Main Core Retention on Graph-of-Words for Single-Document Keyword Extraction

François Rousseau and Michalis Vazirgiannis

LIX, École Polytechnique, France

Abstract. In this paper, we apply the concept of *k-core* on the *graph-of-words* representation of text for single-document keyword extraction, retaining only the nodes from the main core as representative terms. This approach takes better into account proximity between keywords and variability in the number of extracted keywords through the selection of more *cohesive* subsets of nodes than with existing graph-based approaches solely based on *centrality*. Experiments on two standard datasets show statistically significant improvements in F1-score and AUC of precision/recall curve compared to baseline results, in particular when weighting the edges of the graph with the number of co-occurrences. To the best of our knowledge, this is the first application of graph degeneracy to natural language processing and information retrieval.

Keywords: single-document keyword extraction, graph representation of text, weighted graph-of-words, k-core decomposition, degeneracy.

1 Introduction

Keywords have become ubiquitous in our everyday life, from looking up information on the Web via a search engine bar to online ads matching the content we are currently browsing. Researchers use them when they write a paper for better indexing as well as when they consult or review one to get a gist of its content before reading it. Traditionally, keywords have been manually chosen by the authors but the explosion of the number of available textual contents made the process too time-consuming and costly. Keyword extraction as an automated process then naturally emerged as a research issue to satisfy that need.

A graph-of-words is a syntactic graph that encodes co-occurrences of terms as opposed to the traditional bag-of-words and state-of-the-art approaches in keyword extraction proposed to apply PageRank and HITS on it to extract its most salient nodes. In our work, we capitalize on the k-core concept to propose a novel approach that takes better into account proximity between keywords and variability in the number of extracted keywords through the selection of more cohesive subsets of vertices. The proposed approach presents some significant advantages: (1) it is totally *unsupervised* as it does not need any training corpus; (2) it is *corpus-independent* as it does not rely on any collection-wide statistics such as IDF and thus can be applied on any text out of the box; (3) it *scales* to any document length since the algorithm is linearithmic in the number of unique

A. Hanbury et al. (Eds.): ECIR 2015, LNCS 9022, pp. 382–393, 2015.

terms as opposed to more complex community detection techniques and (4) the method in itself is *parameter-free* as the number of extracted keywords adapts to the structure of each graph through the k-core principle.

The rest of the paper is organized as follows. Section 2 provides a review of the related work. Section 3 defines the preliminary concepts upon which our work is built. Section 4 introduces the proposed approach and compares it with existing graph-based methods. Section 5 describes the experimental settings and presents the results we obtained on two standard datasets. Finally, Section 6 concludes our paper and mentions future work directions.

2 Related Work

In this section, we present the related work published in the areas of *keyword extraction* and *graph representation of text*. Mihalcea and Tarau in [19] and Litvak and Last in [16] are perhaps the closest works to ours since they also represent a text as a graph-of-words and extract the most salient keywords using a graph mining technique, solely based on centrality unlike k-core though.

2.1 Keyword Extraction

In the relevant literature, keyword extraction is closely related to text summarization. Indeed, Luhn in [17], which is one of the earliest works in automatic summarization, capitalizes on the *term frequency* to first extract the most salient keywords before using them to detect sentences. Later, the research community turned the task into a *supervised learning problem* with the seminal works of Turney in [24] based on genetic algorithms and of Witten *et al.* in [26] based on Naive Bayes. We refer to the survey of Nenkova and McKeown in [20] for an in-depth review on automatic summarization and by extension on keyword extraction. Briefly, the published works make several distinctions for the general task of keyword extraction: (a) *single-* [11,16,19] vs. *multi-document* [18] depending on whether the input is from a single document or multiple ones (e.g., a stream of news), (b) *extractive* [11,16,19] vs. *abstractive* [4] depending on whether the extracted content is restricted to the original text or not (e.g., use of a thesaurus to enrich the keywords), (c) *generic* [11,16,19,24] vs. *query-based* [25] vs. *update* [12] depending on whether the extracted keywords are generic or biased towards a specific need (e.g., expressed through a query) or dependent of already-known information (e.g., browsing history) and finally (d) *unsupervised* [7,16,19] vs. *supervised* [11,16,24] depending on whether the extraction process involves a training part on some labeled inputs. Our work falls within the case of *unsupervised generic extractive single-document keyword extraction.*

2.2 Graph Representation of Text

Graph representations of textual documents have been around for a decade or so and we refer to the work of Blanco and Lioma in [3] for an in-depth review. These representations have been mainly investigated as a way of taking into account *term dependence* and *term order* compared to earlier approaches that did not. In NLP, text has historically been represented as a *bag-of-words*, i.e. a multiset of terms that assumes independence between terms and the task of keyword extraction was no exception to that representation. But graph structures allow to challenge that *term independence assumption* and somewhat recently Rousseau and Vazirgiannis introduced in [22] the denomination of *graph-of-words* to encompass that idea of using a graph whose vertices represent unique term and whose edges represent some meaningful relation between pairs of terms. This relation can either be based solely on statistics or use deeper linguistic analysis, leading respectively to *syntactic* [16,19] and *semantic* [15] graphs. More generally, what a vertex of the graph represents depends entirely on the level of granularity needed. This can be a sentence [7,19], a word [3,16,19,22] or even a character [9]. Most of the existing works based on graph-of-words were explored for automatic summarization, in particular the seminal works of Erkan and Radev in [7] and Mihalcea and Tarau in [19]. More recent papers [3,22] proposed applications in ad hoc IR to challenge the well-established tf-based retrieval models.

3 Preliminary Concepts

In this section, we define the preliminary concepts upon which our work is built: the notions of graph, k-core and graph-of-words.

3.1 Graph

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a *graph* (also known as a network), $\mathcal{V}$ its set of *vertices* (also known as nodes) and $\mathcal{E}$ its set of *edges* (also known as arcs or links). We denote by n the number of vertices ($n = |\mathcal{V}|$) and m the number of edges ($m = |\mathcal{E}|$). A graph can represent anything, from a protein-interaction network to a power grid or in our case a textual document. This is the natural representation to model interactions between entities and we believe text makes no exception.

Depending on the nature of these interactions, an edge and by extension a graph can be weighted and/or directed. This impacts the definition of the *degree* of a vertex, which measures the interactions a node has with its neighbors and somewhat its importance or influence in the network. We denote by $deg_{\mathcal{G}}(v)$ the *degree* of a vertex $v \in \mathcal{G}$ in $\mathcal{G}$. In the undirected case, this corresponds to the sum of the weights of the adjacent edges (unit weight in the unweighted case). In the directed case, the notion of degree is usually split in two: *indegree* and *outdegree* corresponding to the (weighted) number of inlinks and outgoing links.

3.2 K-core

The idea of a *k-degenerate* graph comes from the work of Bollobás in [5, page 222] that was further extended by Seidman in [23] into the notion of a *k-core*, which explains the use of *degeneracy* as an alternative denomination in the literature. Henceforth, we will be using the two terms interchangeably. Let k be an integer. A subgraph $\mathcal{H}_k = (\mathcal{V}', \mathcal{E}')$, induced by the subset of vertices $\mathcal{V}' \subseteq \mathcal{V}$ (and a fortiori by the subset of edges $\mathcal{E}' \subseteq \mathcal{E}$), is called a *k-core* or a *core of order k* iff $\forall v \in \mathcal{V}', \ deg_{\mathcal{H}_k}(v) \geq k$ and $\mathcal{H}_k$ is the maximal subgraph with this property, i.e. it cannot be augmented without losing this property. In other words, the k-core of a graph corresponds to the maximal connected subgraph whose vertices are at least of degree k within the subgraph.

The *core number* core(v) of a vertex v is the highest order of a core that contains this vertex. The core of maximum order is called the *main core* and the set of all the k-cores of a graph (from the 0-core to the main core) forms what is called the *k-core decomposition* of a graph.

Thanks to Batagelj and Zaveršnik in [2], the k-core decomposition of a weighted graph can be computed in linearithmic time ($\mathcal{O}(n + m \log n)$) and linear space ($\mathcal{O}(n)$) using a *min-oriented binary heap* to retrieve the vertex of lowest degree at each iteration (n in total). We implemented their algorithm in our experiments. Note that in the unweighted case, there exists a linear version in time that uses *bin sort* since there are at most $\Delta(\mathcal{G}) + 1$ distinct values for the degrees where $\Delta(\mathcal{G}) = max_{v \in \mathcal{V}}(deg_{\mathcal{G}}(v)) = \mathcal{O}(n)$. For the directed case, Giatsidis *et al.* in [10] proposed a two-dimensional k-core decomposition that is beyond the scope of this paper. We still explored the effects of degeneracy on directed graphs in our experiments, considering either the indegree or the outdegree instead of the degree but not both of them at the same time.

3.3 Graph-of-words

We model a textual document as a *graph-of-words*, which corresponds to a graph whose vertices represent unique terms of the document and whose edges represent co-occurrences between the terms within a fixed-size sliding window. The underlying assumption is that all the words present in a document have some relationships with the others, modulo a window size outside of which the relationship is not taken into consideration. This is a statistical approach as it links all co-occurring terms without considering their meaning or function in the text. The graph can be weighted to take into account the number of co-occurrences of two terms. Similarly, the graph can be directed to encode the *term order*, forward edges indicating the natural flow of the text.

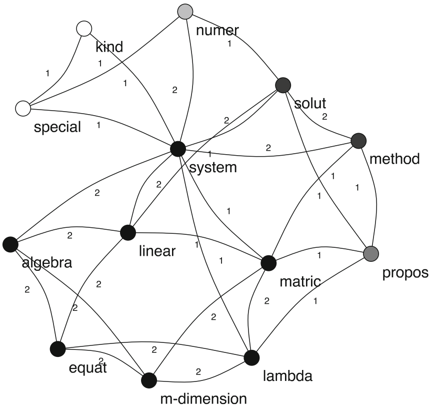
Regarding the preprocessing steps, we applied the following on the input text: (1) tokenization; (2) part-of-speech[1] annotation and selection (nouns and adjectives like in [19]); (3) stopwords[2] removal; and (4) stemming[3]. All the remaining terms constitute the vertices of the graph-of-words. The edges were drawn between terms co-occurring within a fixed-size sliding window W of size 4 over the processed text, value consistently reported as working well [3,7,16,19,22]. For the whole process, the complexity is $\mathcal{O}(nW)$ in time and $\mathcal{O}(n + m)$ in space.

Figure 1a illustrates the weighted graph-of-words representation of one of the documents (id 1938) from the dataset introduced by Hulth in [11]. Edge weight

[1] http://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html

[2] http://jmlr.org/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

[3] http://tartarus.org/~martin/PorterStemmer



(a) Graph-of-words representation

WK-core		PageRank		HITS	
system	6	**system**	1.93	**system**	0.45
matric	6	**matric**	1.27	**matric**	0.38
lambda	6	solut	1.10	**linear**	0.32
linear	6	**lambda**	1.08	**lambda**	0.31
equat	6	**linear**	1.08	solut	0.30
algebra	6	**equat**	0.90	method	0.28
m-dim...	6	**algebra**	0.90	propos	0.25
method	5	**m-dim...**	0.90	**algebra**	0.25
solut	5	propos	0.89	**m-dim...**	0.23
propos	4	method	0.88	**equat**	0.22
numer	3	special	0.78	**numer**	0.18
specia	2	**numer**	0.74	special	0.15
kind	2	kind	0.55	kind	0.12

(b) Ranked list of scored keywords

Fig. 1. Subfigure (a) illustrates a graph-of-words representation of a textual document. Edge weight corresponds to the number of co-occurrence, node color to the node core number (gray scale). Table (b) shows the ranked lists of scored keywords extracted with weighted k-core, PageRank and HITS. Bold font indicates the golden keywords and the dashed lines the cutoff for each method.

corresponds to the number of co-occurrences. Node color indicates the highest core a vertex belongs to, from 2-core (white) to 6-core (black).

4 Graph-based Keyword Extraction

In this section, we present the existing state-of-the-art graph-based methods for keyword extraction as well as our proposed approach.

4.1 Existing Approaches: PageRank and HITS on Graph-of-words

There exist two algorithms that have been successfully used for graph-based keyword extraction: PageRank and HITS, considered in this context first in [19] and [16] respectively. Both methods are based on eigenvector centrality and define recursively the weight of a vertex as a measure of its *influence* inside the network, regardless of how cohesive its neighborhood is. For PageRank, it is defined as the sum of the weights of its incoming neighbors (the ones giving it support) and the vertex itself gives a weighted portion of its own weight to each of its outgoing neighbors. For HITS, it is slightly different as it defines two types of influential nodes: the *authorities* that are being pointed at by a lot of nodes and the *hubs* that are pointing to a lot of nodes (these are the same in the undirected case).

4.2 Our Contribution: k-core Decomposition on Graph-of-words

Our idea was to consider the vertices of the main core as the set of keywords to extract from the document. Indeed, it corresponds to the most cohesive connected component(s) of the graph and thus its vertices are intuitively good candidates for representing the entire graph-of-words. Additionally, assuming a set of golden keywords to compare with, we considered more and more cores in the decomposition and expected an increase in the recall of the extracted keywords without a too important decrease in precision.

We illustrate in Table 1b the process by presenting the list of keywords extracted using all three algorithms on the graph-of-words presented in Figure 1a. The horizontal dashed lines indicate where the cutoff is applied for each method and in bold the golden keywords according to human annotators. For our approach, it is the main core (of order 6 in this example). For PageRank and HITS, Mihalcea and Tarau suggested extracting the top third of the vertices (top 33%), relative numbers helping accounting for documents of varying length.

Overall, we notice that "numer" is never extracted by any method. The term appears only once in the original text and around terms of lesser importance (except "system"). Extracting it can be considered as a form of *overfitting* if we assume that the pattern to extract is related to term repetition, standard assumption in NLP and IR. Both PageRank and HITS retrieve "solut", which is not a golden keyword, because of its centrality but not k-core because of its neighbors. Again, the main core corresponds to *cohesive set(s) of vertices* in which they all contribute equally to the subgraph they belong to – removing any node would collapse the entire subgraph through the *cascading effect* implied by the k-core condition. PageRank and HITS, on the other hand, provide scores for each vertex based on its centrality yet somewhat independently of its neighborhood, therefore not capturing the proximity between keywords.

4.3 Keywords are Bigrams

For the 500 abstracts from the *Inspec* database that we used in our experiments, only 662 out of the 4,913 keywords manually assigned by human annotators are unigrams (13%). The rest of them range from bigrams (2,587 – 52%) to 7-grams (5). Similar statistics were observed on the other dataset. Because higher order n-grams can be considered as multiple bigrams, we make the general claim that human annotators tend to select *keywords that are bigrams*. Thus, to improve the performances of an automated system, one needs to capture the interactions between keywords in the first place – hence, the explored graph-of-words representation to challenge the traditional bag-of-words.

Even if both the existing models and our approach extract unigrams because of the way the graph-of-words is constructed, the edges do represent co-occurrences within a sliding window. And for small-enough sizes (which is typically the case in practice), we can consider that two linked vertices represent a long-distance bigram [1], if not a bigram. Hence, by considering cohesive subgraphs, we make sure to extract unigrams that co-occur together and thus are bigrams, if not higher order n-grams. On the contrary, PageRank and HITS may extract unigrams that are central because they co-occur with a lot of other words but these words may not be extracted as well because of a lower weight. Extracting salient bigrams would require to include bigrams as vertices but the number of nodes increases exponentially with the order of the n-gram.

4.4 K-cores are Adaptive

Most current techniques in keyword extraction assign a score to each term of the document and then take the top ones. For a given collection of homogeneous documents in size or because of specific constraints, an *absolute* number may make sense. For example, Turney in [24] limited to the top 5 keywords while Witten *et al.* in [26] to the top 15. Mihalcea and Tarau argued in [19] that a *relative* number should be used for documents of varying length or when no prior is known. We claim that the numbers of retrieved keywords should be decided at the document level and not at the collection level. For instance, for two documents, even of equal size, one of them might require more keywords to express the gist of its content (because it deals with more topics for example).

The size of each core, i.e. the number of vertices in the subgraph, depends on the structure of the graph. In the unweighted case, it is lower-bounded by $k + 1$ since each vertex has at least k neighbors but can potentially be up to n in the case of a complete graph. Hence, we think that degeneracy can capture this variability in the number of extracted keywords, in particular for a fixed document length (PageRank and HITS still extract more and more keywords as the document length increases when using relative numbers). In Section 5, we will show distributions of extracted keywords per document length for all models and from human annotators to support our claim.

5 Experiments

In this section, we describe the experiments we conducted to test and validate our approach along with the results we obtained.

5.1 Datasets

We used two standard datasets publicly available[4]: (1) *Hulth2003* – 500 abstracts from the *Inspec* database introduced by Hulth in [11] and also used by Mihalcea and Tarau in [19] with PageRank; and (2) *Krapi2009* – 2,304 ACM full papers (references and captions excluded) introduced by Krapivin *et al.* in [14]. For Hulth2003, we used the “uncontrolled” golden keywords since we do not want to restrict the keywords to a given thesaurus and for Krapi2009, we used the ones chosen by the authors of each ACM paper. Since all approaches are unsupervised and single-document, the scalability of the methods are measured with regards to the document length, not the collection size (that only needs to be large enough to measure the statistical significance of the improvements).

[4] https://github.com/snkim/AutomaticKeyphraseExtraction

5.2 Models

For the graph-of-words representation, we experimented with undirected, forward edges (natural flow of the text – an edge $term_1 \rightarrow term_2$ meaning that $term_1$ precedes $term_2$ in a sliding window) and backward edges (the opposite). In terms of keyword-extracting methods, we considered (1) PageRank, (2) HITS, (3) k-core on an unweighted graph-of-words and (4) k-core on a weighted one (the edge weight being the number of co-occurrence). We extracted the top third keywords (top 33%) on Hulth2003 and the top 15 keywords on Krapi2009 for PageRank and HITS and the main core for our approaches (the k values differs from document to document). The choice between relative (top X%) and absolute numbers (top X) comes from the fact that for relatively short documents such as abstracts, the longer the document, the more keyword human annotators tend to select while past a certain length (10-page long for ACM full papers), the numbers vary far less. Hence, in all fairness to the baselines, we selected the top 33% on the abstracts like in the original papers and the top 15 for the full papers (15 being the average number of unigrams selected as keywords by the papers' authors). Note that for HITS, we only display the results for the authority scores since the hub scores are the same in the undirected case and symmetric in the directed case (the hub score for forward edges corresponds to the authority score for backward edges).

5.3 Evaluation

For each document, we have a set of golden keywords manually assigned by human annotators and a set of extracted keywords, leading to *precision*, *recall* and *F1-score* per document and per method that are then *macro-averaged* at the collection level. The statistical significance of improvement over the PageRank baseline for each metric was assessed using the Student's paired t-test, considering two-sided p-values less than 0.05 to reject the null hypothesis.

Note that we convert the golden keywords into unigrams to easily compute precision and recall between this golden set and the set of extracted unigrams. Mihalcea and Tarau in [19] suggested to "reconcile" the n-grams as a post-processing step by looking in the original text for adjacent unigrams but then questions arise such as whether you keep the original unigrams in the final set, impacting the precision and recall – hence an evaluation based on unigrams. Indeed, it is not clear how to penalize a method that, given a golden bigram to extract, would return part of it (unigram) or more than it (trigram).

5.4 Macro-averaged Results

We present in Table 1 the macro-averaged precision, recall and F1-score (in %) for PageRank, HITS, k-core and weighted k-core (columns) for the different variants of graph-of-words considered (rows) on each dataset. Overall, PageRank and HITS have similar results, with a precision higher than the recall as reported in previous works. It is the opposite for k-core, which tends to extract a main core

Table 1. Macro-averaged precision, recall and F1-score for PageRank, HITS, K-core and Weighted K-core (WK-core). Bold font marks the best performance in a block of a row. * indicates statistical significance at $p < 0.05$ using the Student's t-test w.r.t. the PageRank baseline of the same block of the same row.

Graph	Dataset	Macro-averaged precision (%)				Macro-averaged recall (%)				Macro-averaged F1-score (%)			
		PageRank	HITS	K-core	WK-core	PageRank	HITS	K-core	WK-core	PageRank	HITS	K-core	WK-core
undirected edges	Hulth2003	58.94	57.86	46.52	**61.24***	42.19	41.80	**62.51***	50.32*	47.32	46.62	49.06*	**51.92***
	Krapi2009	50.23	49.47	40.46	**53.47***	48.78	47.85	**78.36***	50.21	49.59	47.96	46.61	**50.77***
forward edges	Hulth2003	55.80	54.75	42.45	**56.99***	41.98	40.43	**72.87***	46.93*	45.70	45.03	**51.65***	50.59*
	Krapi2009	47.78	47.03	39.82	**52.19***	44.91	44.19	**79.06***	45.67	45.72	44.95	46.03	**47.01***
backward edges	Hulth2003	59.27	56.41	40.89	**60.24***	42.67	40.66	**70.57***	49.91*	47.57	45.37	45.20	**50.03***
	Krapi2009	51.43	49.11	39.17	**52.14***	49.96	47.00	**77.60***	50.16	**50.51**	47.38	46.93	50.42

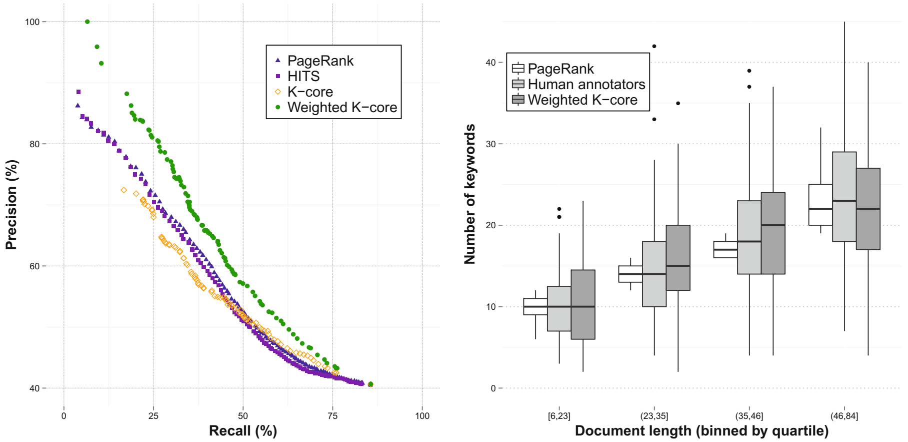
with a lot of vertices since the k-core condition can be interpreted as a set of keywords that co-occur with at least k other keywords. For the weighted case, it corresponds to a set of keywords that co-occur at least k times in total with other keywords leading to cores with fewer vertices but with stronger links and the extraction of important bigrams, hence the increase in precision (at the cost of a decrease in recall) and an overall better F1-score.

Edge direction has an impact but not necessarily a significant one and is different across methods. This disparity in the results and the lack of a dominant choice for edge direction is consistent with the relevant literature. Mihalcea and Tarau in [19] and Blanco and Lioma in [3] used undirected edges, Litvak and Last in [16] backward edges and Rousseau and Vazirgiannis in [22] forward edges. Hence, we recommend the use of undirected edges for ease of implementation but other techniques that would try to extract paths from the graph-of-words for instance might need the edge direction to follow the natural flow of the text like for multi-sentence compression [8].

5.5 Precision/Recall Curves

Additionally, instead of just considering the main core or the top X% vertices, we computed precision and recall at each core and at each percent of the total number of terms to get *precision/recall curves*. We used relative numbers because the documents are of varying length so the top 10 keywords for a document of size 10 and 100 do not mean the same while the top 10% might.

We show on Figure 2a the resulting curves on the Hulth2003 dataset, one for each model (100 points per curve, no linear interpolation following Davis and Goadrich in [6]). The final recall for all models is not 100% because human annotators used keywords that do not appear in the original texts. We observe that the curve for the weighted k-core (green, solid circle) is systematically above the others, thus showing improvements in *Area Under the Curve* (AUC) and not just in point estimates such as the F1-score. The curve for k-core (orange, diamond) is overall below the other curves since it tends to only find a few cores with a lot of vertices, lowering the precision but insuring some minimum recall (its lowest value of recall is greater than for the other curves).



(a) Precision/recall curves

(b) Distribution of the number of keywords

Fig. 2. Weighted k-core on graph-of-words consistently captures more golden keywords than PageRank and HITS (subfigure a) and provides a variability in the number of extracted keywords closer to the human one (subfigure b)

5.6 Distribution of the Number of Keywords

Human annotators do not assign the same number of keywords to all documents. There is a variability that is partially due to varying document length (the number increases with the length) but not only. Indeed, when computing a distribution per document length, we can still observe some dispersion. With PageRank, by extracting a relative number of unigrams (top 33%), one accounts for varying length but does not fully capture the variability introduced by human annotators while k-core does better. Similarly, for Krapi2009, where documents are of the same length (10-page), some authors may have chosen more keywords for their paper than others because for instance there are alternative denominations for the concept(s) developed in their work and as a result more keywords.

We present in Figure 2b above groups of three box plots computed for Hulth2003. In each group, the left one corresponds to PageRank (in white, similar results for HITS), the middle one to human annotators (light gray) and the right one to weighted k-core (dark gray). We do not show any box plot for unweighted k-core since the method tends to overestimate the number of keywords (higher recall, lower precision). For space constraints and also for sparsity reasons, we binned the document lengths by quartile (i.e. the bins are not of equal range but contains the same number of documents – 25% each).

As expected, the number of keywords for a given bin varies little for PageRank – the increase in median value across the bins being due to the baseline taking the top third unigrams, relative number that increases with the document length. For weighted k-core, we observe that the variability is taken much more into account: the first, second and third quartiles' values for the number of keywords are much closer to the golden ones. For PageRank and HITS, it would be much harder to learn the number of keywords to extract for each document while it is inherent to graph degeneracy. Alone, these results would not mean much but because higher accuracy has already been established through consistent and significant higher macro-averaged F1-scores, they support our claim that k-core is better suited for the task of keyword extraction because of its adaptability to the graph structure and therefore to the document structure.

6 Conclusions and Future Work

In this paper, we explored the effects of k-core on graph-of-words for single-document keyword extraction. Similarly to previous approaches, we capitalized on syntactic graph representations of text to extract central terms. However, by retaining only the main core of the graph, we were able to capture more cohesive subgraphs of vertices that are not only central but also densely connected. Hence, the extracted keywords are more likely to form bigrams and their number adapts to the graph structure, as human annotators tend to do when assigning keywords to the corresponding document.

As a final example, here are the stemmed unigrams belonging to the main core of the graph-of-words corresponding to this paper (references, captions and this paragraph excluded): {*keyword, extract, graph, represent, text, weight, graph-of-word, k-core, degeneraci, edg, vertic, number, document*}. Using PageRank, "work" appears in the top 5, "term" and "pagerank" in the top 10, and "case" and "order" in the top 15. Existing methods tend to extract central keywords that are not necessarily part of a cohesive subgraph as opposed to our proposed approach, which provides closer results to what humans do on several aspects.

Possible extension of this work would be the exploration of the clusters of keywords in the top cores to elect representatives per cluster for topic modeling.

References

1. Bassiou, N., Kotropoulos, C.: Word clustering using PLSA enhanced with long distance bigrams. In: Proceedings of ICPR 2010, pp. 4226–4229 (2010)
2. Batagelj, V., Zavernik, M.: Fast algorithms for determining core groups in social networks. Advances in Data Analysis and Classification 5(2), 129–145 (2011)
3. Blanco, R., Lioma, C.: Graph-based term weighting for information retrieval. Information Retrieval 15(1), 54–92 (2012)
4. Blank, I., Rokach, L., Shani, G.: Leveraging the citation graph to recommend keywords. In: Proceedings of RecSys 2013, pp. 359–362 (2013)
5. Bollobs, B.: Extremal Graph Theory. Academic Press, London (1978)
6. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: Proceedings of ICML 2006, pp. 233–240 (2006)
7. Erkan, G., Radev, D.R.: LexRank: graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research 22(1), 457–479 (2004)
8. Filippova, K.: Multi-sentence compression: finding shortest paths in word graph. In: Proceedings of COLING 2010, pp. 322–330 (2010)

9. Giannakopoulos, G., Karkaletsis, V., Vouros, G., Stamatopoulos, P.: Summarization system evaluation revisited: N-gram graphs. ACM Transactions on Speech and Language Processing 5(3), 1–39 (2008)
10. Giatsidis, C., Thilikos, D.M., Vazirgiannis, M.: D-cores: Measuring collaboration of directed graphs based on degeneracy. In: Proceedings of ICDM 2011, pp. 201–210 (2011)
11. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of EMNLP 2003, pp. 216–223 (2003)
12. Karkali, M., Plachouras, V., Stefanatos, C., Vazirgiannis, M.: Keeping keywords fresh: A BM25 variation for personalized keyword extraction. In: Proceedings of TempWeb 2012, pp. 17–24 (2012)
13. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
14. Krapivin, M., Autaeu, A., Marchese, M.: Large dataset for keyphrases extraction. Technical Report DISI-09-055, University of Trento (May 2009)
15. Leskovec, J., Grobelnik, M., Milic-Frayling, N.: Learning semantic graph mapping for document summarization. In: Proceedings of KDO 2004 (2004)
16. Litvak, M., Last, M.: Graph-based keyword extraction for single-document summarization. In: Proceedings of MMIES 2008, pp. 17–24 (2008)
17. Luhn, H.P.: The automatic creation of literature abstracts. IBM Journal of Research and Development 2(2), 159–165 (1958)
18. McKeown, K., Passonneau, R.J., Elson, D.K., Nenkova, A., Hirschberg, J.: Do summaries help. In: Proceedings of SIGIR 2005, pp. 210–217 (2005)
19. Mihalcea, R., Tarau, P.: TextRank: bringing order into texts. In: Proceedings of EMNLP 2004, pp. 404–411 (2004)
20. Nenkova, A., McKeown, K.R.: Automatic summarization. Foundations and Trends in Information Retrieval 5(2-3), 103–233 (2011)
21. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web. Technical Report 1999-0120, Stanford University (1999)
22. Rousseau, F., Vazirgiannis, M.: Graph-of-word and TW-IDF: new approach to ad hoc IR. In: Proceedings of CIKM 2013, pp. 59–68 (2013)
23. Seidman, S.B.: Network structure and minimum degree. Social Networks 5, 269–287 (1983)
24. Turney, P.D.: Learning to extract keyphrases from text. Technical report, National Research Council of Canada, Institute for Information Technology (1999)
25. Turpin, A., Tsegay, Y., Hawking, D., Williams, H.E.: Fast generation of result snippets in web search. In: Proceedings of SIGIR 2007, pp. 127–134 (2007)
26. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: KEA: practical automatic keyphrase extraction. In: Proceedings of DL 1999, pp. 254–255 (1999)