Keyword Extraction Using Unsupervised Learning on the Document's Adjacency Matrix

Eirini Papagiannopoulou and **Grigorios Tsoumakas** and **Apostolos N. Papadopoulos**
School of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
{epapagia,greg,papadopo}@csd.auth.gr

Abstract

This work revisits the information given by the graph-of-words and its typical utilization through graph-based ranking approaches in the context of keyword extraction. Recent, well-known graph-based approaches typically employ the knowledge from word vector representations during the ranking process via popular centrality measures (e.g., PageRank) without giving the primary role to vectors' distribution. We consider the adjacency matrix that corresponds to the graph-of-words of a target text document as the vector representation of its vocabulary. We propose the distribution-based modeling of this adjacency matrix using unsupervised (learning) algorithms. The efficacy of the distribution-based modeling approaches compared to state-of-the-art graph-based methods is confirmed by an extensive experimental study according to the F_1 score. Our code is available on GitHub[1].

1 Introduction

Automatic Keyword Extraction (AKE) intends to discover a limited but concise set of words that reflect the main topics discussed within a text document, avoiding the expensive and time-consuming process of manual annotation by experts (Vega-Oliveros et al., 2019). Besides, many keyphrase extraction methods form and rank the candidate phrases using the previously scored candidate unigrams by a keyword extractor, as keyphrases consist of n-grams with $n \geq 1$ (Wan and Xiao, 2008a; Hasan and Ng, 2014; Florescu and Caragea, 2017).

Both supervised and unsupervised approaches are quite famous for the AKE task (Papagiannopoulou and Tsoumakas, 2020). During the last two years, the research community pays significant attention on (supervised) deep learning methods (Chan et al., 2019; Wang et al., 2019; Zhao and Zhang, 2019; Chen et al., 2020) as the performance of the unsupervised ones shows a relative stagnation (or minimal improvements) compared to the supervised techniques. The use of standard external tools for grammatical/syntactic analysis and information sources such as (pre-trained) static word embeddings (Bennani-Smires et al., 2018; Mahata et al., 2018) that have a bias over the corpora domains used for training may also exacerbate the problem. Moreover, most methods suggest a fixed or relative with the document length number of keywords dissociating the number of returned keywords from the number of topics discussed in the document (Rousseau and Vazirgiannis, 2015). However, unsupervised methods are of timeless interest. They often are domain or language-independent and do not need any labelled data to train models compared to the supervised ones. The graph-based approaches (i.e., the most popular category of the unsupervised AKE) consider the "central" nodes of a graph-of-words as the most representative ones usually according to (variations of) the PageRank (Brin and Page, 1998) centrality, i.e., the most effective graph-based ranking method employed by the majority of the state-of-the-art (Mihalcea and Tarau, 2004; Wan and Xiao, 2008b; Florescu and Caragea, 2017; Vega-Oliveros et al., 2019). Additionally, traditional (semi-)supervised, or even deep learning approaches (Wang and Li, 2017; Gollapalli et al., 2017; Ye and Wang, 2018) utilize the unsupervised methods mentioned earlier to improve their performance.

This work takes a novel unsupervised path to keyword extraction revisiting the information provided by the graph-of-words and its conventional utilization via PageRank. Inspired by the recent approach of Papagiannopoulou et al. (2020), we investigate the effectiveness of the distribution-based modeling of the adjacency matrix, that corresponds to various versions of the (unweighted,

[1]https://github.com/epapagia/KE_adjacency_matrix_modelling

weighted or/and enhanced with positional information) graph-of-words for the target document. We also propose the use of more advanced unsupervised algorithms to model the main distribution of the adjacency matrix, determine the number of the retrieved keywords at document level, and score/rank the corresponding candidate keywords. To the best of our knowledge, this is the first work that proposes such modeling of the adjacency matrix using unsupervised machine learning approaches in the context of keyword extraction. Our empirical study confirms the efficacy of the distribution-based modeling approaches (regarding the F_1 score) on six datasets (full-texts of scientific publications, paper abstracts and news articles) compared to state-of-the-art graph-based methods.

The main contributions of this work are as follows: (a) We propose multiple ways for the distribution-based modeling of the adjacency matrix that corresponds to various versions of the graph-of-words for a target document. Specifically, we investigate the use of unsupervised (learning) algorithms to model the distribution of the adjacency matrix, score the candidate words, and (b) discover the appropriate number of keywords (Section 3). (c) Our empirical study provides strong evidence about the relationship between the graph-based techniques and the proposed ones emphasizing the cases that the distribution-based modeling is more promising (Section 4). Finally, Section 2 and 5 present related work on unsupervised AKE (issues/trends) as well as conclusions and future directions of our work, respectively.

2 Related Work

Issues. The comprehensive representation of the information via graphs and the efficiency of the graph-based ranking methods (e.g., PageRank (Brin and Page, 1998), HITS (Kleinberg, 1999), etc.) in many applications (including keyword extraction) led the research community to show a preference to graph-based AKE using unsupervised approaches (Papagiannopoulou and Tsoumakas, 2020). The popular TextRank (Mihalcea and Tarau, 2004) first builds an undirected, unweighted graph-of-words representation and runs the PageRank algorithm until convergence. In this vein, SingleRank (Wan and Xiao, 2008b), RAKE (Rose et al., 2010), ExpandRank (Wan and Xiao, 2008b), and CollabRank (Wan and Xiao, 2008a) are extensions of TextRank. The first two methods add weights to edges, equal to the number of co-occurrences of the two corresponding words within the predefined window, whereas, the last ones incorporate information from relevant documents. Much later, PositionRank (Florescu and Caragea, 2017) achieved significantly higher performance proposing a biased PageRank that considers both the word-word co-occurrences and the word's positions. Then, Biswas et al. (2018) and Vega-Oliveros et al. (2019) proposed graph-based keyword extraction methods that combine multiple centrality measures.

Another important issue is choosing the right number of keywords for a document. Rousseau and Vazirgiannis (2015) apply the concept of K-Core on the graph-of-words of a document retaining only the nodes from the main core as keywords. Their method is parameter-free as the K-Core principle adjusts the number of keywords concerning each graph's structure. Later, Tixier et al. (2016) show that retaining only the main core (or truss (Cohen, 2008)) is suboptimal as the complete set of a document's gold keywords cannot appear within a single subgraph and propose alternative heuristics (stopping criteria) to remove undesired words.

Trends. Information coming from word embeddings (Mikolov et al., 2013) proved useful for the AKE task. Numerous AKE methods use word embeddings (Mnih and Hinton, 2007; Bojanowski et al., 2017; Joulin et al., 2017) as an (external) semantic knowledge source. Representative graph-based approaches are the one of Wang et al. (2015) and Key2Vec (Mahata et al., 2018) that incorporate semantic information from pre-trained distributed word representations and word embeddings trained on a domain-specific corpus, respectively. Both methods utilize the information from word embeddings through the usual way of graph-based ranking without giving to the vector representation of terms the primary role. On the contrary, Papagiannopoulou and Tsoumakas (2018) present the Reference Vector Algorithm (RVA) that uses *local* GloVe (Pennington et al., 2014) word vectors (i.e., trained only on the target document). EmbedRank (Bennani-Smires et al., 2018) uses pre-trained sentence embeddings, Sent2Vec (Pagliardini et al., 2018), to embed both candidate terms and documents in the same high-dimensional vector space. Finally, Papagiannopoulou et al. (2020) proposed an unsupervised AKE method that uses the weighted adjacency matrix's rows as word vectors to model their distribution. The authors show that the centre of the distribution is closer to the non-keywords, as the main bulk of words are neutral or slightly relevant to the documents' topics.

3 Our Approach

3.1 Document Pre-processing

First, we eliminate from the target document punctuation marks and words that (a) are stopwords , (b) consist only of numbers, (c) have length less than two characters to avoid trivial or insignificant terms. Before we get the final set of candidate words as keywords, we use stemming.

3.2 Creation of the Adjacency Matrix

The majority of the graph-based approaches measure the importance of the graph-of-words' nodes using PageRank (see Section 2) that adds more value to a node connected with high-scoring nodes rather than low-scoring ones through an iterative process. However, our approach follows a different direction by identifying the most central (i.e., significant) words of the graph-of-words via the distribution-based modeling of the corresponding adjacency matrix.

We investigate our approach's effectiveness on three distinct versions of the adjacency matrix that correspond to three variants of the graph-of-words (i.e., unweighted, weighted with/without enhanced with positional information) for a target document. Specifically, given a set of unique, valid words of the text, $d \in D$, we could have one of the following types of word vectors (each row of the adjacency matrix constitutes a vector representation of a specific word):

a. *Unweighted adjacency matrix*, $A_{N\times N}$ where $N = |D|$. The $A_{N\times N}$ matrix represents the undirected[2] unweighted graph-of-words $G = (U, E)$, U is the set of vertices (that correspond to the set of words $d \in D$) and E is the set of edges; Each element $A_{i,j}$ is 1 when there is an edge from vertex u_i to vertex u_j ($u_i \neq u_j$) of G, i.e., the corresponding words d_i and d_j co-occur within a window of T words, and 0 when there is no edge, $i, j \in [1..N]$.

b. *Weighted adjacency matrix*, $A'_{N\times N}$ with $N = |D|$. The $A'_{N\times N}$ matrix represents the undirected weighted graph-of-words $G' = (U, E')$, U is the set of vertices and E' is the set of edges; Each element $A'_{i,j}$ contains the weight of the edge from vertex u_i to vertex u_j ($u_i \neq u_j$), i.e., the number of co-occurrences of the corresponding words d_i and d_j within a window of T words, $i, j \in [1..N]$. In case that there is no edge connecting the two nodes, $A'_{i,j} = 0$.

c. *Weighted adjacency matrix with positional information*, $Q_{N\times N}$ where $N = |D|$. The $Q_{N\times N}$ matrix represents the undirected weighted graph-of-words $A' = (U, E')$ but also incorporates positional information, i.e.,

$$Q = A' \odot P$$

where $\odot$ is the element-wise multiplication symbol, A' is the weighted adjacency matrix, $A'_{N\times N}$, detailed in (b) and $P_{N\times N}$ is a positional matrix such that each element is defined as:

$$P_{i,j} = \frac{1}{s(d_i) + s(d_j)}$$

where $s(d)$ gives the first sentence where the word d occurs in the document.

3.3 Distribution-based Modeling and Candidates Scoring

The next step of our approach is the distribution-based modeling of the adjacency matrix that corresponds to one of the various versions of the target document's graph-of-words described above (Section 3.2) and the candidate words' scoring. In this section, we detail three distribution-based modeling alternatives describing the intuition behind each one approach and the scoring functions used to give the final ranking of the words as keywords.

3.3.1 The Mean Vector Approach

Papagiannopoulou et al. (2020) proposed *Local Vectors* (LV), an unsupervised AKE method that uses the weighted adjacency matrix of the graph-of-words as word vectors to model the distribution of the target document's words by averaging the corresponding vectors (i.e., rows of the matrix). The authors show that the centre of the distribution is closer to the non-keywords, as the main bulk of words are neutral or slightly relevant to the documents' topics. Moreover, in the same work, they show through an empirical study that the *local* word vectors coming from the weighted adjacency

[2]i.e., the adjacency matrix is symmetric.

matrix mentioned above encode statistical information equivalent to the one encoded by the local run of GloVe on a single document proposed in Papagiannopoulou and Tsoumakas (2018).

Particularly, in this work, we consider the sample's estimated mean $\boldsymbol{\mu}$ of the corresponding vector matrix, i.e., the $A_{N\times N}$ or $A'_{N\times N}$ or $Q_{N\times N}$, as the distribution's center (each word participates once in the computation). Then, we score each word with a value S, according to the following formula: $S(\boldsymbol{x}) = d(\boldsymbol{\mu}, \boldsymbol{x})$, where $d(\boldsymbol{\mu}, \boldsymbol{x})$ is the Euclidean distance between the mean vector $\boldsymbol{\mu}$ and the vector representation $\boldsymbol{x}$ of the word, as distance metrics that incorporate the vectors' magnitude capture the similar behaviour of the non-keywords' vectors over the keywords' ones (Papagiannopoulou et al., 2020), i.e., $d(\boldsymbol{\mu}, \boldsymbol{x}) = \sqrt{\sum_{i=1}^{N}(x_i - \mu_i)^2}$, where N is the number of dimensions. The higher the score S, the more important the word for the document, i.e., we are interested in words with high distance values, as most of the words, which determine the distribution's center, are non-keywords. The main difference with the approach proposed by Papagiannopoulou et al. (2020) is that we score the words based only on their distance from the mean vector without involving any external heuristics such as the word's position. This way, we consider in advance any positional information via the Q adjacency matrix (i.e., incorporated in the vector representation).

3.3.2 Unsupervised Learning Approaches

One-Class SVM. Instead of calculating the distribution's center of the adjacency matrix by averaging its rows, we could use geometric concepts such as hyperspheres or hyperplanes to delimit the area of space that includes most of the word vectors (i.e., the main bulk of unimportant words) and, then, score the candidates using functions that express the vectors' deviation from the main distribution. According to this approach, the most important words are the most outlying ones (i.e., outliers) as most words are neutral or slightly relevant to the documents' topics (i.e., inliers). Tax and Duin (1999a,b) proposed a method based on SVM (Cortes and Vapnik, 1995), called One-Class SVM, that seeks the smallest hypersphere consisting of all the dataset points. Thus, training this model may reject a fraction of the positively-labelled training objects when this adequately minimizes the hypersphere volume.

There are also other approaches, such as the one of Schölkopf et al. (1999), which is similar, but instead of using a small hypersphere, it uses a hyperplane which is far from the origin (this is the version implemented by scikit-learn[3] and used in our study). This algorithm employs a function f that takes the value +1 in a "small" region, covering most of the data points, and -1 elsewhere. Formally, suppose the dataset consists of the word vectors (samples) $\boldsymbol{x}$ coming from the corresponding adjacency matrix $X_{N\times N}$ (i.e., $X_{N\times N}$ can be one of the $A_{N\times N}$, $A'_{N\times N}$, $Q_{N\times N}$). Let Φ be a feature map $X \to F$[4], i.e., a specific dot product space. Then, we can separate the dataset's word vectors from the origin by solving the following quadratic optimization problem:

$$\min_{\boldsymbol{w},\xi,\rho} \frac{1}{2}||\boldsymbol{w}||^2 + \frac{1}{\nu l}\sum_i \xi_i \; - \rho$$

subject to $(\boldsymbol{w} \cdot \Phi(\boldsymbol{x_i})) \geq \rho - \xi_i,\ \xi_i \geq 0$
where i represents the i^{th} sample, $l = N$, ν is the percentage of samples considered as outliers (the expected keywords' ratio), ξ_i are the slack variables that relax the constraints, ρ refers to the distance of the hyperplane from the origin[5] and $\boldsymbol{w}$ represent the parameters of the SVM that define the hyperplane (we need to learn them using the dataset's samples)[6]. Then, the decision function $f(\boldsymbol{x}) = sgn((\boldsymbol{w} \cdot \Phi(\boldsymbol{x})) - \rho)$ will be positive for the most samples $\boldsymbol{x_i}$ in the dataset. In our case, an ideal scoring function that ranks the corresponding document's words is the signed distance to the separating hyperplane that will be positive for the main bulk of words and negative for the different ones. We consider only the words with a negative score as candidate keywords (the lower the value, the higher the word's importance).

We have experimented with various kernel functions, e.g., polynomial, sigmoid, etc. but the most suitable in our case is the Radial Base Function (RBF). Formally, The RBF kernel on two samples $\boldsymbol{x}$ and $\boldsymbol{x'}$ is defined as:

$$K(\boldsymbol{x}, \boldsymbol{x'}) = exp\left(- \frac{||\boldsymbol{x} - \boldsymbol{x'}||^2}{2\sigma^2} \right)$$

[3] `https://scikit-learn.org/stable/index.html`

[4] F is a dot product space such that the dot product in the image of Φ can be computed by evaluating some kernel.

[5] This distance is equal to $\frac{\rho}{||\boldsymbol{w}||}$.

[6] i.e., ρ and $\boldsymbol{w}$ solve the problem.

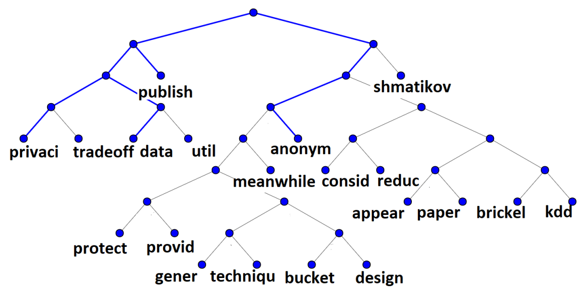


Figure 1: An isolation tree built based on the adjacency matrix's word vector representations for the article entitled "On the Tradeoff between Privacy and Utility in Data Publishing" with golden keywords: *anonymity, data, publishing, privacy*. The isolation tree recursively divides the 20 samples (i.e., words used to train the tree) by randomly selecting an attribute (matrix dimension) b and a split value m until either the node has only one instance or all node's data have the same values. Keywords tend to have shorter path lengths than the non-keywords that constitute the main bulk of (unimportant) words.

where $\sigma \in \mathbb{R}$ is a kernel parameter and $||\boldsymbol{x} - \boldsymbol{x}'||^2$ can be considered as the squared Euclidean distance between the two word vectors. These distances between the pairs of the word feature vectors incorporated by the RBF kernel make it the best choice. Moreover, distance metrics that consider the vectors' magnitude (e.g., the Euclidean distance) can capture the non-keywords' vectors' similar behavior over the keywords' ones as mentioned earlier in Section 3.3.1.

Isolation Forest. Instead of modeling the vectors' distribution and then estimating the distance from a reference point (e.g., the mean vector or the hyperplane), we propose to detect and rank the few different (important) word vectors via the mechanism of isolation Liu et al. (2008, 2012) utilizing the binary tree structure, called isolation tree. Because of the susceptibility to isolation, the few outlying word vectors (i.e., expected to be the important ones) are more prone to be isolated closer to the root of an isolation tree than the common ones. An isolation forest builds an ensemble of isolation trees for the given set of word vectors. The forest of random trees collectively produces shorter path lengths[7] for the outlier samples, i.e., the ones we search.

In other words, isolation forest is a tree-based algorithm built around the theory of decision trees and random forests. It also creates many isolation (decision) trees, but it calculates the *path length* necessary to isolate an observation in the tree. The idea is that keywords as a minority in a document can be treated as anomalies and thus are easier to be isolated because there are fewer conditions required to separate them from the "normal" non-keywords. Therefore, outliers (i.e., keywords) will have shorter paths than the "normal" non-keywords and reside closer to the tree's root. When many isolation trees are created, the forest is necessary to average the corresponding scores (path length calculations), providing a sense about the words that are indeed outliers.

Figure 1 shows an isolation tree built based on the Q adjacency matrix's word vector representations for a computer science abstract from the KDD collection (Caragea et al., 2014). The article entitled "On the Tradeoff between Privacy and Utility in Data Publishing" is accompanied by the following golden keywords: *anonymity, data, publishing, privacy*. The number of samples to draw from X to train each base estimator is equal to 20. We also applied PCA on X and use the two first principal components to facilitate visualization. The isolation tree recursively divides the 20 samples by randomly selecting an attribute b and a split value m, until either the node has only one instance or all data at the node have the same values. We notice that keywords tend to have shorter path lengths than the non-keywords. Similar isolation trees, supportive of our crucial intuition, are obtained from other documents, too.

A more in-depth view of the Isolation Forest scoring function reveals that itself defines a "natural" threshold that determines whether a sample belongs to inliers or not by borrowing the analysis of Binary Search Trees (BSTs) as isolation trees have an equivalent structure (Preiss, 2000). This property is remarkable as the number of topics a document discusses should determine the corresponding number of keywords instead of suggesting a fixed or proportional to the text size number of keywords as most methods do. According to the theory, the average path length c(ψ) of unsuccessful searches in a BST (i.e., the equivalent of external node terminations in an isolation tree) given a sample set of ψ instances is:

$$c(\psi) = \begin{cases} 2H(\psi - 1) - 2\frac{\psi-1}{n}, & \psi > 2, \\ 1, & \psi = 2, \\ 0, & otherwise. \end{cases}$$

[7]The path length of a point x is measured by the number of edges x traverses an isolation tree from the root node until the traversal is terminated at an external node.

where $H(i)$ is the harmonic number (estimated by $ln(i)+$ the Euler's constant). Hence, the Isolation Forest scoring function is:

$$s(x, \psi) = 2^{-\frac{E(h(x))}{c(\psi)}}$$

where x is the sample, $h(x)$ is the path length of x and $E(h(x))$ is the average of $h(x)$ for a collection of isolation trees. The above function ensures that samples with scores close to 1 imply diversity from the majority (i.e., $E(h(x)) \to 0$), whereas scores much lower than 0.5 indicate normal samples (i.e., $E(h(x)) \to \psi - 1$). Also, suppose all instances have a score of approximately equal to 0.5. In that case, we can consider the whole sample as a set of normal instances (i.e., $E(h(x)) \to c(\psi)$). The above findings transform the value 0.5 into an especially important threshold for determining a case as different from the whole sample or not.

4 Experimental Study

4.1 Setup

We choose six popular datasets: three collections with full-text publications, i.e., NUS (Nguyen and Kan, 2007), Semeval (SE) (Kim et al., 2010), and ACM (Krapivin et al., 2008) with 211, 244, and 2304 documents, respectively, two with scientific abstracts, i.e., KDD (Caragea et al., 2014) and WWW (Gollapalli and Caragea, 2014) with 755, and 1330 documents, respectively, and one with news texts, i.e., DUC-2001 (DUC) (Wan and Xiao, 2008b) with 308 documents. This way, we include to our study both long and short texts, either scientific or news articles. SE is already separated into training (144) and test (100) sets, and for the ACM separation, most works choose the first 400 papers from the ACM following Meng et al. (2017) as test set. However, there are no guidelines for separating the NUS, KDD, WWW, and DUC datasets. Thus, we pick the first 330 from WWW, the last 100 papers from NUS (Papagiannopoulou et al., 2020), and the last 100 from DUC, alphabetically ordered as the test data. We use the whole KDD dataset as test set as we do not use it for parameters' tuning.

In addition to the proposed approaches for AKE, i.e., the new version of LV, the Isolation Forest (IF) and the One-Class SVM (OC), four state-of-the-art unsupervised graph-based AKE methods participate in this empirical study: K-Core (K) (Seidman, 1983; Batagelj and Zaversnik, 2011), PageRank (P) (Mihalcea and Tarau, 2004; Wan and Xiao, 2008b), Betweenness (B), and Node degree (N) (the last one proposed first by Rose et al. (2010). We present the experimental results organized in three groups based on the type of information used to run (Tables 2, 3). The first two groups include the methods' runs on the unweighted and weighted graphs-of-words/adjacency matrices, i.e., with an A and A' subscript on the right of each method according to the notation introduced in Section 3.2, respectively (e.g., K_A means that K method runs on an unweighted graph-of-words, whereas $\mathrm{K}_{A'}$ runs on the weighted one, i.e., weighted K-Core of Batagelj and Zaversnik (2011)). The third group includes the proposed methods' runs on the Q adjacency matrix (weighted with words' co-occurrences and positional information) and a Personalized weighted variant of PageRank that considers both node as well as the typical edge weights ($P_{A''}$). The node weight is equal to $\frac{1}{s}$, where s is the first sentence's index that the corresponding word occurs in the document. In all cases, the methods build the graph-of-words following the pre-processing steps described in Section 3.1.

After splitting the golden keyphrases into unigrams, we use exact string matching to determine the number of correctly matched words with the golden ones for a document following the paradigm of Tixier et al. (2016). We also apply stemming to the output of the methods and the article's golden unigrams as a pre-processing step before the evaluation process. We employ the authors' keywords as a gold evaluation standard for all academic documents (long/short) except for the news dataset where only the readers' keywords are available. We used the IsolationForest and OneClassSVM classes from the scikit-learn[8] library for the IF and OC, respectively. For the implementation of the competitive approaches, we employ the PKE toolkit (Boudin, 2016), the NetworkX[9] and the gowpy[10] python libraries.

We use one dataset per text category (full-texts, abstracts, news), i.e., the training sets of NUS, WWW, DUC, to determine IF and OC models' tuning parameters to optimize the F_1 score. The parameters chosen for the experiments on test sets are for the IF $n_{estimators}$=200, $max_{samples}$=auto, $max_{features}$=0.75 and the OC *kernel*=rbf, *gamma*=scale. The best percentages

[8]https://scikit-learn.org/stable/index.html

[9]https://networkx.org/

[10]https://github.com/GuillaumeDD/gowpy

for the IF's *contamination* and OC's *nu* parameters (related to outliers' ratio) are 0.05, 0.1, 0.2 for the full-text publications, news texts and abstracts, respectively (used in the context of F_1@20 scores calculation, see Table 2). Furthermore, we compute the F_1 scores in Table 3 based on the number of keywords returned by the K_A, $K_{A'}$, IF_A, $IF_{A'}$ and IF_Q (the value of the IF's contamination parameter is equal to *auto*[11]) approaches, i.e., F_1@$|K_A|$, F_1@$|K_{A'}|$, F_1@$|IF_A|$, F_1@$|IF_{A'}|$, F_1@$|IF_Q|$, respectively. The LV approach is parameter-free.

We follow the paradigm of existing experimental studies from the related task of keyphrase extraction (Wan and Xiao, 2008a; Bougouin et al., 2013), and set a window size T equal to 10 to construct the graphs-of-words used by the graph-based approaches. Earlier, Wan and Xiao (2008b) show that in the case of weighted graphs-of-words, the greater the window size, the higher the extractor's accuracy (window sizes greater than 10 achieve more or less the same accuracy level).

F_1	NUS		WWW		DUC	
	T_5	T_{10}	T_5	T_{10}	T_5	T_{10}
LV_A	0.319	**0.324**	0.279	**0.282**	0.389	**0.410**
IF_A	0.313	**0.319**	**0.294**	0.284	0.372	**0.377**
OC_A	0.093	**0.126**	0.237	**0.243**	0.265	**0.284**
$LV_{A'}$	0.311	**0.315**	0.283	**0.285**	0.401	**0.405**
$IF_{A'}$	0.319	**0.324**	0.313	**0.314**	0.376	**0.395**
$OC_{A'}$	0.129	**0.142**	0.256	**0.261**	0.282	**0.293**
LV_Q	**0.339**	0.336	**0.280**	0.278	0.408	**0.416**
IF_Q	0.338	**0.342**	0.321	**0.335**	0.383	**0.413**
OC_Q	**0.339**	0.338	0.266	**0.271**	**0.344**	0.343

Table 1: F_1@20 of the LV, IF and OC keyword extraction methods using different types of adjacency matrix (A, A', Q) created with two different window sizes, $T = 5$ (T_5) and $T = 10$, on three different datasets (NUS, WWW, DUC). The highest values appear in bold font.

Moreover, Table 1 shows the F_1@20 scores of LV, IF, and OC on the training sets of three representative datasets NUS, WWW and DUC (one from each category of documents, long/short scientific articles and news texts, respectively) using unweighted (A), weighted (A'), and weighted with positional information (Q) adjacency matrix with two different window sizes, a lower widow size $T = 5$ (T_5) and the usual one $T = 10$ (T_{10}). The highest F_1 scores are highlighted in bold font. The experimental results confirm that smaller window sizes led to lower F_1 scores for most of the proposed methods. However, in few cases where the methods with T_5 give higher F_1 scores compared to those with T_{10}, the differences are not statistically significant according to the two-sided Wilcoxon signed-rank nonparametric test. Finally, another reason to consider the same co-occurrence window size for both the state-of-the-art graph-based approaches and the proposed ones is our interest in investigating the methods' efficacy employing the same words' context (captured in a specific window size).

[11] value for the "natural" threshold of outliers for IF, see Section 3.3.2

4.2 Performance Evaluation Results

Table 2 shows the F_1@20 scores of various keyword extraction methods, whereas Table 3 presents the F_1 scores calculated based on the returned number of keywords by K_A, $K_{A'}$, IF_A, $IF_{A'}$, and IF_Q on the six datasets using unweighted (A), weighted (A'), and weighted with positional information (Q) graph-of-words or adjacency matrix. The highest F_1 scores in both tables are highlighted in red bold font. Table 2 presents the best scores for each group of methods in bold, whereas the second best are underlined. We have also checked the statistical significance of the results using two-sided Wilcoxon signed-rank nonparametric test between the graph-based (the ones in the gray background) and the distribution-based modeling approaches (LV, IF, OC) at significance level 0.01. Our analysis shows that differences in values $> 2\%$ are statistically significant. In case of statistical significance between the values of two methods whose difference is $\leq 2\%$, a superscript with the name of the corresponding graph-based method is added on the distribution-based modeling approach. We compute statistical significance separately for the groups of methods that use edge weights, node and edge weights, and no weights, respectively, to facilitate the results' interpretation.

Table 2 shows that in most cases except for the news collection (DUC), the more information we consider, i.e., both words' co-occurrences and positional info, the higher F_1 scores we achieve (e.g., OC_Q's high scores compared to the ones of OC_A, $OC_{A'}$). Particularly, the transition from the unweighted graph-of-words/adjacency matrix to their weighted versions slightly improves the performance in almost all methods besides the B's (in all datasets) and LV's scores (in longer documents). $IF_{A'}$ outperforms the competitive approaches that consider edge weights to score the

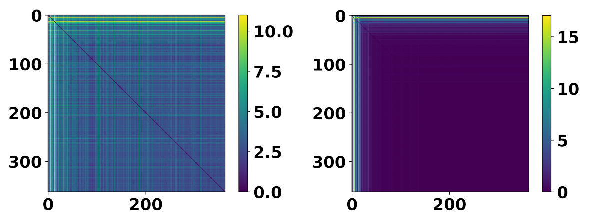
F_1	ACM	NUS	SE	DUC	WWW	KDD
P_A	$\underline{0.329}$	**0.331**	0.261	0.381	$\underline{0.279}$	$\underline{0.265}$
B_A	0.318	0.325	0.260	0.354	0.270	0.256
N_A	**0.330**	$\underline{0.328}$	0.263	**0.385**	**0.284**	**0.269**
LV_A	0.327^{B}	0.321	$\underline{0.267}$	$\underline{0.384}$	0.277^{N}	$\underline{0.265}^{B}$
IF_A	0.319^{P,N}	0.320	**0.268**	0.362^{P}	**0.284**^{B}	0.260
OC_A	0.154	0.144	0.133	0.244	0.232	0.221
$P_{A'}$	**0.331**	**0.326**	$\underline{0.269}$	$\underline{0.383}$	0.284	0.270
$B_{A'}$	0.314	$\underline{0.317}$	0.257	0.349	0.257	0.247
$N_{A'}$	0.324	0.314	0.268	**0.388**	$\underline{0.288}$	$\underline{0.272}$
$LV_{A'}$	0.311$_{P,N}$	0.300	0.263	0.380	0.279^{N}	0.269
$IF_{A'}$	$\underline{0.328}^{B}$	**0.326**^{N}	**0.275**^{B}	0.368	**0.316**	**0.289**^{P,N}
$OC_{A'}$	0.158	0.153	0.142	0.289	0.249	0.242
$P_{A''}$	**0.374**	**0.358**	**0.300**	$\underline{0.383}$	0.285	0.269
LV_Q	$\underline{0.373}$	0.343^{P}	$\underline{0.299}$	0.328	0.268^{P}	0.260
IF_Q	0.353	$\underline{0.348}$	0.287	**0.384**	**0.338**	**0.305**
OC_Q	0.372	0.340^{P}	0.290	0.311	$\underline{0.286}$	$\underline{0.275}$

Table 2: F_1@20 of various keyword extraction methods using different types of graph-of-words/adjacency matrix (A, A', Q) on six different datasets. Superscripts on a method's score show statistical significance between the current method and the one whose name appears as superscript (see Section 4.2). The highest values appear in bold red font. The best scores for each group of methods are in bold, whereas the second best are underlined.

candidates (2nd group of methods with subscript A') in four out of six datasets (NUS, SE, WWW, KDD) and achieves high scores in ACM and DUC with statistically insignificant differences compared to the competitive methods. Moreover, the addition of positional weights compared to the typical use of edge weights increases most methods' performance remarkably apart from the LV's (in shorter documents) and the P's that remains almost invariable in shorter texts. In the 3rd group of methods, IF_Q ranks first in half datasets (DUC, WWW, KDD) and performs high in NUS and SE (without statistically significant differences from $P_{A''}$). Additionally, LV achieves quite high F_1 scores as well.

Figure 2 shows a visual interpretation of why the additional information facilitates the distribution-based approaches to distinguish the keywords from the non-keywords via heatmaps of the Euclidean distances between the word vectors of the A (2a) and Q (2b), respectively, for a news text. We notice that positions combined with words' co-occurrences help the text's keyword vectors diverge from the main distribution (see the few high distances/yellow or light green values that correspond mostly to the first words of the document that include many keywords). We also see that most vectors (common words - group of inliers) are close to each other (low/dark distance values). However, the distances between word vectors of A do not reveal any clear separation between the main distribution of common words and the minority of keywords making difficult the outliers' detection. Note that the words' ids (range from 0 to 363) correspond to the order of the words' presence in the text, confirming the importance of the positional information in the AKE task (keywords tend to appear at the beginning of a document). Similar plots are also obtained from multiple documents.

Next, we focus on the AKE methods that determine the number of returned keywords at document level, i.e., the K_A, $K_{A'}$, IF_A, $IF_{A'}$ and IF_Q. We study the results of Table 3, considering Table 4



(a) (b)

Figure 2: The distances between the main bulk of word vectors from the Q adjacency matrix (2b) range in low (dark) values compared to a minority of distant word vectors (yellow/green values). However, the word vectors of A (2a) do not provide such clear separation between the main distribution of common words and the minority of keywords (high and low distances are just as many).

F_1@T	ACM	NUS	SE	DUC	WWW	KDD
K_A	0.176	0.160	0.132	0.240	0.250	0.234
IF_A	0.273	0.305	0.274	0.186	0.307	0.267
$K_{A'}$	0.297	0.309	0.278	**0.300**	0.343	**0.323**
$IF_{A'}$	0.323	0.372	0.322	0.183	0.315	0.283
IF_Q	**0.360**	**0.413**	**0.345**	0.223	**0.347**	0.313

Table 3: F_1@T of K_A, $K_{A'}$, IF_A, $IF_{A'}$ and IF_Q methods on the 6 datasets, where T is equal to $|K_A|$, $|K_{A'}|$, $|IF_A|$, $|IF_{A'}|$ and $|IF_Q|$, respectively. The highest values appear in red bold font.

@	ACM	NUS	SE	DUC	WWW	KDD
$\|K_A\|$	74.1	70.2	71.5	53.8	24.5	25.4
$\|IF_A\|$	10.5	8.7	10.3	2.6	5.8	5.4
$\|K_{A'}\|$	8.7	8.4	7.6	18.1	12.4	12.4
$\|IF_{A'}\|$	6.7	6.1	7.1	2.3	5.0	4.6
$\|IF_Q\|$	7.5	6.6	7.5	2.3	4.8	4.4
$\|V\|$	757.7	772.4	641.8	268.8	58.2	60.1

Table 4: Average number of keywords returned by the K and IF methods using different types of information (A, A', Q). The last row shows the average number of candidate words $|V|$ per dataset.

that shows the average numbers of keywords returned by the methods mentioned earlier for each dataset. The last row in Table 4 shows the average number of candidate words |V| per dataset to give an impression for the texts' vocabulary sizes. We are interested in investigating which method returns the most "accurate" keywords' sets in terms of the corresponding F_1 scores. However, we should keep in mind that the methods are not evaluated on the same number of keywords. Through this discussion, our goal is to discover which AKE approach is more suitable for each type of documents in a general sense. The low F_1@$|K_A|$ scores of K_A compared to the F_1 scores of $K_{A'}$, IF_A, $IF_{A'}$ and IF_Q are plausible due to the low precision scores of K_A as the number of the words included in the K-Core of the unweighted graph-of-words is quite high (greater than 70, 53 and 24 returned keywords for the academic full-texts, news texts and scientific abstracts, respectively). In all datasets $K_{A'}$ outperforms K_A giving reasonable number of keywords. Moreover, in most datasets (scientific full-texts and abstracts), IF_Q outputs more accurate keyword sets (i.e., higher F_1@$|IF_Q|$ scores) than those returned by rest approaches. Exceptions are the performance on (a) the DUC (news) dataset where IF_A, $IF_{A'}$ and IF_Q detect lower number of words as keywords compared to the golden ones and (b) the KDD collection where the F_1@$|IF_Q|$ score achieved by IF_Q is slightly worse than the one of $K_{A'}$.

We also present the correlation according to the Spearman correlation coefficient between the IF's scoring function described in Section 3.3.2 and traditional weighting schemas, i.e., P, N, B, and K, for each information type used by IF and the rest graph-based methods, i.e., unweighted, weighted and weighted with positional information graphs-of-words/adjacency matrices. Table 5 shows that there is a very strong positive correlation (≥ 0.8) between the words' scores returned by IF and those produced by P and N for all information (input) types for almost all datasets, interpreting (partly) the comparable F_1 scores achieved by these methods. In this vein, there is a strong positive correlation (≥ 0.6) between IF and B in most cases. Moreover, the very strong positive correlation (≥ 0.8) on the datasets with full-texts of scientific publications goes hand-in-hand with the similar accuracy levels achieved in case there are no weights on the corresponding input. Finally, the K's output does not seem to correlate with the IF's output when no weights are used. However, if the methods use weights, the correlation between them turns into a strong/moderate positive one.

S.C.C.	ACM	NUS	SE	DUC	WWW	KDD
IF_A-P_A	**0.91**	**0.90**	**0.92**	**0.84**	**0.82**	**0.81**
IF_A-N_A	**0.92**	**0.91**	**0.92**	**0.84**	**0.81**	**0.81**
IF_A-B_A	**0.84**	**0.82**	**0.85**	0.77	0.76	0.75
IF_A-K_A	0.26	0.27	0.27	0.35	0.32	0.30
$IF_{A'}$-$P_{A'}$	**0.91**	**0.91**	**0.92**	**0.85**	**0.84**	**0.83**
$IF_{A'}$-$N_{A'}$	**0.88**	**0.88**	**0.90**	**0.84**	**0.82**	**0.81**
$IF_{A'}$-$B_{A'}$	0.75	0.75	0.78	0.71	0.63	0.62
$IF_{A'}$-$K_{A'}$	0.71	0.71	0.73	*0.51*	*0.49*	*0.50*
IF_Q-P_Q	**0.87**	**0.87**	**0.88**	0.75	**0.81**	**0.80**

Table 5: Spearman's correlation coefficient (S.C.C.) between the proposed approach IF and traditional graph-based methods.

5 Conclusions and Future Work

In this article, we address the AKE task via the distribution-based modeling of the adjacency matrix that corresponds to various versions of the graph-of-words for a target document. More specifically, we propose capitalizing on unsupervised learning algorithms for the distribution-based modeling and scoring of the candidate words. Based on our performance evaluation, the IF approach shows the best effectiveness results in almost all datasets, concerning the F_1 score determining the number of keywords in document level.

There are many interesting future research directions, such as i) improving the scoring functions of the unsupervised learning approaches used in the context of the keyword extraction task, ii) adapting the proposed approach to the keyphrase extraction task , iii) developing novel distribution-based modeling methods that simultaneously utilize the information from one/multiple adjacency matrices , and iv) applying the adjacency matrix's distribution-based modeling in other tasks where only graph-based methods are used to date.

References

Vladimir Batagelj and Matjaz Zaversnik. 2011. Fast algorithms for determining (generalized) core groups in social networks. *Adv. Data Anal. Classif.*, 5(2):129–145.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learn-*

ing, pages 221–229, Brussels, Belgium. Association for Computational Linguistics.

Saroj K. Biswas, Monali Bordoloi, and Jacob Shreya. 2018. A graph based keyword extraction model using collective node weight. *Expert Syst. Appl.*, 97:51–59.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Florian Boudin. 2016. pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 69–73, Osaka, Japan. The COLING 2016 Organizing Committee.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Networks*, 30(1-7):107–117.

Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446, Doha, Qatar. Association for Computational Linguistics.

Hou Pong Chan, Wang Chen, Lu Wang, and Irwin King. 2019. Neural keyphrase generation via reinforcement learning with adaptive rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2163–2174, Florence, Italy. Association for Computational Linguistics.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1095–1105, Online. Association for Computational Linguistics.

Jonathan Cohen. 2008. Trusses: Cohesive subgraphs for social network analysis. *National security agency technical report*, 16:3–29.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Mach. Learn.*, 20(3):273–297.

Corina Florescu and Cornelia Caragea. 2017. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics.

Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1629–1635. AAAI Press.

Sujatha Das Gollapalli, Xiaoli Li, and Peng Yang. 2017. Incorporating expert knowledge into keyphrase extraction. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3180–3187. AAAI Press.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, Baltimore, Maryland. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden. Association for Computational Linguistics.

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632.

Mikalai Krapivin, Aliaksandr Autayeu, and Maurizio Marchese. 2008. Large dataset for keyphrases extraction. In *Technical Report DISI-09-055*. Trento, Italy.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pages 413–422. IEEE Computer Society.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2012. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data*, 6(1):3:1–3:39.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, Volume 2 (Short Papers), pages 634–639, New Orleans, Louisiana. Association for Computational Linguistics.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Vancouver, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 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*.

Andriy Mnih and Geoffrey E. Hinton. 2007. Three new graphical models for statistical language modelling. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 641–648. ACM.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers, 10th International Conference on Asian Digital Libraries, ICADL 2007, Hanoi, Vietnam, December 10-13, 2007, Proceedings*, volume 4822 of *Lecture Notes in Computer Science*, pages 317–326. Springer.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 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)*, pages 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Eirini Papagiannopoulou and Grigorios Tsoumakas. 2018. Local word vectors guiding keyphrase extraction. *Inf. Process. Manag.*, 54(6):888–902.

Eirini Papagiannopoulou and Grigorios Tsoumakas. 2020. A review of keyphrase extraction. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 10(2).

Eirini Papagiannopoulou, Grigorios Tsoumakas, and Apostolos N Papadopoulos. 2020. Keywords lie far from the mean of all words in local vector space. *arXiv preprint arXiv:2008.09513*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Bruno R. Preiss. 2000. *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. John Wiley & Sons Incorporated.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20.

François Rousseau and Michalis Vazirgiannis. 2015. Main core retention on graph-of-words for single-document keyword extraction. In *Advances in Information Retrieval - 37th European Conference on IR Research, ECIR 2015, Vienna, Austria, March 29 - April 2, 2015. Proceedings*, volume 9022 of *Lecture Notes in Computer Science*, pages 382–393.

Bernhard Schölkopf, Robert C. Williamson, Alexander J. Smola, John Shawe-Taylor, and John C. Platt. 1999. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 582–588. The MIT Press.

Stephen B Seidman. 1983. Network structure and minimum degree. *Social networks*, 5(3):269–287.

David M. J. Tax and Robert P. W. Duin. 1999a. Data domain description using support vectors. In *ESANN 1999, 7th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 21-23, 1999, Proceedings*, pages 251–256.

David M. J. Tax and Robert P. W. Duin. 1999b. Support vector domain description. *Pattern Recognit. Lett.*, 20(11-13):1191–1199.

Antoine Tixier, Fragkiskos Malliaros, and Michalis Vazirgiannis. 2016. A graph degeneracy-based approach to keyword extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1860–1870, Austin, Texas. Association for Computational Linguistics.

Didier Augusto Vega-Oliveros, Pedro Spoljaric Gomes, Evangelos E. Milios, and Lilian Berton. 2019. A multi-centrality index for graph-based keyword extraction. *Inf. Process. Manag.*, 56(6).

Xiaojun Wan and Jianguo Xiao. 2008a. CollabRank: Towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 969–976, Manchester, UK. Coling 2008 Organizing Committee.

Xiaojun Wan and Jianguo Xiao. 2008b. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, pages 855–860. AAAI Press.

Liang Wang and Sujian Li. 2017. PKU_ICL at SemEval-2017 task 10: Keyphrase extraction with model ensemble and external knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 934–937, Vancouver, Canada. Association for Computational Linguistics.

Rui Wang, Wei Liu, and Chris McDonald. 2015. Using word embeddings to enhance keyword identification for scientific publications. In *Databases Theory and Applications - 26th Australasian Database Conference, ADC 2015, Melbourne, VIC, Australia, June 4-7, 2015. Proceedings*, volume 9093 of *Lecture Notes in Computer Science*, pages 257–268. Springer.

Yue Wang, Jing Li, Hou Pong Chan, Irwin King, Michael R. Lyu, and Shuming Shi. 2019. Topic-aware neural keyphrase generation for social media language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2516–2526, Florence, Italy. Association for Computational Linguistics.

Hai Ye and Lu Wang. 2018. Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4142–4153, Brussels, Belgium. Association for Computational Linguistics.

Jing Zhao and Yuxiang Zhang. 2019. Incorporating linguistic constraints into keyphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5224–5233, Florence, Italy. Association for Computational Linguistics.