A Graph-Based Extension for the Set-Based Model Implementing Algorithms Based on Important Nodes

Nikitas-Rigas Kalogeropoulos, Ioannis Doukas, Christos Makris, and Andreas Kanavos(✉)

Computer Engineering and Informatics Department, University of Patras, Patras, Greece
{kalogeropo,doukas,makri,kanavos}@ceid.upatras.gr

Abstract. The purpose of this paper is the expansion of the set-based model, namely an information retrieval model, with the use of graphs. The indexing process implements a graphical representation, while the querying and document representation are based on the classical set-based model. The root of the set-based model corresponds to the use of term sets to complete the querying process based on the terms of the query. However, in the weighting process, this paper presents a wholly different approach elaborating on algorithms that may clearly benefit the process based on the k-core decomposition of each single graph. The main focus will finally be the estimation and presentation of the most important nodes belonging to each graph. These intend to be regarded as keywords presenting the evaluation of their major influence.

Keywords: Information retrieval · Retrieval models · Set-based model · Important nodes · K-core decomposition · Graph degeneracy

1 Introduction

In Information Retrieval (IR), two of the main research subjects involve the creation of an effective and efficient retrieval model as well as the detection of keywords in textual data. In this study, following the work presented in [16], we make an effort to combine those subjects, thus creating a new model by implementing a graphical representation of texts using keywords as important nodes, which are boosted at the weighting process. The case of the study is focused on whether the keywords can actually improve the retrieval process or not. The main difference among the proposed methods lies in the algorithms of discovering important nodes.

The set-based model is responsible for the retrieval process whereas, it is considered a combination of a set-theoretic and algebraic model. The algebraic notion of the model is expressed by the similarity function, which resembles the vector space model. More to the point, given a query, the model creates clustered

Published by Springer Nature Switzerland AG 2020
I. Maglogiannis et al. (Eds.): AIAI 2020 Workshops, IFIP AICT 585, pp. 143–154, 2020.
https://doi.org/10.1007/978-3-030-49190-1_13

sets of terms by utilizing the query terms. The termsets are then formed as the apriori algorithm proposes [1].

A well-structured graphical IR model is a rare subject of study. The main onus of the study is aimed at demonstrating either the graphical representation or the scoring functions. However, authors in [10] proposed a complete graphical model; their implementation consists of unweighted directed graphs for each text of the collection and thus, the flow of the text is captured. The nodes of the graph represent the unique terms of the text while the edges refer to the relationship of the corresponding nodes within a partition of the text, expressed by a fixed-sized sliding window. As scoring function for the model, their implementation was based on the TF-IDF and BM25 using successive normalizations. One main difference regarding their scoring function, was the hypothesis that the weight of a vertex (TW), which corresponds to a unique text term, contains more information than the term frequency. Therefore, their function can be summarized as $TW - IDF$, challenging the term independence assumption.

To further consolidate our point, we should mention that the above graph-of-words model was utilized in [16] on an attempt to estimate the keywords of a given text using algorithms based on graph degeneracy [14]. There, a method known as k-core decomposition, which creates subgraph having specific structural properties, was proposed in order to study the cohesion on social networks. The k-core decomposition of a graph G is considered as a maximal connected subgraph consisting of nodes with a degree at least equal to k. Following the above definition, a subgraph has core equal to k only when the entity of its nodes has a degree of k or more. The k-core, where k is maximal, is often called MainCore; it is used as the starting point of the keyword finding study, which afterwards is expanded on the remaining levels of decomposition. At lower levels, the cores tend to be large, and the core cohesion rises at higher levels of decomposition. A well known algorithm for the purpose of finding the cores of a graph at $O(m)$ complexity was optly introduced in [2].

The decomposition aspect of defining the important nodes of a graph can be considered as a novelty. Centrality measures such as degree, closeness, and betweenness centralities are constituted as the most common alternatives [3,4,9]. More specifically, the degree centrality is based on the node degree, thus it can not necessarily reflect the magnitude of its importance. On the other hand, the closeness and betweenness centralities depend on the calculation of geodesic paths, which are difficult to be computed in complex graphs, especially on complete ones. Authors in [6] have focused on keyword extraction using graphical representation on texts as they elaborated on supervised methods using node centrality as well as unsupervised methods exploiting the HITS algorithm [5].

The information retrieval models consist of set theoretic, algebraic and probabilistic models, where the main emphasis is given on the set-based model [7,8]. A model, based on set theory, implements algebraic notions on scoring functions and document-queries representation, which is derived from the vector space model [12]. The set-based model considers sets of terms; these terms in turn consist of a given query, known as termsets. The sets are mined using the apriori

algorithm [1] that is responsible for supporting a minimum frequency boundary for each termset. There are however some cases where the set-based model uses a proximity measure to assure that terms are close enough, so as to have a rational relationship.

As mentioned above, the set-based model represents queries and documents in a similar concept as the vector space model. The same applies to the scoring function, which calculates the similarity between a given query and the collection documents. The gist of our proposal is based on the logical assumption that every text consists of words that carry great importance, known as keywords. Therefore, the corresponding text can be graphically represented by important nodes. If we can initially detect them and, in following amplify their importance, expressed by a boost on their weight, the retrieval process will be augmented.

2 Graphical Extension of Set-Based Model Using Important Nodes

Each document is represented as an undirected weighted complete graph. When node importance is considered, then this complete graph is in need of further processing aiming at edge pruning; this procedure is essential due to the fact that the core decomposition has only one level and therefore the whole graph is considered as important [14]. In following, a collection sized graph is created by the union of the respective text graphs. That is the point where, depending on the method, the important nodes are boosted by a value equal to their "importance". Finally, with the use of graph theory, a weight is produced for each term, which is considered at the retrieval process by the set-based model.

2.1 Rational Path Graph

Each document graph is created with the assumption that every term is related likewise to the other terms, as this particular word has been included in the text. Every term of a given document is represented by a node in the graph and the relationship between two nodes as a weighted edge with weight W_{out}. Moreover, each node has a self loop edge, that is an edge with an identical starting and ending node having weight W_{in}. The graph construction process is simplified by taking into consideration the two following theorems.

Theorem 1. *Given a node N_i in document D_j with term frequency of the corresponding term equal to tf_i, the in-weight of the self edge (N_i, N_i) is computed as*

$$W_{in} = \frac{tf_i \times (tf_i + 1)}{2} \tag{1}$$

Theorem 2. *Given the nodes N_i, N_j in document D_j with term frequency of the corresponding terms equal to tf_i and tf_j respectively, the out-weight of the edge (N_i, N_j) is computed as*

$$W_{out} = tf_i \times tf_j, \quad \text{WHEN} \quad N_i \neq N_j \tag{2}$$

With the use of these two theorems, we propose an alternative algorithm of the rational path graph that can be cost efficient as the time complexity is $O(n^2) + O(m^2)$, where n is the number of the text terms and m the number of unique terms encountered in the text, as presented in Algorithm 1. We have to take into account the fact that this algorithm depends also on the way the documents are stored and consists of two steps; the term counting process as well as the graph creation. If the counting process is eliminated by the pre-processing of the text then the time complexity is $O(m^2)$.

Algorithm 1. Proposed Graph Construction Algorithm

input Document D_j
output Graph G of Document D_j
Given array A, list L, term t, node n, neighbour ng
Insert each t in L
for all $t \in L$ **do**
  if $t \notin A$ **then**
    $tf = L.count(term)$
    $A = (term, tf)$
  end if
end for
for $t \in A$ **do**
  $n = A[0]$
  $tf_t = A[1]$
  if $n \notin G$ **then**
    Add n to G with $W_{in} = \frac{tf_t \times (tf_t + 1)}{2}$
  end if
  for $ng \in A$ **do**
    if $edge(n, ng[0]) \notin G$ **then**
      Initialize $edge(n, ng[0])$ with $W_{out}(n, ng[0]) = tf_t \times ng[1]$
    end if
  end for
end for

Initially, the algorithm creates a tuple of unique term and its respective term frequency by counting each and every individual word in the concrete text. In following, for each unique term, a node is created and the in-edge as well as the out-edge with every neighbour is initialized, if such an edge does not exist. Finally, the respective edge weight using the aforementioned theorems is calculated.

2.2 Edge Pruning

The proposed methods depend on the degree of each node. Concretely, due to the fact that the graph is complete, the degree of each node is equal to $N - 1$,

where N stands for the number of nodes that the graph consists of. The in-edge is not taken into consideration and therefore, there is a need to delete edges with minimum information loss. At first glance, someone would propose methods related to the connectivity of the graph [17]. However, the completeness of the graphs does not allow to calculate paths between nodes in a realistic manner. The computational complexity of such an act is $O(n!)$ [13]. Thus, a more naive approach was decided to be used, where initially the average weight of all graph edges are calculated and in following, all edges, with less weight than the average edge multiplied by a percentage P, are removed. P value is experimentally defined and is different in each collection. After this process, the important node estimation can be implemented on each graph separately.

2.3 Important Nodes Detection

The proposed methods focus on discovering the important nodes of a graph while simultaneously based on this graph's k-core decomposition. The decomposition of a graph is used for detecting influential nodes, where, in their absence, the graph might lose its connectivity. Authors in [14] suggested that given a graph $G(V, E)$, a subgraph $S(V', E')$ is considered as the k-core decomposition of G if the degree of each node in V' is less than k, considering that $V' \supseteq V$ and $E' \supseteq E$. For example, the following Fig. 1 presents the decomposition of a random graph G; the respective cores start with light grey nodes moving on to darker and finally to red ones.

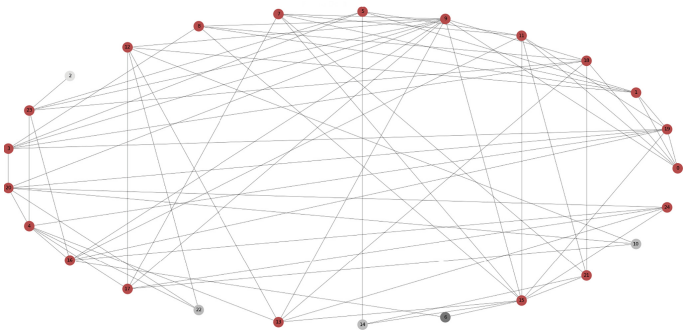


Fig. 1. K-core decomposition of a graph

It is important to note that the k core, where the $k+1$ level of decomposition does not exist, is called main core.

2.3.1 Main Core

Initially, we consider that the main core includes all the key nodes of the graph similarly to [11]. The edges regarding those nodes have their weight amplified in the union graph.

2.3.2 Density Method

Following the logical process of [16], i.e. the assumption of existence of all key nodes in the main core is too simplistic, we attempt to estimate the density of each core level of the text graph and consider as key nodes the nodes of the most dense subgraph. The densest subgraph was chosen by using the elbow method and the density of a subgraph $S(V', E')$ is computed as

$$density(S) = \frac{|E'|}{|V'| \times (|V'| - 1)} \tag{3}$$

2.3.3 CoreRank Method

Finally, the scoring method [16] is implemented into our model. CoreRank attempts to sort the nodes by the sum of the core number in which every neighbour exists. Thereafter, we assume that the important nodes should be chosen on document size basis and thus, the first 30% of the sorted node list are considered as important.

2.4 Union Graph and Graphs Weights

Ultimately, a collection size graph is created as the union of each text graph. Its nodes represent all the terms of the collection and the respective edges consist of the edges of each graph. The union graph edge weight is calculated as the sum of the weight in each graph, in which it exists.

The proposed union graph algorithm updates the current union graph U with the graph G as presented in Algorithm 2. Initially, the union graph U is an empty graph, while the proposed algorithm checks for each term whether it constitutes an important node and determines the value of importance h. At this point, in case the model does not consider the node importance, then the important node list will be empty and therefore the value h will be equal to 1. In following, it checks if the term exists or not in the union graph. In case the condition is true, the W_{in} weight of the term node in union graph is updated by the sum of the respective W_{in} in graph multiplied by h (line 11). Subsequently, the update of the value of the out-edge weight for each node of the union graph takes place (lines 13–14). In the absence of an edge between the term and the union graph node, one is initialized, whereas if the term does not exist in the union graph, the node is initialized and the update process begins.

When the union graph construction process is completed, a new weight NW for each node k emerges, derived from the union graph.

$$NW_k = \log\left(1 + a \times \frac{W_{out_k}}{(W_{in_k} + 1) \times (ng_k + 1)}\right) \times \log\left(1 + b \times \frac{1}{ng_k + 1}\right) \tag{4}$$

where W_{out_k} is the sum of out-edges and W_{in_k} stands for the weight sum of all in-edges terms for a node k. The number of neighbours that corresponds to that particular node is expressed as ng_k. Finally, a and b correspond to the gravity of the under review model against the set-based model.

Algorithm 2. Proposed Union Graph Construction Algorithm

1: **input** Graph G, Union Graph U
2: **output** Updated Union Graph including Graph UG
3: Given term t, node n, k-core list kc, value of importance h
4: **for all** $t \in G$ **do**
5: **if** $t \in kc$ **then**
6: Gain value of importance
7: **else**
8: $h = 1$
9: **end if**
10: **if** $t \in U$ **then**
11: $W_{in}(t, U) += W_{in}(t, G) \times h$
12: **for all** $n \in G$ where $n \neq t$ **do**
13: **if** $edge(n, t) \in U$ **then**
14: $W_{out}(n, t, U) += W_{out}(n, t, G) \times h$
15: **else**
16: Initialize $edge(n, t) \in U$ with $W_{out}(n, t, U) = W_{out}(n, t, G) \times h$
17: **end if**
18: **end for**
19: **else**
20: Add $t \in U$ with $W_{in}(n, t, U) = W_{in}(n, t, G) \times h$
21: Repeat lines 12 to 18 for this case
22: **end if**
23: **end for**

3 Documents and Query Representation

The retrieval process is handled by the set-based model. As mentioned, the introduced termset notion contributes to the reduction of computational complexity due to the fact that the volume of processed data is significantly lower [8]. Termsets are created with the use of association rule mining algorithms [1] and the gist of the process is that we combine two termsets, different in only one element of the set, in order to produce a new termset expanded by one new term. The sets should have elements greater than a lower frequency bound, thus decreasing the complexity even more.

3.1 Weighting Process

At this point, the need of combining the derived graph weight with the set-based model's weight appears. For each termset S_i, we calculate a new measure TN_{S_i} as the product of the graph weights of its terms.

$$TN_{S_i} = \prod_{k \in S_i} NW_k \tag{5}$$

Thereafter, we include the new measure at the termset weight formula of a termset S_i in document D_j, where Sf_{ij} represents the termset frequency *(TF)* and $\frac{N}{dS_i}$ the inverse document frequency *(IDF)* of the corresponding termset

$$W_{S_{ij}} = (1 + \log Sf_{ij}) \times \log\left(1 + \frac{N}{dS_i}\right) \times tnw_{S_i} \tag{6}$$

The query scoring function is more simplistic as it is expressed by the inverse document frequency of the termset S_i

$$W_{S_{iq}} = \log\left(1 + \frac{N}{dS_i}\right) \tag{7}$$

3.2 Document Representation and Similarity

Finally, the document and query are represented as vectors in a similar manner with the vector space model. However, the vector space model uses TF/IDF as weight, instead of the above mentioned termset weighting process. Each vector d_j is associated with a text document and contains the weights of every termset in that document. Because the termset is based on the query terms, it is obvious that the size of the vector would be no more than 2^n, where n is the number of words that the query contains.

$$\begin{aligned} \overrightarrow{d_j} &= (W_{S_{1j}}, W_{S_{2j}}, \cdots, W_{S_{2^{n_j}}}) \\ \overrightarrow{Q} &= (W_{S_{1q}}, W_{S_{2q}}, \cdots, W_{S_{2^{n_q}}}) \end{aligned} \tag{8}$$

As the set-based model dictates, the similarity between a document of the collection and the query is calculated using the cosine similarity. The result of this process is used for the ranking function of each model.

$$sim(Q, d_j) = \frac{\overrightarrow{d_j} \times \overrightarrow{Q}}{\|\overrightarrow{d_j}\| \times \|\overrightarrow{Q}\|} \tag{9}$$

The issue at hand in this case is that the calculation of the $\|\overrightarrow{d_j}\|$ is extremely difficult due to the high number of frequent termsets generated by the model, aggravated especially in large documents. Therefore, following the approach of the original set-based model, we should only consider the sets, which contain only one element as an estimation of the above mentioned norm; however, in our case, due to the size of the collection, we were able to compute the norm in a reasonable time.

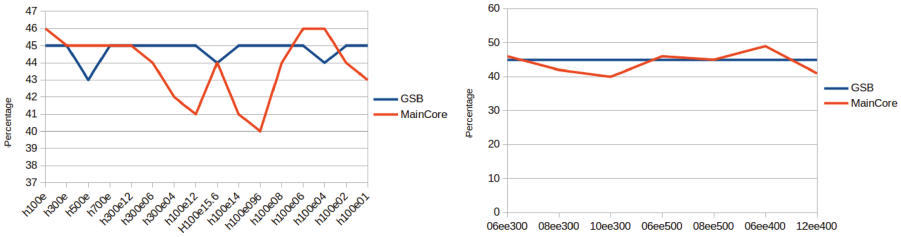
4 Results and Discussion

The Cystic Fibrosis (CF) Database [15] was used for the experimental evaluation, where all the included queries and documents were utilized. The number of

queries and documents was 100 and 1238, respectively. We have initially examined whether the query size affected the retrieval process by using a subset of queries where the size of each one was less than 15. In following, all the proposed models as well as the set-based model were utilized as a basis of comparison. For each given query, the average precision of each model was calculated and afterwards, was compared with the average precision of the specific query of the set-based model. This specific comparison was expressed by the difference of the corresponding average precision values. In this way, we can determine if the under review model outperforms the set-based model.

Furthermore, the performance of each model against the set-based model in the set of queries can be quantified; this can be implemented by counting the number of queries where the model has greater average precision. It is important to note that a query is disregarded in case of equal results in that specific query between the models. Specifically, we have made evaluations with different values of importance variable h as well as pruning percentage value P in more than 84 experimental combinations.

For starters, the focus of study was to determine the performance of the important nodes amplification on the in-edges as well as out-edges of the main core model ($MainCore$) in comparison with the simple graphical extension of the set-based model (GSB). It seems necessary to amplify the weights of both type of edges since the results were better in that case as presented in Fig. 2.



(a) With Considering (b) Without Considering

Fig. 2. Amplification on important nodes-edges

In following, we examined the performance of all 4 models, namely the simple graphical extension of the set-based model (GSB), the graphical extension of the set-based model using main core's nodes as important nodes ($MainCore$), the graphical extension of the set-based model utilizing the most dense subgraph nodes as important nodes ($Density$) and the graphical extension of the set-based model employing CoreRank function ($CoreRank$) to determine the important nodes on 4 types of pre-processed textual data as depicted in Table 1. All the relevant information for each document has been initially included while this information was gradually pre-processed. Therefore 4 clusters of results are created, having different P values in each case. We can observe that in instances

with low pre-process, the MainCore and the CoreRank models perform close to the set-based model, which is less affected by the noisy data.

Table 1. Proposed models evaluation for different pre-processed cases

Pre-processed case	Size of index	Result	Model
Low	48452	49	MainCore
None	58621	41.17	CoreRank
Slight	36988	47.42	MainCore
Full	11366	60.20	MainCore, GSB

Emphasizing on the full pre-processed case, which only consists of the document title and text, we can observe in Fig. 3 that our proposed models surpass the set-based model. At the best case, both GSB and $MainCore$ achieved a score of $60,2\%$, having statistical significance p-value equal to $0,02\%$. The $Density$ model tends to be similar to the $MainCore$ model due to the fact that the majority of graphs have low decomposition levels. However, they differentiate at cases, which the results are not so impressive.

In addition, we examined the impact of the significance variable h in Fig. 4. When the percentage of pruning is equal to 30%, the GSB, $MainCore$ and $Density$ models tend to present similar results. On the other hand, the $coreRank$ model comes last because of the need for higher pruning percentage value.

Fig. 3. Full pre-processed case

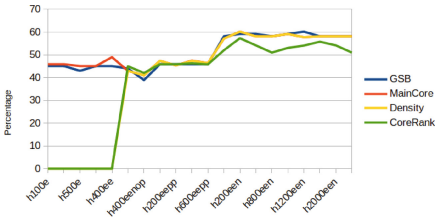


Fig. 4. Each significance variable case

Until this point, our aim was primarily focused on the number of queries that proposed models, which are actually superior to the set-based model disregarding the magnitude of that difference. In Fig. 5, all results are sorted in ascending order based on the set-based model, thus allowing to elaborate on the difference in average precision on each query. We choose to depict an average case and a case where $MainCore$, $Density$ and $CoreRank$ fall behind GSB.

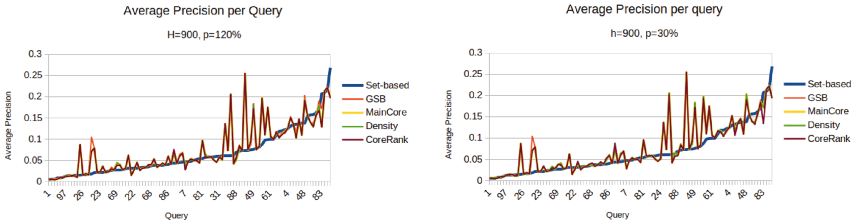


Fig. 5. Difference with set-based model

Finally, considering only the average precision on each query and ignoring the queries where those values occurred, we can introduce Fig. 6, which illustrates the course of average precision on each model in the above mentioned cases.

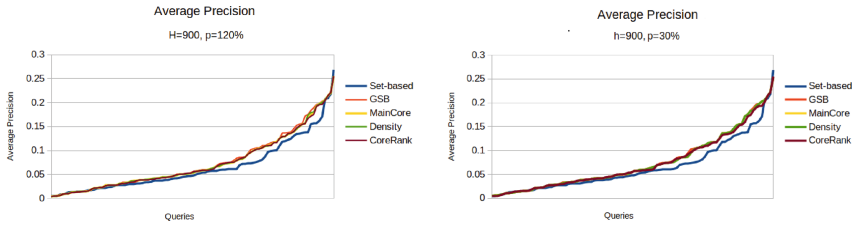


Fig. 6. Average precision of each model

5 Conclusions and Future Work

In this paper, we have proposed a number of methods for extending the set-based model with the use of graphs. These methods are improvements of the set-based model and are capable of handling noisy data in a fairly well manner. Experimental evaluation proves that the query size affects the retrieval results.

Nonetheless, the proposed methods perform similarly in some occurrences. Because of the fact that the graph is complete, as we assumed during the graph creation process, the pruned percentage variable seems to be more important than the significance variable; that precisely should be the motive for further research. One alternative is to relate each term of the document in sentence or paragraph level implemented by a moving window or alternatively utilize a proximity penalty at the edge weighting process.

Acknowledgement. Christos Makris and Andreas Kanavos have been co-financed by the European Union and Greek national funds through the Regional Operational Program "Western Greece 2014–2020", under the Call "Regional Research and Innovation Strategies for Smart Specialisation - RIS3 in Information and Communication Technologies" (project: 5038701 entitled "Reviews Manager: Hotel Reviews Intelligent Impact Assessment Platform").

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: 20th International Conference on Very Large Data Bases (VLDB), pp. 487–499 (1994)
2. Batagelj, V., Zaversnik, M.: An o(m) algorithm for cores decomposition of networks. CoRR cs.DS/0310049 (2003)
3. Freeman, L.C.: A set of measures of centrality based on betweenness. Sociometry **40**, 35–41 (1977)
4. Freeman, L.C.: The development of social network analysis: a study in the sociology of science. Soc. Netw. **27**, 377–384 (2004)
5. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM **46**(5), 604–632 (1999)
6. Litvak, M., Last, M.: Graph-based keyword extraction for single-document summarization. In: Workshop on Multi-source Multilingual Information Extraction and Summarization (MMIES), pp. 17–24 (2008)
7. Pôssas, B., Ziviani, N., Meira Jr., W., Ribeiro-Neto, B.A.: Set-based model: a new approach for information retrieval. In: 25th International SIGIR Conference on Research and Development in Information Retrieval, pp. 230–237 (2002)
8. Pôssas, B., Ziviani, N., Meira Jr., W., Ribeiro-Neto, B.A.: Set-based vector model: an efficient approach for correlation-based ranking. ACM Trans. Inf. Syst. (TOIS) **23**(4), 397–429 (2005)
9. Prountzos, D., Pingali, K.: Betweenness centrality: algorithms and implementations. In: SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP), pp. 35–46 (2013)
10. Rousseau, F., Vazirgiannis, M.: Graph-of-word and TW-IDF: new approach to ad hoc IR. In: 22nd International Conference on Information and Knowledge Management (CIKM), pp. 59–68 (2013)
11. Rousseau, F., Vazirgiannis, M.: Main core retention on graph-of-words for single-document keyword extraction. In: Hanbury, A., Kazai, G., Rauber, A., Fuhr, N. (eds.) ECIR 2015. LNCS, vol. 9022, pp. 382–393. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16354-3_42
12. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Commun. ACM (CACM) **18**(11), 613–620 (1975)
13. Sedgewick, R.: Algorithms in C, Part 5: Graph Algorithms, 3rd edn. Addison-Wesley-Longman, Boston (2002)
14. Seidman, S.B.: Network structure and minimum degree. Soc. Netw. **5**(3), 269–287 (1983)
15. Shaw, W.M., Wood, J.B., Wood, R.E., Tibbo, H.R.: The cystic fibrosis database: content and research opportunities. Libr. Inf. Sci. Res. **13**(4), 347–366 (1991)
16. Tixier, A.J., Malliaros, F.D., Vazirgiannis, M.: A graph degeneracy-based approach to keyword extraction. In: Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1860–1870 (2016)
17. Zhou, F., Mahler, S., Toivonen, H.: Simplification of networks by edge pruning. In: Berthold, M.R. (ed.) Bisociative Knowledge Discovery. LNCS (LNAI), vol. 7250, pp. 179–198. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31830-6_13