SCS (**left**) and gV + SCS (**right**). Moreover, in all cases, the distance decreases as the shared text between the files grows.

In Figure 3, we analyze the behavior of Words Mover's Distance. Like in the previous case, the distances decrease as the number of copied sentences from one document to the other increases. Again, most of the documents belonging to the same research group have smaller distances than the distances to papers of other research groups. In this case, the relative positions of the papers at the end are fully preserved with the three embeddings, and all of them share the initial and final rankings.

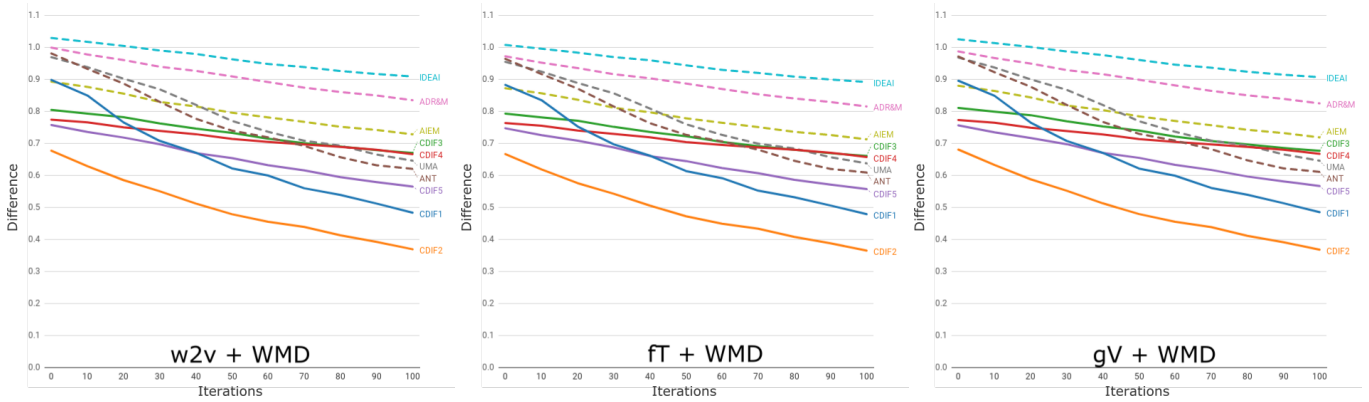


Figure 3. Performance of WMD in the validation task for w2v (**left**), fT (**center**), and gV (**right**). In this case, some papers of the same group (CDIF1, CDIF2, and CDIF5) are always considered as more similar to the base paper (CDIF) than the rest. Furthermore, in all cases, the distance decreases with larger number of phrases passed from one to the other.

The relaxed version of WMD is analyzed in Figure 4 for the different embeddings. Its behavior is very similar to the previous cases: monotonically decrease, most of the relative distances rankings preserved. Furthermore, it is noticeable that the behavior of the papers from the same research group than the base one seems better than in previous cases.

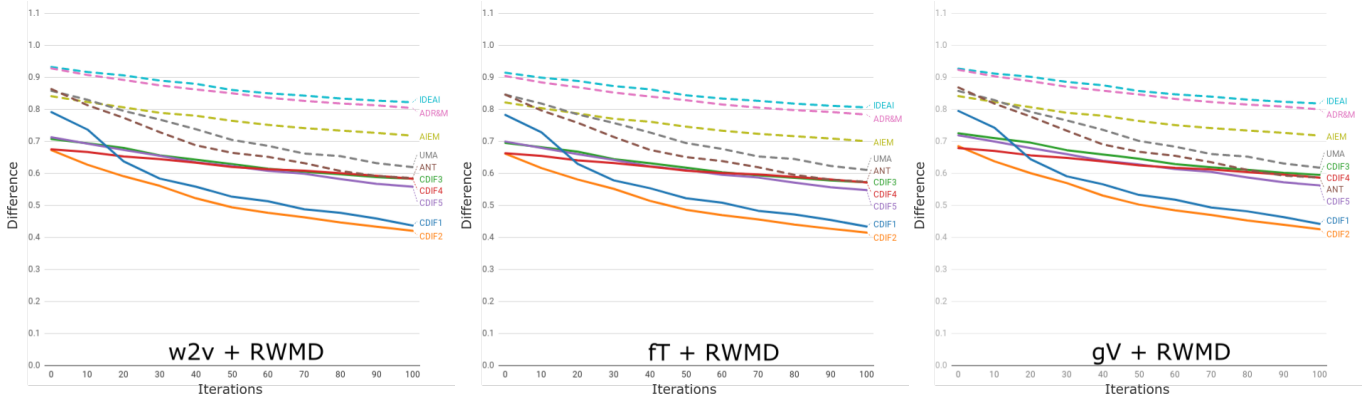


Figure 4. Analysis of RWMD with three different embeddings: w2v (**left**), fT (**center**), and gV (**right**). Distances decrease, and in two cases (w2v and fT), the papers of the CDIF group are also closer to the base paper, as expected.

We also analyze cosine similarity with the two embeddings that consider the whole document, d2v, and ELMo. The results are shown in Figure 5 (left two columns). Some differences are noticeable. Though d2v behaves similarly to the previous techniques, ELMo seems to work less efficiently: First, some papers exhibit larger distances after transferring the full set of sentences (e.g., CDIF2), and in other cases, the distances seem not to decrease at all. Second, some papers show a larger distance (beginning and end) than expected, such as the CDIF5, which is the one with the largest distance, while being compared to a paper of the same group.

Finally, the results obtained with the remaining similarity measure, NRC, are shown in Figure 5 (right column). NRC behaves very similarly to the first techniques: CDIF papers exhibit smaller differences, and the distance decreases as the number of sentences transferred increases.

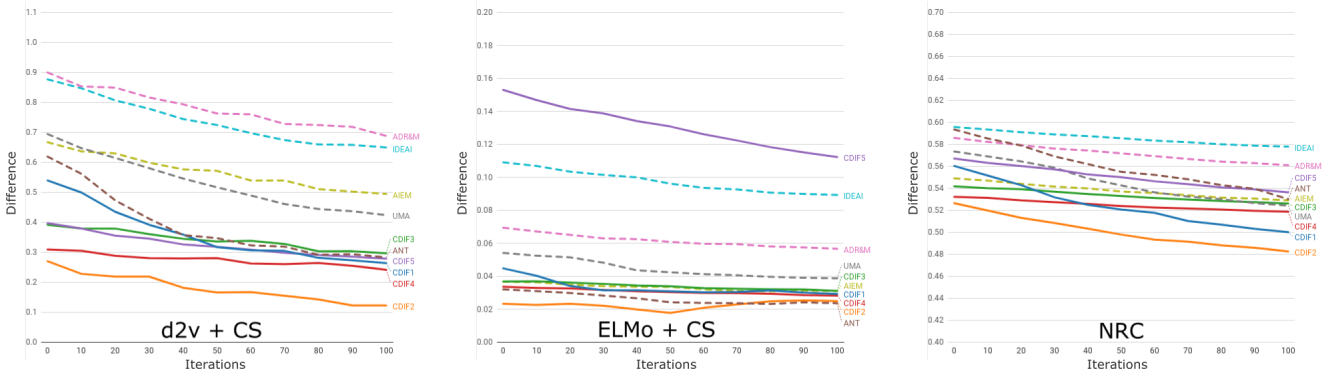


Figure 5. Analysis of the CS distance with doc2vec (**left**) and ELMo (**center**) for the validation scenario. Here, ELMo does not seem to behave as expected, since some papers do not exhibit decreasing distance as their contents become more similar. On the other hand, NRC (**right**) behaves as expected, with distance decreasing, and the papers within the same group showing smaller distances.

To sum up, we can say that, for all the distances, the behavior remains relatively equal for all the embeddings, essentially varying the values and the start/end values of some of them for some sets. All the distance measures show a quite stable monotonic decrease,

except for the pair ELMo+CS, as mentioned. WMD and RWMD generally have both the same behavior, not showing a clear difference between them, except for the distances values which slightly vary for each set. From the point of view of the embeddings, w2v and fT seem to have a relatively identical performance for the different distances, being the difference regarding GV the distances values. This is probably due to the statistical base of GV.

The second part of our validation experiment has to do with the comparison of even larger documents. In this case, we used two Ph.D. theses and performed the same evaluation as for the papers. Notice that, as mentioned in the previous sections, results for WMD and CS + ELMo will not be provided because their executions do not end after half a day. However, we still get a glimpse of how the other combinations behave with larger scientific texts.

The results of the comparisons are shown in Figure 6. Since only a pair of Ph.D. theses were compared, we rendered all the plots in a single chart. We can see that all embeddings and distances behave very similarly. The only one with slightly erratic behavior, though still yielding overall decreasing values with the increase in sentences passed to the other document, is d2v + SCS.

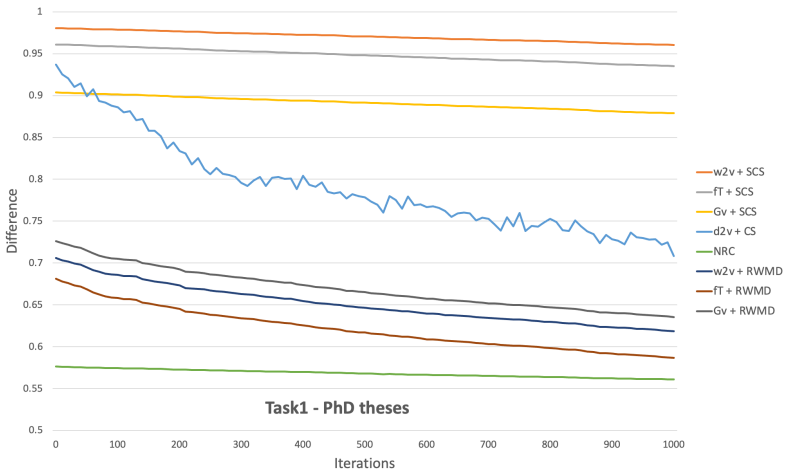


Figure 6. Task I–Ph.D. theses. We can see that most of the measures behave as expected, with a monotonic decrease. Only doc2vec seems to behave in an unexpected way, with those bumps.

7. Task II–Clustering

A more common task in NLP, and real-world scenario is unsupervised clustering. Our objective was to analyze up to which point embeddings and similarity measures help automatically clustering papers belonging to the same research area. We used the Agglomerative Hierarchical Clustering algorithm. The input is a set of tagged documents, and the output is the result of the clustering.

For the implementation, the chosen procedure consists of precomputing the distance matrix for all pairs of documents, saving it, and then using the linkage() function from the *SciPy* library to obtain the hierarchical clustering. The clustering method used is the Weighted Pair Group Method with Arithmetic Mean (WPGMA) [37], which is a classic and commonly used algorithm. At each step, it combines the nearest pair of clusters. Then, the distance of this new cluster to another cluster is the mean of the average distances between the items of the two original clusters and the items from the other cluster:

$$d_{(i \cup j),k} = \frac{d_{i,k} + d_{j,k}}{2} \tag{3}$$

7.1. Data

The data used for this experiment was a set of documents curated by the authors. It consists of 80 papers in PDF format. These papers have been listed in the Appendix C.

All of them are close to the areas of Computer Graphics and Visualization and belong to the following subfields: Ambient Occlusion Rendering (AO, avg. #tokens = 4834.3; avg. #different tokens= 1257.9), Bicycle Sharing Systems Analysis (BSS, avg. #tokens = 6358.4; avg. #different tokens= 1577.2), Immersive Analytics (IA, avg. #tokens = 8019.5; avg. #different tokens= 1971.0), Mobile Rendering (MR, avg. #tokens = 6894.1; avg. #different tokens= 1418.7), Molecular and Biological Visualization (MBV, avg. #tokens = 5789.8; avg. #different tokens= 1574.2), Rankings Visualization (Rank, avg. #tokens = 7473.7; avg. #different tokens= 1673.9), Sports Analytics (SA, avg. #tokens = 6354.7; avg. #different tokens= 1623.2), and Analysis and Visualization of Transportation Methods (TM, avg. #tokens = 6905.7; avg. #different tokens= 1562.4). As it can be appreciated, some groups are quite related. This, as it is obvious, adds more complexity and a greater need for finesse in the embeddings and measures. Note that the average number of different tokens (without stopwords) is quite similar among groups. For these papers, the preprocessing process applied is the same one explained previously for Task I.

7.2. Execution

The execution times for this task will not be analyzed deeply because they can be inferred from the ones sampled for Task I. The time complexity of these experiments lies in the computation of the distances' matrix, which for 80 documents has 6400 cells, i.e., 6400 distances must be computed, or at least, one of the two triangles in the matrix.

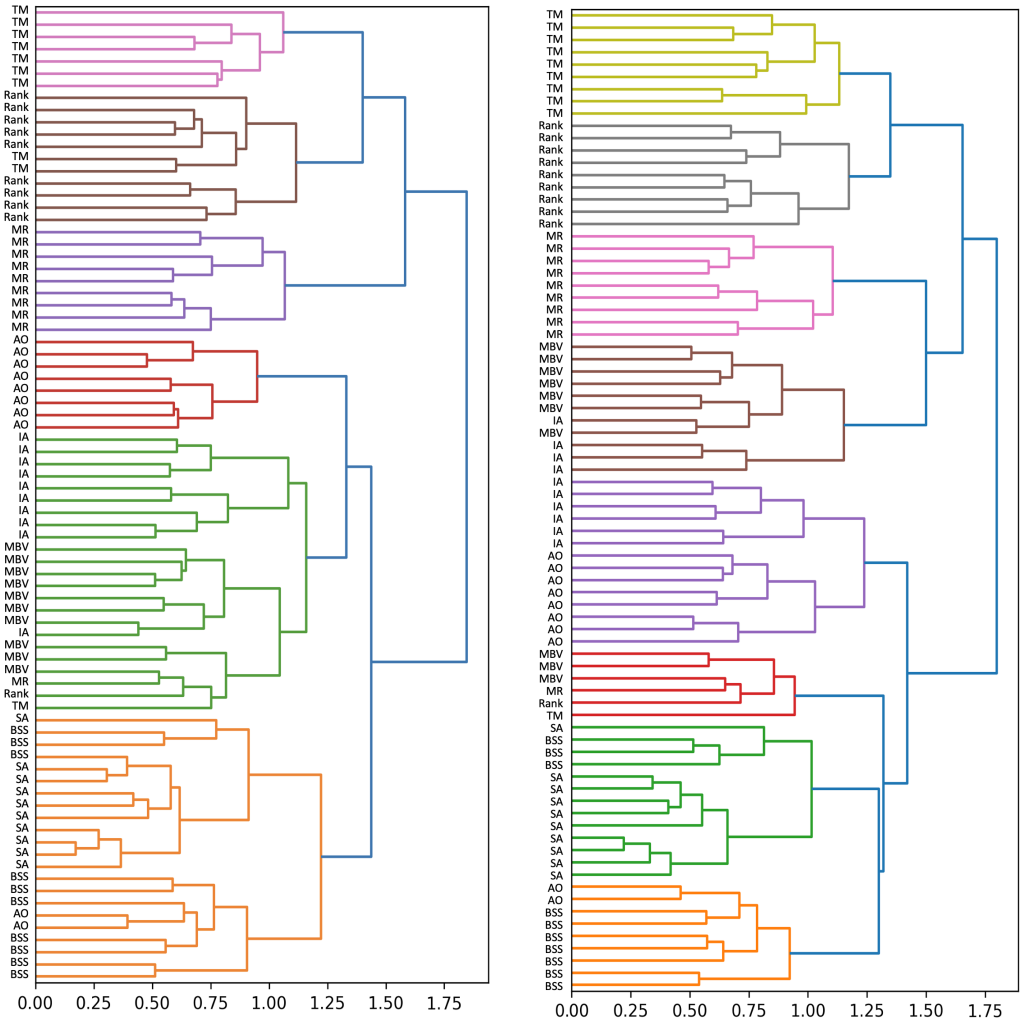
The computation of all the matrices took roughly half a day in the laptop used for the experiments. Unfortunately, for this task, the WMD runtime becomes unfeasible. The computation of the distance matrix for as many documents becomes a tough task for the used computer. The line of reasoning for CS + ELMo is the same.

7.3. Analysis of Results

To illustrate the results, we have used a set of dendrograms obtained for each combination of measure and embedding, except for the ones that were too costly to calculate, as mentioned earlier.

Notice that the coloring of the dendrograms has been left to the judgment of the function dendrogram() from the *SciPy* library. Moreover, we edited the images to slightly stretch the charts so that they fit in a single column. We also changed the verbose labels for abbreviations (which were already introduced in the previous section).

For the SCS case, both w2v and fT (see Figure 7) offer a similar performance. The super clusters obtained are positively consistent (slightly less with fT) and the small ones have some inconsistencies. For w2v, the Bicycle Sharing Systems set is combined with the Sports Analytics set. However, both fall in the same super cluster, which is acceptable. On the other hand, for fT, both previous cases are better clustered, but Ambient Occlusion, Immersive Analytics, and Molecular and Biological Visualization sets are combined more dubiously. Even with these small conflicts, both produced an excellent result for the Transport Visualization Methods, Rankings Visualization, and Mobile Rendering Systems.

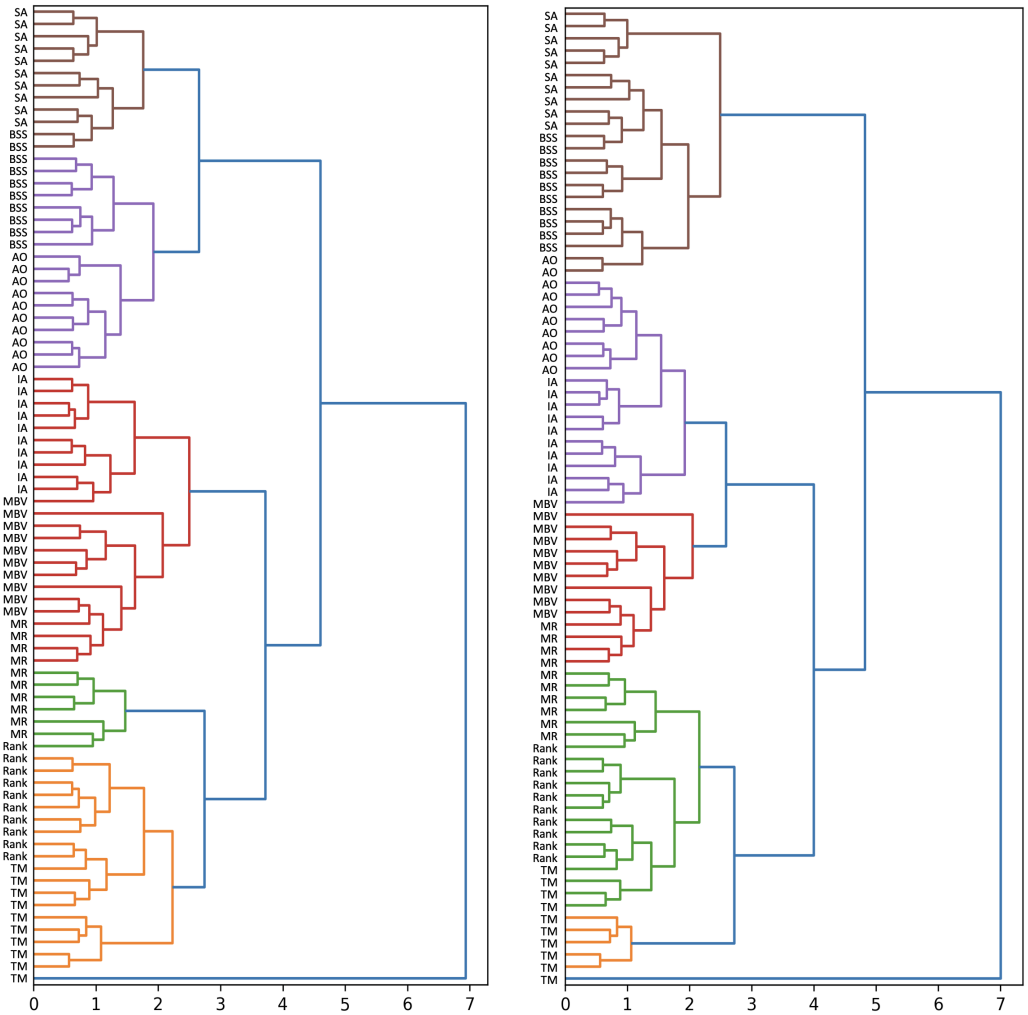


(**a**) w2v + SCS (**b**) fT + SCS

Figure 7. Task II–clustering. Comparing w2v + SCS and fT + SCS shows that their behavior is similar, and that the results are good for some categories (e.g., TM, Rank, and MR.). Colors denote clusters that have been determined automatically by the plotting algorithm, and the threshold can be fine-tuned if necessary.

Gv, which works with statistical information obtained using the context window approach, as explained previously, is the one that works better generally. Obtained super clusters are completely consistent. As a counterpart, small clusters exhibit several mistakes larger than the others. When using SCS as a distance, its performance appears to be good for a less detail-oriented clustering.

Ignoring the one-element-cluster, which constitutes a "single-large-cluster", in w2v and fT (see Figure 8), the RWMD case is, in general, slightly worse in comparison with SCS.



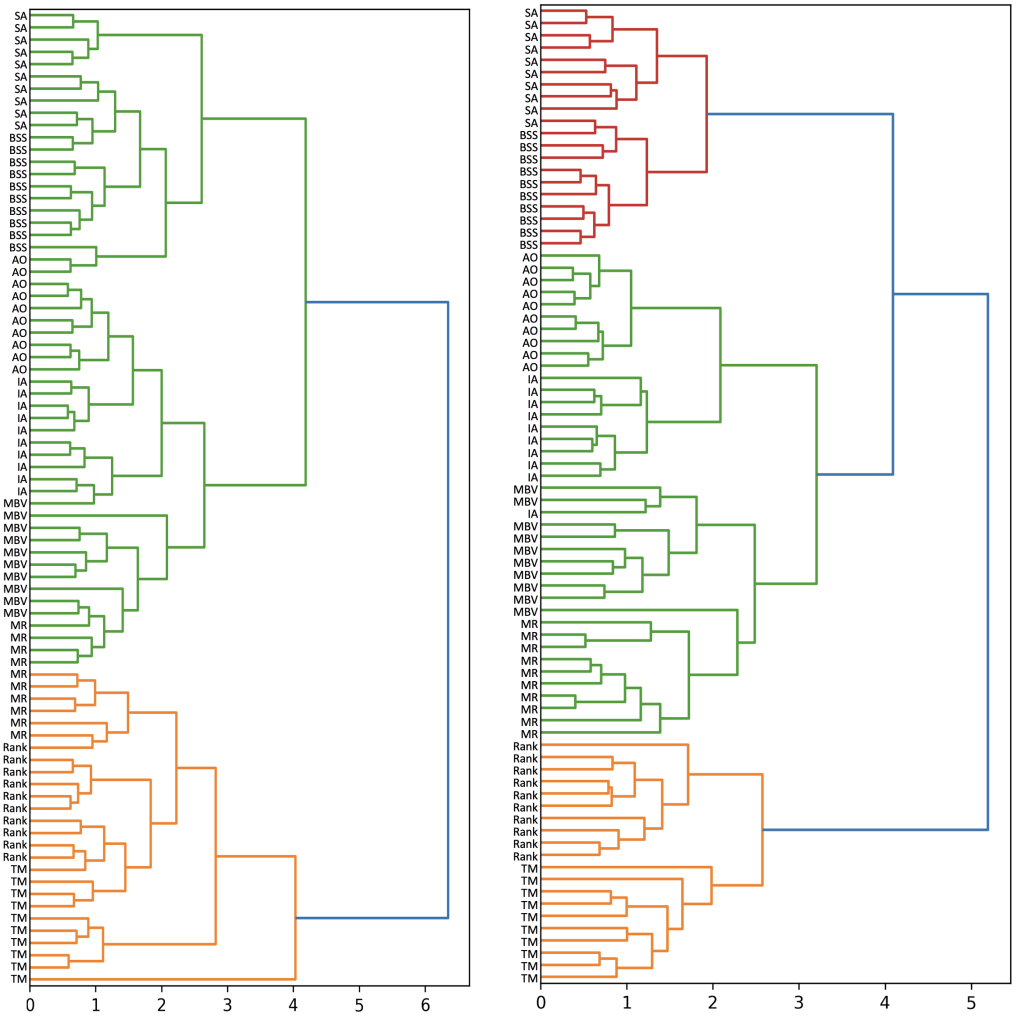
(**a**) w2v + RWMD (**b**) fT + RWMD

Figure 8. Task II. Clustering using w2v + RWMD and fT + RWMD shows less consistency in clustering: some groups are broken and grouped with other ones (e.g., TM or MR). Colors are assigned automatically by the dendrograms drawing algorithm. The threshold could be changed.

The consistency at the third and fourth levels is high and approximately the same as for SCS, but at the fifth level, the clusters become inconsistent. Notice that, for both cases, the behavior is the same, just changing the conformation of the clusters.

Remarkably, fT + RWMD separates well Bicycle Sharing Systems and Sports Analytics sets, while in SCS, this does not occur. On the other hand, w2v + RWMD clusters well Immersive Analytics and Ambient Occlusion sets, both highly impaired in SCS. These cases report the difficulty behind the clustering of papers from subareas of the same thematic area. The information captured by the embeddings, and its use by the distance measures, directly impacts their performances, dramatically varying the low-level results.

GV + RWMD (see Figure 9a) has generally the same performance as GV + SCS, but is more mixed-up and forked at small clusters. This seems to be a direct consequence of RWMD, given that the performance of GV with SCS is more consistent, steady, and better.



(**a**) Gv + RWMD (**b**) d2v + CS

Figure 9. Task II. Clustering using Gv using RWMD as a distance does not improve over Gv and SCS, maybe on the contrary. On the right, we can see d2v + CS, which performs very well in general for all groups, both for small and large clusters. Colors refer to clusters assigned with a threshold automatically determined by the drawing algorithm.

The combination that behaves more consistently for small and big clusters is d2v + CS. In the experiments, only two errors can be spotted: one document from the Immersive Analytics set, placed in the Molecular and Biological Visualization cluster, and one document from the Molecular and Biological Visualization set, located in the Mobile Rendering cluster.

NRC is logical in big clusters but, on the other hand, less consistent in small clusters, as shown in Figure 10. It separates papers from the same sets over many levels, being the measure that exhibits a stronger inclination in this direction.

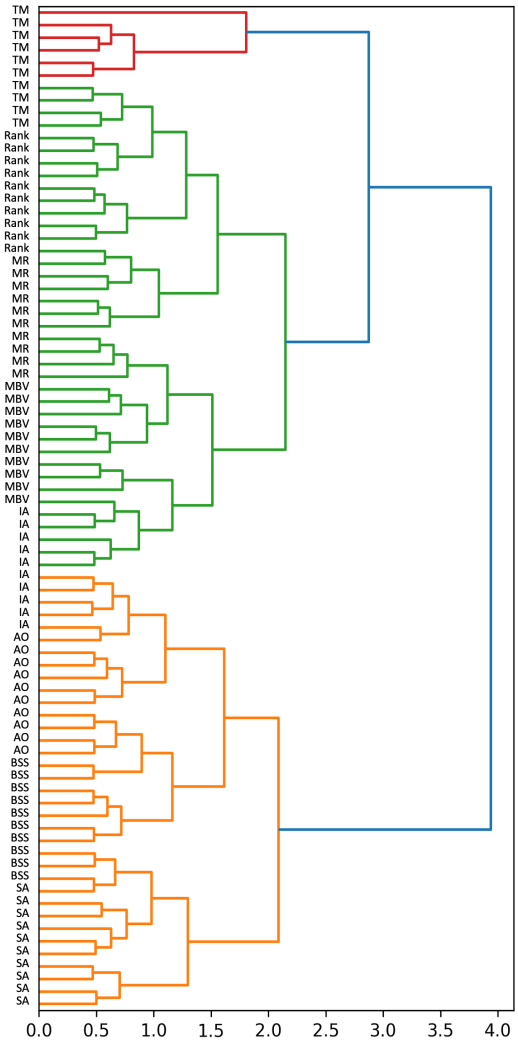


Figure 10. Task II–Clustering using NRC achieves average results at most: small groups are separated in several cases. Colors are assigned automatically by the plotting algorithm.

8. Discussion

The use of techniques from the Machine Learning set is sometimes justified by the results produced, ignoring, to a greater or lesser extent, formal or theoretical aspects. This is also because obtaining a curated corpus is often quite complex and time-consuming. The results presented throughout this chapter have shed light on the combination analyzed from both points of view: analytically and real-world scenarios.

Task I, which has provided a piece of more analytical information about the measures (and embeddings), has shown that all the measures, except CS, are good ones. In general, all of them perform well with some non-remarkable imperfections, but for the case of CS + d2v with the thesis set, the problems become relevant, and we, therefore, would be cautious in the use of such measure for very long documents.

With the information provided in Section 6, the best measures, from a theoretical standpoint, seem to be RWMD and WMD for our context; being nothing negligible to the results of SCS.

On the other hand, the executions times show how the use of ELMo could become unjustified because other embeddings provide better results with a lower time cost. Another remarkable example is RWMD, which, albeit relaxing a constraint over WMD, it still achieves good results. Furthermore, it also takes much less computation time.

The practical results, i.e., Task II, show examples of the mentioned casuistic at the beginning of this section. d2v + CS yielded a non very satisfactory result on very long

documents for Task I, but it turns out to be the best combination for Task II. It is the one with fewer clustering errors, at least for the used set, which the authors consider not trivial whatsoever.

Task II shows also how the embeddings impact the measures' performance. The GV cases have different performance in comparison with the rest of the embeddings, and it seems to be caused by its statistical base. That is to say, GV works well making large clusters, which is expected behavior for statistical-based techniques.

Concerning the other measures analyzed in Task II, i.e., SCS, RWMD, and NRC, the first two work similarly, being dependent on the embedding used. There are no clear arguments to opt for one or the other, except for the insignificant execution time of RWMD in comparison with the rest.

Finally, NRC, the out-of-the-box selected measure for completeness, has, in general, a good performance. It is extremely fast, at least in comparison with the other ones, and monotonously decreasing, as depicted previously in the literature. Despite that, the clustering is one of the worst, probably because it does not work at a very detailed level with the context, as it seems to happen with the embeddings.

All these results are summarized in a compact form in Table 5.

Table 5. Summary of the results. "+" means good results, "∼" average results, and "-" bad results. Results that were not possible to evaluate (e.g., the calculation took too long) are indicated by "?". The combination of d2v and CS is labelled as good because the results with the papers are very positive. However, for very long documents, it exhibited a bumpy decrease instead of a monotonic behavior (therefore we marked it with a "*". However, for the clustering case, its behavior is superior to Gv + RWMD, though the latter also works acceptably.

Distance	**SCS**			**WMD**			**RWMD**			**CS**		**NRC**
Embedding	**w2v**	**fT**	**GV**	**w2v**	**fT**	**GV**	**w2v**	**fT**	**GV**	**d2v**	**ELMo**	
Task 1–Article/thesis comparison	+	+	+	∼	∼	∼	∼	∼	∼	+*	-	∼
Task 2–Clustering	∼	∼	+		?		∼	∼	∼	+	?	∼

9. Conclusions and Future Work

In this paper, we have presented a deep and systematic analysis of several word embeddings and distance metrics used in the literature. We covered most of the recently developed techniques, such as ELMo, Fast Text, and Global Vectors, together with more popular techniques such as Word2vec. Since we wanted to analyze real-world practical scenarios where high performing computing infrastructure might not be available, we discarded some techniques presented recently, which, in most cases, require huge power consumption. As a result, many of these have also a substantial pollution impact for the model training. For example, Liu et al. [28], pretrained BERT for improved results, and used 1024 V100 GPUs for approximately one day.

For the experiments, we concentrated on research documents in PDF format. These were processed with the common pipeline used in Natural Language Processing that involves generating the plain text, cleaning strange characters commonly generated by the PDF-to-text translators, and stemming.

We carried out two experiments, one intended to more formally evaluate the distance metrics, and a second one intended to mimic a real-world problem, clustering documents according to their similarity.

The results show that not all combinations of embeddings and distances are equally suitable. A notable result is a fact that some techniques have a huge cost, which makes them impractical, while others perform equally, even yielding superior results, albeit incurring a lower cost. From the developed techniques, the combination of d2v embedding with CS similarity seems to be the most robust for both problems, except for extremely large documents such as a Ph.D. thesis.

Though the project presented here has produced several useful insights, there is more work that can be performed from here. First, it would be interesting to curate a larger set of documents that can be used for further experiments. Second, it would be interesting to further investigate the problems of d2v + CS with very long documents. Moreover, other tasks beyond clustering could also be addressed.

In this paper, we addressed an empirical evaluation with the most popular and well-known techniques in the literature. However, some other methods of document representations are being presented now, in most cases, only in arXiv (e.g., [38]). In the future, we would like to study these new techniques, especially the ones that achieve a certain degree of popularity, and test them against the other techniques evaluated here.

Author Contributions: Conceptualization, P.-P.V.; methodology, J.G. and P.-P.V.; software, J.G.; validation, J.G. and P.-P.V.; investigation, J.G. and P.-P.V.; data curation, J.G. and P.-P.V.; writing—original draft preparation, J.G. and P.-P.V.; writing—review and editing, J.G. and P.-P.V.; visualization, J.G.; supervision, P.-P.V.; funding acquisition, P.-P.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Project TIN2017-88515-C2-1-R funded by Ministerio de Economía y Competitividad, under MCIN/AEI/10.13039/501100011033/FEDER "A way to make Europe".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Articles are described in the appendix and can be obtained through the public research website of our university: https://upcommons.upc.edu/ or the publishers' websites

Acknowledgments: The authors want to thank the personnel at the UPC library for providing the initial dataset.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BERT	Bidirectional Encoder Representations from Transformer
BoW	Bag-of-Words
CBOW	Continuous Bag-of-Words
CS	Cosine Similarity
CSG	Continuous Skip–Gram Model
d2v	doc2vec embedding
ELMo	Embeddings from Language Models
EMD	Earth Mover's Distance
fT	Fast Text embedding
GV	Global Vectors embedding
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
NLP	Natural Language Processing
NNLM	Neural Network Language Models
NRC	Normalized Relative Compression
RNN	Recurrent Neural Network
RWMD	Relaxed Words Mover's Distance
SCS	Soft Cosine Similarity
WMD	Words Mover's Distance
VSM	Vector Space Model

Appendix A. Training Details

For the training phase, we have used several methods of different libraries, that are detailed here for reproducibility. As mentioned earlier, w2v and fT training work in the

same way, just changing the training class. We need to read the corpus using an iterator class that processes the corpus of documents, and extract a list of tokenized sentences from it.

The split of the documents in text is performed using the function sent_tokenize() of the *NLTK* (Available at http://www.nltk.org (accessed on 1 March 2022).) library, and the tokenization of each sentence is carried out by the simple_process() function of the *Gensim* library. This list is the direct training input for both models, constructing after that a vocabulary using the function build_vocab() of the training classes.

The training scheme for d2v is essentially the same as for the previous ones but, instead of listing the sentences extracted from documents, all the documents are tokenized with simple_process(), tagged using the class TaggedDocument() and added to a list. This used method, from the ones available, is slower but efficient in terms of memory.

Since GV has no implementation in *Gensim*, we chose the original implementation (Available at https://nlp.stanford.edu/projects/glove/ (accessed on 1 March 2022).). This implementation, coded in C, requires the data formatted specially, i.e., all the corpora in a single file with all the words separated by one or more spaces, or tabs, and the different documents separated by new line characters. This is achieved by coding a script that reads the preprocessed files and assembles all the documents in a single file.

The training of the GV model just requires the described file, since the implementation is accompanied by a training script that makes all the process steps, i.e., creation of the vocabulary counts, the building of the co-occurrence matrix, and model training. To use the GV model, a conversion to a *Gensim* format is required. This can be carried out using the function glove2word2vec() from the *Gensim* library.

Using *spaCy* for RWMD poses a problem: it uses a specific pipeline that requires a conversion from the standard representation of the word2vec vectors to a special one. In addition, *spaCy* has no explicit representation for fT and GV. Our workaround consisted in transforming the data to word2vec. Since this conversion conserves the vectors and their relations with the word, everything should work fine. This format conversion can be performed saving the *Gensim* model as plain text, using the function save_word2vec_format() from the models class. Then, after compression, *spaCy* has an API (Available at https://spacy.io/usage/vectors-similarity accessed on 15 May 2022) which, after consuming the compressed file, returns the embedding in a proper format. For the distance computation, *spaCy* works with models interpreting them as Language Processing pipelines, on which new steps can be added. After adding the measure at the end of the pipeline, the distance computation must be performed passing one document through the pipeline and, with the returned object, perform a call to the distance computation, passing to it another call to the pipeline, to obtain an object for the second document.

Appendix B. Papers Used in Task I

As said before, for this task we have a base paper, and several papers that will be compared to it. The base paper, randomly picked, is "Dynamic response of the MICA runner. Experiment and simulation" by Valentín et al. from MCIA.

From the same group, the following papers have been randomly picked:

- CFID1: "Thermo-fluid numerical simulation of the crotch absorbers' cooling pinholes for alba storage ring" by Escaler et al.
- CFID2: "Failure investigation of a Francis turbine under the cavitation conditions" by Liu et al.
- CFID3: "Condition monitoring of pump-turbines. New challenges" by Egusquiza et al.
- CFID4: "Transmission of high frequency vibrations in rotating systems. Application to cavitation detection in hydraulic turbines" by Valentín et al.
- CFID5: "Experimental mode shape determination of a cantilevered hydrofoil under different flow conditions" by De La Torre et al.

From different groups, the following papers have been randomly selected:

- ANT–"The 234U neutron capture cross section measurement at the n TOF facility" by Lampoudis et al.
- ADR&M–"Turning barriers into alleyways: unsolved transitions from old Barcelona to the post-Cerdà city" by Millán et al.
- UMA–"What to be considered when you buy a sprayer: the SPISE advice" by Gil et al.
- AIEM–"A Framework for Structural Systems Based on the Principles of Statistical Mechanics" by Andújar et al.
- IDEAI–"A model for continuous monitoring of patients with major depression in short and long term periods" by Múgica et al.

Appendix C. Papers Used in Task II

The sets of papers used in Task II (Section 7) are listed below.

Appendix C.1. Ambient Occlusion

- McGuire, M., Osman, B., Bukowski, M., & Hennessy, P. (2011, August). The alchemy screen-space ambient obscurance algorithm. In Proceedings of the ACM SIGGRAPH Symposium on High-Performance Graphics (pp. 25–32).
- Zhang, D., Xian, C., Luo, G., Xiong, Y., & Han, C. (2020). DeepAO: Efficient screen space ambient occlusion generation via deep network. IEEE Access, 8, 64434–64441.
- Diaz, J., Vazquez, P. P., Navazo, I., & Duguet, F. (2010). Real-time ambient occlusion and halos with summed area tables. Computers & Graphics, 34(4), 337–350.
- Schott, M., Pegoraro, V., Hansen, C., Boulanger, K., & Bouatouch, K. (2009, June). A directional occlusion shading model for interactive direct volume rendering. In Computer Graphics Forum (Vol. 28, No. 3, pp. 855-862). Oxford, UK: Blackwell Publishing Ltd.
- Doronin, O., Kara, P. A., Barsi, A., & Martini, M. G. (2017). Screen-space ambient occlusion for light field displays.
- Umenhoffer, T., Tóth, B., & Szirmay-Kalos, L. (2009, April). Efficient methods for ambient lighting. In Proceedings of the 25th Spring Conference on Computer Graphics (pp. 87–94).
- Ropinski, T., Kasten, J., & Hinrichs, K. (2008). Efficient shadows for gpu-based volume raycasting.
- Reinbothe, C. K., Boubekeur, T., & Alexa, M. (2009). Hybrid Ambient Occlusion. Eurographics (Areas Papers), 5.
- Luft, T., Colditz, C., & Deussen, O. (2006). Image enhancement by unsharp masking the depth buffer. ACM Transactions on Graphics (TOG), 25(3), 1206–1213.
- Bauer, F., Knuth, M., Kuijper, A., & Bender, J. (2013, November). Screen-space ambient occlusion using a-buffer techniques. In 2013 International Conference on Computer-Aided Design and Computer Graphics (pp. 140–147). IEEE.

Appendix C.2. Immersive Analytics

- Goddard, T. D., Brilliant, A. A., Skillman, T. L., Vergenz, S., Tyrwhitt-Drake, J., Meng, E. C., & Ferrin, T. E. (2018). Molecular visualization on the holodeck. Journal of molecular biology, 430(21), 3982–3996.
- Richardson, M., Jacoby, D., & Coady, Y. (2018, November). Retrofitting Realities: Affordances and Limitations in Porting an Interactive Geospatial Visualization from Augmented to Virtual Reality. In 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 1081–1087). IEEE.
- Billinghurst, M., Cordeil, M., Bezerianos, A., & Margolis, T. (2018). Collaborative immersive analytics. In Immersive Analytics (pp. 221–257). Springer, Cham.
- Liu, J., Dwyer, T., Marriott, K., Millar, J., & Haworth, A. (2017). Understanding the relationship between interactive optimization and visual analytics in the context of prostate brachytherapy. IEEE transactions on visualization and computer graphics, 24(1), 319–329.

- Marai, G. E., Leigh, J., & Johnson, A. (2019). Immersive analytics lessons from the electronic visualization laboratory: a 25-year perspective. IEEE computer graphics and applications, 39(3), 54–66.
- Cordeil, M., Cunningham, A., Bach, B., Hurter, C., Thomas, B. H., Marriott, K., & Dwyer, T. (2019, March). IATK: An immersive analytics toolkit. In 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR) (pp. 200–209). IEEE.
- Dwyer, T., Marriott, K., Isenberg, T., Klein, K., Riche, N., Schreiber, F., ... & Thomas, B. H. (2018). Immersive analytics: An introduction. In Immersive analytics (pp. 1–23). Springer, Cham.
- Tadeja, S. K., Kipouros, T., & Kristensson, P. O. (2019, May). Exploring parallel coordinates plots in virtual reality. In Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (pp. 1–6).
- Gračanin, D. (2018, July). Immersion versus embodiment: Embodied cognition for immersive analytics in mixed reality environments. In International Conference on Augmented Cognition (pp. 355–368). Springer, Cham.

Appendix C.3. Mobile Rendering

- Noon, C. J. (2012). A volume rendering engine for desktops, laptops, mobile devices and immersive virtual reality systems using GPU-based volume raycasting. Iowa State University.
- Noguera, J. M., Jiménez, J. R., Ogáyar, C. J., & Segura, R. J. (2012, February). Volume Rendering Strategies on Mobile Devices. In GRAPP/IVAPP (pp. 447–452).
- Pattrasitidecha, A. (2014). Comparison and evaluation of 3D mobile game engines (Master's thesis).
- Noguera, J. M., & Jimenez, J. R. (2015). Mobile volume rendering: past, present and future. IEEE transactions on visualization and computer graphics, 22(2), 1164–1178.
- Schultz, C., & Bailey, M. (2016). Interacting with Large 3D Datasets on a Mobile Device. IEEE computer graphics and applications, 36(5), 19-23.
- Xin, Y., & Wong, H. C. (2016, October). Intuitive volume rendering on mobile devices. In 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI) (pp. 696–701). IEEE.
- Lee, W. J., Hwang, S. J., Shin, Y., Ryu, S., & Ihm, I. (2016). Adaptive multi-rate ray sampling on mobile ray tracing GPU. In SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications (pp. 1–6).
- Hachaj, T. (2014). Real-time exploration and management of large medical volumetric datasets on small mobile devices—evaluation of remote volume rendering approach. International Journal of Information Management, 34(3), 336–343.
- Heller, F., Jevanesan, J., Dietrich, P., & Borchers, J. (2016, September). Where are we? evaluating the current rendering fidelity of mobile audio augmented reality systems. In Proceedings of the 18th International Conference on Human–Computer Interaction with Mobile Devices and Services (pp. 278–282).
- Lee, W. J., Hwang, S. J., Shin, Y., Yoo, J. J., & Ryu, S. (2017, January). Fast stereoscopic rendering on mobile ray tracing GPU for virtual reality applications. In 2017 IEEE International Conference on Consumer Electronics (ICCE) (pp. 355–357). IEEE.

Appendix C.4. Molecular and Biological Visualization

- Hülsmann, M., Köddermann, T., Vrabec, J., & Reith, D. (2010). GROW: A gradient-based optimization workflow for the automated development of molecular models. Computer Physics Communications, 181(3), 499–513.
- Reith, D., & Kirschner, K. N. (2011). A modern workflow for force-field development–Bridging quantum mechanics and atomistic computational models. Computer Physics Communications, 182(10), 2184–2191.

- Hanwell, M. D., Curtis, D. E., Lonie, D. C., Vandermeersch, T., Zurek, E., & Hutchison, G. R. (2012). Avogadro: an advanced semantic chemical editor, visualization, and analysis platform. Journal of cheminformatics, 4(1), 1–17.
- Goodstadt, M., & Marti-Renom, M. A. (2017). Challenges for visualizing three-dimensional data in genomic browsers. FEBS letters, 591(17), 2505–2519.
- Stone, M. (2012). In color perception, size matters. IEEE Computer Graphics and Applications, March/April 2012.
- Knoll, A., Wald, I., Navrátil, P. A., Papka, M. E., & Gaither, K. P. (2013, November). Ray tracing and volume rendering large molecular data on multicore and many-core architectures. In Proceedings of the 8th International Workshop on Ultrascale Visualization (pp. 1–8).
- Mohammed, H., Al-Awami, A. K., Beyer, J., Cali, C., Magistretti, P., Pfister, H., & Hadwiger, M. (2017). Abstractocyte: A visual tool for exploring nanoscale astroglial cells. IEEE transactions on visualization and computer graphics, 24(1), 853–861.
- Bezerianos, A., Isenberg, P., Chapuis, O., & Willett, W. (2013, April). Perceptual affordances of wall-sized displays for visualization applications: Color. In Proceedings of the CHI Workshop on Interactive, Ultra-High-Resolution Displays (PowerWall).
- Schatz, K., Krone, M., Pleiss, J., & Ertl, T. (2019). Interactive visualization of biomolecules' dynamic and complex properties. The European Physical Journal Special Topics, 227(14), 1725–1739.
- Xiao-Zhong Shen. (2010). Comparison of protein surface visualization techniques in different environments (tools). INF358 Seminar in Visualization.
- Xu, C., Liu, Y. P., Jiang, Z., Sun, G., Jiang, L., & Liang, R. (2020). Visual interactive exploration and clustering of brain fiber tracts. Journal of Visualization, 23(3), 491–506.

Appendix C.5. Rankings Visualization

- Han, D., Pan, J., Guo, F., Luo, X., Wu, Y., Zheng, W., & Chen, W. (2019). Rankbrushers: Interactive analysis of temporal ranking ensembles. Journal of Visualization, 22(6), 1241–1255.
- Gratzl, S., Lex, A., Gehlenborg, N., Pfister, H., & Streit, M. (2013). Lineup: Visual analysis of multi-attribute rankings. IEEE transactions on visualization and computer graphics, 19(12), 2277–2286.
- Wall, E., Das, S., Chawla, R., Kalidindi, B., Brown, E. T., & Endert, A. (2017). Podium: Ranking data using mixed-initiative visual analytics. IEEE transactions on visualization and computer graphics, 24(1), 288–297.
- Bačík, V., & Klobučník, M. (2018). Possibilities of using selected visualization methods for historical analysis of sporting event–an example of stage cycling race Tour de France.
- Batty, M. (2006). Rank clocks. Nature, 444(7119), 592–596.
- Shi, C., Cui, W., Liu, S., Xu, P., Chen, W., & Qu, H. (2012). Rankexplorer: Visualization of ranking changes in large time series data. IEEE Transactions on Visualization and Computer Graphics, 18(12), 2669–2678.
- Miranda, F., Lins, L., Klosowski, J. T., & Silva, C. T. (2017). Topkube: A rank-aware data cube for real-time exploration of spatiotemporal data. IEEE Transactions on visualization and computer graphics, 24(3), 1394–1407.
- Gousie, M. B., Grady, J., & Branagan, M. (2014, February). Visualizing trends and clusters in ranked time-series data. In Visualization and Data Analysis 2014 (Vol. 9017, p. 90170F). International Society for Optics and Photonics.
- Lei, H., Xia, J., Guo, F., Zou, Y., Chen, W., & Liu, Z. (2016). Visual exploration of latent ranking evolutions in time series. Journal of Visualization, 19(4), 783–795.
- Chen, Y., Xu, P., & Ren, L. (2017). Sequence synopsis: Optimize visual summary of temporal event data. IEEE transactions on visualization and computer graphics, 24(1), 45–55.

Appendix C.6. Bicycle Sharing Systems

- Feng, Y., Affonso, R. C., & Zolghadri, M. (2017). Analysis of bike sharing system by clustering: the Vélib'case. IFAC-PapersOnLine, 50(1), 12422–12427.
- Midgley, P. (2011). Bicycle-sharing schemes: enhancing sustainable mobility in urban areas. United Nations, Department of Economic and Social Affairs, 8, 1–12.
- Frade, I., & Ribeiro, A. (2014). Bicycle sharing systems demand. Procedia-Social and Behavioral Sciences, 111, 518–527.
- How land-use and urban form impact bicycle flows: Evidence from the bicycle-sharing system (BIXI) in Montreal
- Younes, H., Zou, Z., Wu, J., & Baiocchi, G. (2020). Comparing the temporal determinants of dockless scooter-share and station-based bike-share in Washington, DC. Transportation Research Part A: Policy and Practice, 134, 308–320.
- Saas, A., Guitart, A., & Periánez, A. (2016, September). Discovering playing patterns: Time series clustering of free-to-play game data. In 2016 IEEE Conference on Computational Intelligence and Games (CIG) (pp. 1–8). IEEE.
- O'brien, O., Cheshire, J., & Batty, M. (2014). Mining bicycle sharing data for generating insights into sustainable transport systems. Journal of Transport Geography, 34, 262–273.
- Froehlich, J. E., Neumann, J., & Oliver, N. (2009, June). Sensing and predicting the pulse of the city through shared bicycling. In Twenty-First International Joint Conference on Artificial Intelligence.
- Boufidis, N., Nikiforiadis, A., Chrysostomou, K., & Aifadopoulou, G. (2020). Development of a station-level demand prediction and visualization tool to support bike-sharing systems' operators. Transportation Research Procedia, 47, 51–58.
- Wood, J., Beecham, R., & Dykes, J. (2014). Moving beyond sequential design: Reflections on a rich multichannel approach to data visualization. IEEE transactions on visualization and computer graphics, 20(12), 2171–2180.

Appendix C.7. Sports Analytics

- Perin, C., Vuillemot, R., Stolper, C. D., Stasko, J. T., Wood, J., & Carpendale, S. (2018, June). State of the art of sports data visualization. In Computer Graphics Forum (Vol. 37, No. 3, pp. 663–686).
- Chung, D. H., Parry, M. L., Griffiths, I. W., Laramee, R. S., Bown, R., Legg, P. A., & Chen, M. (2015). Knowledge-assisted ranking: A visual analytic application for sports event data. IEEE Computer Graphics and Applications, 36(3), 72–82.
- Grignard, A., Macià, N., Alonso Pastor, L., Noyman, A., Zhang, Y., & Larson, K. (2018, July). Cityscope andorra: a multi-level interactive and tangible agent-based visualization. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (pp. 1939–1940).
- Dietrich, C., Koop, D., Vo, H. T., & Silva, C. T. (2014, October). Baseball4d: A tool for baseball game reconstruction & visualization. In 2014 IEEE Conference on Visual Analytics Science and Technology (VAST) (pp. 23–32). IEEE.
- Beecham, R., & Wood, J. (2014). Characterising group-cycling journeys using interactive graphics. Transportation Research Part C: Emerging Technologies, 47, 194–206.
- Pileggi, H., Stolper, C. D., Boyle, J. M., & Stasko, J. T. (2012). Snapshot: Visualization to propel ice hockey analytics. IEEE Transactions on Visualization and Computer Graphics, 18(12), 2819–2828.
- Carlis, J. V., & Konstan, J. A. (1998, November). Interactive visualization of serial periodic data. In Proceedings of the 11th annual ACM symposium on User interface software and technology (pp. 29–38).
- Dendir, S. (2016). When do soccer players peak? A note. Journal of Sports Analytics, 2(2), 89–105.
- Bruce, S. (2016). A scalable framework for NBA player and team comparisons using player tracking data. Journal of Sports Analytics, 2(2), 107–119.

- McFarlane, P. (2019). Evaluating NBA end-of-game decision-making. Journal of Sports Analytics, 5(1), 17–22.

Appendix C.8. Analysis and Visualization of Transportation Methods

- Woodcock, J., Tainio, M., Cheshire, J., O'Brien, O., & Goodman, A. (2014). Health effects of the London bicycle sharing system: health impact modelling study. Bmj, 348.
- Xie, X. F., & Wang, Z. J. (2015). An empirical study of combining participatory and physical sensing to better understand and improve urban mobility networks (No. 15-3238).
- Shi, X., Wang, Y., Lv, F., Liu, W., Seng, D., & Lin, F. (2019). Finding communities in bicycle sharing system. Journal of Visualization, 22(6), 1177–1192.
- Shi, X., Lv, F., Seng, D., Xing, B., & Chen, B. (2019). Visual exploration of mobility dynamics based on multi-source mobility datasets and POI information. Journal of Visualization, 22(6), 1209-1223.
- Corcoran, J., Li, T., Rohde, D., Charles-Edwards, E., & Mateo-Babiano, D. (2014). Spatio-temporal patterns of a Public Bicycle Sharing Program: the effect of weather and calendar events. Journal of Transport Geography, 41, 292–305.
- Zaltz Austwick, M., O'Brien, O., Strano, E., & Viana, M. (2013). The structure of spatial networks and communities in bicycle sharing systems. PloS one, 8(9), e74685.
- Beecham, R., Wood, J., & Bowerman, A. (2014). Studying commuting behaviours using collaborative visual analytics. Computers, Environment and Urban Systems, 47, 5–15.
- SEo, J., Shneiderman, B. (2001). Understanding hierarchical clustering results by interactive exploration of dendrograms: A case study with Genomic Microarray Data", Technical Report, Human–Computer Interaction Laboratory, Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 USA
- Oliveira, G. N., Sotomayor, J. L., Torchelsen, R. P., Silva, C. T., & Comba, J. L. (2016). Visual analysis of bike-sharing systems. Computers & Graphics, 60, 119–129.

References

1. Rydning, D.R.J.G.J. *The Digitization of the World from Edge to Core*; International Data Corporation: Framingham, MA, USA, 2018; p. 16.
2. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 NAACL HLT, Vol 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
3. Cilibrasi, R.; Vitányi, P.M. Clustering by compression. *IEEE Trans. Inf. Theory* **2005**, *51*, 1523–1545. [CrossRef]
4. Grnarova, P.; Schmidt, F.; Hyland, S.L.; Eickhoff, C. Neural Document Embeddings for Intensive Care Patient Mortality Prediction. *arXiv* **2016**, arXiv:1612.00467.
5. Zhang, W.E.; Sheng, Q.Z.; Lau, J.H.; Abebe, E. Detecting duplicate posts in programming QA communities via latent semantics and association rules. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 1221–1229.
6. Strubell, E.; Ganesh, A.; McCallum, A. Energy and Policy Considerations for Deep Learning in NLP. *arXiv* **2019**, arXiv:1906.02243.
7. Alvarez, J.E.; Bast, H. A Review of Word Embedding and Document Similarity Algorithms Applied to Academic Text. Bachelor's Thesis, University of Freiburg, Breisgau, Germany, 2017.
8. Dai, A.M.; Olah, C.; Le, Q.V. Document embedding with paragraph vectors. *arXiv* **2015**, arXiv:1507.07998.
9. Shahmirzadi, O.; Lugowski, A.; Younge, K. Text Similarity in Vector Space Models: A Comparative Study. In Proceedings of the IEEE-ICMLA (18th International Conference on Machine Learning and Applications 2019), Boca Raton, FL, USA, 16–19 December 2019.
10. Vázquez, P.P. Visual analysis of research paper collections using normalized relative compression. *Entropy* **2019**, *21*, 612. [CrossRef]
11. Lau, J.H.; Baldwin, T. An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation. *arXiv* **2016**, arXiv:1607.05368.
12. Arora, S.; Liang, Y.; Ma, T. A simple but tough-to-beat baseline for sentence embeddings. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.
13. Kiros, R.; Zhu, Y.; Salakhutdinov, R.; Zemel, R.S.; Torralba, A.; Urtasun, R.; Fidler, S. Skip-thought vectors. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Volume 2, pp. 3294–3302.

14. Wieting, J.; Bansal, M.; Gimpel, K.; Livescu, K. Towards Universal Paraphrastic Sentence Embeddings. In Proceedings of the 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May 2016.
15. Le, Q.V.; Mikolov, T. Distributed Representations of Sentences and Documents. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1188–1196.
16. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
17. Pennington, J.; Socher, R.; Manning, C. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543. [CrossRef]
18. Levy, O.; Goldberg, Y.; Dagan, I. Improving distributional similarity with lessons learned from word embeddings. *Trans. Assoc. Comput. Linguist.* **2015**, *3*, 211–225. [CrossRef]
19. Baroni, M.; Dinu, G.; Kruszewski, G. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 22–27 June 2014; pp. 238–247.
20. Naili, M.; Chaibi, A.H.; Ghezala, H.H.B. Comparative study of word embedding methods in topic segmentation. *Procedia Comput. Sci.* **2017**, *112*, 340–349. [CrossRef]
21. Deerwester, S.; Dumais, S.T.; Furnas, G.W.; Landauer, T.K.; Harshman, R. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.* **1990**, *41*, 391–407. [CrossRef]
22. Chen, M. Efficient Vector Representation for Documents through Corruption. ICLR (Poster), 2017. Available online: https://arxiv.org/abs/1707.02377 (accessed on 1 March 2022).
23. Kusner, M.; Sun, Y.; Kolkin, N.; Weinberger, K. From word embeddings to document distances. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 957–966.
24. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 1st ICLR, Scottsdale, Arizona, USA, 2–4 May 2013.
25. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
26. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep Contextualized Word Representations. *arXiv* **2018**, arXiv:1802.05365.
27. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.
28. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
29. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942.
30. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv* **2019**, arXiv:1901.02860.
31. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv* **2020**, arXiv:1906.08237.
32. Beltagy, I.; Peters, M.E.; Cohan, A. Longformer: The long-document transformer. *arXiv* **2020**, arXiv:2004.05150.
33. Reimers, N.; Gurevych, I.; Reimers, N.; Gurevych, I.; Thakur, N.; Reimers, N.; Daxenberger, J.; Gurevych, I.; Reimers, N.; Gurevych, I.; et al. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Hong Kong, China, 3–7 November 2019; pp. 671–688.
34. Sidorov, G.; Gelbukh, A.; Gómez-Adorno, H.; Pinto, D. Soft similarity and soft cosine measure: Similarity of features in vector space model. *Comput. Sist.* **2014**, *18*, 491–504. [CrossRef]
35. Rubner, Y.; Tomasi, C.; Guibas, L.J. A metric for distributions with applications to image databases. In Proceedings of the Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271), Bombay, India, 7 January 1998; pp. 59–66.
36. Pinho, A.J.; Pratas, D.; Ferreira, P.J. Authorship attribution using relative compression. In Proceedings of the 2016 Data Compression Conference (DCC), Snowbird, UT, USA, 30 March–1 April 2016; pp. 329–338.
37. Sokal, R.R.; Rohlf, F.J. The comparison of dendrograms by objective methods. *Taxon* **1962**, *11*, 33–40. [CrossRef]
38. Wu, L.; Yen, I.E.; Xu, K.; Xu, F.; Balakrishnan, A.; Chen, P.Y.; Ravikumar, P.; Witbrock, M.J. Word mover's embedding: From word2vec to document embedding. *arXiv* **2018**, arXiv:1811.01713.