Finding the most similar textual documents using Case-Based Reasoning

Marko Mihajlović*, Prof. Dr. Ning Xiong†
Academy of Innovation, Design, and Engineering, Mälardalen University
Västerås, Sweden
Email: *marko.mihajlovic@elfak.rs, †ning.xiong@mdh.se,

***Abstract*—In recent years, huge amounts of unstructured textual data on the Internet is a big difficulty for AI algorithms to provide best recommendations for users and their search queries. Since the Internet became widespread, a lot of research has been done in the field of Natural Language Processing (NLP) and machine learning. Almost every solution transforms documents into Vector Space Models (VSM) in order to apply AI algorithms over them. One such approach is based on the Case-Based Reasoning (CBR). Therefore, the most important part of those systems is to compute the similarity between numerical data points. In 2016, the new similarity TS-SS metric is proposed, which showed state-of-the-art results in the field of textual mining for unsupervised learning. However, no one before has investigated its performances for supervised learning (classification task). In this paper, we devised a CBR system capable of finding the most similar documents for a given query aiming to investigate performances of the new state-of-the-art metric, TS-SS, and several other geometrical similarity measures — Euclidean distance, and Cosine similarity — that showed the best predictive results over several benchmark corpora. The results show surprising inappropriateness of TS-SS measure in the proposed setting.**

***Index Terms*—CBR, machine learning, NLP, similarity measures, AI**

I. Introduction

Nowadays, almost every person in the World generates data on the Internet; social media, news, public comments, blogs, searching the Internet, etc. All this information is recorded in a database which makes a problem known as Big Data [1]. Accordingly, vast amounts of data are unstructured textual data which makes inconvenience for machine learning models to harness their predictive power.

Consequently, diverse approaches and algorithms have been proposed to deal with this issue. However, most of them work well only with numerical data. To use those algorithms, we need to convert textual data into numerical feature vectors. Then, AI algorithms are fed with those data in order to learn the important features that lead the final successful prediction. But first, before the whole system is implemented, we need a textual corpus to show whether our approach is better than the available ones. After the data is acquired, it needs to be preprocessed, which means that the original text is altered in a way to be more suitable for further use - reducing vocabulary, outliers, and other impurities. Such a modified text is then ready for feature engineering. This process uses statistics to construct numerical feature vectors from raw textual data. In data mining and information retrieval this procedure of converting text into a numerical vector is known as Vector Space Model (VSM) [2]; a set of linearly independent basis vectors that represent textual documents. Afterwards, popular approaches, such as Case-Based Reasoning (CBR) [3], can be used to find the most similar documents based on already seen cases (training data). The system's recommendation for the new document can be evaluated by an expert and added to the pool of training data in order to further improve the system's predictive performances. However, to harness the power of CBR system, we need to construct a similarity metric that can capture the important characteristics of feature vectors.

The major contribution of this work is to investigate the performances of the similarity function TS-SS (Triangle's area Similarity - Sector's area Similarity) [4], proposed by Heidarian and Dinneen, that has shown state-of-the-art performances for document clustering. The results of this algorithms are compared with the other well-known similarity measures, Euclidean Distance (ED) and Cosine Similarity (CS). Also, we provide a theoretical justification why TS-SS measure is incapable of recognizing the important characteristics between data points.

Section II discusses state-of-the-art methods and related work done in the field of information retrieval and data mining for each of the stages in the process. Section III describes an implementation of our system and methods for performance evaluation. Section IV has performance measurements of our implementation for a variety of similarity metrics. Section V explores the reasons and provides theoretical justification for achieved performances. Finally, a short summary of our research and future work is given in Section VI.

II. Related Work

Many systems have been devised to overcome difficulties with recognizing the most similar textual documents. This process includes several independent steps, and a lot of research has been done in each of these steps to improve systems' predictive performances.

STANDARDIZED DATASETS

The first step required by any machine learning system is to find a benchmark dataset for performance evaluation. According to Larson, standard test collections for information

†Supervisor

retrieval are *The Cranfield collection*, *NTCIR*, *Reuters Corpus*, *20 NewsGroups*, and several other corpora [5].

DATA PREPROCESSING

Before the features are extracted from the raw textual data, the textual data should be altered in order to reduce the vocabulary size and decrease inaccuracies in feature representation. This process can be briefly divided into 3 categories — dropping specific terms, word replacement, and stemming.

Some specific words do not bring any value, and they should be excluded from the dataset. This procedure mainly depends on textual corpora; for example, if the data includes web pages, then HTML tags should be removed, if it contains XML files, then XML tags should be eliminated, etc. Additionally, some words are contained only in a couple of documents, which make them too specific to be used in overall system. On the other side, some words too frequently occur in every document and they should be removed as well; for example, if all documents are about computer science, then term computer is irrelevant, and should be excluded from feature space. In our scenario, we are going to focus on regular textual data, so we will not further examine specific cases as with HTML and XML. Larson [5] recommends some procedures that are considered as a good practice — removing stop words, eliminating punctuation, making the word lowercase, and many other procedures which are described in detail in the implementation phase.

The purpose of the word replacement procedure is to reduce the vocabulary size. This process includes spelling correction, synonym replacement, and specific replacements. For spelling correction, it is widely used edit distance based on Levenshtein distance [6] to find a well spelled word. State-of-the-art approach for synonym replacement is based on WordNet [7]. Other corrections include simple word concatenation, number mapping and other procedures to overcome dataset peculiarities; for example, mapping mac book to *macbook*, every number to the one token, etc.

The next process is stemming – replacing each words with its base form. The Porter stemming algorithm [8] has been recommended by most authors for natural language processing tasks. For example, by applying the Porter stemmer, the word women will be replaced by woman, plays by play, etc.

FEATURE ENGINEERING

Several successful feature extraction methods for NLP tasks have been proposed and improved over the past decades. These methods map words or phrases from the vocabulary to vectors of real numbers. The most popular approaches for constructing feature vectors are: bag-of-words model (BoW) [9], tf-idf [10], Glove [11], and word2vec [12].

BoW method simply counts each word and its number of occurrences is recorded in feature vector at a specific position for that term. One obvious drawback of this approach is that it favors longer documents, therefore, tf-idf measure was introduced to overcome this problem by calculating product between term frequencies and inverse document frequency. Now inverse document frequency will decrease bias towards longer documents. Glove method also counts how frequently a word appears in a context/document, but it uses dimension reduction techniques to achieve low-dimension representation, hence feature vectors lose interpretability, whereas word2vec suffer from the same drawback. It is constructed by a neural network, which results in representing words/phrases as their probability distribution.

SIMILARITY METRICS

To find the most similar document, numerical feature vectors should be compared by calculating a similarity metric. A lot of metrics have been invented to capture the most important features of a vector. Those metrics can be divided into two subcategories: geometrical and non-geometrical methods. Summary of these approaches can be found in [4].

EVALUATION METHODS

A standard way of evaluating the quality of classification algorithms is based on confusion matrix, and derived measures from it, such as accuracy, precision, recall, and F_b score.

III. METHODS AND IMPLEMENTATION

Based on the previous work, the architecture of our system is devised accordingly. Fig. 1 depicts modular architecture of our system.

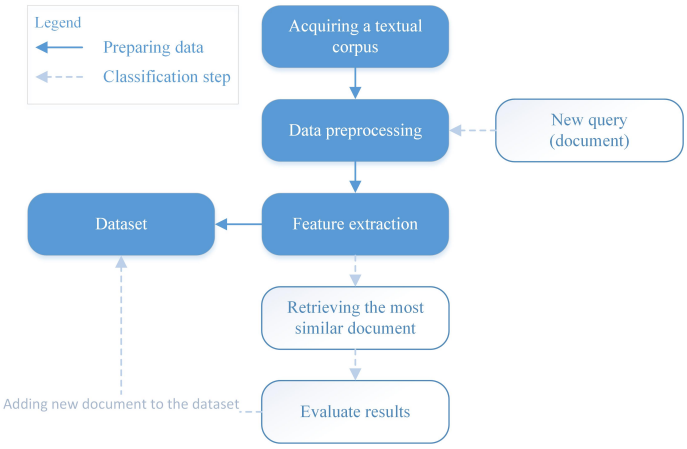

Fig. 1. Architecture of the implemented system

The first phase of our system is to acquire some training data, which will be used for searching similar documents. Those documents are then filtered by a preprocessing procedure in order to get rid of outliers and reduce the size of vocabulary. The next phase is to extract features from modified textual documents and to save them in order to accelerate the classification step. When the features are extracted, the most similar document is retrieved for a new query (document). The similarity between two vectors is calculated by several similarity measures. The outcome for each query is recorded and reviewed by an expert, then, the new document can be added to the pool of training documents, which will lead to the improvement in predictive performance for the future queries.

TABLE I
FEATURE EXTRACTION PARAMETERS

Parameters	Value
Minimum word length	3
Maximum document frequency	50%
Minimum document frequency	1%
Lowercase	True
Stop words	English
Analyzer	Only words
Feature extraction	tf-idf
Feature vector dimensionality	Different values are evaluated

STANDARDIZED DATASETS

The first step is to acquire some data. We used the 20 NewsGroups dataset which comprises of 18864 newsgroups posts on 20 topics, and Reuters dataset which contains 10,788 news documents on 90 topics. The reason for our choice is that these corpora are different in the number of documents and labels, which can reveal different characteristics of used similarity metrics. Both the datasets are split into training and test dataset.

DATA PREPROCESSING

After the data is acquired, it should be altered in order to reduce the size of vocabulary used for feature extraction. Different parameters for the procedures of altering the textual documents are summarized in the Table I.

The performances of our system deeply depend on the data preparation procedure. First, words that are shorter of three characters are eliminated. Then, based on the term frequency in documents, terms are kept or discarded. Those that are present too frequently — appear in 50% of the total amount of documents — and those that are too specific — 1% appearance — are eliminated. After this procedure, all characters are lowercase, and words that do not contain any valuable information, stop words, are removed.

In the next step, by observing the textual documents it is concluded that specific words need to be eliminated, in our example email and web addresses are also removed. The final step was to discard every character that is not a letter and to apply Porter stemmer to simplify word forms. It can be noted that some recommended procedures, such as mapping numbers to a specific token, are not implemented because achieving the best performances was not an aim of the project. The goal is to explore the behavior of different similarity metrics when finding the most similar documents.

FEATURE ENGINEERING

The data preprocessing procedure has reduced the size of vocabulary significantly. Now, the feature extraction method should convert each document into a numerical feature vector.

Vector space model based on *tf-idf* method usually outperforms other methods with the smaller amount of data.

For this purpose, *tf-idf* procedure is applied over the given training dataset. To evaluate the performances of different similarity metrics, different lengths of feature vectors are considered. Thus, forcing feature vectors to be of a fixed dimensionality. This is done by removing features with the smallest values. The aim of this experiment is to show how the system behaves in high-dimension space.

SIMILARITY METRICS

To retrieve the most similar document, a similarity between two documents should be calculated. Each of the documents is first converted to a numerical feature vector, then the similarity is calculated. In this work, we used three similarity measures – ED (1), CS (2), and TS-SS (5) - that showed state-of-the-art performances in many NLP tasks. All these metrics have useful geometrical representation and a short summary of drawbacks and advantages of these methods are well described in [4].

$$ED(x_i, x_j) = \sqrt{\sum_{k=1}^{M} (x_{ik} - x_{jk)^2}} \tag{1}$$

$$CS(x_i, x_j) = \frac{\sum_{k=1}^{M} x_{ik} x_{jk}}{\|x_i\| \, \|x_j\|} \tag{2}$$

$$TS(x_i, x_j) = \frac{\|x_i\| \, \|x_j\| \sin\theta'}{2} \tag{3}$$

$$SS(x_i, x_j) = \pi(ED(x_i, x_j) + |\|x_i\| - \|x_j\||)^2 \frac{\theta'}{360} \tag{4}$$

$$TS\text{-}SS(x_i, x_j) = TS(x_i, x_j) SS(x_i, x_j) \tag{5}$$

$$\theta'(x_i, x_j) = \arccos CS(x_i, x_j) + 10 \tag{6}$$

The reason for introducing a novel similarity measure, TS-SS is justified by the weaknesses of Euclidean distance and cosine similarity. The drawback of ED can be illustrated in 2-dimensional space if two documents are represented by two vectors, v_1, v_2, in Euclidean space, and both vectors are shifted for $-v_{1x}$ and $-v_{1y}$ in x and y direction respectively, $v1$ will at the origin, and $v2$ will be at the distance d from the origin. Then every point on the circle drawn by diameter d, will be equally similar as every other point on the circle's boundary, no matter angle. On the other side, cosine similarity does not suffer from this issue because it only considers the angle between two given vectors. However, the problem with cosine similarity is that it does not consider the magnitude of vectors. To address this weaknesses, vector magnitude and angle between two vectors, TS-SS measure was proposed. This measure is calculated as a product between Triangle's Area Similarity (TS) and Sector's Area Similarity (SS). The former is calculated based on the Euclidean surface between two vectors, but TS suffer from the similar problem as ED. Hence SS was introduced to overcome this, it calculates another surface of a triangle that is constructed by considered ED with vector magnitude and the angle between those two vectors. Now, the product of TS and SS should perform better than ED and cosine similarity by addressing drawbacks of those metrics.

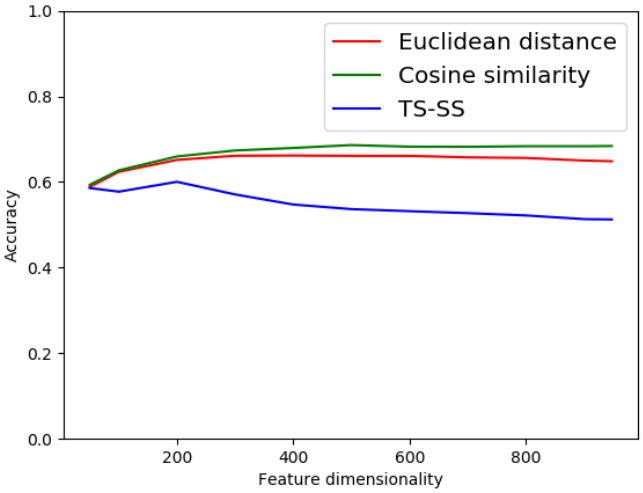

Fig. 2. System's performances over the Reuters dataset without normalized feature vectors

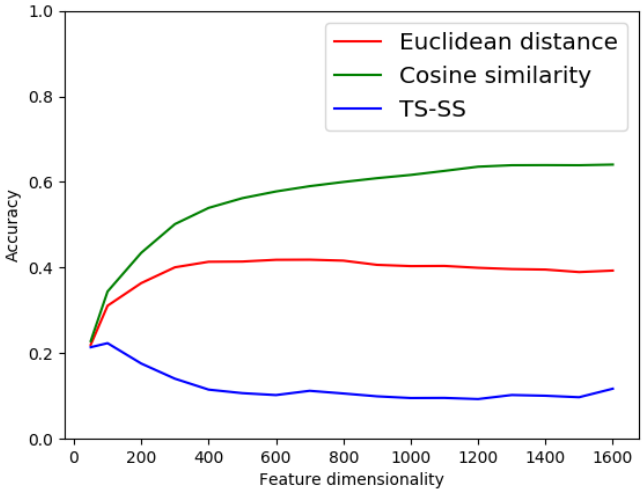

Fig. 3. System's performances over the 20 NewsGroups dataset without normalized feature vectors

EVALUATION METHODS

The performances are evaluated on the test dataset. For each retrieved document the answer is recorded and compared with the solution, in the end, the probability of correct retrieval is calculated — accuracy. New documents are not added to the pool of training documents because then it would be inconvenient to evaluate predictive performances for different parameters.

IV. Results

The performances of the implemented system deeply depend on the data preprocessing and feature extraction procedure. Those procedures require tuning several parameters that are summarized in Table I.

Three similarity metrics - ED, CS and TS-SS similarity – are used for finding the most similar document in training dataset for a given textual query (document), then accuracy is calculated to evaluate performances of our system for different similarity metrics. The whole process of extracting features and searching feature space for the most similar ones is repeated with different tuning parameters and over two datasets, Reuters and 20 NewsGroups.

In the first experiment, the Reuters dataset is used. Fig. 2 shows the accuracy achieved for different dimensionality of the feature vectors.

From Fig. 2 it can be clearly seen that the cosine similarity performed the best, whereas the similarity based on the product of triangle-sector areas was the worse. The method based on the Euclidean distance for the small feature dimensionality follow the same predictive pattern as cosine similarity, but it levels for the feature length bigger than 300, while the accuracy of cosine similarity constantly increases. In contrast, accuracy of TS-SS similarity fluctuates until the feature dimensionality of 200, after which it constantly decreases.

For the second experiment, the 20 NewsGroups dataset is used. This dataset has more documents, and so richer vocabulary, which means that feature vectors if are not limited to a certain length are of bigger dimensionality. Therefore, the performances of our classifiers showed a different pattern. Fig. 3 depicts probability that our system will recognize a document's category successfully for different feature length.

In this scenario, the gap between performances of these three metrics is wider. Although the metric based on the cosine angle is still superior to the other ones, it follows logarithmic incline in performances. However, the TS-SS metric, after the slight increase in performances until the feature length of 100, gets worse dramatically until the feature dimensionality of 400, after which it levels out with small oscillations. Euclidean distance's accuracy, again, shows the similar growth pattern as cosine similarity, in the beginning it follows cosine similarity until the feature length of 100, after which it levels out, but now the accuracy gap between ED's and CS's accuracy is bigger.

The cosine similarity shows a constant trend of increase in performances with the incline of feature dimensionality. In other words, the more information a feature vector contains, the better the performances will be. On the other side, the two other methods lack this ability to exploit highly dimensional feature vectors.

After these two experiments, we further examine the performances of our system when the *tf-idf* feature vectors are normalized. To normalize the data vectors, the *l2* normalization is employed, each data point x_i is subjected to (7) constraint.

$$\sum_{k=1}^{M} x_i = 1 \tag{7}$$

where M is the feature length, and parameter i indexes

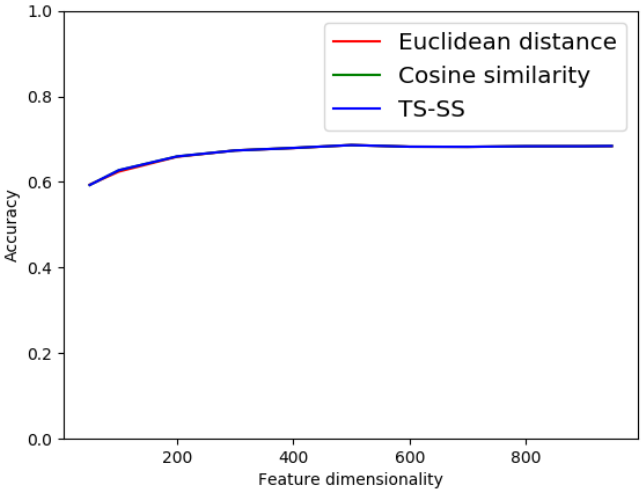

Fig. 4. System's performances over the Reuters dataset with normalized feature vectors

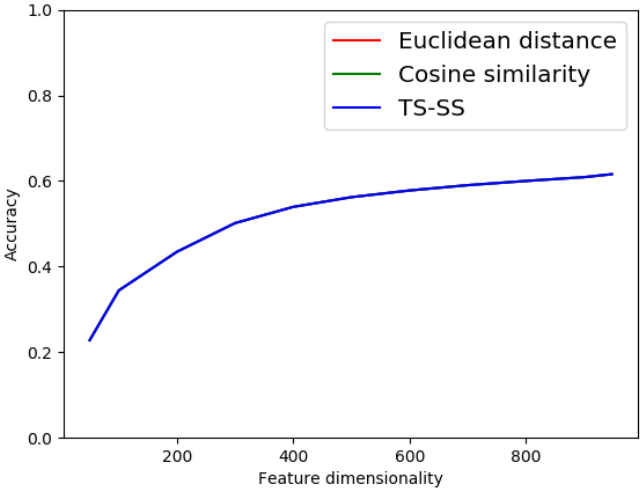

Fig. 5. System's performances over the 20 NewsGroups dataset with normalized feature vectors

documents, $i \in [1, N]$, N is the total number of documents in the training dataset.

Subjecting feature vectors to the constraint (7), two first two experiments are repeated. Now, the results achieved for these two experiments differ significantly. All the three similarity metrics have shown a similar accuracy growth pattern.

For the Reuters dataset, the performances of cosine and TS-SS metrics are almost the same, whereas Euclidean distance is slightly worse for the feature vector length between 100 and 200. From 50 to 250 they show a rapid increase in performances, while after dimensionality of 250 with small oscillations it levels out. For the 20 NewsGroups dataset, the performances of these similarity measures follow the same logarithmic increase in performances. However, the normalization constraint (7) did not impair predictive performances of our system.

The reason for this phenomenon of showing almost the same growth accuracy pattern for each similarity measure is that constraining data points to (7) we get the similar mathematical equations. Now, the similarity metrics are described by mathematical equations (1) for ED, (8) for CS, and (9) for TS-SS.

$$CS(x_i, x_j) = \sum_{k=1}^{M} x_{ik} x_{jk} \tag{8}$$

$$TS\text{-}SS(x_i, x_j) = \theta' \sin \theta' ED(x_i, x_j) \tag{9}$$

Note, that fixed scaling parameters that are same for each data point are excluded from the equations because static scaling does not contribute to different document ranking.

The authors who proposed the TS-SS similarity claim that the features should not be normalized because a normalization constraint would diminish diversity among vectors. However, this characteristic may be beneficial for clustering problems, but for the classification task it is clearly undesirable.

V. Discussion

From the results, we can conclude that the metrics based on the Euclidean distance between data points and Triangular similarity suffer from the problem known as *the curse of dimensionality*, which was first introduced by Bellman [13].

To exemplify why these measures perform worse as dimensionality increases, consider N data points uniformly distributed in an M-dimensional unit ball centered at the origin. Then, our system will classify a given query by finding the closes data point, and assigning it the same class. The median distance from the origin to the closest data point is given by the expression (10).

$$distance(m, N) = (1 - \frac{1}{2}^{\frac{1}{N}})^{\frac{1}{M}} \tag{10}$$

From this equation we can derive a formula (11) to calculate the number of data points required for the given feature length M and expected distance d to the closes data point.

$$N = \log_{1-d^M} \frac{1}{2} \tag{11}$$

Let us fix the expected distance d to the closes data point to the value 0.21, $d = 0.21$, then for $M = 1$, we need 3 data points, for $M = 3$, $N = 74.5$, and for $M = 10$, $N = 4155587$. This means that if we want to achieve the same expected accuracy as with the low dimensional feature vectors, we need the size of our training set to grow exponentially. That is why the Euclidean and TS-SS measures show worse performances for high-dimensional feature vectors.

The authors who proposed TS-SS algorithm, evaluated its performances by four different methods; Uniqueness, Number of Booleans, Minimum gap score, and Purity. However, these

methods are more relevant for evaluating document clustering than document classification.

VI. Conclusion and Future Work

In this work, we performed simple document recommendation based on CBR system in order to evaluate different similarity measures; Euclidean distance, Cosine similarity, and TS-SS similarity.

We implemented a system that is capable of finding the most similar document in two datasets for a given query (unseen document). Both, the training dataset and queries, are subjected to the same procedures of data preprocessing and feature extraction. Then, three different similarity metrics are employed to retrieve the most similar document, each prediction is recorded so that their performances can be evaluated. After exhaustive testing, we concluded that similarity metric based on cosine outperformed ED and TS-SS in high dimensional space, which leads us to conclusion that ED and TS-SS suffer from a problem known as *the curse of dimensionality*. However, when feature vectors are subjected to constraint (3), then all three similarity metrics show the similar predictive pattern.

Recognizing the most similar documents is an active research area in recent years due to increasing use of the Internet. The content on the Internet is mostly unstructured, and for each searched textual document, the similar ones should be retrieved in order to provide the best recommendation to users. Even though a lot of research has been done to improve each step of such a predictive system, there is still no the best solution to deal with NLP problems.

During our research, we stumbled upon a few interesting research question that we consider worthy of further investigation:

- Investigate the performances of similarity metrics for document clustering tasks.
- How the system would perform if feature vectors are constrained to *l1* norm.
- Whether the similarity metrics will show different predictive pattern for feature extraction methods such as Glove or word2vec.
- Investigate performances for datasets that have longer textual documents.

References

[1] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Analyze the future*, vol. 2007, no. 2012, pp. 1–16, 2012.

[2] V. V. Raghavan and S. M. Wong, "A critical analysis of vector space model for information retrieval," *Journal of the American Society for information Science*, vol. 37, no. 5, p. 279, 1986.

[3] M. U. Ahmend, S. Begum, P. Funk, N. Xiong, and B. Von Schéele, "Case-based reasoning for diagnosis of stress using enhanced cosine and fuzzy similarity," *Transactions on Case-Based Reasoning for Multimedia Data*, vol. 1, no. 1, pp. 3–19, 2008.

[4] A. Heidarian and M. J. Dinneen, "A hybrid geometric approach for measuring similarity level among documents and document clustering," in *Big Data Computing Service and Applications (BigDataService), 2016 IEEE Second International Conference on*. IEEE, 2016, pp. 142–151.

[5] R. R. Larson, "Introduction to information retrieval," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 4, pp. 852–853, 2010.

[6] L. Yujian and L. Bo, "A normalized levenshtein distance metric," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1091–1095, 2007.

[7] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[8] P. Willett, "The porter stemming algorithm: then and now," *Program*, vol. 40, no. 3, pp. 219–223, 2006.

[9] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[10] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[11] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[13] R. Bellman, "Curse of dimensionality," *Adaptive control processes: a guided tour. Princeton, NJ*, 1961.