Journal of Intelligent Information Systems
https://doi.org/10.1007/s10844-019-00558-9

Automatic keyphrase extraction: a survey and trends

Zakariae Alami Merrouni[1] · **Bouchra Frikh**[1] · **Brahim Ouhbi**[2]

Received: 21 July 2018 / Revised: 15 April 2019 / Accepted: 16 April 2019 /
Published online: 02 May 2019

Abstract
Due to the exponential growth of textual data and web sources, an automatic mechanism is required to identify relevant information embedded within them. The utility of Automatic Keyphrase Extraction (AKPE) cannot be overstated, given its widespread adoption in many Information Retrieval (IR), Natural Language Processing (NLP) and Text Mining (TM) applications, and its potential ability to solve difficulties related to extracting valuable information. In recent years, a wide range of AKPE techniques have been proposed. However, they are still impaired by low accuracy rates and moderate performance. This paper provides a comprehensive review of recent research efforts on the AKPE task and its related techniques. More concretely, we highlight the common process of this task, while also illustrating the various approaches used (supervised, unsupervised, and Deep Learning) and released techniques. We investigate the major challenges that such techniques face and depict the specific complexities they address. Besides, we provide a comparison study of the best performing techniques, discuss why some perform better than others and propose recommendations to improve each stage of the AKPE process.

Keywords Information retrieval · Natural language processing · Text mining · Automatic keyphrase extraction · Supervised approaches · Unsupervised approaches · Deep learning

1 Introduction

Given the exponential growth and development of information available on textual data and the Internet, effectively seeking and managing relevant information becomes an

✉ Zakariae Alami Merrouni
zakariae.alamimerrouni@usmba.ac.ma

Bouchra Frikh
bfrikh@yahoo.com

Brahim Ouhbi
ouhbib@yahoo.co.uk

[1] TTI Laboratory, Higher School of Technology (EST), Sidi Mohamed Ben Abdellah University, B.P. 2427, Route d'imouzer, Fez, Morocco

[2] Mathematical Modeling & Computer Laboratory (LM2I), National Higher School of Arts and Crafts (ENSAM), Moulay Ismail University (UMI), Marjane II, B.P. 4024, Meknes, Morocco

important research matter. On the Internet, existing search engines aid in extracting information through a pattern that matches keywords from large text collection. The textual data, unstructured or semi-structured, operate in different domains (e.g., online news, discussion forums, online books, scientific publications, and search engine usage logs). The associated challenges related to the textual data reveal a substantial number of research efforts in IR, TM and NLP. A major part of handling textual data is identifying the key terms that are informative and relevant. Keyphrases, key terms or keywords all function to characterize and capture the main topics of a large text data collection or a single document. They not only provide the main idea of the document, but also aid readers in deciding whether to continue further reading or search for additional details. For a single document, keyphrases can serve as a condensed summary (e.g. keyphrases of a journal article). They enable the reader to decide quickly whether the given article is interesting or not. For a collection of documents (e.g. digital libraries, scientific articles, web pages, email messages, magazine articles, business papers, and news reports), keyphrases can be used for browsing, searching, categorizing, classifying, indexing, clustering or managing. Document keyphrases are widely used in various TM, IR and NLP tasks (Jones and Staveley 1999; Moldovan et al. 2000; Zha 2002; Zhang et al. 2004; D'Avanzo and Magnini 2005; Medelyan and Witten 2006; Turney 2003; Bougouin et al. 2013) (see Fig. 1). Although these known benefits of keyphrases, the manual assignment of them is expensive, time-consuming and difficult especially with large documents or web data. Thus, there is a need to automate this task. The task of AKPE is defined as the process of automatically extracting relevant, descriptive and expressive phrases from a document or web data. AKPE techniques have received much consideration in the past decades due to their importance. Several AKPE techniques have been proposed (Kim et al. 2010). Nevertheless, the performance on keyphrase extraction as well as the results both remain moderate (Kim et al. 2010; Liu et al. 2010). To the best of our knowledge, there are few systematic reviews which shape the process of the AKPE task and highlight the current progress of existing works. This survey seeks to provide a comprehensive summary of current research on AKPE task. This includes highlighting its common process, while also illustrating the various approaches used and released techniques. We examine the complexities they address and the major challenges that such AKPE techniques face. Besides, we provide a comparison study of the best performing techniques, discuss why some perform better than others and propose recommendations to improve each stage of the AKPE. This paper is organized as follows. In Section 2, we present a review of the utilization of the keyphrases in various IR, NLP and TM tasks, then, we present the research methodology used for conducting this review. Section 3 presents the detailed in-depth analysis for each stage of the AKPE, its approaches (supervised, unsupervised, and deep learning), and its released techniques (including examining their strengths and weaknesses). In Section 4, we select some representative techniques and compare their performance. Then, we discuss the obtained results and give recommendations for each stage of the AKPE. Section 5 concludes the paper.

2 Materials and methods

2.1 From keyword to keyphrase

A keyword is "a single word that is highly relevant". A keyphrase is "a sequence of one or more words that are considered highly relevant" (Hammouda et al. 2005). In IR systems and search engines, keyphrases are much more convenient for users because they can express

more precise information. Keyphrases, key terms or keywords have the same roles that characterize and capture the main topics of a single large document, or data collection. Since keyphrases are considered more significant and informative than keywords in documents and search systems, users prefer keyphrases over keywords (Zhang et al. 2004).

2.2 Some uses of keyphrases in IR, NLP and TM tasks

Keyphrases are widely used in various TM, IR and NLP tasks (see Fig. 1). In this section, we describe some uses of keyphrases in these tasks to justify their importance.

- **IR systems:** keyphrases are used extensively to describe documents returned by a query (Jones and Staveley 1999; Moldovan et al. 2000), as the list of keyphrases can help to decide whether a given document is relevant to the user's interest. They improve the functionality of IR systems by using them to build an automated index for document collection, or instead of using them for document representation in classification or categorization tasks (Hulth and Megyesi 2006).
- **Text or document summarization:** given the overload of information that is available on textual databases and the Web, the core application area of keyphrases is focused on Document Summarization (DS). DS involves the creation of an informative summary from a document (or a collection of documents) by distilling and extracting the most relevant parts of the text, which can presented as a set of keyphrases. Thus, keyphrases are useful in DS. They can serve as a form of semantic metadata indicating the significance of sentences and paragraphs in which they appear (Zha 2002; Zhang et al. 2004; D'Avanzo and Magnini 2005). Web Search Engines present a list of documents and informative summaries (parts of the text matching query terms). Such summaries help the user to decide which documents are relevant. They facilitate users to better understand large amounts of text information, especially when the documents contain various topics.

Fig. 1 Some uses of keyphrases in various TM, IR and NLP tasks

- **Document indexing:** in large text data (for example Digital Libraries (DLs)), readers face difficulties in locating their desired sections. Keyphrases are widely used in organizing Digital Library holdings and providing thematic access to documents. The DLs are often supplemented with an extra section called the index. This section consists of important topics described in the document and their locations. The index can be built using an alphabetically categorized list of keyphrases, extracted from parts of a single long document or from a collection of documents (Medelyan and Witten 2006; Newman et al. 2012; Gutwin et al. 1999). Moreover, keyphrases can also be used for the task of thesaurus creation for these DLs (Medelyan and Witten 2006).
- **Document clustering:** document clustering is the act of gathering similar documents into groups (or clusters), where similarity is decided by functions (for example Euclidean Distance) on these documents. Text clustering can be in different levels of granularities where clusters can be documents, paragraphs, sentences, keyphrases or terms. Text clustering approaches based on key phrases help to achieve more effective clustering results (Zhang and Dong 2004; Osiński et al. 2004; Hammouda et al. 2005; Han et al. 2007; Liu et al. 2009). Indeed, using keyphrases as document features has several benefits: (1) it can improve the clusters' quality by leveraging more information presented in the document, (2) it is helpful in building concise and accurate descriptions (labels) for the produced clusters (Zamir and Etzioni 1998), (3) it can aid in producing clusters that group the documents that are relevant to the user's query apart from the irrelevant ones (Zamir and Etzioni 1998).
- **Web mining:** the rapid growth of the Web has led to an increase in the web data (e.g. page contents, usage logs, and web hyperlinks). Web mining aims to discover and extract valuable information from these various types of data. For example, using keyphrases in web mining techniques can aid in extracting document topics and ranking web documents. Thus, users can gather information both efficiently and comfortably (Turney 2003; Chen et al. 2005).
- **Search engines, Web logs:** keyphrases are used beneficially in (1) Search Engines (SE) which help to extract information through pattern matching, using keyphrases or keywords for a given user query. SE provides a conceptualization of a large text collection. Thus, keyphrases of web pages can serve as metadata for indexing and retrieving these web pages efficiently. And (2) weblogs, where keyphrases can be used to analyze the usage patterns of a user and his or her trends and interests (Gong and Liu 2009).
- **Query expansion, Query suggestion, Query refinement, Relevance feedback:** Query expansion, query suggestion, and query refinement are some of the most well-known query modification techniques in IR systems and search engines. They can add, suggest, reformulate, refine or transform relevant terms (to form keyphrases) that will aid the user in removing the ambiguities of natural language and express the information idea in a more detailed way into the original query (Krovetz and Croft 1992; Chandrasekar et al. 2006; Song et al. 2006). The majority of these techniques are based on the user relevance feedback or the result list of the previous query. Relevance Feedback (RF) is a proven approach for expanding a query by selecting important keyphrases or expressions attached to documents retrieved from the original query. Users in the result set then marked them as relevant, or the system deduced that the top-ranked documents were relevant (Song et al. 2006).
- **Recommender systems, opinion mining:** Content-Based Recommender Systems (CBRS) (Lops et al. 2011; Ferrara et al. 2011) and Advertisement Targeting (AT) (Yih et al. 2006) are based on keywords automatically extracted from the text of the page. They help users in discovering relevant information to their previously expressed

interests. Opinion mining and sentiment analysis deal with the treatment of opinion, sentiment, and subjectivity in the text. Keyphrases aid in representing and understanding these opinions (for example, in web reviews, how an opinion-holder reacts toward a given product) (Berend 2011).

- **Ontology:** ontologies ease knowledge sharing and re-use. More recently, the notion of ontology has become widespread in information retrieval and knowledge management. Keyphrases are used to facilitate: the automatic construction of concept maps (Leake et al. 2003), ontology learning and building (Fortuna et al. 2006; Frikh et al. 2011; El Idrissi et al. 2014; Do and Ho 2015). Precisely, they help in identifying relevant concepts and provide a basis for building ontologies.
- **Information extraction and Named entity extraction:** Information Extraction (IE) is the task of automatically extracting information from unstructured or semi-structured documents. Such information can be present as a set of keyphrases or a block of text that include various named entities in different domains (e.g. biomedical, News, Business, Politics, etc.) (Dostal and Jeżek 2011). A named entity is a term or phrase that identifies a real-world object. These entities categorize people, geographic locations, addresses, organizations, products, etc.
- **Topic analysis:** due to the exponential growth of textual data, the complexity of document corpora has led to considerable interest in using topics to facilitate tasks like browsing and searching. The topic represents an underlying semantic theme of a document. Topic keyphrases briefly describe and represent the key content of topics. They help users grasp the key content of a topic and make the decision to keep reading the document or not (Liu et al. 2009; Grineva et al. 2009; Bougouin et al. 2013).

Additionally, AKPE task is used in several real-world domains like medicine (Sarkar 2013; Liu et al. 2015), agriculture (El-Beltagy and Rafea 2009), national security and terrorism (Elovici et al. 2010), social media (Matsuo et al. 2007; Mori et al. 2007), and law (Jungiewicz and Łopuszyński 2014).

2.3 Research methodology and criteria

How do we collect the papers? In this survey, relevant works are retrieved by querying multiple electronic databases (see Table 1). In addition, most of the related high-profile journals, conferences and workshops (see Table 2) are investigated manually to analyze AKPE works from different perspectives.

More generally, the survey selection process in both databases and target journals-conferences-workshops consists of three steps: (1) a set of major keywords are used including: "keyphrase extraction" or "automatic keyphrase extraction", next, (2) formal searches are performed sequentially on (i) the selected databases, and (ii) the target domain journals and conferences. Then, (3) in each formal search, titles, abstracts, and full-texts of

Table 1 Electronic databases investigated in this survey

#	Electronic databases
EDB1	Science Direct
EDB2	IEEE Xplore
EDB3	ACM Digital Library
EDB4	SpringerLink
EDB5	Google Scholar

Table 2 Some target journals-conferences-workshops investigated in this survey

#	*Name*	*Publisher*
J1	Knowledge and Information Systems Journal	Springer
J2	Information Retrieval Journal	Springer
J3	Information Systems Journal	Elsevier
J4	Journal of The Association for Information Science and Technology	Wiley Online Library
J5	Information Processing & Management	Elsevier
J6	Information Sciences	Elsevier
C1	Symposium on Document Engineering (DocEng)	ACL
C2	International Conference on Computational Linguistics (COLING)	ACL
C3	International Conference on World Wide Web (WWW)	ACM
C4	International Conference on Web-Age Information Management (WAIM)	Springer
C5	International Conference on Data Mining ICDM	IEEE
C6	International ACM SIGIR Conference on Research and Development in Information Retrieval	ACM
C7	Conference on Empirical Methods in Natural Language Processing (EMNLP)	ACL
C8	ACM/IEEE-CS Joint Conference on Digital libraries (JCDL)	ACM
WS1	International Workshop on Semantic Evaluation (SemEval)	ACL
WS2	Workshop on Multiword Expressions (MWE)	ACL

potential works are analyzed against predefined criteria to decide whether each paper should be included or not. Effectively, articles and conferences papers are reviewed and analyzed to ensure that they are, in fact, present theoretical or/and practical solution (s) to this task. More than 200 papers are collected when applying the first step. Given this scale of papers, a presentation of every existing AKPE work in this survey is not possible. The second step was to identify additional criteria to lessen the number of selected papers. Therefore, papers from the past two decades are considered, focusing on the most referenced, relevant, and recent case works. More precisely, works that implement an AKPE technique in TE, NLP or IR domains are prioritized. Applying these criteria, a total of 34 techniques are analyzed and listed chronologically, and more than 85 references are chosen to conduct the overall survey.

3 The automatic keyphrase extraction task

3.1 The automatic keyphrase extraction process

As previously stated, the task of AKPE is defined as the process of automatically extracting relevant, descriptive and expressive phrases from a document or web data. Generally, most AKPE systems identify a set of words and phrases called "candidates" that could convey the topical content of a document, then these candidates are scored and ranked. Finally, the "best" ones are selected as a document's keyphrases. A typical AKPE process consists

of five main stages (see Fig. 2): (1) pre-processing the input data, (2) identification of candidate phrases, (3) feature selection and scoring, and (4) keyphrase ranking. Depending on the goal of every system, the ranked extracted keyphrases can be presented in two sets: single document keyphrases or document collection keyphrases, and finally (5) the evaluation of the extracted keyphrases. Figure 2 depicts the overall process of automatic keyphrase extraction, whose main stages are analyzed and explained in the following subsections.

3.1.1 Documents pre-processing and candidate phrases identification stage

The document(s) preprocessing stage consists of formatting the document (s) into a machine-readable format to decreasing its (their) complexity. This requires some text operations including sentence segmentation, stemming (reduction of word to grammatical root), tokenization (converting a sequence of characters into a sequence of tokens), part-of-speech tagging, noisy symbols and stop words removal, and named entity recognition (that seeks to locate and classify named entity mentions in text into predefined categories such as names, locations, quantities, time expressions, values). After this stage, candidate phrases are identified from the full-text document. A set of phrases are typically extracted as candidate keyphrases using heuristic rules which aim to reduce the number of false-positive candidates while maintaining the true-positives. To detect all candidate keyphrases, most methods usually fall into three categories: (1) N-Gram based, (2) Part-Of-Speech (POS) sequence based, or both. Category (1) was adopted in the pioneer and earlier systems (Witten et al. 1999; Turney 2000), where the input text is separated according to phrase boundaries (e.g., punctuation marks) and to a limited length (bigram, trigram, etc.). Then, some simple rules are applied to candidate phrases to filter meaningless sub-sequences (for example, candidate phrases cannot begin or end with a stop word) (Huang et al. 2006; Liu et al. 2009). The commonly used category (2) allows words with definite part-of-speech tags (for example, adjectives, nouns, verbs) to be candidate keywords by using a POS tagger (Barker and Cornacchia 2000). The noun phrase-chunking category is also performed especially when we search for text chunks matching to individual noun phrases. Hulth (2003) compares three different phrase identification methods and concludes that extracting noun phrase- chunks and word sequences (that match any set of a POS tag pattern) both give better precision than n-grams. Grineva et al. (2009) use n-grams that appear in Wikipedia article titles as candidates. Haddoud et al. (2015) identified five existing POS tag sequence definitions of a noun phrase and proposed a new POS tag sequence that shows significant improvements over other filters. In case the results of this stage generates a long list of candidates, Huang et al. (2006), Kumar and Srinathan (2008), El-Beltagy and Rafea (2009), You et al. (2009), and Newman et al. (2012) have designed different pruning heuristics to limit

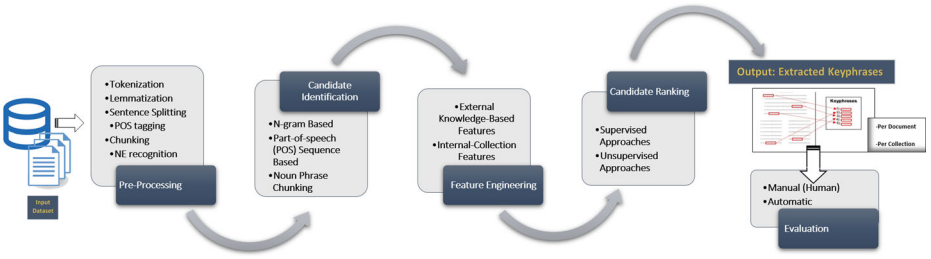

Fig. 2 Automatic keyphrase extraction process stages

candidates that are unlikely to be keyphrases. Using these pruning heuristics increases the recall rates in extracting relevant keyphrases.

3.1.2 Feature engineering and ranking candidates stages

- **Feature engineering:** the feature engineering step involves developing properties that could characterize individual keyphrase candidates from other terms. These properties are called features. Features perform in both (1) supervised learning approaches (feature weights rely on priors or statistics attained in the training process), such as $tf - idf$, Accuracy, F1-measure, Mutual Information, Term Strength, Chi-Square, Information Gain, and Odds Ratio, and in (2) unsupervised learning approaches (features which are extracted without training like structural rules (tags or positions) or frequency-based information). Regardless, features have been used extensively as an instance for supervised keyphrase extraction and reflect how well a candidate phrase represents the topic and content of the document. Practically, several features are calculated for each candidate phrase to measure its importance during the feature engineering stage. The majority of the used features combine: (1) frequency statistics within a single document and across an entire collection, (2) semantic similarity among keyphrases (i.e. keyphrase cohesion), (3) popularity of keyphrases among manually assigned sets, (4) heuristics, such as locality and the length of phrases (Medelyan and Witten 2006), and (5) lexical and morphological analysis (for example, hyphenated phrases and morphological affixes). Usually, features are classified according to their origin (see Fig. 3): phrase-level features (such as structural features, syntactic and morphological ones), document-level features (e.g. statistical and baseline set features within a single document), corpus-level features (applied across an entire collection), and external knowledge-based features (such as article data-banks (Wikipedia,[1] etc.), terminological databases (MeSH,[2] GRISP,[3] etc.), linguistic resources (WordNet,[4] etc.), and Web Search Query Log.

  The most popular features are: $tf - idf$ (that select those candidate phrases which are frequent in a given document, but less frequent in the whole document collection) (Frank et al. 1999; Witten et al. 1999; Nguyen and Kan 2007; Liu et al. 2009), and first occurrence (Witten et al. 1999) which are often used as a baseline by most existing AKPE systems. The KEA++ system (Medelyan and Witten 2006) adds the feature of phrase length in words and the node degree (representing the number of thesaurus links that connect the term to other candidate phrases) as a refinement of KEA system. Keyphrase cohesion is another extensively-used feature. Turney (2003) introduces two new sets of features and measures keyphrase cohesion (the statistical association) within the top-N keyphrase candidates against the remaining candidates using web frequencies. Context, document type, and structure are also considered boosting information for feature calculation. Zhang et al. (2006) propose to use not only 'global' but also 'local' context information, such as the linkage feature. Kumar and Srinathan (2008) use the position of sentences and position of phrases in a sentence as a feature. Chen et al. (2005) use three new features considering the structure of web pages. In scientific publications, Nguyen and Kan (2007), Kim and Kan (2009), Lopez and Romary (2010), and Nguyen and Luong (2010)

[1] https://en.wikipedia.org/

[2] https://www.ncbi.nlm.nih.gov/mesh

[3] https://hal.archives-ouvertes.fr/inria-00490312/en/

[4] https://wordnet.princeton.edu/download

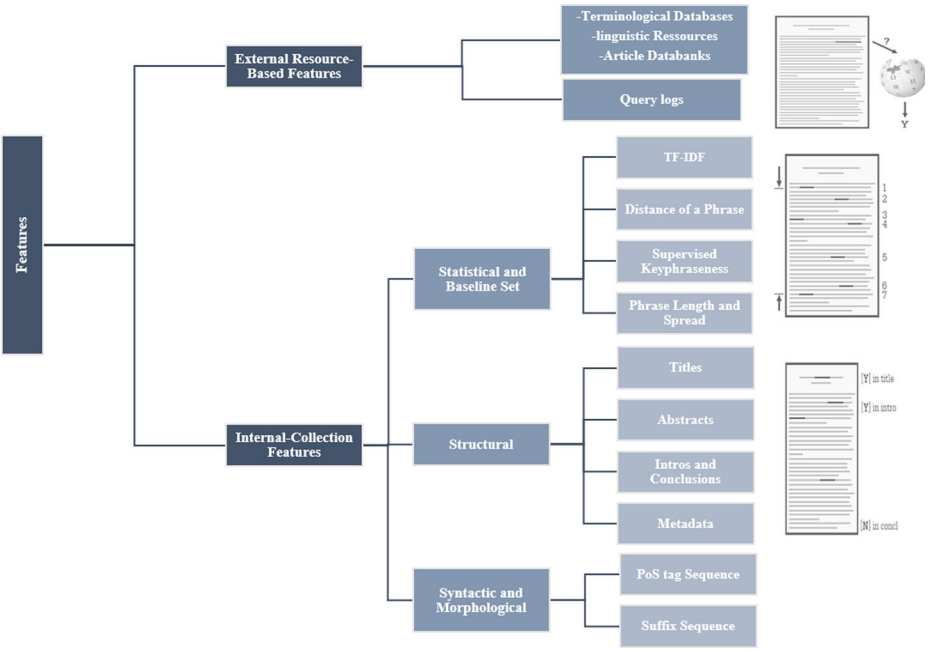

Fig. 3 Baseline features that characterize a keyphrase

propose the section occurrence vectors as a feature. They are based on the idea that keyphrases are distributed non-uniformly in different logical sections of a paper, preferring sections such as abstracts, titles, introductions, and related works. Berend and Farkas (2010)proposed features regarding the document level, the corpus level, and the knowledge-based level. They rely on external knowledge sources too (i.e. Wikipedia) to achieve additional enhancements in the system. Based on the semantic relations among candidate phrases, Kelleher and Luz (2005) used the "Semantic Ratio" (SR) feature that divides the number of occurrences of a phrase in the present document by the number of times it occurs in all documents directly linked to that document. Huang et al. (2006) treat each document as a semantic network and analyze its connectedness and compactness to symbolize the importance of a node, which represents a candidate phrase. Wan and Xiao (2008) use a set of neighborhood documents to improve the single document keyphrase extraction. Other authors have used the term co-occurrence of candidates (Mihalcea and Tarau 2004; Matsuo and Ishizuka 2004; Liu et al. 2009).

- **Keyphrases ranking:** keyphrase extraction is generally considered a ranking problem (i.e. keyphrase candidates are ranked based on their feature values, and the top-N ranked ones are returned as keyphrases). The ranking approaches aim to develop machine-learning models that determine which of the candidate keyphrases generated in a prior stage are relevant and then rank them. Three learning approaches can be used to achieve this aim: the supervised, unsupervised, and deep learning (see Fig. 4).

The supervised approaches keyphrase extraction is considered a binary classification problem (Frank et al. 1999; Turney 2000). To cope with this kind of situation, different supervised machine learning (ML) models are used (see Fig. 5). In ML terminology, the

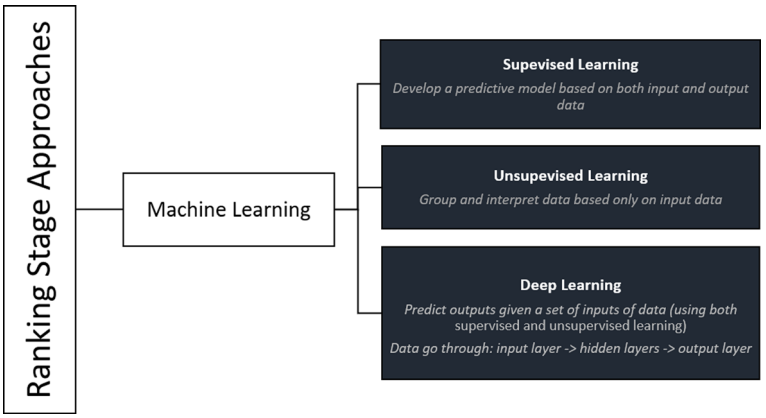

Fig. 4 Automatic keyphrase extraction ranking approaches

phrases in a document are "examples" and the learning problem is defined as a mapping from the examples to the two classes: "keyphrase" and "not-keyphrase". Supervised ML models can automatically generate this mapping if they are provided with a set of training examples. Hence, by training a classifier on documents annotated with keyphrases, we can determine whether a candidate phrase is a keyphrase. Different supervised ML models are used such as: bagged C4.5 (Turney 2000), Naïve Bayes (Frank et al. 1999), neural networks (Kamal Sarkar Mita Nasipuri 2010), SVM (Jiang et al. 2009), maximum entropy (Yih et al. 2006). Below we examine some representative AKPE systems that adopt the supervised approach.

– KEA System (Witten et al. 1999): in KEA, the candidate keyphrases are identified using lexical methods, then, for each candidate phrase, a set of features is calculated. Finally, a machine-learning model (Naïve Bayes) is used to predict and classify candidates as good keyphrases or not. This model is used on a candidate phrase with the following feature values: (1) t (for $tf - idf$) a measure of a phrase frequency in a document compared to its rarity in general use (general usage is represented by document

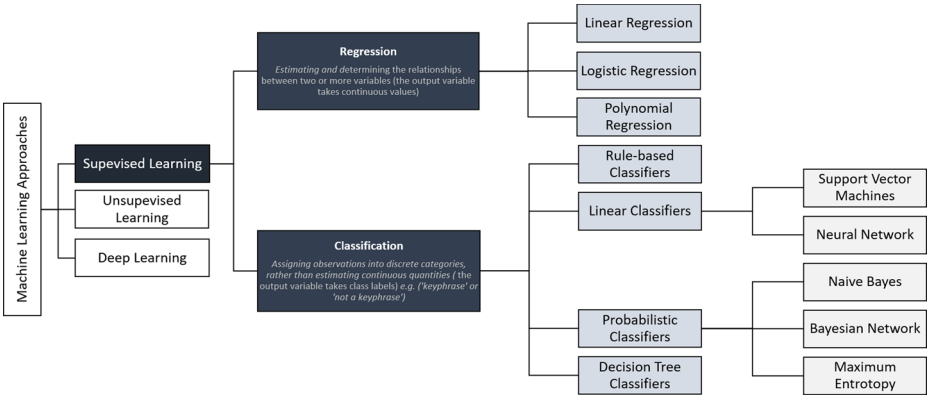

Fig. 5 Supervised machine learning models

frequency which is the number of documents containing the phrase in some large corpus), (2) d (for distance) first occurrence which is calculated as the number of words that precede the phrase's first appearance, divided by the number of words in the document. The result can be a number between 0 and 1 that represents how much of the document precedes the phrase's first appearance. Generally, the model determines the overall probability that each candidate is a keyphrase, and then a post-processing operation selects the best set of keyphrases. Two probabilities are calculated: $P[yes]$ (the probability that a candidate phrase is a keyphrase) and $P[no]$ (the probability that a candidate phrase is not a keyphrase), calculated similarly (1):

$$P\,[yes] = \frac{Y}{Y+N} P_{tf\times idf}\,[t \mid yes]\, P_{distance}\,[d \mid yes] \tag{1}$$

where $P_{TF\times IDF}\,[t|yes]$ is the proportion of positive examples that have a discrete $tf\times idf$ value, $P_{distance}\,[d|yes]$ is the probability of the first appearance of a keyphrase into the document. Y (that is, the author-identified keyphrases) is the number of positive instances in the training files, and N (that is, candidate phrases that are not keyphrases) is the number of negative instances (the Laplace Estimator is used to avoid zero probabilities. This simply replaces Y and N by $Y+1$ and $N+1$). The overall probability (2) that the candidate phrase is a keyphrase can then be calculated and ranked according to this value:

$$p = \frac{P\,[yes]}{P\,[yes] + P\,[no]} \tag{2}$$

However, the KEA system depends on the training set and may provide poor results when the training set does not properly fit the processed documents. KEA also has a limited configuration of keyphrase selection and filtering. Moreover, it does not offer capabilities for changing the entire pre-process or adding filters.

– GenEx Algorithm, Turney (2000): GenEx, is one of the most representative AKPE techniques. It is based on a genetic algorithm which optimizes the number of correctly identified keyphrases in the training documents. GenEx has two components: the Genitor genetic algorithm and the Extractor keyphrase extraction algorithm. Extractor takes a document as input and produces a list of keyphrases as output. It uses three features: term frequency (tf), phrase length, and first occurrence information. It is based on the C4.5 decision-tree-like process that achieves both candidate selection and weighting. The parameters of the Extractor are tuned by the Genitor genetic algorithm, to maximize performance (fitness) on the training data. GenEx has the ability to preserve its performance across different domains by not using any domain-dependent attributes for classification and weighting purposes. Later, Turney (2003) proposed to model the coherence of an entire set of candidate phrases. This approach uses the degree of statistical association among candidate keyphrases as evidence that they may be semantically related, which is done by using the pointwise mutual information (PMI) between a candidate and k previously selected phrases.The PMI can be used in conjunction with a web search engine in order to exploit the extra-large corpus. Experiments demonstrate that this enhancement improves the quality of the extracted keyphrases (Turney 2003). However, it is difficult to obtain the PMI of these sets without using sufficiently large datasets. Additionally, calculating the coherence between features is time-consuming.
– Hulth System (Hulth 2003): unlike the KEA and GenEx systems, the Hulth system places no limit on the length of the extracted keyphrases. It uses four different features in conjunction with a bagging technique. These features are term frequency, collection frequency, the relative position of the first occurrence of a term and the part-of-speech

tag of a term. In the identification of candidates, Hulth considers the 56 most frequently occurring POS tag sequences among keyphrases in the training data. Hulth argues that using a POS tag as a feature significantly improves the results of keyphrase extraction, as certain POS-patterns are more likely to denote keyphrases. Experimentation carried out by Hulth has shown that using a combination of several prediction models applied on noun phrase candidates produced the best results. Unfortunately, in the Hulth system (1) there is no relation between the different POS tag feature values, (2) this system has not been evaluated on any of the GenEx and KEA datasets, and (3) the recall value reported in this work is low. Consequently, there is no common basis for comparing the results of the three systems (Hulth, KEA, and GenEx).

- Huang et al. Algorithm (2006): Huang et al. propose an AKPE algorithm using two novel weight features that can be adapted in both supervised and unsupervised tasks. This system performs well on digital book datasets. However, due to the lack of phrase redundancy in short documents, the system does not perform well with all types of documents.
- Maui Algorithm (Medelyan et al. 2009): Maui is a general algorithm for automatic topical indexing, which is based on the KEA system (Frank et al. 1999). Maui extended the KEA system to integrate information from Wikipedia. Two stages are performed in this system: candidate selection and machine learning. In the first stage, Maui determines textual sequences defined by orthographic boundaries and splits these sequences into tokens. Then all n-grams up to a maximum length of 3 words that do not begin or end with a stop word are extracted as candidate tags. In the second stage, several features are computed for each candidate, then, they are combined in a machine-learning model to obtain the probability that the candidate is indeed a tag. Maui added a novel feature set that consists of document-specific "keyphraseness", a knowledge-based feature that represents the likelihood of a phrase being a link in the Wikipedia corpus. However, a shortcoming of Maui is the lack of any evaluation capabilities.
- Core Word Expansion Algorithm (You et al. 2009): You et al. developed and evaluated an AKPE technique for scientific documents. This technique can be adapted to both supervised and unsupervised tasks. The core word expansion algorithm is used to generate new candidate phrases. First, a core word set is used to localize potential positions of keyphrases, then, candidate phrases are obtained by expanding the core words. In the feature calculation step, You et al. introduce an idf related feature for selecting the proper granularity when a phrase and its sub-phrases coexist as candidates. Experimental results show the efficiency and effectiveness of the refined candidate set and demonstrate an overall performance of this system compared with KP-MINER (El-Beltagy and Rafea 2009) and KEA (Witten et al. 1999) systems.
- HUMB System (Lopez and Romary 2010): In HUMB, three kinds of features have been used: (1) structural features (the position of a term with respect to the document structure for each candidate, and the position of the first occurrence), (2) the content features (Phraseness, Informativeness, Keywordness) and finally, (3) the Lexical/Semantic features. In the ranking stage, HUMB combined 3 learning models with boosting and bagging techniques. These models are the decision tree (C4.5), Multi-Layer Perceptron (MLP) and Support Vector Machine (SVM). The selected model for the final run was bagged decision tree. However, HUMB used external knowledge bases (GROBID/TEI, GRISP, and HAL) which are associated with scientific domains.
- DPM-index (Haddoud 2014): Haddoud et al. define the document phrase maximality index (DPM-index), a new measure to discriminate overlapping keyphrase candidates in a text document. They developed a supervised learning system that uses 18

statistical features, among them the DPM-index and five other new features which are transformed into input attributes for a logistic regression learning algorithm. This system offers the possibility of evaluating the efficiency of keyphrase candidate selection based on linguistic features. By not using any external knowledge or structural document features, the results of this system demonstrate an improvement over almost all the other compared systems. Later Haddoud et al. (2015) investigate the impact of candidate term-filtering using linguistic information on the accuracy of AKPE from scientific papers. They proposed a new definition for personalized tag sequence. The DPM-index system showed prominent results on the SemEval-2010/Task-5 dataset[5] (a workshop on Semantic Evaluation) (Kim et al. 2010) compared to six of the finest systems (Turney 2000; Kim and Kan 2009; Liu et al. 2009; Nguyen and Kan 2007; Lopez and Romary 2010).

– CeKE model (Gollapalli and Caragea 2014): CeKE is a supervised citation-enhanced keyphrase extraction model. CeKE built on a combination of novel features that capture information from citation contexts and existing features from previous works. The novel designed features consider citation network information to build supervised models that classify (using Naïve Bayes classifiers) candidate phrases as keyphrases, or non-keyphrases. They divided features into three categories: (1) existing features for keyphrase extraction including: $tf-idf$, relative position, POS tag, (2) other novel features like: Citation Network Based (inCited and inCiting (Boolean features that are true if the candidate phrase occurs in cited and citing contexts)), Citation $tf-idf$ feature (the $tf-idf$ score of each phrase computed from the citation contexts), and (3) other features that include: first position of a candidate phrase, $tf-idf-Over$ (a Boolean feature, which is true if the $tf-idf$ of a candidate phrase is greater than a threshold 0), firstPosUnder (a Boolean feature which is true if the distance of the first occurrence of a phrase from the beginning of a target paper is below some value). Later, Bulgarov and Caragea (2015) showed that adding the keyphraseness feature (related to how often a candidate phrase appears as a tag or a keyphrase in the training dataset) to document textual content and textually-similar neighbors features, improves the keyphrase extraction results. This CeKE+ keyphraseness model shows interesting results in the authors' experiments compared to other systems.
– KeyEx Method (Xie et al. 2017): is a sequential pattern mining based document-specific keyphrase extraction method. The key innovation of this method is to use wildcards (or gap constraints) to help in extracting sequential patterns, so the flexible wildcard constraints within a pattern can capture semantic relationships between words, giving the system full flexibility to discover different types of sequential patterns as candidates for keyphrase extraction (Xie et al. 2017). This method discovers a rich set of keyphrase candidates and further uses a supervised learning approach to build a classification model for keyphrase extraction. The author's experiments showed that the KeyEx system is effective in improving the quality of the extracted keyphrases. Furthermore, their method outperforms other sequential pattern mining methods (Xie et al. 2017).

The unsupervised approaches: in unsupervised approaches, the keyphrase extraction task is considered a ranking problem and performs without prior knowledge. These approaches are grouped as either statistical-based or graph-based (see Fig. 6). In statistical-based

[5] http://semeval2.fbk.eu/semeval2.php?location=data

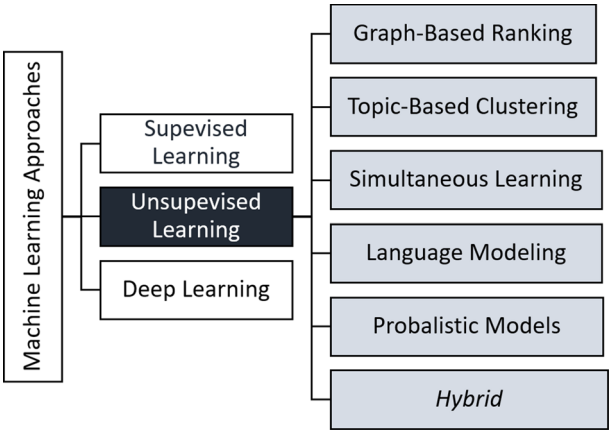

Fig. 6 Unsupervised approaches for automatic keyphrase extraction

models, texts are usually represented as matrices in which the statistical metrics are applied to rank the words, such as $tf - idf$ term weighting metric (Zhang et al. 2006; Liu et al. 2009). Matsuo and Ishizuka (2004) and Frikh et al. (2011) present a study that applies word co-occurrence distribution. Frantzi et al. (1998) combined linguistic and statistical information to extract technical terms from documents in digital libraries.

In recent years, many graphical representation models of the text have been raised. The **graph-based ranking models** are based on the idea of "recommendation" or "voting". They build a graph from the input documents. Each document is represented as a graph where vertices or nodes represent phrases and/or words and the edges are connected based on lexical or semantic relations, such as a co-occurrence relation (Mihalcea and Tarau 2004). Nodes or vertices are then ranked using graph centrality measures such as PageRank and its variants (Page et al. 1999; Mihalcea et al. 2004; Wan and Xiao 2008; Gollapalli and Caragea 2014). These measures assign weights to the node-words reflecting their semantic significance in the text. In **Probabilistic models**, candidate keyphrases are typically scored based on the simple multiplication of feature values (Frikh et al. 2011; Liu et al. 2009). To rank candidates, other authors include the PageRank family measures (Mihalcea et al. 2004; Wan and Xiao 2008; Gollapalli and Caragea 2014). **Hybrid models** combine two or more models in a single model such as 'probabilistic and graph-based' to take advantage of two or more models (Danesh et al. 2015). In **topic based clustering models**, grouping the candidate keyphrases is highly suggestive of an underlying semantic theme, which is called a topic. Based on the assumption that candidate keyphrases that are related to a topic are more likely to occur together, it is possible to attribute phrases to a specific topic (Liu et al. 2009; Grineva et al. 2009; Bougouin et al. 2013). Various methods have been proposed concerning these models such as Latent Dirichlet Allocation (LDA) (Blei et al. 2003), which is a representative of topic models, Latent Semantic Analysis (LSA) (Landauer et al. 1998) and probabilistic LSA (pLSA) (Hofmann 1999). In **language modeling models**, several probabilistic models have been developed to assign a probability to each sequence of words in a document, and assign a probability for the likelihood of this given sequence to follow a sequence of words (Tomokiyo and Hurst 2003). In **simultaneous learning models**, the text summarization and keyphrase extraction tasks can benefit from one another if they are performed simultaneously. Consequently, they will make full use of the

reinforcement between summary and keyphrases (Zha 2002; Wan et al. 2007). Below are some representative AKPE systems that adopt the unsupervised approach.

- Tomokiyo and Hurst Approach (Tomokiyo and Hurst 2003): Tomokiyo and Hurst propose an approach that extracts keyphrases based on statistical language models by using the pointwise KL-divergence between multiple language models. Two features have been used in this system: (1) phraseness: describes the degree to which a given word sequence is considered to be a phrase, and (2) informativeness that refers to how well a phrase captures or illustrates the key ideas in a set of documents. Intuitively, a phrase with high phraseness and informativeness scores is likely to be a keyphrase. However, the results of this approach are difficult to evaluate. Moreover, this approach provides no qualitative evaluation and uses a language model instead of heuristics to identify phrases. Consequently, the system's performance is decreased.
- The TextRank Algorithm (Mihalcea and Tarau 2004): TextRank algorithm is one of the most well represented graph-based models for AKPE. It represents a document as a graph. Each vertex in the graph corresponds to a word, and there is an edge between any two words occurring together in the text. A weight w_{ij} is assigned to the edge connecting two vertices, v_i and v_j, and has a value as the number of times the corresponding words co-occur within a window of W words in the document. TextRank adapts the original PageRank (Brin and Page 1998) algorithms to calculate word ranks. The original PageRank algorithm works on directed unweighted graphs, $G = (V, E)$. Let $In(v_i)$ be the set of vertices that point to a vertex v_i, and $Out(v_j)$ be the set of vertices to which v_i point. The score which reflects the importance of v_i is calculated by PageRank as:

$$S(\vartheta_i) = (1-d) + d \times \sum_{J \in In(\vartheta_i)} \frac{1}{|Out(\vartheta_j)|} S(\vartheta_j) \tag{3}$$

  where d is a damping factor that can be set between 0 and 1 (it has the role of integrating into the model the probability of jumping from a given vertex to another random vertex in the graph). In TextRank, the in-degree of a vertex equals its out-degree, since thegraph is undirected. Formally, let d denote a document, and w denote a word, then $d = w_1, w_2, .00.., w_n$. The weight of a vertex calculated by TextRank is:

$$WS(v_i) = (1-d) + d \times \sum_{\vartheta_j \in In(\vartheta_i)} \frac{w_{ij}}{\sum_{v_k \in Out(v_j)} w_{jk}} WS(v_j) \tag{4}$$

  where w_{ij} is the strength of the connection between two vertices v_i and v_j, and d is the damping factor, usually set to 0.85 (Brin and Page 1998). Intuitively, a vertex will receive a high score if it has many high-score neighbors. As noted before, after convergence, the top-scored vertices are selected as keywords. Adjacent keywords are then collapsed and output as a keyphrase. TextRank can be run without syntactic filtering during the formative stages of document representation, and therefore, all words can be considered. However, TextRank does not make full use of statistical information, such as the length and the position of the phrase, and its best score is achieved only when nouns and adjectives are used to create a uniformly weighted graph for text.
- CorePhrase Algorithm (Hammouda et al. 2005): CorePhrase is an algorithm for t opic discovery that uses keyphrase extraction from multi-document sets, and clusters that are based on frequent and significant shared phrases between documents. CorePhrase (1) first constructs a list of candidate keyphrases by comparing every pair of documents (using the Document Index Graph (DIG) (Hammouda and Kamel 2002), (2) it scores

each candidate keyphrase according to some criteria (df: document frequency, average weight, average phrase frequency, and average phrase depth), (3) ranks keyphrases according to their score, and finally selects a number of the top ranked keyphrases for output. CorePhrase achieves high accuracy in identifying the topic of a document cluster on a web dataset. Compared with other keyword-based cluster labeling algorithms, CorePhrase did not achieve any significant improvements in performance.

- SingleRank Algorithm (Wan and Xiao 2008): SingleRank algorithm follows the same approach as the TextRank algorithm but differs in the following aspect: while in TextRank, phrases containing the top-ranked words are selected, SingleRank does not filter out any low-scored words (vertices). Each candidate key phrase in SingleRank is given a score by summing the scores of its constituent words obtained from the graph. Then, the Top-N highest-scored phrases are extracted as keyphrases.
- ExpandRank Algorithm (Wan and Xiao 2008): In ExpandRank Wan & Xiao propose using a small number of nearest neighbor documents to provide more knowledge in improving single document keyphrase extraction. A graph-based ranking algorithm is applied on the expanded document set to make use of both: (1) the local information in the specified document and (2) the global information in the neighbor documents. Each document is represented by a term vector where each vector dimension corresponds to a word type present in the document, and its weight is computed by $tf-idf$. Specifically, for a particular document, the approach first finds its k nearest neighbors from a text corpus adopting the widely-used cosine measure. Then, the graph for the document is built using the co-occurrence statistics of the candidate words collected from a larger document set of $k+1$ documents, $D = d0, d1, d2, ..., dk$. In the graph, each vertex v_i corresponds to a candidate word type in D, and each edge connects two vertices v_i and v_j if the corresponding word types co-occur within a window of w units in the document set. The weight of an edge $w(v_i, v_j)$ is computed as follows:

$$w\left(v_i, v_j\right) = \sum_{d_p \in D} sim_{doc}(d_0, d_p) \times count_{dp}(v_i, v_j) \tag{5}$$

  where $sim_{doc}(d_0, d_p)$ is the cosine similarity to reflect the confidence value for document d_p in the expanded document set, and $count_{dp}(v_i, v_j)$ is the count of the occurrence between words v_i and v_j in d_p. However, ExpandRank builds upon SingleRank by incorporating neighboring documents without significantly improving its performance.
- KP-Miner System (El-Beltagy and Rafea 2009): The keyphrase extraction in the KP-Miner system is a three-step process: candidate keyphrases selection, candidate keyphrases weight calculation and finally keyphrase refinement. KPMiner is a non-learning, ranking-based system, which operates on n-grams and uses a modified version of $tf - idf$, where the document frequency for n-grams with n greater than one is assumed to be one. KP-Miner also boosts the weights of multiword candidates in proportion to the ratio of the frequencies of single word candidates to all candidates. In a reranking step, the $tf - idf$ of each term is recalculated based on the number of times it is subsumed by other candidates in the top 15 candidates list. KPMiner has the advantage of being configurable as the rules and heuristics adopted by the system are related to the general nature of documents (Arabic and English) and keyphrases. This requires the users of this system to understand the document(s) being input to the system and adjust it to their particular needs. Nevertheless, KP-Miner does not subtract the overlapping count during its weight stage calculation and only considers terms as candidates

which occur on their own in the text (i.e. candidates surrounded by punctuation marks or stop words). A further improvement to the system would entail allowing certain stop words to appear within the produced keyphrases.

– Keycluster Technique (Liu et al. 2009): Liu et al. involve an unsupervised clustering technique (henceforth Keycluster). They assign an importance score to a word, and then pick the top ranked words as keywords by clustering semantically similar candidates using Wikipedia and co-occurrence-based statistics. The Wikipedia and co-occurrence-based statistics ensure that the extracted keyphrases cover the entire document. The main idea is that each cluster represents a unique aspect of the document and takes a representative word from each cluster so that the document is covered from all aspects. However, Keycluster gives each topic equal importance scores while extracting keyphrases from each topic cluster process. Later, Liu et al. (2010) propose Topical PageRank (TPR), an approach that overcomes the above-mentioned weakness of KeyCluster. It runs TextRank multiple times for a document, once for each of its topics generated by a Latent Dirichlet Allocation LDA (Blei et al. 2003). Consequently, TPR ensures that the extracted keyphrases cover the main topics of the document. The final score of a candidate is computed as the sum of its scores for each of the topics, weighted by the probability that a given topic is in the document.
– CommunityCluster Method (Grineva et al. 2009): Grineva et al. present a novel method for key term extraction from text documents, where a document is modeled as a graph of semantic relationships between terms of that document. In this method, the terms related to the main topics of the document tend to bunch up into densely interconnected subgraphs or communities, while non-important terms fall into weakly interconnected communities, or even become isolated vertices. This technique introduces a criterion function to select groups that contain key terms discarding groups with unimportant terms. Weighing terms and determining the semantic relatedness between them exploits the information extracted from Wikipedia. Using such an approach creates two advantages. First, it allows for the effective processing of multi-themed documents. Second, it reduces the noisy unstructured text, such as navigational bars or headers in web pages, online chats, SMS, emails, newsgroups, and blogs. Method evaluations show that CommunityCluster outperforms Keycluster algorithm, Topical PageRank Algorithm and Yahoo! term extractor (in recall scores without losing precision).
– Sarkar et al. approach (Kamal Sarkar Mita Nasipuri 2010): In this novel Multilayer Perceptron (MLP) based approach, Sarkar et al. present a neural network-based approach to keyphrase extraction from scientific articles. This method consists of three primary components: (1) document preprocessing, (2) candidate phrase identification directed by a classifier based on a set of features (phrase frequency, phrase links to other phrases and Inverse Document Frequency, Phrase Position, Phrase Length and Word Length), (3) keyphrases are extracted using a neural network by training the MLP to predict whether a phrase is a keyphrase or not. Experiment results of the proposed system reveal that this system performs better than the publicly available keyphrase extraction system KEA. Later, Sarkar (2013) proposed a hybrid approach to extract keyphrases from medical documents, which combines two methods: the first one assigns weights to candidate keyphrases based on an effective combination of features such as term position, term frequency, idf, while the second one assigns weights to candidate keyphrases using some knowledge about their similarities to the structure, and characteristics of keyphrases available in the memory (stored list of keyphrases). The proposed approach results outperform some state-of-the-art keyphrase extraction systems like KEA and KP-Miner.

- TopicRank Algorithm (Bougouin et al. 2013): TopicRank is an enhanced variant of the TextRank algorithm(Mihalcea and Tarau 2004). It extracts noun phrases that represent the main topics of a document. The noun phrases are clustered into topics and used as vertices in a complete graph. The resulting graph stands as a topical representation of the document. Topics are scored using the TextRank ranking model and keyphrases are then extracted by selecting the most representative candidate from each of the top-ranked topics. This approach offers several advantages over existing graph-based keyphrase extraction methods: (1) first, as redundant keyphrase candidates are clustered, extracted keyphrases cover the main topics of the document better, (2) the use of a complete graph captures the relations between topics without any manually defined parameters, and (3) it induces better or similar performance than the state-of-the-art connection method that uses a co-occurrence window (Bougouin et al. 2013). However, in keyphrase selection, the experiments show that the current method did not provide the best solution that could have been achieved with the ranked clusters.
- CiteTextRank Algorithm (Gollapalli and Caragea 2014): CiteTextRank extracts keyphrases from research articles. It is a graph-based algorithm that incorporates evidence from both (1) document content, and (2) contexts in which the document is referenced within a citation network. This evidence is used in a flexible manner to score keywords that are later used to score keyphrases. Unlike other PageRank-based keyphrase extraction models, CiteTextRank builds a graph to compute scores (of nodes or vertexes) using information from multiple contexts (Mihalcea and Tarau 2004; Wan and Xiao 2008). Moreover, CiteTextRank effectively captures the notion of word importance across multiple global and citation contexts. CiteTextRank improves precision by 9-20% over state-of-the-art baseline algorithms such as TextRank, Keycluster, and Topical PageRank.
- Cho and Lee (2015) propose a new approach to select latent keyphrases (keyphrases that do not appear in the document). To do this, they extract candidate phrases by referencing neighbor documents, which are similar to the given document and evaluate the individual words of each candidate by considering the topic. They use Latent Dirichlet Allocation (Blei et al. 2003) to evaluate the importance of single words by considering the document's topics. The candidates are then measured by calculating the harmonic mean of their component's importance. By averaging, this method mitigates the overlapping problem. However, during averaging, this method does not consider the relationship between components. Consequently, performance deteriorates with more candidates.
- SGRank Algorithm (Danesh et al. 2015): Danesh et al. present a hybrid statistical-graphical algorithm that capitalizes on the heuristics of two major algorithm families of unsupervised keyphrase extraction: statistical and graph-based. The SGRank algorithm processes an input document in four stages. In the first stage, all possible N-grams (including the removal of the unlikely keyphrases) are extracted from the input text. In the second stage, the remaining n-grams are ranked based on a modified version $tf - idf$, in the third stage, the top-ranking candidates from the second stage are reranked based on additional statistical heuristics such as the position of the first occurrence and term length. A Position of First Occurrence factor (PFO) is defined according to the following formula:

$$PFO\,(t,d) = \log\left(\frac{cutoffPosition}{p(t,d)}\right) \tag{6}$$

where $p(t, d)$ is the position of term t's first occurrence in a document d, a $cutoffPosition$ value (a cutoff threshold of the first occurrence of candidates) is set to 3000 as it performed best in experiments on the development set, in the fourth and final stage, the ranking produced in the third stage is incorporated into a graph-based algorithm which produces the final ranking of keyphrase candidates. Experiment results on 2 datasets show that the average performance is considerably better than graph-based algorithms (KP-Miner (El-Beltagy and Rafea 2009) and TextRank (Mihalcea and Tarau 2004)).

- Smatana & Butka Alogorithm (Smatana and Butka 2016): Samatana and Butka propose a modification of single document keyphrase extraction models (without external information). This modification is based on hierarchical concepts. A Formal Concept Analysis (FCA) is used for the creation of a hierarchical concepts method, which organizes objects into a concept lattice (structure of hierarchically organized clusters known as formal concepts). In this technique, after preprocessing the input documents, the objects extracted are sentences or paragraphs, and the attributes are frequencies of terms in particular objects. For this input data, a conceptual model is created using an FCA-based algorithm known as a generalized one-sided concept lattice. Hierarchical concepts from this model are used for the extraction of keyphrases from a document. Compared with other standard keyphrase extraction weight metrics (tf, $tf - idf$, Chi-square, Information Gain), Smatana & Butka have shown improvement (in precision-recall) in keyphrase extraction, especially when uni-grams, bi-grams and tri-grams are extracted in the preprocessing step from documents. However, the authors did not compare their method with some of the state-of-the-art keyphrase extraction systems.
- KeyphraseDS Algorithm (Yang et al. 2017): KeyphraseDS is a novel document summarization mechanism that can organize scientific articles into multi-aspect and informative scientific project summaries by exploiting keyphrases. KeyphraseDS consists of three steps: keyphrase graph construction, semantic aspect generation, and content selection. First, keyphrases are extracted through a CRF-based model exploiting various features, such as syntactic features, correlation features, etc. Spectral clustering is then performed on the keyphrase graph to generate different aspects, where the semantic relatedness between keyphrases is computed through knowledge-based similarity and topic-based similarity. Significant sentences are then selected with respect to the generated aspects through integer linear programming (ILP), which takes semantic relevance, semantic diversity, and keyphrase salience into consideration (Yang et al. 2017). Extensive experiments, including automatic evaluation and human evaluation, demonstrate the effectiveness and the feasibility of this method. Compared with other models (KEA, SVM), a CRF-based model yields the best performances in both precision and recall rates in keyphrase extraction.

The deep learning approaches Deep learning (DL) (also known as hierarchical learning or deep structured learning) is part of the machine learning methods family based on learning data representations (see Fig. 7). DL allows computational models that are composed of multiple processing layers to learn these representations of data with multiple levels of abstraction. The DL architectures models include: Deep Neural Networks (DNNs), and Deep Belief Networks (DBNs), Recurrent Neural Networks (RNNs), Deep Recurrent Neural Networks (DRNNs). DL (LeCun et al. 2015) techniques present promising approaches for various NLP tasks. As a result, DL models have recently gained popularity and interest in AKPE systems (Zhang et al. 2016; Meng et al. 2017). Recurrent neural networks

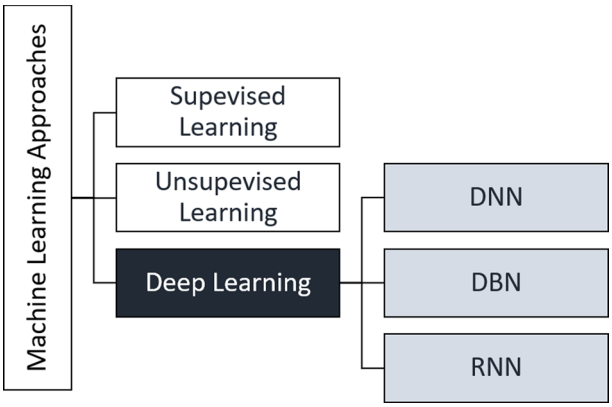

Fig. 7 Deep learning approaches

(RNNs) (Elman 1990) have been applied to several sequential prediction tasks. RNNs and bidirectional RNNs are considered the most suitable model in NLP tasks. They deal with a keyphrase as a sequence of tokens and incorporate information from previous ones. They are beneficial, especially when making a decision on the current token. The information provided by the subsequent tokens is generally valuable. Below, we examine some representative AKPE systems that adopt the DL approach.

– Meng et al. (2017) apply an RNN-based generative model (deep keyphrase generation) to predict keyphrases. They incorporated a copying mechanism in RNN, which enables the model to successfully predict "absent keyphrases" that rarely occur (phrases that do not match any contiguous subsequence of source text). The authors built a dataset comprised of 20,000 scientific documents to train and evaluate the system.
– Zhang et al. (2016) proposed a novel RNN model to tackle the problem of extracting relevant keyphrases from tweets, where the length limitations of Twitter-like sites make the performance of existing AKPE systems decrease. To solve this problem, the RNN model combines keywords and context information. In addition, a massive dataset of tweets with gold standard keyphrases was constructed to evaluate the proposed approach.

3.1.3 Evaluation stage

The goal of this final step is to evaluate the extracted keyphrases. The familiar metrics for evaluating keyphrase extraction systems are:

- **The Manual Evaluation (ME):** ME is based on human judgments. Various authors have used human judges to evaluate the results of AKPE techniques (Turney 2000; Matsuo and Ishizuka 2004), as well as to determine if the retrieved keyphrases are representative of a document's content. However, human evaluation of the extracted keyphrases is very expensive and time-consuming, and it is not appropriate for any kind of parameter tuning.
- **The Automatic Evaluation (AE):** the AE relies on (1) matching and (2) scoring. (1) Matching a set of annotated gold standard keyphrases with a ranked list of extracted keyphrases (output of a keyphrase extraction system). This evaluation approach is also

adopted by the SemEval-2010 shared task on keyphrase extraction (Kim et al. 2010). This *exact matching* can be true or false, depending on whether gold standard keyphrases and the extracted ranked keyphrases are equivalent according to the matching strategy. After matching, (2) scoring the output using evaluation metrics is performed. Among these metrics, there are the well-known: (1) precision (P), recall (R), and F-score (F), (2) keyphrase ranking (as an evaluation metric) to distinguish between systems that score the same number of exact matches (Liu et al. 2010), or to determine which systems rank the most correct keyphrases above incorrect candidate keyphrases (Zesch and Gurevych 2009), (3) the information retrieval R-precision (R-p) (Zesch and Gurevych 2009), which is defined as the precision when the number of retrieved documents equals the number of relevant documents in the document collection. An R-precision of 1.0 is equivalent to perfect keyphrase ranking and perfect recall. This means that a system that achieves a perfect (R-p) value ranks all the keyphrases on top of the non-keyphrases (Zesch and Gurevych 2009). However, the exact matching of keyphrases is an excessively strict condition, which is problematic since it overlooks near matches that are basically semantically identical (such as different grammatical forms, and synonyms). To overcome this issue, in some cases, near matches have also been considered. Some researchers have developed a number of systems for automatic evaluation via near or partial matching. Mihalcea and Tarau (2004) and Jarmasz and Barriere (2004) have suggested treating semantically-similar keyphrases as correct based on similarities calculated over a large corpus. Similar to Mihalcea and Tarau (2004) and Jarmasz and Barriere (2004), Medelyan and Witten (2006) used semantic relations defined in a thesaurus to treat near match keyphrases. Later, Zesch and Gurevych (2009) used an n-gram based approach relative to the gold standard to compute near matches. Additionally, they proposed some metrics that consider semantic similarity and character n-grams in order to differentiate between probable near matches and completely erroneous keyphrases (Zesch and Gurevych 2009; Kim et al. 2010). Nevertheless, these systems only give a limited solution as they can only detect a subset of near-misses to exact match terms (Kim et al. 2010).

4 Discussion and recommendations

Reviewing the 34 noteworthy keyphrase extraction techniques revealed interesting trends in the various stages of keyphrase extraction. The performance of these techniques is presented in Table 4. The evaluated datasets are presented in Table 3. Table 4 presents a characterization summary of the aforementioned AKPE along with the used dataset, target task, ranking approach, features, and best evaluation scores achieved by each technique as it is described in their papers. In Table 5, we compare the most prominent and representative of these aforementioned AKPE techniques. The evaluation was carried out on the well-known shared dataset "SemEval-2010",[6] which is composed of 284 technical and scientific documents (40 trial documents, 144 training documents, and 100 test documents) (Kim et al. 2010). Additionally, in this dataset, a set of gold-standard keyphrases is available for the evaluation. For each article, three sets of gold-standard keyphrases are provided: author-assigned keyphrases, reader-assigned keyphrases, and a combination of the two previous sets. In the two tables, P, R, and F denote precision, recall, and F-score respectively.

[6] http://semeval2.fbk.eu/semeval2.php?location=data

Table 3 Some evaluated datasets in this survey

Name or source	*Number of files*	*Files type*	*Dataset domain*
NZDL (New Zealand Digital Library)	1800	Technical reports	Multi domains
Inspec	500 or 2000	Paper abstracts	Scientific
uw-can-data	314	Web pages	Multi domains
DUC 2001	308	News Articles	News
citeulike.org collection	–	Scientific documents	Scientific
CSTR, NASA, FIPS, Journals, Aliweb	502	Documents	Multi domains
ACM Digital Library, Cogprints, Springerlink, Wikipedia	570	Scientific papers	Scientific
Geeking with Greg, DBMS2, Stanford Infoblog	252	Blogs	Technical
Elsevier, Springer	150	Full journal articles	Scientific
CiteSeerx	790	Papers published in WWW and KDD conferences	Scientific
ACL Anthology Network (AAN) corpus	+ 20000	Research Papers	Scientific
Geeking with Greg, DBMS2, Stanford Infoblog	252	Blogs	Technical Domains
Reuters collection	21578	NewsWire articles	News
SemEval-2010 Task-5	284	Scientific articles	Technical and Scientific

4.1 Discussion

The top-performing systems return F-scores in the upper forties (see Table 5). Apparently, this number is low, which indicates that much more improvement can be made. However, since keyphrase extraction is a subjective task, an F-score of 100% is not possible. In Table 4, the F-score of the best-performing systems on their testing data is achieved by: KeyphraseDS (Yang et al. 2017), Keycluster Algorithm (Liu et al. 2009),Community Cluster Algorithm (Grineva et al. 2009) and the Maui Algorithm (Medelyan et al. 2009). In Table 5, Kp-Miner system (El-Beltagy and Rafea 2009), HUMB system (Lopez and Romary 2010), DPM-Index system (Haddoud 2014) showed the best performance on the shared 'SemEval-2010/Task-5' dataset (Kim et al. 2010). Given these results and the analysis of several aforementioned AKPE techniques, various challenges are revealed in different stages of keyphrase extraction: candidate keyphrases identification, feature engineering, candidates ranking and evaluation. First, most of these systems apply pre-processing before candidate identification. In **the candidate identification stage**, numerous systems used either POS-based or n-grams, noun phrase chunks, or a combination of the two of them. However, in this step, there is a difficulty in dealing with the structure of a keyphrase which is sometimes irregular and comprised of only a single word or multi-word noun phrase or multiple multiword noun phrases connected by prepositions. Additionally, most systems face different challenging tasks whereby they: (1) fail at predicting candidate

Table 4 Characteristic summary of some aforementioned AKPE techniques along with the used features, dataset, target domain, approach and performance rates

Technique/ Contributor	Target/ Domain	Dataset	Approach	Features		Top N/ P R F			
				Internal	*External*	N	P	R	F
KEA System (Witten et al. 1999)	Keyphrase Extraction	NZDL	Supervised (Naïve Bayes model)	*First occurrence position*, $tf-idf$	–	5 10 15	0.300 – 0.165	– – –	– – –
GenEx Algorithm (Turney 2000)	Keyphrase Extraction	75 Journal Articles, 311 Email Messages, 266 Web Pages	Supervised (C4.5 Decision Tree)	*Term Frequency* (tf), *first occurrence position, phrase Length*	–	5 10 15	0.239 – 0.128	– – –	0.118 – –
TextRank Algorithm (Mihalcea and Tarau 2004)	Keywords and Sentence Extraction	Inspec	Unsupervised (graph-based)	*POS, distance between word occurrences*	–	5 10 15	– 0.142 –	– 0.125 –	– 0.127 –
CorePhrase Algorithm (Mihalcea and Tarau 2004)	Topic Discovery using Keyphrase Extraction	uw-can-data	Unsupervised (graph-based)	*DF-average weight, Phrase frequency, Average phrase depth*	–	– – –	– 0.41 –	– – –	– – –
Single Rank Algorithm (Wan and Xiao 2008)	Single Document Keyphrase Extraction	DUC 2001	Unsupervised (graph-based)	*POS tagging, neighborhood, Level saliency score*	–	5 10 15	0.34 0.35 0.26	0.25 0.28 0.31	0.26 0.31 0.30
Maui Algorithm (Medelyan et al. 2009)	Tags Extraction	citeulike.org collection	Supervised (Naïve Bayes and bagged decision trees models)	$tf-idf$, *position of first occurrence, Keyphraseness, Phrase length, spread, node degree*	*Wikipedia based keyphraseness, spread, semantic relatedness, Inverse Wikipedia linkage*	– – –	– 0.457 –	– 0.486 –	– 0.471 –

Table 4 (continued)

Technique/ Contributor	Target/ Domain	Dataset	Approach	Features		Top N/ P R F			
				Internal	*External*	N	P	R	F
KP-Miner System (El-Beltagy and Rafea 2009)	keyphrase Extraction (Agriculture domain)	CSTR, NASA, FIPS, Journals, Aliweb	Unsupervised (Weighting model)	$tf-idf$, *first occurrence position, Boosting Factor for weight bias*	–	7 15 20	0.214 0.143 0.214	0.277 0.358 0.277	0.241 0.205 0.241
Core Word Expansion Algorithm (You et al. 2009)	keyphrase Extraction	ACM digital library, Cogprints, Springerlink, Wikipedia	Supervised; Unsupervised	*Inverse document frequency* (idf), *N-gram*	–	5 15 20	0.272 0.192 0.169	0.189 0.267 0.353	0.223 0.223 0.228
Community Cluster Algorithm (Grineva et al. 2009)	Keyphrase Extraction	Geeking with Greg, DBMS2, Stanford Infoblog	Unsupervised (semantic graph-based)	*N-gram*	*Wikipedia-based keyphraseness*	– – –	– 0.351 –	– 0.615 –	– 0.447 –
Keycluster Algorithm (Liu et al. 2009)	Keyphrase Extraction	Inspec	Unsupervised (topic clustering model)	*Co-occurrence-based term relatedness* (idf), *N-gram*	*Wikipedia-based term relatedness*	– Using spectral clustering –	– 0.350 –	– 0.660 –	– 0.457 –
MLP based Algorithm (Kamal Sarkar Mita Nasipuri 2010)	Keyphrase Extraction	Elsevier, Springer	Supervised (MLP model)	*Noun Phrase, Phrase frequency, phrase links to other phrases,* idf, *phrase position, phrase length and word length*	–	5 10 15	0.34 0.22 0.17	0.35 0.46 0.51	0.344 0.297 0.255

Table 4 (continued)

Technique/ Contributor	Target/ Domain	Dataset	Approach	Features		Top N/ P R F			
				Internal	*External*	N	P	R	F
CeKE Algorithm (Bulgarov and Caragea 2015)	Keyphrase Extraction	CiteSeerx	Supervised (Naïve Bayes model)	$tf - idf$; *relativePos, POS; inCited, inCiting; citation tf-idf, first position, tf-idf-Over, firstPosUnder, keyphraseness*	–	WWW KDD –	0.25 0.254 –	0.46 0.440 –	0.32 0.321 –
KeyphraseDS System (Yang et al. 2017)	Document Summarization	ACL Anthology Network (AAN) corpus	Supervised (CRF-based model, ILP-based optimization, LDA model)	*Syntactic features(POS and Relation), correlation features (MutInfoPre, MutInfoNext), semantic relatedness*	*InWiki*	– – –	– 0.38 –	– 0.62 –	– 0.47 –
KeyEx Method (Xie et al. 2017)	Keyphrase Extraction	Reuters 21578 collection, SemEval-2010	Supervised ()	$tf-idf$; *POS; first occurrence position, statistical pattern features (pattern support, pattern length,number of closed patterns,* $PFIDF$ *value)*	–	– – –	– 0.134 –	– 0.35 –	– 0.19 –

Table 5 Characteristic summary of the most representative AKPE techniques and their performance on the shared "SemEval-2010 dataset"

Technique/ Contributor	Approach	Features		Top N/ P R F			
		Internal	*External*	N	P	R	F
KP-Miner (El-Beltagy and Rafea 2009)	Unsupervised	$tf-idf$, *first ccurrence position, boosting factor for weight Bias*	–	5 10 15	0.296 0.233 0.253	0.123 0.205 0.261	0.253 0.261 0.258
SZTERGAK (Berend and Farkas 2010)	Supervised	$tf-idf$, *first occurrence, phrase length, POS, suffix, acronymity, PMI, syntactic*, $sf-isf$, *keyphraseness*	*Wikipedia*	5 10 15	0.342 0.285 0.248	0.117 0.194 0.254	0.174 0.231 0.251
HUMB (Lopez and Romary 2010)	Supervised	*Position of a term, position of the first occurrence, phraseness, informativeness, keywordness, length of the term*	*Wikipeia keyphrase-ness, GRISP based term, GRO-BID/TEI*	5 10 15	0.300 0.320 0.272	0.133 0.218 0.278	0.198 0.260 0.275
DPM-index (Haddoud 2014)	Supervised	*Length n of the n-gram t in words;* tf; idf; $tf-idf$, *first position, first sentence, head frequency, average sentence length, substrings frequencies sum, generalized dice coefficient, maximum likelihood estimate; kullback–leibler divergence; document phrase maximality index(DPM-Index); DPM-index cross tf-idf, tf-idf ratio of the term and its main compound, k-Means of the normalized positions (k = 1, 2)*	–	5 10 15	0.448 0.362 0.283	0.153 0.247 0.289	0.228 0.294 0.286
KeyEx (Xie et al. 2017)	Supervised	$tf-idf$; *POS; first occurrence position, statistical pattern features (pattern support, pattern length,number of closed patterns,* $PFIDF$ *value)*	–	5 10 15	0.720 0.600 0.510	0.150 0.280 0.360	0.248 0.381 0.422

keyphrases with different sequential orders, or (2) encounter keyphrases that contain synonyms or abbreviations. In **feature engineering stage**, some authors applied a variety of features: statistical, structural and syntactic/morphological. Other authors added external resource-based features, such as Wikipedia and external terminological corpora databases. Nevertheless, the use of external knowledge bases is more computationally expensive than baseline features. Information such as document structure or document statistics appears to be effective for automatic keyphrase extraction across the results (see Tables 4 and 5). However, these features only detect the relevance of each word in a document based on the statistics of word occurrence and co-occurrence, and some of them are unable to benefit from the full semantics that underlies the document content. To **rank candidates**, *supervised systems* have used different machine learners such as maximum entropy, logistic regression, the learn-to-rank classifier based on SVMrank, Naïve Bayes and C4.5 bagged decision trees. However, supervised approaches depend on the predetermination of features before running the system. They use off-line analysis and they need a considerable amount of computation time for training. Consequently, these features cannot be modified or learned during the system process. It is difficult to enumerate all the features associated with a specific domain because the linguistic, statistical and external knowledge of the text should be exploited. Additionally, some keyphrases are positively recognized as candidate keyphrases but fail to be ranked at the top for extraction. In *unsupervised systems*, as shown in Tables 4 and 5, the F-score of the best-performing systems was for those that used topic clustering methods or simple probabilistic ones. For systems that used graph-based methods, there is a limitation in covering the various relevant topics of a document. Nevertheless, they work well for single documents. However, unsupervised keyphrase extraction techniques sometimes produce less relevant keyphrases than supervised ones, and they are less specific. In *deep learning systems*, most of them used the RNN model. It was used for the joint processing of keyword ranking, keyphrase generation, and keyphrase ranking. Especially, this model enhanced the generation of embedded keyphrases. Despite the fact that DL reduces the need for engineering features, they are computationally expensive in training. However, DL approaches require a large amount of data to perform better. Additionally, there are just a few AKPE systems using DL. In **the evaluation stage**, traditionally, most AKPE systems have been evaluated using the proportion of Top-N candidates that exactly match the gold standard keyphrases (Frank et al. 1999; Witten et al. 1999; Turney 2000). In most papers, the authors commonly manually assign keyphrases based on their semantic meaning, instead of following the written content in the document. Consequently, current evaluation methods do not cover keyphrases that are semantically relevant. In this case, the exact matching of keyphrases can be very strict and problematic because it ignores near matches that are largely semantically identical. Furthermore, it is difficult to measure the relative superiority of different machine learning approaches over the task, as they were combined with different candidate selection techniques and feature sets. There is no doubt, however, that there is certainly room for improvement on the APKE task. Generally, some limits and drawbacks of existing AKPE systems can be summarized as follows: (1) the redundancy is induced by semantically equivalent keyphrases, (2) overgeneration: frequency of keyphrases alone cannot be a criterion for extracting keyphrases, (3) infrequency: many systems overlook the infrequent relevant keyphrases, and focus on frequent words in the associated documents, (4) as the length or the number of documents increases, the global performance of all ranking algorithms can drop, (5) the evaluation is insufficient since many studied systems have evaluated their approaches on only one dataset and (6) the evaluation of exact matches is deemed insufficient to cover all relevant keyphrases.

4.2 Recommendations

The aforementioned discussion reveals significant trends across different stages of AKPE. The objective of this study is to (i) enhance the understanding of the AKPE process and differences between the various approaches and techniques used, (ii) provide the guidelines and necessary components to be considered in future AKPE solutions. The analysis regarding every stage, approach, and technique have already been given in the above sections. Thus, for future AKPE works (depending on the final systems' objectives), and to overcome the mentioned limitations, we suggest some recommendations for each stage of the AKPE process.

- Enhancing **the preprocessing stage** and candidate selection stages will ameliorate the selection of good (the morphologic form) candidate phrases that can improve the time, quality and complex efficiency of a keyphrase extraction system. Apparently, using a combination between POS tag sequence (noun phrase based) and n-grams showed worthwhile results in candidates' identification.
- In **the feature engineering stage**, the success of AKPE systems in this stage depends primarily on the quality, variety, and quantity (structural, linguistic, statistic, semantic) of the used features. Apparently, $tf - idf$, co-occurrence, length of the candidate phrase, the relative position of the phrase, the keyphraseness, have revealed worthwhile results in different systems (Lopez and Romary 2010; Haddoud 2014). Involving semantic information is also very important in reducing redundancy problems, and exploiting semantic relationships between words may potentially help in extracting more relevant and informative keyphrases (see Table 2). While the use of external knowledge bases is computationally expensive, using them has many benefits: (1) they will make the keyphrases more grammatically understandable, (2) they can be used to boost the infrequent relevant keyphrases which may solve the infrequency problem and (3) they can increase the precision, recall and exact match rates.
- In **the candidates ranking stage**, *supervised approaches* are more precise in extracting the exact match keyphrases, especially when the concerned classes are determined. They can improve the quality of keyphrases with repeated training. It is notable that the use of bagged decision trees C4.5 with a variety of good features performs better than other supervised models (Haddoud 2014). Otherwise, using *unsupervised approaches* has the benefit of being more widely applicable, since their models do not require any pre-knowledge of the domain or trained data. These models have some practical advantages such as (1) notable time saving, (2) computing resource saving and (3) the ability to produce a large high-quality keyphrase list, which makes the system more effective especially in real-time and query refinement applications. In the case of small and medium-sized documents ($\approx 100 - 1000$ tokens), using co-occurrence graph-based ranking is highly recommended, and it outperforms supervised systems by a wide margin (Zesch and Gurevych 2009). Awareness of the domain or topic is necessary in order to obtain robust systems. Topic-based clustering models have notable potential (see Table 4). They use meaningful concepts or contents of documents, and they undertake an important role in predicting and extracting latent keyphrases (keyphrases that do not appear in documents) by using methods such as LDA. Similarly, topic models can aid to solve the problem of infrequency. Indeed, future systems should take into account keyphrases based on an understanding of the text, regardless of the presence or absence of keyphrases in the document text. Recently *deep learning approaches* have attracted the attention of the research community. Indeed, APKE can gain significant benefits

Table 6 The strengths and weaknesses of supervised, unsupervised and DL approaches for the AKPE task

Approaches	Strengths	Weaknesses
Supervised	(+) More precision and very specific (in the definition of the labels).	(-) Complex models (need to understand very well and label the inputs).
	(+) The input data is well known and is labeled.	(-) Needs to relearn and establish the model every time the domain changes (impractical to label training set from time to time by users).
	(+) Better decision boundary.	
	(+) Parameterized heuristic rules.	(-) Not suitable for real-time tasks (off-line analysis).
	(+) Ability to determinate the number of classification classes.	(-) Need gold standard keyphrases and training data to compare and evaluate.
	(+) More accurate results.	(-) Often dependent on the domain.
		(-) Not suitable for dynamic and growing data.
		(-) Time-consuming in training (computation time) and extraction phases.
		(-) Uses absolute feature values.
Unsupervised	(+) Less complexity (no labeling of data inputs is needed) and more flexible.	(-) Not specific in data sorting and output definitions.
	(+) Suitable for small and medium-sized documents.	(-) No coverage guarantees for all main topics.
	(+) Takes place in real time.	(-) Low accuracy of the results.
	(+) Does not exploit any manually tagged corpus or training data.	(-) input data is not known and not labeled.
	(+) Time efficient.	
	(+) Good solution for real-time applications.	
	(+) Simplicity-syntactic representation requires almost no language-specific, advanced knowledge linguistics or processing.	
	(+) Takes into account term co-occurrence patterns.	
	(+) Domain and language independent.	
	(+) More availability of unlabeled data.	
Deep Learning	(+) Performs very well on text data.	(-) Not suitable as general-purpose algorithms.
	(+) Reduces the need for feature engineering.	(-) Requires a very large amount of data.
	(+) Adaptable architecture.	(-) Computationally intensive to train.
	(+) Can be easily updated with new data (best-in-class performance on problems).	(-) Requires much more expertise to tune.
	(+) Can learn extremely complex patterns.	(-) Needs high-performance hardware.
	(+) Learns from high-dimensional data.	(-) Needs high-performance hardware.
		(-) Does not have much in the way of strong theoretical foundation (the learning process is considered a Black-Box).
		(-) What is learned is not easy to comprehend.

from using DL. DL models aid in combining keywords and context information to perform the keyphrase extraction task. We recommend the use of DL models, especially for extra-large document sets. There is no doubt that we will continue to see a growth in the application of DL methods on the APKE.

Table 6 illustrates some of the strengths and weaknesses of supervised, unsupervised and DL approaches. Depending on the application at hand, this table can give an idea of the approaches that can be used before building an AKPE system.

– In **the evaluation stage**, we believe that the context and topics of keyphrases should be considered for two reasons: (1) better performance in extracting keyphrases and (2) sufficient coverage of each document. To better improve near matches, we suggest treating semantically similar keyphrases as correct based on similarities computed over a large corpus (Mihalcea and Tarau 2004; Jarmasz and Barriere 2004), or using semantic relations defined in a thesaurus (Medelyan and Witten 2006). Indeed, more semantically-motivated evaluation received less attention and should be engaged to give more accurate keyphrase suitability. A suitable keyphrase should satisfy the following properties: (1) clear and understandable to readers, (2) semantically relevant to the document topic, (3) sufficient coverage (cover all the document entities). Additionally, it is necessary to evaluate keyphrase extraction systems on multiple and variant datasets to gain a better view of performance. Certainly, evaluating the same system can vary depending on the type, volume, and content of the dataset.

5 Conclusion

In this paper, we have presented a survey and trends on the automatic keyphrase extraction task. Many approaches were used (supervised, unsupervised and deep learning) in this task and have proven their efficiency on different datasets and on the one most commonly used (Semeval-2010). Our analysis focuses on the overall automatic keyphrase extraction process. During our study, 34 existing techniques revealed that there are several major challenges ahead. Challenges are related to every stage of the AKPE process such as: (1) the redundancy induced by semantically equivalent keyphrases, (2) the infrequency of latent keyphrases: many systems overlook the infrequent relevant keyphrases, and focus on frequent words in associated documents, (3) the length or number of documents that increase, which can be solved by deep learning approaches, but these approaches need more attention from the research community and (4) the commonly used evaluation methods that are still insufficient for considering latent and semantically related keyphrases. We have suggested several recommendations regarding each stage of this task to (i) address these challenges, and to (ii) enhance this task in future works.

References

Barker, K., & Cornacchia, N. (2000). Using noun phrase heads to extract document keyphrases. In: conference of the canadian society for computational studies of intelligence, pp. 40–52. Springer.

Berend, G. (2011). Opinion expression mining by exploiting keyphrase extraction. In: Proceedings of the 5th international joint conference on natural language processing. Asian Federation of Natural Language Processing.

Berend, G., & Farkas, R. (2010). SZTERGAK: Feature engineering for keyphrase extraction. In: proceedings of the 5th international workshop on semantic evaluation, pp. 186–189. Association for Computational Linguistics.

Blei, D.M., Ng, A.Y., Jordan, M.I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Bougouin, A., Boudin, F., Daille, B. (2013). TOPICRANK: Graph-based topic ranking for keyphrase extraction. In: International joint conference on natural language processing (IJCNLP), pp. 543–551.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*(1-7), 107–117.

Bulgarov, F., & Caragea, C. (2015). A comparison of supervised keyphrase extraction models. In: Proceedings of the 24th international conference on World Wide Web, pp. 13–14. ACM.

Chandrasekar, R., James, C.F.I., Watson, E.B. (2006). System and method for query refinement to enable improved searching based on identifying and utilizing popular concepts related to users' queries. *US Patent*, *7*, 136,845.

Chen, M., Sun, J.T., Zeng, H.J., Lam, K.Y. (2005). A practical system of keyphrase extraction for web pages. In: Proceedings of the 14th ACM international conference on information and knowledge management, pp. 277–278. ACM.

Cho, T., & Lee, J.H. (2015). Latent keyphrase extraction using LDA model. *Journal of Korean Institute of Intelligent Systems*, *25*(2), 180–185.

Danesh, S., Sumner, T., Martin, J.H. (2015). SGRANK: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. In: Proceedings of the fourth joint conference on lexical and computational semantics, pp. 117–126.

D'Avanzo, E., & Magnini, B. (2005). A keyphrase-based approach to summarization: The LAKE system at DUC-2005. In: Proceedings of DUC.

Do, N., & Ho, L. (2015). Domain-specific keyphrase extraction and near-duplicate article detection based on ontology. In: International conference on computing & communication technologies, research, innovation, and vision for the future (RIVF), pp. 123–126. IEEE.

Dostal, M., & Ježek, K. (2011). Automatic keyphrase extraction based on NLP and statistical method. In: Dateso Conference. Západoćeská Univerzita v Plzni.

El-Beltagy, S.R., & Rafea, A. (2009). KP-MINER: A keyphrase extraction system for English and Arabic documents. *Information Systems*, *34*(1), 132–144.

El Idrissi, O., Frikh, B., Ouhbi, B. (2014). HCHIRSIMEX: An extended method for domain ontology learning based on conditional mutual information. In: 3rd IEEE international colloquium in information science and technology (CIST), pp. 91–95. IEEE.

Elman, J.L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179–211.

Elovici, Y., Shapira, B., Last, M., Zaafrany, O., Friedman, M., Schneider, M., Kandel, A. (2010). Detection of access to terror-related web sites using an advanced terror detection system (ATDS). *Journal of the association for information science and technology*, *61*(2), 405–418.

Ferrara, F., Pudota, N., Tasso, C. (2011). A keyphrase-based paper recommender system. In: Italian research conference on digital libraries, pp. 14–25. Springer.

Fortuna, B., Grobelnik, M., Mladenić, D. (2006). Semi-automatic data-driven ontology construction system. In: Proceedings of the 9th international multi-conference information society, pp. 223–226.

Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C., Nevill-Manning, C.G. (1999). Domain-specific keyphrase extraction. In *Proceedings of the 16th international joint conference on artificial intelligence, IJCAI '99* (pp. 668–673). San Francisco: Morgan Kaufmann Publishers Inc. http://dl.acm.org/citation.cfm?id=646307.687591.

Frantzi, K.T., Ananiadou, S., Tsujii, J. (1998). The C-VALUE/NC-VALUE method of automatic recognition for multi-word terms. In: International conference on theory and practice of digital libraries, pp. 585–604. Springer.

Frikh, B., Djaanfar, A.S., Ouhbi, B. (2011). A new methodology for domain ontology construction from the Web. *International Journal on Artificial Intelligence Tools*, *20*(06), 1157–1170.

Gollapalli, S.D., & Caragea, C. (2014). Extracting keyphrases from research papers using citation networks. In: AAAI, pp. 1629–1635.

Gong, Z., & Liu, Q. (2009). Improving keyword based web image search with visual feature distribution and term expansion. *Knowledge and Information Systems*, *21*(1), 113–132.

Grineva, M., Grinev, M., Lizorkin, D. (2009). Extracting key terms from noisy and multitheme documents. In: Proceedings of the 18th international conference on World Wide Web, pp. 661–670. ACM.

Gutwin, C., Paynter, G., Witten, I., Nevill-Manning, C., Frank, E. (1999). Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, *27*(1-2), 81–104.

Haddoud, M. (2014). Abdeddaïm, S.: Accurate keyphrase extraction by discriminating overlapping phrases. *Journal of Information Science*, *40*(4), 488–500.

Haddoud, M., Mokhtari, A., Lecroq, T. (2015). Abdeddaïm, S.: Accurate keyphrase extraction from scientific papers by mining linguistic information. In: CLBib@ ISSI, pp. 12–17.

Hammouda, K.M., & Kamel, M.S. (2002). Phrase-based document similarity based on an index graph model. In: Proceedings of international conference on data mining (ICDM), pp. 203–210. IEEE.

Hammouda, K.M., Matute, D.N., Kamel, M.S. (2005). COREPHRASE: Keyphrase extraction for document clustering. In: International workshop on machine learning and data mining in pattern recognition, pp. 265–274. Springer.

Han, J., Kim, T., Choi, J. (2007). Web document clustering by using automatic keyphrase extraction. In: 2007 IEEE/WIC/ACM international conferences on web intelligence and intelligent agent technology - workshops, pp. 56–59. IEEE.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In: Proceedings of the fifteenth conference on uncertainty in artificial intelligence, pp. 289–296. Morgan Kaufmann Publishers Inc.

Huang, C., Tian, Y., Zhou, Z., Ling, C.X., Huang, T. (2006). Keyphrase extraction using semantic networks structure analysis. In: 6th international conference on data mining (ICDM'06), pp. 275–284. IEEE.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of the 2003 conference on empirical methods in natural language processing, pp. 216–223. Association for Computational Linguistics.

Hulth, A., & Megyesi, B.B. (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, pp. 537–544. Association for Computational Linguistics.

Jarmasz, M., & Barriere, C. (2004). Using semantic similarity over tera-byte corpus, compute the performance of keyphrase extraction. Proceedings of CLINE.

Jiang, X., Hu, Y., Li, H. (2009). A ranking approach to keyphrase extraction. In *Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval, SIGIR '09* (pp. 756–757). New York: ACM. https://doi.org/10.1145/1571941.1572113.

Jones, S., & Staveley, M.S. (1999). PHRASIER: A system for interactive document retrieval using keyphrases. In: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval, pp. 160–167. ACM.

Jungiewicz, M., & Łopuszyński, M. (2014). Unsupervised keyword extraction from Polish legal texts. In: International conference on natural language processing, pp. 65–70. Springer.

Kamal Sarkar Mita Nasipuri, S.G. (2010). A new approach to keyphrase extraction using neural networks. arXiv:1004.3274.

Kelleher, D., & Luz, S. (2005). Automatic hypertext keyphrase detection. In: IJCAI, vol. 5, pp. 1608–1609.

Kim, S.N., & Kan, M.Y. (2009). Re-examining automatic keyphrase extraction approaches in scientific articles. In: Proceedings of the workshop on multiword expressions: identification, interpretation, disambiguation and applications, pp. 9–16. Association for Computational Linguistics.

Kim, S.N., Medelyan, O., Kan, M.Y., Baldwin, T. (2010). SEMEVAL-2010 Task 5: Automatic keyphrase extraction from scientific articles. In: Proceedings of the 5th international workshop on semantic evaluation, pp. 21–26. Association for Computational Linguistics.

Krovetz, R., & Croft, W.B. (1992). Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems (TOIS)*, *10*(2), 115–141.

Kumar, N., & Srinathan, K. (2008). Automatic keyphrase extraction from scientific documents using n-gram filtration technique. In: Proceedings of the eighth ACM symposium on document engineering, pp. 199–208. ACM.

Landauer, T.K., Foltz, P.W., Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, *25*(2-3), 259–284.

Leake, D.B., Maguitman, A., Reichherzer, T., Cañas, A.J., Carvalho, M., Arguedas, M., Brenes, S., Eskridge, T. (2003). Aiding knowledge capture by searching for extensions of knowledge models. In: Proceedings of the 2nd international conference on knowledge capture, pp. 44–53. ACM.

LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

Liu, F., Pennell, D., Liu, F., Liu, Y. (2009). Unsupervised approaches for automatic keyword extraction using meeting transcripts. In: Proceedings of human language technologies: the 2009 annual conference of the North American chapter of the Association for Computational Linguistics, pp. 620–628. Association for Computational Linguistics.

Liu, W., Chung, B.C., Wang, R., Ng, J., Morlet, N. (2015). A genetic algorithm enabled ensemble for unsupervised medical term extraction from clinical letters. *Health Information Science and Systems*, *3*(1), 5.

Liu, Z., Huang, W., Zheng, Y., Sun, M. (2010). Automatic keyphrase extraction via topic decomposition. In: Proceedings of The 2010 conference on empirical methods in natural language processing, pp. 366–376. Association for Computational Linguistics.

Liu, Z., Li, P., Zheng, Y., Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In: Proceedings of the 2009 conference on empirical methods in natural language processing: vol. 1, pp. 257–266. Association for Computational Linguistics.

Lopez, P., & Romary, L. (2010). HUMB: Automatic key term extraction from scientific articles in GROBID. In: Proceedings of the 5th international workshop on semantic evaluation, pp. 248–251. Association for Computational Linguistics.

Lops, P., De Gemmis, M., Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In: Recommender Systems Handbook, pp. 73–105. Springer.

Matsuo, Y., & Ishizuka, M. (2004). Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, *13*(01), 157–169.

Matsuo, Y., Mori, J., Hamasaki, M., Nishimura, T., Takeda, H., Hasida, K., Ishizuka, M. (2007). POLYPHONET: An advanced social network extraction system from the web. Web Semantics: Science. *Services and Agents on the World Wide Web*, *5*(4), 262–278.

Medelyan, O., Frank, E., Witten, I.H. (2009). Human-competitive tagging using automatic keyphrase extraction. In: Proceedings of the 2009 conference on empirical methods in natural language processing, vol. 3, pp. 1318–1327. Association for Computational Linguistics.

Medelyan, O., & Witten, I.H. (2006). Thesaurus based automatic keyphrase indexing. In: Proceedings of the 6th ACM/IEEE-CS joint conference on digital libraries, pp. 296–297. ACM.

Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y. (2017). Deep keyphrase generation. arXiv:1704.06879.

Mihalcea, R., & Tarau, P. (2004). TEXTRANK: Bringing order into text. In: Proceedings of the 2004 conference on empirical methods in natural language processing.

Mihalcea, R., Tarau, P., Figa, E. (2004). PageRank on semantic networks, with application to word sense disambiguation. In: Proceedings of the 20th international conference on computational linguistics, p. 1126. Association for Computational Linguistics.

Moldovan, D., Harabagiu, S., Pasca, M., Mihalcea, R., Girju, R., Goodrum, R., Rus, V. (2000). The structure and performance of an open-domain question answering system. In: Proceedings of the 38th annual meeting on Association for Computational Linguistics, pp. 563–570. Association for Computational Linguistics.

Mori, J., Ishizuka, M., Matsuo, Y. (2007). Extracting keyphrases to represent relations in social networks from web. In: IJCAI, vol. 7, pp. 2820–2827.

Newman, D., Koilada, N., Lau, J.H., Baldwin, T. (2012). Bayesian text segmentation for index term identification and keyphrase extraction. *Proceedings of COLING*, *2012*, 2077–2092.

Nguyen, T.D., & Kan, M.Y. (2007). Keyphrase extraction in scientific publications. In: International conference on asian digital libraries, pp. 317–326. Springer.

Nguyen, T.D., & Luong, M.T. (2010). WINGNUS: Keyphrase extraction utilizing document logical structure. In: Proceedings of the 5th international workshop on semantic evaluation, pp. 166–169. Association for Computational Linguistics.

Osiński, S., Stefanowski, J., Weiss, D. (2004). LINGO: Search results clustering algorithm based on singular value decomposition. In: Intelligent information processing and web mining, pp. 359–368. Springer.

Page, L., Brin, S., Motwani, R., Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web, Stanford InfoLab, Tech. rep.

Sarkar, K. (2013). A hybrid approach to extract keyphrases from medical documents. arXiv:1303.1441.

Smatana, M., & Butka, P. (2016). Extraction of keyphrases from single document based on hierarchical concepts. In: IEE 14th international symposium on applied machine intelligence and informatics (SAMI), pp. 93–98. IEEE.

Song, M., Song, I.Y., Allen, R.B., Obradovic, Z. (2006). Keyphrase extraction-based query expansion in digital libraries. In: Proceedings of the 6th ACM/IEEE-CS joint conference on digital libraries, pp. 202–209. ACM.

Tomokiyo, T., & Hurst, M. (2003). A language model approach to keyphrase extraction. In: Proceedings of the ACL 2003 workshop on multiword expressions: analysis, acquisition and treatment-volume 18, pp. 33–40. Association for Computational Linguistics.

Turney, P.D. (2000). Learning algorithms for keyphrase extraction. *Information Retrieval*, *2*(4), 303–336.

Turney, P.D. (2003). Coherent keyphrase extraction via web mining. arXiv:0308033.

Wan, X., & Xiao, J. (2008). Single document keyphrase extraction using neighborhood knowledge. In: AAAI, vol. 8, pp. 855–860.

Wan, X., Yang, J., Xiao, J. (2007). Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In: Proceedings of the 45th annual meeting of the association of computational linguistics, pp. 552–559.

Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G. (1999). KEA: Practical automatic keyphrase extraction. In: Proceedings of the fourth ACM conference on digital libraries, pp. 254–255. ACM.

Xie, F., Wu, X., Zhu, X. (2017). Efficient sequential pattern mining with wildcards for keyphrase extraction. *Knowledge-Based Systems*, *115*, 27–39.

Yang, S., Lu, W., Yang, D., Li, X., Wu, C., Wei, B. (2017). KEYPHRASEDS: Automatic generation of survey by exploiting keyphrase information. *Neurocomputing*, *224*, 58–70.

Yih, W.T., Goodman, J., Carvalho, V.R. (2006). Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web, WWW '06* (pp. 213–222). New York: ACM. https://doi.org/10.1145/1135777.1135813.

You, W., Fontaine, D., Barthes, J.P. (2009). Automatic keyphrase extraction with a refined candidate set. In: Proceedings of the 2009 IEE/WIC/ACM International joint conference on web intelligence and intelligent agent technology-volume 01, pp. 576–579. IEEE Computer Society.

Zamir, O., & Etzioni, O. (1998). Web document clustering: A feasibility demonstration. In: SIGIR, vol. 98, pp. 46–54. Citeseer.

Zesch, T., & Gurevych, I. (2009). Approximate matching for evaluating keyphrase extraction. In: Proceedings of the international conference ranLP, pp. 484–489.

Zha, H. (2002). Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In: Proceedings of the 25th annual international acm sigir conference on research and development in information retrieval, pp. 113–120. ACM.

Zhang, D., & Dong, Y. (2004). Semantic, hierarchical, online clustering of web search results. In: Asia-Pacific Web Conference, pp. 69–78. Springer.

Zhang, K., Xu, H., Tang, J., Li, J. (2006). Keyword extraction using support vector machine. In: international conference on web-age information management, pp. 85–96. Springer.

Zhang, Q., Wang, Y., Gong, Y., Huang, X. (2016). Keyphrase extraction using deep recurrent neural networks on Twitter. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp. 836–845.

Zhang, Y., Zincir-Heywood, N., Milios, E. (2004). World Wide Web site summarization. *Web intelligence and agent systems: an international journal*, *2*(1), 39–53.