applied sciences

MDPI

Article

Multi-Feature-Enhanced Academic Paper Recommendation Model with Knowledge Graph

Le Wang, Wenna Du and Zehua Chen *

College of Computer Science and Technology (College of Data Science), Taiyuan University of Technology, Taiyuan 030024, China; wl187887010001@163.com (L.W.); duwenna1387@link.tyut.edu.cn (W.D.)
* Correspondence: chenzehua@tyut.edu.cn

Abstract: This paper addresses the challenges of data sparsity and personalization limitations inherent in current recommendation systems when processing extensive academic paper datasets. To overcome these issues, the present work introduces an innovative recommendation model that integrates the wealth of structured information from knowledge graphs and refines the amalgamation of temporal and relational data. By applying attention mechanisms and neural network technologies, the model thoroughly explores the text characteristics of papers and the evolving patterns of user behaviors. Additionally, the model elevates the accuracy and personalization of recommendations by meticulously examining citation patterns among papers and the networks of author collaboration. The experimental findings show that the present model surpasses baseline models on all evaluation metrics, thereby enhancing the precision and personalization of academic paper recommendations.

Keywords: academic paper recommendation; knowledge graph; neural networks; attention mechanism; sequential recommendation

check for updates

Citation: Wang, L.; Du, W.; Chen, Z. Multi-Feature-Enhanced Academic Paper Recommendation Model with Knowledge Graph. *Appl. Sci.* **2024**, *14*, 5022. https://doi.org/10.3390/app14125022

Academic Editor: João M. F. Rodrigues and Valentino Santucci

Received: 21 April 2024
Revised: 27 May 2024
Accepted: 7 June 2024
Published: 9 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid development of information technology and the internet, human society has quickly entered the information age. The application of technologies such as cloud computing, the Internet of Things, and big data on the internet has not only greatly enriched data resources but also led to a geometric increase in data scale [1,2]. This phenomenon has promoted the emergence of new disciplines and interdisciplinary integration, thereby driving an explosive growth in the number of academic papers [3]. Researchers in the academic community are therefore facing an unprecedented challenge of information overload [4,5] as they need to search and filter relevant information from a vast array of literature, which undoubtedly increases the difficulty and time cost of acquiring knowledge [6]. To effectively address this challenge, academic paper recommendation systems have emerged [7] that aim to help researchers by alleviating the burden of information overload through advanced computing and ranking mechanisms and recommending papers closely related to their research interests and focuses [8]. Although traditional recommendation methods, such as collaborative filtering and content-based filtering, have achieved some success [9], they still have shortcomings in terms of recommendation efficiency and accuracy when dealing with large-scale datasets. These methods often fail to fully utilize key content such as the text information of papers (e.g., titles, abstracts, etc.) and the citation relationships between papers [10]. In addition, they also face issues like data sparsity and cold starts, which limit their ability to provide personalized recommendations.

In recent years, recommendation models based on deep learning and graphs have received widespread attention for their potential in integrating multi-source information and have made significant progress in the field of academic paper recommendation [11,12]. In particular, the introduction of knowledge graphs has brought a new perspective to recommendation systems. As a structured semantic knowledge base, knowledge graphs

can richly represent the relationships between entities and provide a new method of data representation and reasoning for recommendation systems.

The present work aims to explore how to enhance the accuracy and personalization of academic paper recommendation systems by integrating deep learning, knowledge graphs, and strategies for the fusion of temporal and relational information. A feature-enriched paper recommendation model that integrates knowledge graphs is proposed that not only utilizes attention mechanisms to deeply mine the text and relational features of papers but also integrates the temporal information of paper publication and citation, as well as the edge information in the academic citation network. Through this multidimensional feature fusion, the model presented in this paper can more accurately capture the dynamic research interests of users and the complex academic relationships between papers. The main contributions of this paper are as follows:

1. The model presented in this paper integrates textual and relational features of academic papers, fusing temporal information of publication and citation, along with edge information from the academic citation network, to construct a more representative vector representation of papers.

2. This paper's model introduces advanced neural network architectures and attention mechanisms to effectively capture the temporal information of users' behavioral patterns and the complex relationships within the academic network, thereby generating a more accurate representation of user interest features.

3. The effectiveness of the proposed model has been validated through experiments. Compared to traditional recommendation algorithms and baseline models, the model has demonstrated improved accuracy and personalization in recommendations.

2. Related Work

2.1. Traditional Academic Paper Recommendation Methods

Traditional academic paper recommendation systems predominantly employ methods such as collaborative filtering and content-based filtering. Collaborative filtering methods are based on user–paper interaction behaviors and recommend by mining the similarities between users and papers. For instance, Naak et al. [13] proposed a multi-criteria collaborative filtering approach for research paper recommendation on a platform. Liu et al. [14] introduced an innovative context-aware collaborative filtering method. Sugiyama et al. [15] presented a recommendation method that combines citation network embedding and collaborative filtering techniques by mining the latent citation relationships of papers. However, collaborative filtering is susceptible to the sparsity problem and has lower recommendation efficiency when dealing with large-scale paper corpora. Content-based filtering methods, on the other hand, focus on analyzing textual content and features, such as titles, abstracts, and citation relationships among papers. The paper recommendation method by Philip et al. [16] utilizes the textual content and metadata of academic papers in conjunction with user preferences and query history. Nevertheless, traditional content-based filtering methods do not fully leverage the textual information and complex relationships of academic papers, making it challenging to achieve personalized recommendations.

2.2. Graph-Based Academic Paper Recommendation Methods

Recently, graph-based academic paper recommendation methods have also garnered the interest of researchers. Graphs, as a structured form of knowledge representation, can effectively express the knowledge structure and relationships within academic fields. Some researchers have proposed methods for constructing academic paper recommendation systems using graphs, aiming to achieve more accurate and personalized recommendations by mining academic entities and relationships within the graph. For example, Ma et al. [17] proposed a heterogeneous graph representation-based recommendation method called HGRec, which uses two meta-path-based proximity measures to jointly update the node embeddings in a heterogeneous graph. Cai et al. [18] introduced a bibliographic network representation (BNR) model that integrates the bibliographic network structures and

content of different entities (authors, papers, and venues) for efficient recommendation. Liu et al. [19] presented a keyword-driven and popularity-aware paper recommendation method based on an undirected paper citation graph. Pan et al. [20] constructed a heterogeneous graph to represent citation and content information in papers and then applied a graph-based similarity learning algorithm for the paper recommendation task. Hao et al. [21] proposed a method based on the periodic interests of authors and the academic graph network structure to obtain as much valid information as possible for paper recommendation. However, these graph-based methods also have some limitations. For example, processing large-scale graphs may be constrained by computational and storage resources. The complexity of graph structures can lead to decreased algorithmic efficiency. The quality and completeness of graph data may affect the accuracy of recommendation results, and some useful information may not be fully utilized.

3. Approach of This Paper

The section outlines the proposed Multi-Feature-Enhanced Academic Paper Recommendation Model with Knowledge Graph (MFPRKG). The model is structured around the following components: the Paper-Embedding Representation Module (PERM), the User-Embedding Representation Module (UERM), and the Recommendation Module (RM), as depicted in Figure 1. The main work performed by this study to achieve multi-feature enhancement includes the following: temporal information acquisition and utilization, text vector representation and utilization, and knowledge graph information acquisition and utilization.

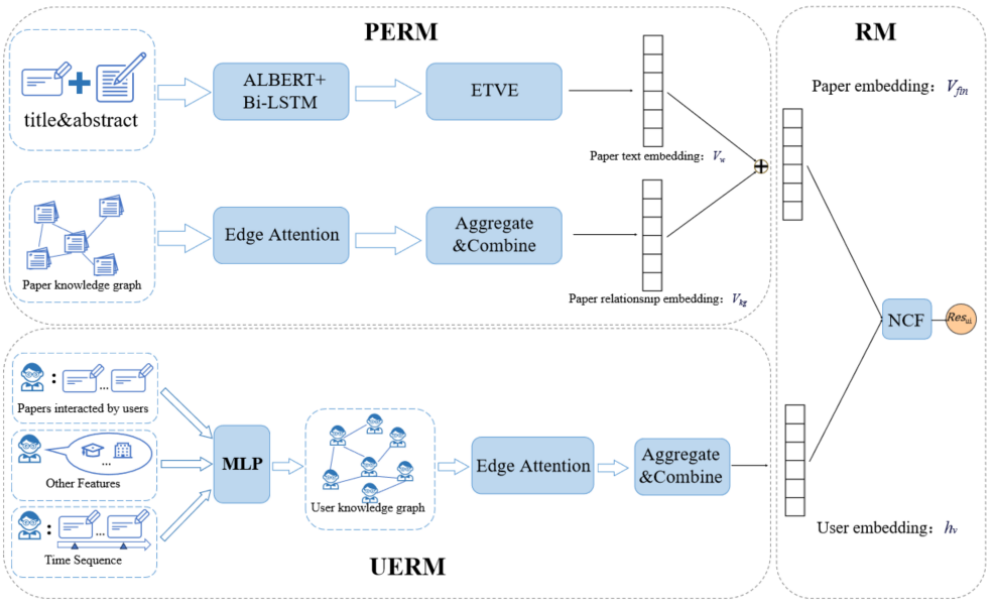


Figure 1. MFPRKG model.

In the Paper-Embedding Module, the text embeddings of the papers' titles and abstracts are first processed using ALBERT and Bi-LSTM. These processed sentence embeddings are then input into the ETVE module, where they are adjusted based on their importance to the paper. Combined with the sentence position embeddings and the paper's temporal information, the final paper text embeddings are obtained. A knowledge graph of papers is constructed from the dataset, and edge information from this graph is extracted for use in the node aggregation formula. Additionally, other information from the knowledge graph is combined to aggregate and update the initial nodes, resulting in the final paper relationship embedding. This paper text embedding and paper relationship embedding are concatenated to obtain the final academic paper embedding.

In the User-Embedding Module, a user knowledge graph is constructed from the dataset. Interactions with papers by users are transformed into embeddings, user temporal information is transformed into embeddings, and some unused user attributes are also transformed into embeddings. These are input into the multi-layer perceptron (MLP) module for fusion and serve as the initial node embeddings for the user knowledge graph. Using a method similar to that in the paper knowledge graph, the user embeddings are obtained. Finally, the paper embeddings and user embeddings are jointly input into the neural collaborative filtering (NCF) model to obtain the recommendation results.

3.1. Acquisition of Temporal Information

Temporal information not only reveals the changes in the influence of papers over time but also reflects the dynamic evolution of user interests [22]. The dataset used in this paper provides temporal information, such as the publication date of each paper and the citation dates, which form the foundation for analyzing the research directions and interest trajectories of users. With these data, a time series can be constructed that reflects the shifts in users' research focus and the changes in the influence of papers.

Regarding the temporal information of papers, this paper selects the "citations in Year", "cumulative Citations", and "citation growth rate" as features because they offer intuitive indicators of the novelty, research trends, and academic value of the papers. Assuming there is a paper with the following example of temporal information, Table 1 below illustrates the yearly data from the year of publication of 2015 to 2020.

Table 1. Example of paper timing information.

Specific Time	Year	Citations in Year	Cumulative Citations	Citation Growth Rate
2015	1	2	2	/
2016	2	1	3	−50%
2017	3	7	10	+600%
2018	4	5	15	−28.57%
2019	5	3	18	−40%
2020	6	5	23	+66.67%

After obtaining the features, to enhance the training efficiency and performance of the model, it is common to perform standardization on numerical features. These standardization steps ensure that the feature values are on a unified scale, which helps the model better capture the relationships between features. In this paper, the "Citations in Year" and "Cumulative Citations" were subjected to min–max scaling, which scales the feature values to a specified range, typically [0, 1]. For the "Citation Growth Rate", Z-score standardization was used, which transforms the feature values into a distribution with a mean of 0 and a standard deviation of 1. After the standardization of the data, the feature values are converted into vectors. For example, for the data of the sample paper in Table 1 for the year 2017, their feature vector is $Feature_{2017}$ = [Citations in Year 2017, Cumulative Citations 2017, Citation Growth Rate 2017]. Assuming that, after standardization, these values are 0.5, 0.3, and 1.5, respectively, then $Feature_{2017}$ = [0.5, 0.3, 1.5]. Subsequently, Feature2017 is transformed into a feature vector $FeatureVector_{2017}$ through an embedding layer, and by combining the feature vectors for each year into a sequence, a comprehensive temporal representation for each paper is obtained. For the sample paper, its sequence information is illustrated in Equation (1):

$$Sequence = [FeatureVector_{2015}, FeatureVector_{2016}, \ldots, FeatureVector_{2020}] \quad (1)$$

For users, this paper transforms the sequence of user–paper interactions (such as publishing, citing, and co-authoring) into a sequence of temporal feature vectors. For each time step t, the user's interaction with papers can be represented as $j_t = [j_{t,1}, j_{t,2}, \ldots, j_{t,n}]$,

where j_t is the i-th feature in the feature vector at time step t, transformed into a feature vector sequence through an embedding layer. A bidirectional recurrent neural network (Bi-RNN) [23] is then utilized to process these feature vector sequences. The forward propagation of the Bi-RNN helps understand users' historical behavior patterns, while the backward propagation helps grasp changes in users' interests. This capability is crucial for understanding long-term dependencies and contextual information within the sequence. In academic paper recommendation systems, the Bi-RNN can assist the model in comprehending the evolution of user behavior. For instance, a user's research focus may gradually develop and change over time, or their interest in a particular research topic may intensify or diminish due to the latest academic developments.

The advantage of the bidirectional recurrent neural network (Bi-RNN) lies in its capability to process sequential data, analyzing both historical user behavior and predicting future trends simultaneously. This makes Bi-RNN particularly adept at capturing the evolution of user interests, providing strong support for delivering accurate academic paper recommendations. The size and number of hidden layers affect the Bi-RNN model's complexity and capacity, while the learning rate and choice of optimizer impact the training dynamics and convergence speed. Considering experimental outcomes and time costs, the hidden layer size is set to 256 with two layers, a learning rate of 0.001, and the Adam optimizer is selected.

The hidden state h_t of the Bi-RNN can be obtained through Equation (2), where h_{t-1} and h_{t+1} are the hidden states at the preceding and succeeding time points, respectively:

$$h_t = \text{Bi-RNN}(j_t, h_{t-1}, h_{t+1}) \tag{2}$$

To highlight the contributions of significant time points in the sequence and to capture the user's recent interests, an attention score a_t is first defined for each time step t. This score reflects the relative importance of the interaction at that time point compared to others. The attention score can be obtained through Equation (3):

$$a_t = \text{softmax}\left(\frac{\exp(\mathrm{v}^{\mathrm{T}} \tanh(W_a h_t + b_a))}{\sum_k \exp(\mathrm{v}^{\mathrm{T}} \tanh(W_a h_k + b_a))}\right) \tag{3}$$

where v is the weight vector used to calculate the attention weights, W_a and b_a are learnable parameters, and h_t is the hidden state vector at time point t. Next, a time-sensitive weight w_t is introduced to adjust the attention scores for each time point, allowing the model to place greater emphasis on recent interactions, that is, the user's most current points of interest. The time-sensitive weight can be obtained through Equation (4):

$$w_t = \lambda e^{-\alpha(T-t)} \tag{4}$$

In the aforementioned equation, λ is a positive constant used to adjust the magnitude of the weights, α is a hyperparameter that controls the rate of temporal decay, and T represents the total number of time points in the sequence. Ultimately, the user's temporal sequence vector S_u is computed using the weighted attention scores and time-sensitive weights, as shown in Equation (5). The symbol $\odot$ in the equation denotes element-wise multiplication:

$$S_u = \sum_t h_t \odot a_t \odot w_t \tag{5}$$

Through this approach, the model can learn the importance of each interaction and place greater emphasis on the user's recent interests. The method that combines the attention mechanism with time-sensitive weights allows the model to focus more on the user's short-term interests while retaining the contextual information of long-term interactions.

3.2. Text Vector Representation

In this paper, the ALBERT [24] model is employed to process the titles and abstracts of academic papers to generate vector representations of words. ALBERT (A Lite BERT) is a lightweight variant of the BERT [25] model that leverages the advantages of the Transformer [26] model. It optimizes the model structure through parameter sharing and the compression of embedding layers, thereby enhancing computational efficiency. Compared to traditional word embedding methods such as Word2Vec [27] and GloVe [28], ALBERT is better at utilizing contextual information to provide richer and more accurate semantic representations. This capability gives ALBERT a significant advantage in handling phenomena of polysemy and capturing the deep semantics of text. Subsequently, the word vectors obtained from the ALBERT model are input into a bidirectional long short-term memory (Bi-LSTM) [29] network for processing. The Bi-LSTM network is capable of capturing broader contextual information between words, consisting of forward and backward LSTM layers that capture contextual information in both directions.

The forward hidden state $h_{f,t}$ is calculated as follows:

$$h_{f,t} = LSTM(x_{f,t}, h_{f,t-1}) \tag{6}$$

The backward hidden state $h_{b,t}$ is calculated as follows:

$$h_{b,t} = LSTM(x_{b,t}, h_{b,t-1}) \tag{7}$$

Ultimately, the forward hidden state and the backward hidden state are concatenated to obtain the final hidden state h_t.

$$h_t = [h_{f,t}, h_{b,t}] \tag{8}$$

In the formula above, "LSTM" represents the function of the bidirectional long short-term memory (Bi-LSTM) unit, and its specific form is as follows:

$$LSTM(x, h) = i \odot \tanh(W_i \cdot [h, x] + b_i) + f \odot c_{t-1} + g \odot c_t \tag{9}$$

$$c_t = f \odot c_{t-1} + i \odot \tanh(W_c \cdot [h, x] + b_c) \tag{10}$$

The i, f, and g in the aforementioned formulas represent the outputs of the gate control units, where the symbol $\odot$ denotes element-wise multiplication. W_i, W_c, b_i, and b_c are the weight matrices and bias terms, respectively. After processing through the bidirectional LSTM module, this model obtains title and abstract vectors that encapsulate the hidden semantic relationships between words.

Since titles play a crucial role in a paper, this paper further enhances the processing of the title portion. To ensure a more representative paper vector, a self-attention mechanism is introduced to better capture the relevance of various words in the title sequence. Each word in the title is assigned a weight, and the title vector representation is adjusted according to these weights to obtain a weighted word vector W_h and a new title vector H'_h. The calculation is as follows:

$$W_h = softmax(\tanh(W_q H_h + b_q)) \tag{11}$$

$$H'_h = \sum H_h \times W_h \tag{12}$$

Inspired by the vision transformer (ViT) model [30], this paper segments the abstracts of papers into independent sentences and converts each sentence into a vector representation. By breaking down the abstract into sentence-level representations, each sentence vector encapsulates distinct semantic information. This approach better captures the semantic content of each sentence in the abstract, enhancing the model's ability to understand the semantics of the abstract. It also allows for more effective modeling within the encoder, resulting in a more representative and semantically rich paper text vector. Compared to

treating the entire abstract as a single entity for model input, this method, which combines attention mechanisms at the sentence level, enables the model to more subtly comprehend the structure and content of the abstract.

After processing the abstract with ALBERT and Bi-LSTM, the grammatical and semantic information within the abstract is accurately captured. The approach involves averaging the vectors of each word within every sentence of the abstract to obtain a vector representation for that sentence. Assuming there is a sentence containing the words m_1, m_2, m_3, ..., m_n, with each word corresponding to the vectors v_1, v_2, v_3, ..., v_n, the vector representation S_i of the sentence is expressed by Equation (13).

$$S_i = \frac{1}{n}\sum_{i=1}^{n} v_i \tag{13}$$

Subsequently, these sentence vectors are fed into an improved Enhanced Temporal-Aware Embedding (ETVE) module, which is depicted in Figure 2. This module is an enhancement derived from the decoder of the Transformer model, which ultimately outputs the text vector V_w of the paper.

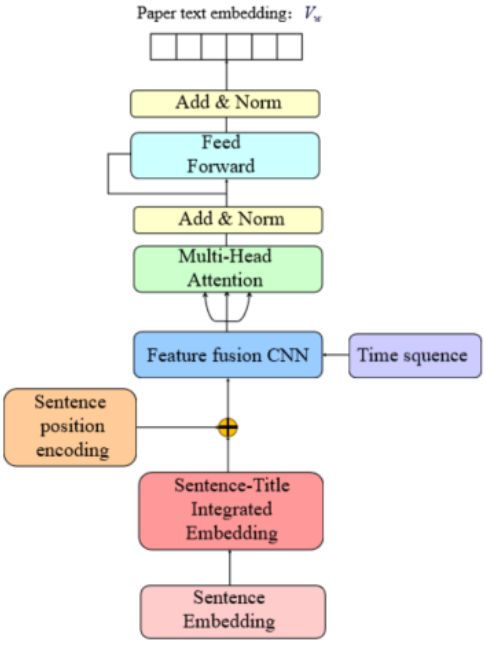


Figure 2. ETVE module.

To further enrich the representation of these sentence vectors, this paper incorporates three additional types of information: the importance of the sentence relative to the title, positional information, and the temporal information obtained in Section 2.1.

The importance of a sentence relative to the title can be obtained using Equation (14). In this formula, S_i represents the vector of the i-th sentence, T denotes the title vector, and the symbol $\cdot$ signifies the dot product operation. The softmax function normalizes the scores into attention weights, allowing us to calculate the attention weight α_i that the sentence has with respect to the title. Subsequently, each sentence vector is multiplied by its corresponding attention weight to produce a sentence vector that integrates the importance of the sentence relative to the title. These sentence vectors, after being combined with positional information vectors representing the sentences' locations within the abstract, are then input into a convolutional neural network (CNN) feature mixing module. CNNs are capable of handling high-dimensional data by effectively reducing the dimensionality of the feature space while preserving critical information. This is particularly useful for integrating complex features from different sources. In this way, CNNs not only enhance the efficiency

of feature fusion but also strengthen the recommendation system's ability to capture user interests and textual information from papers. Taking into account experimental results and computational costs, this paper selects an initial number of 64 convolutional kernels for the CNN to capture various features, with the activation function being ReLU to introduce nonlinearity and a dropout rate of 0.5 to help reduce overfitting.

$$\alpha_i = softmax(S_i \cdot T^T) \tag{14}$$

Positional information is implemented by adding positional encoding to help the model understand the relative positioning of sentences within the abstract. In an abstract, sentences at different locations hold varying levels of representativeness for the entire article. For instance, the beginning of most abstracts is typically used to introduce background information, while the end often focuses on experimental results, and the middle section is more indicative of the article's thematic ideas. Positional encoding can be calculated using the following formula:

$$PE_{(pos,2i)} = \sin(\frac{pos}{10,000^{2i/d_{model}}}) \tag{15}$$

$$PE_{(pos,2i+1)} = \cos(\frac{pos}{10,000^{2i/d_{model}}}) \tag{16}$$

where pos indicates the position of the sentence within the abstract, i represents the index of the dimension for the positional encoding, and d_{model} denotes the dimensionality of the model.

Regarding temporal information, this paper takes into account the temporal information sequence (*Sequence*) obtained in Section 2.1, integrating *Sequence* with the previously derived information into a convolutional neural network (CNN) feature mixing module. This approach achieves a deep integration of textual and temporal information.

With the structure depicted in Figure 2, the Enhanced Temporal-Aware Embedding (ETVE) module is capable of capturing the complex semantic relationships within textual data. Additionally, it can identify and leverage the evolving influence and trends of papers over time. This capability enables the recommendation system to more accurately pinpoint the research interests of users.

3.3. Knowledge Graph Information

3.3.1. Knowledge Graph Construction

This section of the paper focuses on constructing a knowledge graph from the dataset to capture the relational information between papers and users, as well as to enhance semantic expression. This paper treats papers and users as entity nodes, constructing separate knowledge graphs for papers and users based on various relationships between these entities as edges. For the paper knowledge graph, the focus is on analyzing citation relationships, research topics, and publication information. For the user knowledge graph, the emphasis is on capturing the evolution of user interests, publication history, and citation behavior. The knowledge graph enriches the recommendation system with contextual and associative information, allowing for a deeper understanding of paper content, a precise identification of user interests, and the discovery of direct and indirect connections between entities.

The knowledge graph is defined as $G = \{(h, r, t) \mid h, t \in E, r \in R\}$, where each triplet denotes a relationship r between the head entity h and the tail entity t. E represents the set of entities, and R represents the set of relationships. Initially, the TransR [31] model is utilized to process the complex information within the knowledge graph. TransR addresses complex relationships and entity polysemy in knowledge graphs by assigning separate vector spaces to entities and relationships and using transformation matrices, which improves the accuracy of the knowledge graph.

To construct the paper and user knowledge graphs, this paper first obtains initial vectors from the knowledge graph. For users, as illustrated in the UERM module in Figure 1,

an initial user vector is derived by attention-weighting the papers interacted with by the user and calculating the average of these weighted paper vectors. This vector provides a comprehensive reflection of the user's interests. Following the method in Section 2.1, user temporal information is integrated by analyzing the user interaction sequence to capture the evolution of their interests and behavioral patterns. This temporal information offers a dynamic perspective, enabling the recommendation system to better understand the user's changing needs over time. Additionally, other user attributes, such as research fields and institutions, are considered and converted into embedding vectors through an embedding layer, further enriching the semantic content of the user vector. Finally, by combining the initial vector, the temporal information representation S_u, and other attribute embedding vectors with a multi-layer perceptron (MLP) module, this paper achieves a more comprehensive and refined initial node vector for the user knowledge graph. For papers, the initial node vector in the paper knowledge graph is obtained by multiplying each sentence vector by its corresponding importance weight, derived from the ALBERT and LSTM processing of titles and abstracts, and then summing these weighted sentence vectors.

The initial relationship vectors are derived, as described in Section 3.3.1, by extracting the feature information of nodes and edges from the knowledge graph, along with their attributes. Ultimately, using the formula presented in Section 3.3.2, referred to as Equation (20), the edge vector f_e is obtained, which represents the initial relationship vector r within the knowledge graph.

Given the initial entity vector e and the initial relationship vector r, the entity vectors projected via the TransR method are denoted as e', and the relationship vector as r', with the projection matrix M_r. TransR learns the embedding vectors for entities and relationships by optimizing a principle based on the translational algorithm's fundamental premise ($e'_h + r \approx e'_t$), which establishes relationships through projection. The projected paper entity vector representation is expressed as $e' = e \cdot M_r$. The update rule for the relationship vector is $r' = r$. Its credibility formula is as follows:

$$g(h, r, t) = \| e'_h + r - e'_t \| \tag{17}$$

After utilizing the TransR model to learn the representation vectors of entities and relationships, thereby mapping entities and relationships into a continuous vector space, the scope of collected and processed information is extended from a single layer of the knowledge graph to multiple layers. This extension aims to obtain a richer vector representation. Specifically, the information of neighboring nodes can be aggregated layer by layer, resulting in the L-order vector representation for each node. The L-order user vector representation for an entity aggregates the information of neighboring entities within L-1 hops.

The node aggregation update formula is as follows:

$$h_v^{(l)} = Aggregate(\left\{ h_u^{(l-1)} : u \in N(v) \right\}) \tag{18}$$

The node update formula is as follows:

$$h_v^{(l)} = Combine(h_u^{(l-1)}, h_u^{(l)}) \tag{19}$$

$h_v^{(l)}$ represents the vector representation of node v at layer l, and $N(v)$ denotes the set of neighboring nodes of node v. The aggregate function is used to aggregate information from neighboring nodes. This paper employs an approach that utilizes edge attention weights to update the features of the nodes, with the specific formula provided in Section 3.3.2, Equation (22). The combine function serves to integrate information from the current layer with that of the previous layer. This paper utilizes a fully connected layer for this purpose, which effectively consolidates information from the current and previous layers to better capture the relationships between nodes.

3.3.2. Utilization of Edge Information

In academic paper recommendation systems, the edge information within the knowledge graph plays a crucial role. It reveals the complex patterns of interaction between academic entities through relationships such as citations between papers and author collaboration networks. This edge information not only enriches the semantics of the paper content but also has a significant impact on understanding user interests and enhancing the performance of the recommendation system. However, many existing recommendation systems have not fully leveraged this resource.

Deep mining of edge information can evaluate the influence and academic value of papers, thereby improving the accuracy and personalization level of recommendations. It also helps to explore users' potential interests and effectively mitigates the cold start problem. Therefore, in the design of this academic paper recommendation system, by mining and integrating the edge information of the knowledge graph, the relevance of the recommendation results and user satisfaction are enhanced.

Firstly, it is necessary to extract the feature information of nodes and edges from the knowledge graph constructed in Section 3.3.1. For each node (e.g., a paper or a user), collect its attributes (such as the field of the paper, the user's workplace, etc.). For each edge (e.g., citation relationships between papers, collaborations between authors, etc.), extract its attributes (such as the number of citations, the strength of collaboration, etc.). Assuming in this knowledge graph that each edge e connects two nodes u and v and is associated with a set of attributes $\{a_1, a_2, \ldots, a_n\}$, then the edge vector f_e can be constructed using Equation (20):

$$f_e = \text{integrate}(\Phi(a_1), \Phi(a_2), \ldots, \Phi(a_n)) \tag{20}$$

In the aforementioned equation, Φ represents the process of creating an embedding matrix for each type of attribute, where the number of rows is equal to the count of attributes, and the number of columns is the dimension of the embedding vector. This matrix is initially filled with simple pretrained embeddings and then refined by training a multi-layer perceptron (MLP) to adjust the embedding matrix. This results in obtaining the embeddings for the respective attributes. The term "integrate" refers to a self-attention layer that takes a collection of attribute embeddings as input, determines the importance of each attribute, normalizes these importance scores, and then performs a weighted combination to ultimately output the integrated edge vector f_e.

Subsequently, to capture the importance of edges in the recommendation system, an edge attention mechanism is introduced. This mechanism adjusts the attention weights between nodes based on the features of the edges. Specifically, for a neighbor node v of node u, the attention weight α_{uv} of edge e_{uv} can be calculated using Equation (21):

$$\alpha_{uv} = \text{softmax}\left(\frac{\exp(f_e^T \mathrm{W}_e)}{\sum_{w \in N(u)} \exp(f_{e_{uw}}^T \mathrm{W}_e)}\right) \tag{21}$$

where W_e is a learnable weight matrix used to transform the edge feature vector, and $N(v)$ is the set of neighboring nodes of node u. By leveraging the edge attention weights, the features of the nodes can be updated. The knowledge graph node update formula in this paper, which incorporates edge information, is shown in Equation (22), and it is also the specific implementation of Equation (18), as discussed in Section 3.3.1. For a node u, its updated feature $h_u^{(l+1)}$ can be calculated by aggregating the features of its neighboring nodes v as well as the associated edge information:

$$h_u^{(l+1)} = \sigma\Big(\sum_{v \in N(u)} \alpha_{uv} \mathrm{W}_v \mathrm{h}_v^{(l)} \oplus f_e\Big) \tag{22}$$

where W_v is the node feature transformation matrix. $h_v^{(l)}$ represents the feature vector of node v at layer l. $\oplus$ denotes vector concatenation, and σ is the activation function ReLU

(rectified linear unit). Ultimately, the user vector h_v and the paper relation vector V_{kg} are obtained.

3.4. Recommendation Module

The recommendation module in this paper is primarily an improvement based on the NCF [32] model. The core idea of the NCF recommendation model is to represent users and papers as vectors and to use a neural network to learn the interactions between them. The NCF recommendation model can be represented by the following formula:

$$Res_{ui} = MLP(u_i, v_j) + GMF(u_i, v_j) \tag{23}$$

where Res_{ui} represents the predicted rating of user u for paper i, then u_i and v_j, respectively, denote the feature vectors of user i and paper j.

The original NCF model only utilized the paper's identifier information to obtain the paper vectors, while the user vectors were randomly initialized. This led to an inadequate expression of information for both users and papers. Therefore, this paper introduces a multi-feature-enhanced representation for both users and papers, which is input into the improved NCF model.

In the preceding sections, the paper text vector V_w and the paper relationship vector V_{kg} were obtained separately. By concatenating these two vectors and passing them through a fully connected layer, the model derived the final paper vector V_{fin}. Ultimately, the final user vector h_v and the final paper vector V_{fin} are used as inputs to the improved NCF model, yielding the final result r_{ui}. The loss function is defined by Equation (24), and the Adam optimizer is employed to learn the model parameters:

$$L = -\sum_{(i,j)\in\gamma\cup\gamma^-} y_{ij}\log(Res_{ui}) + (1 - y_{ij})\log(1 - Res_{ui}) \tag{24}$$

In Formula (24), y_{ij} equal to 1 indicates that user u_i has interacted with paper v_j, while y_{ij} equal to 0 indicates that user u_i has not interacted with paper v_j. The symbol γ represents the set of positive samples, and γ^- represents the set of negative samples [33].

4. Experiments

4.1. Experimental Dataset

This paper utilizes the DBLP-v11 dataset, a comprehensive and in-depth academic literature resource that focuses on the field of computer science and various related disciplines. The dataset meticulously records a wide array of basic information about academic papers, such as article titles, lists of authors, publication years, associated conferences or journals, etc. Moreover, it meticulously captures and presents the citation network between papers, forming an intricate academic link graph. With this graph and other related information, a knowledge graph can be constructed, enabling a series of in-depth analyses. In addition, the DBLP-v11 dataset contains a wealth of temporal sequence data, which meticulously record the publication time of each paper and its subsequent citation timestamps. This temporal information provides solid data support for the research presented in this paper, playing a crucial role in analyzing the impact of academic papers, exploring the evolution of academic research trends, and constructing dynamic temporal recommendation systems.

After selecting this dataset, the most recent five years of data were extracted, followed by preprocessing of the data. This included, for instance, removing data from papers that lacked certain important fields (such as title, authors, publication year, publishing organization, abstract, cited papers, etc.), eliminating data from authors' historical behavior that had no publication or citation records, and performing data transformation, normalization, and integration operations. The resulting dataset is presented in Table 2.

Table 2. DBLP-v11 experimental dataset.

Data Type	Quantity
Paper	716,549
Author	976,544
Year	5
Publishing Institution	4023
Author's Affiliated Institution	571,659

4.2. Baseline Models

CF: Traditional collaborative filtering is a recommendation algorithm based on user behavior data that recommends items that users may be interested in by analyzing the similarities between users and items.

NCF: The neural collaborative filtering model employs a neural network structure to represent users and items as vectors and learns the implicit features of their interactions to output predicted scores.

DeepCoNN [34]: DeepCoNN is a deep neural network model specifically designed to predict user ratings for items. It achieves this by jointly considering domain and language features between users and items.

DKN [35]: DKN is a knowledge graph-based deep learning recommendation algorithm that leverages semantic information from knowledge graphs to understand user needs and item content, thereby better comprehending user interests and behaviors.

PNA [36]: PNA is an innovative graph neural network model that significantly enhances the model's ability to capture and utilize graph-structured data by integrating multiple aggregators and scalers, thus achieving more accurate predictions and more efficient decision-making.

RGCN [37]: RGCN is a graph convolutional network capable of handling multi-relational heterogeneous graphs. It captures the complex interactive relationships between entities in the graph by considering the features of nodes on different types of edges during the node feature update process.

EGAT [38]: EGAT is a graph neural network architecture that improves the learning ability on graph data by integrating both node and edge features, which is suitable for scenarios where edge features significantly impact the outcome of the task.

4.3. Evaluation Metrics

This paper employs the leave-one-out method as the validation approach. It is a form of cross-validation where the dataset is divided into n subsets, with each subset used as the test set once while the remaining $n - 1$ subsets serve as the training set for training and validation. In the experiments in this paper, for each user, the most recent paper interacted with is taken as the test set, and the remaining papers interacted with constitute the training set. This method helps prevent overfitting and underfitting and can more accurately assess the model's generalization capability.

For each user u_i, 99 papers not interacted with by that user are randomly selected and ranked together with the test paper to determine the ranking position of the test paper among these 100 papers, thereby evaluating the recommendations' effectiveness. This paper selects the top 10 in the ranking as the recommendation list for evaluation, allowing a certain margin for error [33].

Regarding this recommendation list, hit ratio (HR) and normalized discounted cumulative gain (NDCG) are used as metrics to judge the recommendation performance:

$$HR = \frac{1}{N} \sum_{i=1}^{N} hits(i) \tag{25}$$

In Equation (25), N represents the total number of users, and $hits(i)$ indicates whether the test paper of the i-th user is present in the recommendation list. If the test paper is

in the list, $hits(i)$ is 1; otherwise, it is 0. A larger HR (hit ratio) value indicates a higher recommendation accuracy rate of the model.

$$NDCG = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\log_2(p_i+1)} \tag{26}$$

In Equation (26), N represents the total number of users, and p_i represents the position of the i-th user's test paper in the top 10 recommendation list. If the test paper is not included in the recommendation list, p_i tends toward infinity. Therefore, a larger NDCG (normalized discounted cumulative gain) value indicates better recommendation performance by the model.

To calculate precision and recall, this paper collects and constructs positive samples from the papers that the authors have actually published or cited and randomly selects negative samples from research areas where the user has not shown interest.

$$Precision = \frac{TP}{TP+FP} \tag{27}$$

Precision refers to the proportion of papers in the recommendation list that the user is genuinely interested in, relative to the total number of papers in the recommendation list. The formula is given by Equation (27), where TP (true positives) is the number of papers in the recommendation list that the user is interested in, and FP (false positives) is the number of papers in the recommendation list that the user is not interested in.

$$Recall = \frac{TP}{TP+FN} \tag{28}$$

Recall refers to the proportion of papers in the recommendation list that the user is genuinely interested in, relative to the total number of papers the user is interested in. The formula is given by Equation (28), where FN (false negatives) is the number of papers that the user is interested in but were not recommended in the recommendation list.

4.4. Experimental Results and Analysis

Experiments were conducted on the DBLP-v11 dataset with the following parameters: the vector dimension D = 512, the number of neighbors extracted per layer in the knowledge graph M = 16, and the number of aggregation layers in the knowledge graph L = 2. The experimental results are presented in Table 3 below.

Table 3. The comparative experimental results.

Model	HR	NDCG	Prec@10	Rec@10
CF	0.386	0.245	0.197	0.238
NCF	0.655	0.390	0.355	0.372
DeepCoNN	0.714	0.431	0.423	0.455
DKN	0.757	0.453	0.484	0.522
RGCN	0.769	0.459	0.512	0.526
EGAT	0.786	0.477	0.525	0.565
PNA	0.791	0.481	0.534	0.551
MFPRKG	0.812	0.497	0.553	0.589

The results indicate that the model presented in this paper outperforms other baseline methods based on the selected evaluation metrics. The CF method is susceptible to the adverse effects of data sparsity and only considers the similarity between users and papers without utilizing richer feature information, leading to very poor recommendation performance. The NCF model, which is based on CF, shows a significant improvement, likely due to the introduction of neural networks to learn the interaction features between users and papers, thereby mitigating the impact of data sparsity. However, the NCF

model only considers implicit feedback information and does not fully utilize the textual information of papers. In contrast, the DeepCoNN model, which takes into account the domain and language features of users and papers, achieved a notable improvement in recommendation accuracy. The recommendation methods that incorporate knowledge graphs, such as DKN, RGCN, PNA, and EGAT, consider the connections between various entities and integrate higher-order information into the recommendation process, resulting in a considerable enhancement in recommendation performance. DKN leverages semantic information from knowledge graphs to deeply understand user needs and paper content, while RGCN further improves the accuracy of recommendations by dealing with complex interactions in multi-relational heterogeneous graphs. The PNA model, by integrating multiple aggregators and scalers, significantly enhances the capture and utilization of graph-structured data, demonstrating excellent performance in recommendation tasks. The EGAT model, by integrating both node and edge features, effectively addresses the impact of edge features on task results, enhancing the personalization of recommendations.

The MFPRKG model achieves better performance than other baseline methods in evaluation metrics. The MFPRKG model comprehensively utilizes various information sources and advanced recommendation algorithms to more accurately capture users' implicit and explicit feedback, the deep semantic information of paper content, and the interconnections between various entities, ultimately resulting in higher accuracy and personalized recommendation outcomes.

The selection of hyperparameters is crucial to the performance of the recommendation model. This paper experimentally investigates the impact of adjusting the vector dimension D for users and papers, the number of aggregation layers L in the knowledge graph, and the number of neighbors M extracted per layer on the recommendation effect. Experiments were conducted to evaluate the recommendation performance under different hyperparameter values, and the results are depicted in Figures 3 and 4.

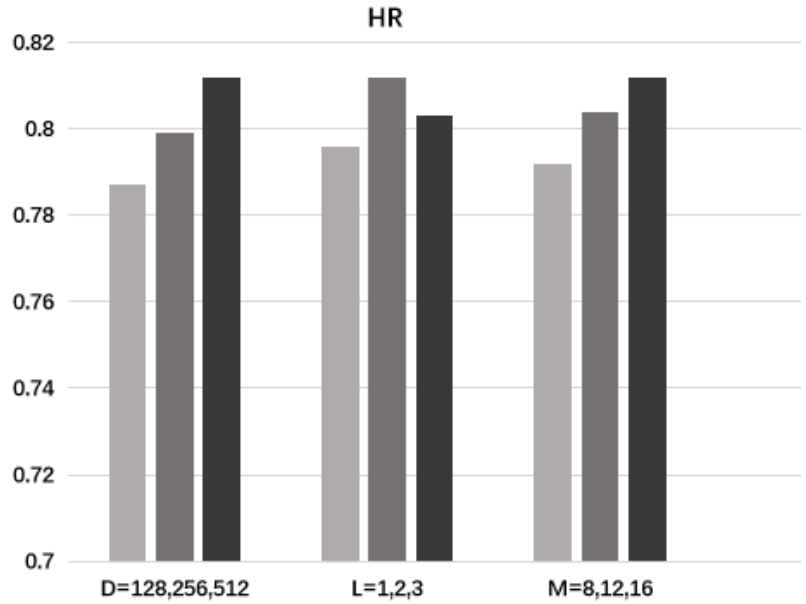


Figure 3. The impact of different parameters on HR.

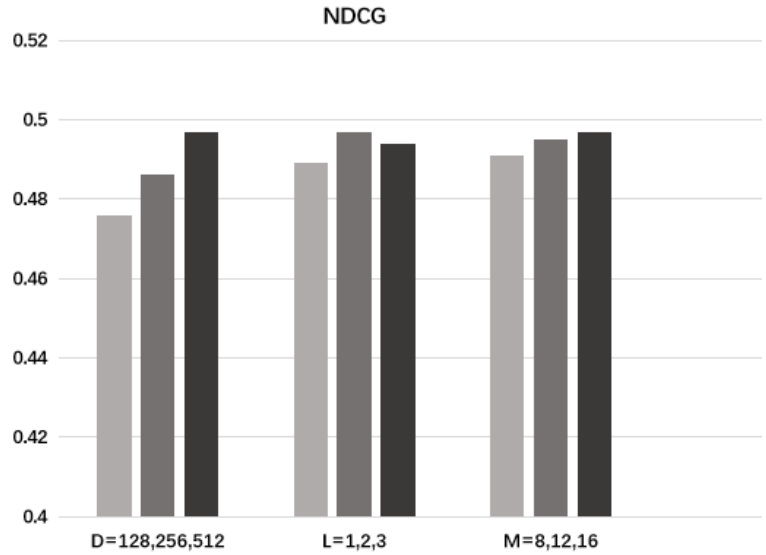


Figure 4. The impact of different parameters on NDCG.

Firstly, experiments were conducted with different values for the vector dimension D between users and papers, specifically selecting D = 128, D = 256, and D = 512. As the value of D increased, the performance of the recommendation model gradually improved. A lower dimension may lead to information loss and insufficient model expressiveness, while a higher dimension might increase the model's complexity. After considering the trade-offs, D was set to 512 to achieve better recommendation results.

Next, the impact of the number of knowledge graph aggregation layers L on model performance was studied, with considerations of L = 1, L = 2, and L = 3. The results showed that the model performed best when L was set to 2. A lower number of layers might limit the model's consideration of multi-layered information interactions, while a higher number of layers, although increasing the model's complexity, could lead to overfitting. Therefore, L was set to 2.

Finally, the effect of the number of neighbors M extracted per layer in the knowledge graph on the recommendation effect was explored, with experiments conducted using M = 8, M = 12, and M = 16. The experimental results indicated that as the value of M increased, the performance of the recommendation model also improved gradually. A smaller number of neighbors might restrict the model's perception range around a node, while a larger number of neighbors allows for a more comprehensive consideration of the node's neighboring information. Based on the experimental results and time considerations, M was set to 16 in this paper.

To verify the performance of the user vector representation and paper vector representation modules proposed in this paper within the model, comparisons were made between this paper's model and its variants. In the experiments, MFPRKG-P represents the model that includes only the paper vector representation module, while the user vector is obtained through random processing. MFPRKG-U, on the other hand, represents the model that includes the user vector representation module, but the paper vector is obtained using the Word2Vec model. The experimental results of the MFPRKG model and its variants on the dataset are presented in Table 4.

Table 4. Results of ablation experiment.

Evaluation Metrics	HR	NDCG	Prec@10	Rec@10
MFPRKG-U	0.739	0.440	0.492	0.523
MFPRKG-P	0.716	0.427	0.478	0.507
MFPRKG	0.812	0.497	0.553	0.589

From the ablation study results, it can be observed that the MFPRKG-U model performs better than the MFPRKG-P model in terms of evaluation metrics. This may be attributed to the fact that while the MFPRKG-U model only includes the user vector representation module, the paper vector representation obtained through the Word2Vec model still provides a certain level of recommendation accuracy. In contrast, the MFPRKG model lacks a deeper level of processing for user features and offers little analysis of user interests and characteristics, which results in slightly lower performance. Meanwhile, in the FEPRKG model, the features of both users and papers are enhanced through a multi-feature approach; hence, it outperforms both the FEPRKG-P and FEPRKG-U models in these two metrics.

5. Conclusions

This paper aims to explore ways to enhance the accuracy and personalization of academic paper recommendation systems. It proposes a Multi-Feature-Enhanced Academic Paper Recommendation Model with Knowledge Graph and validates its effectiveness through experiments. Utilizing attention mechanisms and deep learning methods, this paper delves into mining textual and relational features of papers, integrating temporal information of publication and citation, as well as edge information from the academic citation network. Through this multidimensional feature fusion, more representative paper vectors and user vectors are obtained. These vectors are then combined with an improved neural collaborative filtering (NCF) model to produce the final recommendation results. The experimental results demonstrate that, on the DBLP-v11 dataset, the proposed model achieves improvements in evaluation metrics, showing better recommendation accuracy and personalization compared to traditional recommendation algorithms and baseline models. The ablation experiments confirm the importance of the user vector representation and paper vector representation modules within the model. The hyperparameter experiments analyze the impact of different parameters on the results. However, the model proposed in this paper still has some limitations. For instance, the model may be constrained by the dataset and the timeliness of the recommendations may not be sufficiently high. Future work will focus on further optimizing the model's design to better meet the needs of researchers.

Author Contributions: Investigation, L.W. and W.D.; methodology, L.W., W.D. and Z.C.; software, L.W.; experiment, L.W.; supervision, W.D. and Z.C.; writing—original draft, L.W.; writing—review and editing, L.W., W.D. and Z.C.; data curation, L.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kurek, J.; Latkowski, T.; Bukowski, M.; Świderski, B.; Łępicki, M.; Baranik, G.; Nowak, B.; Zakowicz, R.; Dobrakowski, Ł. Zero-shot recommendation AI models for efficient job–candidate matching in recruitment process. *Appl. Sci.* **2024**, *14*, 2601. [CrossRef]
2. Siet, S.; Peng, S.; Ilkhomjon, S.; Kang, M.; Park, D.-S. Enhancing sequence movie recommendation system using deep learning and kmeans. *Appl. Sci.* **2024**, *14*, 2505. [CrossRef]
3. Gündoğlu, E.; Kaya, M.; Daud, A. Deep learning for journal recommendation system of research papers. *Scientometrics* **2023**, *128*, 461–481. [CrossRef]
4. Bai, X.; Wang, M.; Lee, I.; Yang, Z.; Kong, X.; Xia, F. Scientific paper recommendation: A survey. *IEEE Access* **2019**, *7*, 9324–9339. [CrossRef]

5. Tanner, W.; Akbas, E.; Hasan, M. Paper recommendation based on citation relation. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3053–3059.
6. Ali, Z.; Kefalas, P.; Muhammad, K.; Ali, B.; Imran, M. Deep learning in citation recommendation models survey. *Expert Syst. Appl.* **2020**, *162*, 113790. [CrossRef]
7. Kreutz, C.K.; Schenkel, R. Scientific paper recommendation systems: A literature review of recent publications. *Int. J. Digit. Libr.* **2022**, *23*, 335–369. [CrossRef] [PubMed]
8. Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; He, Q. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3549–3568. [CrossRef]
9. Wang, F.; Zhu, H.; Srivastava, G.; Li, S.; Khosravi, M.R.; Qi, L. Robust Collaborative Filtering Recommendation with User-Item-Trust Records. *IEEE Trans. Comput. Soc. Syst.* **2021**, *9*, 986–996. [CrossRef]
10. Zhu, Y.; Xie, R.; Zhuang, F.; Ge, K.; Sun, Y.; Zhang, X.; Lin, L.; Cao, J. Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 11–15 July 2021; pp. 1167–1176.
11. Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; Xie, X. Self-supervised graph learning for recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 11–15 July 2021; pp. 726–735.
12. Jin, B.; Gao, C.; He, X.; Jin, D.; Li, Y. Multi-behavior recommendation with graph convolutional networks. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Xi'an, China, 25–30 July 2020; pp. 659–668.
13. Naak, A.; Hage, H.; Aïmeur, E. A multi-criteria collaborative filtering approach for research paper recommendation in papyres. In Proceedings of the 4th International Conference on E-Technologies: Innovation in an Open World (MCETECH), Ottawa, ON, Canada, 4–6 May 2009; pp. 25–39.
14. Liu, H.; Kong, X.; Bai, X.; Wang, W.; Bekele, T.M.; Xia, F. Context-based collaborative filtering for citation recommendation. *IEEE Access* **2015**, *3*, 1695–1703. [CrossRef]
15. Sugiyama, K.; Kan, M.Y. Exploiting potential citation papers in scholarly paper recommendation. In Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries, Indianapolis, IN, USA, 22–26 July 2013; pp. 153–162.
16. Philip, S.; Shola, P.; Ovye, A. Application of content-based approach in research paper recommendation system for a digital library. *Int. J. Adv. Comput. Sci. Appl.* **2014**, *5*. [CrossRef]
17. Ma, X.; Wang, R. Personalized scientific paper recommendation based on heterogeneous graph representation. *IEEE Access* **2019**, *7*, 79887–79894. [CrossRef]
18. Cai, X.; Zheng, Y.; Yang, L.; Dai, T.; Guo, L. Bibliographic network representation based personalized citation recommendation. *IEEE Access* **2018**, *7*, 457–467. [CrossRef]
19. Liu, H.; Kou, H.; Yan, C.; Qi, L. Keywords-driven and popularity-aware paper recommendation based on undirected paper citation graph. *Complexity* **2020.** [CrossRef]
20. Pan, L.; Dai, X.; Huang, S.; Chen, J. Academic paper recommendation based on heterogeneous graph. In Proceedings of the China National Conference on Chinese Computational Linguistics, Guangzhou, China, 13–14 November 2015; Springer International Publishing: Cham, Seitzerland, 2015; pp. 381–392.
21. Hao, L.; Liu, S.; Pan, L. Paper recommendation based on author-paper interest and graph structure. In Proceedings of the 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Dalian, China, 5–7 May 2021; pp. 256–261.
22. Fan, Z.; Liu, Z.; Zhang, J.; Xiong, Y.; Zheng, L.; Yu, P.S. Continuous-time sequential recommendation with temporal graph collaborative transformer. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Virtual Event, 1–5 November 2021; pp. 433–442.
23. Dhyani, M.; Kumar, R. An intelligent Chatbot using deep learning with Bidirectional RNN and attention model. *Mater. Today Proc.* **2021**, *34*, 817–824. [CrossRef] [PubMed]
24. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations. In Proceedings of the 8th International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020; pp. 1–14.
25. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
26. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30*; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
27. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
28. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
29. Imrana, Y.; Xiang, Y.; Ali, L.; Abdul-Rauf, Z. A Bidirectional LSTM Deep Learning Approach for Intrusion Detection. *Expert Syst. Appl.* **2021**, *185*, 115524. [CrossRef]

30. Sharir, G.; Noy, A.; Zelnik-Manor, L. An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
31. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2022.
32. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.S. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.
33. Chen, Y.M.; Li, D.X.; Yan, Y.F.; L.ü; C.J.; Chen, Z.H. Method for Recommending Academic Papers Combining Text and Implicit Feedback. *J. Chin. Comput. Syst.* **2023**, *44*, 2471–2476.
34. Zheng, L.; Noroozi, V.; Yu, P.S. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, Cambridge, UK, 6–10 February 2017.
35. Wang, H.; Zhang, F.; Xie, X.; Guo, M. DKN: Deep Knowledge-Aware Network for News Recommendation. *arXiv* **2018**, arXiv:1801.08284.
36. Corso, G.; Cavalleri, L.; Beaini, D.; Liò, P.; Veličković, P. Principal neighbourhood aggregation for graph nets. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 13260–13271.
37. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Van Den Berg, R.; Titov, I.; Welling, M. Modeling Relational Data with Graph Convolutional Networks. In Proceedings of the European Semantic Web Conference, Heraklion, Greece, 3–7 June 2018; Springer: Cham, Switzerland, 2018; pp. 593–607.
38. Chen, J.; Chen, H. Edge-Featured Graph Attention Network. *arXiv* **2021**, arXiv:2101.07671.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.