Enhancing Collaborative Filtering with Deep Neural Networks and User Reviews

Ayman Shehda Ghabayen , Ahmed Mohammed Sadi Elaklouk * , Muhammad Rusyaidi Zunaidi * , Norizan Mat Diah , Abdul Azeem Khan , Anton Satria Prabuwono

Posted Date: 24 October 2024

doi: 10.20944/preprints202410.1949.v1

Keywords: Deep learning Bi-GRU; Collaborative filtering; Recommender systems; Top-N recommendation

Article

Enhancing Collaborative Filtering with Deep Neural Networks and User Reviews

Ayman Shehda Ghabayen [1], Ahmed Mohammed Sadi Elaklouk [2,*], Muhammad Rusyaidi Zunaidi [3,*], Norizan Mat Diah [4], Abdul Azeem Khan [5] and Anton Satria Prabuwono [6]

[1] Department of Computer Science, University College of Science and Technology, Palestine
[2] School of Computing and Informatics, Universiti Teknologi Brunei, Brunei Darussalam
[3] School of Computer Science University College Dublin, Dublin, Ireland
[4] School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Malaysia
[5] Faculty of Islamic Technology, University Islam Sultan Sharif Ali, Brunei Darussalam
[6] Faculty of Computing and Information Technology in Rabigh, King Abdulaziz University, Saudi Arabia
* Correspondence: ahmad.elaklouk@utb.edu.bn (A.M.S.E.); muhammad.zunaidi@ucdconnect.ie (M.R.Z.)

Abstract: In the era of information explosion in e-commerce, users face an overload problem. Recommender systems alleviate this by filtering target products based on user preferences. Collaborative Filtering (CF) is a popular method but struggles with high-quality recommendations due to data sparsity. Utilizing review texts to capture user preferences has improved CF recommendations. This paper proposes a deep learning-based recommendation framework leveraging bidirectional gated recurrent units (Bi-GRUs) and an attention-based Recurrent Neural Networks (RNN) structure. This framework addresses insufficient user rating data by utilizing user reviews to infer preferences. Experimental results on three datasets show that the proposed Bi-GRUCF model outperforms other state-of-the-art methods, achieving a 55.4% improvement in Mean Absolute Error (MAE) over traditional methods and a 77.8% improvement over other deep learning approaches. This demonstrates the framework's superior recommendation quality, particularly in sparse data scenarios.

Keywords: deep learning Bi-GRU; collaborative filtering; recommender systems; Top-N recommendation

1. Introduction

Nowadays, e-commerce websites offer a great variety of items and brands for sale. They differ in shape, size, color, and other variances. The vast quantity of products available causes consumers to experience information overload. This may lead to customers feeling confused and overstimulated because of the influence of a huge amount of information that is beyond their ability to process. Developers create recommender systems (RS) to address the problem of information overload [1]. RS is a type of expert system that assists customers in obtaining personalized relevant items of interest from a large set of possible choices. Companies such as Amazon, Spotify, Netflix, and YouTube are all examples of popular recommender systems in use [2,3]. CF is deemed to be one of the most promising and widely used methods in building a RS. Collaborative Filtering (CF) techniques offer personalized recommendations to target users by identifying commonalities between the target user and other users within the community. The Corresponding users or items are identified by calculating the similarities among users' common ratings on certain items. Though, CF approaches perform well when there is sufficient rating information [4]. Nevertheless, CF approach efficiency often suffers from limitations like cold-start problems and sparsity of data [5]. Data sparsity occurs when there are few co-rated items between users or when there is no overlap in co-rated items. The cold-start problem arises when new items are introduced into the system that either have no ratings or have not received enough ratings. Another limitation is that CF approaches do not capture the reason behind user numerical ratings and, accordingly, cannot capture the accurate personal preferences of a target user [6,7].

 doi:10.20944/preprints202410.1949.v1

To enrich, enhance, and handle problems associated with CF efficiently, numerous methods have been proposed namely the representation of users and items by exploiting additional knowledge sources, including user tags [8], items' product descriptions [9], and social media aspects [10]. However, these solutions remain inadequate, particularly when dealing with high levels of rating sparsity or insufficient rating preferences for target users.

Currently, e-commerce platforms such as Yelp and Amazon provide users with the opportunity to express their opinions on items through written reviews. These review texts contain valuable information and reliable user preferences, which can enhance CF approaches. User text reviews offer semantic information that, when combined with users' rating preferences, can significantly improve the efficiency of CF recommendations [11,12].

Several studies have proposed methods to uncover the hidden aspects of user review texts using techniques such as Latent Dirichlet Allocation (LDA) or Non-Negative Matrix Factorization (NMF) [13]. These studies have shown improvements in recommendation quality compared to traditional collaborative filtering (CF), which solely relies on numerical ratings. However, LDA and NMF utilize bag-of-words (BOW) representations that overlook the contextual information embedded in the review text. Deep learning models address this limitation by learning and analyzing sequential textual features in review text [14].

The current study aims to enhance the quality of recommendations through CF systems by leveraging users' review texts with Bi-GRU deep recurrent networks. This proposed approach addresses the limitations of traditional rating-based CF methods by incorporating the content of user reviews to infer rating preferences and generate recommendations. This method improves recommendation accuracy, particularly in sparse data scenarios, by utilizing deep learning to extract richer feature representations from review texts.

The rest of the paper is organized as follows: Section 2 provides background information on traditional collaborative filtering (CF) recommender systems and recurrent neural networks. Section 3 reviews related work in CF. Section 4 introduces the proposed recommendation framework. Section 5 details the experiments, including datasets, methodology, metrics, comparisons results, and discussion. Finally, Sections 6 summarize the conclusions and suggest future research directions.

2. Background Study

This section comprises two parts. The first part delves into the background of traditional collaborative filtering recommendation systems, while the second part elucidates an overview of Recurrent Neural Networks (RNN).

2.1. Traditional Collaborative Filtering

Collaborative filtering recommender systems are classified into two categories: memory-based (also known as heuristic-based) and model-based. The first category, memory-based approaches, utilizes the entire user-item database to identify the "neighborhood" of a new user or item. Based on the distance or correlation between users or items, each neighbor is assigned a weight, and the algorithm aggregates their preferences to generate predictions or recommendations for the new or target user[15]. When the objective is to provide top-N recommendations, these approaches focus on finding the most similar users or items. The memory-based approach is further classified into two methods: user-based and item-based, depending on whether the similarities are determined between users or items [16]. User-based collaborative filtering operates on the assumption that users with similar preferences will enjoy the same items. This approach typically employs metrics such as cosine similarity or Pearson's correlation coefficient to compare the target user's choices with those of other users, identifying a group of "like-minded" users. Once this group is identified, items that have received high ratings from the group are recommended to the target user [17]. Conversely, the item-based approach focuses on finding the most similar items rather than users.

In comparison to the memory-based approach, the model-based approach recommends items by creating a model of user ratings rather than relying on identifying similar users. Model-based recommendations employ machine learning and data mining techniques, such as rule-based

 doi:10.20944/preprints202410.1949.v1

approaches, matrix factorization, and clustering models, to develop prediction models. To estimate missing ratings in the user-item matrix, Recommender systems use a model-based approach. According to [18], model-based collaborative filtering tends to be more accurate and generates more relevant recommendations than memory-based recommender systems.

In traditional CF the recommendation process can be formulated as follows. Let $U = \{u_1, u_2, u_3, \ldots\ldots, u_N\}$ be a set of registered users in the system, and let $I = \{i_1, i_2, i_3, \ldots\ldots, i_M\}$ (*see Table 1 for symbol definition presented in this article*) be a set of all possible items users have access to in the recommender system. Furthermore, assume $\hat{r}: U \times I \rightarrow R$, where R is a set of real or non-negative numbers to be a utility function. The utility function $\hat{r}(u, i)$ measures the gain of how likely a user u is interested in item i. According to Adomavicius and Tuzhilin [19], the recommendation problem consists of selecting unseen item $I \in I$ for each user $u \in U$, that maximizes the utility function formally, through Eq. (1),

$$\hat{r}(u,i) = \bar{r}_u + \frac{\sum_{v \in \hat{U}} sim(u,v) \times (r_v - \bar{r}_v)}{\sum_{v \in \hat{U}} |sim(u,v)|} \qquad (1)$$

where $sim(u, v)$ indicate the similarity between user u and user v, $\bar{r}_u$ denote the average ratings of user u and $\bar{r}_v$ the average ratings of user v. There are different methods to compute the similarity between items or users. The most commonly used methods to compute the similarity between two users u and v are Cosine-based and Pearson correlation coefficient similarity measures [20]. The resemblance concerning user u and v is defined as follows, by Eq. (2),

$$Sim(u,v) \frac{\sum_{i_m \in I_{n,k}} (r_{n,m} - \bar{r}_n) \times (r_{k,m} - \bar{r}_k)}{\sqrt{\sum_{i_m \in I_{n,k}} (r_{n,m} - \bar{r}_n)^2} * \sqrt{\sum_{i_m \in I_{n,k}} (r_{k,m} - \bar{r}_k)^2}} \qquad (2)$$

where $I_{n,k}$ denote the co-rated items between users u and v. In other words, it refers to the items rated by both users.

The primary challenge for CF systems is data sparsity [3]. which occurs when the number of co-rated items between the target user and others is not enough. Locating comparable users to the target user in a sparse and large dataset is particularly challenging in CF. Therefore, sparsity leads to poor-quality recommendations.

2.2. Recurrent Neural Networks (RNN)

Concerning the human brain, artificial intelligence algorithms have different approaches for handling individual and sequential data. The first generation of artificial neural networks (ANNs) gained popularity in the past years. ANNs are feedforward networks proficient in learning individual parts of data, such as single images or fixed-length records of information. However, they are not suitable for sequential data, where the order of input data matters. On the other hand, a class of neural networks naturally suited to capture the sequential information present in the input data are Recurring Neural Networks[21]. RNNs are considered feedback neural networks, where the current output from neurons is used as an input for the next neuron's output each time, forming a directed forward and backward cycle between neurons' input and output.

Although basic RNNs are utilized in many applications for predicting short-term sequence data, they suffer from vanishing gradient and exploding gradient problems [22]. These problems arise when there is a large sequence of input data, making it challenging for RNNs to learn from data with long-term dependencies, such as processing text with long sentences. Advanced RNN models have been proposed in the literature to address the vanishing gradient problem inherent in standard RNNs. Hochreiter and Schmidhuber[23] introduced the long short-term memory (LSTM) model, which was one of the first successful RNN models specifically designed to tackle this issue. Despite its effectiveness, the LSTM model has a complex structure with many parameters, making it challenging to learn long-term dependencies. To address these complexities, Cho et al. [24] proposed the gated recurrent unit (GRU) as an alternative to the standard RNN. This study utilizes the GRU model. The following subsection provides a comprehensive overview of its application.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 24 October 2024 doi:10.20944/preprints202410.1949.v1

2.1.1. Gated Recurrent Unit (GRU)

The GRU model delivers excellent performance in sequence modeling, particularly in handling long-term data dependencies such as those in text modeling. The GRU architecture introduces a gating mechanism within RNNs, featuring two types of gates: update and reset gates. The reset gate determines how much past information should be discarded from the model. Conversely, the update gate decides how much past information (from earlier time steps) should be carried forward to the next state. In the text modeling process, the first step is word embedding. This involves mapping each word in a sentence or document into a low-dimensional semantic space vector. Given an input sequence $s_t \in R^{d_s}, t \in (1,2,3,\dots,T)$, a basic RNN defines the sequence of hidden states $h_t \in \mathbb{R}^{d_h}$ update at each time step according to the following rule in Eq (3),

$$h_t = \sigma(W_h h_{t-1}) + (W_s s_t) + b) \quad (3)$$

Where $W_h \in \mathbb{R}^{d_h \times d_h}$, $W_s \in \mathbb{R}^{d_s \times d_h}$ and $b \in \mathbb{R}^{d_h}$ are the model parameters and σ is a nonlinear activation function.

In GRU the input sentence in a document is mapped into a set of hidden states h_j^i as follow, Eq (4), Eq (5), Eq (6) and Eq (7),

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h_t} \quad (4)$$

$$z_t = \sigma(W_z w_t) \odot U_z h_{t-1} + b_z) \quad (5)$$

$$\tilde{h_t} = \tanh(W_z w_t) + U_h(r_t \odot h_{t-1}) + b_h) \quad (6)$$

$$r_t = \sigma(W_r w_t) \odot U_r h_{t-1} + b_r) \quad (7)$$

Error! Reference source not found. described below presents the comprehensive architecture of the basic GRU unit.

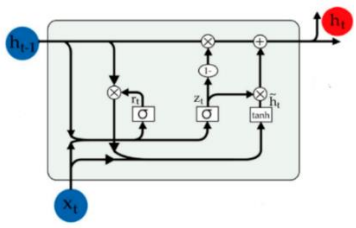

Figure 1. Comprehensive Overview of GRU Unit Design.

3. Related Works

Recommender systems (RS) have become ubiquitous, guiding user choices on online platforms by suggesting relevant products, services, or content tailored to individual preferences [19] . Collaborative Filtering (CF) remains a dominant approach, leveraging user-item interaction data like ratings or purchase history to generate recommendations [3]. However, CF methods struggle with data sparsity, where limited interactions hinder the generation of high-quality recommendations [3].

This section explores recent advancements in recommendation systems that address data sparsity and enhance recommendation accuracy. We focus on various approaches, including advanced matrix factorization techniques, leveraging user reviews and user behavior, deep learning architectures, and how these approaches compare to the one proposed in this paper.

3.1. Beyond Matrix Factorization: Exploring New Avenues

Matrix factorization (MF) techniques like Singular Value Decomposition (SVD) and Non-negative Matrix Factorization for Recommendation Systems with Dynamic Bias (NMFRS) have been foundational for CF, they primarily rely on explicit user-item interactions [25,26]. This dependence

 doi:10.20944/preprints202410.1949.v1

on explicit ratings can limit their ability to capture the nuances of user preferences, especially when dealing with data sparsity issues.

Alternative approaches explore different aspects of recommendation tasks, moving beyond the limitations of traditional MF methods. However, these methods often struggle to fully capture the richness of user preferences, particularly when dealing with sparse data.

Here, we propose the BI-GRUCF model, which addresses this limitation by incorporating user reviews alongside collaborative filtering. By leveraging the valuable insights within reviews, BI-GRUCF can provide more comprehensive user profiles and improve recommendation accuracy, even in scenarios with data sparsity.

3.2. Utilising User Reviews for Recommendations

Several approaches address data sparsity by incorporating user reviews and deep learning architectures. For instance, Vuong et al.[27] leverage Word2Vec to capture semantic relationships between items from reviews, alleviating sparsity issues. Their method offers a significant improvement over traditional MF techniques by considering semantic relationships. However, it focuses solely on item relationships and doesn't explicitly model user preferences within reviews.

Other studies mine user opinions from reviews to enrich CF models. D'Addio et al. [28] use vector representations of user reviews, while Yang et al. [30] employ sentiment analysis to extract user opinions. These approaches offer valuable insights from reviews but lack a mechanism to capture the sequential nature of user reviews and the contextual dependencies within them. Ghabayen and Ahmed [29] combine review clustering with user ratings, but their method doesn't leverage the rich sequential information embedded in review text.

While these approaches offer valuable insights into user preferences from reviews, they often fail to capture the full context and sequential nature of user opinions. This can limit their effectiveness in overcoming data sparsity challenges.

The proposed BI-GRUCF model addresses this limitation by employing Bidirectional GRUs to analyze the sequential nature of user reviews. This allows BI-GRUCF to capture richer contextual information and user preferences, leading to more accurate and personalized recommendations.

By combining the strengths of collaborative filtering and user review analysis through a deep learning approach, the BI-GRUCF model offers a significant advancement over existing methods, particularly in its ability to address data sparsity and generate more accurate recommendations.

3.3. Deep Learning and Recommendation Systems

The advancements in deep learning techniques, particularly within natural language processing and related fields, have significantly influenced the trajectory of recommendation systems (RS). Deep learning methods have been increasingly employed to address RS challenges such as sparsity, cold start issues, and the enhancement of recommendation quality through the extraction and inference of embedding features [30].

While Recurrent Neural Network (RNN) architectures like Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM) have gained traction for sequential learning tasks involving textual data, this study underscores the suitability of Gated Recurrent Units (GRUs) for RS due to their intrinsic mechanism for processing and retaining information over sequences [30].

The integration of Convolutional Neural Networks (CNNs) with Collaborative Filtering (CF) techniques for recommendation systems has shown promising results. CF effectively captures user-item interaction data, identifying users with similar preferences for relevant recommendations. CNNs excel in extracting features from complex data such as reviews or images, enhancing the understanding of user preferences and item characteristics. Recently, Arliyanna et al. [31] proposed a film recommendation system utilizing content-based filtering and CNN classification methods. This approach demonstrated the potential of CNNs in enhancing CF techniques for more accurate and personalized film recommendations.

Illustrating the practical application of RNNs in recommendation models, Kim et al. proposed a context-aware model that integrates CNNs with probabilistic matrix factorization (PMF) to enhance

 doi:10.20944/preprints202410.1949.v1

rating predictions in RS [32]. Similarly, Zheng et al. introduced the DeepCoNN model, employing dual CNNs to respectively learn user preferences and item characteristics from review text, culminating in a layered approach that captures user-item interactions for accurate predictions [33].

Moreover, the classification-based deep neural architecture proposed by [34] introduces a novel approach using binary classifications to determine item relevance and user voting, streamlining the prediction process through efficient model execution. Additionally, Ahmed and Ghabayen present a framework for review rating prediction, utilizing bidirectional GRU architectures to predict sentiment polarity and subsequent user ratings, demonstrating substantial improvements in prediction metrics such as accuracy, recall, F1-score, and root-mean-square error (RMSE) [35].

While these approaches offer valuable advancements in specific aspects of recommender systems, they may not fully capture the richness of user preferences and the complex relationships between users, items, and reviews.

Despite recent advancements in recommendation systems (RS), several limitations persist. Traditional CF and matrix factorization techniques struggle with data sparsity and fail to capture the full scope of user preferences, particularly in sparse data scenarios [25,26].

Approaches leveraging user reviews, such as those utilizing Word2Vec [27] and sentiment analysis [28], often neglect the sequential nature and contextual dependencies of reviews. The proposed BI-GRUCF model addresses these limitations by employing Bidirectional GRUs to analyze the sequential nature of user reviews, capturing richer contextual information and nuanced user preferences. This leads to more accurate and personalized recommendations, effectively mitigating data sparsity issues and enhancing recommendation accuracy, thereby offering a significant advancement over existing methods.

4. Proposed Approach

This paper introduces a novel approach to CF that aims to enhance the quality of recommendations and address the sparsity issue common to CF. The proposed approach consists of two distinct phases. In the first phase, a deep learning model incorporating stacked bi-bidirectional gated recurrent units (Bi-GRUs) is utilized to infer user ratings from textual review text. The second-phase recommender system involves the CF component, which generates Top-N recommendations based on the estimated ratings, which generates Top-N recommendations based on the estimated ratings. To provide a visual representation of the proposed collaborative filtering method, refer to Figure 2. The subsequent subsection delves into the detailed steps of the proposed method.

4.1. Phase I: Deep Learning Bi-GRU Model

This section encompasses the data preprocessing steps, including the word embedding process applied to the resulting data. Additionally, it outlines the structure of the Bi-GRU neural network layers.

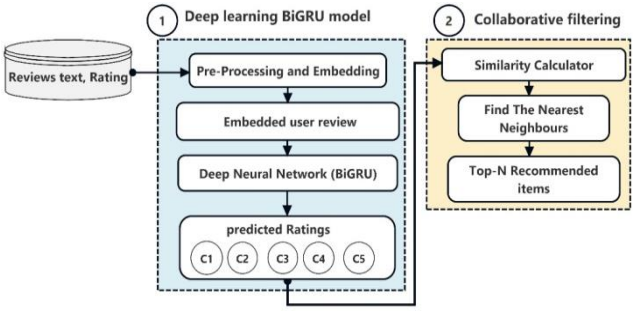

Figure 2. Proposed CF System Architecture.

 doi:10.20944/preprints202410.1949.v1

4.1.1. Data Pre-Processing

The data preprocessing phase is a crucial step in the development of recommender systems that leverage text data. The goal of data preprocessing is to prepare the text data for analysis by removing noise and irrelevant information, and by ensuring that the data is consistent and well-formatted. The following are some of the most common data preprocessing techniques used in recommender systems:

- Tokenization: The process of breaking down text into individual words or tokens is known as tokenization. This is accomplished by splitting the text based on whitespace and punctuation marks.
- Stop word removal: These are the common words that do not add much meaning to the text. Hence, these words are typically removed from the text to improve the efficiency of the analysis.
- Stemming: The process of reducing words to their base or root form is known as stemming. This is achieved by removing prefixes and suffixes.
- Lemmatization: Lemmatization is a process of grouping words that have the same base or root form, regardless of their inflection. This is done by considering the contextual meaning of words.
- Case normalization: Case normalization is the process of converting all words to either lowercase or uppercase. This is done to ensure that the text is consistent and well-formatted.
- Removal of extra white spaces: Extra white spaces are any additional spaces that are present in the text. These spaces are typically removed to improve the readability of the text.
- Removal of URLs: URLs are web addresses that are typically used to link to external websites. These URLs are typically removed from the text to improve the efficiency of the analysis.
- Removal of hashtags: Hashtags are used to tag text with keywords. These hashtags are typically removed from the text to improve the readability of the text.
- Removal of date and time references: To improve the efficiency of the analysis, Date and time references are typically removed from the text.

By applying these preprocessing techniques, the text data can be cleaned and prepared for analysis. This can improve the accuracy and effectiveness of recommender systems by removing noise and irrelevant information, thereby ensuring data consistency and its proper formating.

4.1.2. Bi-GRU Model

Bidirectional Gated Recurrent Units (Bi-GRUs) are a powerful recurrent neural network (RNN) that excels in natural language processing tasks. They have unique ability to capture long-term dependencies in sequential data, that are crucial for understanding the intricate nuances present in the user review text. Bi-GRUs offer not only superior performance but also efficient training, thereby making them highly suitable for scaling recommender systems. Research conducted by [36] has demonstrated that Bi-GRUs outperform other types of RNNs, including LSTMs, particularly when they are dealing with user review text. This advantage stems from their remarkable capacity to comprehend long-range dependencies, which enables them to have a deeper understanding of the text and ultimately leading to a more precise and accurate recommendations. The specific architecture of the Bi-GRU layers is illustrated in **Error! Reference source not found.**.

As illustrated in **Error! Reference source not found.**, the Bi-GRUs are trained on the input sequence in reverse order. Therefore, the output of hidden layers at time t is determined based on the outputs of both the backward and forward layers hidden at times t and t-1, by Eq (8), Eq (9), Eq (10) below:

 doi:10.20944/preprints202410.1949.v1

$$\vec{h}_{t-1} = GRU(x_t, \vec{h}_{t-1}) \quad (8)$$

$$\overleftarrow{h}_{t-1} = GRU(x_t, \overleftarrow{h}_{t-1}) \quad (9)$$

$$\overleftarrow{h} = w_t\vec{h}_t + v_t\overleftarrow{h}_t + b_t \quad (10)$$

The GRU units apply a nonlinear transformation to the word embedding vector. As a result, this word embedding vector is encoded into a corresponding GRU unit, which resides in the hidden layer state. Where w_t and v_t are the weights related to the forward hidden layer state $\vec{h}_t$ and the backward hidden state $\overleftarrow{h}_t$, correspondingly at time moment t.; $b_t$refers to the offset value at the time moment t.

- Review Word Embedding Layer: The Word Embedding Layer is a crucial component of neural networks used for representing words as vectors. These vectors typically have dimensions ranging from 100 to 300 and are trained on extensive text corpora. By capturing the semantic meaning of words, these vectors significantly enhance the accuracy of natural language processing tasks. In the proposed CF system, the Review Word Embedding Layer plays a pivotal role. It accepts a fixed sequence of 200 words as input and transforms each word into its corresponding 300-dimensional GloVe word vector [37]. These embedding vectors are subsequently fed into the Bi-GRU layers, allowing them to learn the long-term dependencies at each specific time step present in the user review text.
- Drop-out layer: The embedded resulting word vectors can be encoded either with numerical values associated with the input words or by using one-hot vectors that is, vectors with only one position filled with a value of one and the rest filled with zeros. However, an optimized embedding matrix can cause overfitting, which can be prevented by using dropout on one-hot vectors [38]. The 1D of spatial dropout layer with a dropout rate of 0.5 plays a role in preventing the activation functions from becoming strongly correlated, leading to better generalization, and preventing overfitting of the model.
- Context Attention layer: The context attention layer is applied to more effectively predict the rating class of a given review since words do not equally contribute to indicating the importance of meaning or the message of the theme. The attention layer obtains such words that significantly demonstrate the meaning of a sentence. This model utilizes the attention mechanisms as in [39]. This mechanism initiates a context vector, which can be viewed as a fixed query, that helps to extract the informative words. The context vector is randomly initialized and jointly learned with the rest of the attention layer weights, formally: Let H be a matrix composed of output vectors$[h_1, h_2, h_3, \ldots, h_T]$ that are produced by the last BiGRUlayer and T is the length of the sentence. Context attention r of the sentence is determined by the weighted sum of the produced vectors as by Eqs. (11), (12) and (13),

$$M = \tanh(H) \quad (11)$$

$$\alpha = softmax(w^T M) \quad (12)$$

$$r = H\alpha^T \quad (13)$$

Where $H \in \mathbb{R}^{d^w \times T}$, d^w is the dimension of the word vectors, w is a trained parameter vector and $w^T$is the transpose. Hence the dimension of w, α, r is, d^w T, d^w separately. The output of this layer is calculated as by Eq (14),

$$h^* = \tanh(r) \quad (14)$$

 doi:10.20944/preprints202410.1949.v1

- Concatenation layer: The concatenation layer performs global max pooling and global average pooling to generate one feature map for each class in the classification task. The use of global average pooling requires no parameter optimization, which helps prevent overfitting. Additionally, global average pooling ignores spatial information, making it more robust against spatial variations in the input data. After applying global max pooling and global average pooling to each Bi-GRU output, the resulting vectors are concatenated into a 1D array. Finally, this concatenated output is passed through a dense output layer with a sigmoid function to provide the rating prediction class.

4.2. Phase II: Recommender System

The user-based CF methodology is employed in the second phase of the recommender system. This phase aims to leverage the predicted rating results from the previous phase to calculate the similarity between users.

The cosine similarity metric, widely used in information retrieval and recommendation systems, is employed for this purpose, and is adopted to quantify the similarity between user vectors representing their rating patterns. Following the similarity calculation, the subsequent step involves identifying the most similar users, commonly known as nearest neighbors. These users exhibit comparable preferences and behaviors based on their historical rating data. The nearest-neighbor approach enables the recommender system to harness the collective wisdom of users who share similar tastes and preferences, facilitating the capture of valuable user insights.

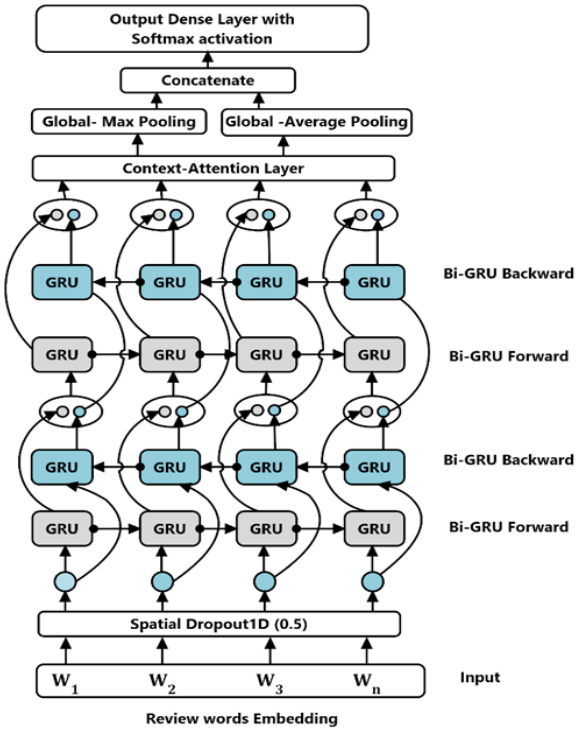

Figure 3. The Structure of (BiGRU) layers.

Once the set of K's similar or like-minded users is determined, the final step entails generating the top-N recommended items for the target user. This is accomplished by considering the preferences and choices of the identified similar users and incorporating their historical rating data and preferences. The recommended items are selected based on the collective wisdom and experiences of these users, ensuring a personalized and relevant recommendation. By incorporating user-based CF in this manner, the recommender system effectively taps into the collective intelligence and preferences of similar users, thereby improving the precision and significance of the

 doi:10.20944/preprints202410.1949.v1

recommendations. Thus, this collaborative approach enables the whole system to leverage the shared behaviors and tastes of a community of users, providing tailored recommendations that align with individual user preferences.

5. Experiments

This section outlines a series of experiments conducted using three real-world datasets from Amazon. The goal of these experiments is to address the following research question:

RQ1: How does the proposed approach compare to traditional baseline collaborative filtering (CF) methods?

The experiments aim to assess the effectiveness and superiority of the proposed approach by comparing it against established collaborative filtering (CF) baselines. By benchmarking the proposed approach against these traditional methods, the study aims to assess its ability to provide improved recommendations and address the limitations of existing CF techniques.

The evaluation of the proposed approach involves measuring various performance metrics, including precision, recall, accuracy, and F1-score. These metrics provide insights into the quality and effectiveness of the recommendation outcomes. By comparing these metrics between the proposed approach and the baseline methods, the study aims to determine how well the proposed approach outperforms or surpasses traditional collaborative filtering (CF) methods.

The real-world datasets selected and obtained from Amazon ensure the relevance and applicability of the findings to practical recommendation scenarios. These datasets comprise diverse user preferences, item characteristics, and rating patterns, thereby enabling a thorough assessment of the proposed approach's performance across different domains.

Through this experimental analysis, the study aims to provide empirical evidence and quantitative results that determine the dominance of the recommended approach in comparison to traditional CF baselines. These findings contribute in advancing the field of recommendation systems and provide insights into the effectiveness of novel approaches in improving recommendation quality and user satisfaction.

5.1. Experiments Parameters Settings

The dataset was randomly divided into five folds, with 80% of the data used for the training set and the remaining 20% used as the test set. To ensure unbiased recommendation results, the process was repeated five times, ensuring that no specific training or test set influenced the outcomes. The dataset was randomly divided into five folds, with 80% of the data used for the training set and the remaining 20% used as the test set. To ensure unbiased recommendation results, the process was repeated five times, ensuring that no specific training or test set influenced the outcomes. The constraint that is significant to note here is that the ratings in the test set were never included in the training set. The average values obtained from these iterations were considered the results, ensuring reliable prediction of outcomes. One of the significant advantages of this technique is its ability to maintain a consistent class distribution between the training and test sets, which is crucial in training neural networks, as they enable the model to generalize beyond the training data. By avoiding biases in the training set and addressing overfitting, the model becomes more robust in handling new and unseen data.

The proposed model uses an embedding dimension of 200, representing the size of the word vectors used to encode the input data. This dimension helps capture the semantic meaning of words and their relationships within the text.

The Review Word Embedding Layer in the proposed collaborative filtering (CF) system takes a fixed sequence of 200 words as input and converts each word into its corresponding 200-dimensional pre-trained GloVe word vector. This study uses the "Wikipedia 2014 + Gigaword 5" version of GloVe, which contains 6 billion tokens and a vocabulary size of 400,000.

The model sets the maximum number of features, or vocabulary size, to 20,000, determining the number of unique words it can handle. To prevent overfitting and enhance generalization, a spatial

 doi:10.20944/preprints202410.1949.v1

dropout layer is incorporated with a dropout rate of 0.5. This layer randomly sets a fraction of input units to zero during training, helping in regularization and preventing the model from overly relying on specific features. The model architecture includes a bidirectional Gated Recurrent Unit (GRU) layer with 40 units. The bidirectional aspect allows the model to process the input sequence in both forward and backward directions, capturing contextual information from both past and future contexts. The return sequences parameter is set to true, meaning the layer returns the full sequence of hidden states as output. The output layer is a dense layer with several units equal to the number of classes. The activation function used is the sigmoid function, which produces probabilities for each class, indicating the likelihood of the input belonging to each class.

Early stopping and checkpoint callbacks were configured to trigger at the end of each epoch. Early stopping automatically halts model training when the evaluation metric stops improving. In contrast, the checkpoint saves the weights from the best epoch for later use. Additionally, all experiments were conducted on a machine with an Intel Xeon CPU family 6, model 85, running at 2000.180 MHz and equipped with 13.021 GB of RAM.

5.2. Datasets

Various collaborative filtering (CF) recommendation methods are evaluated using the standard Amazon dataset [40]. This dataset includes millions of user reviews and ratings on Amazon, spanning from May 1996 to October 2018 Various collaborative filtering (CF) recommendation methods are evaluated using the standard Amazon dataset [40] This dataset includes millions of user reviews and ratings on Amazon, spanning from May 1996 to October 2018[41]. The Amazon dataset comprises 24 product categories. Due to the vast size of the dataset, handling all categories is challenging. Therefore, three product categories from different domains with the largest number of users and reviews were selected to train, develop, and evaluate the proposed method. The chosen datasets are Electronics, CDs & Vinyl, and Movies & TV. These datasets include user profiles and item numerical ratings ranging from 1 to 5, along with the corresponding plaintext reviews. Table 1 summarizes the detailed statistics of these datasets after pre-processing

Table 1. Classification of Amazon Dataset Categories.

Dataset	**User #**	**Item #**	**Reviews#**	**Sparsity**
Electronics	1,92102	63,001	16,60038	98.62
CDs &Vinyl	7,5127	64,443	10,77845	97.77
Movies &TV	1,23633	50,052	16,65265	97.30

5.3. Evaluation Protocol and Metrics

One of the most standard strategies for evaluating RS is predictive accuracy, which measures the predictive ability of these systems on unseen observations. This study uses the standard mean absolute error (MAE) and root-mean-square error (RMSE) to measure predictive accuracy. However, in most recommendation scenarios, users receive a ranked list of Top-N recommended items rather than predictions for specific item ratings. Therefore, classification accuracy methods such as precision, recall, and F1-measure are used to evaluate the predicted values that the recommender system can generate for the user.

5.3.1. Accuracy Metrics

According to [42] the MAE measurement is not effective when the experimental dataset is sparse. In such cases, the Root Mean Squared Error (RMSE) metric is more suitable. The RMSE metric[43] is calculated by considering the mean squared error, making large errors more pronounced

 doi:10.20944/preprints202410.1949.v1

than small ones. Both RMSE and MAE measure the error rate in recommendations. MAE and RMSE are computed as follows[26,42], by Eq (15) and EQ (16) below,

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} |r(u,m) - \hat{r}(u,m)|}{N} \quad (15)$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (r(u,i) - \hat{r}(u,i))^2}{N}} \quad (16)$$

where N denotes the total number of rating overall users in the test set, $r(u,m)$ is the actual rating of the user for item m, $\hat{r}(u,m)$ is the predicted rating for the user u on item m. The lower the value of MAE and RMSE the better the accuracy result. The sparsity level of a dataset can be computed as follows, Eq (17),

$$\mathrm{Sparsity} = 100\% - \mathrm{Density} \quad (17)$$

where, Density is calculated using the following equation, Eq (18) below:

$$\mathrm{Density} = \frac{Number\ of\ Item \times Number\ of\ Users}{Number\ of\ Item \times Number\ of\ Users} \times 100 \quad (18)$$

5.3.2. Classification Metrics

The following presents the classification accuracy metrics the Precision, Recall, and F1-measure present Eq (19), Eq (20) and Eq (21). These metrics are used to evaluate the predicted classification accuracy that RS generates for the target users.

$$\mathrm{Precision}(u) = \frac{|\mathrm{Test}(u) \cap \mathrm{TopN}(u)|}{|\mathrm{TopM}(u)|}. \quad (19)$$

$$\mathrm{Recall}(u) = \frac{|\mathrm{Test}(u) \cap \mathrm{TopN}(u)|}{|\mathrm{Test}(u)|}. \quad (20)$$

$$\mathrm{F1-measure} = \frac{2 \times \mathrm{recall} \times \mathrm{precision}}{\mathrm{recall} + \mathrm{precision}}. \quad (21)$$

where Test (u) the item list of user u in the test set, and TopN(u) is the top N recommended item list for user u.

5.4. Comparisons

This section compares several recommender systems approaches, highlighting their strengths and limitations in the context of our proposed method Bi-GRUCF, as follows:

i. SVD: Singular Value Decomposition (SVD) is a sophisticated collaborative filtering (CF) technique that uses matrix factorization to transform users and items into a latent factor space. This method, widely recognized as a benchmark, effectively predicts ratings even for users with sparse data by leveraging the latent factors [26,44,45].

ii. NMFRS: Recommender Systems with Dynamic Bias (NMFRS) enhances traditional matrix factorization by incorporating non-negative updates and dynamic bias matrices, which improves interpretability and accuracy [25] . It focuses on probabilistic distributions and minimizes differences between observed and predicted ratings.

 doi:10.20944/preprints202410.1949.v1

iii. Word2Vec: This approach captures semantic relationships between items from reviews, which helps alleviate data sparsity issues. It improves over traditional matrix factorization techniques by considering semantic relationships, although it primarily focuses on item relationships [27].

iv. RNN (LSTM): Recurrent Neural Network (RNN) architectures like Long Short-Term Memory (LSTM) are effective for sequential learning tasks involving textual data. They process and retain information over sequences, making them suitable for capturing user preferences in recommendation systems [31]

v. CNN: Convolutional Neural Networks (CNNs) excel in extracting features from complex data, such as reviews or images. When integrated with CF techniques, they enhance the understanding of user preferences and item characteristics, leading to more accurate recommendations. Recent studies have demonstrated the potential of CNNs in improving CF methods for personalized recommendations [33].

In summary, while traditional methods like SVD and NMFRS provide robust frameworks for recommendation systems, deep learning approaches such as Word2Vec, RNNs, and CNNs offer significant advancements by capturing richer contextual information and addressing data sparsity more effectively.

5.5. Results and Discussion

This section investigates the performance of various recommender system approaches on the Amazon Reviews datasets (Electronics, Movies & TV, CDs & Vinyl) characterized by high sparsity levels (above 97%). Sparsity refers to the proportion of missing entries in the user-item interaction matrix.

5.5.1. Performance Comparison on Sparse Data

In recommender systems, sparse data can pose challenges as traditional methods struggle to learn robust relationships between users and items. We employ Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) as evaluation metrics to assess the accuracy of recommendations.

The performance of various recommender system approaches on the three datasets with varying sparsity levels is summarized in Table 1. Here, we focus on the MAE metric to illustrate the impact of sparsity.

The results presented in Table 2, focusing on MAE, reveal a clear trend across all datasets. Traditional collaborative filtering (CF) methods (SVD, NMFRS) consistently achieve higher MAE scores compared to deep learning-based approaches (CNN, Word2Vec, Bi-GRUCF). This suggests that regardless of the specific sparsity level (above 97% in all cases), CF methods struggle to capture the intricate relationships between users and items in sparse data scenarios. Deep learning methods, on the other hand, excel at extracting richer feature representations from limited data points, leading to recommendations with lower absolute errors (MAE).

As shown in Table 2, Bi-GRUCF consistently achieves the lowest Mean Absolute Error (MAE) across all datasets, outperforming traditional methods (SVD, NMFRS) by an average of 55.4% and deep learning approaches like CNN and Word2Vec by an average of 77.8%. This significant improvement suggests that Bi-GRUCF's ability to leverage user reviews through a deep learning architecture grants it a substantial advantage in sparse data scenarios.

Further strengthening this notion is the superior performance of the proposed Bi-GRUCF method. As shown in Table 3, Bi-GRUCF achieves the lowest MAE scores across all datasets, consistently outperforming other approaches. This significant improvement underscores the effectiveness of Bi-GRUCF in leveraging user reviews through a deep learning architecture. By incorporating textual information from reviews, Bi-GRUCF can potentially learn more nuanced user preferences and item characteristics, leading to more accurate recommendations, especially when dealing with limited data.

 doi:10.20944/preprints202410.1949.v1

The high sparsity levels (above 97%) in all three datasets likely play a crucial role in the observed performance differences. Traditional CF methods, which rely primarily on user-item interaction data, might struggle to learn robust representations in such sparse environments. Deep learning approaches, like Bi-GRUCF, which can extract additional information from user reviews, might be more advantageous under these circumstances.

It's important to note the limitations of CNN and Word2Vec approaches included for comparison. As evidenced by the high MAE scores in Table 1, relying solely on item features (CNN) or word embeddings (Word2Vec) might not be as effective as incorporating user reviews through a sequential learning approach like Bi-GRUCF. CNNs might struggle to capture the sequential nature of user preferences within reviews, while Word2Vec, while effective for item relationships, might not explicitly model user behavior.

Table 5. MAE and RMSE results for the three CF models on three datasets.

Dataset	Sparsity#	Method	MAE	RMSE
Electronics	98.62	SVD	0.88	1.22
		NMFRS	0.94	1.25
		CNN	1.72	2.27
		Word2Vec	1.71	2.26
		MCNN	1.71	2.26
		Bi-GRUCF	0.37	0.81
Movies &TV vedio	97.77	SVD	0.78	1.08
		NMFRS	0.79	1.07
		CNN	2.21	2.86
		Word2Vec	2.21	2.86
		CNN	2.2	2.86
		Bi-GRUCF	0.38	0.8
CDs & Vinyl media	97.3	SVD	0.71	1.01
		NMFRS	0.84	1.14
		CNN	1.86	1.82
		Word2Vec	1.86	1.81
		CNN	1.85	1.81
		Bi-GRUCF	0.42	0.84

5.5.2. Evaluation Using Classification Accuracy Metrics

As mentioned earlier, classification accuracy metrics like Recall, Precision, and F-measure provide a more user-centric indicator of recommendation quality compared to MAE and RMSE. Here, items with true numerical ratings above 3.5 are considered relevant, and those below 3.5 are considered non-relevant. These values serve as a general guideline for the following experiments.

In the context of recommendation systems, the Top-N recommendation task involves presenting users with a list of N items, where N represents the number of recommended items. The value of N is crucial as it determines the amount of information presented to users. Based on extensive experimentation, we observed that a value of N=20 consistently yielded the best results across all models and datasets. This suggests that users tend to focus on the top 20 recommendations, making it a critical benchmark for evaluating recommendation quality.

The detailed results of the evaluation for Recall, Precision, and F1-measure are presented in Tables 4, 5 and 6. As shown in the tables, the Bi-GRUCF model consistently outperforms all baseline

 doi:10.20944/preprints202410.1949.v1

approaches (SVD, NMFRS, Word2Vec, RNN, CNN) across all three datasets (Electronics, Movies & TV, CDs & Vinyl).

A notable observation is the consistent superiority of Bi-GRUCF across different recommendation list lengths (N). For instance, in the Electronics dataset (Table 4), Bi-GRUCF achieves a Recall of 97.31% at N=20, significantly higher than the closest competitor, RNN (92.64%). This indicates that Bi-GRUCF is more effective in retrieving relevant items for users in this domain, regardless of the number of recommendations presented. Similar improvements are observed for Precision (Table 3) and F1-measure (Table 5) across all datasets, highlighting the overall superiority of the Bi-GRUCF model in terms of recommendation accuracy.

Table 9. Comparison of the Precision obtained on 3 datasets.

Dataset	SVD	NMFRS	Word2Vec	RNN	CNN	BI-GRUCF
Electronics	83.25	81.79	83.14	92.64	93.28	97.31
Movies & TV	87.16	81.88	87.05	90.7	92.39	96.82
CDs & Vinyl	84.71	82.28	84.6	90.89	92.65	95.95

Table 10. Comparison of the Recall obtained on 3 datasets.

Dataset	SVD	NMFRS	Word2Vec	RNN	CNN	BI-GRUCF
Electronics	84.84	83.52	84.73	93.11	93.44	95.04
Movies & TV	87.72	85.02	87.61	92.79	93.46	94.91
CDs & Vinyl	85.59	83.8	85.48	92.9	92.65	94.05

Table 11. Comparison of the F1-Meausre obtained on 3 datasets.

Dataset	SVD	NMFRS	Word2Vec	RNN	CNN	BI-GRUCF
Electronics	86.49	85.32	86.38	93.23	93.46	92.87
Movies & TV	88.28	88.41	87.05	93.62	93.56	93.08
CDs & Vinyl	86.49	85.37	85.48	92.41	92.4	92.23

The superior performance of Bi-GRUCF stems from its unique architecture, which effectively captures sequential information and context from user reviews. This is achieved through two key components: GRUs and an attention mechanism. GRUs selectively remember or forget information from the sequence, allowing Bi-GRUCF to capture long-term dependencies within user reviews better than RNNs. This leads to a deeper understanding of user preferences and context. Additionally, the attention mechanism focuses on the most informative parts of reviews by dynamically assigning weights to different words or phrases. This selective focus, in contrast to RNNs that process the entire sequence uniformly, enables Bi-GRUCF to extract more meaningful insights from user reviews. Consequently, Bi-GRUCF achieves superior Recall, Precision, and F1-measure across diverse datasets.

Table 6. List of Symbols.

Symbol	Description	Category
R	User-item rating matrix	Variable
U	Set of registered users	Set
I	Set of all possible items	Set

 doi:10.20944/preprints202410.1949.v1

u	User ID	Variable
i	Item ID	Variable
r_ui	Rating of item i by user u	Variable
N(u)	Neighborhood of user u (similar users)	Set
K	Number of nearest neighbors	Variable
pred_rui	Predicted rating of item i for user u	Variable
W	Weight matrix in Bi-GRU layer	Variable
b	Bias vector in Bi-GRU layer	Variable
H	Hidden state vector in Bi-GRU layer	Variable
x(t)	Input word vector at time step t	Variable
f	Forget gate function in GRU unit	Function
i	Input gate function in GRU unit	Function
o	Output gate function in GRU unit	Function
σ	Sigmoid activation function	Function
c(t)	Cell state vector in GRU unit at time step t	Variable
γ	Context vector in Attention layer	Variable
α(t)	Attention weight at time step t	Variable
φ	Embedding function (maps item/user ID to embedding vector)	Function

6. Conclusions

The challenge of data sparsity in CF recommender systems is revisited in this study by leveraging deep learning techniques. Limitations of traditional CF methods, such as SVD and NMFRS, in sparse data environments due to their reliance solely on numerical ratings are highlighted. The proposed approach addresses these limitations by integrating users' review texts through Bi-GRU deep recurrent networks, significantly improving recommendation accuracy. Experimental results on the Amazon dataset demonstrate that the Bi-GRU-based method consistently outperforms other state-of-the-art techniques, achieving substantial gains in both MAE and RMSE. These findings emphasize the importance of incorporating textual data to capture user preferences more effectively.

In conclusion, the potential of deep learning models to enhance CF systems by extracting rich, contextual information from user reviews is underscored by this research. Future research directions include exploring the scalability of this approach and its applicability to other domains, encouraging further investigation into advanced deep learning architectures for recommender systems.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, and writing—original draft preparation were performed by Authors 1, 2, 3, and 4. Visualization, writing—review, and editing were conducted by Authors 4, 5, and 6. All authors have read the manuscript and granted permission for publication.

Acknowledgments: Acknowledgments: The authors extend their gratitude to Universiti Teknologi Brunei for its comprehensive support throughout this study.

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

References

1. X. Wang and S. Kadıoğlu, "Modeling uncertainty to improve personalized recommendations via Bayesian deep learning," *International Journal of Data Science and Analytics,* vol. 16, no. 2, pp. 191-201, 2023.
2. K. Wang, Y. Zhu, H. Liu, T. Zang, and C. Wang, "Learning aspect-aware high-order representations from ratings and reviews for recommendation," *ACM Transactions on Knowledge Discovery from Data,* vol. 17, no. 1, pp. 1-22, 2023.
3. Y. Li, K. Liu, R. Satapathy, S. Wang, and E. Cambria, "Recent developments in recommender systems: A survey," *IEEE Computational Intelligence Magazine,* vol. 19, no. 2, pp. 78-95, 2024.
4. M. Srifi, A. Oussous, A. Ait Lahcen, and S. Mouline, "Recommender systems based on collaborative filtering using review texts—a survey," *Information,* vol. 11, no. 6, p. 317, 2020.
5. U. A. Bhatti, H. Tang, G. Wu, S. Marjan, and A. Hussain, "Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence," *International Journal of Intelligent Systems,* vol. 2023, pp. 1-28, 2023.
6. S.-H. Noh, "Analysis of gradient vanishing of RNNs and performance comparison," *Information,* vol. 12, no. 11, p. 442, 2021.
7. G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review,* vol. 53, no. 8, pp. 5929-5955, 2020.
8. H. Israr, S. A. Khan, M. A. Tahir, M. K. Shahzad, M. Ahmad, and J. M. Zain, "Neural Machine Translation Models with Attention-Based Dropout Layer," *Computers, Materials & Continua,* vol. 75, no. 2, 2023.
9. C. Zhang *et al.*, "Multi-aspect enhanced graph neural networks for recommendation," *Neural Networks,* vol. 157, pp. 90-102, 2023.
10. K. Bi, Q. Ai, and W. B. Croft, "Learning a fine-grained review-based transformer model for personalized product search," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 123-132.
11. C. Liu, T. Jin, S. C. H. Hoi, P. Zhao, and J. Sun, "Collaborative topic regression for online recommender systems: an online and Bayesian approach," *Machine Learning,* vol. 106, no. 5, pp. 651-670, 2017/05/01 2017, doi: 10.1007/s10994-016-5599-z.
12. Y. Bao, H. Fang, and J. Zhang, "Topicmf: simultaneously exploiting ratings and reviews for recommendation," presented at the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, Québec, Canada, 2014, 2-8.
13. C. Gao *et al.*, "A survey of graph neural networks for recommender systems: Challenges, methods, and directions," *ACM Transactions on Recommender Systems,* vol. 1, no. 1, pp. 1-51, 2023.
14. M. Ibrahim, I. S. Bajwa, N. Sarwar, F. Hajjej, and H. A. Sakr, "An intelligent hybrid neural collaborative filtering approach for true recommendations," *IEEE Access,* 2023.
15. V. R. Yannam, J. Kumar, K. S. Babu, and B. Sahoo, "Improving group recommendation using deep collaborative filtering approach," *International Journal of Information Technology,* vol. 15, no. 3, pp. 1489-1497, 2023.
16. N. Alharbe, M. A. Rakrouki, and A. Aljohani, "A collaborative filtering recommendation algorithm based on embedding representation," *Expert Systems with Applications,* vol. 215, p. 119380, 2023.
17. L. Huang, C.-R. Guan, Z.-W. Huang, Y. Gao, C.-D. Wang, and C. P. Chen, "Broad recommender system: An efficient nonlinear collaborative filtering approach," *IEEE Transactions on Emerging Topics in Computational Intelligence,* 2024.
18. F. Abbasi and A. Khadivar, "Collaborative filtering recommendation system through sentiment analysis," *Turkish Journal of Computer and Mathematics Education,* vol. 12, no. 14, pp. 1843-1853, 2021.
19. G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Trans. on Knowl. and Data Eng.,* vol. 17, no. 6, pp. 734-749, 2005, doi: 10.1109/tkde.2005.99.
20. M. Srifi, A. Oussous, A. A. Lahcen, and S. Mouline, "Recommender Systems Based on Collaborative Filtering Using Review Texts—A Survey," *Information,* vol. 11, no. 6, p. 317, 2020, doi: doi.org/10.3390/info11060317.
21. J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks,* vol. 61, pp. 85-117, 2015/01/01/ 2015, doi: 10.1016/j.neunet.2014.09.003.
22. Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks,* vol. 5, no. 2, pp. 157-166, 1994.
23. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural computation,* vol. 9, no. 8, pp. 1735-1780, 1997, doi: 10.1162/neco.1997.9.8.1735.
24. K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078,* 2014.
25. W. Song and X. Li, "A Non-Negative Matrix Factorization for Recommender Systems Based on Dynamic Bias," in *International Conference on Modeling Decisions for Artificial Intelligence*, 2019: Springer, pp. 151-163.

26. S. Wang, G. Sun, and Y. Li, "SVD++ Recommendation Algorithm Based on Backtracking," *Information*, vol. 11, no. 7, doi: 10.3390/info11070369.
27. L. Vuong Nguyen, T.-H. Nguyen, J. J. Jung, and D. Camacho, "Extending collaborative filtering recommendation using word embedding: A hybrid approach," *Concurrency and Computation: Practice and Experience,* vol. 35, no. 16, p. e6232, 2023, doi: https://doi.org/10.1002/cpe.6232.
28. R. M. D'Addio and M. G. Manzato, "A sentiment-based item description approach for kNN collaborative filtering," presented at the Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, 2015. [Online]. Available: https://doi.org/10.1145/2695664.2695747.
29. Ayman S. Ghabayen and Basem H. Ahmed, "Enhancing collaborative filtering recommendation using review text clustering," *Jordanian Journal of Computers and Information Technology (JJCIT),* vol. 7, no. 9, 2021, doi: 0.5455/jjcit.71-1609969782.
30. S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys,* vol. 52, no. 1, pp. 1-38, 2019.
31. A. Nilla and E. Setiawan, "Film Recommendation System Using Content-Based Filtering and the Convolutional Neural Network (CNN) Classification Methods," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika,* vol. 10, p. 17, 02/12 2024, doi: 10.26555/jiteki.v9i4.28113.
32. D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 233-240, doi: 10.1145/2959100.2959165.
33. L. Zheng, V. Noroozi, and P. S. Yu, "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, United Kingdom, 2017, in WSDM '17, pp. 425-434, doi: 10.1145/3018661.3018665.
34. R.-C. Chen and Hendry, "User Rating Classification via Deep Belief Network Learning and Sentiment Analysis," *IEEE Transactions on Computational Social Systems,* vol. 6, no. 3, pp. 535-546, 2019, doi: 10.1109/TCSS.2019.2915543.
35. B. H. Ahmed and A. S. Ghabayen, "Review rating prediction framework using deep learning," *Journal of Ambient Intelligence and Humanized Computing,* vol. 13, no. 7, pp. 3423-3432, 2022/07/01 2022, doi: 10.1007/s12652-020-01807-4.
36. J. H. Wang, M. Norouzi, and S. M. Tsai, "Multimodal Content Veracity Assessment with Bidirectional Transformers and Self-Attention-based Bi-GRU Networks," in *2022 IEEE Eighth International Conference on Multimedia Big Data (BigMM)*, 5-7 Dec. 2022 2022, pp. 133-137, doi: 10.1109/BigMM55396.2022.00030.
37. J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532-1543.
38. Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," presented at the Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona,Spain, 2016.
39. P. Zhou *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Berlin, Germany, 2016: Association for Computational Linguistics, pp. 207-212, doi: 10.18653/v1/P16-2034.
40. J. McAuley, 2021, "Recommender Systems and Personalization Datasets," [Online] http://jmcauley.ucsd.edu/data/amazon/)
41. J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019: Association for Computational Linguistics, pp. 188-197, doi: 10.18653/v1/D19-1018.
42. M. R. McLaughlin and J. L. Herlocker, "A collaborative filtering algorithm and evaluation metric that accurately model the user experience," presented at the Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, Sheffield, United Kingdom, 2004. [Online]. Available: https://doi.org/10.1145/1008992.1009050.
43. N. Good *et al.*, "Combining collaborative filtering with personal agents for better recommendations," presented at the Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference, Orlando, Florida, USA, 1999.
44. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender system-a case study," Minnesota University Minneapolis Department of Computer Science, In Proceedings of the ACM WebKDD 2000 Web Mining for E-Commerce Workshop, 2000.
45. P. Symeonidis and A. Zioupos, *Matrix and Tensor Factorization Techniques for Recommender Systems*. Springer International Publishing, 2016.

 doi:10.20944/preprints202410.1949.v1