HAL open science

Terminology-based Text Embedding for Computing Document Similarities on Technical Content

Hamid Mirisaee, Éric Gaussier, Cedric Lagnier, Agnes Guerraz

To cite this version:

Hamid Mirisaee, Éric Gaussier, Cedric Lagnier, Agnes Guerraz. Terminology-based Text Embedding for Computing Document Similarities on Technical Content. TIA-PFIA 2019 (atelier TALN-RECITAL), 2019, Toulouse, France. hal-03352371

HAL Id: hal-03352371

https://hal.science/hal-03352371

Submitted on 23 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Terminology-based Text Embedding for Computing Document Similarities on Technical Content

Hamid Mirisaee[1] Eric Gaussier[2] Cedric Lagnier[1] Agnes Guerraz[1]
(1) Skopai, Grenoble, France
(2) LIG, University of Grenoble Alps, Grenoble, France
Hamid.Mirisaee@skopai.com, Eric.Gaussier@imag.fr
Cedric.Lagnier@skopai.com, Agnes.Guerraz@skopai.com

RÉSUMÉ

Nous proposons dans cet article une nouvelle approche hybride de calcul de représentation de documents dans le but de calculer des similarités entre documents techniques. Dans un premier temps, nous utilisons une méthode de l'état de l'art basée sur le graphe des mots pour extraire des phrases-clés, afin de calculer un score pour chacune des phrases du document. Nous proposons deux approches pour calculer la représentation d'un document en utilisant ce classement des phrases. En utilisant des annotations d'experts du domaine, nous montrons que les méthodes proposées obtiennent de meilleurs résultats en terme de NDCG pouvant aller jusqu'à 27% d'amélioration.

ABSTRACT

We propose in this paper a new, hybrid document embedding approach in order to address the problem of document similarities with respect to the technical content. To do so, we employ a state-of-the-art graph techniques to first extract the keyphrases (composite keywords) of documents and, then, use them to score the sentences. Using the ranked sentences, we propose two approaches to embed documents and show their performances with respect to two baselines. With domain expert annotations, we illustrate that the proposed methods can find more relevant documents and outperform the baselines up to 27% in terms of NDCG.

MOTS-CLÉS : Représentation de document, similarité entre documents, k-core, phrases-clés.

KEYWORDS: Document embedding, document similarity, k-core, keyphrase.

1 Introduction

Many Machine Learning (ML) applications require the calculation of similarities between instances of the task under consideration. Those instances could be of any nature such as numerical and/or categorical time series (Aghabozorgi *et al.*, 2015), gene expression (Wang *et al.*, 2017) or textual documents (Nikolentzos *et al.*, 2017; Song & Roth, 2015).

Perhaps the application of similarity assessment between objects that is vastly used on an everyday basis is the one used in search engines. As the most evident example, when a user is looking, via a given query and a search engine such as Google, for relevant documents, there is basically a matching process happening behind. In such application, the user provides a query, as a set of keywords, and

looks for the documents which best correspond to those keywords. This mainly leads to a ranking problem where the goal is to rank all the available documents (webpages in case of a web search engine) and provide the user with the most relevant documents. Session search (Guan *et al.*, 2012) is an interesting challenge of such applications where the user reformulates the query based on the results returned by the system until the desired documents are found.

As one can notice, the principal notion of the above-mentioned task is the similarity between instances. For example, a very common similarity measure between textual documents is the tf-idf which is based on two notions : the frequency of the terms appearing in the document, *tf*, and the importance of those terms through the entire set of documents, *i.e.* inverse document frequency, *idf*. Many studies still use tf-idf to perform text-related task as it can, for many tasks, properly project the textual data into the numerical space such that the content is reflected accordingly. One can then use the tf-idf vectors of documents to measure the similarity between them, via cosine for instance. The tf-idf representation is widely used in different applications such as document clustering (Bafna *et al.*, 2016) and topic modeling (Zhu *et al.*, 2013).

As one can see, tf-idf embeds the document by operating at the word-level, *i.e.* it takes the tf-idf score of each word and represent the document as a (sparse) vector of size of the entire vocabulary. There is, however, another recent word-level operated technique, namely word2vec (Mikolov *et al.*, 2013a), which also aims at representing words. Nevertheless, there are two main differences between these two approaches. Firstly, tf-idf assigns a scalar to each word while word2vec represents each word by a vector. Secondly, and more importantly, word2vec embeds each word using its context, *i.e.* its surrounding words. That being said, it is able to capture the semantical context of words. The latter has shown very good results in many applications such as text clustering and text classification (Wang *et al.*, 2015b; Lilleberg *et al.*, 2015).

As mentioned previously, calculating document similarity and, consequently, finding similar documents is at the core of many ML tasks. Sometimes, however, focusing on the entire document may not lead to capturing desired similar documents as not all parts of a document have the same importance level. For instance, if a document describes a novel device for people suffering from diabetes, then taking the entire document may not necessarily result in the similar documents talking about the same particular issue, but rather about the medical domain in general. Note that this issue is different from data cleaning, and should be rather considered as data selection/weighting for the task of document embedding.

In this paper, we investigate the above-mentioned problem and propose a technique for document embedding w.r.t. technical content of the documents. We show that the proposed method is able to find better similar documents once the technical content is concerned which, to the best of our knowledge, is the first research targeting this objective. To do so, we propose to first capture the keyphrases, *i.e.* composite keywords, of the document and, then, rank the sentences based on the keyphrases they contain. To detect keyphrases, we use the state-of-the-art technique explained in (Rousseau & Vazirgiannis, 2015) which is based on the k-core concept of graph-of-words. We then use the two previously-mentioned embedding techniques to represent the text and show that the proposed, hybrid method can efficiently capture relevant documents based on the technical content.

The reminder of this paper is organized as follows : Section 2 overviews the most related studies. Section 3 first details the graph representation of documents and the k-core concept, before providing the framework of embedding documents using the sentence ranking obtained via the k-core approach. Then, in Section 4, we describe the experimental settings and the baselines as well as the collected dataset. We then report our results in the same section. Finally, Section 5 concludes the paper.

2 Related work

Over the past few decades, a large body of studies has been considering the relatedness of documents for a wide range of tasks such as text categorization or document classification. Assessing the similarities between documents is at the core of many machine learning applications such as information retrieval, recommendation systems and text generation. In this section, we consider the most relevant studies with respect to the topic under investigation and position our work in regard to them.

Classically, the similarity between two documents has been measured by the cosine similarity between their tf-idf vectors, see (Trstenjak *et al.*, 2014) and (Schultz & Liberman, 1999) for example. Alternatively, studies like (Zhang *et al.*, 2011) considered other techniques such as LSI and multi-word methods and investigated their performance in different tasks. Although tf-idf can still capture many characteristics of a document and boost the performance of many tasks such as topic modeling (Hong & Davison, 2010; Mehrotra *et al.*, 2013), they still fail to detect the entire context of a document such as the semantic relation between words (Tapaswi *et al.*, 2016).

The paper of Mikolov et al. (Mikolov *et al.*, 2013a) introduced an entire new idea of textual representation where a document is analyzed at the word level and each word is represented based on its context, *i.e.* it carries semantic features of the document. The context of the word is defined by the co-occurring words. The model is simply a neural network where the vectorial embedding of a word is determined by those of its surrounding words. Many variations of this approach have been later studied to, for instance, operate directly at the document level (Lau & Baldwin, 2016; Mikolov *et al.*, 2013b). In addition, the combination of tf-idf and word2vec has been widely used and shown to bring significant improvement in many applications (Lilleberg *et al.*, 2015; Acosta *et al.*, 2017). As it will be illustrated in Section 4, we will make use of the combination of these two techniques to conduct a part of our experiments, and will show that they work very efficiently in capturing the technical side of documents.

Measuring the similarity between documents has been widely studied in the literature through different approaches. For instance, (Cooper *et al.*, 2002a) used a phrase recognition approach to detect similar documents, and (Pereira & Ziviani, 2003) employed radix tree to calculate similarities between web documents. In a different approach, (Paul *et al.*, 2016) introduces a graph-based method to exploit the hierarchical relations in order to efficiently calculate the similarity between documents. Similarly, (Wang *et al.*, 2015a) proposes to represent documents as typed heterogeneous information networks and, following the notion of graphs, it computes the distance between documents for the task of document clustering.

Perhaps the most related studies to ours are (Brants & Stolle, 2002) and (Cooper *et al.*, 2002b). In (Brants & Stolle, 2002), the authors propose to use a word-based method to capture similar documents via PLSA. The document matching in this particular work is done based on the words appearing in the document. More precisely, the similarity between two documents is defined as linear combination of the cosine between the tf-idf vectors and the PLSA-based representations of the words. Although the objective of this paper is similar to our work in that they also rely on words-basis scoring, two main differences distinguish that research and ours. Firstly, we focus on keyphrases and pay more attention on the parts of the text where those keyphrases are used, while (Brants & Stolle, 2002) relies on the entire content of the document. Secondly, the present study uses the context-based techniques, such as word2vec to particularly target the semantic aspects of a document. In other words, if two documents describe the similar subjects with different vocabulary, the purely tf-idf based or topic modeling based approaches fail to see their similarities, while context-based techniques are able to capture that. In

a similar manner, (Cooper *et al.*, 2002b) defines two documents to be similar if they have the same pieces of text such as sentences or paragraph, which has similar limitations as mentioned before. This current study, to the best of our knowledge, is the first one to propose technical content based similarity between documents using their semantic terminology.

3 Framework

Graph-based techniques have been widely used in representing textual documents, where, in general, entities of the text construct the nodes and the relation between them defines the edges. Depending on the task in mind, the nodes could be words, phrases, sentences, etc. (Parveen & Strube, 2014; Mihalcea & Tarau, 2004; Erkan & Radev, 2004) are only a few examples of such approach.

Following this line of thought, (Rousseau & Vazirgiannis, 2015) investigated the task of keyword extraction by representing a document as a *graph-of-words* and retrieving the *main core* of the graph. In the following, we briefly explain how this task is accomplished and, then, use the results of this technique to rank the sentences and eventually embed a document.

3.1 Graph-of-words

In (Rousseau & Vazirgiannis, 2015), the authors propose to represent a document as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. In a graph-of-words, nodes are basically the words and edges are the proximity degree between them. More precisely, if two words of the document, say $w_i \in \mathcal{V}$ and $w_j \in \mathcal{V}$, co-occur within a fixed-size sliding window, then $e_{ij} \in \mathcal{E}$. Note that $\mathcal{E}$ can be weighted or unweighted. In the weighted setting, the weight of e_{ij} is the number of times that w_i and w_j co-appear in the sliding window within a document. One can also consider a directed (weighted/unweighted) version of where the order of words in the document determines the direction of edges.

With such representation of the document, (Rousseau & Vazirgiannis, 2015) proposes a technique which focuses more on cohesiveness of the nodes, rather than classic methods, such as (Litvak & Last, 2008) and (Mihalcea & Tarau, 2004), which rely on the notion of centrality. To do that, the *k-core* approach (Seidman, 1983) has been employed :

Definition 1 *$\mathcal{H} = (\mathcal{V}', \mathcal{E}')$ is called a k-core of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if $\mathcal{E}' \subset \mathcal{E}$, $\mathcal{V}' \subset \mathcal{V}$ and $\forall v \in \mathcal{V}'$, $Deg(v) \geq k$, where $Deg(.)$ denotes the degree of nodes. The core of maximum order is then called the main core.*

The idea introduced in (Rousseau & Vazirgiannis, 2015) is to take all the nodes of the main core as keywords. Via extensive experiments, the authors show that their approach outperforms the traditional methods such as HITS and PageRank (Litvak & Last, 2008; Mihalcea & Tarau, 2004). Additionally, unlike other techniques that need the number of keywords to extract, the size of the main core basically handles this issue. On top of that, this technique can effectively be used to detect keyphrases, *i.e.* composite keywords(Rousseau, 2015). Once again, in such context, each word of the text is represented as a node in the graph and two connected nodes can potentially construct a keyphrase.

For example, in the sentence "our platform is based advanced artificial intelligence techniques",

both "artificial" and "intelligence" can be considered as keywords, and "artificial intelligence" can construct a keyphrase. Note that if the size of sliding window explained above is larger than 2, then keyphrases like "artificial techniques" and "advanced intelligence" can also be extracted from the graph as keyphrases.

With that in mind, one can simply use the nodes of the main core to construct the keyphrases using different approaches, by for instance taking the words corresponding to the connected nodes. We use a slight modification of the k-core method in order to, first, extract the keyphrases of size 2, *i.e.* combination of two and only two words, and, then, rank the sentences. Using those ranked sentences, we propose a technique to embed the document such that it encodes the main technical content of the document. The proposed algorithm is detailed in the following.

3.2 `TDE` : Terminology-based Document Embedding

As explained above, the k-core, graph-based methods can efficiently be used to extract keywords or keyphrases, where keyphrases often better reflect the semantics of the document as they tend to reduce the noise significantly. Knowing that the objective of this study is to eventually embed a document w.r.t. its technical content through the keyphrases, we first establish a link between the embedded information in the graph-of-words and the resultant keyphrases. We then score the sentences based on the keyphrases they contain and, finally, calculate the embedding of the document via its scored sentences. In other words, the embedding of a document is a weighted average of the embedding of its sentences, where the weights are derived from the graph-of-words which has been explained above.

Let $\mathcal{C}$ be the set of all cores of the graph-of-words of document d. Also, let $\mathcal{T}_c$ be the set of all keyphrases appearing in core $c \in \mathcal{C}$. More formally, $\mathcal{T}_c = \{(t,t')|t \in c \wedge t' \in c\}$. Then, the embedding of the document d, denoted as $\vec{d}$, can be calculated as the weighted average of the embedding of its sentences :

$$\vec{d} = \frac{1}{\sum_{s \in S} \Gamma(s)} \sum_{s \in S} \vec{s} \times \Gamma(s) \tag{1}$$

where S is the set of sentences of d, $\vec{s}$ is the embedding of the sentence s and $\Gamma(s)$ is the score of sentence s. The score of each sentence can then be computed as :

$$\Gamma(s) = \sum_{c \in \mathcal{C}} \sum_{\substack{(t,t') \in \mathcal{T}_c \\ (t,t') \in s}} \phi\big((t,t')\big) \tag{2}$$

where the function $\phi(.)$ returns a score for each keyphrase (t,t') using the degree of the edge between t and t' :

$$\phi\big((t,t')\big) = Deg(t,t') \times F(c) \tag{3}$$

where $F(.)$ is a function that assigns a weight to each core c such that cores get monotonically decreasing weights. Obviously the main core has the highest weight.

In our experiments, we use the rational function $F(x) = \frac{1}{x}$ where $x = 1$ for the main core, $x = 2$ for the next core and so on. We implemented the graph-words as an undirected weighted graph where the number of co-occurrences of two words determines the weight of the edge linking them. One should note that the directed version has also been investigated and, according to our observations, was not as efficient as the undirected one. For the embedding of the sentences (Eq. (1)) we propose

two different techniques which are further explained in the next Section. Finally, it should be noted that unlike (Rousseau & Vazirgiannis, 2015), we use all the cores of the graph as it is shown in Eq. (2) where the more focus is still on the most important cores as expressed in Eq. (3).

4 Experiments

4.1 Baselines

To evaluate the validity of the proposed technique, we compare the method explained in Section 3 with two document embedding baselines. As mentioned in Section 2, many studies investigate the problem of document embedding as an end-to-end problem where the document is embedded all at once, *i.e.* the embedding is learned at the document level. Accordingly, we chose, as our first baseline, the document embedding method investigated in (Lau & Baldwin, 2016) which is a robust improvement over the original *doc2vec* (Le & Mikolov, 2014). This baseline will be denoted as `D2V`.

Following the ideas presented in (Wieting *et al.*, 2016) and (Arora *et al.*, 2016), one can compute the document embedding by averaging the embedding of the words appearing in the document. The word embeddings can be learned via classical methods such word2vec explained in (Mikolov *et al.*, 2013a). According to our experiments, tf-idf weighted average performs significantly better than simple averaging and, consequently, we chose to use it as the second baseline. This baseline is referred to as `TWA` (Tf-idf Weighted Average).

These two baselines are then compared to the graph-based methods discussed in Section 3. As mentioned in that section, a document is represented as a weighted average of the embeddings of the sentences appearing in it, where the weights are the score of sentences which, in turn, are calculated using the graph-based representation of the document (see Section 3). This technique is denoted as `TDE` (Terminology-based Document Embedding).

In `TDE`, the embedding of a sentence can be calculated in different ways. Here, we propose two ways of doing that : the first one consists of using the models which are trained to directly produce the sentence embedding ; and the second one is to use the embeddings of the words forming the sentence. For the former, the state-of-the-art sentence embedding technique, described in (Pagliardini *et al.*, 2018) is employed. For the latter, we perform a tf-idf weighted average of the words constructing the sentence to calculate the embedding of the sentence. One should note that as in a sentence, it is rarely the case that we have repetition of words, the calculation is almost an idf weighted averaging. To avoid an ambiguity, the first approach is denoted as $\texttt{TDE}_{\texttt{s2v}}$ (for sent2vec) and the second as $\texttt{TDE}_{\texttt{iw}}$ (for idf weighted). In the following, we explain our dataset from which the embeddings of words, sentences and documents are learned.

4.2 Dataset, models and content filtering

We crawled websites of around 68K startups (all around the world, with no constraint on the area of work) with a total number of 3.4M webpages. After filtering those without sufficient textual information or non-English ones, we ended up with around 43K startups and 2.8M pages. We only used the English sentences of the pages for training our models and performing the evaluations.

To train the word2vec, we used *gensim* [1] on the sentences extracted from the 2.8M pages mentioned above, with a minimum count of words and window size equal to five. The number of sentences extracted reached to 950K. We used the authors' implementation [2] of (Pagliardini *et al.*, 2018) to train the sent2vec model.

Furthermore, keeping in mind that the objective of this study is to investigate document similarities based on the technical content, not all parts of a document are of interest. Knowing that each document in our dataset is the combination of all the textual content of a startup's website, many parts of that can possibly be considered as noise. For instance, the pages describing privacy policies or legal information must be ignored before performing any document embedding process. As a result, we used multiple classifiers, trained using thousands of pages, to filter out such contents. On top of that, not all the remaining content contribute to the technical aspect of a document. As a consequence, we used several ranking and filtering techniques to capture sufficiently large amount of relevant information to embed. However, the details of these processes are beyond the scope of this paper.

4.3 Evaluation

To fairly evaluate the performance of the above-mentioned baselines, `TWA` and `D2V`, with that of the proposed ones, $\texttt{TDE}_{\texttt{s2v}}$ and $\texttt{TDE}_{\texttt{iw}}$, we adapted the following strategy : we selected a set of 100 documents (startups) from four different domains : medical, agriculture, energy and biology. Evidently, the texts of these documents have not been used for training the above-mentioned models.

For each test case and for each method, we extracted the 5 most similar startups (using cosine similarity between the indexed embedding of each method and the entire dataset). The results are then combined, shuffled and assigned to the corresponding domain expert for evaluation. The experts then give a score to each of the startups from the score set $\{1, 2, 3, 4, 5\}$ where 1 denotes the least relevant and 5 denotes the most relevant. Note that the experts have been clearly informed that the combined list is not in any way ordered/ranked and the outputs of a test case should be evaluated independently. Obviously, the results do not necessarily contain all the possible scores.

To assess the performance of the proposed approach, we use NDCG (Normalized Discounted Cumulative Gain) which perfectly matches our experimental settings. We report here two values, namely NDCG@1 and NDCG@5, to investigate the behavior of each technique. In the following, we discuss the results of our experiments and illustrate the efficiency of the proposed methods.

4.4 Results

Table 1 shows the results of our experiments on all four approaches detailed in Section 4.1 using NDCG@1 and NDCG@5, compared via t-test at 5% for the significance test. As one can notice, the performance of `D2V` falls way below other methods with both NDCG@1 and NDCG@5. That could be explained by the fact that `D2V` needs long documents to be trained properly, and since our documents are only English sentences of only a part of a website, they could be as short as few sentences. As a consequence, `D2V` has difficulties to project the context into the numerical space. As a result, we will not investigate this technique in our discussion below.

1. https://radimrehurek.com/gensim/
2. https://github.com/epfml/sent2vec

	D2V	TWA	TDE_{iw}	TDE_{s2v}
NDCG@1	0.26	0.54	0.63	**0.69**
NDCG@5	0.24	0.60	0.60	**0.65**

TABLE 1 – NDCG@1 and NDCG@5 on all four methods. The best approach is shown in bold.

The second baseline, *i.e.* TWA, performs well w.r.t. the proposed methods ; as one can observe, it achieves the same NDCG@5 compared to TDE_{iw}. However TDE_{s2v} outperforms this baseline by **8%**. When it comes to NDCG@1, both graph-based techniques outperform the baselines significantly : TDE_{iw} is around **16%** better that TWA and TDE_{s2v} has a better NDCG@1 value by **27%** w.r.t. TWA.

These experiments show that the proposed technique, be it with the tf-idf weighting or with the sent2vec method, outperform the baselines. In other words, they are able to capture better similar documents when the technical content is of interest. Additionally, according to the results reported on NDCG@1, the proposed methods can find way better results when finding the most relevant document is the task in mind.

5 Conclusion

In this paper, we have studied the problem of finding similar documents with respect to technical content. To do that, we proposed a hybrid approach which employs the graph representation techniques and sentence embedding. Using the graph-of-words representation of a document and computing the cores of the graph, we first extract the keyphrases of the document of all cores of the graph. We then use this information to assign a score to each sentence ; once the scores are calculated, we use the embedding of each sentence and its score to calculate the embedding of the document. We proposed two approaches to embed sentence : (*i*) sent2vec, a state-of-the-art technique for sentence embedding, and (*ii*) a tf-idf weighted average of the words appearing in the sentence. We then compared those to two baselines, doc2vec, an existing method in the state-of-the-art to embed documents, and a tf-idf weighted average of the words appearing in the document.

As dataset, we used 2.8M webpages of 43K startups that we crawled from the web, where we considered the combination of all webpages of a startup as a document. To evaluate, we asked human experts to score the output of all techniques. Using the NDCG metric, we illustrated that the proposed metric can outperform the baselines up to 27% and, as a result, can provide better similar documents when technical content is concerned.

Acknowledgments

The authors would like to thank the domain experts of Skopai, Adeline Tarantini, Olivier Berengario, Elie Gehin and Guillaume Emery who spend a considerable amount of time to carefully evaluate the results.

Références

ACOSTA J., LAMAUTE N., LUO M., FINKELSTEIN E. & ANDREEA C. (2017). Sentiment analysis of twitter messages using word2vec. *Proceedings of Student-Faculty Research Day, CSIS, Pace University*, p.$\tilde{7}$.

AGHABOZORGI S., SHIRKHORSHIDI A. S. & WAH T. Y. (2015). Time-series clustering–a decade review. *Information Systems*, **53**, 16–38.

ARORA S., LI Y., LIANG Y., MA T. & RISTESKI A. (2016). A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, **4**, 385–399.

BAFNA P., PRAMOD D. & VAIDYA A. (2016). Document clustering : Tf-idf approach. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, p. 61–66 : IEEE.

BRANTS T. & STOLLE R. (2002). Finding similar documents in document collections. In *Using Semantics for Information Retrieval and Filtering : State of the Art and Future Research. Workshop at LREC-2002*.

COOPER J. W., CODEN A. R. & BROWN E. W. (2002a). Detecting similar documents using salient terms. In *Proceedings of the eleventh international conference on Information and knowledge management*, p. 245–251 : ACM.

COOPER J. W., CODEN A. R. & BROWN E. W. (2002b). A novel method for detecting similar documents. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, p. 1153–1159 : IEEE.

ERKAN G. & RADEV D. R. (2004). Lexrank : Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, **22**, 457–479.

GUAN D., YANG H. & GOHARIAN N. (2012). *Effective structured query formulation for session search*. Rapport interne, DTIC Document.

HONG L. & DAVISON B. D. (2010). Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, p. 80–88 : acm.

LAU J. H. & BALDWIN T. (2016). An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv :1607.05368*.

LE Q. & MIKOLOV T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning*, p. 1188–1196.

LILLEBERG J., ZHU Y. & ZHANG Y. (2015). Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, p. 136–140 : IEEE.

LITVAK M. & LAST M. (2008). Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, p. 17–24 : Association for Computational Linguistics.

MEHROTRA R., SANNER S., BUNTINE W. & XIE L. (2013). Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, p. 889–892 : ACM.

MIHALCEA R. & TARAU P. (2004). Textrank : Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*.

MIKOLOV T., SUTSKEVER I., CHEN K., CORRADO G. S. & DEAN J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, p. 3111–3119.

NIKOLENTZOS G., MELADIANOS P., ROUSSEAU F., STAVRAKAS Y. & VAZIRGIANNIS M. (2017). Shortest-path graph kernels for document similarity. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, p. 1890–1900.

PAGLIARDINI M., GUPTA P. & JAGGI M. (2018). Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, volume 1, p. 528–540.

PARVEEN D. & STRUBE M. (2014). Multi-document summarization using bipartite graphs. In *Proceedings of TextGraphs-9 : the workshop on Graph-based Methods for Natural Language Processing*, p. 15–24.

PAUL C., RETTINGER A., MOGADALA A., KNOBLOCK C. A. & SZEKELY P. (2016). Efficient graph-based document similarity. In *European Semantic Web Conference*, p. 334–349 : Springer.

PEREIRA A. & ZIVIANI N. (2003). Syntactic similarity of web documents. In *Proceedings of the IEEE/LEOS 3rd International Conference on Numerical Simulation of Semiconductor Optoelectronic Devices (IEEE Cat. No. 03EX726)*, p. 194–200 : IEEE.

ROUSSEAU F. (2015). Ph.D. Dessertation, http ://frncsrss.github.io/papers/rousseau-dissertation.pdf.

ROUSSEAU F. & VAZIRGIANNIS M. (2015). Main core retention on graph-of-words for single-document keyword extraction. In *European Conference on Information Retrieval*, p. 382–393 : Springer.

SCHULTZ J. M. & LIBERMAN M. (1999). Topic detection and tracking using idf-weighted cosine coefficient. In *Proceedings of the DARPA broadcast news workshop*, p. 189–192 : San Francisco : Morgan Kaufmann.

SEIDMAN S. B. (1983). Network structure and minimum degree. *Social networks*, **5**(3), 269–287.

SONG Y. & ROTH D. (2015). Unsupervised sparse vector densification for short text similarity. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, p. 1275–1280.

TAPASWI M., ZHU Y., STIEFELHAGEN R., TORRALBA A., URTASUN R. & FIDLER S. (2016). Movieqa : Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, p. 4631–4640.

TRSTENJAK B., MIKAC S. & DONKO D. (2014). Knn with tf-idf based framework for text categorization. *Procedia Engineering*, **69**, 1356–1364.

WANG B., ZHU J., PIERSON E., RAMAZZOTTI D. & BATZOGLOU S. (2017). Visualization and analysis of single-cell rna-seq data by kernel-based similarity learning. *Nature methods*, **14**(4), 414.

WANG C., SONG Y., LI H., ZHANG M. & HAN J. (2015a). Knowsim : A document similarity measure on structured heterogeneous information networks. In *2015 IEEE International Conference on Data Mining*, p. 1015–1020 : IEEE.

WANG P., XU J., XU B., LIU C., ZHANG H., WANG F. & HAO H. (2015b). Semantic clustering and convolutional neural network for short text categorization. In *Proceedings of the 53rd Annual*

Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers), volume 2, p. 352–357.

WIETING J., BANSAL M., GIMPEL K. & LIVESCU K. (2016). Charagram : Embedding words and sentences via character n-grams. *arXiv preprint arXiv :1607.02789*.

ZHANG W., YOSHIDA T. & TANG X. (2011). A comparative study of tf* idf, LSI and multi-words for text classification. *Expert Systems with Applications*, **38**(3), 2758–2765.

ZHU Z., LI M., CHEN L. & YANG Z. (2013). Building comparable corpora based on bilingual lda model. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, volume 2, p. 278–282.