A Graph Degeneracy-based Approach to Keyword Extraction *

Antoine J.-P. Tixier[1], **Fragkiskos D. Malliaros**[1,2], **Michalis Vazirgiannis**[1]
[1]Computer Science Laboratory, École Polytechnique, Palaiseau, France
[2]Department of Computer Science and Engineering, UC San Diego, La Jolla, CA, USA
{anti5662,fmalliaros,mvazirg}@lix.polytechnique.fr

Abstract

We operate a change of paradigm and hypothesize that keywords are more likely to be found among *influential* nodes of a graph-of-words rather than among its nodes high on *eigenvector*-related centrality measures. To test this hypothesis, we introduce unsupervised techniques that capitalize on *graph degeneracy*. Our methods strongly and significantly outperform all baselines on two datasets (short and medium size documents), and reach best performance on the third one (long documents).

1 Introduction

Keyword extraction is a central task in NLP. It finds applications from information retrieval (notably web search) to text classification, summarization, and visualization. In this study, we focus on the task of *unsupervised single-document keyword extraction*. Following (Mihalcea and Tarau, 2004), we concentrate on *keywords* only, letting the task of *keyphrase* reconstruction as a post-processing step.

More precisely, while we capitalize on a graph representation of text like several previous approaches, we deviate from them by making the assumption that keywords are not found among *prestigious* nodes (or more generally, nodes high on eigenvector-related centrality metrics), but rather among *influential* nodes. Those nodes may not have many important connections (like their prestigious counterparts), but they are ideally placed at the core of the network. In other words, this switches the objective from capturing the *quality* and *quantity* of single node connections, to taking into account the *density* and *cohesiveness* of groups of nodes. To operate this change of paradigm, we propose several algorithms that leverage the concept of *graph degeneracy* (Malliaros et al., 2016a).

Our contributions are threefold: (1) we propose new unsupervised keyword extraction techniques that reach state-of-the art performance, (2) we apply the K-truss algorithm to the task of keyword extraction for the first time, and (3) we report new insights on the interplay between window size, graph-of-words structure, and performance.

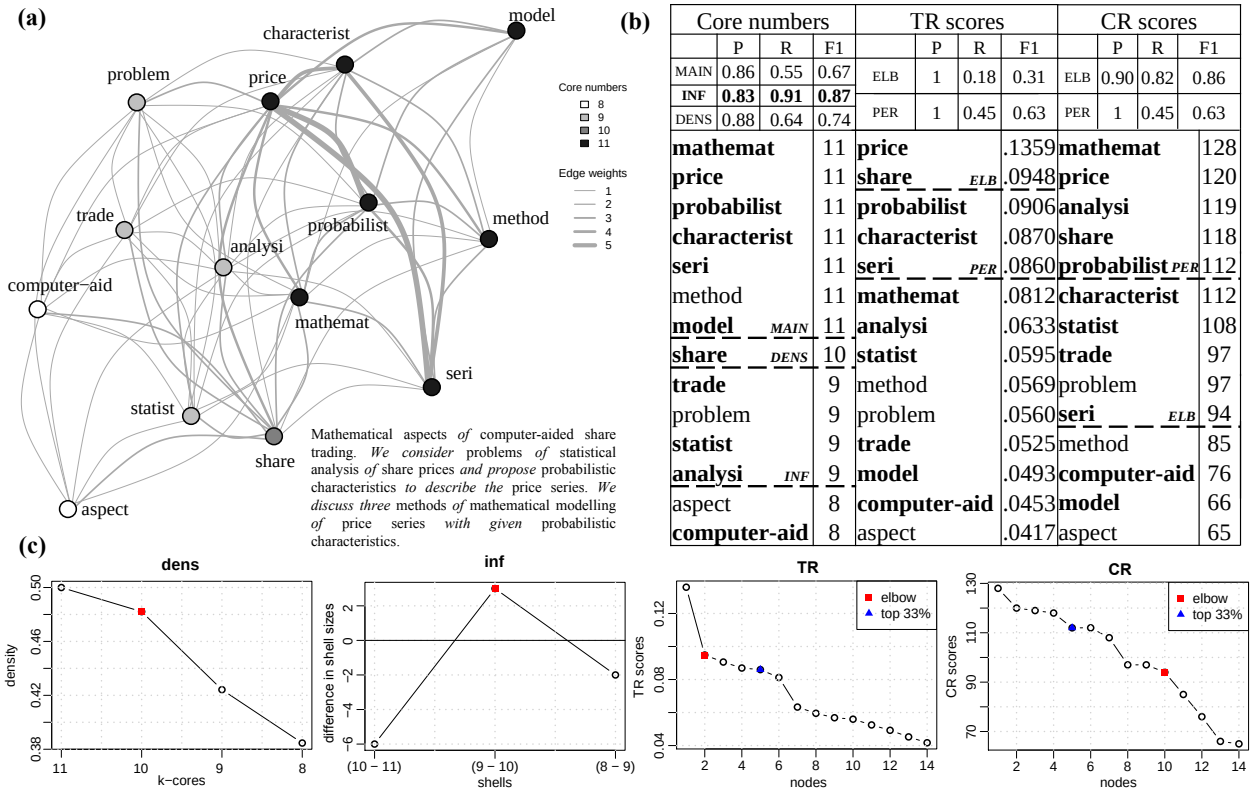
2 Graph-of-Words Representation

Many ways of encoding text as a graph have been explored in order to escape the very limiting *term-independence* assumption made by the traditional vector space model. In this study, we adopt the seminal Graph-of-Words representation (GoW) of (Mihalcea and Tarau, 2004), for its simplicity, high historical success, and above all because it was recently used in several approaches that reached very good performance on various tasks such as information retrieval (Rousseau and Vazirgiannis, 2013), document classification (Malliaros and Skianis, 2015; Rousseau et al., 2015), event detection (Meladianos et al., 2015), and keyword extraction (Rousseau and Vazirgiannis, 2015).

While sophisticated variants of the GoW model would be worth exploring (edge weights based on mutual information or word embeddings, adaptive window size, etc.), we aim here at making a bet-

*This research is supported in part by the OpenPaaS::NG project.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1860–1870,
Austin, Texas, November 1-5, 2016.



Core numbers	P	R	F1
MAIN	0.86	0.55	0.67
INF	**0.83**	**0.91**	**0.87**
DENS	0.88	0.64	0.74

TR scores	P	R	F1
ELB	1	0.18	0.31
PER	1	0.45	0.63

CR scores	P	R	F1
ELB	0.90	0.82	0.86
PER	1	0.45	0.63

		TR scores		CR scores	
mathemat	11	**price**	.1359	**mathemat**	128
price	11	**share** *ELB*	.0948	**price**	120
probabilist	11	**probabilist**	.0906	**analysi**	119
characterist	11	**characterist**	.0870	**share**	118
seri	11	**seri** *PER*	.0860	**probabilist** *PER*	112
method	11	**mathemat**	.0812	**characterist**	112
model *MAIN*	11	**analysi**	.0633	**statist**	108
share *DENS*	10	**statist**	.0595	**trade**	97
trade	9	method	.0569	problem	97
problem	9	problem	.0560	**seri** *ELB*	94
statist	9	**trade**	.0525	method	85
analysi *INF*	9	**model**	.0493	**computer-aid**	76
aspect	8	**computer-aid**	.0453	**model**	66
computer-aid	8	aspect	.0417	aspect	65

Figure 1: (a) Graph-of-words ($W = 8$) for document #1512 of Huth2003 decomposed with k-core (non-(nouns and adjectives) in *italic*). (b) Keywords extracted by each proposed and baseline method (human assigned keywords in **bold**). (c) Selection criterion of each method except *main* (does not apply).

ter use of an existing representation of text, not at proposing a new one. This is why, to demonstrate the additional skill brought by our proposed methods, we stick to the basic GoW framework.

As shown in Figure 1 (a), the GoW representation of (Mihalcea and Tarau, 2004) encodes a piece of text as an undirected graph where nodes are unique nouns and adjectives in the document, and where there is an edge between two nodes if the terms they represent co-occur within a window of predetermined size W that is slided over the entire document from start to finish, overspanning sentences. Furthermore, edges are assigned integer weights matching co-occurrence counts. This fully statistical approach is based on the *Distributional Hypothesis* (Harris, 1954), that is, on the premise that the relationship between words can be determined by the frequency with which they share local contexts of occurrence.

3 Graph Degeneracy

The concept of graph degeneracy was introduced by (Seidman, 1983) with the k-core decomposition technique and was first applied to the study of cohesion in social networks. Here, we consider it as an umbrella term also encompassing the K-truss algorithm (Cohen, 2008). In what follows, $G(V, E)$ is a graph with $|V|$ nodes and $|E|$ edges. Note that for graphs-of-words, the nodes V are labeled according to the unique terms they represent.

3.1 k-core subgraph

A core of order k (or k-core) of G is a maximal connected subgraph of G in which every vertex v has at least degree k (Seidman, 1983). If the edges are unweighted, the degree of v is simply equal to the count of its neighbors, while in the weighted case, the degree of v is the sum of the weights of its incident edges. Note that with the definition of GoW previously given, node degrees (and thus, k) are integers in both cases since edge weights are integers.

As shown in Figure 2 (a), the **k-core decomposition** of G is the set of all its cores from 0 (G itself) to k_{max} (its main core). It forms a hierarchy of nested subgraphs whose cohesiveness and size respectively increase and decrease with k (Seidman, 1983). The main core of G is a coarse approximation of its densest subgraph (Wang and Cheng, 2012), and should be seen as a *seedbed* within which it is possible to find more cohesive subgraphs (Seidman, 1983). Finally, the **core number** of a node is the highest order of a k-core subgraph that contains this node.

3.2 K-truss subgraph

K-truss is a triangle-based extension of k-core introduced by (Cohen, 2008). More precisely, the K-truss subgraph of G is its largest subgraph where every edge belongs to at least $K-2$ triangles. In other words, every edge in the K-truss joins two vertices that have at least $K-2$ common neighbors. Compared to k-core, K-truss thus does not iteratively prune nodes out based on the number of their *direct links*, but also based on the number of their *shared connections*. This more accurately captures cohesiveness.

As a result, the K-trusses are smaller and denser subgraphs than the k-cores, and the maximal K-truss of G better approximates its densest subgraph. In essence, and as illustrated in Figure 2 (b), the K-trusses can be viewed as the essential parts of the k-cores, i.e., what is left after the less cohesive elements have been filtered out (Wang and Cheng, 2012). By analogy with k-core, the **K-truss decomposition** of G is the set of all its K-trusses from $K=2$ to K_{max}, and the **truss number** of an *edge* is the highest order of a truss the edge belongs to. By extension, we define the truss number of a *node* as the maximum truss number of its incident edges.

3.3 k-shell

Depending on context, we will refer to the k-shell as the part of the k-core (or truss) that is not included in the $(k+1)$-core (or truss).

3.4 Graph degeneracy and spreading influence

In social networks, it has been shown that the best spreaders (i.e., the nodes able to propagate information to a large portion of the network at minimal time and cost) are not necessarily the highly connected individuals (i.e., the hubs), but rather those located at the core of the network (i.e., forming dense and cohesive subgraphs with other central nodes), as indicated by graph degeneracy algorithms (Kitsak et al., 2010). Put differently, the spreading influence of a node is related to its structural position within the graph (*density* and *cohesiveness*) rather than to its *prestige* (random walk-based degree). More recently, (Malliaros et al., 2016b) found that the truss number is an even better indicator of spreading influence than the core number. Motivated by these findings, we posit that taking *cohesiveness* into account with the core and truss decomposition of a graph-of-words could improve keyword extraction performance. That way, by analogy with the notion of influential spreaders in social networks, we hypothesize that *influential words* in graphs-of-words will act as representative keywords.

3.5 Complexity

(Batagelj and Zaveršnik, 2002) proposed $\mathcal{O}(|V| + |E|)$ and $\mathcal{O}(|E| \log |V|)$ time algorithms for k-core decomposition in the unweighted (resp. weighted) case. These algorithms are both $\mathcal{O}(|V|)$ in space. Computing the K-truss decomposition is more expensive, and requires $\mathcal{O}(|E|^{1.5})$ time and $\mathcal{O}(|V| + |E|)$ space (Wang and Cheng, 2012). Finally, building a graph-of-words is linear in time and space: $\mathcal{O}(|V|W)$ and $\mathcal{O}(|V| + |E|)$, respectively.

4 Related Work and Point of Departure

TextRank. One of the most popular approaches to the task of *unsupervised single-document keyword extraction* is TextRank (Mihalcea and Tarau, 2004), or TR in what follows. In TR, the nodes of graphs-of-words are ranked based on a modified version of the PageRank algorithm taking edge weights into account, and the top $p\%$ vertices are kept as keywords.
Limitations. TR has proven successful and has been widely used and adapted. However, PageRank, which is based on the concept of random walks and is also related to *eigenvector centrality*, tends to favor nodes with many important connections regardless of any cohesiveness consideration. For undirected graphs, it was even shown that PageRank values are proportional to node degrees (Grolmusz, 2015). While well suited to the task of

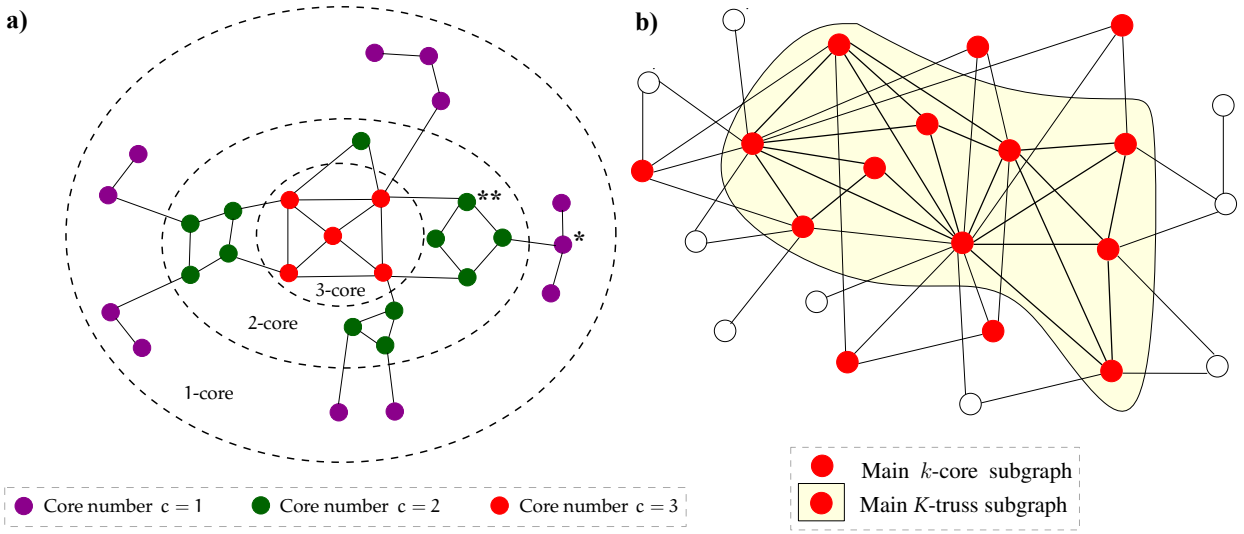


Figure 2: (a) k-core decomposition illustrative example. Note that while nodes * and ** have same degree (3), node ** makes a more influential spreader as it lies in a higher core than node *. (b) k-core versus K-truss. The main K-truss subgraph can be considered as the *core* of the main core.

prestige-based ranking in the Web and social networks (among other things), PageRank may thus not be ideal for keyword extraction. Indeed, a fundamental difference when dealing with text is the paramount importance of cohesiveness: keywords need not only to have many important connections but also to form *dense substructures* with these connections. Actually, most of the time, keywords are n-grams (Rousseau and Vazirgiannis, 2015). Therefore, we hypothesize that keywords are more likely to be found among the *influential spreaders* of a graph-of-words – as extracted by degeneracy-based methods – rather than among the nodes high on *eigenvector*-related centrality measures.

Topical vs. network coherence. Note that, unlike a body of work that tackled the task of keyword extraction and document summarization from a *topical* coherence perspective (Celikyilmaz and Hakkani-Tür, 2011; Chen et al., 2012; Christensen et al., 2013), we deal here with *network* coherence, a purely graph theoretic notion orthogonal to topical coherence.

Graph degeneracy. The aforementioned limitation of TR motivated the use of graph degeneracy to not only extract central nodes, but also nodes forming dense subgraphs. More precisely, (Rousseau and Vazirgiannis, 2015) applied both unweighted and weighted k-core decomposition on graphs-of-words and retained the members of the main cores as keywords. Best results were obtained in the weighted case, with small main cores yielding good precision but low recall, and significantly outperforming TR. As expected, (Rousseau and Vazirgiannis, 2015) observed that cores exhibited the desirable property of containing "long-distance n-grams".

In addition to superior quantitative performance, another advantage of degeneracy-based techniques (compared to TR, which extracts a constant percentage of nodes) is *adaptability*. Indeed, the size of the main core (and more generally, of every level in the hierarchy) depends on the structure of the graph-of-words, which by nature is uniquely tailored to the document at hand. Consequently, the distribution of the number of extracted keywords matches more closely that of the human assigned keywords (Rousseau and Vazirgiannis, 2015).

Limitations. Nevertheless, while (Rousseau and Vazirgiannis, 2015) made great strides, it suffers the following limitations: (1) k-core is *good* but not *best* in capturing cohesiveness; (2) retaining only the main core (or truss) is suboptimal, as one cannot expect all the gold standard keywords to be found within a unique subgraph – actually, many valuable keywords live in lower levels of the hierarchy (see Figure 1); and (3) the coarseness of the k-core

decomposition implies to work at a high granularity level (selecting or discarding a large group of words at a time), which diminishes the flexibility of the extraction process and negatively impacts performance.

Research objectives. To address the aforementioned limitations, we investigate in this study (1) the use of K-truss to get a finer-grained hierarchy of more cohesive subgraphs, in order to filter unwanted words out of the upper levels and improve flexibility; (2) the automated selection of the best level in the core (or truss) hierarchy to increase recall while preserving most of the precision; and (3) the conversion of node core (or truss) numbers into ranks, to decrease granularity from the *subgraph* to the *node* level, while still leveraging the valuable cohesiveness information captured by degeneracy.

5 Proposed Methods

In what follows, we introduce the strategies we devised to implement our research objectives.

5.1 Density

With the underlying assumption that keywords are found in cohesive subgraphs-of-words and are not all contained in the main core (or truss), an intuitive, straightforward stopping criterion when going down the core (or truss) hierarchy is a density-based one. More precisely, it may be advantageous to go down the hierarchy as long as the desirable cohesiveness properties of the main core or truss are maintained, and to stop when these properties are lost. This strategy is more formally detailed in Algorithm 1, where $G(V, E)$ is a graph-of-words, $levels$ corresponds to the vector of the n_{levels} unique k-core (or truss) numbers of V sorted in decreasing order, and the density of G is defined as:

$$density(G) = \frac{|E|}{|V|\,(|V|-1)} \qquad (1)$$

As can be seen in Figure 1 (c) and as detailed in Algorithm 2, the elbow is determined as the most distant point from the line joining the first and last point of the curve. When all points are aligned, the top level is retained (i.e., main core or truss). When there are only two levels, the one giving the highest density is returned.

Algorithm 1: *dens* method

Input : core (or truss) decomposition of G
Output: set of keywords

D $\leftarrow$ empty vector of length n_{levels}
for $n \leftarrow 1$ **to** n_{levels} **do**
  $D[n] \leftarrow density(levels[n]$-core (or truss)$)$
end
$k_{best} \leftarrow levels[elbow(n, D[n])]$
return k_{best}-core (or truss) of G as keywords

Algorithm 2: *elbow*

Input : set of $|S| \geq 2$ points $S = \{(x_0, y_0), ..., (x_{|S|}, y_{|S|})\}$
Output: x_{elbow}

$line \leftarrow \{(x_0, y_0); (x_{|S|}, y_{|S|})\}$
if $|S| > 2$ **then**
  $distance \leftarrow$ empty vector of length $|S|$
  $s \leftarrow 1$
  for (x, y) *in* S **do**
    $distance[s] \leftarrow$ distance from (x, y) to $line$
    $s \leftarrow s + 1$
  end
  if $\exists! s \mid distance[s] = max(distance)$ **then**
    $x_{elbow} \leftarrow x \mid (x, y)$ is most distant from $line$
  else
    $x_{elbow} \leftarrow x_0$
  end
else
  $x_{elbow} \leftarrow x \mid y$ is maximum
end
return x_{elbow}

5.2 Inflexion

The Inflexion method (*inf* in what follows) is an empirically-grounded heuristic that relies on detecting changes in the variation of shell sizes (where size denotes the number of nodes). Recall from subsection 3.3 than the k-shell is the part of the k-core (or truss) that does not survive in the $(k+1)$-core (or truss), that is, the subgraph of G induced by the nodes with core (or truss) number exactly equal to k. In simple terms, the *inf* rule-of-thumb consists in going down the hierarchy as long as the shells in-

crease in size, and stopping otherwise. More precisely, *inf* is implemented as shown in Algorithm 3, by computing the consecutive differences in size across the shells and selecting the first positive point before the drop into the negative half (see Figure 1c). If no point satisfies this requirement, the main core (or truss) is extracted.

Algorithm 3: *inf* method

Input : core (or truss) decomposition of G
Output: set of keywords

CD $\leftarrow$ empty vector of length $n-1$
for $n \leftarrow 1$ **to** $(n_{levels}-1)$ **do**
  $k_l \leftarrow levels[n+1]$; $k_u \leftarrow levels[n]$
  $CD[n] \leftarrow$ size$\big(k_l$-shell$\big)-$size$\big(k_u$-shell$\big)$
end
if $\exists n \mid (CD[n+1] < 0 \wedge CD[n] > 0)$ **then**
  $n_{best} \leftarrow n$
else
  $n_{best} \leftarrow 1$
end
$k_{best} \leftarrow levels[n_{best}]$
return k_{best}-core (or truss) as keywords

Note that both *dens* and *inf* enjoy the same adaptability as the main core retention method of (Rousseau and Vazirgiannis, 2015) explained in Section 4, since the sizes of *all* the subgraphs in the hierarchy suit the structure of the graph-of-words.

5.3 CoreRank

Techniques based on retaining the k_{best}-core (or truss), such as *dens* and *inf* previously described, are better than retaining only the top level but lack flexibility, in that they can only select an entire batch of nodes at a time. This is suboptimal, because of course not all the nodes in a given group are equally good. To address this issue, our proposed CoreRank method (CR in what follows) converts nodes core (or truss) numbers into scores, ranks nodes in decreasing order of their scores, and selects the top $p\%$ nodes (CRP) or the nodes before the elbow in the scores curve (CRE). Note that for simplicity, we still refer to the technique as CR even when dealing with truss numbers.

Flexibility is obviously improved by decreasing granularity from the *subgraph* level to the *node* level. However, to avoid going back to the limitations of TR (absence of cohesiveness considerations), it is crucial to decrease granularity while retaining as much of the desirable information encoded by degeneracy as possible. To accomplish this task, we followed (Bae and Kim, 2014) and assigned to each node the sum of the core (or truss) numbers of its neighbors.

Our CRE method is outlined in Algorithm 4, where $N(v)$ denotes the set of neighbors of vertex v, and $number(v)$ is the core (or truss) number of v. CRP implements the exact same strategy, the only difference being $n_{best} \leftarrow round(|V| * percentage)$ at step 8 (where $percentage$ is a real number between 0 and 1).

Algorithm 4: CRE method

Input : core (or truss) decomposition of G
Output: set of keywords

CR $\leftarrow$ empty vector of length $|V|$
for $n \leftarrow 1$ **to** $|V|$ **do**
  $v \leftarrow V[n]$
  $CR[n] \leftarrow \sum_{u \in N(v)} number(u)$
  $name(CR[n]) \leftarrow label(v)$
end
sort CR in decreasing order
$n_{best} \leftarrow elbow(n, CR[n])$
return $names(CR[1 : n_{best}])$ as keywords

6 Experimental Setup

6.1 Baseline methods

TextRank (TR). We used as our first benchmark the system of (Mihalcea and Tarau, 2004) discussed in Section 4. For the sake of fair comparison with our CRE and CRP methods, we considered two variants of TR that respectively retain nodes based on both the elbow (TRE) and percentage criteria (TRP).

Main. Our second baseline is the main core retention technique of (Rousseau and Vazirgiannis, 2015), also described in Section 4. This method is referred to as *main* in the remainder of this paper.

6.2 Datasets

To evaluate performance, we used three standard, publicly available datasets featuring documents of

various types and sizes. Figure 3 shows the distributions of document size and manually assigned keywords for each dataset.

The **Hulth2003**[1] (Hulth, 2003) dataset contains abstracts drawn from the Inspec database of physics and engineering papers. Following our baselines, we used the 500 documents in the validation set and the "uncontrolled" keywords assigned by human annotators. The mean document size is 120 words and on average, 21 keywords (in terms of unigrams) are available for each document.

We also used the training set of **Marujo2012**[1], containing 450 web news stories of about 440 words on average, covering 10 different topics from art and culture to business, sport, and technology (Marujo et al., 2012). For each story, the keyphrases assigned by at least 9 out of 10 Amazon Mechanical Turkers are provided as gold standard. After splitting the keyphrases into unigrams, this makes for an average of 68 keywords per document, which is much higher than for the two other datasets, even the one comprising long documents (Semeval, see next).

The **Semeval**[2] dataset (Kim et al., 2010) offers parsed scientific papers collected from the ACM Digital Library. More precisely, we used the 100 articles in the test set and the corresponding author-and-reader-assigned keyphrases. Each document is approximately 1,860 words in length and is associated with about 24 keywords.

Notes. In Marujo2012, the keywords were assigned in an extractive manner, but many of them are verbs. In the two other datasets, keywords were freely chosen by human coders in an abstractive way and as such, some of them are not present in the original text. On these datasets, reaching perfect recall is therefore impossible for our methods (and the baselines), which by definition all are extractive.

6.3 Implementation

Before constructing the graphs-of-words and passing them to the keyword extraction methods, we performed the following pre-processing steps:

Stopwords removal. Stopwords from the SMART information retrieval system[3] were discarded.

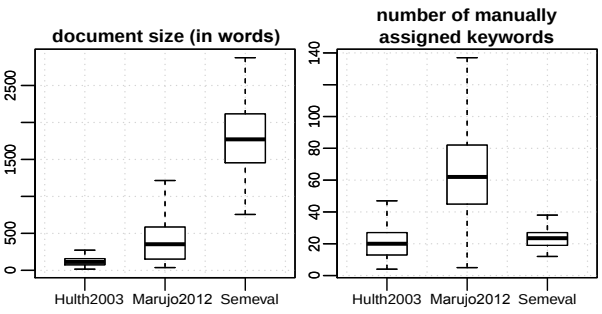


Figure 3: Basic dataset statistics.

Part-of-Speech tagging and screening using the `openNLP` (Hornik, 2015) R (R Core Team, 2015) implementation of the Apache OpenNLP Maxent POS tagger. Then, following (Mihalcea and Tarau, 2004), only nouns and adjectives were kept. For Marujo2012, as many gold standard keywords are verbs, this step was skipped (note that we did experiment with and without POS-based screening but got better results in the second case).

Stemming with the R `SnowballC` package (Bouchet-Valat, 2014) (Porter's stemmer). Gold standard keywords were also stemmed.

After pre-processing, graphs-of-words (as described in Section 2) were constructed for each document and various window sizes (from 3, increasing by 1, until a plateau in scores was reached). We used the R `igraph` package (Csardi and Nepusz, 2006) to write graph building and weighted k-core implementation code. For K-truss, we used the C++ implementation offered by (Wang and Cheng, 2012).

Finally, for TRP and CRP, we retained the top 33% keywords on Hulth2003 and Marujo2012 (short and medium size documents), whereas on Semeval (long documents), we retained the top 15 keywords. This is consistent with our baselines. Indeed, the number of manually assigned keywords increases with document size up to a certain point, and stabilizes afterwards.

The code of the implementation and the datasets can be found here[4].

[1] `https://github.com/snkim/AutomaticKeyphraseExtraction`

[2] `https://github.com/boudinfl/centrality_measures_ijcnlp13/tree/master/data`

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

[4] `https://github.com/Tixierae/EMNLP_2016`

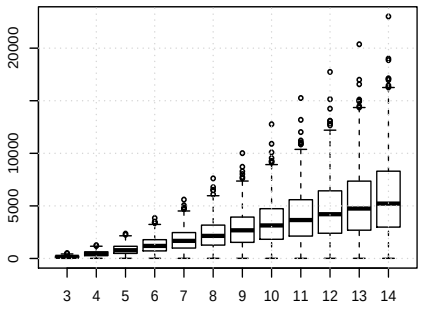


Figure 4: Triangle count versus window size (Hulth2003).

	precision	recall	F1-score
dens	**48.79**	**72.78**	**56.09***
inf	48.96	72.19	55.98*
CRP	61.53	38.73	45.75
CRE	65.33	37.90	44.11
main†	51.95	54.99	50.49
TRP†	65.43	41.37	48.79
TRE†	71.34	36.44	45.77

Table 1: Hulth2003, K-truss, $W = 11$.
*statistical significance at $p < 0.001$ with respect to all baselines.
†baseline systems.

	precision	recall	F1-score
dens	**47.62**	**71.46**	**52.94***
inf	53.88	57.54	49.10*
CRP	54.88	36.01	40.75
CRE	63.17	25.77	34.41
main†	64.05	34.02	36.44
TRP†	55.96	36.48	41.44
TRE†	65.50	21.32	30.68

Table 2: Marujo2012, k-core, $W = 13$.
*statistical significance at $p < 0.001$ with respect to all baselines.
†baseline systems.

7 Experimental Results

7.1 Evaluation

We computed the standard precision, recall, and F-1 measures for each document and averaged them at the dataset level (macro-averaging).

7.2 Precision/Recall trade-off

As shown in Figure 5, our methods *dens* and *inf* outperform the baselines by a wide margin on the datasets containing small and medium size documents (Hulth2003 and Marujo2012). As expected, this superiority is gained from a drastic improvement in recall, for a comparatively lower precision loss. TR and *main* exhibit higher precision than recall, which is in accordance with (Rousseau and Vazirgiannis, 2015). The same observation can be made for our CR method. For TR, the unbalance is more severe on the Hulth2003 and Marujo2012 datasets (short/medium documents) when the elbow method is used (TRE), probably because it tends to retain only a few nodes. However, on Semeval (long documents), using the elbow method (TRE) gives the best trade-off between precision and recall. For CR, still on Semeval, using the elbow method (CRE) even gives better recall than precision.

Overall, compared to the baselines, the unbalance between precision and recall for our methods is less extreme or equivalent. On the Marujo2012 dataset for instance, our proposed *inf* method is almost perfectly balanced and ranks second (significantly better than all baselines).

7.3 Impact of window size

The performance of k-core does not dramatically increase with window size, while K-truss exhibits a surprising "cold start" behavior and only begins to kick-in for sizes greater than 4-5. A possible explanation is that the ability of K-truss (which is triangle-based) to extract meaningful information from a graph depends, up to a certain point, on the amount of triangles in the graph. In the case of graph-of-words, the number of triangles is positively correlated with window size (see Figure 4). It also appears that document size (i.e., graph-of-words structure) is responsible for a lot of performance variability. Specifically, on longer documents, performance plateaus at higher window sizes.

7.4 Best models comparison

For each dataset, we retained the degeneracy technique and window size giving the absolute best performance in F1-score, and compared all methods under these settings (see Tables 1–3). We tested for statistical significance in macro-averaged F1 scores using the non-parametric version of the t-test, the Mann-Whitney U test[5].

On Hulth2003 and Marujo2012 (short and medium size documents), our methods *dens* and *inf* strongly and significantly outperform all baselines, with respective absolute improvements of more than 5.5% with respect to the best performing baseline

[5] https://stat.ethz.ch/R-manual/R-devel/library/stats/html/wilcox.test.html

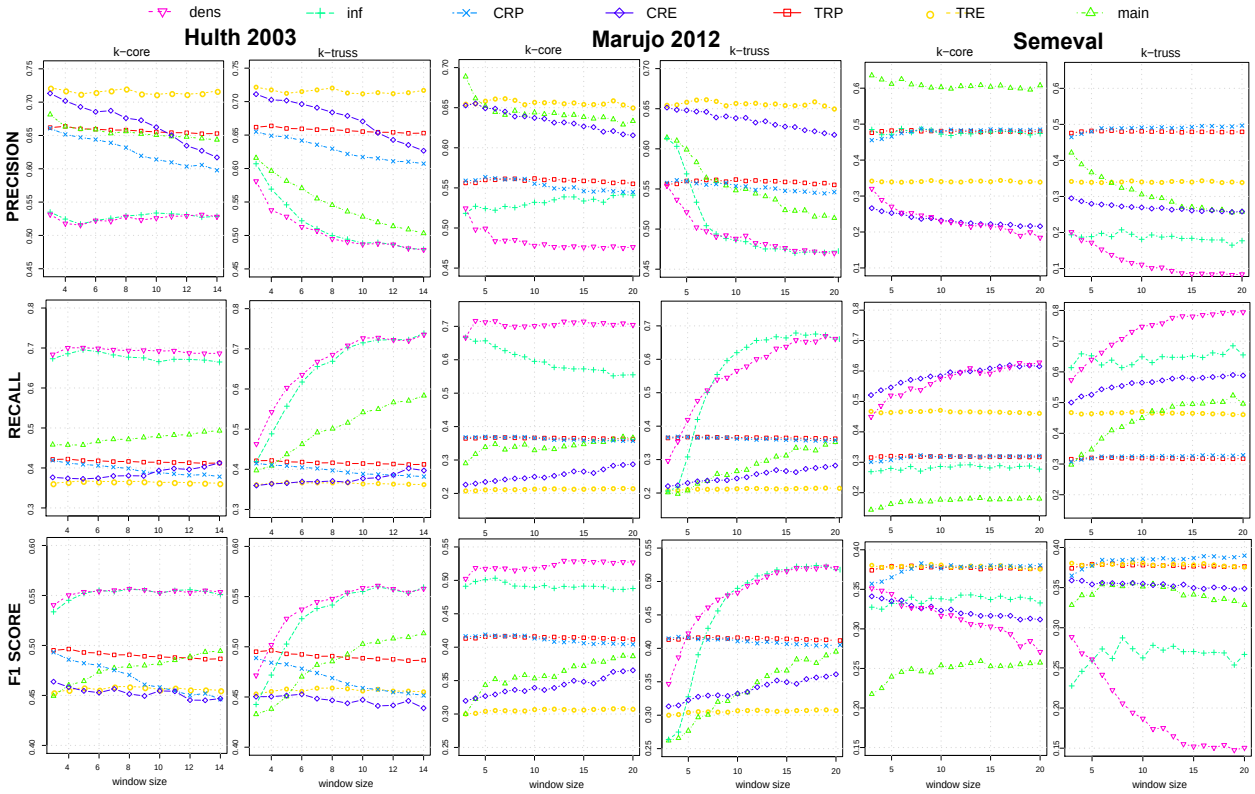


Figure 5: Impact of window size on performance.

	precision	recall	F1-score
dens	8.44	79.45	15.06
inf	17.70	65.53	26.68
CRP	**49.67**	**32.88**	**38.98***
CRE	25.82	58.80	34.86
main†	25.73	49.61	32.83
TRP†	47.93	31.74	37.64
TRE†	33.87	46.08	37.55

Table 3: Semeval, K-truss, $W = 20$.
*statistical significance at $p < 0.001$ w.r.t. *main*. †baseline systems.

(*main*). On Semeval, which features larger pieces of text, our CRP technique improves on TRP, the best performing baseline, by more than 1 %, altough the difference is not statistically significant. However, CRP is head and shoulders above *main*, with an absolute gain of 6%. This suggests that converting the cohesiveness information captured by degeneracy into ranks may be valuable for large documents.

Finally, the poor performance of the *dens* and *inf* methods on Semeval (Table 3) might be explained by the fact that these methods are only capable of selecting an entire batch of nodes (i.e., subgraph-of-words) at a time. This lack of flexibility seems to become a handicap on long documents for which the graphs-of-words, and thus the subgraphs corresponding to the k-core (or truss) hierarchy levels, are very large. This analysis is consistent with the observation that conversely, approaches that work at a finer granularity level (node level) prove superior on long documents, such as our proposed CRP method which reaches best performance on Semeval.

8 Conclusion and Future Work

Our results provide empirical evidence that *spreading influence* may be a better "keywordness" metric than *eigenvector* (or *random walk*)-based criteria. Our CRP method is currently very basic and leveraging edge weights/direction or combining it with other scores could yield better results. Also, more meaningful edge weights could be obtained by merging local co-occurrence statistics with external semantic knowledge offered by pre-trained word embeddings (Wang et al., 2014). The direct use of density-based objective functions could also prove valuable.

References

Joonhyun Bae and Sangwook Kim. 2014. Identifying and ranking influential spreaders in complex networks by neighborhood coreness. *Physica A: Statistical Mechanics and its Applications*, 395:549–559.

Vladimir Batagelj and Matjaž Zaveršnik. 2002. Generalized cores. *arXiv preprint cs/0202039*.

Milan Bouchet-Valat, 2014. *SnowballC: Snowball stemmers based on the C libstemmer UTF-8 library*. R package version 0.5.1.

Asli Celikyilmaz and Dilek Hakkani-Tür. 2011. Discovery of topically coherent sentences for extractive summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 491–499. Association for Computational Linguistics.

Yun-Nung Chen, Yu Huang, Hung-Yi Lee, and Lin-Shan Lee. 2012. Unsupervised two-stage keyword extraction from spoken documents by topic coherence and support vector machine. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5041–5044. IEEE.

Janara Christensen, Stephen Soderland Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *HLT-NAACL*, pages 1163–1173. Citeseer.

Jonathan Cohen. 2008. Trusses: Cohesive subgraphs for social network analysis. *National Security Agency Technical Report*, page 16.

Gabor Csardi and Tamas Nepusz. 2006. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695.

Vince Grolmusz. 2015. A note on the pagerank of undirected graphs. *Information Processing Letters*, 115(6):633–634.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Kurt Hornik, 2015. *openNLP: Apache OpenNLP Tools Interface*. R package version 0.2-5.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 216–223. 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. Association for Computational Linguistics.

Maksim Kitsak, Lazaros K Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H Eugene Stanley, and Hernán A Makse. 2010. Identification of influential spreaders in complex networks. *Nature Physics*, 6(11):888–893.

Fragkiskos D Malliaros and Konstantinos Skianis. 2015. Graph-based term weighting for text categorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1473–1479. ACM.

Fragkiskos D. Malliaros, Apostolos N. Papadopoulos, and Michalis Vazirgiannis. 2016a. Core decomposition in graphs: concepts, algorithms and applications. In *Proceedings of the 19th International Conference on Extending Database Technology, EDBT*, pages 720–721.

Fragkiskos D Malliaros, Maria-Evgenia G Rossi, and Michalis Vazirgiannis. 2016b. Locating influential nodes in complex networks. *Scientific Reports*, 6:19307.

Luis Marujo, Anatole Gershman, Jaime Carbonell, Robert Frederking, and Jo ao P. Neto. 2012. Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. In *Proceedings of LREC 2012*. ELRA.

Polykarpos Meladianos, Giannis Nikolentzos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. 2015. Degeneracy-based real-time sub-event detection in twitter stream. In *Ninth International AAAI Conference on Web and Social Media (ICWSM)*.

Rada Mihalcea and Paul Tarau. 2004. TextRank: bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

R Core Team, 2015. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and tw-idf: new approach to ad hoc ir. In *Proceedings of the 22nd ACM international conference on Conference on Information & Knowledge Management (CIKM)*, pages 59–68. ACM.

François Rousseau and Michalis Vazirgiannis. 2015. Main core retention on graph-of-words for single-document keyword extraction. In *Advances in Information Retrieval*, pages 382–393. Springer.

François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *ACL*, volume 15, page 107.

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

Jia Wang and James Cheng. 2012. Truss decomposition in massive networks. *Proceedings of the VLDB Endowment*, 5(9):812–823.

Rui Wang, Wei Liu, and Chris McDonald. 2014. Corpus-independent generic keyphrase extraction using word embedding vectors. In *Software Engineering Research Conference*, page 39.