Key2Vec: Automatic Ranked Keyword Extraction from Scientific Articles using Keyphrase Embeddings

Anonymous ACL submission

Abstract

Keyword extraction is a fundamental task in natural language processing that facilitates mapping of documents to a concise set of representative single and multi-word phrases. Keywords from text documents are primarily extracted using supervised and unsupervised approaches. In this paper, we present a novel and effective unsupervised technique named *Key2Vec* that extensively takes into account neural keyphrase embeddings for extracting and ranking keywords from scientific articles. We introduce an efficient way of processing text documents and training keyphrase embeddings for the purpose of extracting and ranking keywords. We share a new evaluation dataset used for choosing the underlying model for *Key2Vec*, which is based on semantic similarity of textual segments. The evaluations for ranked keyword extraction are performed on two benchmark datasets comprising of short abstracts (Inspec), and long scientific papers (SemEval 2010), and is shown to produce results better than the state-of-the-art systems. We also show how keyphrase embeddings could be further used for capturing semantic similarity between text documents, and for performing efficient extractive summarization using the *Key2Vec* framework.

1 Introduction

Keywords are single or multi-word linguistic units that represent the salient aspects of a document. In this paper, we use the term keyword uniformly to represent both single and multi-word phrases. Keywords are useful in many tasks such as indexing documents (Gutwin et al., 1999), summarization (Litvak and Last, 2008), clustering (Hammouda et al., 2005), ontology creation (Gulla et al., 2007), classification (Hulth and Megyesi, 2006), auto-tagging (Wu et al., 2010) and visualization of text (Collins et al., 2009). Due to its widespread use, keyword extraction has received a lot of attention from researchers.

In order to push the state-of-the-art performances of keyword extraction systems, the research community has been hosting shared tasks like SemeEval 2010 Task 5 (Kim et al., 2010) and SemEval 2017 Task 10 (Augenstein et al., 2017). However, the task is far from solved and the performances of the present systems are worse in comparison to many other NLP tasks (Liu et al., 2010). Some of the major challenges are the varied length of the documents to be processed, their structural inconsistency and developing strategies that can perform well in different domains (Hasan and Ng, 2014).

The task of ranked keyword extraction from scientific articles and identifying relationships between them is of great interest to scientific publishers as it helps to recommend articles to readers, highlight missing citations to authors, identify potential reviewers for submissions, and analyse research trends over time (Augenstein et al., 2017). Although scientific articles usually provide them, most of the documents have no associated keywords. Therefore, the problem of automatically extracting the most representative keywords from scientific articles is an active field of research.

Methods for automatic keyword extraction are mainly divided into two categories: *supervised* and *unsupervised*. Supervised methods approach the problem of keyword extraction as a binary classification problem (Hasan and Ng, 2014), whereas the unsupervised methods are mostly based on TFIDF, clustering, and graph-based ranking (Hasan and Ng, 2010). On presence of domain-specific data, supervised methods have shown better performance than the unsupervised methods. The unsupervised methods have the advantage of not requiring any training data and can produce results in any domain. However, the as-

sumptions of unsupervised methods do not hold for every type of document.

With recent advancements in deep learning techniques applied to natural language processing, the latest trend is to represent words as dense real-valued vectors obtained by training a shallow neural network on a fixed vocabulary. These distributional representation of words are popularly known as word embeddings. These neural representations have been shown to equal or outperform other methods (e.g. LSA, SVD) (Baroni et al., 2014). The vector representation of words, are supposed to preserve the semantic and syntactic similarities between them. They have been shown to be useful for several natural language processing (NLP) tasks, like part-of-speech tagging, chunking, named entity recognition, semantic role labeling, syntactic parsing, and also for speech processing (Collobert et al., 2011). Some of the most popular approaches for generating word embeddings are Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014) and Fasttext (Bojanowski et al., 2016).

In this paper, we introduce the notion of *keyphrase embeddings*, and show how they could be trained in a simple and effective way using already existing algorithms belonging to the neural word embedding family (Word2Vec and Fasttext). Instead of distributional representation of words we produce distributional representation of keyphrases for the purpose of ranked keyword extraction from scientific articles. We call our methodology as *Key2Vec*. We also propose a text processing pipeline for training keyphrase embeddings, which enables intelligent tokenization of text into a mix of unigram and meaningful chunks of multi-gram tokens. We show how the embedding models trained using our strategy, can be advantageously used for representing keyphrases and text snippets as dense vectors that are further used for calculating semantic and syntactic similarities. We fully rely on training high quality embedding models and effectively use the similarities between the distributional representation of keyphrases for ranking representative keywords from scientific documents. Table 1, shows keywords extracted from a sample research article abstract, by using *Key2Vec*.

For the purpose of evaluating our embedding models on different similarity tasks, we develop a new evaluation dataset[1] (Section **??**), from an exisiting dataset (Dai et al., 2015). We share the dataset with the community believing that it would facilitate training of better keyphrase embedding models useful for ranked keyword extraction. Contrary, to the analogy task that is widely used for intrinsic evaluation of word embeddings, this dataset can be used for evaluating phrase and word embeddings on similarity tasks, whenever they are being trained for a downstream process that has semantic and syntactic similarity between distributional representation of text at its core. We perform such an evaluation and show that our keyphrase embeddings are more effective in capturing semantic similarity between text documents at lower dimensions, than normal word embeddings, when trained using the same parameters (Section **??**).

Table 1: Keywords extracted by using *Key2Vec* from a sample research article abstract.

Title: Identification of states of complex systems with estimation of admissible measurement errors on the basis of fuzzy information.
Abstract: The problem of identification of states of complex systems on the basis of fuzzy values of informative attributes is considered. Some estimates of a maximally admissible degree of measurement error are obtained that make it possible, using the apparatus of fuzzy set theory, to correctly identify the current state of a system.
Automatically identified keywords: *complex systems, fuzzy information, admissible measurement errors, fuzzy values informative attributes, measurement error, maximally admissible degree, fuzzy set theory*
Manually assigned keywords: *complex system state identification, admissible measurement errors, informative attributes, measurement errors, fuzzy set theory*

We not only aim at extracting the keywords that are statistically relevant for the given document, as mostly attempted by previous studies (Hasan and Ng, 2014), but also to rightly identify meaningful keyphrases that are most semantically related to the main theme of the document. We are motivated by the following properties for the top-K keywords that we select using *Key2Vec* as mentioned in (Liu et al., 2009).

- *Understandability* - The ranked keywords should be understandable to readers, which is made possible by chosing grammatically correct and meaningful phrases that possess the characteristic of high readability by humans. For example, the phrase *scientific articles* has more understandability than the in-

[1] www.example.com

dividual words *scientific* and *articles*.

- *Relevancy* - The top-K keywords should be semantically related to the central theme of the document.
- *Good Coverage* - The keywords should cover all the major topics discussed in the document.

This intrigued us to look at the route of training keyphrase embeddings as presented in this paper, rather than first training embedding models for unigram words and then combining their dense vector representations to obtain similar representations for multi-word phrases. Training keyphrase embeddings for extracting and ranking keywords is relatively a new problem and has not been thoroughly explored. Researchers have just started exploring the advantages of neural word embeddings for representing semantic similarities between words that are further leveraged in graph-based ranking algorithms, in order to rank extracted keywords (Wang et al., 2015). To our knowledge, keyphrase embeddings have not been used for ranked keyword extraction and summarization, and this work is the first attempt to do so.

Some of the major contributions of this paper are,

- *Intelligent processing of text for training neural keyphrase embeddings*.
- Effective *application of keyphrase embeddings for extraction and ranking of keywords*, producing state-of-the-art results.
- *A dataset for evaluating the quality of embedding models*, trained for the purpose of capturing semantic similarities between different types of textual units.
- A new *generic framework* (*Key2Vec*), based on distributional semantics, for *ranked keyword extraction*, and *summarization*.

The rest of the paper is organized as follows. In Section **??**, we discuss about the background literature and works relevant to ours. We fully describe our methodology in Section **??**, and present our empirical experiment results in Section **??**. We discuss our learnings and applications of our framework in Section **??**, followed by our conclusion and plans for future work in Section **??**

References

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. *arXiv preprint arXiv:1704.02853* .

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL (1)*. pages 238–247.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606* .

Christopher Collins, Fernanda B Viegas, and Martin Wattenberg. 2009. Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*. IEEE, pages 91–98.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537.

Andrew M Dai, Christopher Olah, and Quoc V Le. 2015. Document embedding with paragraph vectors. *arXiv preprint arXiv:1507.07998* .

Jon Gulla, Hans Borch, and Jon Ingvaldsen. 2007. Ontology learning for search applications. *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS* pages 1050–1062.

Carl Gutwin, Gordon Paynter, Ian Witten, Craig Nevill-Manning, and Eibe Frank. 1999. Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems* 27(1):81–104.

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. 2010. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, pages 365–373.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *ACL (1)*. pages 1262–1273.

Anette Hulth and Beáta B Megyesi. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and the*

44th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pages 537–544.

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*. Association for Computational Linguistics, pages 21–26.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*. Association for Computational Linguistics, pages 17–24.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics, pages 366–376.

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 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*. Association for Computational Linguistics, pages 257–266.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Rui Wang, Wei Liu, and Chris McDonald. 2015. Using word embeddings to enhance keyword identification for scientific publications. In *Australasian Database Conference*. Springer, pages 257–268.

Wei Wu, Bin Zhang, and Mari Ostendorf. 2010. Automatic generation of personalized annotation tags for twitter users. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics*. Association for Computational Linguistics, pages 689–692.

A Supplemental Material

Submissions may include resources (software and/or data) used in in the work and described in the paper. Papers that are submitted with accompanying software and/or data may receive additional credit toward the overall evaluation score, and the potential impact of the software and data will be taken into account when making the acceptance/rejection decisions. Any accompanying software and/or data should include licenses and documentation of research review as appropriate.

NAACL-HLT 2018 also encourages the submission of supplementary material to report preprocessing decisions, model parameters, and other details necessary for the replication of the experiments reported in the paper. Seemingly small preprocessing decisions can sometimes make a large difference in performance, so it is crucial to record such decisions to precisely characterize state-of-the-art methods.

Nonetheless, supplementary material should be supplementary (rather than central) to the paper. **Submissions that misuse the supplementary material may be rejected without review.** Essentially, supplementary material may include explanations or details of proofs or derivations that do not fit into the paper, lists of features or feature templates, sample inputs and outputs for a system, pseudo-code or source code, and data. (Source code and data should be separate uploads, rather than part of the paper).

The paper should not rely on the supplementary material: while the paper may refer to and cite the supplementary material and the supplementary material will be available to the reviewers, they will not be asked to review the supplementary material.

Appendices (*i.e.* supplementary material in the form of proofs, tables, or pseudo-code) should come after the references, as shown here. Use `\appendix` before any appendix section to switch the section numbering over to letters.

B Multiple Appendices

. . . can be gotten by using more than one section. We hope you won't need that.