A comparative analysis of Temporal Long Text Similarity: Application to Financial Documents

Vipula Rawte[1][0000−0002−4355−1393], Aparna Gupta[2][0000−0002−5275−7756], and Mohammed J. Zaki[1][0000−0003−4711−0234]

[1] Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY, USA
`rawtev@rpi.edu, zaki@cs.rpi.edu`
[2] Lally School of Management, Rensselaer Polytechnic Institute, Troy, NY, USA
`guptaa@rpi.edu`

Abstract. Temporal text documents exist in many real-world domains. These may span over long periods of time during which there tend to be many variations in the text. In particular, variations or the similarities in a pair of documents over two consecutive years could be meaningful. Most of the textual analysis work like text classification focuses on the entire text snippet as a data instance. It is therefore important to study such similarities besides the entire text document. In Natural Language Processing (NLP), the task of textual similarity is important for search and query retrieval. This task is also better known as Semantic Textual Similarity (STS) that aims to capture the semantics of two texts while comparing them. Also, state-of-the-art methods predominantly target short texts. Thus, measuring the semantic similarity between a pair of long texts is still a challenge. In this paper, we compare different text matching methods for the documents over two consecutive years. We focus on their similarities for our comparative analysis and evaluation of financial documents, namely public 10-K filings to the SEC (Securities and Exchange Commission). We further perform textual regression analysis on six quantitative bank variables including *Return on Assets (ROA)*, *Earnings per Share (EPS)*, *Tobin's Q Ratio*, *Tier 1 Capital Ratio*, *Leverage Ratio*, and *Z-score*, and show that textual features can be effective in predicting these variables.

Keywords: temporal changes · text similarity · long text regression · SEC 10-K reports

1 Introduction

In general, over long periods of time, the language tends to evolve [8]. Specifically, the written language present in the form of unstructured text documents changes a lot given the amount of text data generated daily. Studying how the texts change over time is an interesting area. A lot of text mining research is being done on the entire text documents or short texts, such as tweets, messages, reviews, and so on. Some works also target temporal text mining [13] and temporal networks [35]. Temporal text documents exist in real-world domains like clinical texts, news articles, financial statements, and restaurant and product reviews. For this work, we focus on the financial disclosure statements, i.e., 10-K documents from the Securities and Exchange Commission (SEC) in the finance domain. In particular, we look at the Sections 7 and 7A: "Management's Discussion and Analysis of financial conditions and results of operations" (MD&A). The reason to choose the finance domain and Section 7/7A in 10-K documents is threefold:

1. Section 7A contains forward-looking statements about the company, which report the company's operations and financial results. It may also talk about the potential risks.
2. The length of Section 7 is longer than an average text document, the approximate longest section being an average of ∼40000 words as reported in Table 2.
3. The nature of the MD&A section makes it suitable to find if any relationship exists between the textual similarities and bank financial performance variables [6, 19].

Measuring the similarity between texts is a challenging task in NLP because of linguistic, semantic and knowledge-based factors. There exist a number of methods ranging from traditional count based methods to neural networks and knowledge based methods for measuring textual similarity. The basic idea is to map the texts onto a vector space model (VSM) such that there are two vectors for a pair of documents. A cosine similarity metric is used to measure the similarity between these two vectors by computing the dot product between two such normalized vectors. Textual similarity has a wide range of applications in real-world domains such as legal [2,22,31,33], academic [20,21], finance [6,30], and medicine [36,38]. It can also be useful for several NLP tasks such as document classification, clustering, and retrieval.

Contributions: In this study, we compare different document representation techniques and similarities between two temporal text documents over two consecutive years and apply them to the section 7/7A of public 10-K filings. Given its significance and applications in NLP, we choose several state-of-the-art methods in our comparative study. We further use these similarities to predict six bank variables using linear regression. We also study some interesting patterns between the textual similarities and the numeric values.

2 Related Work

Document dating [12, 34], Neural Networks (NN) based diachronic framework for temporal text classification [11], and dynamic topic modeling [4, 24, 37] are a few of the recent works in the area of temporal text document analysis.

In [6], the authors show how the modification score varies with the length of the document over years. This score is computed by using the cosine similarity between the document vectors constructed using the term frequency-inverse document frequency (tf-idf) scores. In more recent work [5], the authors create a multi-dimensional measure of financial statement peer-to-peer similarity which is purely quantitative.

In the field of Information Retrieval (IR), document similarity is widely applicable for tasks for as document clustering, document retrieval, query search, document deduplication, question answering, and so on. In [32], a comparative study of various textual similarity methods such as tf-idf (and other related extensions), topic models (Latent Semantic Indexing (LSI)) [7], and neural models (paragraph vector, doc2vec [18]) is presented. It highlights that tf-idf still remains a good option when it comes to long text documents as compared to the other complex methods.

A lot of work on short text similarity exists with some standard evaluation benchmark datasets such as SemEval STS [1]. [15] shows how word embeddings (vector representations of words) constructed from unlabelled data can be used to compute the semantic similarity by finding the cosine similarity between two vectors. This includes the semantics instead of just lexical or syntactic information. In [29], a NN-based technique is described to measure the textual similarity. The authors used a Siamese Convolutional Neural Network (CNN) to capture the relevance of a word in a sentence in order to create a word representation. Then they used a Siamese LSTM (Long-Short Term Memory) to analyze a pair of sentences, and their similarity is computed using the Manhattan distance. [27] proposed a novel way to compute similarity based on the present term set in order to tackle the issue where several documents have an identical degree of similarity to a specific document. This measure is based on the term weights and the number of terms that exist in at least one of the two documents. External knowledge and word embeddings based method to measure semantic similarity is proposed in [26].

For longer documents, [21] proposed a novel joint word-embedding model based semantic matching of long documents by incorporating domain-specific semantics information into the basic context of the words. This model is then applied to academic documents by including semantic profiles for research purpose, methodology, and domain to create the embedding. Since the lengths of two documents can vary, representing long documents as vectors is not always helpful. Thus, it is important to address the length difference between two documents when computing their similarity. The method in [9] shows how to represent a longer document as a representation of the latent topics, and the shorter document as just an average of the word embedding it is composed of. Finally, they used cosine similarity to measure the document similarity. They also showed the ineffectiveness of doc2vec and Word Mover's Distance [17] in their document similarity task. In [25], the authors present a simple, unsupervised method for pairwise document matching. They try to improve [9] by first averaging over the word embeddings, and then compute the cosine similarity between these two averages. In [3], the authors proposed a knowledge-based semantic textual similarity technique called Context Semantic Analysis (CSA) which relies on a RDF knowledge base such as DBpedia and Wikidata to extract a Semantic Context Vector to represent a document.

3 Methodology

We try to closely follow the work in [6] where the authors define a *rawscore* to measure the changes between two documents. In this work, we instead look at the similarities between two documents. For this purpose, we examine some text-based methods to measure similarity between two financial documents. We further categorize these methods into the following three groups.

3.1 Bag-of-Words (BOW)

It represents text in a vector form as the occurrence of words in a given document.

1. **Count Vectorizer:** This is a classical method to create a document vector, which is solely based on the word counts of the words present in the corpus vocabulary. Such a document vector gives more importance to the most frequent words even if they are not relevant.
2. **tf-idf:** The above issue is overcome by weighting the word scores using a formula given in Eq. 1 to construct a document vector.

$$tf\text{-}idf_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right) \tag{1}$$

   where, $tf_{i,j}$ is the number of occurrences of word i in document j, df_i is the number of documents containing word i, and N is the total number of documents.

3.2 Word/Document embedding

The BOW methods do not capture any semantics of the text and so we study some additional methods below.

1. **tf-idf + Latent Semantic Analysis (LSA):** Unlike bag-of-words or neural network models, LSA works on the core idea of document-term matrix and Singular Value Decomposition (SVD) as a dimensionality reduction technique for the sparse document-term matrix. It decomposes the entire matrix into a separate document-topic matrix and a topic-term matrix. Thus, it is able to capture the semantics from the textual documents. It uses the basic idea of SVD on the document-term matrix by diving it into three matrices U, Σ and V so that: $X = U\Sigma V^t$ where U is the terms-topics matrix, Σ is the topics importance matrix and V^t is the topics-documents matrix.

2. **doc2vec:** This is an unsupervised document embedding technique to create document vectors [18]. It is analogous to the word2vec [23] technique to create word vectors. Similar to word2vec, doc2vec has two variants, distributed memory (DM) and distributed bag of words (DBOW). We use the *gensim* library[1] to create the doc2vec representations. For training, we set the parameters as *epochs=100 and window_size=15*. Next, we use the following three different word embedding techniques in the doc2vec model.

   word2vec: It captures the local context of a given word within a window size to create word vectors [23].

   GloVe: It captures both local and global context of a given word by constructing the co-occurrence matrix of the words in a document [28].

   fasttext: This model breaks down an out-of-vocabulary word into sub-words and has improvements over word2vec and GloVe in some NLP tasks [14]. For doc2vec, we use both the variants, DM and DBOW. These methods have shown improvements in results on tasks such semantic textual similarity, but may perform worse when there are misspellings of the same word. This issue of misspelling is handled well in string matching techniques, discussed below, where a word is reconstructed from the original word using the minimum edit distance.

3.3 String matching

The simplest way to compare two strings is with a measurement of edit distance required to reconstruct a string from the original string.

1. **Fuzzywuzzy**: Fuzzy string matching technique compares two strings to find matches where there are misspellings or just partial words. It is called *fuzzy* because it uses an 'approximate' string matching technique based on Levenshtein Distance to calculate the edit distance, using the formula given in Eq. 2. We use a fast, optimized Python library Fuzzywuzzy[2] for the task of string matching.

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j)=0 \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j)+1 \\ \operatorname{lev}_{a,b}(i,j-1)+1 \\ \operatorname{lev}_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise} \end{cases} \tag{2}$$

   where $1_{(a_i \neq b_j)}$ denotes 0 when $a = b$ and 1 otherwise. Finally, the *Levenshtein similarity ratio* is computed based on the Levenshtein distance, and is calculated using the formula in Eq. 3.

[1] https://radimrehurek.com/gensim/models/doc2vec.html
[2] https://github.com/seatgeek/fuzzywuzzy

$$\frac{(|a|+|b|)-\text{lev}_{a,b}(i,j)}{|a|+|b|} \tag{3}$$

where $|a|$ and $|b|$ are the lengths of sequence a and sequence b, respectively.

3.4 Similarity metrics

For two given documents, we compare them to measure the similarity between them. A similarity metric is thus used to quantify such similarity between two texts. Having looked at the three groups of approaches for representing documents, we next consider some common similarity metrics based on different input representations.

1. **Jaccard Similarity:** It is used to compute similarity between two strings which are represented as sets. The formula to calculate Jaccard similarity is given in Eq. 4.

$$J(A,B)=\frac{|A\cap B|}{|A\cup B|}=\frac{|A\cap B|}{|A|+|B|-|A\cap B|} \tag{4}$$

   where A and B are two sets containing tokens.

2. **Cosine similarity:** It calculates similarity between two vectors by measuring the cosine of the angle between them and it is given by formula in Eq. 5. In our case, we use techniques like tf-idf, tf-idf+LSA, word2vec and doc2vec to represent each document as a vector. We then use cosine similarity to measure the similarity between these two vectors.

$$\text{cosine\_sim}(A,B)=cos(\theta)=\frac{A\cdot B}{\|A\|\times\|B\|}=\frac{\sum_{i=1}^{n}A_i\times B_i}{\sqrt{\sum_{i=1}^{n}A_i^2}\times\sqrt{\sum_{i=1}^{n}B_i^2}} \tag{5}$$

   where, A_i and B_i are the i^{th} components of the vectors A and B, respectively, and θ is the angle between A and B in the n-dimensional vector space.

3. **Pearson Correlation Coefficient:** It is a statistical measure that finds the linear correlation between two vectors A and B. It ranges between -1 and 1, where -1 is total negative linear correlation, 0 is no linear correlation, and 1 is total positive linear correlation. The formula to calculate the correlation is given in Eq. 6.

$$\text{PCC}(A,B)=\frac{\text{cov}(A,B)}{\sigma_A\sigma_B}=\frac{\sum_{i=1}^{n}\left(A_i-\bar{A}\right)\left(B_i-\bar{B}\right)}{\sqrt{\sum_{i=1}^{n}\left(A_i-\bar{A}\right)^2\left(B_i-\bar{B}\right)^2}} \tag{6}$$

   where cov(A, B) is the covariance, σ_A is the standard deviation of A and σ_B is the standard deviation of B.

4 Experiments and Results

4.1 Dataset and preprocessing

For our experiments, we use the dataset described in [30], which contains the 10-K filings for US banks for the period between 2006 and 2016. We only use sections 7 and 7A in our experiments. We report all the dataset statistics in Table 1. For preprocessing, we use *nltk* word tokenizer [3] to tokenize the documents. We then use *beautifulsoup4* [4] to remove all HTML tags. We also remove all the stopwords and punctuation (except ., %, and $), and convert the numerals and email address into #. Finally, we use *nltk PorterStemmer* to stem the tokens. We use *scikit-learn*[5] to compute tf-idf scores. We obtain the bank variable values from the dataset used in [10]. Also, we use *sklearn MinMaxScaler* to scale all the values for our evaluation. We show the distribution of all six bank variables in Fig. 1 and their statistics in Table 2. Code and data of our approach is available at `https://github.com/vr25/temp_change_text_reg`.

Table 1. Dataset Size

	# documents
total	5321
after extracting items 7, 7A	3396
data pairs for two consecutive years	2337

Table 2. Data Statistics: The min is 0 and max is 1 after scaling.

	ROA	**EPS**	**Tobin's Q Ratio**	**Tier 1 Cap Ratio**	**Leverage Ratio**	**Z-score**
mean	7.84E-01	7.50E-01	4.50E-01	2.44E-01	4.40E-01	9.46E-02
std. dev.	5.27E-02	5.92E-02	7.42E-02	5.14E-02	3.23E-02	1.07E-01

It is interesting to note that we also observe that the average length of section 7, 7A increases over time as shown in Fig. 2. Finally, we study the effect of document length on textual similarity in Fig. 3. We also plot the difference in document lengths (denoted delta), i.e., $year2$ - $year1$ lengths, and thus we can get negative values when $year1$ document is longer than $year2$ document. We also find that the similarity increases with the increase in document length beyond 40000 words.

[3] https://www.nltk.org/

[4] https://pypi.org/project/beautifulsoup4/

[5] https://scikit-learn.org/

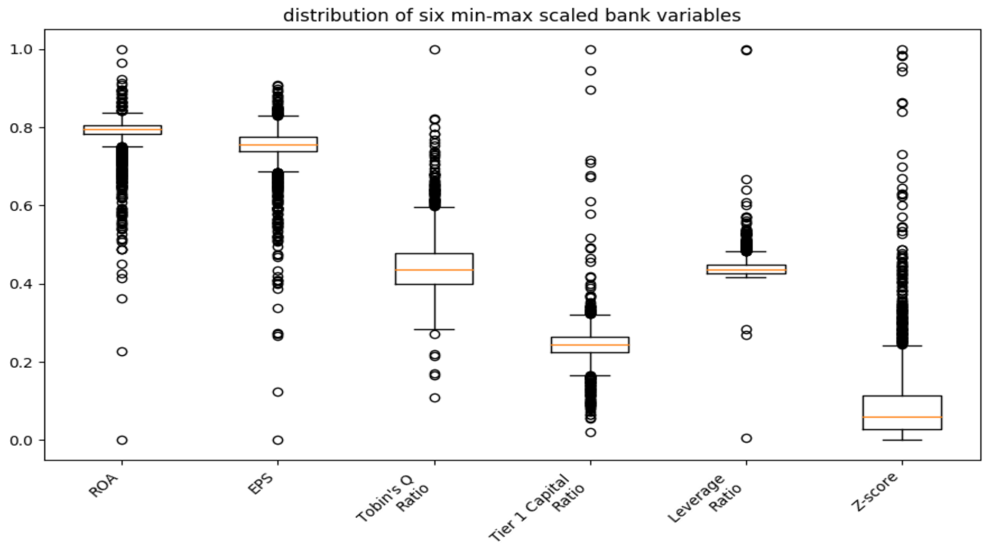

Fig. 1. This box-whisker plot shows the distribution of all six bank variables.

4.2 Evaluation metric

In order to evaluate our experiments on text regression task, we use mean squared error (mse) (also known as standard deviation of residuals) metric given in the following Eq. 7:

$$\text{mean squared error (mse)} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{7}$$

4.3 Results and discussion

All the experiments were performed on a machine with a 2.3GHz Intel Xeon Processor E5-2670 v3 Processor, with 251G RAM and 80 CPU cores and Python 3.6 environment.

We study and compare some standard document embedding techniques and document similarity measures. We use these methods on textual data available from a dataset of sections 7 and 7A from 10-K filings. Our main goal here is to examine how the textual similarities from two consecutive years perform when compared to the entire document. We also compare the textual features with the numeric scores.

Regression: For evaluation, we perform linear regression and ridge regression. We use linear regression where the independent variable (numeric score) is the similarity measure and the dependent variable (numeric score) is one of the

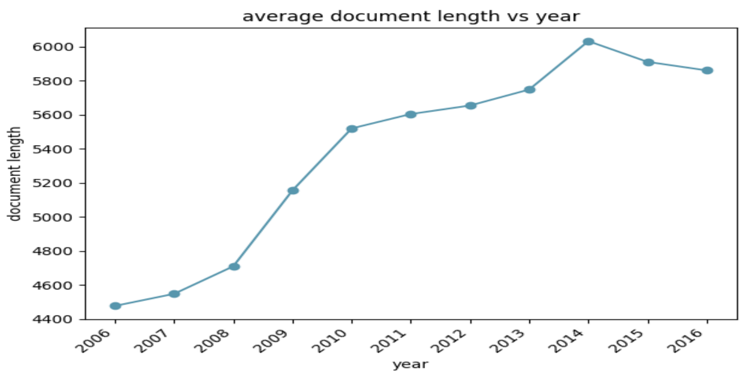

Fig. 2. document length vs. year pairs

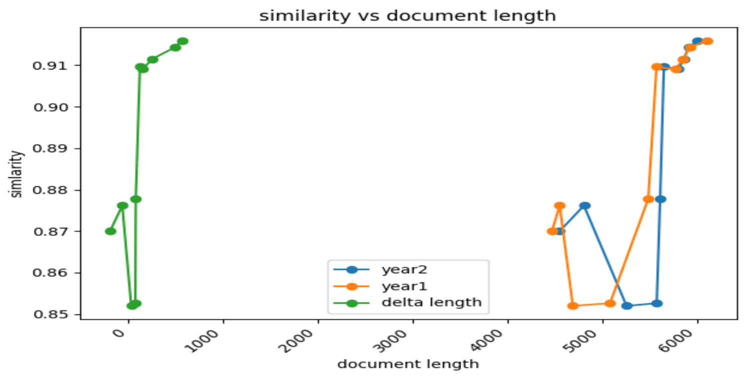

Fig. 3. Textual similarity (cosine) [doc2vec (word2vec)] vs. document length

six bank variables (done independently). Next, we also, compare these textual similarities with the entire textual content. We do this by selecting text from (1) *year1*, (2) *year2*, (3) concatenation of *year1* and *year2* document vectors (using tf-idf method), and (4) combination of *year1* and *year2* documents. In (3), we first compute the tf-idf based vector representation of the text for two consecutive years *year1* and *year2* and then concatenate these vectors, whereas in (4) we first merge the textual strings from *year1* and *year2* and then create a tf-idf based vector for the combined text. We now perform ridge regression on these textual features. For both, linear and ridge regression, we then perform a 10-fold cross validation, and compute the mean squared error.

Baseline: We use *year1* scores to predict *year2* scores for all six bank variables. Thus, we are using historical values to predict the future values as a numeric baseline which is similar to the baseline proposed in [16] for textual regression to predict future stock volatility. Similarly, we also use the delta (*year2* - *year1*) values of these scores to predict the *year2* values.

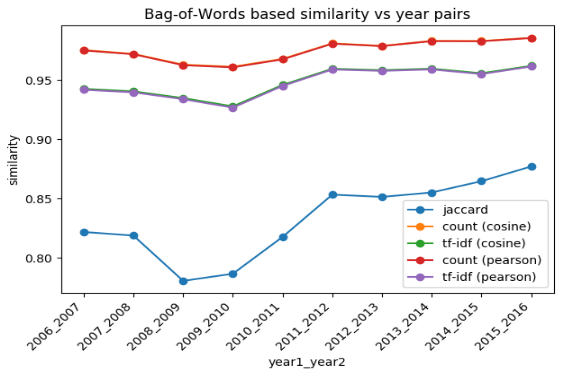

Fig. 4. This plot shows Bag-of-Words similarity across year pairs.

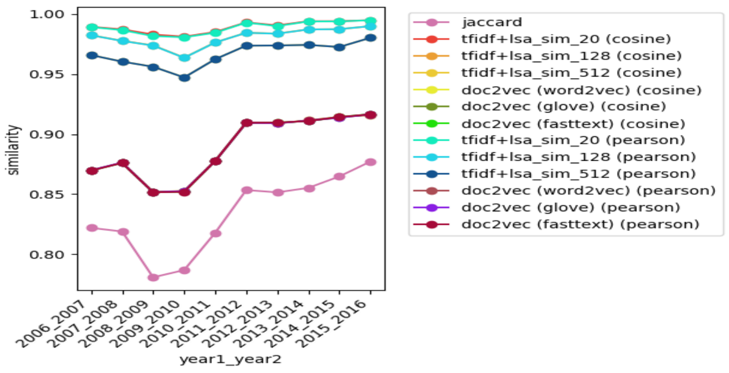

Fig. 5. This plot shows document vector based similarity across year pairs.

Comparative Analysis: We categorize the jaccard similarity metric into the Bag-of-Words type since this similarity measure uses sets of words which have no specific word order. We also use count-based and tf-idf based methods to construct document vectors. We study the relation between these methods across pairs of years between 2007 and 2016 in Fig. 4. We observe a slight drop in similarity between years 2008 and 2010, during the previous financial crisis. In Fig 4 and Fig. 5, since we get identical results for the cosine similarity and PCC these plots overlap each other.

For embedding based methods, we choose tf-idf + LSA (with different number of topics) and doc2vec (with different word embedding techniques). For two given vectors, we compute (1) cosine similarity and (2) PCC (Pearson Correlation Coefficient), as shown in Fig. 5. Interestingly, we find that jaccard smilarity (bag-of-words) and doc2vec (document embedding) show similar trend across year pairs and this pattern is quite similar to tf-idf+lsa methods too.

Bag-of-Words: For BOW methods, we compute similarity between two tokenized texts. For jaccard similarity, we represent each tokenized text as a set of tokens and then find the similarity using the formula in Eq. 4. For count-based approach, we create document vector based on the word counts of the words present in that document. To overcome the drawbacks in count-based approach, we use tf-idf method where documents are represented using the tf-idf scores (Eq. 1). For both these methods, the similarity is calculated using the cosine similarity (Eq. 5) and PCC (Eq. 6).

Word/Document Embedding: In order to capture the semantics, we move on to the embedding based techniques. For tf-idf+LSA methods, we first find the tf-idf scores to filter out unimportant words and then apply LSA to find the important topics where we choose from 20, 128 and 512 number of topics. For doc2vec methods, we choose different word embedding techniques like word2vec, GloVe and fasttext. Again, for all these methods, similarity is computed using cosine similarity and PCC.

String Matching: In this method, we do not tokenize the text unlike the above methods. We directly compare two text strings from *year1* and *year2* because we are trying to find the similarities between two texts as a human would find by matching two strings.

Purely textual features: For text only features, we follow four different methods. (1) *year1*: We consider the tokenized text from *year1* to predict the numeric score for *year2*. (2) *year2*: We follow similar approach as (1) with the only difference being tokenized text from *year2* to predict *year2* numeric scores. For both these methods, we represent the documents using tf-idf scores to create the document vectors. (3) *year1* + *year2* (concat): For each of the two text documents, we first create tf-idf vectors using the same approach as above for (1) and (2). We then concatenate them to create a longer vector. We use this textual representation to predict the *year2* numeric scores. (4) *year1* + *year2* (merge): Unlike (3), for two text documents, we first merge them as a regular text string. Then following the same approach for (1) and (2), we represent this merged text using a tf-idf vector which is then used to predict the *year2* scores.

Regression Results We follow the methods described above to compute the similarities between two texts. The detailed experimental results are shown in Table 3. We do not report the PCC values in this table because we get identical results as cosine similarity in almost all the cases. Hence, we just report the mse for jaccard and cosine similarity values.

We can observe that using the delta values (i.e., *year2*-*year1* numeric values) performs the best overall on all variables except *Tobin's Q Ratio*. Here, we are using the difference in the values from one year to the next to predict next year's (*year2*) value, and so it is not entirely surprising that this is a strong baseline. It is also implicitly capturing information about the dependent variable (*year2* numeric score), and therefore is not entirely fair.

We find that among the textual approaches, the concatenation of *year1* and *year2* text performs the best for *ROA*, *EPS*, *Tobin's Q Ratio* and *Tier 1 Capital Ratio*. For *Tobin's Q Ratio*, we get better mse with purely textual features than the numeric score. For *Leverage Ratio* and *Z-score ratio*, we find that *year1* and *year2* textual features respectively, are the best among the textual approaches. We conclude that using textual features, especially concatenating the text from *year1* and *year2* is the overall best approach to predict the bank performance variables, and results in a better approach than using only the numeric scores.

Table 3. Regression based on financial text

	Bank variables/ Methods	**ROA**	**EPS**	**Tobin's Q Ratio**	**Tier 1 Capital Ratio**	**Leverage Ratio**	**Z-score**
baseline [numeric]	**year1**	2.10E-03	2.89E-03	2.37E-03	**5.32E-04**	8.61E-04	1.03E-02
	delta	**1.89E-03**	**2.64E-03**	4.27E-03	2.45E-03	**7.94E-04**	**7.39E-03**
Bag-of-Words	**jaccard similarity**	2.60E-03	3.71E-03	4.78E-03	2.56E-03	1.11E-03	1.12E-02
	count vectorizer	2.65E-03	3.78E-03	4.86E-03	2.58E-03	1.11E-03	1.14E-02
	tf-idf	2.66E-03	3.79E-03	4.87E-03	2.58E-03	1.11E-03	1.14E-02
word/ document embedding	**tf-idf+ lsa(20)**	2.66E-03	3.79E-03	4.87E-03	2.58E-03	1.11E-03	1.14E-02
	tf-idf+ lsa(128)	2.66E-03	3.79E-03	4.87E-03	2.58E-03	1.11E-03	1.14E-02
	tf-idf+ lsa(512)	2.66E-03	3.79E-03	4.87E-03	2.58E-03	1.11E-03	1.14E-02
	doc2vec (word2vec)	2.62E-03	3.73E-03	4.79E-03	2.57E-03	1.11E-03	1.13E-02
	doc2vec (glove)	2.62E-03	3.73E-03	4.79E-03	2.57E-03	1.11E-03	1.13E-02
	doc2vec (fasttext)	2.62E-03	3.73E-03	4.78E-03	2.57E-03	1.11E-03	1.12E-02
string matching	**fuzzy-wuzzy**	2.65E-03	3.77E-03	4.88E-03	2.58E-03	1.11E-03	1.15E-02
year1	**tf-idf**	2.13E-03	2.91E-03	2.22E-03	8.71E-04	8.79E-04	9.61E-03
year2	**tf-idf**	2.10E-03	2.70E-03	2.22E-03	7.77E-04	9.02E-04	9.38E-03
year1 + year2 (concat)	**tf-idf**	1.93E-03	2.66E-03	**1.92E-03**	6.38E-04	8.91E-04	9.70E-03
year1 + year2 (merge)	**tf-idf**	2.11E-03	2.78E-03	2.22E-03	7.86E-04	9.34E-04	9.48E-03

5 Conclusion and Future Work

We performed a comparative study of different textual similarity techniques to compare long text documents such as sections 7 and 7A of 10-K filings from two consecutive years. We divided these techniques into simple bag-of-words, word/document embeddings and string matching types. Since the bag-of-words model does not capture any semantics of the text, we follow the word and document embedding approach to create document vectors. From our comparative experiments, we observed that doc2vec (using both, cosine similarity and PCC) and jaccard similarity show similar trends. We also saw a slight drop in similarity for years 2008-2010 and we can therefore say that the documents between the years 2008 and 2010 were not as similar when compared to later years. Further, we used these similarity scores to predict the next year's values of different bank variables to see if the textual changes by themselves are predictive. We observe that the combined text from the previous and current years is helpful in predicting several bank variables, namely *ROA*, *EPS*, *Tobin's Q Ratio* and *Z-score*. In future, we plan to use more advanced document embedding methods such as temporal topic models to better understand the similarities qualitatively between the topics present in a pair of documents. It would be also interesting to extend the time period beyond consecutive year pairs to a longer time window. Since the document length increases over the years, using normalized document length can also be used as an additional feature.

6 Acknowledgments

This work was supported in part by NSF Award III-1738895.

References

1. Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A.: Semeval-2012 task 6: A pilot on semantic textual similarity. In: * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012). pp. 385–393 (2012)
2. Alschner, W.: Sense and similarity: Automating legal text comparison. Available at SSRN 3338718 (2019)
3. Benedetti, F., Beneventano, D., Bergamaschi, S., Simonini, G.: Computing inter-document similarity with context semantic analysis. Information Systems **80**, 136–147 (2019)
4. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: Proceedings of the 23rd international conference on Machine learning. pp. 113–120 (2006)
5. Brown, S.V., Ma, G., Tucker, J.W.: A measure of financial statement similarity. Available at SSRN 3384394 (2019)
6. Brown, S.V., Tucker, J.W.: Large-sample evidence on firms' year-over-year md&a modifications. Journal of Accounting Research **49**(2), 309–346 (2011)
7. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American society for information science **41**(6), 391–407 (1990)
8. Dyer, T., Lang, M., Stice-Lawrence, L.: The evolution of 10-k textual disclosure: Evidence from latent dirichlet allocation. Journal of Accounting and Economics **64**(2-3), 221–245 (2017)
9. Gong, H., Sakakini, T., Bhat, S., Xiong, J.: Document similarity for texts of varying lengths via hidden topics. arXiv preprint arXiv:1903.10675 (2019)
10. Gupta, A., Owusu, A.: Identifying the risk culture of banks using machine learning. Available at SSRN 3441861 (2019)
11. He, Y., Li, J., Song, Y., He, M., Peng, H., et al.: Time-evolving text classification with deep neural networks. In: IJCAI. pp. 2241–2247 (2018)

12. Huang, X., Paul, M.: Examining temporality in document classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 694–699 (2018)
13. Hurst, M.F., et al.: Temporal text mining. In: AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs. pp. 73–77 (2006)
14. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
15. Kenter, T., De Rijke, M.: Short text similarity with word embeddings. In: Proceedings of the 24th ACM international on conference on information and knowledge management. pp. 1411–1420 (2015)
16. Kogan, S., Levin, D., Routledge, B.R., Sagi, J.S., Smith, N.A.: Predicting risk from financial reports with regression. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. pp. 272–280 (2009)
17. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: International conference on machine learning. pp. 957–966 (2015)
18. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196 (2014)
19. Li, H.: Repetitive disclosures in the md&a. Journal of Business Finance & Accounting **46**(9-10), 1063–1096 (2019)
20. Liu, M., Lang, B., Gu, Z.: Calculating semantic similarity between academic articles using topic event and ontology. arXiv preprint arXiv:1711.11508 (2017)
21. Liu, M., Lang, B., Gu, Z., Zeeshan, A.: Measuring similarity of academic articles with semantic profile and joint word embedding. Tsinghua Science and Technology **22**(6), 619–632 (2017)
22. Mandal, A., Chaki, R., Saha, S., Ghosh, K., Pal, A., Ghosh, S.: Measuring similarity among legal court case documents. In: Proceedings of the 10th Annual ACM India Compute Conference. pp. 1–9 (2017)
23. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
24. Momeni, E., Karunasekera, S., Goyal, P., Lerman, K.: Modeling evolution of topics in large-scale temporal text corpora. In: Twelfth International AAAI Conference on Web and Social Media (2018)
25. Müller, M.C.: Semantic matching of documents from heterogeneous collections: a simple and transparent method for practical applications. arXiv preprint arXiv:1904.12550 (2019)
26. Nguyen, H.T., Duong, P.H., Cambria, E.: Learning short-text semantic similarity with word embeddings and external knowledge sources. Knowledge-Based Systems **182**, 104842 (2019)
27. Oghbaie, M., Zanjireh, M.M.: Pairwise document similarity measure based on present term set. Journal of Big Data **5**(1), 52 (2018)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
29. Pontes, E.L., Huet, S., Linhares, A.C., Torres-Moreno, J.M.: Predicting the semantic textual similarity with siamese cnn and lstm. arXiv preprint arXiv:1810.10641 (2018)
30. Rawte, V., Gupta, A., Zaki, M.J.: Analysis of year-over-year changes in risk factors disclosure in 10-k filings. In: Proceedings of the Fourth International Workshop on

Data Science for Macro-Modeling with Financial and Economic Datasets. pp. 1–4 (2018)
31. Renjit, S., Idicula, S.M.: Cusat nlp@ aila-fire2019: Similarity in legal texts using document level embeddings. Proceedings of FIRE (2019)
32. Shahmirzadi, O., Lugowski, A., Younge, K.: Text similarity in vector space models: a comparative study. In: 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). pp. 659–666. IEEE (2019)
33. Sugathadasa, K., Ayesha, B., de Silva, N., Perera, A.S., Jayawardana, V., Lakmal, D., Perera, M.: Legal document retrieval using document vector embeddings and deep learning. In: Science and Information Conference. pp. 160–175. Springer (2018)
34. Vashishth, S., Dasgupta, S.S., Ray, S.N., Talukdar, P.: Dating documents using graph convolution networks. arXiv preprint arXiv:1902.00175 (2019)
35. Vega, D., Magnani, M.: Foundations of temporal text networks. Applied network science **3**(1), 25 (2018)
36. Wang, Y., Afzal, N., Fu, S., Wang, L., Shen, F., Rastegar-Mojarad, M., Liu, H.: Medsts: a resource for clinical semantic textual similarity. Language Resources and Evaluation pp. 1–16 (2018)
37. Wright, R.: Temporal text mining: A thematic exploration of don quixote
38. Zheng, T., Gao, Y., Wang, F., Fan, C., Fu, X., Li, M., Zhang, Y., Zhang, S., Ma, H.: Detection of medical text semantic similarity based on convolutional neural network. BMC medical informatics and decision making **19**(1), 156 (2019)