



UNIVERSITÀ  
DEGLI STUDI  
DI UDINE

## Università degli studi di Udine

### Exploiting and Evaluating a Supervised, Multilanguage Keyphrase Extraction Pipeline for Under-Resourced Languages

*Original*

*Availability:*

This version is available <http://hdl.handle.net/11390/1129380> since 2021-03-28T12:19:40Z

*Publisher:*

*Published*

DOI:10.26615/978-954-452-049-6\_012

*Terms of use:*

The institutional repository of the University of Udine (<http://air.uniud.it>) is provided by ARIC services. The aim is to enable open access to all the world.

*Publisher copyright*

(Article begins on next page)

# Exploiting and Evaluating a Supervised, Multilanguage Keyphrase Extraction Pipeline for Under-Resourced Languages

Marco Basaldella<sup>\*</sup> and Muhammad Helmy<sup>†</sup> and Elisa Antolli  
Mihai Horia Popescu and Giuseppe Serra<sup>\*</sup> and Carlo Tasso<sup>\*</sup>

Artificial Intelligence Laboratory, University of Udine  
Via Delle Scienze, 208, Udine, Italy

<sup>\*</sup>{name.surname}@uniud.it, <sup>†</sup>alameldien.muhammad@spes.uniud.it,  
antolli.elisa@spes.uniud.it, popescu.mihaihoria@spes.uniud.it

## Abstract

This paper evaluates different techniques for building a supervised, multilanguage keyphrase extraction pipeline for languages which lack a gold standard. Starting from an unsupervised English keyphrase extraction pipeline, we implement pipelines for Arabic, Italian, Portuguese, and Romanian, and we build test collections for languages which lack one. Then, we add a Machine Learning module trained on a well-known English language corpus and we evaluate the performance not only over English but on the other languages as well. Finally, we repeat the same evaluation after training the pipeline over an Arabic language corpus to check whether using a language-specific corpus brings a further improvement in performance. On the five languages we analyzed, results show an improvement in performance when using a machine learning algorithm, even if such algorithm is not trained and tested on the same language.

## 1 Introduction

Automatic Keyphrase Extraction (herein AKE) is the task of extracting “a short list of phrases (typically from five to fifteen noun phrases) that captures the main topics discussed in a given document” (Turney, 2000). Recently, Automatic keyphrase extraction has received a lot of attention, because it has been successfully used in many natural language processing (hence NLP) and information retrieval tasks, such as text summarization (Zhang et al., 2004) or document clustering (Hammouda et al., 2005).

The AKE problem is tackled with different techniques. For example, researchers proposed

solutions using e.g. supervised Machine Learning algorithms (herein ML), graph-based ranking algorithms, or clustering techniques, the first being the most successful one (Hasan and Ng, 2014; Merrouni et al., 2016).

Resources to train and evaluate AKE algorithms are available for a variety of domains, like scientific papers, abstracts, newswire texts, emails, etc. (Hasan and Ng, 2014). However, AKE is typically performed on the English language, mainly due to the fact that datasets were not available in other languages. Recently, though, there has been a surge in interest in building AKE datasets in different languages, like French or Arabic (Bougouin et al., 2016; Helmy et al., 2016a).

Following this path, in this paper we propose a multilingual keyphrase extraction pipeline, which performs keyphrase extraction in English, Arabic, Italian, Portuguese and Romanian. However, while resources for English and Arabic are readily available, to our knowledge for the latter three languages there are no publicly available AKE datasets. Nevertheless, we show that it is actually possible to build an AKE pipeline for a language which lacks a gold standard. To prove our hypothesis, we first train a ML model that can extract keyphrases in English and Arabic. Then, we use that model to extract keyphrases in the other languages. Finally, we validate the proposed solution using expert knowledge.

## 2 Related Work

The problem of multilinguality in AKE is not new. It started in late 90’s in order to provide approaches for IR users to handle multilingual documents: for example, Tseng (1998) built an unsupervised, language-independent AKE system and demonstrated its effectiveness on English and Chinese. Since then, many language-independent ap-

proaches have been proposed, but most of them are still unsupervised or require the collection of ad-hoc corpora when evaluated on languages other than English.

For example, DegExt (Litvak et al., 2013) was introduced as an unsupervised language independent keyphrase extractor. DegExt uses a simple graph-based syntactic representation of text and web documents (Schenker et al., 2005). The evaluation was performed on an English corpus (DUC 2002) and one purpose-built corpus of 50 Hebrew documents.

Bougouin et al. (2013) proposed TopicRank as an unsupervised, language-independent AKE system. TopicRank creates a graph of the document where each node is a topic appearing in the document, then it uses graph ranking techniques to score the topics and select keyphrases belonging to the top ranked topics. The authors evaluate their approach on four datasets, two on the English language (the SEMEVAL 2010 corpus (Kim et al., 2010) and the Inspec dataset (Hulth, 2003)), and two on the French language, one freely available and one purpose-built by them.

DIKpE-G (Degl’Innocenti et al., 2014) was proposed as a novel multi-language, unsupervised, knowledge-based approach towards keyphrase generation. DIKpE-G integrates several kinds of knowledge for selecting and evaluating meaningful keyphrases, ranging from linguistic to statistical, meta/structural, social, and ontological knowledge, and it has been evaluated on the Italian language using a custom-built dataset of 50 scientific papers.

Finally, LIKE (Aquino et al., 2013) was presented as a supervised method that uses feed-forward neural networks for automatically extracting keywords from a document regardless of the language used in it. While the authors claim that LIKE is a truly language-independent AKE system, it is trained and evaluated only on the English language.

### 3 Multilanguage Keyphrase Extraction

To build an AKE system for a specific language the first step is to understand which parts of the pipeline needs to be adapted. Therefore, we divide the AKE pipeline in modules and we identify the language-dependent ones:

1. Low-level NLP: sentence and word segmentation, part-of-speech tagging and stemming;
2. Candidate generation: selection of the possible keyphrases in the document. It is usually performed by detecting the phrases which match certain known part-of-speech tags patterns;
3. Feature extraction: candidates are assigned some features, like position in the text, frequency, etc.
4. Candidate scoring: the feature extracted in the previous step are used to assign a score to the candidates; then, the top ranked candidates are usually used to evaluate the pipeline.

In this work we focus on the first, second and fourth steps, which are the ones which rely mostly on language-specific knowledge, by implementing several AKE pipelines in the Distiller keyphrase extraction framework (Basaldella et al., 2015).

#### 3.1 Low-level NLP

The first step of the AKE pipeline consists in preparing the document to identify the potential keyphrases, by splitting the text into sentences and the sentences into tokens and, finally, performing part-of-speech tagging, stemming and/or lemmatization.

We use off-the-shelf libraries to perform these tasks. For the English and Arabic languages, we used the Stanford CoreNLP library for all the tasks except stemming, since it offers state-of-the-art performance. Due to the limited availability of languages available for CoreNLP, for the Portuguese and Italian languages we used the the Apache OpenNLP<sup>1</sup> library with the default models for Portuguese and Ciapetti’s models for the Italian language<sup>2</sup>. For the Romanian language, the models were not available, so we built them ourselves; we describe this process in Section 3.1.1.

To stem the tokens, we used the Tartarus stemmer (Porter, 1980) for all languages but Arabic, where lemmatization was used instead of stemming. Thus, the lemmatizer used for the Arabic language is the AraMorph lemmatizer (Buckwalter, 2002).

##### 3.1.1 Romanian

We were not able to find any suitable CoreNLP or OpenNLP models for the Romanian language to

<sup>1</sup><https://opennlp.apache.org/>

<sup>2</sup><https://github.com/aciapetti/opennlp-italian-models>

perform sentence splitting, tokenization, and PoS tagging. Thus, we decided to build our own models for Apache OpenNLP using the ROMBAC<sup>3</sup> dataset (Ion et al., 2012).

The corpus contains about 41,000,000 words including punctuation and it is divided in five domains: journalism, pharmaceuticals and medicine, law, biographies of Romanian literary personalities, and fiction. We tested different training/testing split to obtain the best possible performance. For the training of the sentence detector and tokenizer, we used the journalism domain, while to train the POS tagger we used the journalism, medicine, and fiction domains.

### 3.1.2 Arabic

Being very different from the Western languages we previously described (Farghaly and Shaalan, 2009; Habash, 2010), the Arabic language needed additional text preprocessing steps.

The text is cleaned by removing all unnecessary characters like the special Arabic punctuation marks, diacritics, and Kashida<sup>4</sup>. In addition, some Arabic characters have various forms which we normalize into a single one to decrease the processing complexity.

Another issue about Arabic is that punctuation is used differently than in English and other Western languages. In fact, Arabic has traditionally no punctuation, and it is still usual to find modern Arabic books written in this way (Dickins et al., 2016). In the other languages we analyze, we trivially assume that all the words of a keyphrase have to appear within the same sentence. Since it may be not possible to distinguish sentences in an Arabic input text, we follow the approach described in (Helmy et al., 2016b), assuming that the tokens of a keyphrase in the Arabic language should appear in the same syntactic noun phrase.

## 3.2 Candidate Generation

Candidate generation requires more domain knowledge than simply using an off-the-shelf library. To generate the candidate keyphrases, we scan the text for phrases that match certain part-of-speech tag sequences (we will call such sequences *PoS patterns* from now on). For example, for the present document, a valid keyphrase may be “mul-

tilingual keyphrase extraction”, which PoS pattern is “(adjective, noun, noun)”.

These patterns are typical of the AKE task and they require to be engineered by a domain expert, since they are significantly different from language to language. For example, the English phrase “*software engineering*” is translated in Italian as “*ingegneria del software*”, where *del* is an articulated preposition, i.e. the union of an article with a preposition, a part of speech which does not exist in the English language. This example shows also that in some languages we will look for shorter *n*-grams, while other, more verbose languages may require a larger *n*. For example, in English we look typically up to 3-grams (Pudota et al., 2010; Kim et al., 2010), while in Italian our system will look for 1 to 5-grams.

For the English, Arabic, and Italian language, we used known PoS patterns from the literature (Basaldella et al., 2016; Helmy et al., 2016b; Basaldella et al., 2015). For the other languages, we were not aware of existing PoS patterns. For this purpose, we collected about 500 author assigned keyphrases both in Portuguese and Romanian from scientific repositories and we performed PoS tagging on them. Then, we picked the most frequent patterns, after manually removing or correcting the erroneous ones (e.g. pattern with only conjunctions, etc.).

## 3.3 Features Extraction

After the identification of the candidate keyphrases we assign to each of them seven features. Most of the features assigned to candidate keyphrases rely simply on statistical and structural information (like the position in the text, frequency, TF-IDF, and so on) (Hasan and Ng, 2014). While researchers have developed language-dependent features, based e.g. on part of speech tags (Hulth, 2003), on anaphora resolution (Basaldella et al., 2016) or based on external knowledge like Wikipedia (Medelyan et al., 2009), this kind of features requires long computational times and specialized tools which may not be available for all languages. Thus, we decided to leave them out of our pipeline.

In our system, we take four features from Pudota et al. (2010) and we use them for *unsupervised* keyphrase extraction only. These features are:

**Normalized Frequency** i.e. the number of times

<sup>3</sup>The Romanian Balanced Annotated Corpus

<sup>4</sup>Also called Tatweel, it is a form of Arabic text justification that, instead of adding whitespace, adds an horizontal, slightly curvilinear stroke between certain letters.

a candidate appears in the document normalized to the number of sentences;

**Height** i.e. the relative position of the first occurrence of the candidate in the document;

**Depth** i.e. the relative position of the last occurrence of the candidate in the document;

**Lifespan** i.e. the difference between the last and the first appearance of the candidate, i.e. given a candidate  $kp$  for a document  $d$ ,  $lifespan(kp, d) = depth(kp, d) - height(kp, d)$ .

We add three other features to this set to perform supervised keyphrase extraction, namely:

**Frequency** i.e. the number of times a keyphrase  $kp$  appears in a document  $d$ . While this feature may seem redundant with *normalized frequency*, our experiments showed that using both features leads to better results. To see why, just consider the word “candidate” in the present document: it has been repeated many times in the same sentence, e.g. in the definition of the *lifespan* feature;

**TF-IDF** i.e. one of the first features used for AKE along with *height* (Witten et al., 1999). A common statistic used in information retrieval to identify which candidates are peculiar of that particular document with respect to a corpus, is the product of the *term frequency*  $tf$  and the *inverse document frequency*  $idf$ ;

**DPM** i.e. Document Phrase Maximality, is used to discriminate between overlapping keyphrases and it helped to reach new state-of-the-art performance in the AKE task (Haddoud and Abdeddaim, 2014). Given a document  $d$  and the candidate keyphrase  $kp$ , we define the set  $sup(kp, d)$  of the *supertems* of  $kp$  candidate keyphrase, i.e. the set of the candidates that contain  $kp$  as a substring. For example, if we have a document  $d$  which talks about “*software engineering*”, we have that “*software engineering*”  $\in sup(“software”, d)$ .

### 3.4 Candidate Scoring

To score the candidates, the most trivial approach in this kind of AKE pipeline is to assign heuristically crafted weights to the features, like in Pudota

et al. (2010). Typically, however, one can train a machine learning algorithm over a dataset, like the SEMEVAL 2010 dataset (Kim et al., 2010), and use the generated model to identify keyphrases of new documents. Unfortunately, as pointed out in the introduction, datasets are available only for a minority of languages, so this is not always a viable option.

In our system we used three different scoring techniques. First, since we do not have training sets for all five languages, we assigned manual weights to four of our features, using values which have proven to work on the English language (Pudota et al., 2010). Then, we trained two models, one for English using the SEMEVAL 2010 dataset and one for Arabic using the AKEC dataset. Finally, we used the models trained on these languages to score keyphrases in Italian, Romanian and Portuguese as well.

#### 3.4.1 Manual Weights

In this approach we follow a very simple technique. Given a candidate keyphrase  $kp$ , a feature  $f$ , and the set of the features  $F$ , we define  $value(kp, f)$  as a function which returns the value of  $f$  for the candidate  $kp$ . For each feature, we also assign a weight  $w$ , whose value is defined below. Then, the score of a keyphrase  $kp$  is computed as follows:

$$score(kp) = \sum_{i=1}^{|F|} w_i \times value(kp, f_i)$$

As mentioned in Section 3.3, the features used in this step are *normalized frequency*, *height*, *depth*, and *lifespan*, and the values of their weights  $w_i$  are respectively 0.1, 0.32, 0.16 and 0.12, as presented in Pudota et al. (2010). We did not recompute the values of the weights by ourselves, because authors already proved its effectiveness, and since we just want to use them as a baseline.

#### 3.4.2 Supervised Weights

The manual weights technique is trivially limited by the fact that these weights are used to compute a simple linear function. Many machine learning algorithms are instead able to learn nonlinear functions, and for this reason are commonly used for AKE in literature (Hasan and Ng, 2014).

Thus, in our final step we train a multilayer neural network to extract keyphrases in English and Arabic, using two different training sets.



For the English language, we use the dataset from the “SEMEVAL-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles” challenge (Kim et al., 2010) (herein, simply SEMEVAL 2010). For the Arabic language, the network has been trained on the recent AKEC corpus (Helmy et al., 2016a). The corpora have 244 and 160 documents respectively, of which 144 are used for training and 100 for testing in the SEMEVAL 2010 dataset, and 100 are used for training and 60 for testing in the AKEC dataset (see Table 1).

The neural network has been trained using the `nnet` package in the R programming language using the `entropy` parameter. The network uses one neuron per input feature, a hidden layer with two times the input neurons, and one output neuron. The keyphrases are ranked according to the score assigned by the network, which is the value of the output neuron.

We use the models trained on these datasets to extract keyphrases in all our five languages. This is possible because the neural network ignored the text of the actual candidate keyphrase, since it does not receive any information about its words or about its meaning, but only statistical information about its appearance(s) in the input document.

## 4 Experimental Evaluation

While it is straightforward to analyze the performance of our model in English and Arabic since both the SEMEVAL 2010 and AKEC datasets provide test sets, for Italian, Portuguese and Romanian we are not aware of publicly available collections of keyphrase extraction datasets.

For this reason, we asked mother tongue speakers of these three languages to collect 20 documents per language and assign 15 keyphrases to each document, ranking them by importance. To have a further verification of our techniques, we did the same process for 20 documents in the English language as well. The collected datasets are described in Table 1. For English, Italian and Portuguese, we have collected similar datasets with a majority of scientific documents, which is reflected by mean of about 4000 words per document. For Romanian, we collected *mainly* newswire documents, so we have a mean of 800 words per document, close to the AKEC dataset. All the purpose-built datasets have a greater variability in the number of words with respect to the SEMEVAL 2010 and AKEC datasets, because

Dataset	Size	Mean length	$\sigma$ length
Semeval 2010	244	8020	1946
AKEC	160	757	145
English	20	3717	1877
Italian	20	4699	3412
Portuguese	20	4335	2482
Romanian	20	802	730

Table 1: The datasets used to train and test the pipelines, with their number of documents, their mean length in words, and the standard deviation  $\sigma$  of the length in words.

while these datasets are composed by only one kind of documents with strict constraints on the length, our test datasets are mixed, containing scientific papers, newswire text, web pages, etc.

For all three approaches, we evaluate our algorithms using Precision computed on the top 5 extracted keyphrases (herein Precision@5 or P@5), and Precision and F1-score computed on the top 15 extracted keyphrases (herein P@15 and F1@15) and Mean Average Precision (MAP). Note that for our own datasets, since they have only 15 expert-assigned keyphrases, the Precision score and the F1 score on the top 15 candidates are equal<sup>5</sup>, so for these datasets we show only the former.

## 5 Experimental Results

We present the results obtained in our experiments in Table 2. As expected, the manual weights method achieves the lowest performance. This is true in particular for the SEMEVAL 2010 dataset; on the other datasets, though, the performance is higher, with 40% and 46% P@5 score on the Portuguese language and Arabic language respectively. These scores seem to be particularly good, since the best performing system in the SEMEVAL 2010 challenge obtained 40% in P@5.

Using the English model we obtained better performance on the SEMEVAL 2010 dataset with 21% F-Score, which would be enough to be placed 9<sup>th</sup> over 19 systems in the challenge. Since we are not interested in getting the best score but we want to get an average AKE system, this result looks acceptable. Moreover, the score of our neural network greatly outperforms the manual baseline, so

<sup>5</sup>Because the size of the set of the retrieved documents is equal to the size of the set of the relevant documents, hence  $Precision = Recall$ .

Dataset	Pipeline	P@5	P@15	MAP	F1@15
<b>English (SEMEVAL 2010)</b>	Manual weights	0.13	0.1	0.076	0.10
	English model	<b>0.29</b>	<b>0.21</b>	<b>0.151</b>	<b>0.21</b>
	Arabic model	0.25	0.18	0.117	0.18
<b>Arabic (AKEC)</b>	Manual weights	0.46	0.37	0.158	0.145
	English model	<b>0.63</b>	0.47	0.185	0.18
	Arabic model	0.61	<b>0.48</b>	<b>0.190</b>	<b>0.19</b>
<b>English</b>	Manual weights	0.33	0.26	0.232	
	English model	<b>0.51</b>	<b>0.35</b>	<b>0.320</b>	
	Arabic model	0.47	0.32	0.280	
<b>Italian</b>	Manual weights	0.37	0.23	0.182	
	English model	<b>0.44</b>	<b>0.26</b>	<b>0.209</b>	
	Arabic model	0.41	0.23	0.185	
<b>Portuguese</b>	Manual weights	0.40	0.27	0.226	
	English model	0.42	0.24	0.221	
	Arabic model	<b>0.45</b>	<b>0.31</b>	<b>0.250</b>	
<b>Romanian</b>	Manual weights	0.34	0.26	0.229	
	English model	<b>0.47</b>	<b>0.3</b>	<b>0.266</b>	
	Arabic model	0.39	0.28	0.248	

Table 2: The results obtained with the different 15 experiment we executed.

were are satisfied and we decided to use it on the other languages.

The Arabic model obtains similarly satisfactory results on the AKEC dataset. We have no other systems to compare our model with, since the dataset has been recently released, but the 61% P@5 Score is hugely outperforming the best system in the SEMEVAL 2010 challenge and a 15% improvement over the manual weights, so we're satisfied with this result.

After we validated the machine learning models, we proceeded to use them for the other languages. Using the English and Arabic models to extract keyphrases in Italian, Portuguese, and Romanian offers *always* an improvement with respect to the manual weights, and the same holds for our English countercheck collection.

Analyzing the results for each language, we see that the English model outperforms the Arabic one on the English, Italian and Romanian language, while the Arabic model performs better on the Portuguese language only. Looking at the Precision@15 score, of particular interest is the English model on the English and Romanian collections, and the Arabic model again on the English collection and on the Portuguese one, all of which reach and/or surpass 30%, outperforming again the best performing systems on the SEMEVAL 2010 dataset.

Anyway, a direct comparison of our results with the one obtained in the SEMEVAL 2010 challenge is clearly not fair, since the documents in our collections are significantly shorter, so the problem we had to solve was easier (Hasan and Ng, 2014). This is the reason why our English model performs better on our collections than on the SEMEVAL 2010 dataset, the reason why the Arabic model performs better on the AKEC collection than on the other datasets, and a probable reason of the poor performance of the unsupervised weights, since Pudota et al. (2010) tailored them on documents with a different length.

## 6 Conclusions

Our approach showed that it is possible to build an effective supervised keyphrase extraction pipeline for an under-resourced language which lacks a keyphrase extraction gold standard by training it on another language. In fact, by training our AKE pipeline over English and Arabic, we were able to obtain good performance on Italian, Romanian and Portuguese as well.

As a future work, it should be considered to perform further experiments on documents in several languages coming from several domains and of different lengths, to further investigate if the performance of AKE depends more on the length or on the language of the document.

## References

- Germán Osvaldo Aquino, Waldo Hasperué, César Armando Estrebu, and Laura Cristina Lanzarini. 2013. A novel, language-independent keyword extraction method. In *XIX Congreso Argentino de Ciencias de la Computación*.
- Marco Basaldella, Giorgia Chiaradia, and Carlo Tasso. 2016. Evaluating anaphora and coreference resolution to improve automatic keyphrase extraction. In *Proc. of International Conference on Computational Linguistics (COLING)*.
- Marco Basaldella, Dario De Nart, and Carlo Tasso. 2015. Introducing distiller: a unifying framework for knowledge extraction. In *Proceedings of 1st AI\*IA Workshop on Intelligent Techniques At Libraries and Archives co-located with XIV Conference of the Italian Association for Artificial Intelligence (AI\*IA 2015)*. Associazione Italiana per l'Intelligenza Artificiale.
- Adrien Bougouin, Sabine Barreaux, Laurent Romary, Florian Boudin, and Beatrice Daille. 2016. Termith-eval: a french standard-based resource for keyphrase extraction evaluation. In *Proc. of International Conference on Language Resources and Evaluation (LREC)*.
- Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. [Topicrank: Graph-based topic ranking for keyphrase extraction](#). In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, Nagoya, Japan, pages 543–551. <http://www.aclweb.org/anthology/I13-1062>.
- Tim Buckwalter. 2002. Buckwalter {Arabic} morphological analyzer version 1.0 .
- Dante Degl'Innocenti, Dario De Nart, and Carlo Tasso. 2014. A new multi-lingual knowledge-base approach to keyphrase extraction for the italian language. In *Proc. of 6th International Conference on Knowledge Discovery and Information Retrieval (KDIR)*. pages 78–85.
- James Dickins, Sándor Hervey, and Ian Higgins. 2016. *Thinking Arabic translation: A course in translation method: Arabic to English*. Routledge.
- Ali Farghaly and Khaled Shaalan. 2009. Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)* 8(4):14.
- Nizar Y Habash. 2010. Introduction to arabic natural language processing. *Synthesis Lectures on Human Language Technologies* 3(1):1–187.
- Mounia Haddoud and Said Abdeddaim. 2014. [Accurate keyphrase extraction by discriminating overlapping phrases](#). *Journal of Information Science* 40(4):488–500. <https://doi.org/10.1177/0165551514530210>.
- Khaled M Hammouda, Diego N Matute, and Mohamed S Kamel. 2005. Corephrase: Keyphrase extraction for document clustering. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. Springer, pages 265–274.
- Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*.
- Muhammad Helmy, Marco Basaldella, Eddy Madalena, Stefano Mizzaro, and Gianluca Demartini. 2016a. Towards building a standard dataset for arabic keyphrase extraction evaluation. In *Proc. of 20th International Conference on Asian Language Processing (IALP)*. IEEE, pages 26–29.
- Muhammad Helmy, Dario De Nart, Dante Degl'Innocenti, and Carlo Tasso. 2016b. Leveraging arabic morphology and syntax for achieving better keyphrase extraction. In *Proc. of 20th International Conference on Asian Language Processing (IALP)*. IEEE, pages 340–343.
- Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proc. of the conference on Empirical methods in natural language processing*.
- Radu Ion, Elena Irimia, Dan Stefanescu, and Dan Tufis. 2012. Rombac: The romanian balanced annotated corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA), Istanbul, Turkey, pages 339–344.
- Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proc. of the International Workshop on Semantic Evaluation*.
- Marina Litvak, Mark Last, and Abraham Kandel. 2013. Degext: a language-independent keyphrase extractor. *Journal of Ambient Intelligence and Humanized Computing* 4(3):377–387.
- Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proc of Conference on Empirical Methods in Natural Language Processing*.
- Z. A. Merrouni, B. Frikh, and B. Ouhbi. 2016. Automatic keyphrase extraction: An overview of the state of the art. In *Proc. of IEEE International Colloquium on Information Science and Technology (CiSt)*.
- Martin F Porter. 1980. An algorithm for suffix stripping. *Program* 14(3):130–137.
- Nirmala Pudota, Antonina Dattolo, Andrea Baruzzo, Felice Ferrara, and Carlo Tasso. 2010. Automatic keyphrase extraction and ontology mining for content-based tag recommendation. *International Journal of Intelligent Systems* 25(12):1158–1186.



- Adam Schenker, Abraham Kandel, Horst Bunke, and Mark Last. 2005. *Graph-theoretic techniques for web content mining*, volume 62. World Scientific.
- Yuen-Hsien Tseng. 1998. Multilingual keyword extraction for term suggestion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 377–378.
- Peter D. Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval* 2(4):303–336.
- Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. [Kea: Practical automatic keyphrase extraction](#). In *Proceedings of the Fourth ACM Conference on Digital Libraries*. ACM, New York, NY, USA, DL '99, pages 254–255. <https://doi.org/10.1145/313238.313437>.
- Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2004. World wide web site summarization. *Web Intelli. and Agent Sys.* 2(1):39–53.