Journal of Big Data

RESEARCH Open Access

Contextual topic discovery using unsupervised keyphrase extraction and hierarchical semantic graph model

Hung Du[1], Srikanth Thudumu[1*], Antonio Giardina[1], Rajesh Vasa[1], Kon Mouzakis[1], Li Jiang[2], John Chisholm[2] and Sanat Bista[2]

*Correspondence: srikanth.thudumu@deakin.edu.au

[1] Applied Artificial Intelligence Institute (A2I2), Deakin University, Geelong, VIC 3216, Australia
[2] Department of Defence, Defence Science and Technology Group (DSTG), Canberra 2609, Australia

Abstract

Recent technological advancements have led to a significant increase in digital documents. A document's key information is generally represented by the keyphrases that provide the abstract description contained therein. With traditional keyphrase techniques, however, it is difficult to identify relevant information based on context. Several studies in the literature have explored graph-based unsupervised keyphrase extraction techniques for automatic keyphrase extraction. However, there is only limited existing work that embeds contextual information for keyphrase extraction. To understand keyphrases, it is essential to grasp both the concept and the context of the document. Hence, a hybrid unsupervised keyphrase extraction technique is presented in this paper called ContextualRank, which embeds contextual information such as sentences and paragraphs that are relevant to keyphrases in the keyphrase extraction process. We propose a hierarchical topic modeling approach for topic discovery based on aggregating the extracted keyphrases from ContextualRank. Based on the evaluation on two short-text datasets and one long-text dataset, ContextualRank obtains remarkable improvements in performance over other baselines in the short-text datasets.

Keywords: Context-awareness, Contextual topic discovery, Hierarchical semantic graph, Keyphrase extraction, Topic modeling

Introduction

Technology advancements have resulted in the rapid growth of digital documents over the last two decades. Research communities and industry professionals are now surrounded by vast knowledge bases that contain millions of documents. While these knowledge bases offer significant benefits to research and industry, it has become increasingly difficult to obtain pertinent information in a specific context. In general, the primary information of a document can be distilled down to a few keyphrases. The keyphrases provide abstract information about the document and can be used to summarise what the document is about. A keyphrase can be categorised into topics derived from the document's high-level concepts. Effective keyphrases and topic annotations facilitate

efficient information retrieval and processing. However, annotating and extracting such keyphrases and topics manually is inefficient and time-consuming. Automating the process of extracting keyphrases and topics is, therefore, an important step toward improving the efficiency. Existing approaches in the literature can be categorised into two groups: supervised and unsupervised techniques [1–3].

The extraction of keyphrases from the corpus of a document is the first step in a keyphrase and topic annotation process. This process consists of two tasks: extracting words from the text and detecting keyphrase candidates. A document's text is first broken down into a set of words that do not contain white space. Using the set of words, a phrase is then constructed by applying the following techniques: N-Gram methods, Part-Of-Speech (POS) based methods or both [4]. An N-Gram method aims to extract N consecutive words, whereas a POS tag of a word such as a noun, a verb, an adjective, and others is used to form a linguistic pattern of a phrase. As the importance of an extracted phrase has not yet been determined, the phrase is referred to as a keyphrase candidate. To estimate such importance, supervised or unsupervised techniques are utilised. This results in a set of top-ranking keyphrase candidates that are defined as keyphrases.

Supervised techniques aim to classify the extracted phrases based on the provided classification semantics [4–6]. Logistic regression, support vector machines, decision trees and fully-connected neural networks are commonly used as binary classifiers. Annotated keyphrases are required for each document in order to train the classifier. However, annotating exemplar keyphrases from the same context as the document, to improve the classifier's performance, requires considerable human effort. Furthermore, it is impractical to process documents with domain-specific keyphrases that are not labelled at the outset. Due to these challenges, existing approaches in the literature have largely been focused on the development of unsupervised techniques.

Unsupervised techniques aim to extract keyphrases by ranking the extracted phrases according to their importance. Methods for estimating the importance of phrases can be divided into two categories: statistical-based methods and graph-based methods. The statistical method assigns a quantitative value to each word in a document, such as its number of occurrences. Sequences of words are then estimated according to their statistical features, such as the co-occurrence of words within the sequence, the occurrences of a sequence, etc [4, 7, 8]. Graph-based methods construct a graph of text in which nodes represent words and edges represent word relationships. PageRank [9, 10] and HITS [11] are commonly used techniques to rank nodes in many existing approaches [10, 12–15]. Those phrases that rank highest are then selected as keyphrases. Existing graph-based approaches often establish the relationship between a pair of phrases based on the conceptual meaning of a word [16, 17], its position [14], or statistical measures, such as its co-occurrences [12, 15, 18]. We hypothesise that contextual information may also be significant in determining the relationship between phrases. Contextual information can be extracted from phrases belonging to sentences or paragraphs. Consider the ***"environment"*** keyword in the excerpt: *"The air quality in the polluted* ***environment*** *can be measured and predicted by a machine learning model. The model is operated in a virtual* ***environment****."* In this example, the meaning of the ***"environment"*** keyword in the first sentence refers to the context of ecology whereas the second sentence refers to the context of computer

science. The distinction between the two uses of the keyword ***"environment"*** is difficult without considering the surrounding information. The most effective way to comprehend the contextual meaning of the keyword ***"environment"*** is to model how the word refers to an aggregate of its surrounding words, conditions, and influencing factors.

Topic modeling is the second step in keyphrase and topic annotation. By grouping keyphrases together, topic models are designed to discover the main ideas within a document's corpus. Latent Dirichlet Allocation (LDA) [19] is one of the most commonly used topic modeling techniques. Using this technique, extracted words or phrases are projected into high-dimensional latent spaces and grouped into clusters representing specific topics. A hierarchical topic model, on the other hand, determines the relationship among topics in the hierarchical order formed by statistical estimation [20, 21]. Hierarchical topic graphs are often used to explore document representations and mine meaningful taxonomies.

A conventional method for evaluating extracted keyphrases and topics is to use the exact matching method. As the name implies, it is based on the principle that two keyphrases or topics are matched if they contain the same type of text. Although exact matching can successfully match keyphrases or topics with similar lexical forms, it is less effective when matching those that have subtle variations in their lexical form. For example, the term ***"neural network"*** cannot be equated with the term ***"neural net"***. Another technique for evaluating extracted keyphrases and topics is the approximate matching method [22]. Using this method, two keyphrases or topics can be partially matched if they include the same words or contain words that overlap. As an example, ***"C programming"*** partially matches ***"programming language"*** since it is its overlapping term. Precision and mean reciprocal rank are two well-known evaluation metrics used for assessing the relevance of extracted keyphrases or topics following the application of matching techniques. Precision measures how many relevant keyphrases or topics are retrieved from a fixed set of results, and mean reciprocal rank measures the quality of those relevant keyphrases or topics. Contributions of our study include the following:

- A technique for extracting keyphrases that consider both conceptual and contextual facets or aspects called ContextualRank. In order to rank phrases, the approach considers the position of the phrases as a biased factor in the TextRank algorithm.
- A hierarchical topic modeling technique that extracts topics from a hierarchical graph derived from extracted keyphrases and their semantic similarity.
- Evaluation of the performance of ContextualRank and the hierarchical topic modeling approach using three different datasets by applying BLEU, one of the evaluation metrics used in machine translation.

The rest of the paper is organised as follows: We summarise related work in "Related work" section. ContextualRank for keyphrase extraction is presented in "ContextualRank" section. Hierarchical topic modeling for topic discovery is covered in "Hierarchical topic modeling" section. The results of this study are discussed in "Experiments and results" section followed by the conclusions and future work in "Conclusion and future work" section.

Related work

Keyphrases and topics are essential elements that represent the key concepts of documents and provide a means of determining their relationship with one another. In order to improve the robustness of retrieving the relevant documents for search queries, extracting accurate and relevant keyphrases and topics is critical. While keyphrases refer to the main concepts of the content, topics encapsulate those concepts to derive the main concept.

In the supervised techniques for automatic keyphrase extraction, the classifier is trained using human-annotated keyphrases. The training process consists of two phases such as feature extraction and classification. Logistic regression, support vector machine, decision trees and fully-connected neural network are the commonly used techniques for the classification in many existing approaches. The feature extraction, on the other hand, is the main focus of each approach in the literature. Hulth et al. [23] integrated the thesaurus as the domain knowledge to Term Frequency-inverse Document Frequency (TFIDF) to create domain specific features of phrases. Hulth [4] also applied linguistic knowledge such as POS to the proposed four feature-based methods by Frank et al. [24] in order to create features of phrases. As a method of automatically extracting keyphrases from documents, Witten et al. [5] employed statistical methods such as the TFIDF and the first occurrence of a phrase in a document. Caragea et al. [25] used the citation network of documents to represent features of phrases in a document.

In the unsupervised techniques for automatic keyphrase extraction, words and their properties such as the position in text, the associated POS tag, etc. are converted into features based on statistical measures such as Term Frequency (TF), TFIDF, BM25 [26], etc. The word features are then concatenated to the phrase features using statistical measures such as N-gram TFIDF [7, 27, 28], language models [8, 17, 29] or other probabilistic models. While existing approaches utilises one feature of phrases in the keyphrase extraction, Campos et al. [30] proposed the YAKE! algorithm that weighs the extracted phrases by aggregating multiple features of phrases within a document. The YAKE! algorithm was demonstrated to outperform the other approaches, though faces the following challenges: (i) the extracted keyphrases are often not in the form of noun phrases, and (ii) the importance of the extracted keyphrases can be biased to their position in text as the early occurred keyphrases tend to be in the top ranks.

It is important to understand that the knowledge of the relationships among phrases are absent from existing statistical models. The graph-based approaches, on the other hand, addresses this by reinforcing the relationship among words or phrases in the estimation for ranking. The common process of a graph-based approach consists of four phases: (i) constructing a graph of words or phrases; (ii) estimating the semantic relationship among words or phrases based on statistical measures; (iii) reinforcing the relationship to weigh each word or phrase; and (iv) ranking phrases based on their reinforced weights. TextRank [10] which is the variation of PageRank [9] on a word graph within a document is the commonly used technique to reinforce the relationship among words or phrases and rank them according to their reinforced weights. Wan and Xiao [31] proposed SingleRank that is the extension of TextRank by estimating the relationship among words based on their co-occurrences in fixed-size windows within a document. Wan and Xiao [32] then proposed ExpandRank that is the extension of SingleRank

by adding the importance of nearest neighbor documents close to a particular document to weigh phrases in that document. One of the challenges of SingleRank and ExpandRank is that reinforced weights of keyphrases significantly depend on the features of documents. This results in the fact that non importance phrases tend to gain the high weights. Bougouin et al. [12] proposed TopicRank which applies the clustering algorithm to create clusters of phrases and uses those clusters as additional features to identify keyphrases of a document. Boundin [15] extended TopicRank by adding multipartite graphs that represent the relationships between words in one cluster and that in another cluster. The technique also attempted to integrate the position of phrases into the weighting scheme of the graph by adjusting weights of the first occurring keyphrases in a document. Florescu and Caragea [14] proposed PositionRank that extends TextRank by incorporating all positions of a word's occurrences into the weighting scheme of the graph algorithm. Our ContextualRank differs from PositionRank by estimating the positions of phrases and their occurrences in the content.

The relationship of keyphrases is not only represented by the homogeneous relationships among words but also by the heterogeneous relationships between keyphrases and their belonging sentences. Wan et al. [33] proposed the iterative reinforcement approach to extract keyphrases by reinforcing the relationship between sentences and words via three types of graphs such as the sentence-to-sentence graph, the word-to-word graph and the sentence-to-word graph. In this approach, the homogeneous relationships between sentences or words are represented by the semantic similarity between their TFIDF vector representations. Bennani-Smires et al. [16], on the other hand, proposed EmbedRank that projects documents and keyphrases to the high-dimensional vector space with the support of existing pre-trained models such as Sent2Vec[1] and Doc2Vec.[2] EmbedRank ranks keyphrases in a document based on the cosine distance between their vector representations and the vector representation of that document. Sun et al. [17] then proposed SIFRank which differs from EmbedRank by applying the pre-trained language model ELMo [34] to capture context-dependent semantic information of words via their vector representations. Similar to PositionRank, SIFRank further incorporates all positions of a word's occurrences in the estimation to handle long documents. Our ContextualRank differs from EmbedRank and SIFRank by two folds. In the first fold, ContextualRank establishes the homogeneous relationships between phrases by utilising their conceptual information. In the second fold, ContextualRank incorporates the homogeneous relationships between sentences or paragraphs into the weighting scheme of the graph algorithm.

The enhancement of contextual information of a keyphrase can be expressed via the global context and the local context [35, 36]. In particular, the global context is the theme of the document containing keyphrases, and the local context refers to sentences or paragraphs where the keyphrases are utilised. Liang et al. [35] proposed an approach that ranks keyphrases by aggregating the global and local information. Specifically, the global context was modelled by estimating the Manhattan distance between documents and keyphrases, while the boundary-aware centrality was employed on top of the position

[1] https://github.com/epfml/sent2vec.

[2] https://github.com/jhlau/doc2vec.

of keyphrases in the particular document to model the local context. On the other hand, Ding & Lou [36] proposed AGRank that applies PageRank [9] on the universal graph bewteen documents, sentences and keyphrases to rank keyphrases. In particular, the document-keyphrase and sentence-keyphrase relationships in the universal graph were utilised to represent the global context and the local context of keyphrases, respectively. These techniques mainly focus on using the contextual facet to model the weighting estimation for ranking keyphrases. However, the conceptual facet of a keyphrase is also important to understand the relationships between keyphrases. Our ContextualRank integrates both conceptual facet and contextual facet into the considerations while estimating weights of keyphrases.

Topic modeling techniques aim to discover topics that represent the main idea in content by grouping the extracted keyphrases together. Latent Dirichlet Allocation (LDA) [19] is the commonly used technique to discover a topic (i.e., a cluster of words) via the high-dimensional latent space among words. The technique, however, does not infer the relationship among topics which is one of the important factor to derive the topic taxonomies for information retrieval and text summarisation. Paisley et al. [20] proposed the nested Hierarchical Dirichlet Process (nHDP) that derives the hierarchical order of topics based on the probability distribution that those topics belong to. Viegas et al. [21], on the other hand, proposed the CluHTM algorithm that is a non-probabilistic Hierarchical Topic model for exploring richer text representation of a topic via both statistical and semantic information of words and stabilising the hierarchical structure. Existing topic modeling approaches return clusters of words as topics, and hence, it is necessary for human to provide the encapsulated text representation of those topics. To address this, Duan et al. [37] proposed TopicNet that incorporates the prior structural knowledge such as WordNet [38] to the hierarchical topic model by measuring the semantic similarity between them in the high-dimensional latent space. Being different from the other approaches, our approach derives topics via the hierarchical graph of the keyphrases obtained from various keyphrase extraction techniques. We hypothesise that the encapsulated text representation of a topic in the content may be a high-level keyphrase in the hierarchical graph.

ContextualRank

ContextualRank is an unsupervised hybrid keyphrase extraction technique that embeds conceptual and contextual aspects of phrases in order to weight those phrases. This is then combined with the TextRank algorithm that uses the position of phrases as a biasing factor to rank the phrase. Word2Vec, Glove [39], Fasttext [40], Bidirectional Encoder Representations from Transformers (BERT) [41], and other embedding techniques are used to transform words or phrases into their numeric representations which are vectors. The vectors generated by these techniques were trained on an extremely large corpus of text, and therefore the vectors are likely to reflect the conceptual meaning of the word or phrase. In order to infer the contextual significance of phrases or sentences, ContextualRank utilises vector representations of the sentences or paragraphs that contain contextual information. Graph-based algorithms, such as PageRank [9], are further employed to estimate the importance of individual nodes within a graph by recursively determining the edges that connect the nodes. The ContextualRank algorithm

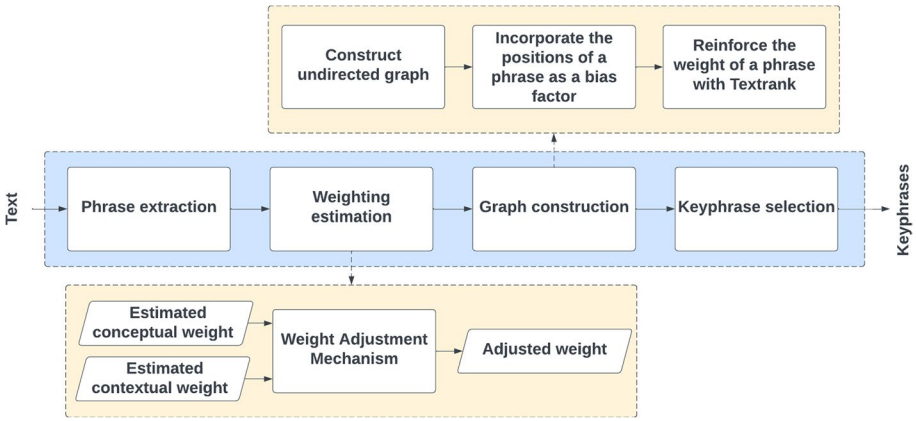

Fig. 1 The workflow for keyphrase extraction using ContextualRank

incorporates the positions of phrases into the biased Textrank algorithm in order to rank phrases by their importance in a particular document.

As shown in Fig. 1, the ContextualRank technique consists of four phases:

(i) extracting phrases from the text; (ii) estimating the importance of phrases; (iii) constructing the undirected graph and using Textrank to rank phrases, and (iv) selecting top phrases as keyphrases. Details of these phases are provided below.

Phrase extraction

In ContextualRank, keyphrase candidates are represented as noun phrases. The given text is first tokenised into words with each having its own POS tag[3]. The text is then purged of words such as *is*, *am*, *are*, *for*, *the*, and others that do not have any inherent meaning. Based on POS tags, the following linguistic pattern is used to extract noun phrases:

$$(N)^* \mid (JJ)^*(N)^+ \tag{1}$$

where 'N' and 'JJ' represent nouns and adjectives, respectively. It is possible to extract noun phrases in two ways, such as (1) one or more nouns (for example, "text summarization") or (2) nouns plus one or more preceding adjectives (for example, "powerful agent").

Weighting estimation

Conceptual facet

A pre-trained BERT model is used to transform the extracted keyphrase candidates into their vector representation [41]. There are several reasons of selecting BERT model against the traditional techniques such as Word2Vec, Glove [39], Fasttext [40] to generate the vector representation of the given text. First, BERT is a language model trained the large dataset that contains 800 million words from BooksCorpus [42] and 2.5 billion words from English Wikipedia. This demonstrates the generality

[3] The nltk library was used to identify POS tags.

of BERT to comprehend and process natural language across diverse tasks and domains. Second, BERT captures the contextual relationship between words and then generates the vector representation that expresses the semantic meaning of the given text. This enables the language contextualisation as a word may have the different meanings based on the context where that word is used. A sequence of words is transformed into a vector of 768 dimensions by the model. As an estimate of semantic similarity between two keyphrase candidates, the cosine similarity between their vector representations is used.

$$sim_{\text{concept}}(k, k') = cos(\vec{v_k}, \vec{v_{k'}}) = \frac{\vec{v_k} \cdot \vec{v_{k'}}}{\|\vec{v_k}\| \cdot \|\vec{v_{k'}}\|} \tag{2}$$

where k and k' are keyphrase candidates, and $\vec{v_k}$ and $\vec{v_{k'}}$ are the vector representation of k and k', respectively. It is important to note that the vector for each keyphrase candidate contains more likely the conceptual significance of the phrase than its context.

Contextual facet

From an intuitive standpoint, it is reasonable to assume that the context of a phrase can be determined by observing its placement within the sentence. Depending on its placement within a sentence or paragraph, a single word may have more than one meaning. Consider the 'model' keyword in the excerpt: "Software engineers typically use a model to represent data. In data science, however, a model is referred to as a learning algorithm." It is important to note that the meaning of the term 'model' in the first sentence differs from that in the second sentence. The context is different between the two sentences. In order to fully understand the meaning of a word, it is important to understand its context. In light of this behaviour, we estimate the degree of similarity between keyphrase candidates based on the sentences they belong to.

The given text is divided into sentences where each sentence contains the contextual information. Based on an average of the embedded vectors of each sentence's constituent words, ContextualRank estimates this contextual information and generates a vector representation of each sentence. For instance, if a sentence consists of n-words, $s = (w_1, \ldots, w_n)$, then $\vec{s} \approx \frac{1}{n}(\vec{w_1} + \ldots + \vec{w_n})$, where $(\vec{w_1}, \ldots, \vec{w_n})$ are the embedded vectors of $(w_1, \ldots, w_n)$.

Following that, the cosine similarity is used to determine the semantic similarity between two sentences, denoted as $sim(s, s')$, and the similarity between keyphrase candidates is estimated as:

$$sim_{\text{context}}(k, k') = \frac{1}{\mid C \mid} \sum_{(i,j) \in C} sim(s_i, s_j) \tag{3}$$

where s_i indicates the ith sentence containing k, s_j indicates the jth sentence containing k', and C consists of pairs of sentences that k and/or k' belong to. For example, if k is in the 1^{st} and the 2^{nd} sentences whereas k' is in the 1^{st} and the 3^{rd} sentences, C will consist of the following pairs: (1, 1), (1, 3), (2, 1) and (2, 3). To avoid double-weighting, ContextualRank ignores one pair in two symmetric pairs (e.g., (1, 2) and (2, 1)).

Weight adjustment mechanism

The semantic similarity of keyphrase candidates can be interpreted through either the conceptual or the contextual facet. When it is only interpreted conceptually, the meaning of the keyphrase candidate may be more general and not applicable to the context in which it is used. On the other hand, if keyphrase candidates are interpreted strictly based on its contextual facet, it is difficult to determine a relationship between them based on their semantics. The adjusted semantic similarity is thus estimated as follows in order to balance the conceptual and contextual facets:

$$sim_{\text{adjusted}}(k, k') = (1 - \lambda) \cdot sim_{\text{concept}}(k, k') + \lambda \cdot sim_{\text{context}}(k, k') \tag{4}$$

where $\lambda \in [0, 1]$ is the dampening factor. The higher value of λ indicates a higher importance for the contextual aspect of keyphrase candidates.

Graph construction and keyphrase selection

We build an undirected graph of keyphrase candidates, denoted as $G = (V, E)$, for each document where V is a set of nodes and the edges E is a subset of $V \times V$. Note that the performance of keyphrase extraction is not significantly affected by the type of graph [10]. Nodes are keyphrase candidates, and an edge between two keyphrase candidates $(k_i, k_j) \in E$ interprets the relationship between those candidates. An edge is weighted according to the adjusted semantic similarity between them, as $sim_{\text{adjusted}}(k_i, k_j)$, where $sim_{\text{adjusted}}(k_i, k_j) = 0$ means there is no relationship between k_i and k_j.

After the graph is constructed, the Textrank algorithm is used to rank keyphrase candidates, and top N keyphrase candidates whose weight is higher than others are selected as keyphrases. As an adaptation from the PageRank algorithm, Textrank [10] recursively ranks text (such as a phrase, a sentence, or a paragraph) based on its weight or probability and the transition between nodes. The weight of each node in the graph is estimated by Textrank as follows:

$$S(k_i) = (1 - \alpha) \cdot \tilde{p}_i + \alpha \cdot \sum_{k_j \in In(k_i)} \frac{w_{j,i} \cdot S(k_j)}{\sum_{k_l \in Out(k_j)} w_{j,l}} \tag{5}$$

where $In(k_i)$ is the set of incoming edges of k_i, $Out(k_j)$ is the set of outgoing edges of k_j, α is a damping factor that controls the convergence of the algorithm per time step, $w_{j,i}$ or $w_{j,l}$ is a weight of an edge, and $\tilde{p}$ is the biased factor that controls the randomness of the transition between nodes. In ContextualRank, $\tilde{p}$ is formulated as:

$$\tilde{p} = \frac{1}{\mid P \mid} \sum_{p \in P} \frac{1}{p} \tag{6}$$

where P is the set of the positions of a keyphrase candidate in the content, and the inverse of the position indicates that a keyphrase candidate in the early position is relatively more important than that in the later position. In contrast to PositionRank [14], $\tilde{p}$ is estimated on the basis of positions of keyphrase candidates and their occurrences in the content. Moreover, by omitting the normalisation, the $\tilde{p}$ value of one keyphrase candidate is not influenced by the value of other keyphrase candidates.

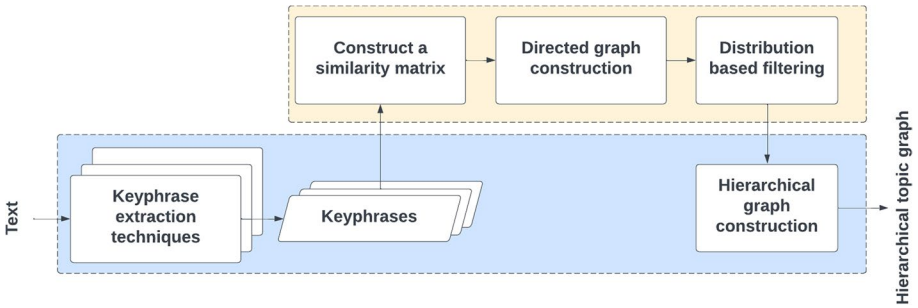

Fig. 2 The workflow for hierarchical topic modeling

Hierarchical topic modeling

Hierarchical topic modeling (HTM) aims to extract topics as keyphrases that are the highest in order in the hierarchical graph. As shown in Fig. 2, it consists of three steps: (1) extracting keyphrases from text as topic candidates; (2) constructing a hierarchical graph; and (3) selecting topics from the graph.

Keyphrase extraction

Keyphrase extraction techniques such as Multipartite Graph [15], PositionRank [14] and ContextualRank are used to extract top-N keyphrases per technique. All of these keyphrases are grouped into one cluster, and those that have the same string of text are excluded. A pre-trained BERT model [41] is then used to convert these keyphrases into vector representations in which each vector is comprised of 768 dimensions. The cosine similarity, as illustrated in Eq. 2, is applied to measure the semantic similarity between keyphrases.

Hierarchical graph construction

A semi-complete directed graph of topic candidates is built, denoted as $H = (V, E)$, where V is a set of nodes and the edges E is a subset of $V \times V$. In particular, a directed edge is represented as an ordered pair of nodes t_i and t_j and denoted by (t_i, t_j) or $t_i \rightarrow t_j$. The weight of a directed edge is estimated according to the cosine similarity between t_i and t_j where $sim_{\text{concept}}(k_i, k_j) = 0$ indicates that there is no relationship between them. The number of directed edges can be exponentially large, and many of them are of no significant importance. A filtering threshold is therefore used to reduce the number of edges. Filtering thresholds are computed based on frequency distributions of weights, rather than being manually dictated. This can be formulated as:

$$\min \left((1-\beta) \cdot \mid E \mid - \sum_{(i,j) \in E} \begin{cases} 1, & \text{if } w_{i,j} \geq \theta \\ 0, & \text{otherwise} \end{cases} \right) \geq 0 \tag{7}$$

where $\mid E \mid$ is a total number of edges, $w_{i,j}$ is a weight of a directed edge $t_i \rightarrow t_j$, $\beta \in [0, 1]$ is the bounded ratio of the number of edges, $\theta \in (0, 1)$ is the filtering threshold. Given the value of β, the value of θ is determined to fulfil Eq. 7. After identifying θ and filtering edges, incoming directed edges of each node that do not have the highest weight are

Table 1 A summary of three datasets used for the evaluation

Type	Dataset	No. Documents	No. Tokens per doc	No. Sections
Short text	Semeval2017	493	176	493
	WWW	675	152	675
Long text	Wiki20	20	4977	251

further removed. The significance of this is that only one root node is selected for each leaf node, and this can be expressed as follows:

$$R(t_i) = \underset{t_j \in In(t_i) \wedge i \neq j}{\operatorname{argmax}} w_{i,j} \tag{8}$$

where $R(t_i)$ indicates a root node of a topic candidate t_i, $In(t_i)$ indicates incoming edges of t_i, $w_{i,j}$ is a weight of a directed edge $t_i \rightarrow t_j$. It is worth noting that self-referencing edges are also removed from Eq. 8. The number of incoming edges and their weights are applied to select an edge for each pair of symmetric edges such as ($t_i \rightarrow t_j$ and $t_j \rightarrow t_i$). Specifically, a node of a pair of symmetric edges can be considered a root node if the weighted sum of its outgoing edges is greater than that of the other node. A node's weighted sum of its outgoing edges is estimated as follows:

$$w_t = \sum_{t' \in Out(t)} w_{t,t'} \tag{9}$$

where t is a topic candidate, $Out(t)$ is a set of outgoing edges of t, t' is another topic candidate that t points to, and $w_{t,t'}$ is the weight of a directed edge $t \rightarrow t'$.

Topic selection

Topic candidates are organised in the separate groups. Each group is a hierarchical graph that contains only one root node at the highest level. Topics are then selected from these root nodes. Furthermore, it is possible that there are topic candidates with no incoming and outgoing edges due to the choice of θ in Eq. 7 and the root selection using Eq. 8. In theory, these topic candidates differ significantly from the other topic candidates, and as a result, they neither have references nor refer to any other topic candidates. As such, they are also considered as topics.

Experiments and results

In this section, we discuss three datasets that were used to evaluate ContextualRank and our hierarchical topic modeling (HTM) approach. We then propose the BLEU-based evaluation metric that measures both the exact match and the approximate match between the extracted keyphrases and the human-annotated ones to evaluate the performance of ContextualRank and HTM. Finally, we analyse the obtained results that compare our approaches with some existing baselines such as PositionRank [14] and Multipartite Graph [15].

Table 2 A summary of three categories of the number of ground-truth keyphrases in three datasets

	Semeval2017	WWW	Wiki20
IN_DOC	17	2	14
PART_IN_DOC	0	2	19
NOT_IN_DOC	0	1	3
Total	17	5	36

Datasets

ContextualRank and HTM were evaluated on three open-source datasets[4] which are classified into two types such as short text and long text, as shown in Table 1. Short-text documents have only one section while long-text documents have several sections. Detailed descriptions of each dataset follow.

- Semeval2017 [43] dataset consists of 493 paragraphs collected from ScienceDirect journal articles covering topics related to Computer Science, Material Sciences, and Physics. Each paragraph contains 17 ground-truth keyphrases and 176 tokens on average. These keyphrases were annotated by undergraduate students studying Computer Science, Material Science, or Physics and validated by experts.
- The WWW [44] dataset was compiled from abstracts found in 1330 papers presented at the World Wide Web (WWW) Conference. There are a few abstracts that do not provide enough detail or provide no information, so these abstracts are annotated as "No contact provided yet" or "An abstract is not available". Having filtered out all missing data, there remain 675 abstracts. Each paper in the WWW collection also consists of the Keyword field where keyphrases were extracted. The average abstract contains five ground-truth keyphrases and 152 tokens.
- The Wiki20 [45] dataset comprises 20 full-text technical reports in the field of computer science. There are 251 sections in each document and sections can be divided into several paragraphs depending on their length. Approximately 30 computer science students worked independently to extract key phrases. The average number of keyphrases in each document is 36, and there are 4977 tokens in each document.

The ground-truth keyphrases for each document are not always present in the document. Thus, they are categorized as follows:

- **IN_DOC** consists of keyphrases that are mentioned in the document.
- **PART_IN_DOC** consists of keyphrases that are partially mentioned in the document. Specifically, words in a keyphrase can be found in the document but not the exact form of that keyphrase. For instance, "programming language" cannot be found in a particular scientific paper related to computer science, but its constituent words such as 'programming' and 'language'.
- **NOT_IN_DOC** consists of keyphrases that are not explicitly mentioned in the document. We hypothesised that this type of keyphrase may be obtained from exter-

[4] https://github.com/LIAAD/KeywordExtractor-Datasets.

nal resources, such as an external corpus of text, or it may be derived from expert knowledge.

Each word in a ground-truth keyphrase is stemmed using the Porter stemming algorithm, which inflects a word into its base form. For instance, 'programming' and 'programmer' are stemmed to 'program'. The exact matching approach is used at the phrase level and the word level to classify stemmed keyphrases into the **IN_DOC** group and the **PART_IN_DOC** group, respectively. The remaining stemmed keyphrases are classified into the **NOT_IN_DOC** group. A summary of these categories of ground-truth keyphrases per dataset is provided in Table 2.

Evaluation metrics

Exact matching is a conventional approach to match the extracted keyphrase as a *candidate* to the ground-truth keyphrase as a *reference*. This approach, however, fails to evaluate the ground-truth keyphrases that belong to the **PART_IN_DOC** group, as discussed in Sect. . Furthermore, Table 2 shows that the number of **PART_IN_DOC** keyphrases are mostly equal or greater than that of **IN_DOC** keyphrases, except for the Semeval2017 dataset. As a result, the BLEU method with modified n-gram precision [46], one of the approximate matching approaches, was applied to evaluate keyphrases in both **IN_DOC** and **PART_IN_DOC** groups. In addition, BLUE was selected over ROUGE [47] which is another approach of approximating string matches because BLEU is more effective in measuring the n-gram overlap than ROUGE. In machine translation, the modified n-gram precision between a *candidate* sentence and a *reference* sentence is estimated as:

$$\begin{aligned} \text{Count}_{\text{Clip}} &= \min\left(\text{Count}(n\text{-gram}), \max_{r \in R} \text{Count}(n\text{-gram}'' \in r)\right) \\ p_n &= \frac{\sum_{c \in C} \sum_{n\text{-gram} \in c} \text{Count}_{\text{Clip}}(n\text{-gram})}{\sum_{c' \in C'} \sum_{n\text{-gram}' \in c'} \text{Count}(n\text{-gram}')} \end{aligned} \tag{10}$$

where p_n is the modified n-gram precision, C and C' are the same set of candidate sentences, r refers to a *reference* keyphrase, and R is a set of reference sentences, n-gram, n-gram$'$ and n-gram$''$ are n-gram phrases in a sentence, the numerator indicates a number of n-gram matches between candidate sentences and reference sentences, and the denominator indicates a number of n-gram phrases in candidate sentences.[5] As an example, "*C programming language is popular*" consists the following bigram phrases: "*C programming*", "*programming languages*", "*language is*" and "*is popular*".

In the context of this evaluation, the modified n-gram precision was applied to measure the approximate match between a *candidate* keyphrase and a *reference* keyphrase. Therefore, Eq. 10 can be simplified as:

$$p_n = \frac{\sum_{n\text{-gram} \in c} \text{Count}_{\text{Clip}}(n\text{-gram})}{\sum_{n\text{-gram}' \in c'} \text{Count}(n\text{-gram}')} \tag{11}$$

[5] The description for $\text{Count}_{\text{Clip}}$ can be found in [46].

where c and c' refer to a *candidate* keyphrase. Due to the dynamic in the number of words per *candidate* keyphrase or *reference* keyphrase, choosing a single value for n introduces the inappropriate quality of a *candidate* keyphrase. For instance, a modified unigram precision of a pair of "machine learning algorithm" and "learning machine algorithm" is 1.0, whereas a modified bigram precision and a modified trigram precision of that pair are both equal to 0.0. To address this issue, an average modified n-gram precision is applied, and the quality of a particular *candidate* keyphrase after matching with all *reference* candidates is, then, estimated as:

$$q_c = \max_{r \in R} \frac{1}{\mid r \mid} \sum_{i=2}^{|r|} p_n \tag{12}$$

where p_n is a modified n-gram precision between c and r, and $|r|$ is a number of words in the *reference* keyphrase. It is worth noting that n is greater than 1 for several reasons. First, a keyphrase contains more than 1 word. Second, there are potential noise while applying unigram estimation. As an example, "***machine learning***" differs from "***visual learning***", but the modified unigram precision between them is equal to 0.5. Precision is used to evaluate the relevancy of a set of *candidate* keyphrases as:

$$P = \frac{1}{\mid C \mid} \sum_{c \in C} \begin{cases} 1, \text{ if } q_c > 0 \\ 0, \text{ otherwise} \end{cases} \tag{13}$$

where $\mid C \mid$ is a number of *candidate* keyphrases. It is worth noting that Precision at N ($P@N$) indicates the precision of N *candiate* keyphrases where $N = \mid C \mid$. The quality of a set of *candidate* keyphrases is further evaluated by using Mean Reciprocal Rank (MRR) as:

$$MRR = \frac{1}{\mid C \mid} \sum_{c \in C} \frac{1}{\text{rank}(c)} \tag{14}$$

The technique computes the reciprocal rank of relevant *candidate* keyphrases such that a particular relevant candidate that appears first obtains the highest rank. Before computing MRR, a set of *candidate* keyphrases is sorted according to the value of q_c. Similarly, $MRR@N$ indicates the MRR of N *candidate* keyphrases.

Results and discussion

The performance of ContextualRank and HTM is compared against two baselines. The first baseline is Multipartite Graph [15] which extracts keyphrases by mutually reinforcing relationship of words in the multipartite graph. The second baseline is PositionRank [14] which leverages the position of the word and its frequency in ranking keyphrases. An open-source software, called pke,[6] was used to compute these baselines [48]. The hyperparameters of Multipartite Graph are derived from [15] and used without any amendments. We fine-tuned the hyperparameters of PositionRank such that an extracted keyphrases should consist at least 2 words and at most 4 words and the

[6] pke is available at https://github.com/boudinfl/pke.

Table 3 The performance of four approaches across three datasets using the average *P*@15

Approach	Semeval2017	WWW	Wiki20
Multipartite Graph	0.332	0.074	0.038
PositionRank	0.46	0.104	0.077
ContextualRank	0.533	0.132	0.077
HTM	0.328	0.076	0.05

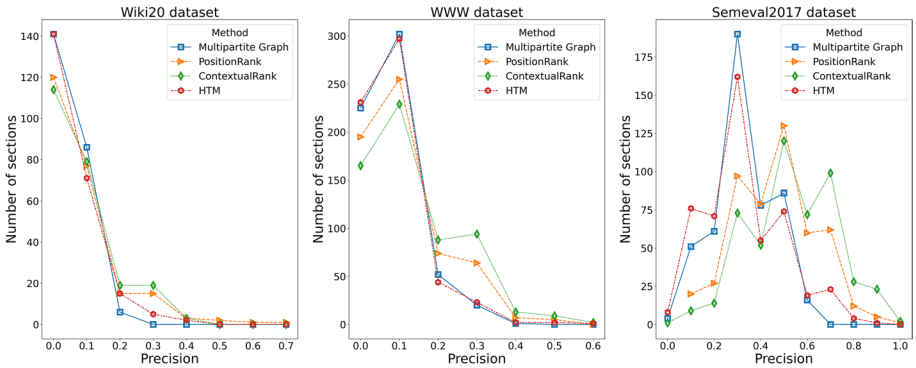

Fig. 3 The performance comparison between approaches such as Multipartie Graph, PositionRank, ContextualRAnk and HTM across three datasets such as Wiki20, WWW, and Semeval2017 (P@15)

window size for estimating all positions of a word's occurrences is 8. In regards to ContextualRank, we identified $\lambda = 0.05$ as the optimal value of λ in Eq. 4 and $\alpha = 0.95$ as the optimal value of α in Eq. 5. This shows the slight effect of the contextual information on the keyphrase extraction process in the context of three datasets in Sect. . In our empirical experiments, we observed the optimal performance of four approaches while selecting top-15 keyphrases, and hence, $N = 15$. These keyphrases were then used in HTM to create the hierarchical graph, and we identified $\beta = 0.7$ as the optimal value of β in Eq. 7.

Precision at 15 (*P*@15) was computed to compare the performance of each approach in extracting relevant keyphrases across three datasets. As demonstrated in Table 3, ContextualRank achieves the best results when extracting relevant keyphrases, followed by PositionRank, HTM and Multipartite Graph. Furthermore, Fig. 3 shows that ContextulRank achieves the high consistency in certain ranges such as 0.2–0.3 in the WWW dataset and 0.6–1.0 in the Semeval2017 dataset. It is worth noting that these datasets consists of short text documents. Meanwhile, the performance of ContextualRank in Wiki20 dataset which contains long text documents is equivalent to that of PositionRank. MRR at 15 (*MRR*@15) was then computed to compare the performance of each approach in extracting high-quality keyphrases across three datasets. Table 4 shows

Table 4 The performance of four approaches across three datasets using the average *MRR*@15

Approach	Semeval2017	WWW	Wiki20
Multipartite graph	0.09	0.025	0.010
PositionRank	0.136	0.04	0.023
ContextualRank	0.136	0.04	0.015
HTM	0.1	0.022	0.011

that the performance of ContextualRank and PositionRank are the same in Semeval2017 and WWW datasets, followed by HTM and Multipartite Graph. PositionRank, however, achieves the considerably high performance in the Wiki20 dataset compared to the other approaches. Overall, ContextualRank achieves the best results in the Semeval2017 and WWW datasets, followed by PositionRank, HTM and Mutipartite Graph. Furthermore, the difference in performance provided in Tables 3 and 4 is significant because it reflects the average performance of four approaches in extracting keyphrases and topics across all documents per dataset. Specifically, 1% increment in the performance of a particular approach indicates 27 and 6 correctly extracted keyphrases more in the WWW dataset and the Wiki20 dataset, respectively.

ContextualRank was further tested at the paragraph-level. Sections were subdivided into paragraphs, which affected the position of phrases in the estimation of ContextualRank. The experiment was conducted using the Wiki20 dataset, whose sections can be divided into paragraphs. *P*@15 and *MRR*@15 were utilised to compare the performance between ContextualRank and ContextualRank (paragraph). The difference between two distributions was calculated to observe the improvement of ContextualRank when being experimented in the paragraph-level scale. Compared with ContextualRank without paragraphs, ContextualRank with paragraphs achieved 0.313% higher in *P*@15 and 10.466% higher in *MRR*@15.

Conclusion and future work

We proposed a novel hybrid unsupervised approach for keyphrase extraction, called ContextualRank, which embeds conceptual and contextual aspects of phrases (in order to weight those phrases) and combines both aspects with the position of phrases as a biasing factor in TextRank to rank phrases. We also proposed the hierarchical topic modeling which leverages the semantic similarity between the extracted phrases from various unsupervised keyphrase extraction techniques including ContextualRank. Our experiments on three datasets including two short-text datasets and one long-text dataset show that ContextualRank achieves better results in both the retrieval of keyphrases and their ranks than our selected baselines in short-text datasets. We further evaluated the effect of the contextual information in the keyphrase extraction by utilising paragraphs as the contextual information in ContextualRank. The experiment showed that the performance of ContextualRank with paragraphs as the contextual information is slightly higher than that without paragraphs.

The overall performance of ContextualRank has potential for improvement through several avenues of future research. First, due to the simplicity and generality of BERT [41], it was utilised in this research work to generate the vector representation of text that supports the weight estimation in ContextualRank. As the other large language models (LLMs) such as T5 [49], GPT-3 [50], PaLM [51], Flan UL2 [52] and Gopher [53, 54] are open access, it would be interesting to utilise these LLMs as alternatives to BERT in ContextualRank. Second, the intermediate layers of these LLMs may contain more linguistic information [55]. The accuracy improvement of the keyphrase extraction was demonstrated by combining such layers of BERT [56]. Therefore, altering the last layer of BERT with the aggregation of its intermediate layers can potentially improve the accuracy of ContextualRank. Third, the empirical analysis in [57, 58] demonstrates that the

dot-product is another potential approach for estimating the text similarity aside from the cosine similarity in ContextualRank.

The integration of ContextualRank into the information retrieval system may provide the relevant information that is conceptually and contextually relevant to a given query as a possible future direction. The investigation of the contextual embedding technique is essential as it will provide the richer contextual information to ContextualRank. The encapsulation of HTM to generate topics from the extracted keyphrases may reduce the performance of HTM as multiple human-annotated keyphrases may refer to a single topic. It is therefore necessary to further evaluate HTM against existing topic modeling approaches on large datasets that suit the context of topic discovery.

Abbreviations

BERT	Bidirectional encoder representations from transformers
nHDP	Hierarchical Dirichlet process
HTM	Hierarchical topic modeling
LDA	Latent Dirichlet allocation
MRR	Mean Reciprocal Rank
POS	Part-Of-Speech
P@N	Precision at N
TF	Term frequency
TFIDF	Term frequency-inverse document frequency
WWW	World Wide Web

Acknowledgements
We would like to thank Sharon Boswell, Paul Lancaster, and Reynalyn Hayes from Defence Science and Technology Group (DSTG), Australia for their support and feedback.

Author Contributions
The authors contributed equally to the research and publication.

Funding
This is part of a collaborative research initiative between the Applied Artificial Intelligence Institute (A^2I^2) at Deakin University and the Defence Science and Technology Group (DSTG), Australia.

Availability of dat and materials
All datasets used in this manuscript are publicly available.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 28 May 2023 Accepted: 26 September 2023
Published online: 12 October 2023

References

1. Hasan KS, Ng V. Automatic keyphrase extraction: a survey of the state of the art. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014; pp. 1262–1273.
2. Papagiannopoulou E, Tsoumakas G. A review of keyphrase extraction. Wiley Interdiscip Rev Data Min Knowl Discov. 2020;10(2):1339.
3. Alami Merrouni Z, Frikh B, Ouhbi B. Automatic keyphrase extraction: a survey and trends. J Intell Inform Syst. 2020;54(2):391–424.
4. Hulth A. Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 2003; pp. 216–223.
5. Witten IH, Paynter GW, Frank E, Gutwin C, Nevill-Manning CG. Kea: Practical automatic keyphrase extraction. In: Proceedings of the Fourth ACM Conference on Digital Libraries. DL '99, pp. 254–255. Association for Computing Machinery, New York, NY, USA 1999. https://doi.org/10.1145/313238.313437.
6. Wu Y-fB, Li Q, Bot RS, Chen X. Domain-specific keyphrase extraction. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, 2005; pp. 283–284.
7. Shirakawa M, Hara T, Nishio S. N-gram idf: A global term weighting scheme based on information distance. In: Proceedings of the 24th International Conference on World Wide Web, 2015; pp. 960–970.

8. Ponte JM, Croft WB. A language modeling approach to information retrieval. In: ACM SIGIR Forum. ACM New York, NY, USA. 2017; vol. 51, pp. 202–208.
9. Page L, Brin S, Motwani R, Winograd T. The pagerank citation ranking: Bringing order to the web. Stanford InfoLab: Technical report; 1999.
10. Mihalcea R, Tarau P. Textrank: Bringing order into text. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004; pp. 404–411.
11. Litvak M, Last M. Graph-based keyword extraction for single-document summarization. In: Coling 2008: Proceedings of the Workshop Multi-source Multilingual Information Extraction and Summarization, 2008; pp. 17–24.
12. Bougouin A, Boudin F, Daille B. Topicrank: graph-based topic ranking for keyphrase extraction. In: International Joint Conference on Natural Language Processing (IJCNLP), 2013; pp. 543–551.
13. Sterckx L, Demeester T, Deleu J, Develder C. Topical word importance for fast keyphrase extraction. In: Proceedings of the 24th International Conference on World Wide Web. 2015; pp. 121–122.
14. Florescu C, Caragea C. Positionrank: an unsupervised approach to keyphrase extraction from scholarly documents. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017; pp. 1105–1115.
15. Boudin F. Unsupervised keyphrase extraction with multipartite graphs. arXiv preprint. 2018; arXiv:1803.08721.
16. Bennani-Smires K, Musat C, Hossmann A, Baeriswyl M, Jaggi M. Simple unsupervised keyphrase extraction using sentence embeddings. arXiv preprint. 2018; arXiv:1801.04470.
17. Sun Y, Qiu H, Zheng Y, Wang Z, Zhang C. Sifrank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model. IEEE Access. 2020;8:10896–906.
18. Danesh S, Sumner T, Martin JH. 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. 2015; pp. 117–126.
19. Blei DM, Ng AY, Jordan MI. Latent dirichlet allocation. J Mach Learn Res. 2003;3(Jan):993–1022.
20. Paisley J, Wang C, Blei DM, Jordan MI. Nested hierarchical dirichlet processes. IEEE Trans Pattern Anal Mach Intell. 2014;37(2):256–70.
21. Viegas F, Cunha W, Gomes C, Pereira A, Rocha L, Goncalves M. Cluhtm-semantic hierarchical topic modeling based on cluwords. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020; pp. 8138–8150.
22. Zesch T, Gurevych I. Approximate matching for evaluating keyphrase extraction. In: Proceedings of the International Conference RANLP-2009. 2009; pp. 484–489.
23. Hulth A, Karlgren J, Jonsson A, Boström H, Asker L. Automatic keyword extraction using domain knowledge. In: International Conference on Intelligent Text Processing and Computational Linguistics. Springer; 2001. pp. 472–482.
24. Frank E, Paynter G, Witten I, Gutwin C, Nevill-Manning C. Domain-specific keyphrase extraction 1999.
25. Caragea C, Bulgarov F, Godea A, Gollapalli SD. Citation-enhanced keyphrase extraction from research papers: a supervised approach. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014; pp. 1435–1446.
26. Robertson SE, Walker S, Beaulieu M, Gatford M, Payne A. Okapi at trec-4. Nist Special Publication Sp; 1996. 73–96.
27. El-Beltagy SR, Rafea A. Kp-miner: participation in semeval-2. In: Proceedings of the 5th International Workshop on Semantic Evaluation. 2010; pp. 190–193.
28. Kang Y-B, Du H, Forkan ARM, Jayaraman PP, Aryani A, Sellis T. Expfinder: an ensemble expert finding model integrating n-gram vector space model and μco-hits. arXiv preprint. 2021. arXiv:2101.06821.
29. Tomokiyo T, Hurst M. A language model approach to keyphrase extraction. In: Proceedings of the ACL 2003 Workshop on multiword expressions: analysis, acquisition and treatment. 2003; pp. 33–40.
30. Campos R, Mangaravite V, Pasquali A, Jorge A, Nunes C, Jatowt A. Yake! keyword extraction from single documents using multiple local features. Inform Sci. 2020;509:257–89.
31. Wan X, Xiao J. Collabrank: towards a collaborative approach to single-document keyphrase extraction. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008). 2008; pp. 969–976.
32. Wan X, Xiao J. Single document keyphrase extraction using neighborhood knowledge. AAAI. 2008;8:855–60.
33. Wan X, Yang J, Xiao J. 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. 2007; pp. 552–559.
34. Sarzynska-Wawer J, Wawer A, Pawlak A, Szymanowska J, Stefaniak I, Jarkiewicz M, Okruszek L. Detecting formal thought disorder by deep contextualized word representations. Psychiatry Res. 2021;304: 114135.
35. Liang X, Wu S, Li M, Li Z. Unsupervised keyphrase extraction by jointly modeling local and global context. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021; pp. 155–164.
36. Ding H, Luo X. Agrank: Augmented graph-based unsupervised keyphrase extraction. In: Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing. 2022; pp. 230–239.
37. Duan Z, Xu Y, Chen B, Wang C, Zhou M, et al. Topicnet: Semantic graph-guided topic discovery. Adv Neural Inform Process Syst. 2021;34:547.
38. Fellbaum C. Wordnet. Dordrecht: Springer, Netherlands; 2010. p. 231–43.
39. Pennington J, Socher R, Manning CD. Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014; pp. 1532–1543.
40. Joulin A, Grave E, Mikolov PBT. Bag of tricks for efficient text classification. EACL. 2017;2017:427.
41. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint. 2018 arXiv:1810.04805.
42. Zhu Y, Kiros R, Zemel R, Salakhutdinov R, Urtasun R, Torralba A, Fidler S. Aligning books and movies: towards story-like visual explanations by watching movies and reading books. In: Proceedings of the IEEE International Conference on Computer Vision. 2015; pp. 19–27.

43. Augenstein I, Das M, Riedel S, Vikraman L, McCallum A. Semeval 2017 task 10: scienceie-extracting keyphrases and relations from scientific publications. arXiv preprint. 2017 arXiv:1704.02853.
44. Gollapalli SD, Caragea C. Extracting keyphrases from research papers using citation networks. In Proceedings of the AAAI Conference on Artificial Intelligence. 2014; 28.
45. Medelyan O, Witten IH, Milne D. Topic indexing with wikipedia. Proceedings of the AAAI WikiAI Workshop. 2008; 1:19–24.
46. Papineni K, Roukos S, Ward T, Zhu W-J. Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2002; pp. 311–318.
47. Lin C-Y. ROUGE: a package for automatic evaluation of summaries. In: Text Summarization Branches Out. Association for Computational Linguistics, Barcelona, Spain. 2004. pp. 74–81. https://aclanthology.org/W04-1013.
48. Boudin F. pke: an open source python-based keyphrase extraction toolkit. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, Osaka, Japan. 2016; pp. 69–73. http://aclweb.org/anthology/C16-2015.
49. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ. Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res. 2020;21(1):5485–551.
50. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, et al. Language models are few-shot learners. Adv Neural Inform Process Syst. 2020;33:1877–901.
51. Chowdhery A, Narang S, Devlin J, Bosma M, Mishra G, Roberts A, Barham P, Chung HW, Sutton C, Gehrmann S, et al. Palm: Scaling language modeling with pathways. arXiv preprint. 2022 arXiv:2204.02311.
52. Tay Y, Dehghani M, Tran VQ, Garcia X, Bahri D, Schuster T, Zheng HS, Houlsby N, Metzler D. Unifying language learning paradigms. arXiv preprint. 2022. arXiv:2205.05131.
53. Rae JW, Borgeaud S, Cai T, Millican K, Hoffmann J, Song F, Aslanides J, Henderson S, Ring R, Young S, et al. Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint. 2021. arXiv:2112.11446.
54. Borgeaud S, Mensch A, Hoffmann J, Cai T, Rutherford E, Millican K, Van Den Driessche GB, Lespiau J-B, Damoc B, Clark A, et al. Improving language models by retrieving from trillions of tokens. In: International Conference on Machine Learning. 2022; 2206–40PMLR.
55. Jawahar G, Sagot B, Seddah D. What does bert learn about the structure of language? In: ACL 2019-57th Annual Meeting of the Association for Computational Linguistics. 2019.
56. Song M, Feng Y, Jing L. Utilizing bert intermediate layers for unsupervised keyphrase extraction. In: Proceedings of the 5th International Conference on Natural Language and Speech Processing (ICNLSP 2022). 2022; pp. 277–281.
57. Zheng H, Lapata M. Sentence centrality revisited for unsupervised summarization. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019; pp. 6236–6247.
58. Zhang Z, Liang X, Zuo Y, Lin C. Improving unsupervised keyphrase extraction by modeling hierarchical multi-granularity features. Inform Process Manag. 2023;60(4): 103356.

Publisher's Note