ITC 2/53

Information Technology and Control
Vol. 53 / No. 2 / 2024
pp. 570-583
DOI 10.5755/j01.itc.53.2.35689

Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation: An Innovative Approach Integrating Plot Summary Information and Contrastive Learning Strategy

Received 2023/11/22 | Accepted after revision 2024/04/29

HOW TO CITE: Wang, M., Hu, X., Xie, P., Du, Y. (2024). Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation: An Innovative Approach Integrating Plot Summary Information and Contrastive Learning Strategy. *Information Technology and Control,* 53(2), 570-583. https://doi.org/10.5755/j01.itc.53.2.35689

Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation: An Innovative Approach Integrating Plot Summary Information and Contrastive Learning Strategy

Mingye Wang
School of Automation Science and Electrical Engineering, Beihang University,
Xue Yuan Lu Str. 37, Beijing, China; e-mail: marcel0829@buaa.edu.cn

Xiaohui Hu
Institute of Software, Chinese Academy of Sciences, Zhong Guan Cun Nan Si Jie Str. 4, Beijing, China

Pan Xie, Yao Du
School of Automation Science and Electrical Engineering, Beihang University,
Xue Yuan Lu Str. 37, Beijing, China

Corresponding author: marcel0829@buaa.edu.cn

This study introduces a novel movie recommender system utilizing a Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation (SeVGAER) architecture. The system harnesses additional information from movie plot summaries scraped from the internet, transformed into semantic vectors via a large language model. These vectors serve as supplementary features for movie nodes in the SeVGAER-based recommender. The system incorporates an encoder-decoder structure, operating on a user-movie bipartite graph, and employs GraphSAGE convolutional layers with modified aggregators as the encoder to extract latent representations of the nodes, and a Multi-Layer Perceptron (MLP) as the decoder to predict ratings with additional graph-based features. To address overfitting and improve model generalization, a novel training strategy is introduced. We employ a random data splitting approach, dividing the dataset into two halves for each training instance. The model then generates outputs on each half of the data, and a new loss function is introduced to ensure consistency between these two outputs, a strategy that can be seen as a form of contrastive learning. The model's performance is optimized using a combination of this new contrastive loss, graph reconstruction loss, and KL divergence loss. Experiments conducted on the Movielens100k dataset demonstrate the effectiveness of this approach in enhancing movie recommendation performance.

KEYWORDS: Variational graph autoencoder, recommender system, graph neural network, semantic vectors, deep learning, contrastive learning.

1. Introduction

The proliferation of social media platforms, streaming services, and e-commerce websites in the digital age has revolutionized the way humans interact with a wide range of content [30]. As these platforms attract an increasingly diverse user base, the demand for robust personalized movie recommendation systems has reached unprecedented levels. These systems have become a cornerstone for enhancing users' engagement and satisfaction [15].

The movie recommender system plays a crucial role in various streaming platforms, greatly improving user experience by suggesting movies that align with their interests. While traditional recommender systems primarily rely on user-movie interaction data, such as ratings or viewing history, the incorporation of additional movie information can further enhance their effectiveness.

In recent years, the rise of deep learning has prompted numerous studies exploring the use of neural networks in recommender systems. Among these, graph-based models have garnered significant attention due to their ability to capture complex relationships in interaction data. However, the potential of these models can be further harnessed by integrating additional movie information, such as plot summaries, into the graph structure.

In this context, our paper introduces a novel approach to movie recommender systems by leveraging a Variational Graph Autoencoder (VGAE), plot summary semantic vectors, and graph structural features such as node degrees and adjacency matrix. The VGAE, a generative model that learns latent representations of nodes in a graph, allows for the capture of complex user-movie interactions, offering a pathway for more personalized movie recommendations. Moreover, to address overfitting and improve model generalization, we introduce a novel training strategy involving random data splitting and a new loss function that ensures consistency between model outputs. This strategy, inspired by contrastive learning, helps to further enhance the model's performance.

We term this adapted model as Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation (SeVGAER). The novelty of our work extends beyond the mere application of VGAE in movie recommendations. We innovate further by adapting the original VGAE architecture to better suit our specific use case. Specifically, we replace the Graph Convolutional Networks (GCN) in the encoder, which are unsuitable for bipartite graphs and exhibit limited scalability, with the more scalable GraphSAGE convolutional layers. This modification enables the extraction of more meaningful latent representations of the nodes, thereby enhancing the quality of recommendations. Additionally, we remake the decoder of VGAE by replacing the simple inner product operation with a Multi-Layer Perceptron (MLP), which integrates additional graph-based features, including node degrees and adjacency matrix, allowing for a more flexible and complex interaction between user and movie latent representations.

Another significant innovation of our SeVGAER model is the integration of additional information as features. Unlike traditional models that utilize only user-movie interaction data, our system leverages movie plot summaries, transformed into semantic vectors using a large language model, and graph structural information, such as node degrees and adjacency matrix. This multi-source feature integration provides additional context and nuance that significantly enhance the recommender system's performance.

In essence, this paper presents a novel, enhanced movie recommender system that employs a modified VGAE structure, integrates supplementary information, and applies a unique training strategy. This combination improves the system's performance and sets a new precedent for future research in the field of movie recommender systems.

The paper is organized as follows: Section 2 discusses related work, Section 3 presents the methodology, Section 4 reports experiments and results, and Section 5 concludes the paper and outlines future research directions.

2. Related Work

2.1. Recommender Systems

Recommender systems can be categorized into three types based on their working method: content-based, collaborative filtering (CF)-based, and hybrid systems [1, 32].

Content-based recommendation algorithms utilize auxiliary information such as texts, images, and videos to provide personalized recommendations by comparing items and users. These algorithms focus on user characteristics and item attributes, suggesting items similar to users' past interactions [34]. For example, a movie website employing a content-based approach considers each user's viewing history and search preferences to recommend movies.

CF models aim to provide personalized recommendations by leveraging users' rating history for items. Instead of relying solely on a user's few ratings, CF introduces the concept of a "neighborhood," which enables recommendations based on community ratings. It suggests items that users are likely to enjoy and predicts their potential interests. This CF recommendation algorithm utilizes a rating matrix that incorporates the item ratings of a user's community neighbors [34]. The critical step in CF involves computing the similarity between users or items [25]. Notably, CF encompasses various algorithms, including Matrix Factorization and Deep Learning, which use known community ratings to predict unknown ones. However, CF-based recommendation suffers from data sparsity and cold start problems [27].

Hybrid models are recommender systems that integrate multiple recommendation strategies [4].

In addition to these three types of recommender systems, there have been other parallel lines of research that utilize different types of information or have different purposes. For example, some researchers have approached recommendation as a sequential prediction task by considering the temporal dynamics of users' behaviors [7, 31]. Chen and Huang [5] further proposed a model that integrates this idea with traditional methods, resulting in improved performance. This perspective allows us to classify recommender systems into static models and dynamic models based on whether they incorporate dynamic information.

In contrast, our proposed model, SeVGAER, focuses on integrating plot summary information and a contrastive learning strategy into the recommendation process. While our model and the dynamic models belong to different sub-domains of recommendation systems, they both contribute to the broader goal of providing better personalized movie recommendations.

2.2. Autoencoder-based Recommender Models

Autoencoder-based recommender models have gained significant attention in recent years due to their ability to effectively capture complex patterns and latent representations in user-item interaction data. These models leverage the power of autoencoders, a type of neural network that is trained to reconstruct its input, to learn low-dimensional representations of users and items. By doing so, they can uncover meaningful latent features that are often not explicitly available in the original data.

AutoRec [26] is one of the successful autoencoders that takes user partial vectors or item partial vectors as input and aims to reconstruct them in the output layer. Darban et al. [6] developed a similarity map to extract relevant user feature information, which they then integrated with the user's basic attributes such as age and gender. They further employed an autoencoder to reduce the dimensionality of the extracted features. By applying a clustering method, they utilized this model, named GHRS, to make personalized recommendations for users.

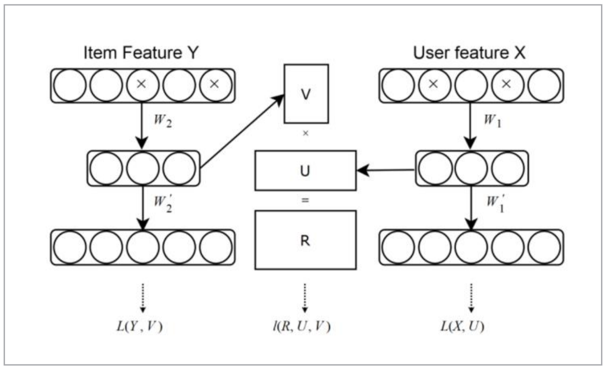
Autoencoder-based recommender models have ability to handle the sparsity and high dimensionality of typical recommendation datasets, and can effectively encode the missing values and fill in the gaps by learning a dense representation of the data.

In addition to handling sparsity, autoencoder-based recommender models also offer flexibility in modeling user preferences and the ability to incorporate side information, such as item attributes or user demographics, into the recommendation process.

Figure 1 illustrates a classic structure of recommender models based on autoencoder [32].

Figure 1

Classic deep collaborative filtering framework based on autoencoder



2.3. Graph Neural Network

In recent years, graph neural network (GNN) variants have exhibited remarkable performance in various tasks involving graph data, including physical systems [3], protein structure [8], and knowledge graphs [9].

In the context of graph data, the fundamental concept behind GNNs is to iteratively gather feature information from neighboring nodes and incorporate this aggregated information into the current representation of the central node during the propagation process. GNNs employ multiple propagation layers that encompass aggregation and update operations.

Graph convolutional network (GCN) [16] and GraphSAGE (Graph Sample and Aggregated) [10] are two popular graph neural network layers used for processing graph data.

GCN is a type of graph neural network layer that operates on a graph structure. It leverages a neighborhood aggregation strategy to propagate information from neighboring nodes to the central node. This process is achieved by performing a linear transformation on the feature vectors of neighboring nodes and combining them with the central node's feature vector.

GraphSAGE, on the other hand, is a graph neural network layer that focuses on sampling and aggregating information from the neighborhood nodes. It employs a sampling mechanism to select a subset of nodes from the neighborhood and aggregates their feature vectors. This aggregated information is then combined with the central node's feature vector to update its representation.

While GCN has proven effective in many applications, it has certain limitations when it comes to scalability. Specifically, GCN applies the same transformation to all nodes in the graph, regardless of their unique context or characteristics. This not only leads to high computational costs for large-scale graphs but also limits the model's flexibility in learning diverse node representations.

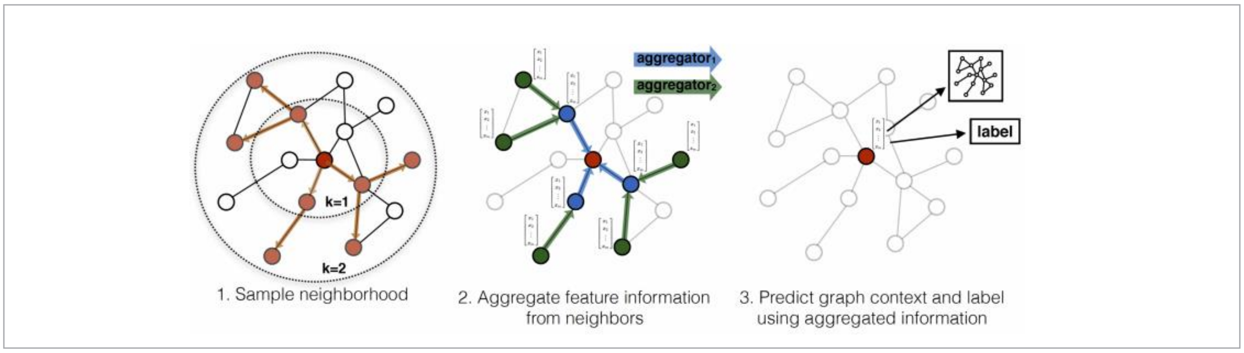
GraphSAGE, by contrast, offers a more scalable and flexible solution. Unlike GCN, GraphSAGE employs a sampling mechanism that selects a subset of nodes from the neighborhood, which significantly reduces the computational cost for large-scale graphs. Moreover, GraphSAGE updates the representation of each node based on its local neighborhood, allowing for more diverse and context-specific node representations. This feature makes GraphSAGE particularly well-suited for tasks involving large and heterogeneous graphs, such as movie recommendation systems.

Figure 2 illustrates the workflow of the GraphSAGE network.

By adopting GraphSAGE in our proposed SeVGAER model, we can effectively address the scalability issue, allowing for more efficient processing of user-movie interaction data and more personalized movie recommendations.

Figure 2

Illustration of GraphSAGE


k=1
k=2
1. Sample neighborhood
aggregator1
aggregator2
2. Aggregate feature information from neighbors
label
3. Predict graph context and label using aggregated information

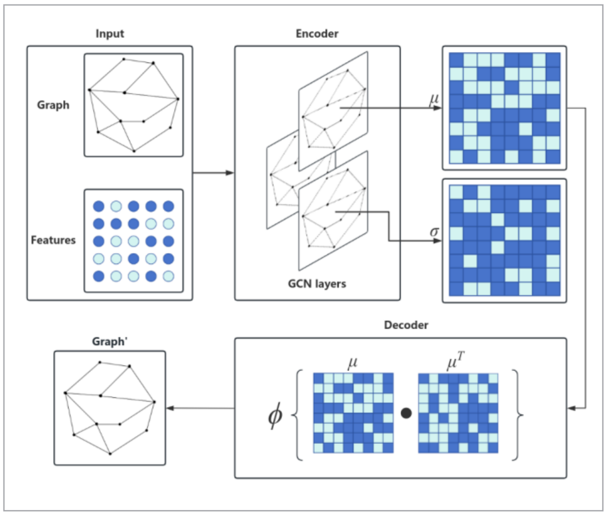
Variational Graph Autoencoder (VGAE) [17] is another popular graph neural network model that focuses on learning low-dimensional representations of graph data. VGAE combines the power of graph convolutional layers with a variational autoencoder framework. It aims to capture the underlying latent structure of the graph by reconstructing the adjacency matrix. VGAE utilizes an encoder network to map the graph's nodes into a lower-dimensional latent space, and a decoder network to reconstruct the adjacency matrix based on the learned latent representations.

One of the key innovations of VGAEs is their utilization of variational inference, which offers a probabilistic approach to model the inherent uncertainty in the process of encoding graph data. Instead of directly embedding inputs, VGAE assumes that the latent representations follow a normal distribution and generates the mean and variance of the latent representation accordingly. In addition to the reconstruction loss, VGAE incorporates a Kullback-Leibler (KL) divergence loss that quantifies the discrepancy between the learned distribution of the latent variables and the normal distribution. By incorporating these components, VGAE effectively learns meaningful representations that capture both the topological and relational information of the graph. Notably, VGAE has demonstrated promising results in various graph-related tasks, including link prediction, node clustering, and graph generation.

Figure 3 shows the architecture of VGAE.

Figure 3

Illustration of VGAE architecture



Most autoencoder-based recommender systems utilize autoencoders to reconstruct input users or items and learn their latent representations. However, in many cases, valuable information is embedded within the interaction graph itself. This motivates us to adopt the VGAE model, which directly reconstructs the graph. However, the original VGAE model is not specifically designed to handle user-item bipartite graphs, as it cannot effectively handle the heterogeneity present in such graphs using GCNs. To address this limitation and improve the performance of movie recommender systems, we propose the SeVGAER model. By modifying the VGAE architecture and integrating semantic features and graph structural information, SeVGAER offers a solution to the recommendation problem in the context of user-item bipartite graphs.

3. Methodology

3.1. Problem Definition

Movie recommendation is a quintessential problem in the field of recommender systems, which aims to predict the ratings that a user would give to unseen movies based on historical interactions. The goal is to provide users with personalized movie recommendations, which could significantly enhance the user experience.

Formally, let us denote $U = \{u_1, u_2, ..., u_m\}$ as the set of users and $M = \{m_1, m_2, ..., m_n\}$ as the set of movies. Each user has a history of interactions with some movies in M. These interactions could take the form of explicit feedback (e.g., ratings) or implicit feedback (e.g., viewing history). The historical interactions can be represented as a bipartite graph $G = (U, M, E)$, where E represents the edges or interactions between users and movies. Each edge e_{ij} =(ui, mj)$\in E$ is associated with a weight w_{ij}, which represents the rating given by user u_i to movie m_j. The aim is to predict the unseen interactions or ratings by learning from the known interactions in the graph G:

$$\tilde{y}_{um} = f(u, m), \qquad (1)$$

where score function f can be dot product, cosine, MLPs, and so forth, and $\tilde{y}_{um}$ denotes the preference score for u on m, which is usually presented in probability.

3.2. Overview of the Proposed Approach

Our proposed SeVGAER model sets out to address the limitations of traditional movie recommender systems by effectively integrating graph-based learning with additional movie information, such as plot summary semantic vectors and graph structural features. The overall structure of our model is depicted in Figure 4.

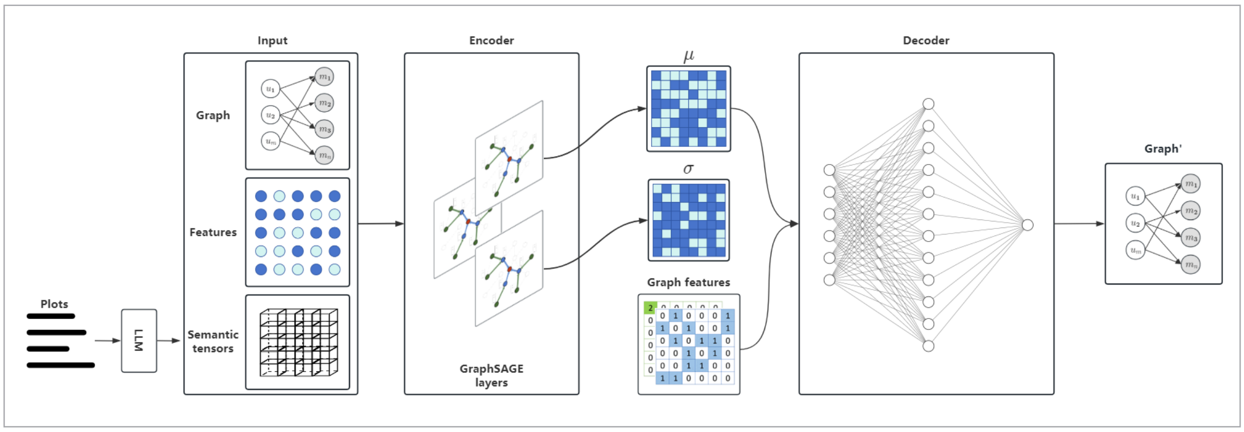
SeVGAER is built upon the foundation of graph-based generative models, enabling it to capture the intricate relationships within user-movie interaction data represented as a bipartite graph.

To enhance the performance and scalability of our model, we have made several modifications to the conventional graph autoencoder architecture. In particular, we replaced the unsuitable and less scalable Graph Convolutional Networks (GCN) in the encoder with GraphSAGE convolutional layers. This adjustment not only enhances the scalability of our model but also enables the extraction of more meaningful latent node representations. In the decoder, we opt for a Multi-Layer Perceptron (MLP) instead of the standard inner product operation, allowing for a more complex interaction between user and movie latent representations.

One of the most innovative aspects of our approach is the introduction of a unique training strategy. We randomly split the dataset for each training instance and ensure consistency between model outputs on each half. This strategy, inspired by contrastive learning, further strengthens the performance of our model.

In sum, our SeVGAER model presents a comprehensive, enhanced solution to movie recommendation by

Figure 4
Overall structure of SeVGAER



effectively integrating user-movie interaction data, additional movie information, and a unique training strategy within a graph-based model. The subsequent sections will delve deeper into the specifics of the SeVGAER model, our innovative modifications, and the process of obtaining and integrating plot summary semantic vectors.

3.3. SeVGAER

This section will detail the function of our model: data preprocessing, encoder design, and decoder design.

3.3.1. Data Preprocessing

The first step in our approach involves preprocessing the input data, mainly movie plot texts. We use a Large Language Model (LLM) to embedding all plots into fixed length vectors. These semantic vectors are then concatenated with original movie features to form the complete movie node features.

For example, a movie node m's features are transformed from original h^{m} to $h^{m} \oplus S_{m}$ where S_{m} is the plot text of m and $\oplus$ is the concatenation operation.

3.3.2. Encoder

The encoder of the proposed SeVGAER model plays a pivotal role in extracting meaningful representations from both user and movie nodes. The encoder is structured into two analogous sections, each specifically designed to handle two types of relationships: user rating movie and movie being rated by user.

Each section of the encoder, corresponding to each relationship type, comprises three GraphSAGE convolutional layers. GraphSAGE operates based on the following formula:

$$h_{N(v)}^{k} = AGGREGATE_{k}(h_{x}^{k-1}, \forall x \in N(v)) \quad (2)$$

$$h_{v}^{k} = \sigma\left(W^{k} \cdot CONCAT\left(h_{v}^{k-1}, h_{N(v)}^{k}\right)\right). \quad (3)$$

Here, h_{x}^{k-1} refers to the feature vector of the node x at the (k-1)th layer, represents the neighborhood of node v, $AGGREGATE_{k}$ is the aggregation function at the kth layer which we could customize, W^{k} is the weight matrix at the kth layer, and σ is an activation function. The first layer of GraphSAGE operates on the input features of the nodes, converting them into a preliminary form. The original GraphSAGE tries three aggregation methods including mean operator, LSTM and pooling.

Sun et al. [28] observed that some recent models assigned identical scores to multiple items, severely limiting the model's generalization ability on diverse datasets. They traced this issue back to the Rectified Linear Unit (ReLU) activation function, which zeros out an excessive number of neurons.

In the context of our movie recommendation system, this would translate to similar scores being assigned to distinctly different movies, limiting the diversity and personalization of the recommendations. Therefore, to address this issue and enhance the generalization ability of our model, we decided to use the Gaussian Error Linear Unit (GELU) activation function in our first GraphSAGE layer instead of the commonly used ReLU. Its expression is shown as follow:

$$GELU(x) = \frac{1}{2}x\left(1 + erf\left(\frac{x}{\sqrt{2}}\right)\right), \quad (4)$$

where $erf(\cdot)$ is the Gauss error function.

GELU is a recently proposed activation function that has gained popularity in deep learning. It is a smooth approximation of the ReLU activation function. The GELU function has a non-monotonic, sigmoid-like shape that provides a better gradient flow during backpropagation compared to ReLU. The GELU function has been shown to improve the performance of neural networks on a variety of tasks, including item recommendation and classification. By adopting the GELU function in our model, we aim to improve the diversity and personalization of our movie recommendations, thereby enhancing the overall performance of our system.

Following this, the second GraphSAGE layer generates the mean of a distribution from these transformed features, essentially forming the latent representation of the nodes. Meanwhile, the third GraphSAGE layer computes the standard deviation of this distribution.

The design of the encoder, inspired by the VGAE framework, considers the latent representation as a distribution defined by its mean μ and standard deviation σ, rather than a single point.

One distinguishing aspect of our approach lies in the modification of the GraphSAGE aggregation function. While the original GraphSAGE model offers three aggregation methods - mean operator, LSTM, and pooling - we have customized our aggregator to better fit our task. Specifically, we employ a multi-aggregation method that merges three softmax aggregations, initially proposed in [14], with varying learnable temperatures for the first hidden layer and the mean layer.

The softmax aggregator, defined by the following formula, controls the softness of the softmax during aggregation over a set of features χ using a temperature parameter t. The softmax aggregator serves as a generalized mean-max aggregation function and adjusts between mean or max aggregator based on the temperature value (operating as a mean for small t and as a max for larger t). This makes it more expressive than the original aggregation functions and more flexible with a learnable t.

$$softmax(\chi|t) = \sum_{h_x \in \chi} \frac{\exp(t \cdot h_x) \cdot h_x}{\sum_{h_x \in \chi} \exp(t \cdot h_x)}. \tag{5}$$

Moreover, to further enhance the encoder's ability, we concatenate multiple aggregators.

For the standard deviation layer, we utilize a softmax aggregator combined with a standard deviation aggregator, which informs the encoder of variables' deviation when predicting embedding deviation.

$$\begin{aligned} std(\chi) &= \sqrt{var(\chi)} \\ &= mean(\{h_x^2 : h \in \chi\}) - mean(\chi)^2. \end{aligned} \tag{6}$$

This approach enables us to capture various aspects of the data, thereby enhancing the robustness and accuracy of our model.

In essence, the encoder effectively converts the input features of both user and movie nodes into encapsulated vectors of the same dimensions. The resulting architecture allows the model to harness the inherent semantics of the data, making it a robust tool for movie recommendation tasks.

3.3.3. Decoder

In the decoder component of our model, we deviate from the traditional approach of employing an inner product operation. Instead, we utilize a MLP to predict user-to-movie ratings. This modification fosters a more intricate and flexible interaction between user and movie latent representations, ultimately resulting in more precise rating predictions.

To further capture the nuanced interplay between user and movie nodes, the decoder accepts not only the latent representations of these two types of nodes but also their differences and element-wise products. This approach enables the model to incorporate both the individual characteristics of users and movies, and their relational dynamics.

Additionally, we incorporate graph features of the nodes as supplementary information. These graph features are composed of two parts: the degree of the nodes and their adjacency vectors. The inclusion of these features adds another layer of context, providing the model with a more comprehensive view of the graph structure.

The complete decoder function can be represented as:

$$\begin{aligned} \tilde{y}_{um} = MLP(&z_u \oplus z_m \oplus (z_u - z_m) \oplus (z_u \odot z_m) \\ &\oplus D_u \oplus D_m \oplus Adj_u \oplus Adj_m), \end{aligned} \tag{7}$$

where z_u and z_m are the latent representations of user and movie nodes, respectively, D_u and D_m represent the degree of user and movie nodes, and Adj_u and Adj_m are their adjacency vectors. The MLP is trained to predict the rating y_{um} from user u to movie m based on this input vector.

Our decoder leverages a specific architecture inspired by the "expansion-and-contraction" design. This architecture is characterized by an initial expansion of the output dimensionality, followed by a contraction. This design is prominent in the Transformer model's fully connected layers and has proven to be effective [29].

The first layer of the MLP doubles the dimensionality of the input, enabling the model to explore a wide array of possible feature interactions. This "expansion" phase provides the MLP with a larger hypothesis space to capture complex patterns and dependencies. Same to the encoder, we choose GELU as activation function of the first layer.

Following this, the second layer reduces the dimensionality to one which is the output rating, consolidating the information learned in the expansion phase. This "contraction" phase helps to synthesize the expanded information, focusing on the most salient features and interactions.

This design approach allows the MLP to effectively capture intricate relationships in the data, thereby enhancing the quality of the movie recommendations generated by our model.

3.3.4. Training

In this section, we delve into the details of the training process of the SeVGAER model. The training process is pivotal to the successful operation of our model, as it determines how well the model can capture and generalize the intricate user-movie interactions in the data.

The training begins with the encoder generating the mean and standard deviation of the latent representation of nodes. These statistical descriptors serve as a summary of the distributions of node features in the latent space. They are then passed on to the decoder, which undertakes the task of reconstructing the original user-movie bipartite graph. This reconstruction is achieved by predicting the ratings of movies by users based on the statistical descriptors provided by the encoder.

The effectiveness of the reconstruction process is evaluated using a loss function, which comprises three parts: the reconstruction loss, the KL divergence loss, and a novel contrastive loss.

The reconstruction loss is essentially a measure of the accuracy of the ratings predicted by the decoder. It measures the deviation between the predicted ratings and the actual ones, defined by the mean square error:

$$\mathcal{L}_{rec} = \frac{1}{N} \sum_{\forall (u,m) \in E} (\tilde{y}_{um} - y_{um})^2, \tag{8}$$

where y_{um} represents the ground truth rating of user u on movie m.

The KL divergence loss, on the other hand, serves a different purpose. It is a fundamental component of variational autoencoders, including our SeVGAER model, and ensures that the distribution of embeddings outputted by the model's encoder adheres to a normal distribution.

This adherence to a normal distribution is based on the assumption of variational autoencoders that the latent space follows a Gaussian distribution. This assumption simplifies the computation and allows for an efficient exploration of the latent space during the decoding process.

Moreover, the KL divergence loss acts as a regularization term in the loss function, preventing overfitting by discouraging the model from learning too complex or specific representations that might not generalize well. It encourages the model to learn more robust and generalized representations that capture the overall structure of the data rather than the specific details of each training example.

The KL divergence loss is computed as follows:

$$\mathcal{L}_{KL} = -\frac{1}{2N}\left(\sum_{i=1}^{K}\left[1 + 2log\sigma_i - \mu_i^2 - \sigma_i^2\right]\right), \tag{9}$$

where μ and σ are the mean and standard deviation of the embeddings distribution, outputted by the model's encoder.

To further bolster the model's performance and enhance its ability to generalize, we introduce a novel training strategy inspired by contrastive learning. In this approach, we randomly split the dataset for each training instance and introduce a contrastive loss to ensure consistency between model outputs on each half. This strategy forces the model to learn more robust and generalized representations that hold true across different subsets of the data. The contrastive loss is defined as the RMSE between outputs generated by each half of dataset:

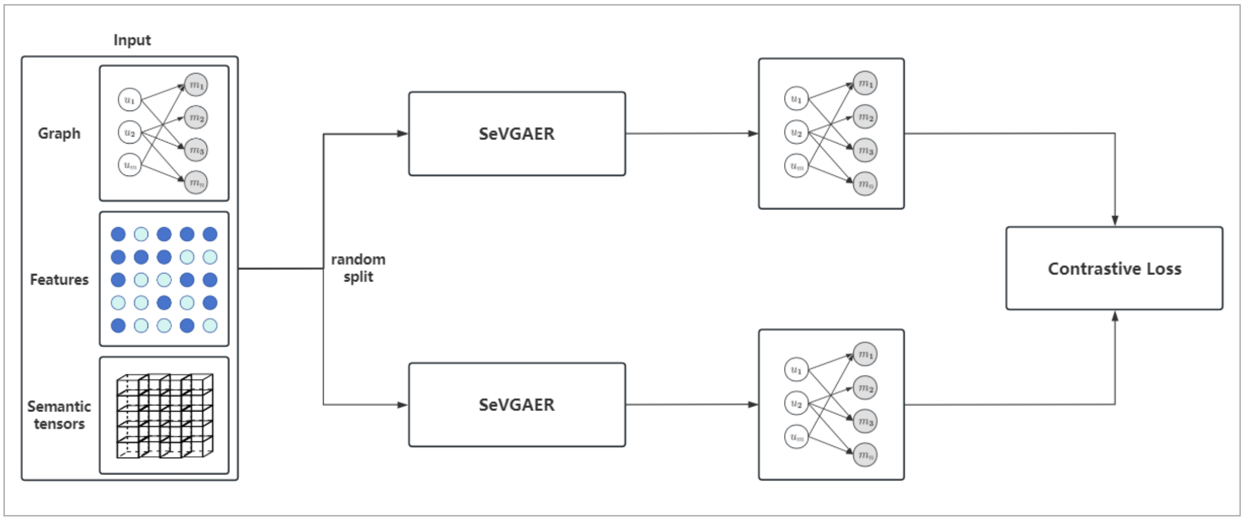
$$\mathcal{L}_{ctt} = \frac{1}{N}\sum (y_{um}^{(1)} - y_{um}^{(2)})^2. \tag{10}$$

This contrastive learning process is illustrated in Figure 5.

The total loss, which guides the training of the model, is then the weighted sum of the reconstruction loss, KL divergence loss, and the contrastive loss. By minimizing this comprehensive loss function, the SeVGAER model can effectively learn from the data and yield accurate and personalized movie recommendations.

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{KL}\mathcal{L}_{KL} + \lambda_{ctt}\mathcal{L}_{ctt}. \tag{11}$$

Figure 5
Contrastive learning process



4. Experiments and Analysis

This chapter presents the experimental evaluation of our proposed SeVGAER model. The results of the experiments are used to validate the effectiveness of our approach.

4.1. Experimental Setup

In this section, we describe the dataset used for the experiments, the evaluation metrics, and the baseline methods for comparison.

The experiments were conducted on the enriched MovieLens 100k dataset [12], which includes user-movie interaction data and additional movie plot summaries. MovieLens 100K contains 100,000 ratings $y \in \{1, 2, 3, 4, 5\}$, 1,682 movies (items) rated by 943 users. We use the official split of the dataset to train and evaluate our model.

We used widely accepted evaluation metric in the field of recommender systems, the Root Mean Square Error (RMSE), to evaluate the performance of our model.

For comparison purposes, we selected several state-of-the-art methods as baseline models. These models represent a variety of recommendation approaches, including collaborative filtering methods, deep learning-based methods, etc..

The hyperparameters of SeVGAER during training are listed in Table 1.

Table 1
The hyperparameters

Hyperparameter	Value
Optimizer	AdamW
Learning rate	0.001
λ_{rec}	0.5
λ_{KL}	6.5
λ_{ctt}	0.5
Dropout rate	0.85
Encoder dim	96
Latent dim	60

Here, dropout rate is used in decoder's hidden layer. "Latent dim" means the dimension of node latent representation or node embedding - the encoders' outputs.

The LLM we used to embedding input plot texts is all-MiniLM-L6-v2 proposed by [22, 23], a pretrained language model that maps sentences or paragraphs to a 384 dimensional dense vector space.

4.2. Experimental Results

This section highlights the experimental results of our proposed SeVGAER model, as well as the baseline methods. The results are outlined in Table 2 and will be thoroughly discussed in the following sections.

Table 2
Experimental results

MODEL	RMSE
Baseline COFILS [2]	0.892
Kernel PCA COFILS [2]	0.898
Slope One [20]	0.937
Regularized SVD [21]	0.989
SVD++ [18]	0.903
FM [24]	0.909
Non-Negative Matrix Factorization [19]	0.944
NFM [13]	0.910
AutoSVD [33]	0.901
AutoSVD++ [33]	0.904
GLocal-K [11]	0.889
GHRS [6]	0.887
JK-DMC [35]	0.906
T-ULVD [14]	0.892
SeVGAER	**0.885**

4.3. Ablation Study

In order to highlight the significance of our proposed modifications and techniques, we conduct an ablation study. The ablation study involves creating variations of our SeVGAER model where certain features or techniques are removed, and comparing their performance to that of the complete SeVGAER model. Three specific features are examined: the contrastive learning strategy, the variational mechanism (KL divergence), and the custom aggregators.

The first variation involves removing the contrastive learning strategy from the training process. In this case, the model is trained using the traditional approach, without the random data splitting and consistency check between the two halves of the dataset. This variation is meant to evaluate the contribution of the contrastive learning strategy to the overall performance of the model.

The second variation of the SeVGAER model does not employ the variational mechanism, specifically the KL divergence loss. The model is instead trained solely with the graph reconstruction loss. This variation is designed to assess the contribution of the variational mechanism to the model's performance.

The third variation of the SeVGAER model does not use the custom aggregators in the GraphSAGE convolutional layers. Instead, it employs the standard aggregators provided in the GraphSAGE framework. This variation aims to evaluate the performance contribution of the custom aggregators.

The results of the ablation study are presented in Table 3. The performance of each variation is compared with that of the complete SeVGAER model to determine the contribution of each component to the overall model performance.

Table 3
Ablation study

MODEL	RMSE
SeVGAER-without CL	0.892
SeVGAER-without KL	0.889
SeVGAER-without aggr.	0.889
SeVGAER-complete	**0.885**

The results of the ablation study clearly underscore the importance of each component in enhancing the performance of the SeVGAER model. Detailed analysis of the results is provided in the following section.

4.4. Results Analysis

The results in Table 2 demonstrate the superior performance of the proposed SeVGAER model in comparison to the baseline methods. Our model achieves the lowest RMSE score of 0.885, outperforming other state-of-the-art models. This superior performance can be attributed to the innovative modifications and techniques incorporated in our model.

Firstly, the SeVGAER model leverages a Semantic-Enhanced Variational Graph Autoencoder, which effectively captures complex user-movie interactions. This enables the model to provide more personalized movie recommendations, thereby improving the overall performance.

Secondly, by integrating additional movie information such as plot summaries and graph structural features, our model gains a richer context for making recommendations. This multi-source feature integration significantly enhances the performance of the recommender system.

Finally, the novel training strategy involving random data splitting and consistency checking between model outputs, inspired by contrastive learning, serves to further improve the model's performance.

The results of the ablation study presented in Table 3 further substantiate the effectiveness of these components. Each variation of the model, where a specific feature or technique is removed, performs worse than the complete SeVGAER model. This underscores the significance of each component in enhancing the performance of the model.

In conclusion, the superior performance of the SeVGAER model can be attributed to its innovative architecture, the integration of additional movie information, and the novel training strategy. These results set a new precedent for future research in the field of movie recommender systems.

5. Conclusion and Future Work

In this work, we presented the Semantic-Enhanced Variational Graph Autoencoder for Movie Recommendation (SeVGAER) model, an innovative approach that leverages graph-based learning, plot summary semantic vectors, and a unique contrastive learning strategy to provide accurate and personalized movie recommendations. The SeVGAER model demonstrated superior performance over state-of-the-art baseline models, and the results of our ablation study further underscored the importance of each component in enhancing the performance of the model.

The key contributions of our work include the integration of additional movie information into the graph structure, the modification of the GraphSAGE aggregator to better fit our task, and the introduction of a contrastive training strategy that improves model generalization. The superior performance of our model on the Movielens100k dataset highlights its potential in effectively addressing the challenges of movie recommendation.

Looking forward, there are several potential directions for future work. First, while we utilized plot summaries as additional movie information in this study, other sources of information such as movie genres, director, actors, and user demographics could be integrated to further enhance the system's performance. Of particular note, our model can also attempt to incorporate users' dynamic temporal information like in [5]. Second, the proposed model could be extended to other types of recommendation systems, such as music, books, or products, demonstrating its versatility. Lastly, the contrastive learning strategy could be further explored and optimized to improve its effectiveness in enhancing model performance.

In conclusion, our research provides a novel perspective and sets a new precedent in the field of movie recommender systems. It demonstrates the potential of integrating additional information and novel training strategies within a graph-based model, opening up new avenues for future research in this area.

References

1. Alhijawi, B., Kilani, Y. The Recommender System: A Survey. International Journal of Advanced Intelligence Paradigms, 2020, 15(3), 229-251. https://doi.org/10.1504/IJAIP.2020.105815
2. Barbieri, J., Alvim, L. G., Braida, F., Zimbrão, G. A. O. Autoencoders and Recommender Systems: COFILS Approach. Expert Systems with Applications, 2017, 89, 81-90. https://doi.org/10.1016/j.eswa.2017.07.030
3. Battaglia, P., Pascanu, R., Lai, M., Jimenez Rezende, D., Kavukcuoglu, K. Interaction Networks for Learning about Objects, Relations and Physics. Advances in Neural Information Processing Systems, 2016, 29. https://dl.acm.org/doi/10.5555/3157382.3157601
4. Burke, R. Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-adapted Interaction, 2002, 12, 331-370. https://doi.org/10.1023/A:1021240730564

5. Chen, C., Huang, J. Temporal-Guided Knowledge Graph-Enhanced Graph Convolutional Network for Personalized Movie Recommendation Systems. Future Internet, 2023, 15(10), 323. https://doi.org/10.3390/fi15100323

6. Darban, Z. Z., Valipour, M. H. GHRS: Graph-based Hybrid Recommendation System with Application to Movie Recommendation. Expert Systems with Applications, 2022, 200, 116850. https://doi.org/10.1016/j.eswa.2022.116850

7. Fan, W., Ma, Y., Yin, D., Wang, J., Tang, J., Li, Q. Deep Social Collaborative Filtering. Proceedings of the 13th ACM Conference on Recommender Systems, 2019, 305-313. https://doi.org/10.1145/3298689.3347011

8. Fout, A., Byrd, J., Shariat, B., Ben-Hur, A. Protein Interface Prediction Using Graph Convolutional Networks. Advances in Neural Information Processing Systems, 2017, 30. https://dl.acm.org/doi/10.5555/3295222.3295399

9. Hamaguchi, T., Oiwa, H., Shimbo, M., Matsumoto, Y. Knowledge Transfer for Out-of-knowledge-base Entities: A Graph Neural Network Approach. arXiv preprint arXiv:1706.05674, 2017. https://doi.org/10.24963/ijcai.2017/250

10. Hamilton, W., Ying, Z., Leskovec, J. Inductive Representation Learning on Large Graphs. Advances in Neural Information Processing Systems, 2017, 30. https://dl.acm.org/doi/10.5555/3294771.3294869

11. Han, S. C., Lim, T., Long, S., Burgstaller, B., Poon, J. GLocal-K: Global and Local Kernels for Recommender Systems. Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, 3063-3067. https://doi.org/10.1145/3459637.3482112

12. Harper, F. M., Konstan, J. A. The Movielens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TIIS), 2015, 5(4), 1-19.https://doi.org/10.1145/2827872

13. He, X., Chua, T. Neural Factorization Machines for Sparse Predictive Analytics. Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, 355-364. https://doi.org/10.1145/3077136.3080777

14. Horasan, F., Yurttakal, A. H., Gündüz, S. A Novel Model Based Collaborative Filtering Recommender System via Truncated ULV Decomposition. Journal of King Saud University-Computer and Information Sciences, 2023, 35(8), 101724. https://doi.org/10.1016/j.jksuci.2023.101724

15. Hsu, P., Chen, C., Chou, C., Huang, S. Explainable Mutual Fund Recommendation System Developed Based on Knowledge Graph Embeddings. Applied Intelligence, 2022, 1-26. https://doi.org/10.1007/s10489-021-03136-1

16. Kipf, T. N., Welling, M. Semi-supervised Classification with Graph Convolutional Networks. arXiv preprint arXiv:1609.02907, 2016. https://doi.org/10.48550/arXiv.1609.02907

17. Kipf, T. N., Welling, M. Variational Graph Auto-encoders. arXiv preprint arXiv:1611.07308, 2016. https://doi.org/10.48550/arXiv.1611.07308

18. Koren, Y. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, 426-434. https://doi.org/10.1145/1401890.1401944

19. Lee, D., Seung, H. S. Algorithms for Non-negative Matrix Factorization. Advances in Neural Information Processing Systems, 2000, 13. https://dl.acm.org/doi/10.5555/3008751.3008829

20. Lemire, D., Maclachlan, A. Slope One Predictors for Online Rating-based Collaborative Filtering. Proceedings of the 2005 SIAM International Conference on Data Mining, 2005, 471-475. https://doi.org/10.1137/1.9781611972757.43

21. Paterek, A. Improving Regularized Singular Value Decomposition for Collaborative Filtering. Proceedings of KDD Cup and Workshop, 2007, 5-8. https://doi.org/10.1137/1.9781611972757.43

22. Reimers, N., Gurevych, I. Making Monolingual Sentence Embeddings Multilingual Using Knowledge Distillation. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020, 0011-01-11. https://doi.org/10.18653/v1/2020.emnlp-main.365

23. Reimers, N., Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019, 0011-01-11. https://doi.org/10.18653/v1/D19-1410

24. Rendle, S. Factorization Machines. 2010 IEEE International Conference on Data Mining, 2010, 995-1000. https://doi.org/10.1109/ICDM.2010.127

25. Salah, A., Rogovschi, N., Nadif, M. A Dynamic Collaborative Filtering System via a Weighted Clustering Approach. Neurocomputing, 2016, 175, 206-215. https://doi.org/10.1016/j.neucom.2015.10.050

26. Sedhain, S., Menon, A.K., Sanner, S., Xie, L. Autorec: Autoencoders Meet Collaborative Filter-

ing. Proceedings of the 24th International Conference on World Wide Web, 2015, 111-112. https://doi.org/10.1145/2740908.2742726

27. Sun, Z., Guo, Q., Yang, J., Fang, H., Guo, G., Zhang, J., Burke, R. Research Commentary on Recommendations with Side Information: A Survey and Research Directions. Electronic Commerce Research and Applications, 2019, 37, 100879. https://doi.org/10.1016/j.elerap.2019.100879
28. Sun, Z., Vashishth, S., Sanyal, S., Talukdar, P., Yang, Y. A Re-evaluation of Knowledge Graph Completion Methods. arXiv preprint arXiv:1911.03903, 2019. https://doi.org/10.18653/v1/2020.acl-main.489
29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.U., Polosukhin, I. Attention is All You Need. Advances in Neural Information Processing Systems, 2017, 30. https://dl.acm.org/doi/10.5555/3295222.3295349
30. Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., Wang, Z. Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, 968-977. https://doi.org/10.1145/3292500.3330836
31. Wu, X., Shi, B., Dong, Y., Huang, C., Chawla, N.V. Neural Tensor Factorization for Temporal Interaction Learning. Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, 2019, 537-545. https://doi.org/10.1145/3289600.3290998
32. Zhang, S., Yao, L., Sun, A., Tay, Y. Deep Learning Based Recommender System. ACM Computing Surveys, 2020, 52(1), 1-38. https://doi.org/10.1145/3285029
33. Zhang, S., Yao, L., Xu, X. Autosvd++ an Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders. Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, 957-960. https://doi.org/10.1145/3077136.3080689
34. Zhong, Y., Huang, C., Li, Q. A Collaborative Filtering Recommendation Algorithm Based on Fuzzy C-means Clustering. Journal of Intelligent & Fuzzy Systems, 2022, 43(1), 309-323. https://doi.org/10.3233/JIFS-212216
35. Zhu, X., Fu, J., Chen, C. Matrix Completion of Adaptive Jumping Graph Neural Networks for Recommendation Systems. IEEE Access, 2023. https://doi.org/10.1109/ACCESS.2023.3305945