









YAKE! Collection-Independent Automatic Keyword Extractor

Ricardo Campos^{1,2} , Vítor Mangaravite² , Arian Pasquali² ,
Alípio Mário Jorge^{2,3} , Célia Nunes⁴ , and Adam Jatowt⁵ 

¹ Polytechnic Institute of Tomar, Tomar, Portugal
ricardo.campos@ipt.pt

² LIAAD – INESC TEC, Porto, Portugal
{vima, arrp}@inesctec.pt

³ DCC – FCUP, University of Porto, Porto, Portugal
amjorge@fc.up.pt

⁴ University of Beira Interior, Covilhã, Portugal
celian@ubi.pt

⁵ Kyoto University, Kyoto, Japan
adam@dl.kuis.kyoto-u.ac.jp

Abstract. In this paper, we present YAKE!, a novel feature-based system for multi-lingual keyword extraction from single documents, which supports texts of different sizes, domains or languages. Unlike most systems, YAKE! does not rely on dictionaries or thesauri, neither it is trained against any corpora. Instead, we follow an unsupervised approach which builds upon features extracted from the text, making it thus applicable to documents written in many different languages without the need for external knowledge. This can be beneficial for a large number of tasks and a plethora of situations where the access to training corpora is either limited or restricted. In this demo, we offer an easy to use, interactive session, where users from both academia and industry can try our system, either by using a sample document or by introducing their own text. As an add-on, we compare our extracted keywords against the output produced by the IBM Natural Language Understanding (IBM NLU) and Rake system. YAKE! demo is available at <http://bit.ly/YakeDemoECIR2018>. A python implementation of YAKE! is also available at PyPi repository (<https://pypi.python.org/pypi/yake/>).

Keywords: Keyword extraction · Information extraction · Text mining

1 Introduction

While considerable progress has been made over the last few years, the task of extracting meaningful keywords is yet to be solved, as the effectiveness of existing algorithms is still far from the ones in many other core areas of computer science. Most traditional approaches follow a supervised methodology, which largely depends on having access to training annotated text corpora. One of the first approaches has been proposed by Turney [5] who developed a custom-designed algorithm named GenEx. The great majority of the approaches developed so far, relied however, on supervised

methods such as Naïve Bayes as a way to select relevant keywords. Arguably the most widespread implementation of such an approach is KEA [7] which uses the Naïve Bayes machine learning algorithm for keyword extraction. Despite their often-superior effectiveness, the main limitation of supervised methods is their relatively long training time process. This contrasts with general unsupervised algorithms [3, 4, 6], which may be quickly applied to documents across different languages or domains in a short time span and which demand reduced effort due to their plug and play nature. In this paper, we describe YAKE!, an online keyword extraction demo which builds upon text statistical features extracted from a single document to identify and rank the most important keywords. While keywords extraction systems have been extensively studied over the last few years, multilingual online single document tools are still very rare. YAKE! is an attempt to fill this gap. Overall, it provides a solution which does not need to be trained on a particular set of documents, and thus can be easily applied to single texts, regardless of the existence of a corpus, dictionary or any external collection. In an era of massive but likely unlabeled collections, this can be a great advantage over other approaches, particularly supervised ones. Another important feature is that YAKE! does not use NER nor PoS taggers, which makes the system to be language-independent, except for the use of different but static lists of stopwords for each language. This enables an easy adaptation of YAKE! to other languages other than English, especially, to minor languages for which open source language processing tools are scarce. It is an advantage over supervised methods, which demand training a custom model beforehand. Finally, the fact that YAKE! relies only on statistical features extracted from the text itself allows for easily scaling to vast collections.

2 Keyword Extraction Pipeline

The proposed system has six main components: (1) Text pre-processing; (2) Feature extraction; (3) Individual terms score; (4) Candidate keywords list generation; (5) Data Deduplication; and (6) Ranking. In the following we will provide a concise description of each of the six steps as a detailed discussion of them is beyond the scope of this paper and can be found on [1]. First, we apply a pre-processing step which splits the text into individual terms whenever an empty space or a special character (e.g., line breaks, brackets, comma, period, etc.) delimiter is found. Second, we devise a set of five features to capture the characteristics of each individual term. These are: (1) *Casing*; (2) *Word Positional*; (3) *Word Frequency*; (4) *Word Relatedness to Context*; and (5) *Word DifSentence*. The first one, *Casing*, reflects the casing aspect of a word. *Word Positional* values more those words occurring at the beginning of a document based on the assumption that relevant keywords often tend to concentrate more at the beginning of a document. *Word Frequency* indicates the frequency of the word, scoring more those words that occur more often. The fourth feature, *Word Relatedness to Context*, computes the number of different terms that occur to the left (resp. right) side of the candidate word. The more the number of different terms that co-occur with the candidate word (on both sides), the more meaningless the candidate word is likely to be. Finally, *Word DifSentence* quantifies how often a candidate word appears within different sentences. Similar to *Word Frequency*, *Word DifSentence*

values more those words that often occur in different sentences. Both features however, are combined with *Word Relatedness to Context*, meaning that the more they occur in different sentences the better, as long as they do not occur frequently with different words on the right or left side (which would resemble a behavior close to the one of stopwords). In the third step, we heuristically combine all these features into a single measure such that each term is assigned a score $S(w)$. This weight will feed the process of generating keywords which is to be taken in the fourth step. Here, we consider a sliding window of 3-grams, thus generating a contiguous sequence of 1, 2 and 3-gram candidate keywords. Each candidate keyword will then be assigned a final $S(kw)$, such that the smaller the score the more meaningful the keyword will be. Equation 1 formalizes this:

$$S(kw) = \frac{\prod_{w \in kw} S(w)}{TF(kw) * (1 + \sum_{w \in kw} S(w))} \quad (1)$$

where $S(kw)$ is the score of a candidate keyword, determined by multiplying (in the numerator) the score $S(w)$ of the first term of the candidate keyword by the subsequent scores of the remaining terms. This is divided by the sum of the $S(w)$ scores to average out with respect to the length of the keyword, such that longer n -grams do not get benefited just because they have a higher n . The result is further divided by $TF(kw)$ -term frequency of the keyword - to penalize less frequent candidates. In the fifth step, we eliminate similar candidates coming from the previous steps. For this, we use the *Levenshtein distance* [2]. Finally, the system will output a list of relevant keywords, formed by 1, 2, 3-grams, such that the lower the $S(kw)$ score the more important the keyword will be.

3 Demonstration Overview

In this demonstration, we highlight some of the major features of YAKE!, in particular, its independence with regards to a training corpus, dictionary, size of the text, languages and domains. The online demo of YAKE! can be accessed at <http://bit.ly/YakeDemoECIR2018>. A python implementation of YAKE! is also available at <https://pypi.python.org/pypi/yake/> meaning that our method can already be used and imported as a library. During the demonstration, we will showcase the behavior of our system on different kinds of datasets with various types of settings and we will show the audience how to interact with YAKE! All the results are immediately put side by side with the IBM NLU commercial solution and Rake [4] system for a comparison. In a nutshell, users can interact and test YAKE! under 3 different scenarios. For the first one, they can try the system by selecting a pre-chosen text from six datasets (500 N-KPCrowd-v1.1, INSPEC, Nguyen 2007, SemEval 2010, PubMed, 110-PT-BN-KP), the first five in English, the latter in Portuguese. Texts are from the scientific domain (including short abstracts, medium and large-size texts), TV broadcast and news articles. All the results can be compared against a ground-truth. Therefore, users can analyze the effectiveness of our system with regards to different domains, sizes and languages of the input text. For the second scenario, we arbitrarily choose as input to our system, a set of sample

texts from different domains (politics, culture, history, tourism, technology, religion, economy, sports, education and biography) and languages (English, Portuguese, German, Italian, Netherland, Spanish, French, Turkish, Polish, Finnish and Arabic), thus enriching our demo with a range of text examples with characteristics different than those of formal datasets. Finally, we offer the user the chance to test YAKE! in an online environment and in real time with his/her own text, to see how it responds to different scenarios/texts and languages/domains. Users can input their text either by hand (copy/paste), or by referring to its URL. PDF files are also accepted. As a rule of thumb, the maximum size of n-grams is set to 3. However, this parameter can be adjusted by the user on the opening page. As an add-on, we also offer researchers access to a web service (under the API tab) so that our system can be used for research purposes. A screenshot of the results obtained for a document extracted from the INSPEC dataset is shown in Fig. 1.

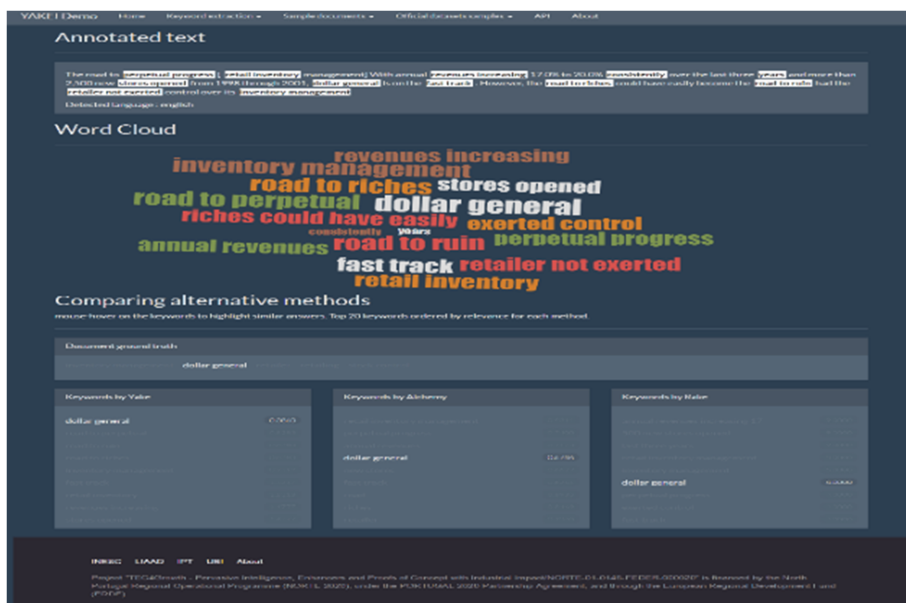


Fig. 1. YAKE! interface.

The results of our demo can be explored through three different functionalities: (1) annotated text; (2) word cloud; and (3) comparing YAKE! with alternative methods. The first one shows the text annotated with the top 10 keywords retrieved by YAKE!. The second, uses the relevance score of each keyword retrieved by YAKE!, to generate a word cloud, where more important keywords are given a higher size. Finally, we compare the results of YAKE! against IBM NLU and Rake [4]. Each result is assigned a relevance ranking value reflecting the importance of the keyword within the text. Note that in the case of YAKE!, the lower the value, the more important the keyword is. As a further additional feature, we offer the user to explore the results by

means of a mouse-hover feature which highlights both the keyword selected as well as similar keywords identified within the three systems. In the example in Fig. 1, we highlight a keyword from the ground-truth (“dollar general”) and observe its disposition within the three systems. Though anecdotal, this example shows that Yake! is able to list this keyword in the 1st position. A formal evaluation however is needed to take valid conclusions.

Acknowledgements. This work is partially funded by the ERDF through the COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT as part of project UID/EEA/50014/2013 and of project UID/MAT/00212/2013. It was also financed by MIC SCOPE (171507010) and by Project “TEC4Growth - Pervasive Intelligence, Enhancers and Proofs of Concept with Industrial Impact/NORTE-01-0145-FEDER-000020” which is financed by the NORTE 2020, under the Portugal 2020, and through the ERDF.

References

1. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., Jatowt, A.: YAKE! collection-independent automatic keyword extractor. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018, LNCS, vol. 10772, pp. 806–810. Springer, Cham (2018)
2. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl* **10**(8), 707–710 (1966)
3. Mihalcea, R., Tarau, P.: TextRank: bringing order into texts. In: EMNLP 2004, pp. 404–411, Barcelona, Spain, 25–26 July 2004
4. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. In: *Text Mining: Theory and Applications* (2010)
5. Turney, P.: Learning algorithms for keyphrase extraction. *Inf. Retr. J.* **2**(4), 303–336 (2000)
6. Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: AAAI 2008, pp. 855–860, 13–17 July 2008
7. Witten, I., Paynter, G., Frank, E., Gutwin, C., Nevill-Manning, C.: KEA: practical automatic keyphrase extraction. In: JCDL 2004, pp. 254–255, 7–11 June 1999