An empirical comparison of distance/similarity measures for Natural Language Processing using Text Graph Convolutional Networks

Dimmy K. S. Magalhães[1], **Aurora T. R. Pozo**[1], **Roberto Santana**[2]

[1]Departamento de Ciência da Computação, Universidade Federal do Paraná
CEP: 81.531-980 – Curitiba – PR – Brasil

[2]Departament of Computer Science and Artificial Intelligence,
University of the Basque Country San Sebastián - Donostia, Spain

{dksmagalhaes, aurora}@inf.ufpr.br, roberto.santana@ehu.es

Abstract. *Text Classification is one of the tasks of Natural Language Processing (NLP). In this area, Graph Convolutional Networks (GCN) has achieved values higher than CNN's and other related models. For GCN, the term frequency–inverse document frequency (TF-IDF) defines the correlation between words in a vector space plays, it determines the weight of the edges between two words (represented by nodes in the graph). In this study, we empirically investigated the impact of thirteen other measures of distance/similarity in GCN. A representation was built for each document using word embedding from word2vec model. Also, a graph-based representation of five dataset was created for each measure analyzed, where each word is a node in the graph, and each edge is weighted by distance/similarity between words. Finally, each model was run in a simple graph neural network. The results show that, concerning text classification, there is no statistical difference between the analyzed metrics and the Graph Convolution Network. Even with the incorporation of external words or external knowledge, the results were similar to the methods without the incorporation of words. However, the results indicate that some distance metrics behave better than others in relation to context capture, with Euclidean distance reaching the best values or having statistical similarity with the best.*

1. Introduction

Text classification is the technique of organizing information by assigning documents to a set of semantic categories. In the context of natural language processing (NLP), the main point is to train a classifier to learn from examples and automatically group them into categories. The initial step for text classification is to create a representation for the text. Some approaches use bag-of-words or n-grams for this task [Yao et al. 2018].

This study conducted an empirical comparison of different measures of distance and similarity between words to build a graph-based representation applied in a Text Graph Convolutional Network (Text GCN) [Yao et al. 2018].

The main objective of this work is to analyze different distance measurements in order to verify their behavior through the new convolution techniques in graphs presented in Graph Neural Networks exploring possibilities of relationship and contextualization

between words, besides TF-IDF as in [Yao et al. 2018]. The study hypothesis is that choosing an appropriate metric has a relevant factor in creating a refined text representation for convolutions in graphs.

In the experiments, we used five datasets as well as benchmark corpora. First, for each dataset, a preprocess was made: tokenized all text from the corpus; removed useless words, such as conjunctions, disjunctions, articles and others non-significant symbols, for a better context analysis; moreover we used stemming techniques to remove morphological affixes from words. Second, with the remained words, a word embedding model was trained. For each created model, a second model was generated by a method that refines vector space representations through relational information from semantic lexicons. Thus, for each parameter set, two word embedding were built, one with retrofitting and another without it. Third, a graph was built using pre-trained word embeddings, where each node i is a vector representation of a word i and each edge (i,j) is the measure of distance or similarity between words i and j. Finally, we run Text GCN to create the computational model. Figure 1 shows the default workflow of our approach.

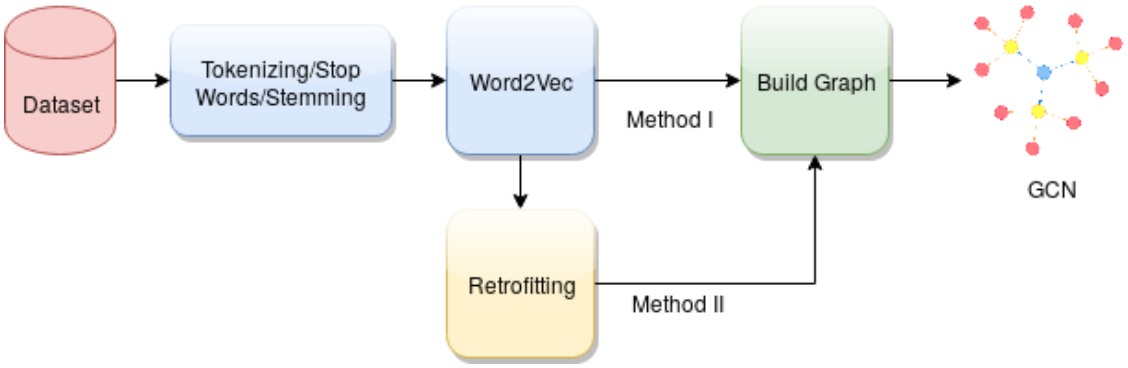

Figure 1. Workflow of methods. Method I uses original word embedding from word2vec. In Method II we build retrofitting in vector representation space

The remainder of the paper is organized as follows. Related work is discussed in Section 2. Section 3 presents an overview about NLP techniques. In Section 4, the methodology of the experiments is presented. Section 5 describes the results and discussions. Finally, advantages, limitations, and future research directions are discussed in Section 6.

2. Related Work

In the last few years, neural networks based on dense vector representations have produced outstanding results on various NLP tasks [Young et al. 2018]. Deep learning enables multi-level automatic feature representation learning. [Goldberg 2016] describes the basic principles to apply neural networks for NLP in a tutorial manner. Many studies conducted in deep neural networks have shown good improvement in NLP tasks, such as Convolutional Neural Networks (CNN).

[Collobert et al. 2011] propose a general CNN-based framework to solve a plethora of NLP tasks, they use a window approach to create a dependency between neighboring words. In [Kalchbrenner et al. 2014], a Dynamic Convolutional Neural Network (DCNN) is described, the network uses Dynamic k-Max pooling, a global pooling operation over linear sequences and use convolution between near words. Recurrent Neu-

ral Networks use the idea of processing sequential information. [Liu et al. 2016] and [Luo 2017] use a recurrent neural networks to build text representation. Recursive neural networks represent a natural way to model sequences [Young et al. 2018] and have been used for a better use of a syntactic interpretations of sentence structure such as in [Socher et al. 2013].

Moreover, [Mikolov et al. 2013, Le and Mikolov 2014] propose two models architectures for computing continuous vector representations of words from very large datasets. The authors concluded that it is possible to train high quality word vectors using very simple model architectures, instead of the popular neural network models. Also, the authors proposed a Word2Vec as framework for word embedding, then, as an extension to this Word2Vec, proposed a Doc2Vec, mapping every paragraph in a document to a unique vector using the average of a paragraph id and the word vector representation.

Recently, [Yao et al. 2018] proposed an approach of a single text graph for a corpus based on word co-occurrence and document word relations, this method is discussed in more detail in Section 3.3. Other works use GCN for NLP tasks: [Marcheggiani and Titov 2017] use GCN to incorporate syntactic information in neural models; [Beck et al. 2018] propose a model for graph-to-sequence learning that leverages recent advances in neural encoder-decoder architectures; [Cetoli et al. 2017] proposed a modification to GCN architecture by introducing a bidirectional mechanism for convolving directed graph leading to an improvement in F1 score in the worked dataset.

In the presented methods, neural networks are used to create a vector representation of a word/text by using convolution [Collobert et al. 2011, Kalchbrenner et al. 2014], Chebyshev distance and angle between vectors [Tai et al. 2015]. In general, the papers presented construct text representations using TF-IDF [Salton and Buckley 1988], and little address the contextualization of terms using distance or similarity metrics. This study proposes to use the [Yao et al. 2018] approach to evaluate different measures using the accuracy in the text classification model.

3. NLP Techniques

This section presents a brief overview of the NLP techniques used in this research.

3.1. Word Embeddings and Word2Vec

Distributional vectors or word embeddings essentially follow the distributional hypothesis, according to which words with similar meanings tend to occur in similar context [Turney and Pantel 2010].

Word2Vec has been getting a lot of attention lately with multiple applications [Goldberg and Levy 2014]. This method is a two-layer neural network that processes text. With word2vec, it is possible, given a set of words in a context, to find its representation in the vector space. It is also possible to capture context and semantics among its elements from some measure of distance or similarity, such as Euclidean or Cosine distance measures.

A trained word2vec model can be sensitive to parameterization, i.e., some parameters have more impact in accuracy and similarity context, such as sub-sampling, dimensionality and context window.

3.2. Graph Convolutional Networks

The Graph Convolution Network (GCN), from [Kipf and Welling 2016] is a multi-layer neural network that operates directly on a graph and induces embedding vectors of nodes based on properties of their neighborhoods according to Equation 1:

$$H^{(l+1)} = \sigma\big(\bar{D}^{-1/2}\bar{A}\bar{D}^{-1/2}H^{(l)}W^{l}\big) \quad (1)$$

$$\bar{A} = A + I_N \quad (2)$$

$$\bar{D}_{ii} = \sum_{j} \bar{A}_{ij} \quad (3)$$

where $\bar{A}$ is the adjacency matrix of the undirected graph G with added self-connections. I_N is the identity matrix, $\bar{D}_{ii}$ is a layer-specific trainable weight matrix. $\sigma(.)$ denotes an activation function, such as the ReLU [Nair and Hinton 2010]. H^l in $\mathbb{R}^{NxD}$ is the matrix of activation in the l^{th}layer [Kipf and Welling 2016].

3.3. Text Graph Convolutional Networks

The Text GCN is a kind of GCN in which a single text graph can be built for a corpus based in word co-occurrence and document word relations. This approach has a large and heterogeneous text graph which contains words nodes and document nodes, the edges among nodes are based on word occurrence in a document (document-word edges) and word co-occurrence in the whole corpus (word-word edges), the weight of the edge between a document node and a word node is the TF-IDF [Berger and Lafferty 2017, Salton and Buckley 1988] of the word in the document, and the weight for the edges between words is applied point-wise mutual information (PMI) as shown by Equation 4.

$$PMI(i,j) = \log \frac{p(i,j)}{p(i)p(j)} \quad (4)$$

$$p(i,j) = \frac{W(i,j)}{W} \quad (5)$$

$$p(i) = \frac{W(i)}{W} \quad (6)$$

where *p(i,j)* represents the ratio of words *i* and *j* on the total of windows and *p(i)* represents the ratio of word *i* on the total of windows, such that *W(i, j)* is the number of sliding windows that contain both word *i* and *j* and *W(i)* is the number of sliding windows in a corpus that contain word *i*, and W is the total number of sliding windows in the corpus.

It is important to note that convolution in a GCN is closely linked to the adjacency matrix (constructed from the graph), denoted by the Equation 1. Therefore, the edge values present in the adjacency matrix can direct the convolution to distinct niches in the vector space, thus justifying an in-depth analysis of this aspect of graph construction.

3.4. Retrofitting

Retrofitting is a method for refining vector space representations by using relational information from semantic lexicons through encouraging linked words which have similar vectors representations. Retrofitting is used as a post-processing step to improve vector quality and is more modular than other approaches that use semantic information while training. It can be applied to vectors obtained from any word vector training method [Faruqui et al. 2015]. In this method, the objective is to minimize:

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i ||q_i - \bar{q_i}||^2 + \sum_{(i,j)\in E} \beta_{ij} ||q_i - q_j||^2 \right] \quad (7)$$

where Q is a refined objective matrix $(q_i, ..., q_n)$, such that columns are both close to original matrix $\bar{Q}$, $\bar{q_i} \in \mathbb{R}^d$, and α and β values control the relative strengths of associations. Ψ is convex in Q and its solution can be found by solving a system of linear equations. The vectors in Q are initialized to be equal to the vectors in original matrix $\bar{Q}$. An implementation that computed retrofitted vector is available from [1].

3.5. Distance and Similarity measures

Defining distance or similarity measures is essential to solve many recognition problems such as classification, clustering and retrieval problems [Cha 2007]. For text classification problem, the choice of the right measure is crucial for defining word context.

According to [Cha 2007], the distance and similarity measures can be grouped by their syntactic similarities in families:

- **Minkowski family** - Euclidean measure and generalizations
- L_1 **family** - measures group that determine more precisely the absolute difference
- **Intersection family** - similarity measures
- **Inner Product family** - deals exclusively with similarity measures which incorporate the inner product
- **Fidelity family** - uses geometric means to calculate similarity or distance
- **Squared** L_2 **family** - uses squared euclidean distance and variants
- **Shannon's entropy family** - based in the concept of probabilistic uncertainty
- **Combinations** - combination of multiple ideas or measures

In this work, distance measures in the Minkowski, L_1, Intersection, and Inner Product families [Cha 2007] were chosen. Table 1 defines the measures used.

4. Methodology

This section presents how each technique was organized in the workflow and the way communication between each layer of the model is performed.

The study's approach follows [Yao et al. 2018]. They build a text graph which contains word nodes and document nodes. In the same way as [Kim 2014], the datasets were preprocessed by cleanning them, tokenizing, removing stop words defined in NLTK[2] and then stemming words [Lovins 1968]. A word embedding model was built by using the Word2Vec implementation available from the gensim[3] library. A 200-dimensional word embedding was used, workers defined as 10, and then removed low frequency words appearing less than 5 times for each datasets.

For all documents, a word co-occurrence with context windows was built, and for each window the words frequencies where computed and a map of word's pair occurrence

[1] https://github.com/mfaruqui/retrofitting

[2] http://www.nltk.org/

[3] https://radimrehurek.com/gensim/models/word2vec.html

Table 1. Distance/Similarity Measures

Minkowski family	
Euclidean Distance	EU = $\sqrt{\sum_{i=1}^{d} \left\|P_i - Q_i\right\|^2}$
City Block Distance	CB = $\sum_{i=1}^{d} \left\|P_i - Q_i\right\|^2$
Minkowski Distance	MI = $\sqrt[p]{\sum_{i=1}^{d} \left\|P_i - Q_i\right\|^p}$
Chebyshev Distance	$CH = max_i(\left\|P_i - Q_i\right\|)$
L_1 **family**	
Bray-Curtis Similarity	BC = $1 - \frac{\sum_{i=1}^{d} \left\|P_i - Q_i\right\|}{\sum_{i=1}^{d} \left\|P_i\right\| + \sum_{i=1}^{d} \left\|Q_i\right\|}$
Canberra Distance	CA = $\sum_{i=1}^{d} \frac{\left\|P_i - Q_i\right\|}{P_i + Q_i}$
Kulczynski Distance	KK = $\frac{\sum_{i=1}^{d} \left\|P_i - Q_i\right\|}{\sum_{i=1}^{d} min(P_i, Q_i)}$
Intersection family	
Ruzicka Similarity	RU = $\frac{\sum_{i=1}^{d} min(P_i, Q_i)}{\sum_{i=1}^{d} max(P_i, Q_i)}$
Jaccard Similarity	JC = $\frac{\sum_{i=1}^{d} P_i Q_i}{\sum_{i=1}^{d} P_i^2 + \sum_{i=1}^{d} Q_i^2 - \sum_{i=1}^{d} P_i Q_i}$
Dice Similarity	DI = $\frac{2\sum_{n=1}^{d} P_i Q_i}{\sum_{i=1}^{d} P_i^2 + \sum_{i=1}^{d} Q_i^2}$
Inner Product family	
Cosine Distance	COS = $\frac{\sum_{n=1}^{d} P_i Q_i}{\sum_{i=1}^{d} P_i^2 \sum_{i=1}^{d} Q_i^2}$
Mahalanobis Distance	MA = $\sqrt{(P-Q)^T V^{-1} (P-Q)}$
Correlation Distance	CO = $\frac{(P - \bar{P}) * (Q - \bar{Q})}{\|\|(P - \bar{P})\|\|_2 \|\|(Q - \bar{Q})\|\|_2}$
V is covariance matrix $\bar{P}$ is the mean of the elements	

was built. Then, a graph is created by using words that appear in a map of word's pair occurrence and words that appear in the documents. Each word or document is a node of the graph.

Formally, the weight of a edge *(i,j)* is defined depending on the nature of the measure (distance or similarity) according to Equation 8.

$$A_{ij} = \begin{cases} M(i,j), & \text{if } i,j \text{ are words} \\ TF - IDF_{ij}, & \text{if } i \text{ is document , } j \text{ is word} \\ 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

The $M(i,j)$ is defined according to the approach of the metric adopted according to Equation 9:

$$M(i,j) = \begin{cases} SIM(i,j), & \text{if similarity measure} \\ DIST(i,j), & \text{otherwise} \end{cases} \quad (9)$$

$$SIM(i,j) = log(\frac{s_{ij} * c_{ij}^2}{w}) \quad (10) \qquad DIST(i,j) = log(\frac{c_{ij}^2}{w * d_{ij}}) \quad (11)$$

where s_{ij} and d_{ij} are, respectively, the similarity and distance values between vectors *i* and *j*, c_{ij} is the number of times in which the word *i* and word *j* appears in the same window. The adjacency matrix obtained was used (from the graph) as input into GCN [Kipf and Welling 2016].

Datasets. The evaluation of the algorithms was conducted using five widely used benchmark corpora including 20-Newsgroups (20NG), Ohsumed, R52, R8 and Movie Review (MR).

- **20NG**[4] dataset (bydate version) contains 18,846 documents evenly categorized into 20 different categories. In total, 11,314 documents are in the training set and 7,532 documents are in the test set.
- **Ohsumed**[5] corpus is from the MEDLINE database, which is a bibliographic database of important medical literature maintained by the National Library of Medicine, 3,357 documents are in the training set and 4,043 documents are in the test set.
- **R52** and **R8**[6] are two subsets of the Reuters 21578 dataset. R8 has 8 categories, and was split to 5,485 training and 2,189 test documents. R52 has 52 categories, and was split to 6,532 training and 2,568 test documents.
- **MR**[7] is a movie review dataset for binary sentiment classification. The corpus has 5,331 positive and 5,331 negative reviews.

Table 2 summarizes the values found for each dataset. The number of words is potentially different from the number of nodes due to two main reasons: 1) in the study's model, each document is also represented by a node. 2) Not all words present in the dataset will become a node in the representation, mainly due to the low frequency reason.

Table 2. Descriptive statistics of each dataset

Dataset	Doc	Words	Pairs	Nodes	Edges	Classes
20NG	18,846	34,135	26,660,361	52,981	26,874,061	20
R8	7,674	6,792	3,435,083	14,466	4,024,898	8
R52	9,100	7,890	4,368,634	16,990	5,118,159	52
Ohsumed	7,400	13,323	8,844,431	20,723	8,384,734	23
MR	10,662	20,473	1,697,773	31,135	1,665,491	2

For GCN, the parameters follow [Yao et al. 2018]. We set the embedding size of the firth convolution layer as 200 and set the window size as 20, learning rate as 0.02, dropout rate as 0.5, L_2 loss weight as 0. The network was trained for 200 epochs.

[4]http://qwone.com/ jason/20Newsgroups/

[5]http://disi.unitn.it/moschitti/corpora.htm

[6]https://www.cs.umb.edu/smimarog/textmining/datasets/

[7]http://www.cs.cornell.edu/people/pabo/movie-review-data/

For each experiment, two different representations of the model were built. First, vectors from Word2Vec model were used and then different measures were employed to build the adjacency matrix. After that, techniques of retrofitting to reduce the distance between word synonyms were used. The code for the experiment is publicly available [8].

The following classification evaluation measures were collected: accuracy, F1-Score, and mean square error. We perform 30 independent rounds on ten training/test sets generated by the 10-fold cross-validation process reporting the average of the selected metrics. In this paper, the accuracy values achieved by the experiments are reported.

5. Results and discussion

This section presents the main results and some small experiments that help to understand the model and illustrate the approach. The results are presented according to the use or not of the retrofitting technique in order to evaluate if in the context of the presented problems, such technique assists in the model.

Table 3 shows the average values and standard deviation (immediately below in parenthesis) for the best measure found for each dataset in each method (I and II). A multiple comparison test was performed using Kruskal-Wallis [Kruskal and Wallis 1952] between all algorithms and all models, with a p-value of *0.05*. In all tests, in this comparison, it is possible to analyze if there is a statistical difference between using or no the retrofitting technique.

Table 3. Comparison test for method I and method II. p-value: 0.05

Dataset	Method I		Method II		Diff.
	Mean	**Best measure**	**Mean**	**Best measure**	
20ng	**0.4477** (0.0028)	Dice	0.4432 (0.0019)	Cityblock	**19.20**
R52	**0.9114** (0.0061)	Ruzicka	0.9093 (0.0021)	Minkowski	**12.06**
R8	0.9526 (0.0013)	Cosine	**0.9545** (0.0014)	Cosine	**28.06**
Ohsumed	**0.6704** (0.0016)	Euclidean	0.6663 (0.0015)	Correlation	**30.00**
MR	0.7358 (0.0029)	Euclidean	**0.7404** (0.0029)	Chebyshev	**28.80**

The retrofitting method, according to the experiments performed, was not sufficiently able to condense contexts between words through their synonyms. The use of synonyms can create closer vectors. However, although not having significant improvement, the retrofiting technique is still recommended when used in conjunction with specific embbeding for each dataset (as it was used with a set of generic synonyms).

Table 4 shows the average accuracy result of each measure for each analyzed dataset without using the retrofitting method. It is important to note that the Euclidean measure has the most constant results among the analyzed metrics even if it does not produce the best average result for the datasets.

[8] https://github.com/dimmykarson/cgcn

Table 4. Results for each measure of distance/similarity using method I

Measure	20NG	R52	R8	Ohsumed	MR
Braycurtis	0.4419	0.8974	0.9414	0.6668	0.7192
Canberra	0.4422	0.9082	0.9410	0.6643	0.7319
Chebyshev	0.4470	0.8735	0.9450	0.6593	0.7309
Cityblock	0.4422	0.9053	0.9450	0.6628	0.7337
Correlation	0.4423	0.8783	0.9523	0.6604	0.7257
Cosine	0.4403	0.8424	**0.9526**	0.6636	0.7298
Dice	**0.4477**	0.8982	0.9327	0.6668	0.7349
Euclidean	0.4438	0.9081	0.9431	**0.6704**	**0.7358**
Jaccard	0.4404	0.9019	0.9474	0.6665	0.7300
Kulczynski	0.4250	0.8473	0.9522	0.6700	0.7322
Mahalanobis	0.3011	0.2591	0.5025	0.1927	0.5157
Minkowski	0.4443	0.9023	0.9409	0.6671	0.7331
Ruzicka	0.4467	**0.9114**	0.9490	0.6637	0.7292

The model shows a stability of results when using the Euclidean measure, due to this, Table 5 was highlighted. In this table, we can observe the difference between the results of representations using Euclidean measure and the other ones for method I. Bold values indicate that there is a statistical difference between the measures.

Table 5. Multiple comparison test after Kruskal-Wallis using euclidean distance. p-value: 0.05, critical difference: 108.3002

Measure	Datasets				
	20ng	R52	R8	Ohsumed	MR
Bay-curtis	57.35	**209.35**	39.23	85.83	**275.33**
Canberra	89.40	13.85	46.96	**160.65**	89.56
Chebyshev	20.61	105.23	44.98	**283.98**	**124.55**
City Block	103.35	67.28	49.16	**204.58**	50.70
Correlation	96.95	6.80	**185.20**	**270.65**	**229.96**
Cosine	105.63	**129.81**	**193.31**	**188.48**	**148.85**
Dice	27.75	**184.48**	**142.73**	85.75	13.68
Jaccard	**141.51**	**134.20**	94.31	101.48	**134.08**
Kulczynski	**168.20**	29.58	**94.31**	6.60	85.05
Mahalanobis	**282.26**	**293.55**	**172.93**	**361.83**	**338.50**
Minkowski	16.28	99.20	47.50	72.38	69.45
Ruzicka	12.31	38.41	**121.38**	**181.60**	**149.35**

In order to refine the analysis of the impact of the measure, we have chosen Ohsumed dataset and build different text representations, varying the word embeddings size. The ohsmed dataset was chosen because of the large amount of words and edges it has, making it a difficult context-sensitive dataset. Figure 2 shows the test accuracy with different word embedding dimensions.

The substantial increase in the size of the text representation vector is not directly linked to the effectiveness of the method. Figure 2 shows that accuracy of the approaches

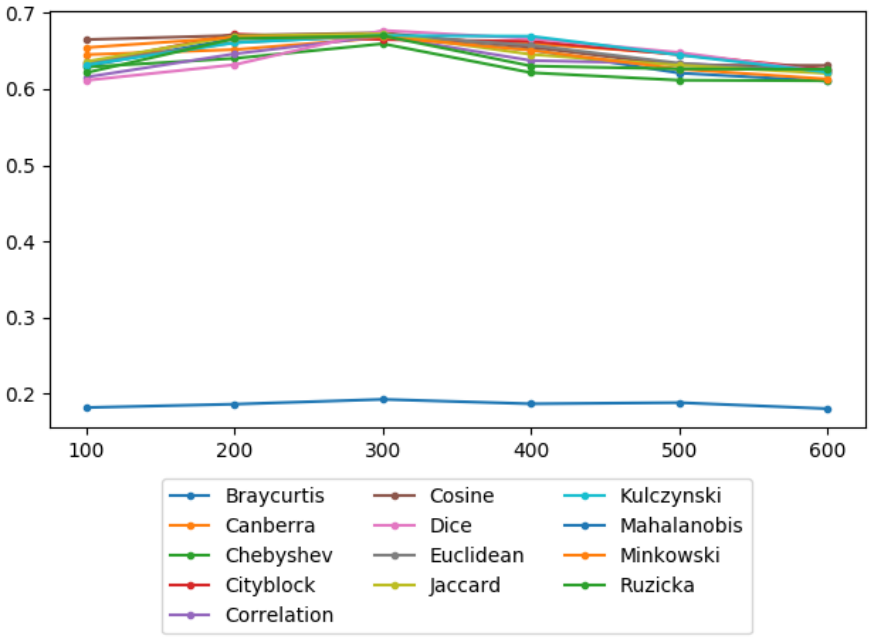

Figure 2. Test accuracy with different word embedding dimensions in dataset Ohsumed

of all the metrics fall or stabilize when they reach a large length, i.e, the word embedding model determines unnecessary or unrepresentative characteristics for the analyzed dataset. It can be proved by the convolution layer of the approach that it ultimately discards most of the surplus features.

[Melamud et al. 2016] propose that picking the optimal dimensionality is critical for obtaining the best accuracy on NLP tasks and that increased dimensionality is not directly linked to improved accuracy, further Improvements can often be achieved by combining complementary word embeddings of different context types with the right dimensionality. The Ohsmed dataset is an example of this, the dataset has 13k single words and 20k nodes, however it has 8 million edges, with this amount of words and relationships, the increase in dimensionality does not reflect the increase in accuracy.

The approach achieved poor results for the dataset 20NG. The number of words presented and number of relations (edges) caused a context dispersion, i. e. , the neural network was not able to refine the presented features through the amount of convolution chosen. The complete results, including all statistical tests is publicly available[9].

6. Conclusion and Future Works

In this study, an empirical comparison between distance/similarity measures for NLP using Text GCN was proposed. The objective was to increase the set of contextual evaluation possibilities between words in a document, adding a contextual bias, especially by the use of word2vec, instead of using TF-IDF. Furthermore, the aim to investigate the impact of using the retrofitting method to emphasizing contextual similarities between words given their chain of synonyms. A graph was built by using words and document as

[9]https://github.com/dimmykarson/cgcn/results

nodes and a distance/similarity measure as weight to edges in order to create a contextual relationship between words.

The results validate the hypothesis that the choice of an appropriate distance/similarity measure directly impacts the generalizability of the model, some metrics capture context better than others, as is the case with Euclidean and Cosine measure. The presence of a statistical difference between the metrics for the same dataset indicates different behaviors according to the dataset configuration (number of words, edges or arrangement in vector space) that can direct future work in the area.

Acknowledgments

This work was funded by CAPES and Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq - Brazil, Tribunal de Justiça do Estado do Piauí - TJPI and supported by Intel® AI DevCloud.

References

[Beck et al. 2018] Beck, D., Haffari, G., and Cohn, T. (2018). Graph-to-sequence learning using gated graph neural networks. *CoRR*, abs/1806.09835.

[Berger and Lafferty 2017] Berger, A. L. and Lafferty, J. D. (2017). Information retrieval as statistical translation. *SIGIR Forum*, 51(2):219–226.

[Cetoli et al. 2017] Cetoli, A., Bragaglia, S., O'Harney, A. D., and Sloan, M. (2017). Graph convolutional networks for named entity recognition. *CoRR*, abs/1709.10053.

[Cha 2007] Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1.

[Collobert et al. 2011] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. P. (2011). Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

[Faruqui et al. 2015] Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E. H., and Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In Mihalcea, R., Chai, J. Y., and Sarkar, A., editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1606–1615. The Association for Computational Linguistics.

[Goldberg 2016] Goldberg, Y. (2016). A primer on neural network models for natural language processing. *J. Artif. Intell. Res.*, 57:345–420.

[Goldberg and Levy 2014] Goldberg, Y. and Levy, O. (2014). word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722.

[Kalchbrenner et al. 2014] Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188.

[Kim 2014] Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

[Kipf and Welling 2016] Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907.

[Kruskal and Wallis 1952] Kruskal, W. H. and Wallis, W. A. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621.

[Le and Mikolov 2014] Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196.

[Liu et al. 2016] Liu, P., Qiu, X., and Huang, X. (2016). Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879.

[Lovins 1968] Lovins, J. B. (1968). Development of a stemming algorithm. *Mech. Translat. & Comp. Linguistics*, 11(1-2):22–31.

[Luo 2017] Luo, Y. (2017). Recurrent neural networks for classifying relations in clinical notes. *Journal of Biomedical Informatics*, 72:85–95.

[Marcheggiani and Titov 2017] Marcheggiani, D. and Titov, I. (2017). Encoding sentences with graph convolutional networks for semantic role labeling. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1506–1515. Association for Computational Linguistics.

[Melamud et al. 2016] Melamud, O., McClosky, D., Patwardhan, S., and Bansal, M. (2016). The role of context types and dimensionality in learning word embeddings. *CoRR*, abs/1601.00893.

[Mikolov et al. 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[Nair and Hinton 2010] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814.

[Salton and Buckley 1988] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523.

[Socher et al. 2013] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642.

[Tai et al. 2015] Tai, K. S., Socher, R., and Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075.

[Turney and Pantel 2010] Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res.*, 37:141–188.

[Yao et al. 2018] Yao, L., Mao, C., and Luo, Y. (2018). Graph convolutional networks for text classification. volume abs/1809.05679.

[Young et al. 2018] Young, T., Hazarika, D., Poria, S., and Cambria, E. (2018). Recent trends in deep learning based natural language processing [review article]. *IEEE Comp. Int. Mag.*, 13(3):55–75.