Movie Recommender System based on Generative AI

Carmen Pei Ling Tung
Faculty of Computing and Informatics
Multimedia University
63100, Cyberjaya, Malaysia
1201102432@student.mmu.edu.my

Su-Cheng Haw*
Faculty of Computing and Informatics
Multimedia University
63100, Cyberjaya, Malaysia
sucheng@mmu.edu.my

Wan-Er Kong
Faculty of Computing and Informatics
Multimedia University
63100, Cyberjaya, Malaysia
kong.wan.er@student.mmu.edu.my

Palanichamy Naveen
Faculty of Computing and Informatics
Multimedia University
63100, Cyberjaya, Malaysia
p.naveen@mmu.edu.my

***Abstract*— Movie recommender systems have revolutionized how people discover and experience movies using sophisticated algorithms, such a system generates personalized recommendations to assist users in reaching the movies that precisely fit their tastes. The traditional movie recommender systems lacks of diversity, unable to capture complex user preferences, and suffers from the cold-start issues. In this paper, we utilize the generative AI, specifically with Variational Autoencoders (VAE), to solve these problems. By combining VAEs, the system aims to produce more rich and individualized recommendations. VAE mitigates these issues by enabling the generation of meaningful item representations, promoting diversity in recommendations, and capturing the complexity of user preferences in a continuous latent space. The implementation is structured as follows: first, review the generative AI in movie recommender systems; second, implement the chosen generative AI approach; and finally, conduct the performance metrics evaluation of the selected generative AI approach. Ultimately, this research will enable the system to offer an achievement of an interactive, changing, and more personalized movie-discovering experience through the amazing power of generative AI.**

Keywords*— *generative AI, VAE, movie recommender, recommender system, recommendation.

I. Introduction

Recommender systems are so important for improving the online experience of customers, but conventional approaches frequently face with synonymy problems, misunderstanding of complex user preferences, and dealing with new items or user cases. Traditionally, recommendation algorithms depended on collaborative filtering, content-based methods, or hybrid techniques to provide people with suggestions for items or contents. Nevertheless, the methods are constantly confronted with the fact that they are unable to formulate unified recommendations containing a variety of characters, understand sophisticated user preferences, and address the cold-start problem when especially new items or users come in.

With the advent of AI techniques and generative AI, the excitement about and the possibility of revolutionizing the recommendation systems was greatly energized. Human-like machine-learning techniques used in Variational Autoencoders (VAEs) [1], [2], [3], [4], generative adversarial networks (GANs) [5], [6], [7], [8]and transformers are a fresh way to enhance recommendation systems. Such techniques are proficient at capturing data distribution patterns and generating new content. They do it possibly the best by recommending variety originality and cold-start problems. Acquiring generative AI can significantly advance recommendation systems by providing the desired personal preferences, diversity, and fitting user choices.

Generative AI and recommendation systems integration create new opportunities in customer experience shaping as the combination of two powerful technologies [9], [10], [11], [12]. The combination of deep learning development with the scope of plenty of data availability has thus far led to high-class models that can understand complicated patterns and produce top-class content.

II. Background and Related Work

A. *Stages of the Recommendation System*

Generative AI Movie Recommender System typically passes through multiple stages in its basic functioning that help to provide tailored movie recommendations. To begin with, the system captures data about the users from which it draws data from historical ratings, viewing wants, and preferences relating to classical ratings and other parameters in the film industry like genres and actors is collected. One of the tools might be a mention of the time and place where my memories take place, if it is relevant. Finally, complex machine learning models are used to dig information of general patterns and relations between users and movies on this data. The core of the system is in fact the content generation which is based on the learned representations that produce movie recommendations that are inclusive of the popular choices, diverse recommendations, and at the same time surprising content with the goal of facilitating a favorable user experience. Lastly, the system does continuous improvement based on the user feedback besides this it takes into consideration rating and preferences to improve with time which altogether leads to better recommendation quality.

B. *Recommendation System Techniques*

Traditional recommender systems such as Content-Based (CB), Collaborative Filtering (CF), Knowledge-Based (KB), and Hybrid recommender systems are the foundation of the broader recommendation systems domain and they robustly employ a number of techniques to offer individualized suggestions to the users. CB systems recommend items to users based on the attributes or content of items and user profiles. For example, in a movie recommendation system, a content-based approach may recommend movies that share similar genres, directors, or actors with the ones the user has liked. CF is a renowned and widely-used technique for information filtering that identifies users within a group who share similarities with a particular user [13]. For instance, if Cary and Johnny both enjoyed the same movies, a movie enjoyed by Johnny but not seen by Cary could be recommended to Cary.

On the other hand, knowledge-based systems use explicit knowledge about users and items to make recommendations. This approach is commonly used in domains where there is structured information available. For example, a knowledge-based approach in a travel recommendation system might consider user preferences, budget constraints, and the availability of hotels, flights, and activities to suggest a personalized travel itinerary. This method employs various hybrid models like cascade, weighted, mixed, and switching hybrids, each distinguished by their operational techniques [14].

The generative AI technology has recently gain its popularity especially in chatbot, healthcare, manufacturing and education domains [9], [10], [11], [12].

1) Boltzmann Machine (BM): The BM is a type of stochastic recurrent neural network known for its energy-based structure, featuring two neuron sets that interact and exhibit binary states. While the visible layer acts as the data input, the hidden layer serves as the feature detector. Despite its efficiency in reducing training times, the interconnectivity of the visible layer's neurons still presents challenges in processing large datasets, limiting its practical application [15].

RBMs are the most well-known and widely used variant of Boltzmann Machines (BM). They consist of a visible layer and a hidden layer, with no connections within the layers. RBMs are used for unsupervised learning tasks like dimensionality reduction, feature learning, collaborative filtering, and more. The main problem with BM is that deploying them in the field is impossible. Due to the bidirectional nature of links between each node to every other node, it is computationally very expensive to sample each walk. This is simply because, mathematically, as the number of nodes increases by 1, the number of links grows exponentially. To overcome this problem and still use the power of BMs, the Restricted Boltzmann Machine (RBM) was proposed [16].

Deep Belief Networks (DBN) are advanced deep learning algorithms that address the challenges of deep neural networks, like slow learning speeds and overfitting, through a hierarchical feature learning strategy [17]. Deep belief networks (DBN) are powerful models that learn features through layer-by-layer learning strategies, improving model training efficiency and generalization ability. These networks combine multiple restricted BM (RBM) and employ greedy methods for learning and approximate inference. High dimensionality input data features enhance DBN's generalization ability[18].

Deep BM (DBM) are an extension of RBMs, consisting of multiple layers of hidden units. A DBM connects the first visible layer to multiple hidden layers in deep architecture network. DBMs learn more intricate representations than RBMs but training of DBMs is more complicated. DBMs are known for their capability of getting self representations that grow more complex with time and studying this method is believed to be a great way of solving speech or object recognition problems [19] Another vital distinction between DBNs and DBMs is that all associations are bidirectional and undirected in the latter. DBMs are additionally utilized in extracting cryptic subtle features hidden in the data hence suitable for speech, recognition, and object detection tasks. Conversely, DBMs instead of DBNs employ the approximate inference procedure initially to accelerate learning, which is combined with the top-down feedback. Then, information about the input ambiguities is incorporated, which enables DBM to deal effectively with uncertainty [16] contrasting to deep belief networks, which demonstrate the ability to develop deep mechanisms for internal representation learning that increases in complexity, which plays a significant role in addressing the object and speech recognition tasks. Another one is that top-down depictions are constructed from a huge pool of unnamed sensory inputs. Yet, hardly labeled data that can be used as a training set is the only information to be slightly used for that purpose. As a final note, approximate inference and the initial sparsely connected bottom-up pass may be supplemented with feedback descent from the top of the network. This would allow the DBM to better handle nuanced inputs [19].

BM and BM's cases, like RBM, have become popular tools for recommender systems. RBMs, a famous RBM derivative, were initially attempted to reduce the complexity issue that was common in original BM. RBMs got its way into recommendation systems mainly because their structure is based on visible and hidden layers and no intralayer connections. RBMs operate effectively in unsupervised learning settings, enabling collaborative filtering and feature learning tasks. RBM was the pioneering deep learning method utilized in recommender systems, employing its capacity to predict missing data by automatically extracting abstract features. Its utility in capturing complex patterns in user-item interactions has marked it as a valuable tool in developing recommendation models. The RBM's ability to extract high-level representations from input data has significantly contributed to enhancing the efficiency and accuracy of recommender systems.

2) Autoencoder: Autoencoders, a type of algorithm used for data compression, have been crucial in tackling the complexity of data by reducing redundancy and minimizing wasted space during data transmission. They work by automatically learning to compress and decompress data based on examples provided. Typically implemented using neural networks, autoencoders are tailored to the specific characteristics of the data they are trained on, meaning they may not perform well when applied to different types of data. For instance, an autoencoder trained on human faces may not effectively compress images of trees due to its focus on facial features [20]. Autoencoders are useful for learning to encode observable data into a latent space of smaller dimensionality and thus perform dimensionality reduction (manifold learning). However, the latent variable space often lacks structure, and it is impossible, by construction, to sample from the data distribution.

The VAE framework addresses these two issues by regularizing the latent space with an adapted training procedure. This allows to train complex generative models with latent variables while providing a way to sample from the learned data distribution, making it useful for unsupervised density modeling. Moreover, VAE leans more towards data generation compared to traditional autoencoders. Once the encoder is trained, new data can be generated by feeding the standard normal distribution into the decoder, resulting in novel samples that differ from the training data, akin to Generative Adversarial Networks (GANs). The output vector generated by the encoder in an autoencoder scenario can be illustrated as follows: In an autoencoder, if an image of a full moon is inputted, the model outputs a corresponding image of the full moon. Similarly, if an image of a crescent moon is fed into the model, it produces an image of a crescent moon. However, when selecting a point from the continuum between the codes for a full moon and a crescent moon, in the context of autoencoders, it's uncertain what type of image the model will generate. This uncertainty arises from the lack of knowledge regarding the transition process within the model from the code representing a full moon to that representing a crescent moon [20].

VAEs are the recommender system (RS) model, incorporating user-level Differential Privacy (DP) to ensure privacy guarantees. VAEs have been introduced into recommender system design to address shortcomings of Collaborative Filtering (CF) methods, specifically related to sparsity in rating information matrices and the cold start issue (difficulty in recommending items to new users without previous rating information). VAEs, being non-linear models powered by neural networks, can capture more complex patterns in the data.[21]

VAEs serve as generative neural models, which take in complex, high-dimensional input data and produce a condensed representation known as the "latent space." This latent space represents the input data probabilistically, detailing the probability distribution of each input value. With an encoding, or latent space representation, at hand, any distribution within this space can be randomly sampled to create new output that closely resembles the original input data [22].

3) Generative Adversarial Networks (GANs): GANs have contributed to tremendous progress in computer vision ranging from image creation and modification to texture translation which is a machine-learning technique for generating imaginary or fake images or texts. Moreover, it is also applicable in natural language processing and music[22]. These models are popular because they learn by producing images or speech directly without human intervention. Due to these advantages, researchers have focused on the Min-Max Game in GANs in various fields[23]. These models are common because they can learn on their own when presented with new information. For this reason, the Min-Max Game (Equation 1) has been a focal point of GAN research in recent years [24].

$$\min_G \max_D V(D,G) = \mathrm{E}_{X\sim P_{\mathrm{data}}(X)}[\log \mathrm{D}(\mathrm{X})] + \mathrm{E}_{Z\sim P_Z(\mathrm{Z})}[\log(1-\mathrm{D}(\mathrm{G}(\mathrm{z})))] \tag{1}$$

The discriminator's output is shown above in the form of a simple formula. they are usually estimated from the data on true logarithm while the constructor is being used. finally, [log(1-D(G(z)))] is either the correct output or the true representation. For example, the the discriminator progress into real data yields a result above 1, thus indicating that is is comparable to a real handwriting image. the output from tester on the data given by the constructor. should approach 0. Consequently, the target function of the algorithm will be set with. maximized. Discriminator function will be trained to either return 1 or 0 so its output will be binary. data being constructed by the constructor means it will be close to 1.In contrast, the trainee designs a formal method that returns 0 given the actual. optimization algorithms that use data and focus on minimizing an objective function [24].

4) Transformer: The Transform architecture lays the basis for latest state-of-the-art models including GPT-3, Codex, Gopher, and DALL-E-2. It has assisted in overcoming the disadvantages of the previous models including the recurrent neural networks especially in a scenario where we are required to process sequences of different lengths and to interpret the context. The attention mechanism at the center of the Transformer is central to the operation of the model. This mechanism allows the model to pay attention to various segments of the input sequence so that the model can better understand the context of the input. It has an encoder that processes input and a decoder that gives output; meanwhile every layer of this model contains a multi-head attention and a feed-forward network. The multi-head attention, a key component, computes the differential importance for every single token, thus boosting the model's capacity to tackle complex long-range dependencies. Transformers are the best in parallel processing and are good to go with large-scale pre-training. On the account mentioned above, they find application across different NLP fields of study.

C. *Related Works*

The researchers who had conducted both studies examined a range of techniques that would be used to improve the performance and effectiveness of recommender systems in tailoring the suggestions.

Ouhbi et al. [8] combined Deep Belief Networks (DBN) and collaborative filtering to address the cold start problem. They extracted item features from Open Movie Database (OMDB) Application Programming Interface (API), produced semantic representations, clustered items via k-means algorithm, and included item-based collaborative filtering in the product recommender system. Similarly, Gupta et al. [2] extended collaborative filtering by integrating VAEs with movie embeddings obtained from a sibling VAE network. This augmentation aimed to enhance the characterization of user-item interactions, ultimately improving recommendation accuracy.

Meanwhile, Yu et al. [9] introduced VAEGAN, a collaborative filtering framework founded on Adversarial VAE, which utilized adversarial training to generate flexible inference models for more accurate posterior distributions and maximum-likelihood estimations. Their model also incorporated an auxiliary discriminative network and a contractive loss term to refine feature robustness and enhance generalization performance. Krishnathasan [10] utilized a situation with VAEs where both models are trained simultaneously and take into account users taste for genres. This method involved preparing two inputs for each user: the dataset was represented by a matrix in which users provided the ratings for each movie and was another matrix in which genre preferences of users were gathered and were then averaged per user to capture genre preferences efficiently. By this method, they sought to improve their movie rating prediction model by feeding it users' genre pattern data. The model would then be able to give a more accurate recommendation.

Furthermore, Gao et al. [11] carried out a broad survey of GAN-based recommendation models, analyzing their capability in acquiring accurate a priori data. To do this they made a classification of these models by problem-oriented spec, which were judged by metrics (precision and NDCG) that reflect their performance. Their performance revealed that the model is good for handling data noise and scarce datasets. Tahmasebi et al. [12] demonstrated a coalition of co-operative and content-based filters that SRDNet had incorporated in their new system to tackle the cold start problem. They achieved this by applying a carefully designed sequence of two filtering approaches using deep learning: first, content-based filtering of previously watched movies, second, predicting ratings for unexplored movies, much deeper autoencoder network combined with collaborative filtering. They demonstrated how successful this method was.

Additionally, Srikanth et al. [13] developed an intelligent recommendation system using Deep AutoEncoders for Netflix movies. The system enhances recommendations by predicting top N movies with 3 months of implicit data, achieving high predictive accuracy in Root Mean Squared Error (RMSE). Harshvardhan et al. [14] introduced a novel Recommender System, UBMTR, employing BM to extract latent user preferences from historical data and incorporating temporal information for personalized content recommendations. Their model demonstrated superior performance compared to existing recommendation techniques, showcasing innovation in leveraging BM for time-aware recommendation systems.

Furthermore, Agrawal et al. [15] developed a multimodal recommender system using audio, meta-data, subtitles, and video embeddings from movie trailers. Their methodology generated embeddings, used knowledge graphs, clustered frames, and used a trident model for rating prediction, achieving competitive RMSE values. Finally, Zhang et al. [16] developed Agent4Rec, a movie recommendation simulator using Large Language Model (LLM) empowered generative agents to simulate user-personalized preferences. However, limitations like data source constraints and occasional hallucinations require further exploration for broader applicability.

Overall, each of these studies contributes unique insights and methodologies to the field of recommender systems, aiming to address various challenges and enhance recommendation accuracy and effectiveness.

III. Proposed Framework

VAEs represent a significant advancement in generative modelling, particularly noted for their application in movie recommender systems. As a specialized form of autoencoder, VAEs are engineered not merely to encode and decode input data but to intricately learn and interpret the distribution of data within a latent space. This dual capability is crucial for developing a nuanced understanding of movie characteristics and user preferences, which, in turn, enhances recommendation quality.

These are some definitions and descriptions associated with VAE.

Encoder: In the encoder part of the VAE, the input features are the items; they are fed into a compressor and a new representation of them, in a latent (hidden) space, is created. This is accomplished by the middle layers and latent space layer that determines the mean and log variance vector.

Reparameterization Trick: VAEs have a special feature that enables population of random sampling while optimizing the parameters in gradient-based optimization. It contributes to neural networks' ability to investigate and learn complex features. This step involves producing samples from the latent space distribution using vector mean and log variance, allowing for uninterrupted transformations through hyperspace.

Latent Space: The latent space (z) is a lower-dimensional space where each point represents a set of latent features. The reparameterization trick ensures that samples in this space have a meaningful and continuous interpretation.

Decoder: The decoder component of the VAE is tasked with reconstructing the initial input from the condensed latent representation. Here, the model is trained to produce fresh data points (recommendations) resembling the original inputs, yet possibly showcasing novel feature combinations. The efficacy of the decoder is evaluated based on its ability to faithfully reproduce the original input data, thereby encapsulating the fundamental attributes of items.

Loss Function: The loss function consists of two components, elaborated as follows.

- Reconstruction Loss (xent_loss): This component (binary cross-entropy in this case) measures how well the model can reconstruct the original input data from the latent representation. A lower reconstruction loss indicates a more accurate model.
- KL Divergence Loss (kl_loss): Kullback–Leibler divergence is added to enforce a regular distribution (usually normal) in the latent space, preventing overfitting and ensuring a smooth latent space representation.

Recommendation Generation: Leveraging the decoder, VAE-based systems can recommend movies by exploring the vicinity of a user's preferred movies in the latent space, enabling the discovery of films with similar but potentially novel features.

Fig. 1 depicts the overall proposed framework. The first step involves choosing the datasets and ML techniques.

A) Choosing the datasets and generative AI techniques: The prototype model was trained on the "100K Dataset" sourced from MovieLens, comprising the following:

- 100,000 ratings ranging from 1 to 5, provided by 943 users for 1682 movies.
- Each user has rated a minimum of 20 movies.
- Basic demographic info for the users (age, gender, occupation, zip)

Prior to training, the dataset underwent cleaning procedures, wherein users with fewer than 20 ratings or incomplete demographic information were excluded from the dataset.

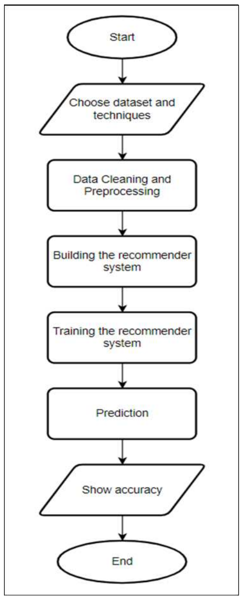

Start
Choose dataset and techniques
Data Cleaning and Preprocessing
Building the recommender system
Training the recommender system
Prediction
Show accuracy
End

Fig. 1. Prototype flowchart.

B) Data Cleaning: Next, a simple data cleaning process is carried out. The first step to be performed when simulating the prototype is to clean a dataset before using it for training models. For data cleansing, a simple pipeline is used, as described as follows.

- Filtering Users: The code filters out users rated fewer than 20 movies. This focuses on users who have provided enough data to contribute to a reliable recommendation system. The assumption is that more ratings per user will lead to better training and prediction accuracy.
- Handling Missing Data: The items dataset may contain missing values. To address this, the code fills missing values with zeros (fillna(0)).

C) Building the recommender system: Following the completion of cleaning and preprocessing the movie dataset, it is now primed for utilization in our prototype. This prototype will incorporate a collaborative filtering approach for the movie recommendation system. The predictive model will allow users to input movies they have enjoyed or are interested in. From here, the system will compare the user's input with the preferences of other users in the dataset to recommend similar movies. Users can choose to either register as a user or as a guest. Registered users will have access to their recommendation history, while guests will not.

Besides VAE, the prototype model will utilize collaborative filtering techniques. Collaborative filtering involves analyzing user-item interaction data to identify patterns and make recommendations based on similar users' preferences. Specifically, user-item collaborative filtering will be used, where user similarities are calculated based on their shared preferences for movies. This method enables the system to suggest movies enjoyed by users with similar tastes that the current user has not yet viewed.

D) Training the Recommender System:

Upon the completion of data preprocessing, the recommender system enters the training phase, where the model learns to identify patterns and preferences from the historical interaction data. The following processes are carried out:

- Data Splitting: The preprocessed dataset is split into training and validation sets, with the typical ratio being 80% for training and 20% for validation. This allows for the assessment of the model's performance on unseen data.
- Model Selection: VAE model and algorithm such as Singular Value Decomposition (SVD) or Alternating Least Squares (ALS) is selected based on its suitability to handle sparse matrices and the ability to uncover latent factors within the data.
- Hyperparameter Tuning: Various hyperparameters such as the number of latent factors, regularization terms, and learning rate are tuned to optimize the model's performance. This step might involve the use of grid search or random search strategies.
- Model Fitting: The training data is fed into the model, which adjusts its weights through iterations of the selected algorithm, to minimize the difference between the predicted and actual ratings.
- Model Validation: After fitting the model, the validation set is used to evaluate the model's performance and prevent overfitting. The early stopping technique can be implemented to halt training when the validation error starts to increase, indicating potential overfitting.

E) Prediction:

Following the training, the system is capable of generating predictions. This phase includes:

- Loading the Model: The trained model is loaded and prepared to make predictions.
- User Input: The system captures input from users, which may include movies they have enjoyed or expressed interest in.
- Generating Predictions: Utilizing the user input and the learned model, predictions are generated. This might involve computing a user's similarity with other users and aggregating ratings of the movies enjoyed by similar users.
- Ranking Recommendations: The predicted ratings are sorted, and a list of top-N recommendations is generated. This list is personalized for each user based on the predicted affinity for unwatched movies.

F) Show Accuracy:

The final phase is to evaluate and demonstrate the accuracy of the recommender system's predictions:

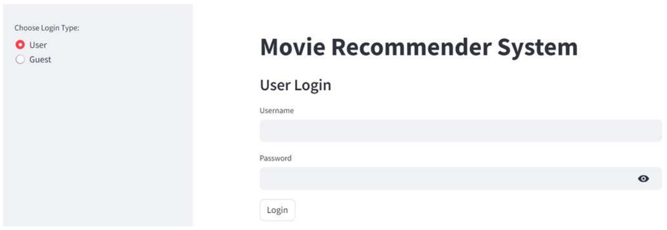
- Evaluation Metrics Calculation: Metrics such as RMSE, MAE, precision, recall, and F1 score are calculated using the test set to measure the accuracy of the model's predictions.

- Interpretation of Results: The calculated metrics are analyzed to interpret the model's performance. Lower values of RMSE and MAE indicate higher accuracy, while higher precision, recall, and F1 scores reflect better relevance and coverage of the recommendations.
- Feedback Loop: The results can be used to further refine the model by going back to the training phase with adjusted hyperparameters or by trying different algorithms to improve the accuracy.

Finally, the recommender system will be integrated into a user-friendly interface, such as deploying it into Streamlit. This will allow users to easily input their movie preferences and receive real-time personalised recommendations.

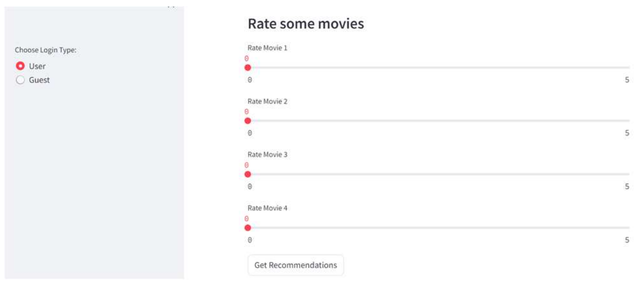
IV. Results and Discussion

The initial graphical user interface (GUI) will be created utilizing VSCode and Streamlit. To launch the prototype, users can execute the command "streamlit run "c :<File Path>\GUI\movie_recommender.py in the command prompt " .This command will open the Streamlit webpage on the browser, as illustrated in Fig. 2. After login, the user will be able to provide rating to movie as depicted in Fig. 3.



Choose Login Type: [✓ User] Guest

Fig. 2. Movie recommender system main page.

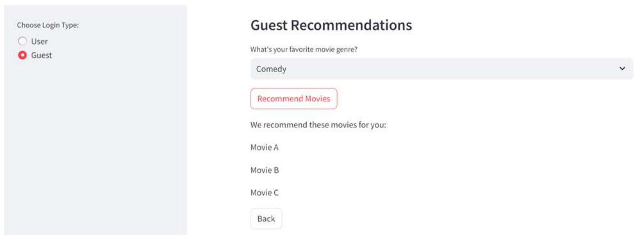


Choose Login Type: [✓ User] Guest

Fig. 3. Rate watched movies.

Fig. 4 shows the login page for guests. Guests must select their movie genre preferences to receive movie recommendations since they do not have a watched movie history that can be used to personalize recommendations. By selecting their preferred genres, guests can provide input that helps the system generate relevant movie suggestions tailored to their interests.

Based on collaborative filtering techniques, the preliminary evaluation of the movie recommender system prototype demonstrates promising performance in generating personalized movie recommendations for users. After cleaning and preprocessing the movie dataset, the prototype was successfully implemented, allowing users to input movies they enjoy or are interested in. Using collaborative filtering, the system compares user preferences with those of other users in the dataset to recommend similar movies. This approach enables the system to provide personalized recommendations tailored to individual users' tastes.



Choose Login Type: User [✓ Guest]

What's your favorite movie genre? Comedy

We recommend these movies for you:

Movie A

Movie B

Movie C

Fig. 4. Enter movie genre preferences.

For the implementation of collaborative filtering, algorithms such as Singular Value Decomposition (SVD) or Alternating Least Squares (ALS) were considered. These algorithms analyze user-item interaction data to predict missing ratings and generate personalized recommendations. While the specific algorithm used is subject to further experimentation and optimization, initial testing suggests promising results in terms of recommendation accuracy.

Regarding evaluation criteria, the model's effectiveness is evaluated through mean squared error (MSE) and MAE. MSE calculates the arithmetical average of the squared variances of the realizations and predicted ratings so that larger deviations are weighted higher. On the contrary, MAE calculates the arithmetic mean of the absolute errors is a more apparent metric, The preliminary result has shown the model can be competitive in the content features reconstruction, and metrics like MSE and MAE can show the model's effectiveness. Fig. 5 depicts the algorithm on VAE that we have implemented.

Algorithm: Variational Autoencoder

```
1   Function create_vae_architecture(n_items, latent_dim)
2   Define the input layer for the encoder with shape (n_items,)
3   Add dense layers to the encoder with 1024, 512, and 256 units using 'relu'
    activation
4   Create latent variables z_mean and z_log_var with 'latent_dim' units
5   Define a sampling function that applies the reparameterization trick
6   Define the input layer for the decoder with shape (latent_dim,)
7   Add dense layers to the decoder with 256, 512, and 1024 units using 'relu'
    activation
8   Define the output layer of the decoder with 'n_items' units and 'sigmoid'
    activation
9   Instantiate the encoder and decoder models
10  Define the VAE model by connecting the encoder and decoder
11  Define the loss function as a combination of MSE and KL divergence
12  Compile the VAE with the Adam optimizer
13  Return the compiled VAE model, encoder, and decoder
14  End Function

15  Function train_vae(vae, train_data, test_data, epochs, batch_size)
16  Initialize EarlyStopping callback with 'val_loss' monitoring and a patience
    of 8
17  Fit the VAE model on the train_data with validation on test_data
18  Apply EarlyStopping callback during training
19  Return the training history
20  End Function

21  Function predict_ratings(vae, test_data)
22  Use the VAE model to predict ratings on the test_data
23  Return the predicted ratings
24  End Function
```

```
Function compute_precision_recall_f1(test_data, predicted_ratings, k)
    Initialize arrays for precision, recall, and F1 scores
    For each user in the test data:
        Calculate top-k predicted items
        Compute precision, recall, and F1 score for the user
    End For
    Return the average precision, recall, and F1 score across all users
End Function

Function evaluate_model(test_data, predicted_ratings)
    Compute MSE and MAE between test_data and predicted_ratings
    Calculate RMSE from MSE
    Display the precision, recall, F1 score, RMSE, and MAE
End Function

Main
    Define number of items (n_items), number of users (n_users), and latent dimension (latent_dim)
    vae, encoder, decoder = create_vae_architecture(n_items, latent_dim=50)
    history = train_vae(vae, train_data, test_data, epochs=10, batch_size=64)
    predicted_ratings = predict_ratings(vae, test_data)
    precision, recall, f1 = compute_precision_recall_f1(test_data, predicted_ratings, k=5)
    evaluate_model(test_data, predicted_ratings)
End Main
```

Fig. 5. VAE for Collaborative Filtering.

V. Conclusion

In conclusion, this paper has conducted a comprehensive review of existing recommendation systems, examining its methodologies and merits and limitations. Subsequently, we have designed and implemented a collaborative filtering prototype that generates recommendations based on user-item interaction history. Looking ahead, there is substantial potential for enhancing the model by exploring various hybrid recommendation approaches. Enriching the system with demographic and contextual data, and refining it through user feedback, are promising avenues for personalization.

References

[1] A. Salah, T. B. Tran, and H. Lauw, “Towards source-aligned variational models for cross-domain recommendation,” in *RecSys 2021 - 15th ACM Conference on Recommender Systems*, 2021. doi: 10.1145/3460231.3474265.

[2] L. Ruthotto and E. Haber, “An introduction to deep generative modeling,” *GAMM Mitteilungen*, vol. 44, no. 2, 2021, doi: 10.1002/gamm.202100008.

[3] B. Weggenmann, V. Rublack, M. Andrejczuk, J. Mattern, and F. Kerschbaum, “DP-VAE: Human-Readable Text Anonymization for Online Reviews with Differentially Private Variational Autoencoders,” in *WWW 2022 - Proceedings of the ACM Web Conference 2022*, 2022. doi: 10.1145/3485447.3512232.

[4] T. Carraro, M. Polato, L. Bergamin, and F. Aiolli, “Conditioned Variational Autoencoder for Top-N Item Recommendation,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2022. doi: 10.1007/978-3-031-15931-2_64.

[5] M. Y. Xin, L. W. Ang, and S. Palaniappan, “A Data Augmented Method for Plant Disease Leaf Image Recognition based on Enhanced GAN Model Network,” *Journal of Informatics and Web Engineering*, vol. 2, no. 1, 2023, doi: 10.33093/jiwe.2023.2.1.1.

[6] A. Abbas, S. Jain, M. Gour, and S. Vankudothu, “Tomato plant disease detection using transfer learning with C-GAN synthetic images,” *Comput Electron Agric*, vol. 187, 2021, doi: 10.1016/j.compag.2021.106279.

[7] R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, “Plant disease detection using CNNs and GANs as an augmentative approach,” in *2018 IEEE International Conference on Innovative Research and Development, ICIRD 2018*, 2018. doi: 10.1109/ICIRD.2018.8376321.

[8] E. A. Ajayi, K. M. Lim, S. C. Chong, and C. P. Lee, “Three-dimensional shape generation via variational autoencoder generative adversarial network with signed distance function,” *International Journal of Electrical and Computer Engineering*, vol. 13, no. 4, 2023, doi: 10.11591/ijece.v13i4.pp4009-4019.

[9] A. Kankanhalli, “Peer Review in the Age of Generative AI,” *Journal of the Association for Information Systems*, vol. 25, no. 1. 2024. doi: 10.17705/1jais.00865.

[10] A. Bandi, P. V. S. R. Adapa, and Y. E. V. P. K. Kuchi, “The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges,” *Future Internet*, vol. 15, no. 8. 2023. doi: 10.3390/fi15080260.

[11] P. Ghimire, K. Kim, and M. Acharya, “Opportunities and Challenges of Generative AI in Construction Industry: Focusing on Adoption of Text-Based Models,” *Buildings*, vol. 14, no. 1, 2024, doi: 10.3390/buildings14010220.

[12] M. Bresson, Y. Xing, and W. Guo, “Sim2Real: Generative AI to Enhance Photorealism through Domain Transfer with GAN and Seven-Chanel-360°-Paired-Images Dataset,” *Sensors*, vol. 24, no. 1, 2024, doi: 10.3390/s24010094.

[13] Z. Yao, “Review of Movie Recommender Systems Based on Deep Learning,” *SHS Web of Conferences*, vol. 159, p. 02010, 2023, doi: 10.1051/shsconf/202315902010.

[14] K. Gupta, M. Y. Raghuprasad, and P. Kumar, “A Hybrid Variational Autoencoder for Collaborative Filtering,” Jul. 2018, [Online]. Available: http://arxiv.org/abs/1808.01006

[15] M. Liu, “Personalized Recommendation System Design for Library Resources through Deep Belief Networks,” *Mobile Information Systems*, vol. 2022, 2022, doi: 10.1155/2022/7870724.

[16] G. M. Harshvardhan, M. K. Gourisaria, S. S. Rautaray, and M. Pandey, “UBMTR: Unsupervised Boltzmann machine-based time-aware recommendation system,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6400–6413, Sep. 2022, doi: 10.1016/j.jksuci.2021.01.017.

[17] International Conference on Intelligent Computing and Internet of Things 2015 Harbin, Ha er bin shi fan da xue, Institute of Electrical and Electronics Engineers, International Conference on Intelligent Computing and Internet of Things 2015.01.17-18 Harbin, and ICIT 2015.01.17-18 Harbin, *Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things ICIT 2015 : January 17-18, 2015, Harbin, China*. IEEE, 2015.

[18] R. Gupta, S. Kumar Shukla, and V. Tripathi, “New Deep Learning Models for Medical Imaging: Deep Belief Network, GAN, Autoencoder,” in *Proceedings of the 4th International Conference on Smart Electronics and Communication, ICOSEC 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 907–913. doi: 10.1109/ICOSEC58147.2023.10276257.

[19] R. Salakhutdinov and G. Hinton, “Deep Boltzmann Machines.”

[20] Z. Ren, “The Advance of Generative Model and Variational Autoencoder,” Institute of Electrical and Electronics Engineers (IEEE), Jan. 2023, pp. 268–271. doi: 10.1109/tocs56154.2022.10016057.

[21] L. Fang, B. Du, and C. Wu, “Differentially private recommender system with variational autoencoders,” *Knowl Based Syst*, vol. 250, Aug. 2022, doi: 10.1016/j.knosys.2022.109044.

[22] A. Karapantelakis, P. Alizadeh, A. Alabassi, K. Dey, and A. Nikou, “Generative AI in mobile networks: a survey,” *Annales des Telecommunications/Annals of Telecommunications*, 2023, doi: 10.1007/s12243-023-00980-9.

[23] J. H. Yoon and B. Jang, “Evolution of Deep Learning-Based Sequential Recommender Systems: From Current Trends to New Perspectives,” *IEEE Access*, vol. 11, pp. 54265–54279, 2023, doi: 10.1109/ACCESS.2023.3281981.

[24] M. Grover, “The Development of Sequential Recommendation Systems Using Deep Learning,” in *2023 International Conference on Data Science and Network Security, ICDSNS 2023*, Institute of Electrical and Electronics Engineers Inc., 2023. doi: 10.1109/ICDSNS58469.2023.10245109.