Incremental TextRank - Automatic Keyword Extraction for Text Streams

Rui Portocarrero Sarmento[1], Mário Cordeiro[1], Pavel Brazdil[2] and João Gama[2]
[2]*LIAAD-INESC TEC, Portugal*
[1]*PRODEI - Faculty of Engineering, University of Porto, Portugal*

Keywords: Automatic Keyword Extraction, Incremental PageRank, Data Streams, Text Mining, Incremental TextRank.

Abstract: Text Mining and NLP techniques are a hot topic nowadays. Researchers thrive to develop new and faster algorithms to cope with larger amounts of data. Particularly, text data analysis has been increasing in interest due to the growth of social networks media. Given this, the development of new algorithms and/or the upgrade of existing ones is now a crucial task to deal with text mining problems under this new scenario. In this paper, we present an update to TextRank, a well-known implementation used to do automatic keyword extraction from text, adapted to deal with streams of text. In addition, we present results for this implementation and compare them with the batch version. Major improvements are lowest computation times for the processing of the same text data, in a streaming environment, both in sliding window and incremental setups. The speedups obtained in the experimental results are significant. Therefore the approach was considered valid and useful to the research community.

1 INTRODUCTION

Automatic keyword extraction from text is an essential area of research. It has been used to automatically summarize documents. From enterprises' document production to the analysis of users' social networks posts, for example, in Twitter or Facebook, the appropriate summarizing of text brings new possibilities to better target advertising and recommendations to users. It is also used to find similarities between authors and improve enterprise departments workflow. The majority of current systems are prepared to be used statically. Thus, with the explosion of large-scale document sets, researchers felt a considerable increase in the need for systems to analyze this evolving and even more extensive datasets. The improvement of previous algorithms or the adaptation of these systems to cope with incoming text is, therefore, imperative nowadays. Although not many researchers were concerned with the issue of incremental keyword extraction, some of them are starting to implement new algorithms for this issue (Shin et al., 2014). Incremental algorithms have been developed extensively in several areas of Data Analysis. Notably, those algorithms that are graph oriented have been proving to be faster or more efficient when their incremental adaptations are compared to their static versions. Our contribution, in this paper, is the transformation of a well-known algorithm for keyword extraction, TextRank (Mihalcea and Tarau, 2004). Taking into account that this algorithm is a graph-based approach to keyword extraction, we updated it to be used in a streaming setting. We validate the results by comparing the behavior and processing time with the original algorithm. The paper is organized as follows. Section 2 presents the related work regarding TextRank and the components of its implementation. Section 3 introduces the algorithm for Incremental TextRank. In Section 4 we explain how the proposed method was tested and evaluated. In Section 5 we present the results and show the effectiveness of the proposed method. Then, in Section 6 we discuss the solution and the results. Finally, the last Section highlights the major contributions, and discuss further work to enhance the method.

2 RELATED WORK

In this section, related work regarding automatic keyword extraction systems, particularly those based on graph approaches is presented. Automatic keyword extraction is a concept that relies on the process of selection of words or phrases that, without human intervention, provide a clear idea about the core area of a document or set of documents. Those implemen-

Sarmento, R., Cordeiro, M., Brazdil, P. and Gama, J.
Incremental TextRank - Automatic Keyword Extraction for Text Streams.
DOI: 10.5220/0006639703630370
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, pages 363-370
ISBN: 978-989-758-298-1

tations can be classified into four classes according to Siddiqi and Sharan (2015), Rule-Based Linguistic approaches, Statistical approaches, Machine Learning approaches, and Domain-specific approaches. These approaches have their advantages and disadvantages when compared to one another. Inside each class, the approach can also be divided into supervised and unsupervised methods according to Siddiqi and Sharan (2015). The use of human tasks or contribution in unsupervised approaches is regarded to be minimal. Otherwise, by using supervised approaches, researchers take into account previous annotations by humans, and their participation in finding quality keyword annotations from texts is a significant concern in these methods. According to Siddiqi and Sharan (2015) survey, there are several unsupervised approaches of interest:

Bracewell et al. (2005) extract noun phrases from a document and then cluster the terms which have the same noun term. Then, the author's proposal consists of the ranking of clusters, taking in account term and noun phrase frequencies. Finally, only the Top-ranked clusters are selected as keyphrases. Liu et al. (2009) proposed to extract keyphrases by utilizing clustering techniques. According to these authors, they ensure that the document is semantically covered by these keyphrases. Rose et al. (2010) described Rapid Automatic Keyword Extraction (RAKE), a method for extracting keywords from individual documents. RAKE is based on the observation that keywords frequently contain multiple words, but they rarely include punctuation or stop words; therefore, any words having minimal content or information. This new method is domain and language-independent. Gazendam et al. (2010) describe the extraction and ranking of keywords with a restricted vocabulary with the help of a thesaurus. For ranking words, it uses a weighting scheme called tf-rr which uses both the term frequency and the number of thesaurus relations realized between the thesaurus terms found in the specific document. This is an approach that does not need any training from a reference corpus. Finally, we can also account for two graph-based methods. Mihalcea and Tarau (2004) proposed TextRank, a new graph-based ranking model for graphs extracted from texts. Thus, to rank keywords based on the co-occurrence links between words, Mihalcea et al. use the concept of PageRank between words to extract important keyphrases with linked words that have higher weight in the word graph. Litvak et al. (2011) proposed DegExt, a graph-based, cross-lingual keyphrase extractor. The authors proposed a graph representation, based on a simple graph-based syntactic representation of a document, and enhanced the model by taking into account structural document features. In this work, we choose to improve TextRank, due to its considerable usefulness in this area, and relatively good results for keyword extraction. Additionally, by being a graph-based approach, it is reasonable to think that an incremental approach can provide a significant improvement in running time or efficiency, in a streaming context. By considering small graph updates instead of all the graph in each update, we will try a parallelism with other works in the graphs area, where researchers consider only affected nodes in each update. The improvements in graph processing efficiency are those authors main contribution.

2.1 TextRank

As previously stated, Mihalcea and Tarau (2004) developed TextRank as a solution to obtain keywords automatically from text. Figure 1 shows the workflow of the original TextRank algorithm. The text corpus processing starts by the pre-processing of the text and the removal of stop words, numbers and punctuation. Then, the document goes through a process of annotation where remaining single words are categorized, for example, like nouns, verbs or adjectives among others. This method is called Part-of-Speech tagging (POS tagging). According to the authors, only a few of these annotated words are essential. The authors studied which group of words delivered best results, and they concluded that the best automatic keyphrases were obtained with nouns and adjectives. Then, with these filtered words, a graph-based approach is used. Each word is considered a graph node and the connections of words in this directed graph is determined by the order they appear in the text. The weight of these links is obtained by counting the number of times these pairs of words occur in the text corpus. The next phase of the algorithm regards the selection of the words of high importance. This is done with the use of the PageRank algorithm by Page et al. (1998). The words with high PageRank values are selected as potential keywords. Finally, the keyphrases are obtained with a post-processing stage. This stage involves the use of a sliding window evolving through the initial text to assess the order of words that are contained in the keyphrases or keywords. This step takes into account punctuation and other structural features of the document to retrieve reasonable keyphrases. TextRank has been widely praised as a consistent method to automatically retrieve keywords from the text. Inclusively, it has been used in prototypes of decision support systems as narrated by Brazdil et al. (2015).

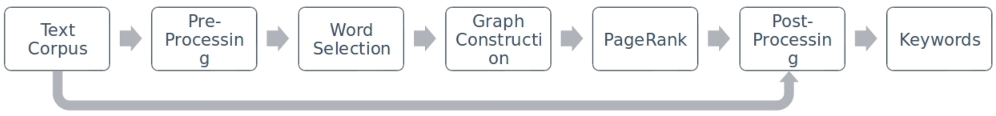

Figure 1: Original TextRank workflow.

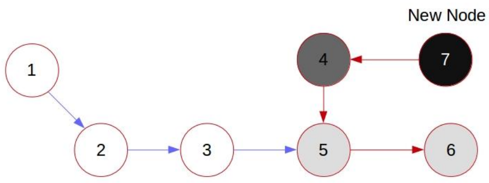

Figure 2: Incremental PageRank workflow - Addition of a Node.

2.2 Incremental PageRank

The reader should note that the original TextRank algorithm uses the original PageRank algorithm in one of its phases. We will show, later in this document, that a significant improvement of TextRank might be possible by transforming some of its phases algorithms to a streaming approach. Revisions of these algorithms were already developed as we will see. Particularly, there have been several attempts to improve the original PageRank algorithm. The purpose of several of these improvements was to adapt this algorithm for streaming data. Desikan et al. (2005) provided a solution for the update of nodes' PageRank values in evolving graphs. The algorithm explores the fact that the web evolves incrementally and with small changes between updates. Thus, according to the author's proposal, we should focus only on the pages that change their PageRank values due to the addition or removal of new pages/nodes and the correspondent change in the affected nodes by this change. Desikan et al. (2005) continued with this work and stress the need to find the nodes that are affected by the changes in the graph and find three categories of nodes:

1. Nodes that are directly affected by changes in the graph, e.g., the nodes that are outgoing nodes of added or removed nodes;
2. Nodes that are indirectly affected. For example, the neighbors of the nodes in 1, and consequently their outgoing neighbors also;
3. The nodes not affected by the changes, but that have their PageRank value updated anyway, due to a simple graph scale update.

Figure 2 shows a small network with these types of nodes. In this figure, changes in the graph are represented by added node 7. In this figure, we can also see nodes of type 1 colored dark gray, nodes of type 2 colored light gray, and nodes of type 3 with white color.

2.3 Space Saving Top-K

The problem of finding the most frequent items, in our case, keywords, in a data stream S of size N is the problem of discovering the elements e_i whose relative frequency f_i is higher than a user-specified support ϕN, with $0 \leq \phi \leq 1$ (Gama, 2010). Given the space requirements that exact algorithms addressing this problem would need (Charikar et al., 2002), several algorithms have been proposed to find the top-K frequent elements, being roughly classified into *counter-based* and *sketch-based* (Metwally et al., 2005). *Counter-based* techniques keep counters for each element in the monitored set, which is usually a much smaller than the entire set of elements. When an element is identified as not currently being monitored, various algorithms take different actions to adapt the monitored set accordingly. *Sketch-based* techniques provide less rigid guarantees, but they do not monitor a subset of elements, providing frequency estimators for the entire set. Simple *counter-based* algorithms, such as *Sticky Sampling* and *Lossy Counting*, were proposed in (Manku and Motwani, 2002), which process the stream in compressed size. They have the disadvantage of keeping a lot of irrelevant counters. *Frequent* (Demaine et al., 2002) keeps only k counters for monitoring k elements, incrementing each element counter when it is observed, and decrementing all counters when an unmonitored element is observed. Zeroed-counted elements are replaced by new unmonitored elements. This strategy is similar to the one applied by *Space-Saving* (Metwally et al., 2005), which gives guarantees for the *top-m* most frequent elements. This was the algorithm we selected to extract the most frequent keywords in the stream of text.

3 TextRank FOR TEXT STREAMS

In this section, we explain our proposal for two versions of an incremental TextRank. We propose a Window-based version and a more complex incremental version. In Figure 1 we showed the sequence of operations the text goes through, in the original TextRank. Although some parts of it are eminently evolving as, for example, the graph construction from the text words, other parts are not prepared for streaming or incremental approach.

3.1 Window-based Streaming TextRank

The first version we developed was based on the concept of using a sliding window to iterate through the stream of data. Thus, with this version, the current snapshot processing is unrelated to the previous snapshots processing. Therefore, each chunk of data from the stream is treated independently from snapshot to snapshot. In each iteration of the Window-based TextRank, we only process the current piece of data, we pre-process this data, and we do the standard word selection by considering only nouns and adjectives. Additionally, when we do the graph construction we just build it considering the selected words for the current chunk of data. Thus, the next phase of the algorithm, i.e., the use of PageRank algorithm to detect strong words in the text also only processes the words in the current snapshot graph. Again, regarding post-processing, it is done considering the words in the text available just for the current snapshot. As previously stated, the *top-K* phase of this Window-based streaming version, uses the *top-K* space saving implementation that is based on the concept of landmark windows. Thus, the *top-K* list of keywords is the only information that transits from snapshot to snapshot processing for the streaming data.

3.2 Incremental TextRank

Figure 3 shows our proposal for the upgraded incremental TextRank. We have changed the workflow of the original algorithm to be able to cope with evolving text streams. The input of a new stream of text starts to be pre-processed in the same way that was used for the original algorithm. At this stage, as in the original algorithm, we filter only the desired words as, for example, nouns and adjectives. Then, the workflow proceeds with the word graph update, considering the previous graph from previous updates, incrementally, with edge weight updates for the repetition of words sequences, and also the addition of new nodes/words if new words happen to exist in the evolving text stream. Then, we use incremental PageRank algorithm to retrieve the words that have a higher importance in the text stream. These keywords are applied yet to a final post-processing stage of the stream to obtain a reasonable combination of keywords as in the original algorithm. Finally, as we do not need to accumulate unnecessary keywords from previous stream updates, and as we want the proposed system to be sensitive to the occurrence of new keywords in the text stream, we process the keywords in a stage where we maintain and update a *top-K* list of keywords with the space-saving algorithm previously described. The *top-K* algorithm application, based on landmark window, enables an efficient approach for large-scale text datasets. It focuses on the relevant keywords and discards less extracted keywords through time or keywords extracted from earlier snapshots. The alternative option for sliding windows (Gama, 2010) would not be appropriate for the *top-K* approach, since it may remove relevant keywords from the extracted *top-K* keywords. Those keywords might yet be included in the *top-K* keyword list we wish to maintain. In our scenario, *top-K* representation of text data streams implies knowing the K keywords of the simulated data stream from the database of publications and received after the post-processing stage of the streaming TextRank algorithms.

4 CASE STUDY AND EVALUATION

In this section, we present the method to analyze both window-based and incremental TextRank, regarding its comparison with the original TextRank. We introduce the reader to the data used for the tests. Finally, we describe the methodology to perform the comparison tests. We also state and introduce the metrics used to compare the algorithms regarding obtained keywords or keyphrases.

4.1 Description of the Data

In this case study, for algorithms efficiency measurements, we selected a dataset publicly available for research purposes by Corney et al. (2016). This dataset has a high amount of information including Reuters news and also several articles extracted from Blogs. The data contains articles titles, content, type of publication (which might include news or Blog posts), the source institution and also date and time of publishing. The high quality and high organization of this structured data make it a good source for text mining or NLP tasks. We selected all Reuters news, from the 1st to the 30th of September 2015. This corresponds to 400 news articles. Regarding the used text, we could choose between the news Titles or the news content to analyze. We choose the news content in all our studies. For qualitative measurements, we selected R&D publications from Czech Republic researchers. The dataset was publicly available in RDICCR

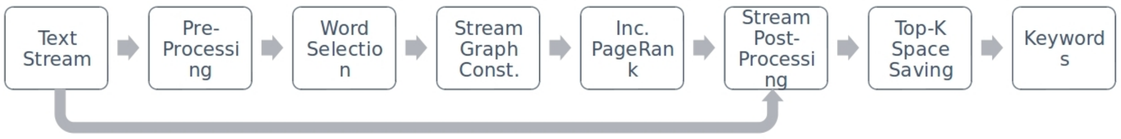

Figure 3: Incremental TextRank workflow.

(2015) when accessed online, in 2015. This is a complete source of information, and there was information for dozens of years starting from around 1985 until 2015. This dataset provides annotated keywords for each publication. During 2015, while this dataset was publicly available, we exported the data for several years, we selected conference papers and book chapters only. Therefore, the number of publications was reduced from around 25000 to 5110 publications.

4.2 Methodology

The Window-based TextRank and the incremental TextRank algorithm were evaluated in an incremental setup, for both the developed versions. Additionally, both algorithms were tested and evaluated regarding Processing Efficiency and also quality.

4.2.1 Processing Efficiency

Results for both streaming TextRank versions were obtained from several increments of text news, compared with the original batch algorithm, using the original TextRank (now on called Batch). The proposed TextRank versions, the Window-based implementation (now on called Window-based) and the incremental (now on called incremental) were tested regarding efficiency in an incremental setting configuration. The original batch results served as a baseline. In this setup, 6 snapshots of news were considered. In total, the first 20 days of September 2015, corresponding to 300 news articles, were passed as input. The snapshots were built by aggregating publications on a daily basis, by using the timestamps available in the dataset. In the Batch algorithm, for every snapshot, the keywords were extracted having all daily news since the day 1 to the current day as input. For the incremental algorithm, in the first snapshot it receives only 15 days of news articles as input. In the following snapshots, the algorithm only receives the set of publications text added to the corpus in that particular day snapshot (incremental). The empirical evaluation performed consisted mainly of comparing run times of each increment (duration of each increment and cumulative execution time) for all versions of TextRank. Additionally, the size of the corpus (number of unique words), and the total number of extracted keywords for each snapshot was also registered. Note that the batch algorithm will always need to extract keywords for all the text, while it is expected that the Window-based version only works with the current snapshot data, and not regarding previous snapshots information except the *top-K* list of keywords that needs to be updated between snapshots. Additionally, the incremental algorithm only performs calculations and keyword extraction for the current snapshot, and particularly for the words affected. Nonetheless, as previously explained, the incremental versions takes into account graph and corpus information from previous snapshots. In the end, an analysis of the total speed-up ratio obtained in each of the steps is added.

4.2.2 Qualitative Comparison

Regarding qualitative results, we used the Czech Republic R&D Dataset with annotated keywords. For this purpose, we selected only articles in proceedings, for 2013. Then, we aggregated the publications by month, in 12 snapshots. For this amount of publications, we selected the abstracts for our automatic keyword extraction tasks. On average, the abstracts have 240 words. Finally, we calculated the 12 measures of similarity between annotated and extracted keywords of all the versions of TextRank. We use a similarity measure to compare the author's keywords and the model results for keywords extraction.

This similarity measurement is more appropriated to test unsupervised keyword extraction algorithms. For example, imagine if we had the annotated keyword "car maintenance" for a text. Additionally, we automatically extracted the keywords "car washing" and "car cleaning" from the text. In this situation, if we used Precision, Recall and F measure we would have no idea that the extraction algorithm is, in fact, extracting concept keywords that are not that much far from the annotated keywords. Thus, by decomposing n-gram keywords we can detect these normal situations when we are dealing with annotated keywords. Particularly for this example, we would have a value of similarity of:

$$similarity_{K,M} = \frac{common^2}{c(w_K)*c(w_M)} = \frac{1^2}{2*2} = \frac{1}{4} = 0.25 \quad (1)$$

where *common* is the intersection of both groups, i.e., the number of words that appear in both the results and the keywords. $c(w_K)$ and $c(w_M)$ are the total number of words the author's keywords group has and the method (for all TextRank algorithms) provides, respectively. For all the experimentation and development, we used an Intel® Core^{TM} i7-7700HQ CPU @ 2.80GHz x 8 processors computer with 16 GBytes of RAM, SSD HDD, and Ubuntu 16.04.3 LTS 64-bit OS. Three runs per algorithm were performed. The values presented in the following charts in this paper are the average values of those three runs.

5 RESULTS

We present the results for all versions of the algorithm here in this section. All algorithms are compared with the Batch version of TextRank. The comparisons regard qualitative measurements and efficiency.

5.1 Processing Efficiency

Figure 4 shows the results for the Reuters News data. From "#Graph Nodes" and "#Words in Text" we can, as expected, conclude that the number of TextRank graph nodes is similar for the batch and incremental version, and variable for the Window-based version of the algorithm. Figure 4 (#Extracted Keywords) shows that the batch version retrieves a growing number of keywords while our streaming implementations retrieve the top-1000 keywords. Figure 4 (Elapsed Time) shows that all algorithms have variable elapsed time for each of the iterations. Nonetheless, both streaming algorithms have a clear decreasing elapsed time compared to the batch algorithm which tends to increase elapsed time as the quantity of the data increases. Otherwise, both streaming versions tend to achieve much lower values. For example, on iteration 4, the Window-based algorithm delivers a very low elapsed time, almost instantly processing of incoming data. The incremental version takes around 100 seconds to perform the same incoming data processing, and the batch version takes approximately 900 seconds. Figure 4 (Cumulative Time) shows a clear advantage of using incremental or Window-based versions of the algorithm regarding total elapsed time/cumulative time when the data is growing, and there is a need to analyze all data since day 1. In the last iteration, both Window-based and the incremental versions take around 10000 seconds to process all the 20 days of data. Otherwise, if we used the original TextRank algorithm to recurrently process all the data and also the incoming text, it would take almost 50000 seconds to achieve results. Figure 4 (Speedup Ratio) shows that both streaming algorithms, the incremental and the Window-based, show growing and significant speedups when we compare cumulative time. At the end of the stream, i.e., the last iteration, the Window-based approach provides a speedup of more than 6 times, and the incremental version achieves a speedup of almost 5 times the speed of processing provided by the original TextRank algorithm.

5.2 Qualitative Comparison

Regarding qualitative comparison, the Table 1 shows the similarity values between extracted keywords and accumulated annotated keywords as the abstracts text data evolves. We selected the top-1000 keywords of the batch algorithm, regarding keyword weight. Therefore, we compare the similarity values for these 1000 keywords obtained from the batch, Window-based and incremental algorithms. To measure the sensitivity to capture recent keywords we repeat the test, but this time we measure the similarity of extracted keywords with each snapshot annotated keywords. This way we can measure the sensibility of all algorithms to the current keywords in the stream. Table 2 shows these results. As we expected, in both tables of results, the incremental version of the algorithm presents higher similarity average. This is even more pronounced in the second table as expected. We should also note, as expected in the second table, that the maximum similarity values for the streaming versions of the algorithms are obtained in the last snapshots. This is explained by the higher sensibility of both proposed algorithms to more recent keywords.

6 DISCUSSION

Regarding processing efficiency, it is clear that if the TextRank user is dealing with streaming data, the use of incremental or Window-based versions of the algorithm is imperative. The Window-based version presents a more pronounced speedup, and this is expected as the processed data is variable instead of incremental. Still, regarding cumulative time, the incremental version of the algorithm has a learning curve. Thus, as the data arrives and the graph construction and update happens, the algorithm will eventually need fewer updates of graph structure in the future. This typically results in a logarithmic type of cumulative time curve. This is the result of less and less needed addition of nodes, resulting in the need to only update the weight of edges between nodes, i.e., the counting of words pairs. Regarding qualitative results of both streaming proposals, we would like to note that, if the researcher using our proposed algo-

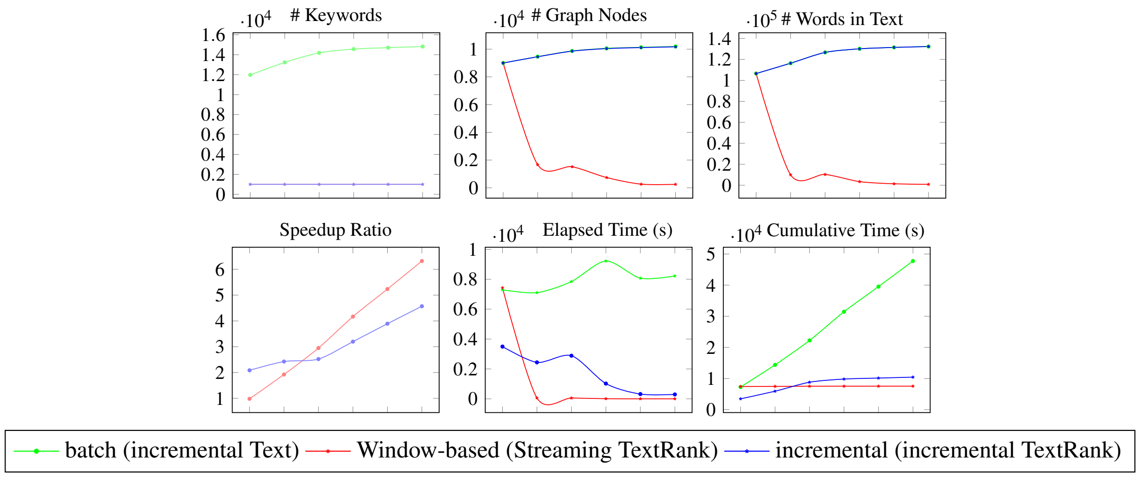

Figure 4: Efficiency results with the Reuters News Dataset (for all algorithms).

Table 1: Qualitative measures of similarity with accumulated annotated keywords.

Algorithm	Avg.	Sd	iter1	iter2	iter3	iter4	iter5	iter6	iter7	iter8	iter9	iter10	iter11	iter12
Batch	0.168	0.108	**0.023**	0.025	0.032	0.067	0.130	0.254	0.308	0.300	0.260	**0.218**	0.204	**0.193**
Window-based	0.114	0.059	**0.023**	0.045	0.042	0.069	0.125	0.173	0.186	0.207	0.136	0.127	0.134	0.102
Incremental	**0.198**	**0.125**	0.011	**0.046**	**0.046**	**0.124**	**0.189**	**0.308**	**0.350**	**0.398**	**0.304**	0.217	**0.206**	0.181

Table 2: Qualitative measures of similarity with current snapshot annotated keywords.

Algorithm	Avg.	Sd	iter1	iter2	iter3	iter4	iter5	iter6	iter7	iter8	iter9	iter10	iter11	iter12
Batch	0.160	0.118	**0.023**	0.008	0.003	0.026	0.086	0.221	0.280	0.254	0.267	0.250	0.270	0.235
Window-based	0.155	0.103	**0.023**	**0.043**	0.016	0.047	0.099	0.183	0.202	0.267	0.211	0.239	0.299	0.237
Incremental	**0.254**	**0.181**	0.011	0.032	**0.023**	**0.105**	**0.133**	**0.311**	**0.331**	**0.480**	**0.336**	**0.371**	**0.425**	**0.487**

rithms is not too preoccupied with the small decrease of quality of extracted keywords from the stream, then he/she can use the Window-based version of the algorithm for fast extraction. Nonetheless, if the researcher needs assured quality of extracted keywords, the incremental version proved to be a better solution when compared to the batch and the Window-based approach for keyword extraction. Additionally, the effect of increasing K in both these *top-K* approaches would intuitively increase the quality results as it is expected that, in Table 1, as the annotated keywords amount increase we continue to test only 1000 keywords extracted with the algorithms. Accordingly, in Table 1, the quality values decrease due to this characteristic, approximately after iteration 8, until the end of the stream, and for all algorithms.

7 CONCLUSIONS

In this paper, we introduce a Window-based and an incremental version of the TextRank system. The Window-based version proved to be similar in quality and much faster than the original TextRank algorithm, in a streaming context. Through the careful transformation of the original algorithm to an incremental version, we achieve a more efficient version and with better quality. Additionally, we introduce a *top-K* feature in our improvement, which allows storing the keywords that are more relevant. As future work, we would like to compare this incremental TextRank system with other systems prepared for text streams. Additionally, we would like to use this streaming version of the algorithm in a prototype prepared to extract visual information from affinity networks of authors, with text mined from their document production.

ACKNOWLEDGMENTS

This work is financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within project “POCI-01-0145-FEDER-006961”, and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013. Rui Portocarrero Sarmento also gratefully acknowledges funding from FCT (Portuguese Foundation for Science and Technology) through a PhD grant (SFRH/BD/119108/2016)

REFERENCES

Bracewell, D. B., Ren, F., and Kuriowa, S. (2005). Multilingual single document keyword extraction for information retrieval. In *2005 International Conference on Natural Language Processing and Knowledge Engineering*, pages 517–522.

Brazdil, P., Trigo, L., Cordeiro, J., Sarmento, R., and Valizadeh, M. (2015). Affinity mining of documents sets via network analysis, keywords and summaries. *Oslo Studies in Language*, 7(1).

Charikar, M., Chen, K., and Farach-Colton, M. (2002). Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, ICALP '02, pages 693–703, London, UK, UK. Springer-Verlag.

Corney, D., Albakour, D., Martinez, M., and Moussa, S. (2016). What do a million news articles look like? In *Proceedings of the First International Workshop on Recent Trends in News Information Retrieval co-located with 38th European Conference on Information Retrieval (ECIR 2016), Padua, Italy, March 20, 2016.*, pages 42–47.

Demaine, E. D., López-Ortiz, A., and Munro, J. I. (2002). Frequency estimation of internet packet streams with limited space. In *Algorithms-ESA 2002*, pages 348–360. Springer.

Desikan, P., Pathak, N., Srivastava, J., and Kumar, V. (2005). Incremental page rank computation on evolving graphs. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, WWW '05, pages 1094–1095, New York, NY, USA. ACM.

Gama, J. (2010). *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition.

Gazendam, L., Wartena, C., and Brussee, R. (2010). Thesaurus based term ranking for keyword extraction. In *Database and Expert Systems Applications, DEXA, International Workshops, Bilbao, Spain, August 30 - September 3, 2010*, pages 49–53.

Litvak, M., Last, M., Aizenman, H., Gobits, I., and Kandel, A. (2011). *DegExt — A Language-Independent Graph-Based Keyphrase Extractor*, pages 121–130. Springer Berlin Heidelberg, Berlin, Heidelberg.

Liu, Z., Li, P., Zheng, Y., and Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 257–266, Stroudsburg, PA, USA. Association for Computational Linguistics.

Manku, G. S. and Motwani, R. (2002). Approximate frequency counts over data streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*.

Metwally, A., Agrawal, D., and El Abbadi, A. (2005). Efficient computation of frequent and top-k elements in data streams. In *Proceedings of the 10th International Conference on Database Theory*, ICDT'05, pages 398–412, Berlin, Heidelberg. Springer-Verlag.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia.

RDICCR (2015). Published data from r&d is of the czech republic - research, development and innovation council of the czech republic (rdiccr). http://www.isvav.cz/. Accessed: 2015-09-30.

Rose, S., Engel, D., Cramer, N., and Cowley, W. (2010). Automatic keyword extraction from individual documents. In Berry, M. W. and Kogan, J., editors, *Text Mining. Applications and Theory*, pages 1–20. John Wiley and Sons, Ltd.

Shin, Y., Ryo, C. Y., and Park, J. (2014). Automatic extraction of persistent topics from social text streams. *World Wide Web*, 17(6):1395–1420.

Siddiqi, S. and Sharan, A. (2015). Article: Keyword and keyphrase extraction techniques: A literature review. *International Journal of Computer Applications*, 109(2):18–23. Full text available.