

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Document Version
Accepted version

Citation for published versión:

Martínez-Cruz, R., Mahata, D., López-López, A.J. *et al.* Enhancing keyphrase extraction from long scientific documents using graph embeddings. *Appl Intell* **55**, 711 (2025). https://doi.org/10.1007/s10489-025-06579-y

Please note that the full-text provided on Comillas' Research Portal is the Author Accepted Manuscript or Post-Print version.

General rights
This manuscript version is made available under the CC-BY-NC-ND 4.0 licence (https://web.upcomillas.es/webcorporativo/RegulacionRepositorioInstitucionalComillas.pdf).

Take down policy
If you believe that this document breaches copyright please contact

Enhancing Keyphrase Extraction from Long Scientific Documents using Graph Embeddings

Roberto Mart´ınez-Cruz[1*†], Debanjan Mahata[2†], Alvaro J. López-López[1] and José Portela[1]

[1*]Institute for Research in Technology, ICAI School of Engineering, Comillas Pontifical University, Madrid, Spain.
[2]Bloomberg, NYC, New York, USA.

*Corresponding author(s). E-mail(s): rmcruz@comillas.edu;
Contributing authors: dmahata@bloomberg.net;
allopez@comillas.edu; jportela@comillas.edu;
†These authors contributed equally to this work.

Abstract

This study explores the integration of graph neural network (GNN) representations with pre-trained language models (PLMs) to enhance keyphrase extraction (KPE) from lengthy documents. We demonstrate that incorporating graph embeddings into PLMs yields richer semantic representations, especially for long texts. Our approach constructs a co-occurrence graph of the document, which we then embed using a graph convolutional network (GCN) trained for edge prediction. This process captures non-sequential relationships and long-distance dependencies, both of which are often crucial in lengthy documents. We introduce a novel *graph-enhanced* sequence tagging architecture that combines PLM-based contextual embeddings with GNN-derived representations. Through evaluations on benchmark datasets, our method outperforms state-of-the-art models, showing notable improvements in F1 scores. Beyond performance on standard benchmarks, this approach also holds promise in domains such as legal, medical, and scientific document processing, where efficient handling of long texts is vital. Our findings underscore the potential for GNNs to complement PLMs, helping address both technical and real-world challenges in KPE for long documents.

Keywords: keyphrase extraction, graph embeddings, long documents, pre-trained language models, natural language processing, deep learning

1 Introduction

Keyphrase extraction (KPE) is a natural language processing (NLP) task that involves identifying the most important phrases in a document that capture its main topics. These keyphrases provide a concise summary of the text and are useful for various downstream applications, such as document classification (Hulth and Megyesi, 2006), clustering (Hammouda et al., 2005), summarization (Qazvinian et al., 2010; Zhang et al., 2004), indexing documents (Gutwin et al., 1999), query expansion (Song et al., 2006), interactive document retrieval (Jones and Staveley, 1999), among others. The task is a form of extreme summarization, also known as keyphrasification (Meng et al., 2022).

Most current algorithms for identifying keyphrases rely on using summarized versions of texts, such as scientific document abstracts, to determine which keyphrases are most important. However, this approach has several drawbacks that limit its effectiveness. Firstly, in real-world scenarios, such as in industry applications, summaries may not always be available. Consequently, the algorithm's performance may decline due to insufficient training on longer texts. Secondly, important extractive keyphrases may be missing from the summaries. Additionally, crucial contextual information in the original text may not be reflected in the summaries, making it impossible to extract certain keyphrases. This limitation can significantly decrease the overall performance of the algorithm.

With the increasing reliance on digitalization and machine learning across diverse industry sectors, such as legal (Chalkidis et al., 2020; Xiao et al., 2023), medical (Lee et al., 2019; Alsentzer et al., 2019), scientific (Beltagy et al., 2019; Yasunaga et al., 2019), financial (Yang et al., 2020; Mienye et al., 2024), and industrial (May et al., 2022; Feng et al., 2023; Ni et al., 2023, 2024; Zhong et al., 2024), the need to process and extract insights from lengthy, multi-page documents has become critical. KPE offers a powerful solution, enabling improved search capabilities, enhanced document understanding, streamlined human annotation, and accelerated knowledge discovery. Developing an effective method for extracting keyphrases from lengthy texts is, therefore, essential. Such a solution would unlock the full potential of KPE, driving significant benefits in real-world scenarios, including higher productivity, improved performance, and better decision-making.

KPE can be approached using unsupervised and supervised methods (Ç ano and Bojar, 2019). Unsupervised techniques typically use ranking algorithms to score phrases in a document based on their information content (Hasan and Ng, 2014). One common approach is graph-based ranking (Mothe et al., 2018), where a graph is constructed with words as nodes and edges representing relationships between these words. The graph is then analyzed using

node centrality measures to determine the importance of each phrase (Boudin, 2013). While graph-based ranking is a popular unsupervised method in this domain, the use of graph embeddings for learning word representations that leverage the relationships between words in text documents remains largely unexplored.

Supervised methods for KPE typically treat it as a token classification task. Tokens are represented numerically using features generated through manual feature engineering, such as term frequency, inverse document frequency, position in the title, and part-of-speech tags (Hasan and Ng, 2014). Alternatively, pre-trained word embeddings (Patel and Caragea, 2019), such as word2vec (Mikolov et al., 2013), GloVe (Global Vectors for Word Representation) (Pennington et al., 2014), or contextualized embeddings (Sahrawat et al., 2020) calculated by a Pretrained Language Model (PLM) such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), can also be used. To optimize the use of these embeddings, a post-processing layer like a Bidirectional Long Short-Term Memory (BiLSTM) (Huang et al., 2015) model is often added to reproject the input embeddings. This step typically reprojects the embeddings into a lower-dimensional representation while considering both preceding and following embeddings in the sequence.

If the embeddings are computed by a PLM, the model can be fine-tuned on the downstream task of KPE using a sequence tagging classification head. While supervised methods often achieve the best results, identifying keyphrases in long documents remains a challenge due to a lack of techniques for modeling long dependencies between words. Such dependencies are crucial for understanding context in longer documents. Therefore, it is important to develop new methods that effectively capture these long dependencies, thereby improving KPE performance in longer texts.

The current state-of-the-art results in KPE have been achieved using either a fine-tuned PLM (Kulkarni et al., 2022) or its raw embeddings as input for a trained BiLSTM-based model (Sahrawat et al., 2020; Park and Caragea, 2020). Both approaches rely on contextualized representations of PLMs, which have the drawback of being limited to a maximum number of words. While this limitation is less significant when the text is short enough to be fully processed by the PLM (e.g., scientific abstracts), it becomes increasingly problematic as the text lengthens.

To overcome the limitations of contextualized representations in KPE, we propose the use of graph embeddings applied to a co-occurrence graph. In this approach, each node represents a word, and the edges represent co-occurrence relationships between words within a sliding window. We use a Graph Convolutional Network (GCN) model (Kipf and Welling, 2017), trained with the self-supervised task of link prediction, to embed each node. Previous approaches have attempted to utilize graph embeddings to improve KPE performance (Santosh et al., 2020). However, none have explored the

potential of using graph embeddings as a solution to address KPE in long documents, and none have leveraged the learned representations of a GCN using a self-supervised approach for this purpose.

Unlike supervised methods, our approach does not require annotated labels for training, which can be expensive and time-consuming to obtain. Instead, it leverages the natural co-occurrence patterns in the text to learn meaningful representations of words. This approach has previously achieved state-of-the-art results in downstream tasks, such as node similarity, demonstrating the effectiveness of GCN-based models for graph representation learning.

By applying this approach to KPE, we aim to capture more comprehensive information about the document's content, including long-term dependencies between words, and enhance the performance of the model on longer texts. In our proposed model, as described in Section 3, we supplement the representations from a PLM with the representations from the learned graph embedding. With our proposed methodology, we hypothesize the following advantages:

- By representing the text as a co-occurrence graph and applying graph embeddings, we can capture relationships between words in a non-sequential manner. This approach has the advantage of placing distant but related words closer together in the semantic space if they have co-occurred or are related through co-occurring words. Unlike sequential models that process words in a fixed order, graph embeddings can capture more complex relationships between words in the context of the input document and reflect the underlying semantic structure of the text. This makes them a more suitable representation for embedding long-term relationships between words, which are often crucial for understanding the context of longer texts.
- The graph-based representation captures all relationships in the text without being constrained by the input word limit. As a result, the graph embedding provides a semantic understanding of a word's usage throughout the entire text, eliminating any length restrictions that may exist. This enables the numerical representation of words to be extended to encompass the entirety of the text.

In this work, we propose a *graph-enhanced sequence tagging* approach (Section 3.4) to address the limitations of existing PLM-based methods for keyphrase extraction on longer texts by augmenting them with graph embeddings. By leveraging the non-sequential nature of graph representations, we aim to capture more semantically rich information about the document's content and enhance the performance of the model on longer texts.

The major challenges and research gaps addressed in this work are:

- Limited exploration of graph embeddings: Current graph-based methods for KPE are predominantly unsupervised and do not leverage advanced graph representation learning techniques, such as graph embeddings, for capturing relationships between words.

- Challenges in modeling long-term dependencies: Existing supervised methods, particularly those relying on PLMs, often struggle to embed long-term dependencies due to the word limit constraints of PLMs.
- Lack of unified frameworks: No prior work has combined the strengths of PLMs and graph embeddings into a unified framework specifically designed to address the challenges of keyphrase extraction in long and complex documents.

The major contributions of the work as proposed in this paper are:

- We introduce a novel framework (Section 3) that combines graph embeddings and PLMs for keyphrase extraction on longer texts. To the best of our knowledge, this is the first study that attempts to do so in the context of long documents.
- We demonstrate the effectiveness of our proposed approach on five popular benchmark datasets (Section 4) and show that it outperforms state-of-the-art methods based on PLMs.
- We provide a comprehensive analysis of the results and highlight the strengths and limitations of our approach, as well as opportunities for future research.

Next, we provide a brief review of the related work in the field (Section 2), focusing on the topic of our paper. We then present our proposed methodology in detail (Section 3), followed by a description of the experiments we conducted (Section 4) and the results we obtained (Section 5). In addition, we discuss the limitations of our approach and suggest future directions for research in this area (Section 7).

2 Related Work

2.1 Keyphrase Extraction

There are two main approaches for KPE: supervised and unsupervised. Unsupervised methods typically rely on graph-based approaches, while supervised methods do not commonly use them. The unsupervised approach involves several phases, starting with the extraction of candidate words or phrases from the text using heuristics such as Part of Speech (POS) patterns for words or n-grams. The next step is to score and rank the extracted phrases according to their importance. Graph-based ranking techniques are popular in this last phase, where centrality measures are applied to the co-occurrence graph of the text. These centrality measures are mostly derived from Google's PageRank (Page et al., 1999) algorithm and are combined with other relevant measurements based on clustering, tf-idf scores, extraction of specific lexical patterns, and word co-occurrence measures. Examples of this methodology include TextRank (Mihalcea and Tarau, 2004) and TopicRank (Bougouin et al., 2013). More recent methods have incorporated word embeddings to improve the accuracy of the ranking phase (Wang et al., 2014; Mahata et al., 018a,b;

Bennani-Smires et al., 2018). Unsupervised methods have proven to be effective when working with unannotated data. However, when labeled data is available, supervised methods tend to perform better and can be tailored to the specific linguistic and contextual characteristics of the topic.

Initially, supervised approaches for KPE relied on manually-engineered features extracted from the text, such as term frequencies (Hulth, 2003), syntactic properties (Kim and Kan, 2009), or location information (Nguyen and Kan, 2007), to represent each word in the sequence. These features were then fed into a classifier to independently classify each candidate phrase as a keyphrase or not. However, it wasn't until the work of (Gollapalli et al., 2017) that KPE was approached as a sequence labeling task, using a conditional random field (CRF) to take advantage of the sequential nature of the data. In an improved version of the sequence labeling approach, (Alzaidy et al., 2019) replaced manual feature engineering with pre-trained GloVe embeddings to represent each word in the sequence. They used a bidirectional long short-term memory (BiLSTM) CRF layer to exploit this representation. The BiLSTM layer updated the representation of the input sequence of words with contextual information, while the CRF modeled the dependencies between the sequence of classifications produced by the BiLSTM outputs.

The transformer architecture (Vaswani et al., 2017) has greatly improved results for various natural language processing tasks, including keyphrase extraction in works such as TransKP (Rungta et al., 2020) and TNT-KID (Martinc et al., 2021). It is effective at embedding words in a sequence, with representations that depend not only on the word itself but also on its context. This has led to the development of PLMs that specialize in providing contextualized embeddings of words in a sentence. In combination with a trained BiLSTM-CRF layer, these embeddings have outperformed all previous models, as demonstrated in the work of (Sahrawat et al., 2020). Some works, such as (Kulkarni et al., 2022), have tried to further enhance the representation of these embeddings by designing self-supervised objectives specifically for keyphrase extraction tasks, resulting in KBIR[1] (Keyphrase Boundary Infilling and Replacement), a PLM that achieved state-of-the-art results.

The sequence labeling approach has a limitation in that it considers only a limited number of words as context. This limitation can result in poor performance on longer texts where the number of words exceeds the maximum input limit. Prior works such as Phrasephormer (Nikzad-Khasmakhi et al., 2021, Preprint) and SaSAKE (Santosh et al., 2020) have integrated graph embeddings into their architecture, however, none have leveraged them to enhance keyphrase identification in long documents. SaSAKE did not utilize any self-supervised learned representation of the graph, while Phrasephormer did not incorporate Graph Neural Networks (GNNs) and instead relied on older graph embedding methods that were found to be less effective than GNN architectures in generating rich representations.

[1] https://huggingface.co/bloomberg/KBIR

It is worth noting that other solutions in the field of Keyphrase Generation (KPG), that addresses the problem as a text generation task to simultaneously extract and abstract keyphrases, have incorporated GNNs to tackle the task, as seen in (Sun et al., 2019) and (Ye et al., 2021). However, these solutions differ from our approach as they focus on text generation instead of sequence tagging and do not utilize self-supervised learned representations of the graph.

2.2 Graph Embeddings

Graphs are widely employed to represent complex relational data across various domains, including NLP, bioinformatics, recommendation systems, and industrial applications. In NLP, constructing co-occurrence graphs from text sequences is a prevalent approach for tasks such as keyword extraction and text summarization (Mihalcea and Tarau, 2004; Rousseau and Vazirgiannis, 2015). These graphs represent words as nodes and their co-occurrences within a sliding window as edges, capturing the syntactic and semantic relationships between words.

Early methods for graph representation learning leveraged pre-trained word embeddings like word2vec (Mikolov et al., 2013) and employed random walks to predict the next node given the current node's one-hot encoding. Techniques such as DeepWalk (Perozzi et al., 2014) and Node2Vec (Grover and Leskovec, 2016) adopted this approach but did not utilize node and edge features or accommodate new nodes after training. To address these limitations, GCNs (Kipf and Welling, 2017) were developed, integrating node and edge features along with information from neighboring nodes into unified representations. Variations like GraphSAGE (Hamilton et al., 2017) and Graph Attention Networks (GAT) (Veli˘ckovi´c et al., 2018) further enhanced performance by introducing sampling strategies and attention mechanisms, respectively.

Our research draws inspiration from the concept of link prediction, as explored in GraphSAGE and other studies leveraging this task to generate low-dimensional node representations. Similar graph construction techniques have been widely used to model complex relationships across various domains. In bioinformatics, graphs represent protein-protein interaction networks, aiding the understanding of biological processes (Stark et al., 2006). In recommendation systems, bipartite graphs model user-item interactions, enhancing recommendation accuracy through graph neural networks (Ying et al., 2018). For fault diagnosis, Chen et al. (2023) introduced the Synergistic Similarity Graph Construction (SSGC) method to model sensor relationships for steel plate fault detection with GATs, while Feng et al. (2023) demonstrated the effectiveness of digital twin-enabled domain adversarial graph networks for bearing fault diagnosis. Our approach adapts these ideas to the NLP domain, focusing on keyphrase extraction from long scientific documents.

Building upon these advancements, our research focuses on constructing co-occurrence graphs from textual data and employing GCNs for node embedding. By transforming text sequences into graphs that encapsulate the relationships between words, we aim to leverage the strengths of graph-based methods across

different domains. Moreover, our work is inspired by the concept of link prediction in graphs, as explored in previous studies (Wang et al., 2021; Saxena et al., 2021). We propose a self-supervised method that utilizes link prediction to train a GCN, effectively embedding the nodes in the graph. By integrating co-occurrence information and link prediction, we aim to capture both local and global structural patterns within the text, enhancing the performance of keyphrase extraction.

2.3 NLP and KPE from Long Documents

Dealing with long sequences of text can significantly increase the complexity of any NLP task, not only due to the added length, but also because long-term dependencies are difficult to embed effectively. Most PLMs have limitations on the maximum number of words they can handle, restricting the context. Additionally, the transformer's multi-attention architecture presents challenges when dealing with a large number of words. Two PLMs have emerged as promising solutions that address these issues by incorporating novel variations in their modeling: Big Bird (Zaheer et al., 2020) and Longformer (Beltagy et al., 2020).

Big Bird, introduced by Google, is specifically designed for processing long documents and employs a sparse attention mechanism that allows the model to attend to a subset of tokens at each layer, reducing the computational cost. It also employs a global attention mechanism to attend to the entire document while maintaining sparsity. The original model can handle up to 2,048 input tokens, while a variation called BigBird-PEGASUS can handle up to 4,096. Longformer, a transformer-based language model designed to process long documents of up to 4,096 tokens, employs a "sliding window attention" mechanism that allows it to attend to tokens across the entire sequence while maintaining a quadratic computational cost. These two models have shown promising results on various NLP tasks that involve processing long documents.

Numerous task-specific solutions have been proposed to tackle the challenge of processing long documents. For instance, in text classification, the authors of (Park et al., 2022) use transformers to efficiently process lengthy documents. In document understanding, the work by (Pham et al., 2022) improves attention mechanisms for long texts. Similarly, (Grail et al., 2021) extends BERT-based architectures for summarizing long documents. Finally, (Zhang et al., 2020) utilizes a combination of a Transformer and a multi-channel LSTM to handle long texts without requiring re-pretraining. In document matching, (Yang et al., 2020) employs a multi-depth transformer-based encoder to handle long-form document matching tasks.

Although KPE is a crucial task in natural language processing, only a few works have attempted to address the challenge of extracting keyphrases from lengthy documents. One example of such work is (Mahata et al., 2022, Preprint), which released two large datasets containing fully extracted text and metadata. This study reported on the performance of various unsupervised and supervised algorithms for keyphrase extraction. Another noteworthy example

is (Doˇcekal and Smrˇz, 2022), which proposes a system that chunks documents while maintaining a global context as a query for relevant keyphrase extraction. To estimate the probability of a given text span forming a keyphrase, the system employs a pre-trained BERT model. Interestingly, the results demonstrate that a shorter context with a query outperforms a longer context without a query. To enhance the generation of keyphrases, (Garg et al., 2021) investigated the inclusion of information beyond the title and abstract as input in the field of KPG. The observed improvement in results through their approach demonstrates that the model must not solely rely on the summary provided by the title and abstract to predict high-quality keyphrases. These studies shed light on the importance of developing effective methods for keyphrase extraction from lengthy documents and offer promising directions for future research.

Our approach is centered on providing the model with a comprehensive representation of the long-term dependencies between words, an aspect that has received limited attention in prior research. We posit that this information can be effectively incorporated through the graph embedding, enabling the model to overcome its contextual limitations and leverage the embedding to expand its understanding of the input document. To the best of our knowledge, no other study has harnessed graph embeddings for this purpose, and we believe our approach has the potential to significantly enrich the state-of-the-art for KPE from lengthy documents.

3 Methods

In this section, we formulate KPE as a sequence tagging task and provide details on the architecture used to embed the nodes of the co-occurrence graph. We also describe the calculation of contextual embeddings of the words and explain how both representations are utilized by a sequence tagging model to identify keyphrases in a given text.

3.1 Problem Formulation

The task is to classify each word w_t in a given input sequence of words $d = \{w_1, w_2, ..., w_n\}$ into one of three possible classes: $Y = K_B, K_I, O$. In this context, K_B represents the beginning word of a keyphrase, K_I indicates that a word is inside a keyphrase, and O denotes that w_t is not part of a keyphrase. This process is commonly referred to as Begin-Inside-Outside (BIO) tagging scheme in the sequence tagging literature.

BIO tagging is particularly well-suited for KPE as it effectively identifies contiguous spans of words, provides token-level granularity for phrases of varying lengths, and aligns with established practices in sequence tagging tasks (Sahrawat et al., 2020; Kulkarni et al., 2022).

3.2 Graph Embedding

In this section, we provide a detailed description of the process of converting a document into a co-occurrence graph and how a GCN is trained for self-supervised edge prediction to obtain node embeddings from the words in the document.

3.2.1 Co-occurrence Graph

To transform the input sequence of words $d = \{w_1, w_2, \ldots, w_n\}$ into an undirected co-occurrence graph $G = (V, E)$, we follow a process commonly used in graph-based NLP methods (Mihalcea and Tarau, 2004; Rousseau and Vazirgiannis, 2015). Each unique word in d is represented as a vertex $v \in V$. We define edges $e \in E$ between words based on their co-occurrence within a sliding window of size s_d. Specifically, for each position i in the text, we consider the window $[w_i, w_{i+1}, \ldots, w_{i+s_d-1}]$ and connect each word within this window to every other word, capturing their local contextual relationships.

The weight of each edge e_{uv} between words u and v is determined by the number of times they co-occur within these sliding windows throughout the document. Formally, the edge weight is given by Equation 1.

$$w_{uv} = \sum_{i=1}^{n-s_d+1} \delta(u, v, i) \tag{1}$$

where $\delta(u, v, i) = 1$ if both u and v appear in the window starting at position i, and 0 otherwise. This weighted graph captures the frequency and strength of associations between words, enabling the graph neural network to leverage both the structural and statistical properties of the text.

We do not remove common words or apply additional preprocessing during graph construction. This decision aligns with our approach to fine-tuning the PLM, where we also retain all tokens to ensure that stopwords, which often encode grammatical relationships, are represented in the contextual embeddings (Devlin et al., 2019; Beltagy et al., 2019). Furthermore, common words are assigned graph-based embeddings alongside contextual embeddings from the PLM. These representations are concatenated during sequence tagging (see Section 3.4), allowing the model to discern the importance of connections formed by such words.

To mitigate the risk of spurious connections from high-frequency terms, we adopt a weighted edge representation. An example of this graph construction process using a sliding window size of 2 is illustrated in Figure 1.

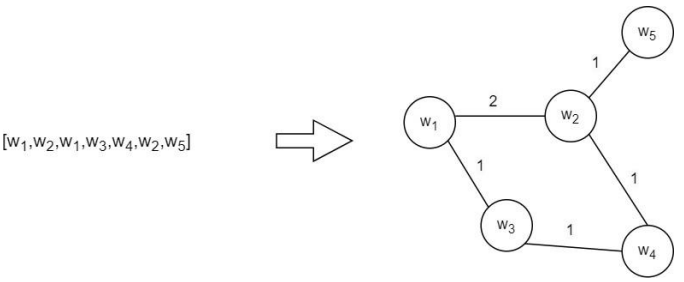


Fig. 1: Co-occurrence graph construction with a sliding window size (s_d) = 2

3.2.2 Graph Convolutional Network

In this section, we present the forward propagation algorithm, as shown in Algorithm 1, which is utilized to embed the nodes in the graph $G = (V, E)$, assuming that the model parameters have already been trained. Specifically, we assume that the set of weight matrices $W_k, \forall k \in \{1, ..., K\}$ is fixed, and these matrices are employed to propagate information between different layers of the model.

Algorithm 1 GCN embedding generation algorithm

Input : Graph $G(V, E)$; input features $\{x_v, \forall v \in V\}$; edge weights $\{ew_{u-v}, \forall u, v \in V\}$; depth K; embedding layer *EMBED*, weight matrices $W_k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $AGGREGATE_k, \forall k \in \{1, ..., K\}$; neighborhood function $N : v \rightarrow 2^V$
Output : Vector representations z_v for all $v \in V$

$h_v^0 \leftarrow x_v, \forall v \in V$
for $k \leftarrow 1$ **to** K **do**
  for $v \in V$ **do**
    $h_{N(v)}^k \leftarrow \{EMBED(u), \forall u \in N(v)\}$
    $h_{N(v)}^k \leftarrow AGGREGATE_k(\{ew_{u-v} * h_u^{k-1}, \forall u \in N(v)\})$
    $h_v^k \leftarrow \sigma(W_k \cdot \text{CONCAT}(h_v^{k-1}, h_{N(v)}^k))$
  end for
  $h_v^k \leftarrow \frac{h_v^k}{\sqrt{k\|h_v^k\|^2}}, \forall v \in V$
end for
$z_v \leftarrow h_v^K, \forall v \in V$

In the scope of this work, we consider all edges without sampling, and the edge weight e_w for nodes that do not have a connection is set to 0. After applying Algorithm 1 to all the nodes, the resulting matrix $Z = \{z_1, z_2, \ldots, z_n\}$ represents the d_g-dimensional embedding of the n^{th} node. A schematic example of the process is illustrated in Figure 2.

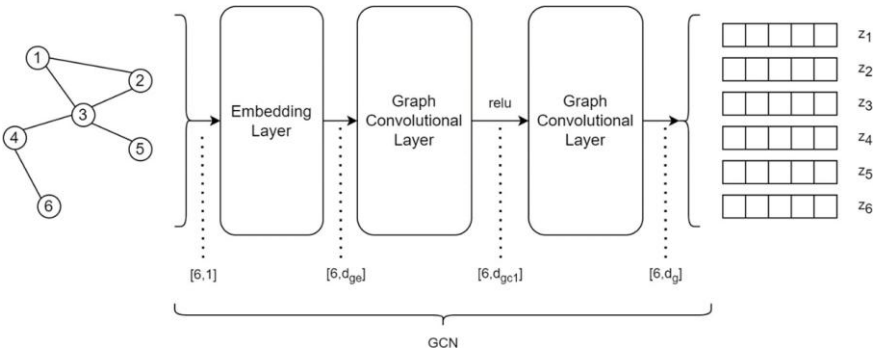


Fig. 2: A schematic example of node embedding using GCN.

3.2.3 Self-supervised Edge Prediction

Utilizing the node representation provided by the GCN, we calculate $E_{p_{ij}}$, which denotes the probability of the i^{th} and j^{th} nodes being connected. To construct the training set for this task, we utilize negative sampling of the existing edges in G. Specifically, for each existing link in the network labeled as 1, we generate n_{ns} non-existent edges and label them as 0.

The training data comprises two sets: $X_g = \{e_1, e_2, \ldots, e_t\}$ and $Y_g = \{y_1, y_2, \ldots, y_t\}$. Each element e_t in X_g represents the t^{th} edge in the dataset and is composed of two nodes, n_{t_1} and n_{t_2}, from the graph. The corresponding element y_t in Y_g is a boolean value that indicates whether the nodes forming e_t are connected.

To estimate the probability $E_{p_{i,j}}$ that nodes i and j are connected, the dot product of their respective embeddings z_i and z_j is calculated. To compute the probabilities for all possible node pairs, a probability matrix E_p of size $n_{nodes} \times n_{nodes}$ is constructed, where each entry (i, j) represents the probability that nodes i and j are connected. E_p is obtained by computing the matrix multiplication of Z with its transpose Z^T, as outlined in Algorithm 1. Figure 3 provides a schematic example of this calculation.

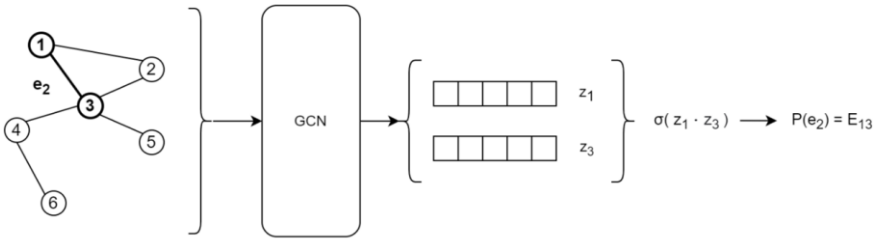


Fig. 3: GCN edge prediction training - Schematic example

To train the network using gradient descent, the binary cross-entropy loss function is utilized, as shown in Equation 2. Here, n_g represents the length of the training dataset, and σ denotes the sigmoid function.

$$L_g = -\frac{1}{n_g} \sum_{i=1}^{x_g} (y_i \cdot \log(\sigma(z_{n_{i1}} \cdot z_{n_{i2}})) + (1 - y_i) \cdot \log(1 - \sigma(z_{n_{i1}} \cdot z_{n_{i2}}))) \quad (2)$$

3.3 Contextual Embedding

This component is responsible for transforming an input sequence of words $d = (w_1, w_2, \ldots, w_n)$ into a sequence of contextual embeddings $H = (h_1, h_2, \ldots, h_n)$ using a PLM. Each $h_n \in \mathrm{R}^{d_c}$ represents the d_c-dimensional embedding of the nth word, encapsulating both semantic and syntactic information from the surrounding context.

Embedding Workflow with PLMs:

KPE is framed as a sequence tagging problem, where specialized PLMs such as BERT or LongFormer are employed to generate contextualized word embeddings. These models leverage transformer encoder architectures and are pre-trained on large text corpora using self-supervised objectives, enabling them to capture nuanced language features.

The input sequence d is first tokenized by the PLM (e.g., BERT's Word-Piece or Byte-Pair Encoding) into smaller sub-word units to handle rare or out-of-vocabulary words effectively. For instance, the word *"scientific"* may be split into sub-words *"sci," "ent," "ific"*, while *"documents"* may be tokenized as *"doc," "um," "ents"*. Each sub-word is processed by the PLM's transformer encoder, generating contextual embeddings $t_{11}, t_{12}, \ldots, t_{1m}$ for the sub-words of w_1, and so on for $w_2, \ldots, w_n$.

Token Mean Pooling for Word-level Embeddings:

To reconstruct word-level embeddings from sub-word embeddings, we employ token mean pooling. For the nth word, its embedding h_n is computed as the average of the embeddings of its constituent sub-words, as shown in Equation 3, where m is the number of sub-words in the tokenized form of w_n and $t_{nj} \in \mathrm{R}^{d_c}$ is the embedding of the jth sub-word of w_n.

$$h_n = \frac{1}{m} \sum_{j=1}^{m} t_{nj}, \quad (3)$$

This method ensures that information from all sub-words contributes equally to the final word-level embedding, mitigating the risk of semantic loss caused by sub-word segmentation. Unlike alternatives such as selecting the first or last sub-word embedding, mean pooling aggregates contextual information from all parts of the word. This is particularly important in PLMs, where meanings are often distributed across sub-words (Devlin et al., 2019; Miaschi

and Dell'Orletta, 2020). Token mean pooling aligns with the principles of contextual representation in PLMs, ensuring cohesive word-level representations. Figure 4 provides a schematic illustration of this process.

Illustrative Example:
Consider the sequence of words d = ("scientific," "documents"). The PLM tokenizer splits *"scientific"* into sub-words *"sci," "ent," "ific"*, and *"documents"* into *"doc," "um," "ents"*. The PLM processes each sub-word, generating contextual embeddings t_{11}, t_{12}, t_{13} for *"scientific"* and t_{21}, t_{22}, t_{23} for *"documents"*. These sub-word embeddings are then aggregated using mean pooling to form the word-level embeddings h_1 and h_2. Figure 4 demonstrates this workflow.

Significance of Contextual Embeddings in Keyphrase Extraction:
The sequence of word embeddings $H = (h_1, h_2, \ldots, h_n)$ forms the input to the sequence tagging module, where each embedding h_n is classified as part of a keyphrase or non-keyphrase. The use of contextual embeddings allows the system to capture both the local and global semantic relationships in the text, which is critical for distinguishing keyphrases from non-keyphrases in scientific documents.

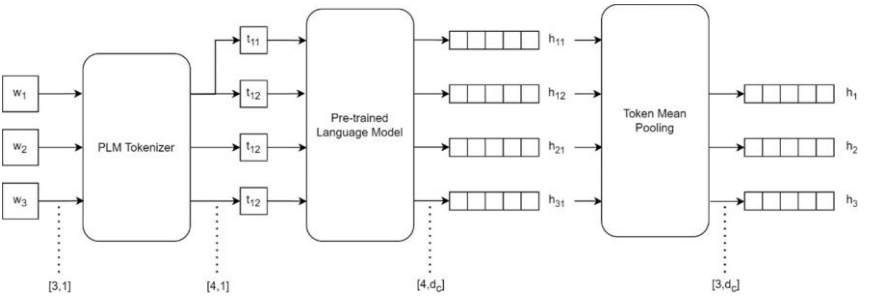


Fig. 4: Schematic diagram for calculating contextual embeddings.

.

3.4 Graph Enhanced Sequence Tagger

After undergoing the aforementioned processes, the input words w are transformed into two sets of embeddings: a sequence of graph embeddings z and a sequence of contextual embeddings h. To further enhance their fusion, these embeddings are individually passed through separate fully connected layers

before being concatenated into a unified numerical representation for each word. This resulting sequence of embeddings is then fed to a fully connected layer followed by a softmax function to obtain a sequence of probabilities for each BIO tag. This architecture is utilized for fine-tuning the PLM. For a visual representation of the process, please refer to the schematic example in Figure 5.

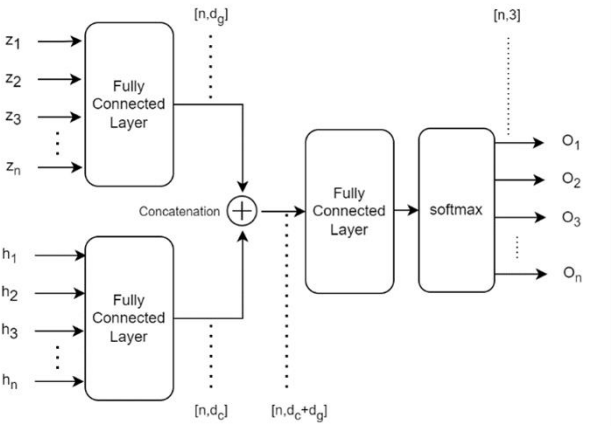


Fig. 5: A schematic diagram of the graph-enhanced sequence tagger.

In cases where the input sequence is larger than the PLM's maximum allowable number of words, a sliding window approach is employed. The window size is set to the upper limit of words for the PLM, which is also used as the stride. This enables the model to process the entire sequence while staying within the constraints of the PLM's input size limitations. In the rest of the paper we refer to this sequence tagging model as the *Graph Enhanced Sequence Tagger*.

4 Experiments

In this section, we describe the experimental settings and report the performance of our proposed model on a series of experiments conducted to investigate its effectiveness on multiple benchmark datasets for KPE. We also discuss the implications of our findings and draw conclusions based on the observations from the obtained results.

4.1 Datasets

We evaluated our approach using four publicly available datasets consisting of full-length long documents from two domains: scientific and news. Specifically, we used the SemEval-2010 (Kim et al., 2010), LDKP3K (Mahata et al., 2022, Preprint), NUS (Nguyen and Kan, 2007), and DUC-2001 (Wan and Xiao, 2008) datasets. To assess the feasibility and effectiveness of our method on short abstracts, we also report performance on the Inspec dataset (Hulth, 2003). As discussed in Section 3, we formulate KPE as a sequence tagging problem and focus on extractive keyphrases that are present in the text. We label each word in the samples using the B-I-O tagging scheme. Any text input longer than 512 tokens/words is considered as a long document. Brief descriptions of each dataset are provided below.

Dataset	Size	Long Doc	Domain	Avg. #Words	Avg. Extractive Keyphrases	Avg. Nodes	Avg. Edges
Inspec	2k	No	Scientific	130.57	6.33	38.02	314.27
DUC-2001	0.308k	Yes	News	740	7.14	60.44	691.25
SemEval-2010	0.24k	Yes	Scientific	7,434.52	9.49	98.70	2,058.61
LDKP3K	26.752k	Yes	Scientific	6,027.10	4.14	86.21	646.43
NUS	0.21k	Yes	Scientific	7,644.43	8.0	77.34	1,558.64

Table 1: Statistics of the datasets used in our experiments, including extractive keyphrase analysis and average graph statistics.

1. The **SemEval-2010** dataset[2] is composed of 284 full-length ACM articles, which have been divided into train, trial, and test sets, containing 144, 40, and 100 articles, respectively.
2. **LDKP3K**[3] is a collection of roughly 100,000 keyphrase-tagged long documents that were generated by mapping the KP20K corpus to S2ORC (Lo et al., 2020). For our experiments, we utilized the smaller version of this dataset, which includes 20,000 training samples, 3,413 validation samples, and 3,339 test samples.
3. The **NUS** dataset[4] comprises 211 scientific documents that were extracted and chosen utilizing the Google SOAP API. These documents were manually annotated and are solely used for evaluation, as the dataset only includes a test split.
4. The **DUC-2001** dataset[5] is composed of 308 news articles gathered from TREC-9, each of which is manually annotated with controlled keyphrases for evaluation purposes. Only a test split is included in the dataset.
5. **Inspec**[6] includes abstracts from 2,000 scientific articles, which have been split into three sets for training, validation, and testing purposes. The training set consists of 1,000 abstracts, while the validation and test sets contain 500 abstracts each.

Detailed statistics for these datasets can be found in Table 1.

4.2 Evaluation Metrics

We utilized $F1@K$ as our evaluation metric (Kim et al., 2010). Equations 4, 5, and 6 demonstrate how to calculate $F1@K$. Before evaluation, we preprocessed both the ground truth and predicted keyphrases by lowercasing, stemming, and removing punctuation. Evaluation relied on exact matching.

[2]https://huggingface.co/datasets/midas/semeval2010
[3]https://huggingface.co/datasets/midas/ldkp3k
[4]https://huggingface.co/datasets/midas/nus
[5]https://huggingface.co/datasets/midas/duc2001
[6]https://huggingface.co/datasets/midas/inspec

Let Y represent the set of ground truth keyphrases, and let $\bar{Y} = (\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_m)$ denote the ranked list of predicted keyphrases, where $\bar{y}_i$ is the i-th predicted keyphrase in descending order of confidence. The subset $\bar{Y}_k$ contains the top k elements from $\bar{Y}$. For consistency, k is set equal to K, which represents the total number of predicted keyphrases. We define the evaluation metrics as follows:

$$Precision@k = \frac{|Y \cap \bar{Y}_k|}{\min\{|\bar{Y}_k|, k\}} \tag{4}$$

$$Recall@k = \frac{|Y \cap \bar{Y}_k|}{|Y|} \tag{5}$$

$$F1@k = \frac{2 \cdot Precision@k \cdot Recall@k}{Precision@k + Recall@k} \tag{6}$$

Illustrative Example:

Suppose the ground truth keyphrases are $Y = \{\text{keyphrase}_1, \text{keyphrase}_2, \text{keyphrase}_3\}$, and the predicted ranked list of keyphrases is $\bar{Y} = (\text{keyphrase}_2, \text{keyphrase}_4, \text{keyphrase}_1)$. For $k = 2$, we take the top k predictions from $\bar{Y}$, giving the subset $\bar{Y}_k = \{\text{keyphrase}_2, \text{keyphrase}_4\}$.

Next, we compute the overlap between the ground truth and the top k predictions: $|Y \cap \bar{Y}|_k = 1$, as keyphrase_2 is the only match. Using this, the evaluation metrics are calculated as follows:

Precision@2:

$$\text{Precision@2} = \frac{|Y \cap \bar{Y}_k|}{\min\{|\bar{Y}_k|, k\}} = \frac{1}{2} = 0.5$$

Recall@2:

$$\text{Recall@2} = \frac{|Y \cap \bar{Y}_k|}{|Y|} = \frac{1}{3} \approx 0.33$$

F1@2:

$$\text{F1@2} = \frac{2 \cdot \text{Precision@2} \cdot \text{Recall@2}}{\text{Precision@2} + \text{Recall@2}} = \frac{2 \cdot 0.5 \cdot 0.33}{0.5 + 0.33} \approx 0.4$$

This example demonstrates the step-by-step process of computing the evaluation metrics and highlights the roles of Y, $\bar{Y}$, and $\bar{Y}_k$ in assessing keyphrase extraction performance.

4.3 Setup

The primary aim of the experiments is to investigate whether the incorporation of graph embeddings can improve the quality of word representations in PLMs by providing additional contextual information. The research employs various

Model	Domain	Maximum Input Tokens	#Parameters
DistilBERT	MultiDomain	512	66.4M
BERT	MultiDomain	512	109.5M
SciBERT	Scientific	512	109.9M
KBIR	MultiDomain - Specialized in KPE task	512	355.4M
LongFormer	MultiDomain - Specialized in Long Documents	4096	148.7M

Table 2: Details of models used in our experiments.

PLMs, including BERT[7] (Devlin et al., 2019), SciBERT[8] (Beltagy et al., 2019) (a specialized variant of BERT for scientific content), DistilBERT[9] (a distilled version of BERT), KBIR[10] (Kulkarni et al., 2022) (Keyphrase Boundary Infilling and Replacement, a RoBERTa-based model that achieves state-of-the-art performance in the KPE task), and Longformer[11] (Beltagy et al., 2020) (a transformer-based model specifically designed for efficient handling of long input sequences). Pertinent model details are presented in Table 2.

Throughout our experiments, we conducted fine-tuning on all PLMs using two approaches: (a) popularly used token classification approach[12] for sequence tagging, (b) *graph-enhanced* sequence tagging approach as explained in Section 3.4. Our aim was to assess the ability of the PLMs to assimilate the additional information derived from the graph embeddings during training and to determine whether the inclusion of such embeddings could enhance the fine-tuning process leading to improvement in performance.

For instance, we utilized SciBERT to explore the possibility of enhancing the PLM's representation by incorporating graph embeddings, despite its customization to a particular domain. Through this approach, we aimed to examine the potential of graph embeddings in providing supplementary information beyond the domain-specific knowledge.

Furthermore, we employed Longformer, a transformer model with an expanded token capacity (4,096 tokens), to investigate the effectiveness of our approach. Specifically, we aimed to assess whether the incorporation of graph information could improve the quality of representation, even in the presence of a robust contextual understanding of long sequences in the longformer model.

[7] https://huggingface.co/bert-base-uncased
[8] https://huggingface.co/allenai/scibert_scivocab_uncased
[9] https://huggingface.co/distilbert-base-uncased
[10] https://huggingface.co/bloomberg/KBIR
[11] https://huggingface.co/allenai/longformer-base-4096
[12] https://huggingface.co/tasks/token-classification

We evaluated the effectiveness of the models on both long and short documents, demonstrating the viability and effectiveness of our strategy for both forms of input length. Specifically, for short documents, we trained and evaluated our models exclusively on the Inspec corpus. For long documents, we trained our models on two distinct datasets, SemEval-2010 and LDKP3K, and assessed their performance on the test splits of all the datasets. Notably, the training set of SemEval-2010 contains only 144 documents, which presents a challenging few-shot learning scenario. In contrast, LDKP3K includes a larger training set of 20,000 samples, enabling us to validate the efficacy of our strategy across both small and large training data.

As the Inspec dataset comprises short documents that can be accommodated within BERT's context limit, we only employed BERT to evaluate the performance of our models. To establish a baseline for comparison, we trained and tested all the proposed architectures without incorporating graph embeddings as input.

To construct the co-occurrence matrix for each input document, we used a sliding window of size 4 to capture token co-occurrence relationships. For graph embedding computation, a link prediction dataset was generated with a negative sampling ratio of 5. These values were empirically determined as optimal during preliminary experiments to balance graph sparsity and computational efficiency. Each training dataset was then used to train a dedicated GCN for 5 epochs, employing the mean aggregation function during the convolutional layers. The embedding dimension, as well as the dimensions of the first and second convolutional layers, was fixed at 192. Model performance was evaluated using the AUC-ROC score, and the best-performing GCN was selected to compute the final node embeddings.

The sequence tagging models were trained using mini-batching with a batch size of 10. The models underwent 100 epochs of training, utilizing the AdamW algorithm with a learning rate of 5e-5, a patience value of 5, and an annealing factor of 0.5. During fine-tuning, the baseline models were trained solely with the classifier head. In contrast, for our proposed strategy of a *Graph Enhanced Sequence Tagger*, we integrated the graph embeddings and fused their representations with that of the PLM, as depicted in Fig 5.

To facilitate replication, we offer a sample code implementation of our experiment on GitHub[13]. The provided notebooks walk through the detailed step-by-step implementation of our keyphrase extraction method for long documents, using the SemEval2010 dataset and Bloomberg's language model KBIR as examples. Additionally, we present a long document fine-tuning procedure for the model, excluding the use of graph embeddings, designed explicitly for benchmarking purposes. Our methodology allows for straightforward replication across different models and datasets by adjusting relevant parameters.

[13] https://github.com/RobertoMCA/keyphrase-extraction-with-graph-embeddings

5 Results

In this section, we present the results of our experiments. For each model variant, we report the average score on the test set for three models trained with different random seeds. We observed that incorporating graph embeddings into the representation consistently improved the performance of the models across all settings. The improvement was particularly notable for longer documents. To name our models, we follow the convention of indicating the PLM embedding used in the first part of the name, and appending the term "Graph Enhanced" for the *Graph Enhanced Sequence Tagger* using the same PLM.

5.1 Long Documents

We conducted experiments to assess the effectiveness of incorporating graph embeddings in contextualized word representations for lengthy documents that exceed the PLM's input length limitation. The models were trained on SemEval-2010 and LDKP3K datasets, and the results of these experiments are presented in Table 3 (for models trained on SemEval-2010 train data) and Table 4 (for models trained on LDKP3K train data), respectively.

Our results demonstrate that graph embeddings enhance the representation in all settings, indicating that they provide crucial contextual information to the sequence tagging model, which is essential for the KPE task. Notably, the performance improvement is particularly significant for domain-specific models like SciBERT. We postulate that there is a mutually beneficial relationship between leveraging domain-specific pre-training and utilizing graph embedding information. Nonetheless, all models benefit from incorporating graph embeddings, highlighting the importance of our approach in capturing essential information for KPE from lengthy sequences that cannot be obtained through domain or task-specific pre-training of PLMs.

The improvement of F1 scores observed in the LongFormer variant across all the experiments highlights the usefulness of incorporating graph representation, even in cases where a PLM can represent longer sequences (4,096 vs 512). Our findings suggest that the training of PLMs with graph representation allows better understanding of long contexts and the development of better word representations by capturing both in-context and out-of-context relationships between words, leading to significant improvements in the results across all the datasets.

The results obtained by training the models on LDKP3K, which has a larger training dataset, are reported in Table 4. As expected, the availability of a larger training dataset results in a more significant improvement in performance. This is evident in the enhanced results across all datasets, including SemEval-2010, LDKP3K, and NUS, as reported in Table 4. These findings support our hypothesis that training the PLM and graph embeddings jointly results in a synergistic effect, which is further amplified by the availability of more data. Moreover, with the increase in the training data, the model can

Model	SemEval-2010	% improve	DUC-2001	% improve	NUS	% improve	LDKP3K	% improve
DistilBERT	0.228 ± 0.011	3.07%	0.034 ± 0.012	67.65%	0.170 ± 0.005	1.76%	0.108 ± 0.006	9.26%
Graph Enhanced DistilBERT	0.235 ± 0.004		0.057 ± 0.003		0.173 ± 0.006		0.118 ± 0.005	
BERT	0.240 ± 0.004	2.08%	0.078 ± 0.005	8.97%	0.173 ± 0.004	2.89%	0.118 ± 0.001	3.39%
Graph Enhanced BERT	0.245 ± 0.001		**0.085** ± 0.014		0.178 ± 0.006		0.122 ± 0.003	
SciBERT	0.236 ± 0.007	4.24%	0.029 ± 0.006	13.79%	0.178 ± 0.010	8.99%	0.133 ± 0.012	10.53%
Graph Enhanced SciBERT	0.246 ± 0.014		0.033 ± 0.014		0.194 ± 0.008		**0.147** ± 0.010	
KBIR	0.251 ± 0.010	1.59%	0.047 ± 0.027	23.40%	0.189 ± 0.025	1.06%	0.133 ± 0.010	1.50%
Graph Enhanced KBIR	**0.255** ± 0.009		0.058 ± 0.016		0.191 ± 0.002		0.135 ± 0.007	
Longformer	0.235 ± 0.012	4.68%	0.043 ± 0.005	34.88%	0.198 ± 0.006	2.02%	0.141 ± 0.010	2.84%
Graph Enhanced Longformer	0.246 ± 0.014		0.068 ± 0.006		**0.202** ± 0.003		0.145 ± 0.012	

Table 3: Performance of models trained on SemEval-2010 train data and evaluated on the test splits of all the datasets.

Model	SemEval-2010	% improve	DUC-2001	% improve	NUS	% improve	LDKP3K	% improve
DistilBERT	0.133 ± 0.008	9.02%	0.046 ± 0.003	19.57%	0.172 ± 0.004	13.37%	0.216 ± 0.005	5.56%
Graph Enhanced DistilBERT	0.145 ± 0.009		0.055 ± 0.004		0.195 ± 0.003		0.228 ± 0.007	
BERT	0.153 ± 0.008	9.80%	0.079 ± 0.008	56.96%	0.188 ± 0.003	19.68%	0.238 ± 0.003	5.88%
Graph Enhanced BERT	**0.168** ± 0.003		**0.124** ± 0.005		0.225 ± 0.007		0.252 ± 0.002	
SciBERT	0.146 ± 0.010	13.70%	0.017 ± 0.010	123.53%	0.190 ± 0.008	26.84%	0.245 ± 0.009	6.94%
Graph Enhanced SciBERT	0.166 ± 0.015		0.038 ± 0.015		**0.241** ± 0.006		0.262 ± 0.010	
KBIR	0.148 ± 0.008	8.11%	0.070 ± 0.010	17.14%	0.191 ± 0.011	9.95%	0.252 ± 0.009	5.16%
Graph Enhanced KBIR	0.160 ± 0.011		0.082 ± 0.011		0.210 ± 0.008		0.265 ± 0.011	
Longformer	0.146 ± 0.015	14.38%	0.085 ± 0.004	7.06%	0.207 ± 0.005	3.86%	0.260 ± 0.012	4.23%
Graph Enhanced Longformer	0.167 ± 0.011		0.091 ± 0.007		0.215 ± 0.004		**0.271** ± 0.008	

Table 4: Performance of models trained on LDKP3K train data and evaluated on the test splits of all the datasets.

better capture the long-term dependencies present in the graph representation, leading to more effective learning.

In addition, when graph embeddings are integrated, the models demonstrate significant improvements in their performance on the news domain (DUC-2001), despite being fine-tuned on scientific documents. This suggests that the inclusion of graph embeddings enhances the models' ability to handle domain-specific distribution shifts. The domain-agnostic nature of graph embeddings makes them a promising approach for addressing challenges related to domain adaptation. Consequently, graph embeddings can serve

Model	F1	F1 (our approach)	% improve
DistilBERT	0.506 ± 0.004	0.509 ± 0.005	0.59%
BERT	0.514 ± 0.012	0.529 ± 0.013	2.92%
SciBERT	0.518 ± 0.001	0.530 ± 0.006	2.32%
KBIR	0.562 ± 0.012	**0.570** ± 0.002	1.42%

Table 5: Performance of PLMs and Graph Enhance Sequence Tagger on Inspec.

as a valuable data augmentation technique, particularly in scenarios where domain-specific data is scarce.

Given that currently keyphrase extraction models heavily rely on training and evaluating models within the scientific domain, domain adaptation becomes a crucial aspect of interest. Our findings in this study strongly indicate that incorporating graph embeddings holds promise as a means to address the domain adaptation problem in keyphrase extraction. This highlights the potential for further exploration in utilizing graph embeddings for effectively tackling domain adaptation challenges in this field.

5.2 Short Documents

In this section, we present the results of our experiments on the Inspec dataset, which aim to assess whether incorporating graph embeddings can improve the performance of PLM-based models even for short documents that fit within the PLM's context. Table 5 summarizes the findings.

Our analysis of the results indicates that incorporating graph embeddings indeed enhances the PLM's representation even for short documents. This improvement is consistently observed across all evaluated models, including SciBERT, which was pre-trained on a domain-specific corpus, and KBIR, which underwent pretraining with task-specific objectives. Specifically, the addition of graph information leads to an average performance gain of more than 1 point in the F1 score.

Based on our experimental observations, it is evident that incorporating graph embeddings enhances the representation of local context by capturing relationships that may not be readily apparent in sequential representations from PLMs. This is attributed to the unique portrayal of words in the graph, which enables the incorporation of connections that may not be fully captured by the PLM alone. The integration of graph and PLM representations synergistically enhances the embedding of local context, resulting in improved overall performance.

6 Case Studies

In this section, we present concrete examples that demonstrate how our methodology enhances KPE from lengthy documents. To illustrate the effectiveness of our approach, we compare the performance of two models: KBIR, which is currently the state-of-the-art model in this task and has been fine-tuned on the Inspec dataset, and LongFormer, a high-performing model specifically designed for processing long documents, fine-tuned on SemEval-2010. We will contrast their results with those generated by one of our models, namely the Graph Enhanced LongFormer, which has been trained on SemEval-2010.

6.1 Case Study 1

This example is taken from the SemEval-2010 test sample with ID 'J-9'. The KPs that all methods were able to predict correctly are highlighted in red, while the KPs that only our model could predict correctly are highlighted in green.

Computation in a Distributed Information Market

ABSTRACT

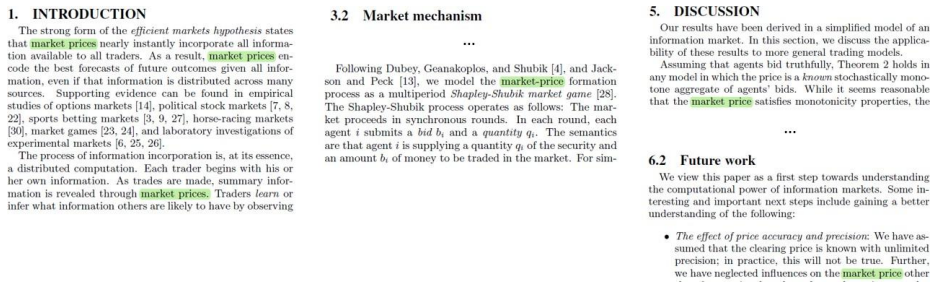
According to economic theory—supported by empirical and laboratory evidence—the equilibrium price of a financial security reflects all of the information regarding the security's value. We investigate the computational process on the path toward equilibrium, where information distributed among traders is revealed step-by-step over time and incorporated into the market price. We develop a simplified model of an information market, along with trading strategies, in order to formalize the computational properties of the process. We show that securities whose payoffs cannot be expressed as weighted threshold functions of distributed input bits are not guaranteed to converge to the proper equilibrium predicted by economic theory. On the other hand, securities whose payoffs are threshold functions *are* guaranteed to converge, for all prior probability distributions. Moreover, these *threshold securities* converge in at most n rounds, where n is the number of bits of distributed information. We also prove a lower bound, showing a type of threshold security that requires at least $n/2$ rounds to converge in the worst case.

Fig. 6: Case Study 1 - Title and Abstract

Figure 6 displays the title and abstract of the sample. As evident from the text, the phrases "information market" and "financial security" are repeated multiple times, including in the title, indicating their significance. However, the term "market price" is mentioned only once, and its importance cannot

be determined from its local context. Further analysis, as illustrated in Figure 7, reveals that although the term appears eight times in the document, it is not identified in the introduction, methodology, or discussion without the use of graph embeddings.

This suggests that the KBIR model, which has a limited local context, cannot assess its importance. Even the LongFormer model, which can capture longer contexts of up to 4,096 tokens, fails to identify it as a keyphrase. In contrast, our graph-enhanced approach demonstrates a richer and more comprehensive understanding of the input contexts, providing the necessary information for the model to recognize "market price" as a keyphrase.



1. INTRODUCTION

The strong form of the *efficient markets hypothesis* states that market prices nearly instantly incorporate all information available to all traders. As a result, market prices encode the best forecasts of future outcomes given all information, even if that information is distributed across many sources. Supporting evidence can be found in empirical studies of options markets [14], political stock markets [7, 8, 22], sports betting markets [3, 9, 27], horse-racing markets [30], market games [23, 24], and laboratory investigations of experimental markets [6, 25, 26].

The process of information incorporation is, at its essence, a distributed computation. Each trader begins with his or her own information. As trades are made, summary information is revealed through market prices. Traders *learn* or infer what information others are likely to have by observing

3.2 Market mechanism

...

Following Dubey, Geanakoplos, and Shubik [4], and Jackson and Peck [13], we model the market-price formation process as a multiperiod *Shapley-Shubik market game* [28]. The Shapley-Shubik process operates as follows: The market proceeds in synchronous rounds. In each round, each agent i submits a *bid* b_i and a *quantity* q_i. The semantics are that agent i is supplying a quantity q_i of the security and an amount b_i of money to be traded in the market. For sim-

5. DISCUSSION

Our results have been derived in a simplified model of an information market. In this section, we discuss the applicability of these results to more general trading models.

Assuming that agents bid truthfully, Theorem 2 holds in any model in which the price is a *known* stochastically monotone aggregate of agents' bids. While it seems reasonable that the market price satisfies monotonicity properties, the

...

6.2 Future work

We view this paper as a first step towards understanding the computational power of information markets. Some interesting and important next steps include gaining a better understanding of the following:

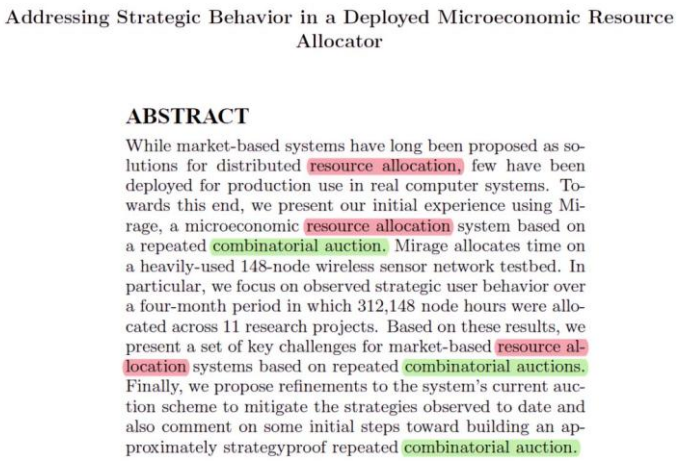
- *The effect of price accuracy and precision*: We have assumed that the clearing price is known with unlimited precision; in practice, this will not be true. Further, we have neglected influences on the market price other

Fig. 7: Case Study 1 - Excerpts from the document where the KPs are mentioned. The KPs that all methods were able to predict correctly are highlighted in red, while the KPs that only our model could predict correctly are highlighted in green.

In order for a human to identify the keyphrase "market price", they would need to comprehend information from various sections of the paper, such as the abstract, introduction, related work, and methodology, and view the document as a cohesive whole. This holistic perspective is essential for comprehending lengthy documents, and it is precisely what graph embeddings aim to capture.

6.2 Case Study 2

This example is taken from the SemEval-2010 test sample with ID 'C-86'. The KPs that all methods were able to predict correctly are highlighted in red, while the KPs that only our model could predict correctly are highlighted in green.



Addressing Strategic Behavior in a Deployed Microeconomic Resource Allocator

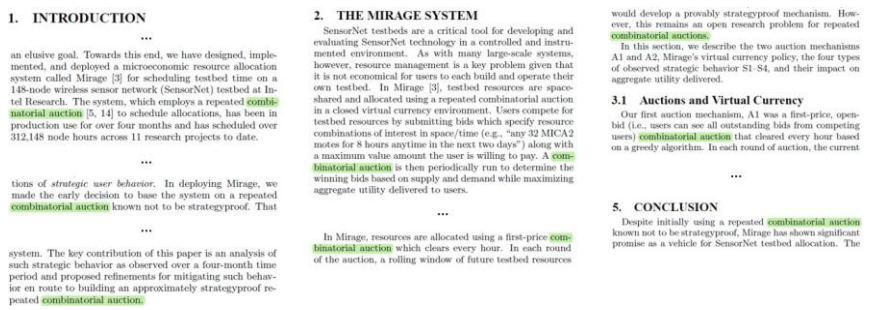
ABSTRACT

While market-based systems have long been proposed as solutions for distributed resource allocation, few have been deployed for production use in real computer systems. Towards this end, we present our initial experience using Mirage, a microeconomic resource allocation system based on a repeated combinatorial auction. Mirage allocates time on a heavily-used 148-node wireless sensor network testbed. In particular, we focus on observed strategic user behavior over a four-month period in which 312,148 node hours were allocated across 11 research projects. Based on these results, we present a set of key challenges for market-based resource allocation systems based on repeated combinatorial auctions. Finally, we propose refinements to the system's current auction scheme to mitigate the strategies observed to date and also comment on some initial steps toward building an approximately strategyproof repeated combinatorial auction.

Fig. 8: Case Study 2 - Title and Abstract.

Figure 8 presents the title and abstract of the sample. In the given input context, "resource allocation" is a keyphrase, while the importance of "combinatorial auction" remains unclear to the KBIR and LongFormer models, despite frequent references throughout the document. As shown in Figure 9, these models struggle to recognize the phrase, despite its appearance 19 times in the paper, potentially due to their limited understanding of the global context.

In contrast, our graph-enhanced approach demonstrates superior performance by identifying "combinatorial auction" as a keyphrase. This highlights the importance of a more comprehensive understanding of the document's global context, which is a critical advantage of our methodology.



1. INTRODUCTION

...

an elusive goal. Towards this end, we have designed, implemented, and deployed a microeconomic resource allocation system called Mirage [3] for scheduling testbed time on a 148-node wireless sensor network (SensorNet) testbed at Intel Research. The system, which employs a repeated combinatorial auction [5, 14] to schedule allocations, has been in production use for over four months and has scheduled over 312,148 node hours across 11 research projects to date.

...

tions of *strategic user behavior*. In deploying Mirage, we made the early decision to base the system on a repeated combinatorial auction known not to be strategyproof. That

...

system. The key contribution of this paper is an analysis of such strategic behavior as observed over a four-month time period and proposed refinements for mitigating such behavior en route to building an approximately strategyproof repeated combinatorial auction.

2. THE MIRAGE SYSTEM

SensorNet testbeds are a critical tool for developing and evaluating SensorNet technology in a controlled and instrumented environment. As with many large-scale systems, however, resource management is a key problem given that it is not economical for users to each build and operate their own testbed. In Mirage [3], testbed resources are space-shared and allocated using a repeated combinatorial auction in a closed virtual currency environment. Users compete for testbed resources by submitting bids which specify resource combinations of interest in space/time (e.g., "any 32 MICA2 motes for 8 hours anytime in the next two days") along with a maximum value amount the user is willing to pay. A combinatorial auction is then periodically run to determine the winning bids based on supply and demand while maximizing aggregate utility delivered to users.

...

In Mirage, resources are allocated using a first-price combinatorial auction which clears every hour. In each round of the auction, a rolling window of future testbed resources

would develop a provably strategyproof mechanism. However, this remains an open research problem for repeated combinatorial auctions.

In this section, we describe the two auction mechanisms A1 and A2, Mirage's virtual currency policy, the four types of observed strategic behavior S1–S4, and their impact on aggregate utility delivered.

3.1 Auctions and Virtual Currency

Our first auction mechanism, A1 was a first-price, open-bid (i.e., users can see all outstanding bids from competing users) combinatorial auction that cleared every hour based on a greedy algorithm. In each round of auction, the current

...

5. CONCLUSION

Despite initially using a repeated combinatorial auction known not to be strategyproof, Mirage has shown significant promise as a vehicle for SensorNet testbed allocation. The

Fig. 9: Case Study 2 - Excerpts from the document where the KPs are mentioned. The KPs that all methods were able to predict correctly are highlighted in red, while the KPs that only our model could predict correctly are highlighted in green.

7 Limitations, Conclusion and Future Work

7.1 Limitations

The integration of graph embeddings has proven to enhance the performance of KPE models for lengthy documents by providing a more comprehensive understanding of a word's context within the entire text. However, it should be acknowledged that this computational approach adds complexity to the pipeline.

To evaluate the computational overhead introduced by graph embeddings, we measured the average processing time required by the KPE pipeline to analyze sample documents, both with and without graph embeddings, using an NVIDIA GeForce RTX 3090 GPU. Incorporating graph embeddings increased the processing time by approximately 20% for shorter documents (e.g., samples from the Inspec dataset; detailed statistics are provided in Section 4.1) and up to 50% for longer documents (e.g., samples from the SemEval-2010 dataset; detailed statistics are provided in Section 4.1), compared to a baseline KPE pipeline without graph embeddings. To ensure robustness, these measurements were averaged across datasets with diverse document lengths.

Despite the additional computational time, we consider the improved performance achieved through our graph-enhanced approach as a valuable trade-off. By enabling a richer and more global understanding of the document's context, our methodology enhances the accuracy of KPE for lengthy texts, which is particularly important in various real-world applications.

It is important to acknowledge that our study focuses on PLMs that fit within our computational budget. However, it is worth noting that significant progress has been made in the domain of Large Language Models (LLMs), which can process text sequences up to 32,768 tokens in length (OpenAI, 2023, Technical Report). Given the limitations of our study, we cannot assess the

performance of our approach on such LLMs. Nonetheless, we anticipate that our graph-enhanced approach can be adapted and applied to these models to further enhance their performance in KPE for lengthy texts.

7.2 Conclusion and Future Work

In this study, we introduced the "Graph Enhanced Sequence Tagger," which effectively combines the strengths of PLMs and graph representations to enhance KPE in long documents. Our experimental results convincingly demonstrated that our proposed graph-enhanced approach outperforms state-of-the-art models, irrespective of document length. Notably, even specialized PLMs like Longformer, known for their proficiency in capturing long contexts in lengthy documents, benefited from our approach. The joint fine-tuning of the PLM and graph embedding yields a synergistic effect, enabling effective capture of long-term relationships and incorporation of topic-specific information. Furthermore, the promising results on out-of-domain datasets highlight the potential of our approach in domain adaptation.

Future Work. While our proposed method achieved notable success, there remain several opportunities for further improvement. One promising direction is to explore mechanisms for dynamically adjusting the influence of high-frequency or common words in the co-occurrence graph. Techniques such as edge-weight scaling or attention-based mechanisms could be employed to reduce the disproportionate impact of such terms, thereby enhancing the graph's ability to capture semantically meaningful relationships without necessitating their outright removal. This refinement could improve consistency between the graph structure and the PLM-based fine-tuning process.

Additionally, to demonstrate the generalizability of our graph-enhanced architecture, future work could evaluate alternative GNN models, such as GAT or Graph Isomorphism Networks (GIN). These models may offer complementary strengths, potentially improving keyphrase extraction performance in various settings.

Finally, we aim to explore other graph representations, including lexical and syntactic dependency graphs, as well as self-supervised training objectives more closely aligned with the KPE task. Such objectives could help further enrich the learned graph representations. Another exciting avenue is the incorporation of graph-based approaches into domain adaptation techniques for keyphrase extraction and generation, expanding the applicability of our method to diverse and unseen datasets.

8 Acknowledgments

In the course of our research, we would like to acknowledge the insightful discussions and ideas generated from the preprint version of our own work, titled 'Enhancing KPE from Long Scientific Documents using Graph Embeddings'

authored by Roberto Mart´ınez-Cruz, Debanjan Mahata, Alvaro J. Lõpez-Lõpez and Jos´e Portela. This preprint was made available on arXiv[14] in May 2023.

9 Statements and Declarations

9.1 Competing interests

The authors declare no competing interests related to this study. All methodologies, analyses, and interpretations of data were conducted independently and without influence from external entities. This research was purely academic and aimed at contributing to the existing body of knowledge on keyphrase extraction from long documents using graph embeddings.

9.2 Authors Contribution Statement

Roberto Mart´ınez-Cruz and Debanjan Mahata contributed equally to the conception and design of the study, as well as the development of the methodology and analysis of data. Alvaro J. Lõpez-Lõpez and Jos´e Portela contributed to the collection and assembly of data, as well as providing critical revisions that added important intellectual content. All authors participated in drafting the manuscript and approved the final version to be published.

9.3 Ethical and Informed Consent

This research did not involve any human participants, data, or tissues, and therefore did not require ethical approval or informed consent. All datasets utilized in this study are publicly available and were openly published for research purposes, adhering to all applicable ethical standards.

9.4 Data Availability and Access

The datasets supporting the conclusions of this article are all publicly available and openly published for research purposes. Links to these datasets can be found in Section 4.1 Datasets of this paper. The availability of these datasets ensures transparency and allows for the reproducibility of the research findings.

For a detailed reference to the datasets used, including URLs and access methods, please refer to Section 4.1 in this document.

References

Alsentzer, E., J. Murphy, W. Boag, W.H. Weng, D. Jindi, T. Naumann, and M. McDermott 2019, June. Publicly available clinical BERT embeddings. In A. Rumshisky, K. Roberts, S. Bethard, and T. Naumann (Eds.), *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, Minneapolis, Minnesota, USA, pp. 72–78. Association for Computational Linguistics.

[14] https://arxiv.org/abs/2305.09316

Alzaidy, R., C. Caragea, and C.L. Giles 2019. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The World Wide Web Conference*, WWW '19, New York, NY, USA, pp. 2551–2557. Association for Computing Machinery.

Beltagy, I., K. Lo, and A. Cohan 2019, November. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, pp. 3615–3620. Association for Computational Linguistics.

Beltagy, I., M.E. Peters, and A. Cohan 2020. Longformer: The long-document transformer.

Bennani-Smires, K., C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi 2018, October. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium, pp. 221–229. Association for Computational Linguistics.

Boudin, F. 2013. A comparison of centrality measures for graph-based keyphrase extraction. In *Proceedings of the sixth international joint conference on natural language processing*, pp. 834–838.

Bougouin, A., F. Boudin, and B. Daille 2013, October. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Nagoya, Japan, pp. 543–551. Asian Federation of Natural Language Processing.

Çano, E. and O. Bojar 2019. Keyphrase generation: A multi-aspect survey. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pp. 85–94. IEEE.

Chalkidis, I., M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos 2020, November. LEGAL-BERT: The muppets straight out of law school. In T. Cohn, Y. He, and Y. Liu (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online, pp. 2898–2904. Association for Computational Linguistics.

Chen, Y., Z. Chen, and H.U. Amin 2023. Synergistic similarity graph construction (ssgc) for steel plate fault diagnosis with graph attention networks. In *2023 IEEE 6th International Conference on Knowledge Innovation and Invention (ICKII)*, pp. 655–660.

Devlin, J., M.W. Chang, K. Lee, and K. Toutanova 2019, June. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, pp. 4171–4186. Association for Computational Linguistics.

Doˇcekal, M. and P. Smrˇz 2022, may. Query-based keyphrase extraction from long documents. Volume 35. University of Florida George A Smathers Libraries.

Feng, K., J. Ji, Y. Zhang, Q. Ni, Z. Liu, and M. Beer 2023. Digital twin-driven intelligent assessment of gear surface degradation. Volume 186, pp. 109896.

Feng, K., Y. Xu, Y. Wang, S. Li, Q. Jiang, B. Sun, J. Zheng, and Q. Ni 2023. Digital twin enabled domain adversarial graph networks for bearing fault diagnosis. Volume 1, pp. 113–122.

Garg, K., J.R. Chowdhury, and C. Caragea 2021. Keyphrase generation beyond the boundaries of title and abstract. In *Conference on Empirical Methods in Natural Language Processing*.

Gollapalli, S.D., X.l. Li, and P. Yang 2017, Feb. Incorporating expert knowledge into keyphrase extraction. Volume 31.

Grail, Q., J. Perez, and E. Gaussier 2021, April. Globalizing BERT-based transformer architectures for long document summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online, pp. 1792–1810. Association for Computational Linguistics.

Grover, A. and J. Leskovec 2016. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, New York, NY, USA, pp. 855–864. Association for Computing Machinery.

Gutwin, C., G. Paynter, I. Witten, C. Nevill-Manning, and E. Frank 1999. Improving browsing in digital libraries with keyphrase indexes. Volume 27, pp. 81–104. Elsevier.

Hamilton, W.L., R. Ying, and J. Leskovec 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, pp. 1025–1035. Curran Associates Inc.

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

Hasan, K.S. and V. Ng 2014. 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)*, pp. 1262–1273.

Huang, Z., W. Xu, and K. Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Hulth, A. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, USA, pp. 216–223. Association for Computational Linguistics.

Hulth, A. and B. Megyesi 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 537–544.

Jones, S. and M.S. Staveley 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.

Kim, S.N. and M.Y. Kan 2009, August. Re-examining automatic keyphrase extraction approaches in scientific articles. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications (MWE 2009)*, Singapore, pp. 9–16. Association for Computational Linguistics.

Kim, S.N., O. Medelyan, M.Y. Kan, and T. Baldwin 2010. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, USA, pp. 21–26. Association for Computational Linguistics.

Kipf, T.N. and M. Welling 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Kulkarni, M., D. Mahata, R. Arora, and R. Bhowmik 2022, July. Learning rich representation of keyphrases from text. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Seattle, United States, pp. 891–906. Association for Computational Linguistics.

Lee, J., W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So, and J. Kang 2019, 09. Biobert: a pre-trained biomedical language representation model for biomedical text mining. Volume 36, pp. 1234–1240.

Lo, K., L.L. Wang, M. Neumann, R. Kinney, and D. Weld 2020, July. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, pp. 4969–4983. Association for Computational Linguistics.

Mahata, D., J. Kuriakose, R. Shah, and R. Zimmermann 2018a. Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 634–639.

Mahata, D., R.R. Shah, J. Kuriakose, R. Zimmermann, and J.R. Talburt 2018b. Theme-weighted ranking of keywords from text documents using phrase embeddings. pp. 184–189.

Martinc, M., B. Škrlj, and S. Pollak 2021, jun. TNT-KID: Transformer-based neural tagger for keyword identification. Volume 28, pp. 409–448. Cambridge University Press (CUP).

May, M.C., J. Neidhöfer, T. Körner, L. Schäfer, and G. Lanza 2022. Applying natural language processing in manufacturing. Volume 115, pp. 184–189. 10th CIRP Global Web Conference – Material Aspects of Manufacturing Processes.

Meng, R., D. Mahata, and F. Boudin 2022. From fundamentals to recent advances: A tutorial on keyphrasification. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*, pp. 582–588. Springer.

Miaschi, A. and F. Dell'Orletta 2020, July. Contextual and non-contextual word embeddings: an in-depth linguistic investigation. In S. Gella, J. Welbl, M. Rei, F. Petroni, P. Lewis, E. Strubell, M. Seo, and H. Hajishirzi (Eds.), *Proceedings of the 5th Workshop on Representation Learning for NLP*, Online, pp. 110–119. Association for Computational Linguistics.

Mienye, E., N. Jere, G. Obaido, I.D. Mienye, and K. Aruleba 2024. Deep learning in finance: A survey of applications and techniques. Volume 5, pp. 2066–2091.

Mihalcea, R. and P. Tarau 2004, July. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, pp. 404–411. Association for Computational Linguistics.

Mikolov, T., K. Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space.

Mothe, J., F. Ramiandrisoa, and M. Rasolomanana 2018. Automatic keyphrase extraction using graph-based methods. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pp. 728–730.

Nguyen, T.D. and M.Y. Kan 2007. Keyphrase extraction in scientific publications. In D. H.-L. Goh, T. H. Cao, I. T. Sølvberg, and E. Rasmussen (Eds.), *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, Berlin, Heidelberg, pp. 317–326. Springer Berlin Heidelberg.

Ni, Q., J. Ji, K. Feng, Y. Zhang, D. Lin, and J. Zheng 2024. Data-driven bearing health management using a novel multi-scale fused feature and gated recurrent unit. Volume 242, pp. 109753.

Ni, Q., J. Ji, B. Halkon, K. Feng, and A.K. Nandi 2023. Physics-informed residual network (piresnet) for rolling element bearing fault diagnostics. Volume 200, pp. 110544.

Page, L., S. Brin, R. Motwani, and T. Winograd 1999. The pagerank citation ranking : Bringing order to the web. In *The Web Conference*.

Park, H., Y. Vyas, and K. Shah 2022, May. Efficient classification of long documents using transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Dublin, Ireland, pp. 702–709. Association for Computational Linguistics.

Park, S. and C. Caragea 2020. Scientific keyphrase identification and classification by pre-trained language models intermediate task transfer learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 5409–5419.

Patel, K. and C. Caragea 2019. Exploring word embeddings in crf-based keyphrase extraction from research papers. In *Proceedings of the 10th International Conference on Knowledge Capture*, pp. 37–44.

Pennington, J., R. Socher, and C. Manning 2014, October. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1532–1543. Association for Computational Linguistics.

Perozzi, B., R. Al-Rfou, and S. Skiena 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, New York, NY, USA, pp. 701–710. Association for Computing Machinery.

Pham, H., G. Wang, Y. Lu, D. Florencio, and C. Zhang. 2022. Understanding long documents with different position-aware attentions.

Qazvinian, V., D. Radev, and A. Özgür 2010. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd international conference on computational linguistics (COLING 2010)*, pp. 895–903.

Rousseau, F. and M. Vazirgiannis 2015. Main core retention on graph-of-words for single-document keyword extraction. In A. Hanbury, G. Kazai, A. Rauber, and N. Fuhr (Eds.), *Advances in Information Retrieval*, Cham, pp. 382–393. Springer International Publishing.

Rungta, M., R. Kumar, M.P. Dhaliwal, H. Tiwari, and V. Vala 2020. Transkp: Transformer based key-phrase extraction. pp. 1–7.

Sahrawat, D., D. Mahata, H. Zhang, M. Kulkarni, A. Sharma, R. Gosangi, A. Stent, Y. Kumar, R.R. Shah, and R. Zimmermann 2020. Keyphrase extraction as sequence labeling using contextualized embeddings. In *European Conference on Information Retrieval*, pp. 328–335. Springer.

Santosh, T., D. Kumar Sanyal, P.K. Bhowmick, and P.P. Das 2020, December. SaSAKE: Syntax and semantics aware keyphrase extraction from research papers. In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online), pp. 5372–5383. International Committee on Computational Linguistics.

Saxena, A., G. Fletcher, and M. Pechenizkiy 2021. Nodesim: node similarity based network embedding for diverse link prediction. Volume 11.

Song, I.Y., R.B. Allen, Z. Obradovic, and M. Song 2006. Keyphrase extraction-based query expansion in digital libraries. In *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'06)*, pp. 202–209. IEEE.

Stark, C., B.J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers 2006, 01. Biogrid: a general repository for interaction datasets. Volume 34, pp. D535–D539.

Sun, Z., J. Tang, P. Du, Z.H. Deng, and J.Y. Nie 2019. Divgraphpointer: A graph pointer network for extracting diverse keyphrases. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, New York, NY, USA, pp. 755–764. Association for Computing Machinery.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, pp. 6000–6010. Curran Associates Inc.

Veli˘ckovi´c, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio 2018. Graph attention networks. In *International Conference on Learning Representations*.

Wan, X. and J. Xiao 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, pp. 855–860. AAAI Press.

Wang, P., K. Agarwal, C. Ham, S. Choudhury, and C.K. Reddy 2021. Self-supervised learning of contextual embeddings for link prediction in heterogeneous networks. In *Proceedings of the Web Conference 2021*, WWW '21, New York, NY, USA, pp. 2946–2957. Association for Computing Machinery.

Wang, R., W. Liu, and C. McDonald 2014. Corpus-independent generic keyphrase extraction using word embedding vectors. In *Software engineering research conference*, Volume 39, pp. 1–8.

Xiao, C., Z. Liu, Y. Lin, and M. Sun 2023. *Legal Knowledge Representation Learning*, pp. 401–432. Singapore: Springer Nature Singapore.

Yang, L., M. Zhang, C. Li, M. Bendersky, and M. Najork 2020. Beyond 512 tokens: Siamese multi-depth transformer-based hierarchical encoder for long-form document matching. CIKM '20, New York, NY, USA, pp. 1725–1734. Association for Computing Machinery.

Yang, Y., M.C.S. UY, and A. Huang. 2020. Finbert: A pretrained language model for financial communications.

Yasunaga, M., J. Kasai, R. Zhang, A.R. Fabbri, I. Li, D. Friedman, and D.R. Radev 2019. Scisummnet: a large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Ye, J., R. Cai, T. Gui, and Q. Zhang 2021, November. Heterogeneous graph neural networks for keyphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, pp. 2705–2715. Association for Computational Linguistics.

Ying, R., R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, and J. Leskovec 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, New York, NY, USA,

pp. 974–983. Association for Computing Machinery.

Zaheer, M., G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed 2020. Big bird: Transformers for longer sequences. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Zhang, R., Z. Wei, Y. Shi, and Y. Chen. 2020. Bert-al: Bert for arbitrarily long document understanding.

Zhang, Y., N. Zincir-Heywood, and E. Milios 2004. World wide web site summarization. Volume 2, pp. 39–53. IOS Press.

Zhong, K., T. Jackson, A. West, and G. Cosma 2024. Natural language processing approaches in industrial maintenance: A systematic literature review. Volume 232, pp. 2082–2097. 5th International Conference on Industry 4.0 and Smart Manufacturing (ISM 2023).