


Article

An Academic Text Recommendation Method Based on Graph Neural Network

Jie Yu *, Chenle Pan, Yaliu Li  and Junwei Wang

School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China; panchenle08@shu.edu.cn (C.P.); yabobo@shu.edu.cn (Y.L.); jun121@shu.edu.cn (J.W.)

* Correspondence: jieyu@shu.edu.cn

Abstract: Academic text recommendation, as a kind of text recommendation, has a wide range of application prospects. Predicting texts of interest to scholars in different fields based on anonymous sessions is a challenging problem. However, the existing session-based method only considers the sequential information, and pays more attention to capture the session purpose. The relationship between adjacent items in the session is not noticed. Specifically in the field of session-based text recommendation, the most important semantic relationship of text is not fully utilized. Based on the graph neural network and attention mechanism, this paper proposes a session-based text recommendation model (TXT-SR) incorporating the semantic relations, which is applied to the academic field. TXT-SR makes full use of the tightness of semantic connections between adjacent texts. We have conducted experiments on two real-life academic datasets from CiteULike. Experimental results show that TXT-SR has better effectiveness than existing session-based recommendation methods.

Keywords: session-based recommendation; text recommendation; graph neural network; attention mechanism



Citation: Yu, J.; Pan, C.; Li, Y.; Wang, J. An Academic Text Recommendation Method Based on Graph Neural Network. *Information* **2021**, *12*, 172. <https://doi.org/10.3390/info12040172>

Academic Editor: Marco Polignano

Received: 18 March 2021

Accepted: 14 April 2021

Published: 16 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 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 smart devices, people are enjoying the convenience brought by Internet big data. However, they are also driving a blowout growth of data traffic which is called information overloading [1,2]. Therefore, how to help users dig out the things that users are most interested in is a major research hotspot at present. As the recommendation system was proposed [3], it becomes a key technology to effectively solve this problem. Among them, Session-based Recommender Systems (SRs) occupy a large proportion. They usually utilize the sequence of user actions in the browser's current sessions to predict their next actions (click on an item). Session-based text recommendation is a type of session-based recommendation. For example, a news recommendation system can quickly recommend interesting news for readers, and an academic text recommendation system can help academic researchers to obtain academic articles of interest more quickly.

Deep neural networks have recently been verified to be very effective in modeling sequence data. Hidasi et al. [4] applied Recurrent Neural Networks (RNN) with Gated Recurrent Units (GRU) for session-based recommendation. Tan et al. [5] further improved this model by data augmentation and taking the shift of user behavior distributed in the input data into consideration. Li et al. [6] proposed an RNN-based encoder-decoder model (NARM), which combines main purpose and sequential behavior to get the session representation. Similar to NARM, Liu et al. [7] purposed the STAMP model which additionally emphasizes the user's current interest reflected in the last click. Wu et al. [8] applied Graph Neural Networks (GNNs) to capture complex transitions of items in the session. However, these existing session-based methods mainly take into account the user's sequential behavior and pay more attention to capturing the user's purpose in the current session. The internal relationship between items in the session is not emphasized. Specifically in the field of text recommendation, the most important inherent semantic

relationship of text is not fully utilized. To tackle this problem, this paper proposes a session-based text recommendation model (TXT-SR) adopting graph neural network and attention mechanism. First, the model uses graph neural network to capture the sequential information and the complex transitions of items. Second, the model integrates the semantic relationship between adjacent texts into the graph neural network, so that it can better maintain the purpose of the session in the training and transmission of sessions (especially in long sessions). Finally, the attention mechanism is used to better obtain the global purpose of the session.

The main contributions of this work are summarized as follows.

- We propose an innovative TXT-SR model which is an application scenario in the text field. This model not only considers the complex transformation characteristics of the items in the text session, but also takes into account the textual semantic relationship between the texts. This is an innovation that applies session-based recommendations to a new field.
- We can represent a session directly only by nodes involved in that session, without relying on the assumption that there exists a distinct latent representation of the user for each session.
- The proposed model is evaluated on two real-world datasets. Extensive experimental results show that TXT-SR outperforms the state-of-art methods and the textual semantic relation plays an important role.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the workflow of the proposed TXT-SR method. Section 4 gives the experimental analysis. Section 5 gives the conclusion of this paper.

2. Related Work

2.1. Conventional Recommendation Method

There are two kinds of conventional recommendation methods: one is the general recommendation method, and the other is the sequence recommendation method.

The general recommendation method can be divided into content-based recommendations (CB) and collaborative recommendations (CF) [9]. Content-based recommendation refers to discovering the relevance of the item based on content, and then recommending similar items to the user based on the user's previous preference records. Recommendations based on collaborative filtering are divided into User-based Collaborative Filtering (UserCF) [10] and Item-based Collaborative Filtering (ItemCF) [11]. UserCF looks for neighbors with the same preferences as the target user and generates recommendations to the target user based on the preferences of the target user's neighbors. However, ItemCF is the evaluation of items. ItemCF finds the similarity between items and recommends similar items to the user [12]. Linden et al. [13] proposed the application of online shopping. Sarwar et al. [11] analyzed various item-based recommendation algorithms. However, what is difficult for collaborative filtering is to deal with the "cold start" problem. When we do not have any data for new users, we cannot recommend items for them. In addition to the collaborative filtering algorithm, there is also a matrix factorization algorithm. Among its specific methods, Singular Value Decomposition (SVD) [14,15] is the most common. This method essentially extracts the eigenvalues from the matrix, which reduces the dimensions to better capture the expression of users and item preferences. The advantage of this method is that it can solve the matrix sparsity problem, but the cost is that storing the decomposed matrix requires a lot of memory.

Sequence recommendation algorithms are generally based on Markov chains [16], using serialized data and the last click behavior of a given user. Zimdars et al. [17] proposed a serialized recommendation model based on Markov chain and explored how to extract serialized patterns through probabilistic decision tree models to learn the user's next behavior state. The method proposed by Shani et al. [18] is called Markov Process (MDP), which aims to make recommendations in a session-based manner. The simplest MDP can be attributed to a first-order Markov chain. The next recommendation made for the user

can be simply calculated by the transition probability between items. For this kind of algorithm, it has to calculate the probability of each item being transferred to other items, and iterative calculations are carried out continuously, so the required state space will be very large.

2.2. Deep Learning-Based Recommendation Method

With the development of deep learning [19], increasingly more scholars began to set about the application of deep learning methods in the recommendation field [20–23]. According to the work of Bobadilla et al. [24], for the Matrix Factorization method, the linear dot product cannot catch the complex nonlinear relations existing among the set of hidden factors. However, neural models do not have this restriction. For another example, in order to deal with different recommendation scenarios, such as the research of Sulikowski et al. [25] on how to design a recommending interface, the multilayer perceptron could perform an accurate prediction.

The emphasis of the session-based recommendation problem lies in how to use the user's short-term interactive information data to predict the content that the user may be interested. Writing on how to observe user behavior, Sulikowski et al. [26] utilized a tool called ECPM, which was implemented as an extension for the Firefox browser to gather a rich set of e-customer behavior data. Later, Sulikowski et al. [27] again proposed two methods to observe user behavior: eye-tracking and document object model (DOM) implicit event tracking in the browser. Because the collected information has complex information and the Recurrent Neural Network (RNN) [28] is suitable for modeling sequential data, increasingly more recommendation systems are starting to build on it [29–32]. Typically, Hidasi et al. [4] proposed the GRU4REC model. Tan et al. [5], on this basis, use “data enhancement” and “popularity sampling” methods to improve the performance, finally achieving satisfactory prediction results. However, the RNN model also has two shortcomings: the first point is that sessions are usually anonymous and numerous, and the user behavior involved in session clicks is usually limited. Therefore, it is difficult to accurately estimate the representation of each user from each session to generate effective recommended content. The second point is the use of RNN to carry out the modeling, which cannot get the user's accurate representation and ignores the complex conversion characteristics between items.

As the use of the attention mechanism can better learn the relative importance of different segments of the target sequence, there has been increasing attention to this issue [33–35]. Jing et al. [6] proposed an RNN-based NARM model, which uses the last hidden state of the RNN as the sequence behavior and the attention mechanism to capture the main purpose by the previous click. Liu et al. [7] also considered the direct correlation between each historical click and the last click, and assigned dynamic weights to each item. STAMP emphasizes the user's current interest reflected in the last click, clarifies the importance of the last click, and integrates it into the recommendation system. Compared with the RNN model, the attention mechanism model can obtain the global and local connections together, and will not be limited by the sequence length for capturing long-term dependencies. The result of each step does not depend on the previous step, and can be made into a parallel mode. In addition, it has fewer parameters and lower model complexity.

In recent years, graph neural networks have become more popular in the fields of business recommendation networks [36], knowledge graphs [37], gesture recognition [38], and recommendation systems [8]. This is because, for the complex data organization in which dependencies between more than one object or activity occur, graphs can represent more accurately. Graph Convolutional Network (GCN) provides a new idea for the processing of graph structure data and combines the convolutional neural network commonly used in images in deep learning to graph data. According to it, Ullah et al. [39] enhanced two of the existing graph convolutional network models by proposing four enhancements. The Gated Graph Neural Network (GGNN) [40,41] is an improvement of GNN, which uses gated

recurrent units to calculate gradients by back-propagation through time (BPTT). In the recommendation field, Wu et al. [8] proposed a recommendation model based on GGNN. The model uses a certain method to describe the nodes. After continuous node status updates, it obtains a state that includes neighbor node information and graph topological structure characteristics, finally outputting these nodes through a specific method to obtain the desired result.

3. The Proposed Method

In this section, we will introduce the recommendation method of text based on graph neural network. Figure 1 shows the overall framework of the proposed TXT-SR method.

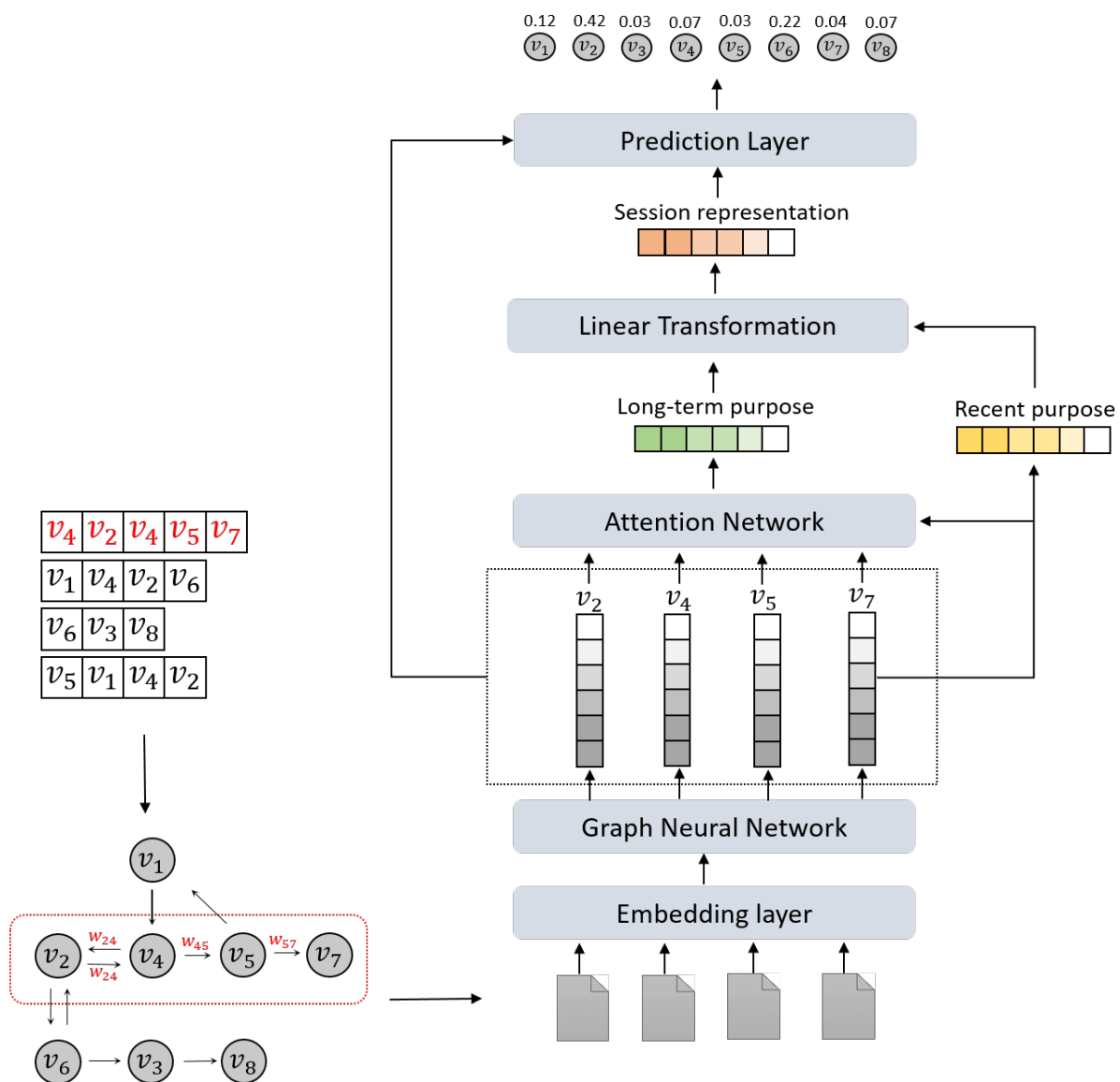


Figure 1. The framework of the proposed TXT-SR method.

First, the user's reading order can be regarded as a session, which is represented by a graph. In the graph, each node represents a single academic text, and each edge represents the reading order. After representing each text semantically, assign a weight to the graph edges. This weight is regarded as the semantic similarity of adjacent texts in the session, which can supplement the graph to a large extent. Inspired by the work of Jing et al. [6], we consider the whole feature representation of session to be composed of recent purposes

and long-term purposes. We take both of them into account. Recent purpose is represented by the trained vector of the session's last-click text. Long-term purpose is obtained by the attention mechanism, which aggregates all trained vector of each text. Finally, we combine these two purposes to gain the feature representation of a session and utilize it to compute the recommendation scores for each candidate academic text.

The details are as follows:

3.1. Notations

Let $V = \{v_1, v_2, \dots, v_m\}$ be the set of all unique texts involved in all the sessions and $s = [v_{s,1}, v_{s,2}, \dots, v_{s,n}]$ be a session sequence, where $v_{s,i} \in V$ represents the i -th clicked text in the s and the sequence of $v_{s,i}$ represents the reading order. The goal of this recommendation is to predict the next click, i.e., for the session s , the task is to predict $v_{s,n+1}$. By utilizing the session-based text recommendation, for a certain session s , we calculate probabilities \hat{y} for all optional texts. According to \hat{y} , we choose the items with top-K value as the candidate. We define the set of common definitions required to understand this paper in Table 1.

Table 1. Commonly used notations.

| Notations | Descriptions |
|----------------------|---|
| G | A graph. |
| V | The set of nodes in a graph. |
| v_i | A node $v_i \in V$. |
| E | The set of edges in a graph. |
| $v_{s,i}$ | the i -th clicked text in the s |
| $v \in \mathbb{R}^d$ | The features vector set of nodes. |
| d | The dimension of node features. |
| s_l | The recent purpose of the session. |
| s_g | The long-term purpose of the session. |
| s_h | The features vector of the whole session. |
| \hat{y} | The probabilities for optional texts. |
| \hat{z}_l | The score for each candidate text option. |

3.2. Using Graph Neural Networks to Learn the Text Feature Representation

Each session sequence s can be modeled as a directed graph $G_s = (V_s, E_s)$, in which each node represents a text $v_{s,i} \in V$. Each edge of the directed graph $(v_{s,i-1}, v_{s,i})$ represents that a user continues to read text $v_{s,i}$ after $v_{s,i-1}$. For example, a reader browses the text in the order of $v_1, v_2, v_3, v_4, v_3, v_5$, so we can model this order as a session s , here $s = [v_1, v_2, v_3, v_4, v_3, v_5]$. Then, it can be transformed into the following Figure 2 according to graph theory.

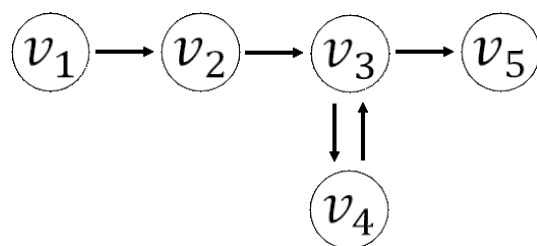


Figure 2. Converted directed graph.

As the text has inherent semantic information, this point should be paid more attention when making text recommendation. In past GNN-related work, more consideration in the graph modeling process has been given to whether there is an edge between two nodes. When updating the state of a node, more consideration is also given to the neighboring

node's own influences. The role of the edge is ignored to a large extent. However, a node may have many adjacent nodes, but not all adjacent nodes have the same impact on it. The degree of these effects can be reflected by the weight of the edge.

Therefore, in our work, in addition to the sequential information of the text reading recorded in each browsing sequence, we can also consider the semantic relationship between adjacent texts in the model, which is reflected in the assignment of each edge with weights. Thus, the information of the graph neural network can be further improved, rather than simply 0 or 1. Specifically, we should pay more attention to two similar texts in two adjacent positions, because they are more likely to be related to the theme of the session. On the contrary, the corresponding degree we focus on should be appropriately reduced. In actual work, we normalize the edges' weight.

Here, we use the "SIF" method proposed by Arora et al. [42] to carry out the work of sentence embedding. For example, for the paragraph, the specific method is to perform the embedding work for each sentence of the text separately, and finally calculate their average value as the feature of the entire text. On this basis, by calculating the cosine similarity of the embedding vectors of the two adjacent texts, the similarity can be obtained, which will be fully embedded in the graph. According to this, we can improve Figure 2 to Figure 3.

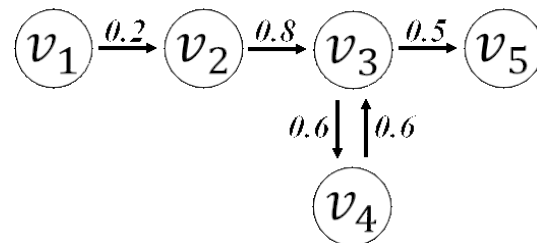


Figure 3. Converted weighted directed graph.

After supplementing the graph information, we use the method of gated GNN [40] to improve GNN, which uses the recurrent neural network mechanism for propagation. The training iteration process is as follows.

First, according to the graph information in Figure 3, we can build the out-degree and in-degree matrix of the corresponding graph, as shown in Figure 4.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|-----|-----|-----|-----|
| 1 | 0 | 0.2 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0.8 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0.6 | 0.5 |
| 4 | 0 | 0 | 0.6 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 |

| | 1 | 2 | 3 | 4 | 5 |
|---|-----|-----|-----|-----|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0.2 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0.8 | 0 | 0.6 | 0 |
| 4 | 0 | 0 | 0.6 | 0 | 0 |
| 5 | 0 | 0 | 0.5 | 0 | 0 |

Figure 4. Out-degree and in-degree matrix.

Second, inspired by the method proposed by Wu et al. [8], the input formula of the model is determined by the in-and-out matrix of the graph as

$$a_{s,i}^t = A_{s,i} : [v_1^{t-1}, \dots, v_n^{t-1}]^\top H + b \quad (1)$$

$$z_{s,i}^t = \sigma(W_z a_{s,i}^t + U_z v_i^{t-1}) \quad (2)$$

$$r_{s,i}^t = \sigma(W_r a_{s,i}^t + U_r v_i^{t-1}) \quad (3)$$

$$\tilde{v}_i^t = \tanh(W_o a_{s,i}^t + U_o(r_{s,i}^t \odot v_i^{t-1})) \quad (4)$$

$$v_i^t = (1 - z_{s,i}^t) \odot v_i^{t-1} + z_{s,i}^t \odot \tilde{v}_i^t \quad (5)$$

where

- v_i^t is the embedding vector corresponding to the i -th text in the reading sequence at time t during the training process. This vector changes continuously with the model training and is a d -dimensional vector.
- $A_s \in R^{n \times 2n}$ is the relationship matrix, which determines how the nodes in the graph are related to each other. n represents the number of different items in the sequence. This matrix will not change during the training process. A_s can be disassembled into $[A_s^{(out)}, A_s^{(in)}]$, corresponding to the in-out matrix, respectively.
- $A_s \in R^{1 \times 2n}$ are two columns related to node $v_{s,i}$ in A_s , and it is $1 \times 2n$. For example, $A_{s,3}$: corresponding to node 3 is equal to $[0, 0, 0, 0.6, 0.5, 0, 0.8, 0, 0.6, 0]^T$, as shown in Figure 5.

| Outgoing | | | | | Incoming | | | | | |
|-------------|---|-----|-----|-----|----------|-----|-----|-----|-----|---|
| | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 0.2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0.8 | 0 | 0 | 0.2 | 0 | 0 | 0 | 0 |
| $A_{s,3}$: | 0 | 0 | 0 | 0.6 | 0.5 | 0 | 0.8 | 0 | 0.6 | 0 |
| 4 | 0 | 0 | 0.6 | 0 | 0 | 0 | 0 | 0.6 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.5 | 0 | 0 |

Figure 5. Example of $A_{s,3}$.

- The function of $a_{s,i}^t$ is to extract the latent vector of the domain at time $t - 1$. $a_{s,i}^t$ is the input of graph neural network.
- H is the weight vector of $d \times 2d$, which can be decomposed into $[H_{in}, H_{out}]$.
- $z_{s,i}^t$ and $r_{s,i}^t$ represent the reset and update gates, respectively.

Finally, we can learn the vector of each text v_i in the session represents $v_i^t (1 < i < n)$ at time t .

3.3. Using Attention Mechanism to Learn the Session Feature Representation

To accurately obtain the representation of a session, we adopt the soft-attention mechanism. By sending all the session graphs into gated graph neural network, the embedding vector of all nodes can be obtained. We divide the feature vector $s_h \in R^d$ of the entire session into recent purpose s_l and long-term purpose s_g to consider. For the recent purpose s_l , we can simply define this of the session $[v_{s,1}, v_{s,2}, \dots, v_{s,n}]$ as the last-clicked text vector $v_{s,n}$, namely, $s_1 = v_n$. For long-term purpose s_g , we combine the embedding vectors of all nodes to calculate with the attention mechanism. The specific formula is as follows:

$$\alpha_i = q^\top \sigma(W_1 v_n + W_2 v_i + c) \quad (6)$$

$$s_g = \sum_{i=1}^n \alpha_i v_i \quad (7)$$

Among them, $q \in R^d$ and $W^1, W^2 \in R^{d \times d}$ are all trainable. With the training iteration, the weight α_i of each node's embedding vector is controlled. Finally, the weighted summation of the word embedding vector corresponding to each node is performed to obtain the final long-term purpose feature vector s_g .

By taking linear transformation over the concatenation of the local and long-term embedding vectors, we can get the feature representation of the session:

$$s_h = W_3[s_1; s_g] \quad (8)$$

where W_3 represents a matrix of d by $2d$ dimensions.

3.4. Obtaining Recommendation Results

After obtaining the feature representation of each session, we can calculate the score \hat{z}_l for each candidate text option v_i :

$$\hat{z}_l = s_h^\top v_i \quad (9)$$

After passing the score through a softmax activation function, the predicted output of the model is obtained:

$$\hat{y} = \text{softmax}(\hat{z}) \quad (10)$$

The loss function uses a common cross function:

$$L(\hat{y}) = - \sum_{i=1}^m y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \quad (11)$$

Algorithm 1 shows the pseudocode of our proposed session-based text recommendation model (TXT-SR).

Algorithm 1 Pseudocode of the TXT-SR algorithm.

Input: One browsing session sequence $s = [v_{s,1}, v_{s,2}, \dots, v_{s,n}]$

Output: Candidate top-K texts

- 1: construct a directed graph within the session s , $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$;
 - 2: **for** each text $v_{s,i}$ in s **do**
 - 3: embed every text $v_{s,i}$;
 - 4: **end for**
 - 5: calculate the similarity of adjacent items in the graph;
 - 6: obtain the vectors $\mathbf{v} \in \mathbb{R}^d$ of all items via graph neural networks;
 - 7: define the recent purpose s_l as the last-clicked text $v_{s,n}$;
 - 8: generate long-term purpose s_g by adopting the attention mechanism;
 - 9: concatenate the recent and long-term embedding vectors to get the session feature representation s_h by taking linear transformation;
 - 10: calculate the score \hat{z}_l for each candidate text option v_i by multiplying its embedding v_i by session representation s_h .
 - 11: select the text corresponding to the top-K values in the candidate options;
-

4. Experiments and Analysis

4.1. Datasets

In order to demonstrate the effectiveness of the proposed recommendation approach, two real-life academic datasets from CiteULike (<http://www.citeulike.org>, accessed on 15 April 2021), where users can create their own collections of articles, are used. Each article has a title and abstract (the other information about the articles, such as the authors, publications, and keywords, is not used in this paper). The first dataset, *citeulike-a* (the dataset can be downloaded from: <https://github.com/js05212/citeulike-a>, accessed on 15 April 2021), is from in [43], and there are 5551 users and 16,980 articles with 204,986 ob-

served user–item pairs, in which the average sequence length is 37. Users with fewer than 3 articles are not included in the dataset. The second dataset *citeulike-t* (The dataset can be downloaded from: <https://github.com/js05212/citeulike-t>, accessed on 15 April 2021), was collected by the authors of [44]. There are 7947 users and 25,975 articles with 134,860 observed user–item pairs. In this dataset, the average sequence length is 17. Users with fewer than 3 articles are not included in the dataset. The content information of the articles is the concatenation of the titles and abstracts. We performed cross-validation by assigning 10% of the randomly chosen train set as the validation set. The statistics of datasets is summarized in Table 2.

Following the work in [5], we use a sequence splitting preprocess that for an input session $s = [s_1, s_2, \dots, s_n]$, we generate the sequences and corresponding labels $([s_1], s_2), ([s_1, s_2], s_3), \dots, ([s_1, s_2, \dots, s_{n-1}], s_n)$ for training and testing on both datasets, which proves to be effective.

Table 2. Statistics of the experiment datasets.

| Statistics | <i>citeulike-a</i> | <i>citeulike-t</i> |
|-------------------|--------------------|--------------------|
| Clicks | 204,986 | 134,860 |
| Training sessions | 199,436 | 127,409 |
| Test sessions | 35,446 | 16,821 |
| Academic texts | 16,980 | 25,975 |

4.2. Evaluation Metrics

In terms of evaluation metrics, we use two measurements of *recall* at N and *mean reciprocal ranking* at N , which are widely used in the sequential recommendation.

- **Recall@N:** It calculates the proportion of the top- N retrieved positive samples. The specific formula is

$$\text{Recall@N} = \frac{\text{Number of correctly recommended papers in top } N}{\text{Number of recommended papers}} \quad (12)$$

- **MRR@20:** It refers to the average value of the inverse of the ranking of the desired items. If the ranking exceeds N , it is set to 0. The specific formula is

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^{|N|} \frac{1}{\text{rank}_i} \quad (13)$$

where $|N|$ is the number of recommended items, and rank_i is the i -th recommended item of the actual ranking in the recommended items required.

4.3. Parameter Setup

We set the corresponding item embedding vector dimension (200) and the mini-batch size (512). All parameters are initialized with a Gaussian distribution with a mean value of 0 and a standard deviation of 0.1. The Adam optimizer with the learning rate of 0.001 is adopted. Attenuation is set to 1.0 after every 5 epochs. The L2 penalty regularization parameter is set to 10^{-5} .

4.4. Baselines

We compared TXT-SR with the below nine baselines.

- **POP** exploits the frequency of items in the training set. It always recommends items that appear most often in the training set.
- **S-POP** is similar to POP; S-POP also exploits the frequency, but it recommends items that appear most often in the current sequence
- **Item-KNN** [11] uses content information to compute the cosine similarity between items.

- **BPR-MF** is a model representing a group of models with matrix factorization (MF) and Bayesian personalized ranking loss (BPR). By introducing the ranking loss, BPR-MF shows a better performance than a typical MF in the recommendation.
- **GRU4Rec** (<https://github.com/hidasib/GRU4Rec>, accessed on 15 April 2021) [4] is a sequential model with GRUs for the recommendation. This model adopts a session parallel batch and a loss function such as CrossEntropy, TOP1, or BPR.
- **GRU4Rec+** [5] is the improvement of the application of RNN in the field of session-based recommendation. It uses a data enhancement technology and changes the data distribution of the input data to improve the performance.
- **NARM** (https://github.com/lijingsdu/sessionRec_NARM, accessed on 15 April 2021) [6] is a model based on GRU4REC with an attention to consider the long-term dependency. Besides, it adopts an efficient bilinear loss function to improve the performance with fewer parameters.
- **STAMP** (<https://github.com/uestcnlp/STAMP>, accessed on 15 April 2021) [7] employs attention layers to replace all RNN encoders in previous work by fully relying on the self-attention of the last item in the current session to capture the user's short-term interest.
- **SR-GNN** (<https://github.com/CRIPAC-DIG/SR-GNN>, accessed on 15 April 2021) [8] employs a gated GNN layer to obtain item embeddings, followed by a self-attention of the last item as STAMP [7] to compute the session level embeddings for session-based recommendation.

Table 3 shows the performance of the baselines and TXT-SR with two measurements of recall at N and mean reciprocal ranking at N. We varied N by 5 and 20.

Table 3. Comparison with the current mainstream algorithms.

| Method | <i>citeulike-a</i> | | | | <i>citeulike-t</i> | | | |
|---------------|--------------------|--------------|-------------|-------------|--------------------|--------------|-------------|-------------|
| | Recall@5 | Recall@20 | MRR@5 | MRR@20 | Recall@5 | Recall@20 | MRR@5 | MRR@20 |
| POP | 1.46 | 5.81 | 0.89 | 1.42 | 1.28 | 4.33 | 0.91 | 1.40 |
| S-POP | 1.54 | 6.36 | 0.98 | 1.54 | 1.65 | 5.07 | 1.12 | 1.86 |
| Item-KNN | 0.00 | 6.91 | 0.00 | 3.76 | 0.00 | 5.76 | 0.00 | 1.89 |
| BPR-MF | 0.49 | 3.72 | 0.29 | 0.93 | 1.69 | 4.23 | 0.31 | 0.97 |
| GRU4Rec | 7.32 | 22.23 | 5.28 | 7.30 | 7.13 | 21.17 | 4.72 | 6.76 |
| GRU4Rec+ | 7.63 | 23.84 | 5.59 | 7.63 | 7.46 | 21.81 | 4.93 | 7.14 |
| NARM | 8.13 | 23.98 | 5.47 | 7.48 | 7.90 | 22.35 | 5.35 | 7.57 |
| STAMP | 7.95 | 21.96 | 4.97 | 6.89 | 7.38 | 21.92 | 5.01 | 6.93 |
| SR-GNN | 8.67 | 24.12 | 5.51 | 7.59 | 8.03 | 22.68 | 5.53 | 7.61 |
| TXT-SR | 8.98 | 25.89 | 6.21 | 8.17 | 8.28 | 24.29 | 5.98 | 7.92 |

Obviously, our proposed TXT-SR has achieved the best performance. The first four methods are obviously not competitive, which is sufficient to prove that traditional methods are no longer suitable for session-based recommendation. As the GRU4REC model only considers the user's sequence performance, it ignores the possible "mutation" behavior of the user's interest. The NARM model has achieved good performance in the test, not only because it uses a GRU unit to model sequence behavior, but also because it takes the main purpose of the user into account. Therefore, it can be seen that the main purpose of the recommendation system is still very important. Although the performance of STAMP is not as good as NARM, it emphasizes the distinction between current interest and general interest by taking into account the importance of the last click. The result of the graph neural network model without incorporating textual semantic relations is seemingly improved thanks to the powerful ability to capture more complex relationships between items in the sequence. However, the improvement effect is very limited. On the basis of the previous network model and ideas, we incorporate textual semantic relations in the graph neural network, which not only take the complex transformation relationship between items into

consideration, but also consider the closeness of this relationship. The improvement in effectiveness is relatively obvious.

4.5. Impact of Whether to Incorporate Textual Semantics

The purpose of this section is to prove that the introduction of different semantic weights will have a beneficial impact on the effect of the model. We use different methods to calculate the similarity:

- **TXT-SR-N**: Not using textual semantics, whose effect is equivalent to SR-GNN [8].
- **TXT-SR-C**: Replacing the weight of the edge in the constructed session graph with the cosine similarity of the adjacent vectors converted previously, which is our proposed model just to highlight the difference.
- **TXT-SR-P**: Like TXT-SR-C, the part being replaced with is Pearson correlation coefficient.
- **TXT-SR-J**: Regarding each text as a unique set of the bag-of-words, then calculating Jaccard coefficient to replace the weight of the edge.

Experimental results are illustrated in Figures 6 and 7 for citeulike-a and citeulike-t, respectively. In the two figures, we regard the experimental result of the session whose length is less than 10 as the result of the point on the abscissa 10, and the result of the point on the abscissa of 20 for the length between 10 and 20, and so on. In the two datasets, for the part of the session length that is less than 100, citeulike-a accounts for 93% and citeulike-t accounts for 91%. Therefore, we only take the session whose length is less than 100 into consideration.

We have the following observations from the results: (1) From the overall effect point of view, it can be observed that the performance of these three methods that use textual semantics (TXT-SR-C, TXT-SR-P, and TXT-SR-J) is better than that of TXT-SR-N, which can highlight the importance of textual semantics in the field of text recommendation. (2) Due to the increased difficulty of capturing user's purpose caused by the excessively long session, recommendation effect with a shorter session length is definitely better than those with a longer length. (3) We can clearly find that the improvement effect is not very obvious when the length is still short. As the length increases, the improvement brought by the three models of using semantics is more obvious than that of TXT-SR-N. As the graph neural network needs to use the information of the constructed graph in the continuous training and iteration process, specifically such information is the in-and-out matrix of the graph. This matrix, which is predefined, will be used to completely guarantee the information of the graph. The richer and more complete the information is, the more efficient it is to maintain the information during the training process. After integrating the textual semantic relationship between items, we replace the weights of the edges of the graph with the similarity of the text. Then, in the training and transmission of a long conversation, the rich matrix information can better maintain the purpose of the whole session. (4) The effects of TXT-SR-C, TXT-SR-P, and TXT-SR-J are different. It can be seen that the effects of TXT-SR-C and TXT-SR-P are better than TXT-SR-J. This is because the Jaccard coefficient only cares about the same words contained in two adjacent articles. The deeper semantic relations are not taken into consideration. TXT-SR-C and TXT-SR-P benefit from the powerful representation ability of embedding work. In addition, although the effects of TXT-SR-C and TXT-SR-P are very close, the performance of TXT-SR-C is still better than TXT-SR-P, indicating that TXT-SR-C is more capable of calculating semantic similarity. This may be because the Pearson Correlation Coefficient is between -1 and 1 , which brings more uncertainty when propagating training in the network.

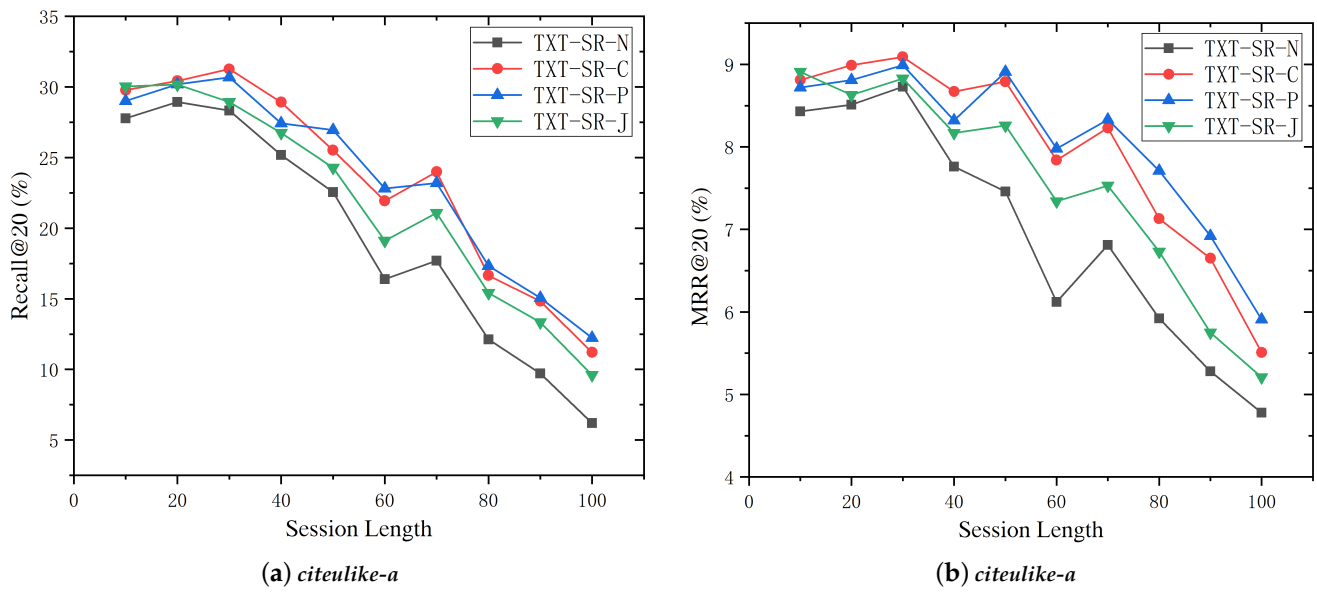


Figure 6. Performance comparison of different methods with different session lengths on *citeulike-a*.

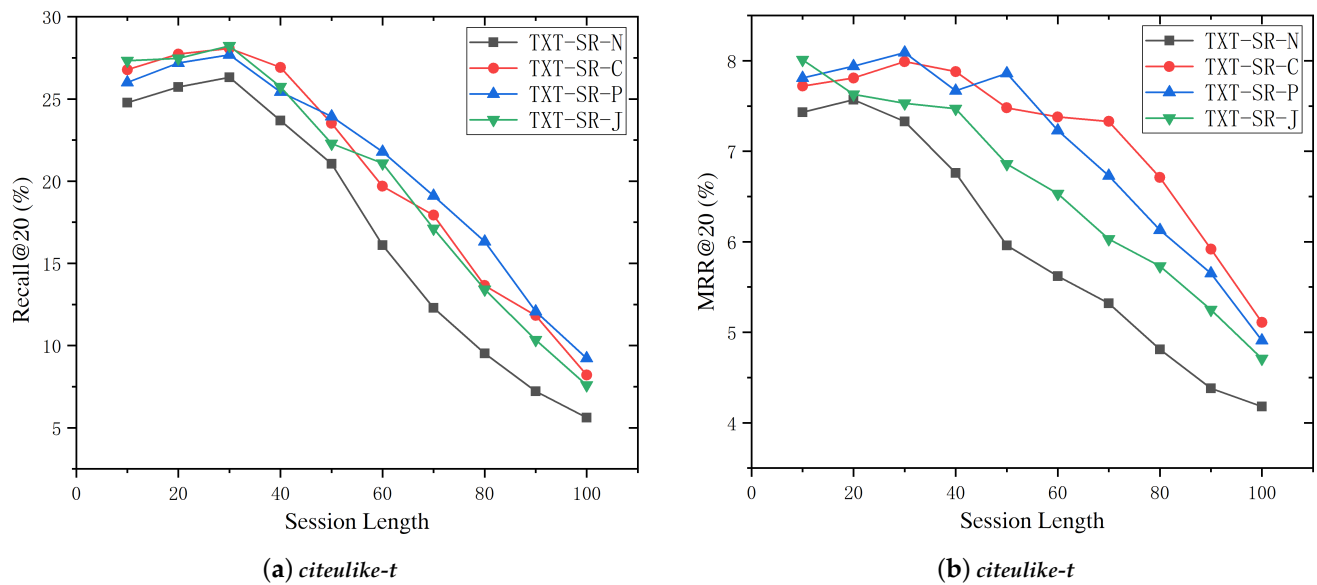


Figure 7. Performance comparison of different methods with different session lengths on *citeulike-t*.

4.6. Impact of Using Different Session Embeddings

In this part, we discuss the influence of using different session embedding approaches: (1) only using the last-click item embedding (TXT-SR-L), (2) long-term purpose embedding by using average embedding (TXT-SR-AVG), and (3) long-term purpose embedding with the attention mechanism (TXT-SR-ATT).

As shown in Figure 8, TXT-SR achieves the best results on two datasets, which indicates the importance of definitely incorporating recent session interests with the long-term preference. The performance of TXT-AVG is not as good as TXT-SR-ATT. This may be caused by some noisy behavior, which obviously should have different levels of priority. Besides, it is shown that attention mechanisms are helpful in extracting the significant behavior from the session data to construct the long-term preference. Although the performance of TXT-SR-L is no match for TXT-SR, it is better than TXT-SR-AVG and close to TXT-SR-ATT, which shows that recent purpose is also very important, and its importance is not inferior to long-term purpose.

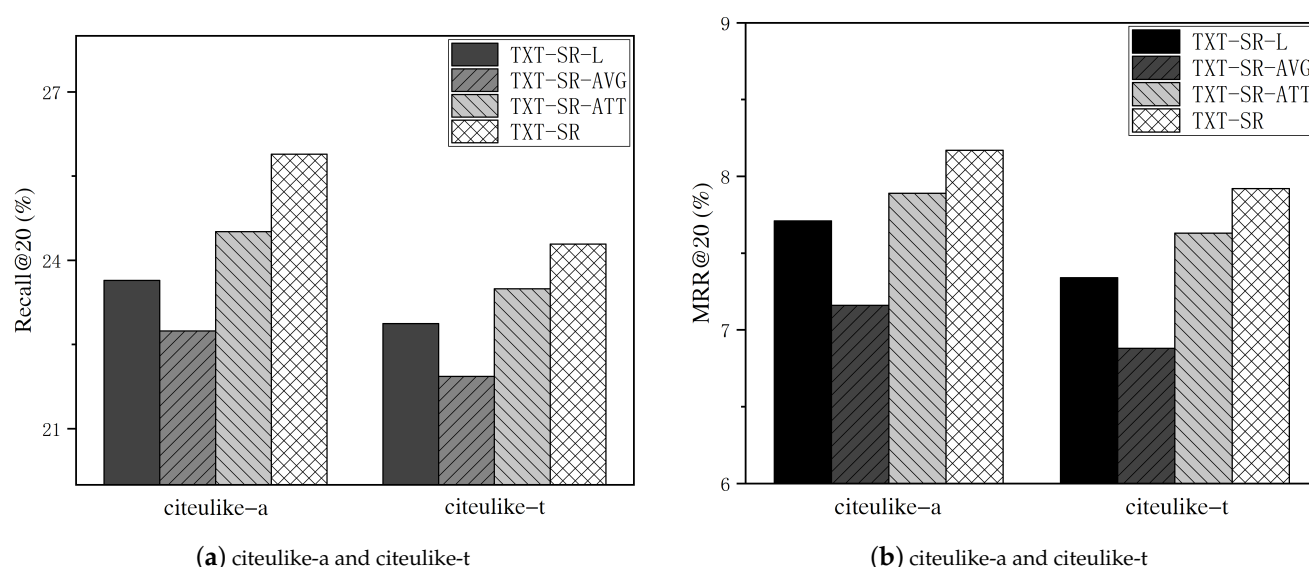


Figure 8. Performance comparison of different session representations.

5. Conclusions

The research on session-based text recommendation has important practical significance. This paper mainly proposes how to integrate the textual semantic relationship into the session-based recommendation system. Due to the advantage that the graph neural network can efficiently reflect the complex relationship between items through nodes, edges, and its own topological structure, the weight of the edge can be used to record the closeness between adjacent items, which is reflected in this paper as the semantic similarity between adjacent texts. Therefore, during training and transmission, this model can continuously carry out the calculation work of updating parameters according to the rich matrix information, so as to better preserve the relationship information between each other, which also effectively avoids the incomplete utilization of information in the traditional recommendation method. We conducted experiments on two real-life datasets from CiteULike. The experimental results show that the effect of incorporating textual semantic relations is very obvious whether it is on Recall or MRR, especially on long sessions. In addition to be applied to recommend academic text, this model can also be suitable for other text recommendations with session sequential information. As the innovation of this paper is to integrate semantic relations into graph neural networks, we can do more exploration on how to effectively extract semantic relations. For example, one limitation of this article is that experiments use the method of calculating the cosine similarity of the text embedding vector to extract semantic relations. In addition, we can attempt to adopt the DSSM model proposed by Huang et al. [45] to calculate text similarity, or the CNN-DSSM and LSTM-DSSM, which are derived from DSSM. The specific effect requires more experiments to know. This article only proposes the idea of incorporating semantic methods.

Author Contributions: Conceptualization, J.Y. and C.P.; Formal analysis, J.Y. and C.P.; Funding acquisition, J.Y. and C.P.; Writing—original draft, C.P.; Writing—review & editing, Y.L. and J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Wang, H.; Chen, B.; Li, W.J. Collaborative Topic Regression with Social Regularization for Tag Recommendation. International Joint Conference on Artificial Intelligence, 2013. The two datasets come from <https://github.com/js05212/citeulike-a> and <https://github.com/js05212/citeulike-t> (accessed on 15 April 2021).

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

References

- Bergamaschi, S.; Guerra, F.; Leiba, B. Guest editors' introduction: Information overload. *IEEE Internet Comput.* **2010**, *14*, 10–13. [\[CrossRef\]](#)
- Geiger, D.; Schader, M. Personalized task recommendation in crowdsourcing information systems—Current state of the art. *Decis. Support Syst.* **2014**, *65*, 3–16. [\[CrossRef\]](#)
- Resnick, P.; Varian, H.R. Recommender systems. *Commun. ACM* **1997**, *40*, 56–58. [\[CrossRef\]](#)
- Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; Tikk, D. Session-based recommendations with recurrent neural networks. *arXiv* **2015**, arXiv:1511.06939.
- Tan, Y.K.; Xu, X.; Liu, Y. Improved recurrent neural networks for session-based recommendations. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 17–22.
- Jing, L.; Ren, P.; Chen, Z.; Ren, Z.; Ma, J. Neural Attentive Session-based Recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017.
- Liu, Q.; Zeng, Y.; Mokhosi, R.; Zhang, H. STAMP: Short-term attention/memory priority model for session-based recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1831–1839.
- Wu, S.; Tang, Y.Y.; Zhu, Y.Q.; Wang, L.; Xie, X.; Tan, T.N. Session-Based Recommendation with Graph Neural Networks. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence/Thirty-First Innovative Applications of Artificial Intelligence Conference/Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 346–353.
- Chen, R.; Hua, Q.; Chang, Y.S.; Wang, B.; Zhang, L.; Kong, X. A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE Access* **2018**, *6*, 64301–64320. [\[CrossRef\]](#)
- Zhao, Z.D.; Shang, M.S. User-Based Collaborative-Filtering Recommendation Algorithms on Hadoop. In Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, WKDD 2010, Phuket, Thailand, 9–10 January 2010.
- Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, Hong Kong, China, 1–5 May 2001; pp. 285–295.
- Wang, J.; Vries, A.; Reinders, M. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In Proceedings of the SIGIR 2006: 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, USA, 6–11 August 2006.
- Linden, G.; Smith, B.; York, J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comput.* **2003**, *7*, 76–80. [\[CrossRef\]](#)
- Paterek, A. Improving regularized singular value decomposition for collaborative filtering. In Proceedings of the KDD Cup and Workshop, San Jose, CA, USA, 12 August 2007; Volume 2007, pp. 5–8.
- Koren, Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008.
- Shani, G.; Heckerman, D.; Brafman, R.I.; Boutilier, C. An MDP-Based Recommender System. *J. Mach. Learn. Res.* **2005**, *6*, 1265–1295.
- Wu, M.L.; Chang, C.H.; Liu, R.Z. Integrating content-based filtering with collaborative filtering using co-clustering with augmented matrices. *Expert Syst. Appl.* **2014**, *41*, 2754–2761. [\[CrossRef\]](#)
- Zimdars, A.; Chickering, D.M.; Meek, C. Using Temporal Data for Making Recommendations. *arXiv* **2013**, arXiv:1301.2320.
- Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [\[CrossRef\]](#) [\[PubMed\]](#)
- Zhang, S.A.; Yao, L.N.; Sun, A.X.; Tay, Y. Deep Learning Based Recommender System: A Survey and New Perspectives. *Acm Comput. Surv.* **2019**, *52*, 1–38. [\[CrossRef\]](#)
- Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M. Wide & Deep Learning for Recommender Systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016.
- Covington, P.; Adams, J.; Sargin, E. Deep Neural Networks for YouTube Recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 191–198.
- Okura, S.; Tagami, Y.; Ono, S.; Tajima, A. Embedding-based news recommendation for millions of users. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 1933–1942.
- Bobadilla, J.; Alonso, S.; Hernando, A. Deep learning architecture for collaborative filtering recommender systems. *Appl. Sci.* **2020**, *10*, 2441. [\[CrossRef\]](#)
- Sulikowski, P.; Zdziebko, T. Deep learning-enhanced framework for performance evaluation of a recommending interface with varied recommendation position and intensity based on eye-tracking equipment data processing. *Electronics* **2020**, *9*, 266. [\[CrossRef\]](#)
- Sulikowski, P.; Zdziebko, T.; Turzyński, D.; Kańtoch, E. Human-website interaction monitoring in recommender systems. *Procedia Comput. Sci.* **2018**, *126*, 1587–1596. [\[CrossRef\]](#)
- Sulikowski, P.; Zdziebko, T.; Coussement, K.; Dyczkowski, K.; Kluza, K.; Sachpazidu-Wójcicka, K. Gaze and Event Tracking for Evaluation of Recommendation-Driven Purchase. *Sensors* **2021**, *21*, 1381. [\[CrossRef\]](#)

28. Heaton, J. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. *Genet. Program. Evolvable Mach.* **2017**, *19*, 1–3.
29. Wu, S.; Ren, W.; Yu, C.; Chen, G.; Zhang, D.; Zhu, J. Personal recommendation using deep recurrent neural networks in NetEase. In Proceedings of the IEEE International Conference on Data Engineering, Helsinki, Finland, 16–20 May 2016.
30. Hidasi, B.; Quadrana, M.; Karatzoglou, A.; Tikk, D. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In Proceedings of the ACM Conference on Recommender Systems, Boston, MA, USA, 15 September 2016.
31. Quadrana, M.; Karatzoglou, A.; Hidasi, B.; Cremonesi, P. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In Proceedings of the Eleventh ACM Conference on Recommender Systems, Como, Italy, 27–31 August 2017.
32. Wu, C.Y.; Ahmed, A.; Beutel, A.; Smola, A.J.; Jing, H. Recurrent Recommender Networks. In Proceedings of the Tenth ACM International Conference on Web Search & Data Mining, Cambridge, UK, 6–10 February 2017.
33. Jhamb, Y.; Ebesu, T.; Yi, F. Attentive Contextual Denoising Autoencoder for Recommendation. In Proceedings of the 2018 ACM SIGIR International Conference, Tianjin, China, 14–17 September 2018.
34. Loyola, P.; Liu, C.; Hirate, Y. Modeling User Session and Intent with an Attention-based Encoder-Decoder Architecture. In Proceedings of the Eleventh ACM Conference on Recommender Systems, Como, Italy, 27–31 August 2017; pp. 147–151.
35. Wang, X.; Yu, L.; Kan, R.; Tao, G.; Zhang, W.; Yong, Y.; Wang, J. Dynamic Attention Deep Model for Article Recommendation by Learning Human Editors’ Demonstration. In Proceedings of the 23rd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017.
36. Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W.L.; Leskovec, J. Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 974–983.
37. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2d knowledge graph embeddings. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
38. Yan, S.; Xiong, Y.; Lin, D. Spatial temporal graph convolutional networks for skeleton-based action recognition. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
39. Ullah, I.; Manzo, M.; Shah, M.; Madden, M. Graph Convolutional Networks: analysis, improvements and results. *arXiv* **2019**, arXiv:1912.09592.
40. Li, Y.; Tarlow, D.; Brockschmidt, M.; Zemel, R. Gated Graph Sequence Neural Networks. *arXiv* **2015**, arXiv:1511.05493.
41. Beck, D.; Haffari, G.; Cohn, T. Graph-to-Sequence Learning using Gated Graph Neural Networks. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia, 15–20 July 2018; Volume 1, pp. 273–283.
42. Arora, S.; Liang, Y.; Ma, T. A simple but tough-to-beat baseline for sentence embeddings. In Proceedings of the International Conference on Learning Representations (ICLR), Caribe Hilton, San Juan, 2–5 May 2016.
43. Wang, C.; Blei, D.M. Collaborative topic modeling for recommending scientific articles. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 21–24 August 2011; pp. 448–456.
44. Wang, H.; Chen, B.; Li, W.J. Collaborative Topic Regression with Social Regularization for Tag Recommendation. In Proceedings of the International Joint Conference on Artificial Intelligence, Beijing, China, 3–9 August 2013.
45. Huang, P.S.; He, X.; Gao, J.; Deng, L.; Acero, A.; Heck, L. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, San Francisco, CA, USA, 27 October–1 November 2013; pp. 2333–2338.