On a Novel Representation of Multiple Textual Documents in a Single Graph

Nikolaos Giarelis, Nikos Kanakaris, and Nikos Karacapilidis

Abstract This paper introduces a novel approach to represent multiple documents as a single graph, namely, the *graph-of-docs* model, together with an associated novel algorithm for text categorization. The proposed approach enables the investigation of the importance of a term into a whole corpus of documents and supports the inclusion of relationship edges between documents, thus enabling the calculation of important metrics as far as documents are concerned. Compared to well-tried existing solutions, our initial experimentations demonstrate a significant improvement of the accuracy of the text categorization process. For the experimentations reported in this paper, we used a well-known dataset containing about 19,000 documents organized in various subjects.

1 Introduction

In recent years, we have witnessed an increase in the adoption of graph-based approaches for the representation of textual documents [3, 26]. Generally speaking, graph-based text representations exploit properties inherited from graph theory (e.g., node centrality and subgraph frequency) to overcome the limitations of the classical *bag-of-words* representation [1]. Specifically, graph-based models (contrary to the bag-of-words ones) are able to (i) capture structural and semantic information of a text, (ii) mitigate the effects of the "curse-of-dimensionality" phenomenon, (iii) identify the most important terms of a text, and (iv) seamlessly incorporate information coming from external knowledge sources.

N. Giarelis · N. Kanakaris · N. Karacapilidis (✉)
Industrial Management and Information Systems Lab, MEAD, University of Patras, 26504 Rio Patras, Greece
e-mail: nkanakaris@upnet.gr

N. Giarelis
e-mail: giarelis@ceid.upatras.gr

N. Kanakaris
e-mail: karacap@upatras.gr

I. Czarnowski et al. (eds.), *Intelligent Decision Technologies*, Smart Innovation, Systems and Technologies 193, https://doi.org/10.1007/978-981-15-5925-9_9

However, in cases where a corpus of documents needs to be considered and analyzed, existing graph-based approaches represent each document of the corpus as a single graph. In such cases, the main weaknesses of these approaches are that (i) they are incapable of assessing the importance of a word for the whole set of documents and (ii) they do not allow for representing similarities between these documents.

To remedy the above weaknesses, this paper expands the graph-based text representation model proposed by Rousseau et al. [21, 22], i.e., the *graph-of-words* model, and introduces a novel approach to represent multiple documents as a single graph, namely, the *graph-of-docs* model, as well as an associated novel algorithm for text categorization. Contrary to existing approaches, the one introduced in this paper (i) enables the investigation of the importance of a term into a whole corpus of documents, (ii) masks the overall complexity by reducing each graph of words to a "document" node, and (iii) supports the inclusion of relationship edges between documents, thus enabling the calculation of important metrics as far as documents are concerned. The proposed approach uses the Neo4j graph database (https://neo4j.com) for the representation of the graph-of-docs model. For the implementation of our experiments, we use the Python programming language and the scikit-learn ML library (https://scikit-learn.org). Compared to well-tried existing solutions, our initial experimental results show a significant improvement of the accuracy of the text categorization process.

The remainder of the paper is organized as follows. Section 2 describes the graph-of-words representation and its application to classical NLP tasks. Our approach, i.e., graph of docs, is analytically presented in Sect. 3. Section 4 reports on the experiments carried out to evaluate the proposed approach. Finally, limitations of our approach, future work directions, and concluding remarks are outlined in Sect. 5.

2 Background Work

2.1 Graph of Words

The graph-of-words textual representation is similar to the bag-of-words representation that is widely used in the NLP field. It enables a more sophisticated keyword extraction and feature engineering process. In a graph of words, each node represents a unique term (i.e., word) of a document and each edge represents the co-occurrence between two terms within a sliding window of text. Nikolentzos et al. [16] propose the utilization of a small sliding window size, due to the fact that the larger ones produce heavily interconnected graphs where the valuable information is cluttered with noise; Rousseau et al. [21] suggest that a window size of four is generally considered to be the appropriate value, since it does not sacrifice either the performance or the accuracy of their approach.

2.2 Graph-Based Keyword Extraction

A set of existing approaches performing classical NLP tasks builds on the graph-of-words textual representation. Ohsawa et al. [18] were the first that use the graph-of-words representation in the keyword extraction and text summarization tasks. Their approach segments a graph of words into clusters aiming to identify frequent co-occurred terms. Adopting a similar research direction, the TextRank model implements a graph-based ranking measure to find the most prestigious nodes of a graph (i.e., the nodes with the highest indegree value) and utilizes them to the tasks of keyword and sentence extraction [12].

The utilization of node centrality measures to the keyword and key-phrase extraction tasks can also be found in the literature [4]; these measures include the "degree" centrality, the "closeness" centrality, the "betweenness" centrality, and the "eigenvector" centrality. Bougouin et al. [5] propose a novel graph-based unsupervised topic extraction method, namely, TopicRank. TopicRank clusters key phrases into topics and identifies the most representative ones using a graph-based ranking measure (e.g., TextRank). Finally, Tixier et al. [27] focus on the task of unsupervised single-document keyword extraction, arguing that the most important keywords correspond to the nodes of the *k-core* subgraph [24].

2.3 Graph-Based Text Categorization

As far as graph-based text categorization is concerned, several interesting approaches have been already proposed in the literature. Depending on their methodology, we can classify them into two basic categories: (i) approaches that employ frequent subgraph mining techniques for feature extraction and (ii) approaches that rely on graph kernels. Popular frequent subgraph mining techniques include *gSpan* [29], *Gaston* [15], and *gBoost* [23]. Rousseau et al. [21] propose various combinations and configurations of these techniques, ranging from unsupervised feature mining using gSpan to unsupervised feature selection exploiting the k-core subgraph. In particular, aiming to increase performance, they rely on the concept of k-core subgraph to reduce the graph representation to its densest part. The experimental results show a significant increment of the accuracy compared to common classification approaches.

Graph kernel algorithms contribute significantly to recent approaches for graph-based text categorization [17]. A graph kernel is a measure that calculates the similarity between two graphs. For instance, a document similarity algorithm based on shortest path graph kernels has been proposed in [17]; this algorithm can be used as a distance metric for common ML classifiers such as SVM and k-NN. The experimental results show that classifiers that are based on graph kernel algorithms outperform several classical approaches. It is noted that the *GraKeL* Python library collects and unifies widely used graph kernel libraries into a single framework [25], providing an

easily understandable interface (similar to that of *scikit-learn*) that enables the user to develop new graph kernels.

2.4 Graph Databases

Compared to conventional relational databases, graph databases provide a more convenient and efficient way to natively represent and store highly interlinked data. In addition, they allow the retrieval of multiple relationships and entities with a single operation, avoiding the utilization of rigid joint operations which are heavily used in relational databases [13]. An in-depth review of graph databases appears in [20]. Our approach builds on top of the *Neo4j* graph database (https://neo4j.com), a broadly adopted graph database system that uses the highly expressive *Cypher* Graph Query Language to query and manage data.

3 Our Approach: Graph of Docs

In this paper, we expand the "graph-of-words" model proposed by Rousseau et al. [22] to introduce a "graph-of-docs" model. Contrary to the former model, where a graph corresponds to a single document, the proposed model represents multiple documents in a single graph. Our approach allows diverse types of nodes and edges to co-exist in a graph, ranging from types of nodes such as "`document`" and "`word`" to types of edges such as "`is_similar`," "`connects`," and "`includes`" (see Fig. 1). This enables us to investigate the importance of a term not only within a single document but also within a whole corpus of documents. Furthermore, the proposed graph-of-docs representation adds an abstraction layer by assigning each graph of words to a document node. Finally, it supports relationship edges between documents, thus enabling the calculation of important metrics as far as the documents are

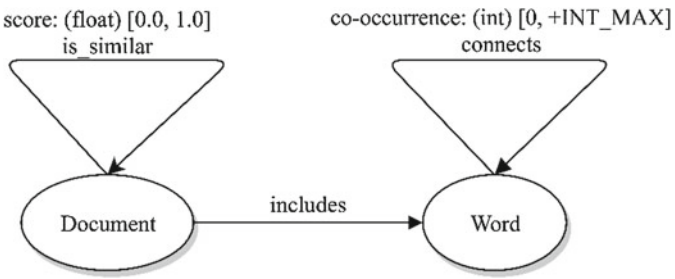

Fig. 1 The schema of the graph-of-docs representation model

concerned (e.g., identifying cliques or neighborhoods of similar documents, identifying important documents, generating communities of documents without any prior knowledge, etc.).

The graph-of-docs representation produces a directed dense graph that contains all the connections between the documents and the words of a corpus (see Fig. 2). Each unique word node is connected to all the document nodes where it belongs to using edges of the "`includes`" type; edges of "`connects`" type are only applicable between two word nodes and denote their co-occurrence within a specific sliding text window; finally, edges of the "`is_similar`" type link a pair of document nodes and indicate their contextual similarity.

The above transformation of a set of documents into a graph assists in the reduction of diverse NLP problems to problems that have been well studied through graph theory techniques [21]. Such techniques explore important characteristics of a graph, such as node centrality and frequent subgraphs, which in turn are applied to identify meaningful keywords and find similar documents.

We argue that the accuracy of common NLP and text mining tasks can be improved by adopting the proposed graph-of-docs representation. Below, we describe how three

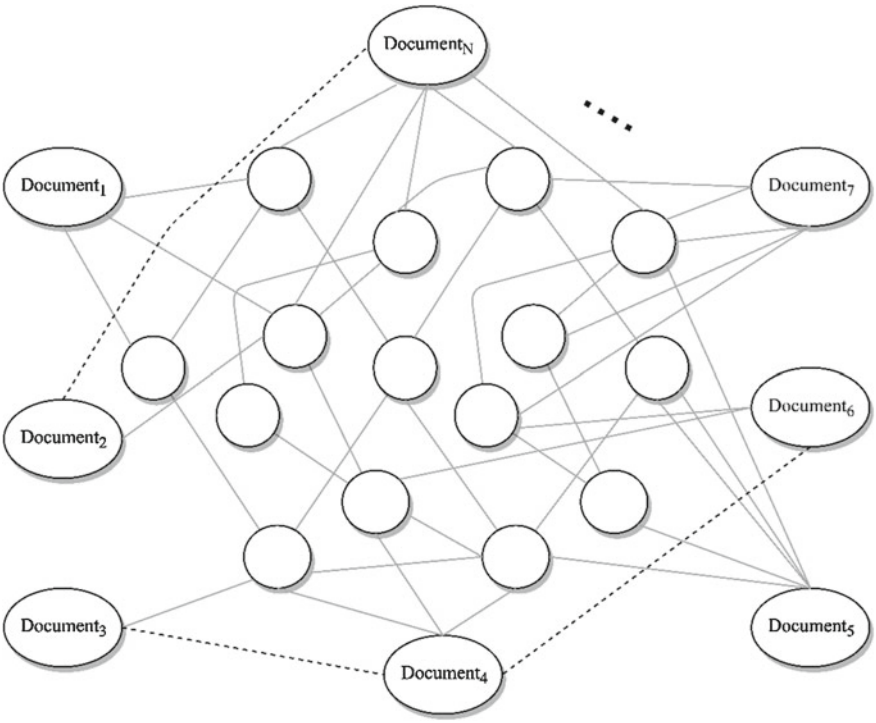

Fig. 2 The graph-of-docs representation model (relationships between documents are denoted with dotted lines)

key NLP tasks (namely, "Keyword Extraction," "Document Similarity," and "Text Categorization") can be carried out using our approach.

3.1 Keyword Extraction

To extract the most representative keywords from each document, we apply centrality measures (an in-depth review of them appears in [10]). In general, these measures identify the most influential nodes of a graph, i.e., those that usually have an indegree score higher than a predefined threshold. The main idea is that the words that correspond to the top-N ranked nodes can be considered as semantically more important than others in a specific document. Recent algorithms to calculate the centrality of a graph include *PageRank, ArticleRank, Betweenness Centrality, Closeness Centrality, Degree Centrality,* and *Harmonic Centrality*. While also utilizing the above algorithms to calculate centrality measures, our approach differs from the existing ones in that it considers the whole corpus of documents instead of each document separately; hence, we are able to detect a holistic perspective of the importance of each term.

3.2 Document Similarity Subgraph

Typically, graph of words derived from similar documents share common word nodes as well as similar structural characteristics. This enables us to calculate the similarity between two documents either by using typical data mining similarity measures (e.g., the *Jaccard* or the *cosine similarity*), or by employing frequent subgraph mining techniques (see Sect. 2.3). In our approach, we produce a similarity subgraph, which consists of document nodes and edges of "`is_similar`" type (we aim to extend the set of supported edge types in the future). It is clear that the creation of such a subgraph is not feasible in approaches that represent each document as a single graph.

3.3 Text Categorization

By exploiting the aforementioned document similarity subgraph, we detect communities of contextually similar documents using the "`score`" property of the "`is_similar`" type edges as a distance value. A plethora of community detection algorithms can be found in the literature, including *Louvain* [11], *Label Propagation* [19], and *Weakly Connected Components* [14]; an in-depth review of them can be found in [6, 30].

Since the documents that belong to the same graph community are similar, as far as their context is concerned, we assume that it is more likely to also share the same class when it comes to performing a text categorization task. Therefore, we can easily decide the class of a document either by using the most frequent class in its community of documents or by running a nearest neighbor algorithm (such as the k-nearest neighbors) using as input the documents of its community.

4 Experiments

4.1 Dataset

We have tested the proposed model by utilizing an already preprocessed version of the well-known *20 Newsgroups* dataset, and specifically the version containing 18,828 documents organized in various subjects (this dataset can be retrieved from http://qwone.com/~jason/20Newsgroups/20news-18828.tar.gz). This version does not contain unnecessary headers or duplicate texts that would require additional work as far as data cleansing is concerned. It is noted that this dataset is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. It has become a popular dataset for experiments in text applications of ML techniques, such as text classification and text clustering. We claim that this dataset fits well to the purposes of our experimentations (i.e., multi-class classification), given the large volume of different documents on the same subjects.

4.2 Implementation

The Neo4j graph database has been utilized for the representation of the proposed graph-of-docs model. Furthermore, we used the Python programming language and the scikit-learn ML library for the implementation of our experiments. The full code and documentation of our approach is freely available at https://github.com/NC0DER/GraphOfDocs.

Our approach consists of four major steps that are described in the sequel. Firstly, we execute a preprocessing function that (i) removes stopwords and punctuation marks from the texts and (ii) produces a list of terms for each document. Secondly, we execute a function that creates a graph of words by using the aforementioned terms. More specifically, this function creates unique word nodes from the list of terms and then links them (if needed), while also calculating the co-occurrence score. In this step, the graph of docs is being created through the progressive synthesis of the graphs of words produced for each document. It is noted that by loading the *20 Newsgroups* dataset, we generated a graph of docs with 174,347 unique nodes and 4,934,175 unique edges. Thirdly, we execute the PageRank algorithm aiming to

identify the most important word nodes in the entire graph. Finally, we implement a function that calculates the Jaccard similarity measure for all document nodes; this function builds the document similarity subgraph and forms communities of similar documents using the Louvain algorithm. Our implementation is sketched in the following pseudocode:

```
function graph_of_docs():
  database = connect_to_the_database()
  dataset[] = read_dataset()
  for document_label, document in dataset:
    document = clean_data(document)
    terms[] = generate_terms(document)
    create_graph_of_words(terms, document_label, database)
  run_word_centrality_measure_algorithm('Pagerank', database)
  create_document_similarity_subgraph('Jaccard', database)
  form_communities_of_similar_documents('Louvain', database)
  conduct_classification_experiments()
  disconnect_from_the_database()
```

4.3 Evaluation

Aiming to evaluate the performance of our approach, we benchmark the accuracy score of the text classifier described in Sect. 3.3 against those of common domain-agnostic classifiers that use the bag-of-words model for their text representation (see Table 1). Considering the accuracy of each text classifier, we conclude that the proposed graph-of-docs representation significantly increases the accuracy of text classifiers (accuracy: 97.5%).

Table 1 Accuracy scores for the existing and the proposed text classifiers

Text classifier	Accuracy (%)
5-NN	54.8
2-NN	61.0
1-NN	76.0
Naive Bayes	93.7
Logistic regression	93.9
Neural network (100 × 50)	95.5
Neural network (1000 × 500)	95.9
Graph-of-docs classifier	**97.5**

5 Conclusions

In this paper, we introduced a novel approach for representing multiple textual documents in a single graph, namely, "graph of docs." To test our approach, we benchmarked the proposed "graph-of-docs"-based classifier against classical text classifiers that use the "bag-of-words" model for text representation. The evaluation outcome was very promising; an accuracy score of 97.5% was achieved, while the second best result was 95.9%. However, our approach has a set of limitations, in that (i) it does not perform equally well with outlier documents (i.e., documents that are not similar to any other document) and (ii) it has performance issues since the generation of a graph of documents requires significant time in a disk-based graph database such as Neo4j [28].

Aiming to address the above limitations as well as to integrate our approach into existing works on knowledge management systems, future work directions include: (i) the experimentation with alternative centrality measures, as well as diverse community detection and graph partitioning algorithms [2]; (ii) the utilization and assessment of an in-memory graph database in combination with Neo4j; (iii) the enrichment of the existing textual corpus through the exploitation of external domain-agnostic knowledge graphs (e.g., DBPedia and Wikipedia knowledge); and (iv) the integration of our approach into collaborative environments where the underlying knowledge is structured through semantically rich discourse graphs (e.g., integration with the approaches described in [7–9]).

Acknowledgements The work presented in this paper is supported by the OpenBio-C project (www.openbio.eu), which is co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH—CREATE—INNOVATE (Project id: T1EDK-05275).

References

1. Aggarwal, C.C.: Machine Learning for Text. Springer (2018)
2. Armenatzoglou, N., Pham, H., Ntranos, V., Papadias, D., Shahabi, C.: Real-time multi-criteria social graph partitioning: a game theoretic approach. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1617–1628, ACM Press (2015)
3. Blanco, R., Lioma, C.: Graph-based term weighting for information retrieval. Inf. Retr. **15**(1), 54–92 (2012)
4. Boudin, F.: A comparison of centrality measures for graph-based keyphrase extraction. In: Proceedings of the Sixth International Joint Conference on Natural Language Processing, pp. 834–838 (2013)
5. Bougouin, A., Boudin, F., Daille, B.: Topicrank: graph-based topic ranking for keyphrase extraction. In: Proceedings of the Sixth International Joint Conference on Natural Language Processing, pp. 543–551 (2013)
6. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)

7. Kanterakis, A., Iatraki, G., Pityanou, K., Koumakis, L., Kanakaris, N., Karacapilidis, N., Potamias, G.: Towards reproducible bioinformatics: the OpenBio-C scientific workflow environment. In: Proceedings of the 19th IEEE International Conference on Bioinformatics and Bioengineering (BIBE), pp. 221–226, Athens, Greece (2019)
8. Karacapilidis, N., Papadias, D., Gordon, T., Voss, H.: Collaborative environmental planning with GeoMed. Eur. J. Oper. Res. Spec. Issue Environ. Plan. **102**(2), 335–346 (1997)
9. Karacapilidis, N., Tzagarakis, M., Karousos, N., Gkotsis, G., Kallistros, V., Christodoulou, S., Mettouris, C., Nousia, D.: Tackling cognitively-complex collaboration with CoPe_it! Int. J. Web-Based Learn Teach. Technol **4**(3), 22–38 (2009)
10. Landherr, A., Friedl, B., Heidemann, J.: A critical review of centrality measures in social networks. Bus Inf. Syst. Eng. **2**(6), 371–385 (2010)
11. Lu, H., Halappanavar, M., Kalyanaraman, A.: Parallel heuristics for scalable community detection. Parallel Comput. **47**, 19–37 (2015)
12. Mihalcea, R., Tarau, P.: Textrank: bringing order into text. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 404–411 (2004)
13. Miller, J.J.: Graph database applications and concepts with Neo4j. In: Proceedings of the Southern Association for Information Systems Conference, vol. 2324, no. 36, Atlanta, GA, USA (2013)
14. Monge, A., Elkan, C.: An efficient domain-independent algorithm for detecting approximately duplicate database records (1997)
15. Nijssen, S., Kok, J. N.: A quickstart in frequent structure mining can make a difference. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 647–652, ACM Press (2004)
16. Nikolentzos, G., Meladianos, P., Rousseau, F., Stavrakas, Y., Vazirgiannis, M.: Shortest-path graph kernels for document similarity. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1890–1900 (2017)
17. Nikolentzos, G., Siglidis, G., Vazirgiannis, M.: Graph Kernels: a survey. arXiv preprint arXiv:1904.12218 (2019)
18. Ohsawa, Y., Benson, N. E., Yachida, M.: KeyGraph: automatic indexing by co-occurrence graph based on building construction metaphor. In: Proceedings IEEE International Forum on Research and Technology Advances in Digital Libraries, pp. 12–18, IEEE Press (1998)
19. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76**(3) (2007)
20. Rawat, D.S., Kashyap, N.K.: Graph database: a complete GDBMS survey. Int. J. **3**, 217–226 (2017)
21. Rousseau, F., Kiagias, E., Vazirgiannis, M.: Text categorization as a graph classification problem. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 1, pp. 1702–1712 (2015)
22. Rousseau, F., Vazirgiannis, M.: Graph-of-word and TW-IDF: new approach to ad hoc IR. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, pp. 59–68, ACM (2013)
23. Saigo, H., Nowozin, S., Kadowaki, T., Kudo, T., Tsuda, K.: gBoost: a mathematical programming approach to graph classification and regression. Mach. Learn. **75**(1), 69–89 (2009)
24. Seidman, S.B.: Network structure and minimum degree. Soc. Netw. **5**(3), 269–287 (1983)
25. Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., Vazirgianis, M.: Grakel: a graph kernel library in python. arXiv preprint arXiv:1806.02193 (2018)
26. Sonawane, S.S., Kulkarni, P.A.: Graph based representation and analysis of text document: a survey of techniques. Int. J. Comput. Appl. **96**(19) (2014)
27. Tixier, A., Malliaros, F., Vazirgiannis, M.: A graph degeneracy-based approach to keyword extraction. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1860–1870 (2016)

28. Wang, W., Wang, C., Zhu, Y., Shi, B., Pei, J., Yan, X., Han, J.: Graphminer: a structural pattern-mining system for large disk-based graph databases and its applications. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 879–881. ACM Press (2005)
29. Yan, X., Han, J.: gspan: Graph-based substructure pattern mining. In: Proceedings of the IEEE International Conference on Data Mining, pp. 721–724. IEEE Press (2002)
30. Yang, Z., Algesheimer, R., Tessone, C.J.: A comparative analysis of community detection algorithms on artificial networks. Sci. Rep. **6**, 30750. https://doi.org/10.1038/srep30750 (2016)