### Movie Recommendation System: An IEEE Report

**Abstract**

This project introduces a movie recommendation system based on two different implementations: one using a deep belief network (DBN) implemented in PyTorch, and the other using a restricted Boltzmann machine (RBM) implemented in TensorFlow. The goal of the project is to predict users' movie ratings and provide personalized recommendations based on the user's rating history. This article describes the initial idea, data preprocessing, model training, evaluation, and main findings of the project in detail. The models are evaluated on the MovieLens dataset, demonstrating their effectiveness in prediction and recommendation.

**Introduction**

With the explosive growth of online content, recommender systems have become an important tool to help users discover relevant items such as movies, books, and products. Recommender systems can not only improve user satisfaction, but also increase platform stickiness and usage. This project aims to develop a movie recommendation system using two different machine learning methods: Deep Belief Network (DBN) and Restricted Boltzmann Machine (RBM). We used the MovieLens dataset, which contains user ratings of various movies, to train and evaluate these models. The main challenge of recommender systems is how to accurately capture user preferences and provide accurate recommendations based on them.

**Data Preprocessing**

The dataset used in this project is the MovieLens dataset, which includes user information, movie information, and user ratings. The preprocessing steps involved merging user and movie data, handling missing values, creating age group features, extracting the year from movie titles, and one-hot encoding categorical variables. These steps ensure that the data is in a suitable format for model training.

**Implementation 1: DBN with PyTorch**

The first implementation is to build a deep belief network (DBN) using PyTorch. DBN is a deep generative model composed of multiple stacked restricted Boltzmann machines (RBM). It captures complex patterns in the data by training layer by layer. The specific steps are as follows:

1. **Loading the Data:** Importing and preparing the MovieLens dataset.
2. **Data Preprocessing:** Handling missing values, feature extraction, and encoding.
3. **Feature Engineering:** Creating new features such as age groups and one-hot encoding.
4. **Model Development:** Building and training the DBN model.
5. **Model Optimization:** Fine-tuning hyperparameters and evaluating model performance.

The DBN model was trained on the MovieLens dataset and evaluated using metrics such as mean squared error (MSE). The results show that the DBN model effectively captures user preferences and provides accurate movie rating predictions.

**Implementation 2: RBM with TensorFlow**

The second implementation leverages Restricted Boltzmann Machines (RBM) with TensorFlow. The key steps are:

1. **Initializing Parameters:** Setting up the initial weights and biases.
2. **Gibbs Sampling:** Performing Gibbs sampling for training the RBM.
3. **Training the RBM:** Optimizing the model parameters.
4. **Predicting with the RBM:** Generating movie rating predictions.

The RBM model is trained and evaluated on the same MovieLens dataset. The results indicate that the RBM model also performs well in predicting movie ratings and making recommendations based on user history.

**Results and Findings**

The models were evaluated based on their ability to predict movie ratings and provide personalized recommendations. Key findings include:

1. Both DBN and RBM models showed consistent improvement in predicting movie ratings over epochs.
2. Users tend to rate movies in specific genres consistently higher, indicating strong genre preferences.
3. Popular genres like Comedy, Drama, and Action receive a high number of ratings, while niche genres have more dedicated viewers.
4. Different age groups exhibit distinct preferences, with teens preferring Animation and Sci-Fi, while seniors favor Classics and Westerns.
5. High-rated movies often feature well-known actors and directors, highlighting the influence of star power.

**Conclusion**

In this project, two different machine learning models for movie recommendation were successfully implemented and evaluated. Both models demonstrated effectiveness in capturing user preferences and providing accurate recommendations. Future work could include exploring additional features, improving model structure, and adopting more advanced techniques to further improve the accuracy of recommendations.

**References**

1. G. Hinton, S. Osindero, and Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets," Neural Computation, vol. 18, no. 7, pp. 1527-1554, 2006.
2. R. Salakhutdinov and A. Mnih, "Probabilistic Matrix Factorization," in Advances in Neural Information Processing Systems 20 (NIPS 2007), 2008.
3. J. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," Computer, vol. 42, no. 8, pp. 30-37, 2009.
4. X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," Advances in Artificial Intelligence, vol. 2009, Article ID 421425, 2009.
5. P. Resnick and H. R. Varian, "Recommender Systems," Communications of the ACM, vol. 40, no. 3, pp. 56-58, 1997.
6. A. Priyanshu, "Deep-Belief-Networks-in-PyTorch," GitHub repository, 2020. [Online]. Available: https://github.com/AmanPriyanshu/Deep-Belief-Networks-in-PyTorch. [Accessed: Jun. 3, 2024].
7. M. Abdelkhalek, "rbm-dbn," GitHub repository, 2020. [Online]. Available: https://github.com/mhdadk/rbm-dbn. [Accessed: Jun. 3, 2024].
8. "Deep Belief Network (DBN) in Deep Learning," GeeksforGeeks, 2023. [Online]. Available: https://www.geeksforgeeks.org/deep-belief-network-dbn-in-deep-learning/. [Accessed: Jun. 3, 2024].
9. "Restricted Boltzmann Machine," GeeksforGeeks, 2023. [Online]. Available: https://www.geeksforgeeks.org/restricted-boltzmann-machine/. [Accessed: Jun. 3, 2024].
10. Packt Publishing, "Hands-On-Deep-Learning-Architectures-with-Python," GitHub repository, 2019. [Online]. Available: https://github.com/PacktPublishing/Hands-On-Deep-Learning-Architectures-with-Python. [Accessed: Jun. 3, 2024].