Comparative Analysis of Neural Collaborative Filtering and Content-Based Collaborative Filtering Using Deep Learning: Book Recommendation.

A study by: Olanrewaju Shobowale (202211841)

Introduction

In a time when we primarily rely on the internet, recommender systems are a great tool for filtering online content. It is more difficult for users and business owners to get and develop customized content when tens of billions of data are generated every second. Businesses invest time in attempting to match customers with goods or services catered to their unique requirements, hence the requirement for recommender system(s). Since the first publications on collaborative filtering appeared in the mid-1990s, recommender systems have become a significant topic of research. Since there are many practical applications that assist users in coping with information overload and provide them with personalized recommendations, content, and services, there has been a lot of work done in both the industry and academia on developing new approaches to recommender systems. Interest in this area continues to be high.

By offering efficient and effective techniques for individualized recommendations, artificial intelligence (AI) has transformed the field of recommender systems. Deep learning-based approaches have demonstrated encouraging outcomes among these techniques in a number of areas, including book recommendations. Deep learning models' capacity to extract intricate patterns from vast amounts of data has allowed for the creation of sophisticated algorithms that can record human preferences and behavior as well as the features of the objects they interact with. Deep learning is part of a broader family of machine learning methods, which is based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised. Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher-level features from the raw input. Most modern deep learning models are based on artificial neural networks, specifically convolutional neural networks (CNN)s, although they can also include propositional formulas or latent variables organized layer-wise in deep generative models such as the nodes in deep belief networks and deep Boltzmann machines(Wikipedia).

Recommender systems work with users and items, with each user giving an item (or product) a rating (or preference value). In general, implicit or explicit metrics are used to collect user ratings. Implicit user ratings are unintentionally obtained from the user through interaction with the objects. On the other hand, the user directly provides explicit ratings by choosing a value on a finite scale of points or specified interval values. For instance, a website may compile implicit ratings for different products based on user involvement, clickstream data, and other variables. The majority of recommender systems use both explicit and implicit methods to get user ratings. After that, the results are put into a utility matrix, also known as a user-item matrix. The utility matrix is often relatively sparse and contains a large number of missing values because there are only ratings for a small number of user-item pairs.

	item1	item2	item3	item4
user1	2	5	1	3
user2	4	?	?	1
user3	?	4	2	?
user4	2	4	3	1
user5	1	3	2	?

Figure 1: A sample of a Utility Matrix

Types of Recommender Systems

There are three fundamental types of recommender systems namely, Content-Based Recommendations, Collaborative Recommendations, and Hybrid Recommender Systems but for this study, we shall be focusing on Collaborative filtering and Content-Based Filtering.

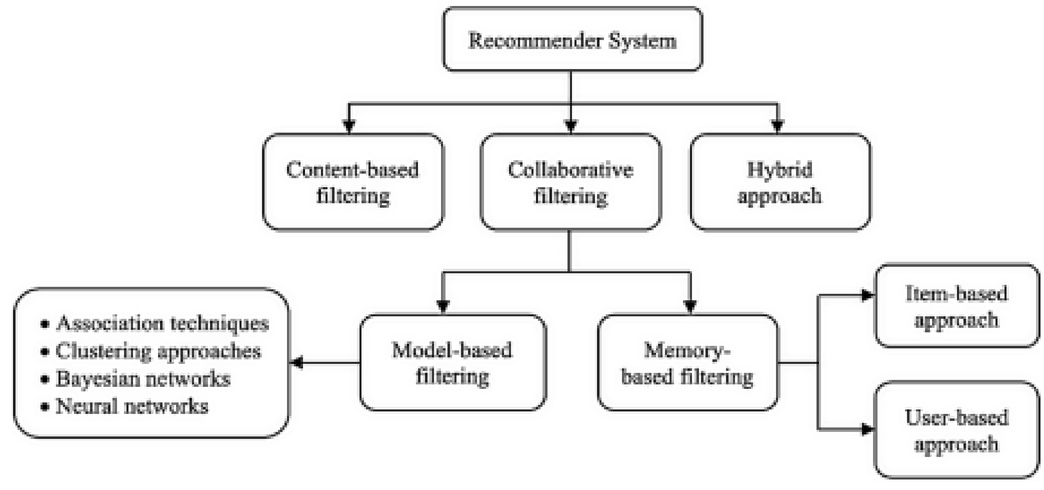

Figure 2: Types of Recommender Systems

Content-Based Filtering

With the help of the item's description and the user's ratings on a select few items, content-based filtering generates recommendations for products based on an estimation of the user's interest. The things that best match the user's interests and should, thus, be recommended are determined using this estimation. In other words, it suggests products that are comparable to the ones based on the ratings the customer has already given to them. It offers individualized content that is catered to the user's preferences, making it a widely desired method. It may also offer recommendations with little to no information about user preferences, which is a benefit. Additionally, the algorithm can still suggest new goods that have not yet been evaluated by any users based on the user's preexisting tastes or interests. Content-based recommender systems face challenges because of the vast amount of item sets, which makes it necessary to search through each item to find ones that match user preferences. As a result, performance decreases, especially for e-commerce companies with a large number of products. Additionally, it is important to consider each item's unique content, and processing unstructured data, particularly for multimedia items, is challenging. Although methods for evaluating multimedia material have improved, this persistent problem may discourage people from proposing multimedia items using content-based algorithms.

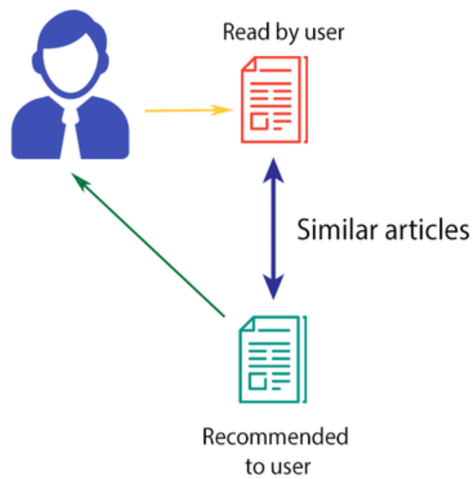

Figure 3: Content-based filtering

Collaborative Filtering

This method offers recommendations while taking into account the preferences of users with related interests. In other words, the utility of an object is evaluated for a user based on the usefulness given to the item by users who are similar to them. The decomposition of the user-item rating matrix into two lower-dimensional matrices, which represent the latent properties of users and objects, is the foundation of traditional collaborative filtering techniques. It is nonetheless constrained because it struggles to grow with sizable datasets. A recent method called Neural Collaborative Filtering (NCF) overcomes these drawbacks by modeling user-item interactions using neural networks. For the suggestion, it mixes collaborative filtering and deep learning. The user-item interaction sub-network and the fusion sub-network are the two parts of NCF. The user-item interaction sub-network determines the degree of similarity between users and items using their training-learned embeddings. The user and item embeddings are combined by the fusion sub-network to produce a final rating prediction. Several neural network architectures, including Multi-Layer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs), can be utilized to do this; however, for the purposes of this study, we primarily focused on MLPs.

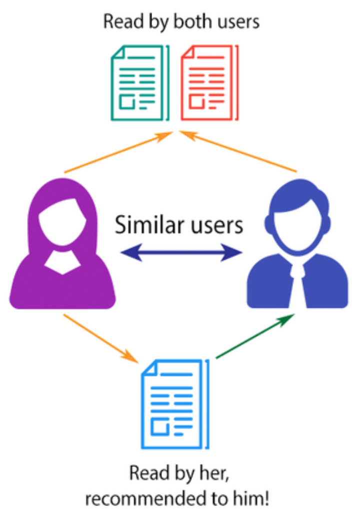

Figure 3.1: Collaborative-based filtering

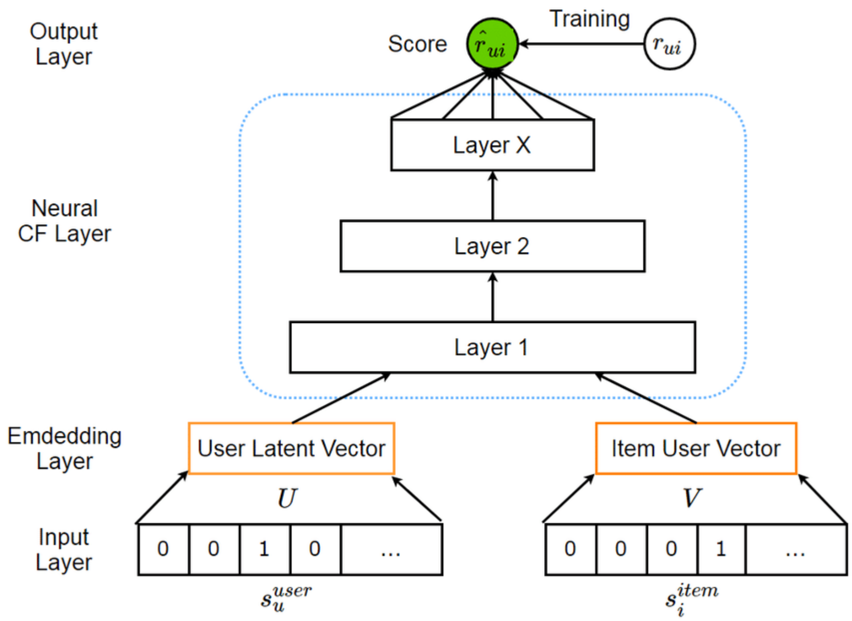

Figure 3.2: Neural Collaborative filtering

Multilayer Perception (MLP)

Both approaches in this study made use of MLP deep learning. A feed-forward neural network having multiple (one or more) hidden layers between the input layer and output layer is known as a multilayer perceptron (MLP). Since NCF uses two pathways to model users and items, it makes sense to concatenate the features of two pathways, a method frequently used in multimodal deep learning research. In order to learn the interaction between user and item latent attributes, hidden layers are added to the concatenated vector using a conventional MLP. MLPs can be thought of as learning hierarchical feature representations by stacking layers of nonlinear transformations. Universal approximators are another characteristic of MLPs.

Evaluation Metrics

These metrics are used to evaluate the effectiveness of the models. Accuracy ratings, mean square and root mean square errors, mean absolute errors, etc. We employed RMSE, MSE, and MAE for this research.

The average squared difference between the anticipated values and the actual values is what the MSE calculates. Note that MSE is sensitive to outliers, which can have a significant impact on the result. The average absolute difference between the expected and actual values is measured by MAE, not MSE. It offers a more accurate depiction of the total error and is less sensitive to outliers. The MSE and RMSE both offer comparable interpretations. The RMSE can, however, be more simply interpreted because it is expressed in the same units as the dependent variable. Better performance is indicated by lower RMSE, MSE, or MAE values, whereas a perfect score of 0 denotes that the predictions exactly match the measured data.

The goal of this study is to compare two deep learning-based methods for book recommendation: Neural Collaborative Filtering (NCF) and Content-Based Collaborative Filtering using deep learning. At the end of this study, we will find that the Content-based filtering method performs lower than the Neural collaborative filtering approach.

Background

With the success of deep learning in other domains in the mid-2010s, researchers began to explore its application to recommender systems. Techniques like Multilayer Perception (MLP), Autoencoders, Deep Belief Networks (DBNs), and Convolutional Neural Networks (CNNs) were adapted to develop more accurate and personalized recommendations.

It is possible to get user input openly or implicitly. A neural collaborative filtering-based recommender system for e-commerce websites was proposed by Li et al. in their 2018 paper. The system was developed to provide customers with personalized product recommendations based on users' prior interactions with the website (explicit feedback) as well as their browsing and search patterns (implicit feedback). integrating linked layers, which predict the user-item interactions, with embedding layers, which capture the latent features of people and items. Utilizing a dataset of user interactions with a big e-commerce website, they put their system to the test and compared it to a number of industry benchmarks, including matrix factorization and collaborative filtering. The results show that the suggested neural collaborative filtering methodology outperforms the alternatives in terms of accuracy and diversity of recommendations. This study uses the same methodology, however our dataset is different. On a book review dataset with explicit feedback (book rating) along with additional factors, we would be testing our NCF system. Regarding our evaluation metric and selection of recommendations, we anticipate that the NCF outperforms Content-based Filtering.

Objectives

The project will test the effectiveness of existing NCF and content-based filtering methods using a specific dataset. By contrasting NCF with Content-based filtering in terms of evaluation metrics, accuracy, and variety of suggestions, I hope to construct two (2) deep learning-based recommender systems for recommending books to users and assessing their performances. The study will aid in the creation of recommender systems that are more useful in the current internet economy.

Research Questions

1. How do the performance metrics of NCF and CNN compare in recommending books to users?
2. What are the factors that influence the performance of the recommender system?
3. How can we improve the accuracy and efficiency of the recommender system?

Methodology

Neural Collaborative Filtering.

Using MLP neural networks, we model user-item interactions. The method learns latent components and aggregates them through hidden layers to produce a projected preference score, in contrast to standard matrix factorization techniques. In order to achieve this, the model learns user and item embeddings, which are then combined through a number of dense layers to provide the anticipated preference score. This method is put into practice by first encoding the user and item data and defining the embedding size. The output layer, hidden layers (which combine both embeddings, users, and items), and input layer are then used to construct the neural network. The dense neural layers for the user and item embeddings have a Relu activation function and a decreasing number of units in powers of two (256, 128, 64). The Adam optimizer and mean squared error loss function are used to create the model. Following that, the model is fitted into a test and train dataset using an 80:10:10 training, testing, and validation ratio. The model is constructed using 10 epoch runs and a batch size of 128. The performance of the model is then assessed using the performance evaluation metrics MSE, MAE, and RMSE.

Content-based Filtering.

There are several similarities to the NCF strategy mentioned above. The difference is that we used Word2vec to integrate the user preference (book author). Word2vec is a method for learning word embeddings, which are vector representations of words in a high-dimensional space, through the use of natural language processing. Words can be represented as dense, low-dimensional vectors that capture semantic and grammatical commonalities using this neural network-based method. Using the same strategy as before, but this time we activated the Relu and Sigmoid in the dense layers.

Experiments

Neural Collaborative Filtering

The model is created using the TensorFlow framework, and the code is written in Python. The BX-Books dataset (https://www.kaggle.com/datasets/ruchi798/bookcrossing-dataset), which contains ratings and details about books, authors, and users, was used for this project. Using a test size of 0.2 and a random seed of 42, the dataset is divided into train and test sets. The model is trained using a batch size of 128 across 10 epochs. The following hyperparameters were used to train the model: Embedding size is 50, the number of dense layers is 3, the number of neurons in the first dense layer is 256, the number of neurons in the second dense layer is 128, and the number of neurons in the third dense layer is 64. ReLU, Adam as the optimizer, and MSE as the loss function.

Baseline and Evaluation Metrics:

Three fundamental techniques are used to gauge the model's performance:

1. Random Recommender: This standard chooses books at random to suggest to users.
2. Popularity Recommender: This standard suggests to users the most well-liked publications. The quantity of ratings each book has gotten serves as a gauge of its popularity.
3. Item-based Collaborative Filtering Recommender: This baseline suggests books to users using item-item collaborative filtering. The cosine similarity between their feature vectors is used to determine how similar the two books are.

MSE, MAE, and RMSE are the assessment metrics used to assess the performance of the model and the baselines.

Content-Based Filtering

The ratings, user, and book information are combined as part of the preprocessing of the dataset, after which the records with null values are removed. With a test size of 0.2 and a random state of 42, the resulting dataset is divided into train and test sets. The following hyperparameters were used to train the model: For training the Word2Vec model to extract characteristics from book authors, set the window size to 5, the minimum count to 1, and the number of workers to 4. The input layer for the neural network model has a shape of 100, and the three hidden layers each include 256, 128, and 64 neurons. Relu and sigmoid are utilized as activation functions for the hidden layers, but there is no activation function employed for the output layer. Mean squared error is the loss function and Adam is the optimizer used when training the model. The model is trained using a batch size of 128 across 10 epochs.

Baselines and Evaluation Metrics:

Three baseline models are utilized to assess the approach's performance. The first baseline is the Popularity-Based model, which ranks books according to how well-liked they are overall, or how many ratings they have received. The User-Based Collaborative Filtering model, which makes book recommendations based on how closely a user's rating history matches that of other users in the dataset, serves as the second baseline. The Item-Based Collaborative Filtering model, the third baseline, suggests books based on how well their ratings match those of other books in the collection. The metrics for evaluation remained the same.

Results

Neural Collaborative Filtering

On the test dataset, the model produced MSE of 16.716, MAE of 2.706, and RMSE of 4.089. The baseline model, which suggested the most well-liked novels in the dataset, was contrasted with these findings. The baseline model underperformed the neural network-based collaborative filtering strategy, demonstrating that the model was able to recognize significant patterns in the data. It's interesting to note that the model did perform better as the number of epochs rose. In conclusion, the collaborative filtering method based on neural networks was effective in generating book recommendations from the Book-Crossing dataset based on user ratings.

```
Epoch 1/10
6445/6445 [==============================] - 38s 5ms/step - loss: 11.7057 - val_loss: 11.2604
Epoch 2/10
6445/6445 [==============================] - 27s 4ms/step - loss: 9.0504 - val_loss: 12.2969
Epoch 3/10
6445/6445 [==============================] - 27s 4ms/step - loss: 7.6117 - val_loss: 12.9044
Epoch 4/10
6445/6445 [==============================] - 27s 4ms/step - loss: 5.9238 - val_loss: 13.5612
Epoch 5/10
6445/6445 [==============================] - 27s 4ms/step - loss: 4.1469 - val_loss: 14.5956
Epoch 6/10
6445/6445 [==============================] - 27s 4ms/step - loss: 2.9887 - val_loss: 15.2286
Epoch 7/10
6445/6445 [==============================] - 27s 4ms/step - loss: 2.3039 - val_loss: 16.1148
Epoch 8/10
6445/6445 [==============================] - 27s 4ms/step - loss: 1.8429 - val_loss: 16.0324
Epoch 9/10
6445/6445 [==============================] - 27s 4ms/step - loss: 1.5222 - val_loss: 16.7990
Epoch 10/10
6445/6445 [==============================] - 27s 4ms/step - loss: 1.3068 - val_loss: 16.5369
<keras.callbacks.History at 0x7fcea28d3f70>
```

```
6445/6445 [==============================] - 9s 1ms/step
Mean Squared Error: 16.536882030642257
Mean Absolute Error: 2.756069279875695
Root Mean Squared Error: 4.066556532330795
```

Content-Based Filtering

The model reportedly had an MSE of 14.55, an MAE of 3.50, and an RMSE of 3.81, according to the results. Lower values represent greater performance, and they offer insight into the model's accuracy. When compared to the baselines, the model's performance is not very strong but appears to be reasonable.

The model's loss gradually decreased during training, proving that it had learned the fundamental patterns in the data. However, the validation loss varied, indicating that there might be overfitting concerns with the model.

.

```
Epoch 1/10
6445/6445 [==============================] - 24s 3ms/step - loss: 14.7975 - val_loss: 14.7865
Epoch 2/10
6445/6445 [==============================] - 25s 4ms/step - loss: 14.6854 - val_loss: 14.6436
Epoch 3/10
6445/6445 [==============================] - 23s 4ms/step - loss: 14.6435 - val_loss: 14.6270
Epoch 4/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.6158 - val_loss: 14.6644
Epoch 5/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5903 - val_loss: 14.6485
Epoch 6/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5734 - val_loss: 14.5906
Epoch 7/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5542 - val_loss: 14.5709
Epoch 8/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5395 - val_loss: 14.5816
Epoch 9/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5221 - val_loss: 14.5769
Epoch 10/10
6445/6445 [==============================] - 22s 3ms/step - loss: 14.5085 - val_loss: 14.5538
<keras.callbacks.History at 0x7f1cb3a661c0>
```

```
6445/6445 [==============================] - 10s 2ms/step
Mean Squared Error: 14.55380140412487
Mean Absolute Error: 3.4984036164277743
Root Mean Squared Error: 3.8149444824433383
```

Conclusion

In conclusion, the MAE validation loss report and the loss function at the end of epoch runs show that the NCF performs content-based filtering even though the MSE and RSME of the content-based filtering outperformed the neural collaborative filtering. These factors include the size of the dataset, the setting of the hyperparameters, the number of epochs, the embedding method, etc. By adding more regularization strategies or by increasing the amount of training data, this scenario could be avoided. However, there is potential for development, and further research may be done to enhance the model's functionality. For instance, more intricate architectures can be investigated or further characteristics added to the model.

Referencing

Deepjyoti Roy & Mala Dutta, Journal of Big Data, (2022). A systematic review and research perspective on recommender systems. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00592-5

Adomavicius, G., & Tuzhilin, A. (2005). Towards the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734-749. http://ids.csom.umn.edu/faculty/gedas/papers/recommender-systems-survey-2005.pdf

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web (pp. 173-182). ACM. https://liqiangnie.github.io/paper/p173-he.pdf

Gediminas Adomavivius, Konstantin Bauman et al. CARS: Workshop on Context-Aware Recommender Systems, (2022). RecSys '22: Sixteenth ACM Conference on Recommender Systems. https://www.researchgate.net/publication/363645771_CARS_Workshop_on_Context-Aware_Recommender_Systems_2022

Zhang, S., Yao, L. Deep Learning Based Recommender System: A Survey and New Perspectives, (2017). https://www.researchgate.net/publication/318671349_Deep_Learning_Based_Recommender_System_A_Survey_and_New_Perspectives

Jesús Bobadilla, Fernando Ortega, Abraham Gutiérrez et al. Classification-based Deep Neural Network Architecture for Collaborative Filtering Recommender Systems. Universidad Politécnica de Madrid, Carretera de Valencia Km 7, 28031 Madrid (Spain) https://core.ac.uk/download/pdf/523306188.pdf

Li N, Guo B, Liu Y, Jing Y, Ouyang Y, Yu Z 2018 Commercial site recommendation based on neural collaborative filtering UbiComp 2018 138-141