Neural Collaborative Filtering Augmentations

Jakub Sachajko 179976
Przemysław Rośleń 180150

Politechnika Gdańska

Keywords: Neural Colaborative Filtering, recommendation systems, NeuMF, MLP, GMF, regularization,

Original article: Neural Colaborative Filtering

Authors: Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, Tat-Seng Chua

1 Abstract

In the digital era, recommendation systems have become indispensable in navigating the vast sea of online information. This study delves into enhancing recommendation systems through Neural Collaborative Filtering (NCF), a deep learning-based approach. While traditional recommendation methods, such as matrix factorization, have shown proficiency, they often fall short in scenarios with new users or items and large data sets. Our research aims to address these challenges by integrating and experimenting with NCF algorithms, specifically focusing on its two main components: Generalized Matrix Factorization (GMF) and Multi-Layer Perceptron (MLP).

We explore the potential of NCF in improving recommendation accuracy, particularly in cases where only implicit feedback is available. By experimenting with various modifications to the NCF framework, including adjustments in algorithm parameters and the introduction of regularization techniques, our goal is to enhance the system's predictive capabilities. The fusion of GMF and MLP in a novel approach, termed NeuMF (Neural Matrix Factorization), seeks to leverage both linear and non-linear processing for a more comprehensive understanding of user-item interactions.

This paper provides a detailed analysis of existing modifications to the collaborative filtering algorithm and introduces new perspectives on utilizing neural networks in recommendation systems. Extensive experiments are conducted on real-world datasets, such as MovieLens, to validate the effectiveness of the proposed methods. Metrics like Normalized Discounted Cumulative Gain (NDCG) and Hit Rate (HR) are employed to assess the performance of the modified NCF models.

2 Introduction

In today's world, where we are surrounded by an overwhelming amount of information, recommendation systems have become very important. These systems help us find what we need and like on various online platforms, like shopping websites, news sites, and social media. They do this by looking at what we have liked or chosen in the past, a process known as collaborative filtering. Our research is focused on making these recommendation systems better by using a special kind of technology called deep neural networks, especially for suggestions made without direct feedback from users.

One common method used in these systems is called matrix factorization. It's like creating a map where both users and items are points, and the system tries to predict how much a user will like an item based on where they are on this map. This method became very popular after the Netflix Prize, a big competition for improving Netflix's recommendation system. Matrix factorization is great because it can simplify complex data and make better suggestions, but it's not perfect. It struggles with new users or items that don't have much data, and it can be hard to use for very large amounts of data.

Deep learning, the technology we are using, has been great in areas like understanding speech, recognizing images, and processing language. However, it hasn't been used as much for recommendation systems. Given how well it works in other areas, we think it can really improve how these systems suggest things to users. So far, deep learning has been used in recommendation systems mainly for extra information, like understanding what an item is about or what features it has. But there's a lot more it can do.

This paper is mainly about trying new things with a specific type of deep learning called Neural Collaborative Filtering. We've seen some good results from this method in other research, but we think we can do even better. Our main goal is to experiment with different ways to tweak and improve this method. We want to see if our changes can make the system suggest even better things to users than what the original research showed. By doing this, we hope to make a big contribution to how recommendation systems work and make them even more helpful for everyone.

3 Neural Collaborative Filtering

3.1 General Framework

The Neural Collaborative Filtering (NCF) algorithm is a way to improve recommendation systems, which suggest items to users online. It works by using a layered approach to process information.

Imagine a user and an item each described by a unique set of features. These features are turned into what's called a "one-hot encoded" format, which is just a special way of representing them so that the algorithm can understand. This representation is then transformed into a more compact form, known as an "embedding vectors".

These embedding vectors, which represent the users and items, are then passed through several layers of the neural network. Think of these layers as filters that try to learn and find patterns in how users interact with items. The final goal of this process is to predict a score that tells us how likely a user is to be interested in a certain item.

To train this system, we use a method where we compare the predicted scores with the actual user-item interactions. This helps the algorithm learn better. One of the key points of NCF is that it treats the recommendation problem like a classification task, where each interaction (or lack of it) is viewed as a binary value, like 'yes' or 'no'.

In short, NCF tries to predict how much a user will like an item based on their past behavior, using a complex network of layers that learn from data. This method is particularly focused on cases where we only know if a user has interacted with an item or not, without additional details like ratings.

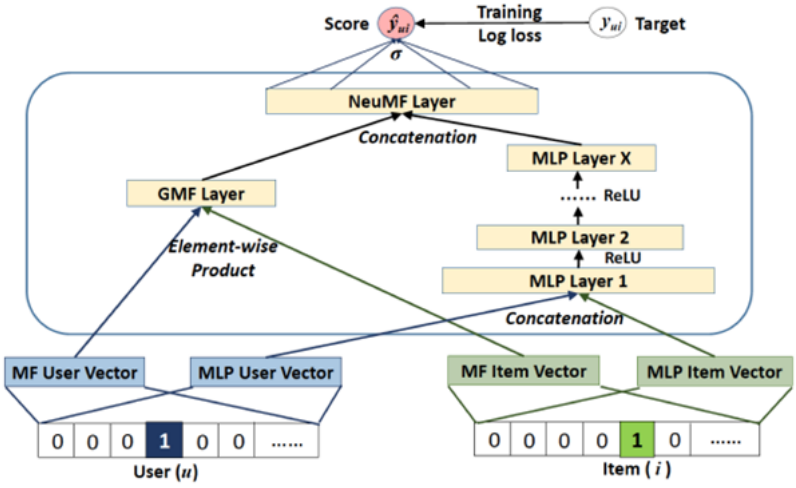

Fig. 1. Neural collaborative filtering framework

3.2 Generalized Matrix Factorization (GMF)

The Generalized Matrix Factorization (GMF) is a part of Neural Collaborative Filtering that deals with recommending items to users, like suggesting movies or products. GMF is a special version of a popular method called Matrix Factorization (MF), which is commonly used in recommendation systems.

In GMF, we start by representing each user and item with unique vectors (we call them latent vectors) based on their IDs. These vectors are created through a process called one-hot encoding, which is just a fancy way of turning user and item IDs into a format that the GMF can understand and work with.

Once we have these vectors, GMF combines the user vector and the item vector using an operation called the element-wise product. This operation multiplies each element of the user vector with the corresponding element in the item vector.

After this, the combined vector is passed through an output layer. This layer uses a specific function (called an activation function) and a set of weights to calculate the final score. This score predicts how likely a user is to be interested in a particular item.

What makes GMF special is that it can be adjusted and improved in different ways. For example, by changing the weights and the activation function, GMF can either stick to the traditional Matrix Factorization approach or adopt a more complex, non-linear approach. In our research, we use a function called the sigmoid function and adjust the weights based on the data, aiming to make better predictions.

In summary, GMF is a flexible and improved version of the traditional Matrix Factorization method used in recommendation systems, designed to better predict user preferences.

3.3 Multi-Layer Perceptron (MLP)

The Multi-Layer Perceptron (MLP) is another key part of Neural Collaborative Filtering, which is all about making better recommendations, like suggesting movies or products to online users. Unlike the GMF method, MLP takes a different approach to understand user preferences.

In MLP, we first create individual representations (or vectors) for users and items. But instead of combining these vectors with a simple multiplication, like in GMF, we put them together end-to-end. This is like putting two lists side by side. However, just putting them side by side isn't enough to really understand how a user's preferences match with an item's features.

To solve this, we add several layers of processing, known as hidden layers, on top of this combined vector. These layers are part of what makes MLP a "Multi-Layer" Perceptron. Each layer takes the information from the previous layer and processes it further, using specific functions and adjustments (like weights and biases). This process helps MLP learn and understand the complex relationship between what users like and the features of different items.

One important part of MLP is choosing the right function for each layer. There are a few options, like sigmoid, tanh, and ReLU. Each of these functions processes the data differently. For instance, ReLU is often a good choice because it handles data in a way that avoids certain technical problems and helps the model learn better from sparse data (which is common in user-item interactions).

Finally, MLP is usually designed in a "tower" pattern. This means the first layer (at the bottom) has the most processing units, and each layer above it has fewer and fewer units. This design helps the model learn more abstract and complex features of the data as it moves through the layers.

In summary, MLP in Neural Collaborative Filtering is about combining user and item information in a detailed way and then using multiple layers of processing to better understand and predict what users will like.

3.4 NeuMF - The Fusion of GMF and MLP

NeuMF model is a blend of two other models we worked on: GMF (Generalized Matrix Factorization) and MLP (Multi-Layer Perceptron). By combining these two, NeuMF aims to make even better recommendations to users.

The main idea behind NeuMF is to take the simple, linear approach of GMF and mix it with the more complex, non-linear approach of MLP. This way, we get the best of both worlds.

Instead of making GMF and MLP share everything, we let them learn separately. This means they each have their own way of understanding users and items. Once they have learned their parts, we combine what they have learned into one model. This helps NeuMF to not be limited by the constraints of either GMF or MLP.

After combining the learning from GMF and MLP, NeuMF uses this combined understanding to predict how likely a user is to like a particular item. It does this by considering both the simple matches (like in GMF) and the more complex relationships (like in MLP).

4 Review of existing modifications to the algorithm

The paper provides a comprehensive review of existing modifications to the collaborative filtering algorithm, shedding light on the evolution of recommendation systems. While traditional collaborative filtering methods heavily rely on explicit feedback such as ratings and reviews, the emergence of implicit feedback has sparked a wave of innovation. The authors emphasize the challenges associated with utilizing implicit feedback, including the scarcity of negative feedback and the need to model noisy signals accurately. In response to these challenges, the paper introduces a neural network architecture, the Neural Collaborative Filtering (NCF) framework, as a pioneering approach to modeling latent features of users and items. This framework not only presents a novel way to interpret matrix factorization (MF) but also leverages multi-layer perceptrons to introduce high levels of non-linearities, addressing the limitations of traditional MF

models. Furthermore, the paper showcases extensive experiments on real-world datasets, demonstrating the effectiveness of NCF approaches and the promise of deep learning in collaborative filtering. This review underscores the transformative potential of neural networks in revolutionizing recommendation systems and highlights the ongoing quest for more effective and efficient methods in the field.

5 Metrics

5.1 NDCG

Normalized Discounted Cumulative Gain (NDCG) is a metric commonly used to evaluate the ranking quality of a model's output. It considers both the relevance and the ranking of items. The formula for NDCG is given by:

$$NDCG = \frac{1}{Z} \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

where:

- Z is the normalization factor (usually the ideal ranking's NDCG),
- k is the number of items,
- rel_i is the relevance of the item at rank i.

5.2 HR (Hit Rate)

Hit Rate (HR) measures the proportion of correctly predicted items in the top k recommendations. It is calculated as:

$$HR = \frac{Number of Hits in Top k}{Total Number of Test Items}$$

where:

- Number of Hits is the count of correctly predicted items in the top k,
- Total Number of Test Items is the total number of items in the test set.

5.3 Squared Loss (without Regularization)

The squared loss, also known as mean squared error (MSE), is a common loss function used in regression problems. It measures the average squared difference between the predicted values and the actual values. The formula for squared loss without regularization is:

$$Loss_{SQR} = w_i \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

where:

- N is the number of data points,
- y_i is the true target value for data point i,
- $\hat{y}_i$ is the predicted value for data point i.

5.4 Lasso Loss (with Regularization)

The Lasso loss, also known as L1 regularization, introduces a penalty term based on the absolute values of the model coefficients. This helps in inducing sparsity in the model. The formula for Lasso loss with an additional regularization parameter (λ) is:

$$Loss_{Lasso} = \sum_{i=1}^{N}(y_i - x_i\hat{\beta})^2 + \lambda \sum_{j=1}^{m} |\beta_j|$$

where:

- N is the number of data points,
- y_i is the true target value for data point i,
- x_i is the predicted value for data point i,
- m is the number of features,
- β_j is the weight (coefficient) associated with feature j,
- λ is the regularization parameter.

6 Datasets

In this study, we conduct a comprehensive analysis of the MovieLens dataset, a widely-used resource in collaborative filtering and recommendation system research. Our exploration covers the dataset's various versions, including MovieLens 100K, 1M, 10M, and 20M, each offering a differing scale of movie ratings and user information. We delve into the dataset's structure, examining key components such as user IDs, movie IDs, ratings, timestamps, and genres. The analysis focuses on understanding user behavior, preferences, and trends by scrutinizing the distribution of movie ratings. Additionally, we discuss the dataset's applications in developing recommendation algorithms, particularly in the realms of collaborative filtering, content-based filtering, and hybrid models. However, we acknowledge potential challenges such as biases in the data, including popularity bias, and the dataset's limitations in capturing diverse user tastes. The study concludes with a summary of key findings and insights, as well as considerations for future research in the dynamic field of recommender systems.

7 Experiments

7.1 Experiment 1 - Modifications of algorithm parameters

Experiment Steps:

- We trained each of the three NCF algorithms for 3 different values of the learning rate parameter and for 4 different values of the MLP layers, optimizer parameters.
- We evaluated the results taking into account HR and NDCG metrics.
- We compared the obtained results with those obtained in previous experiments.

7.1.1 Different learning rate values .

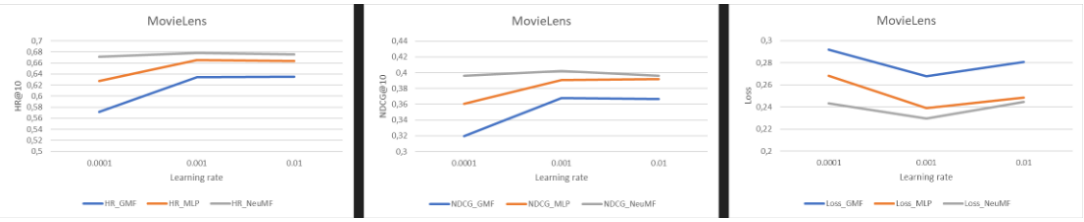

Fig. 2. Experiment 1.1 Different learning rate values

7.1.2 Various cost function optimizers .

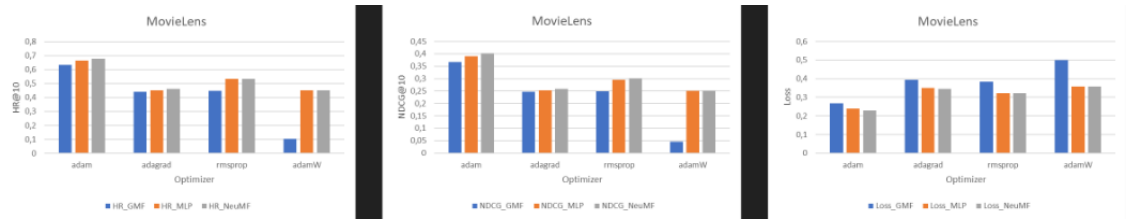

Fig. 3. Experiment 1.2 Various cost function optimizers

7.1.3 Different sizes of MLP layers .

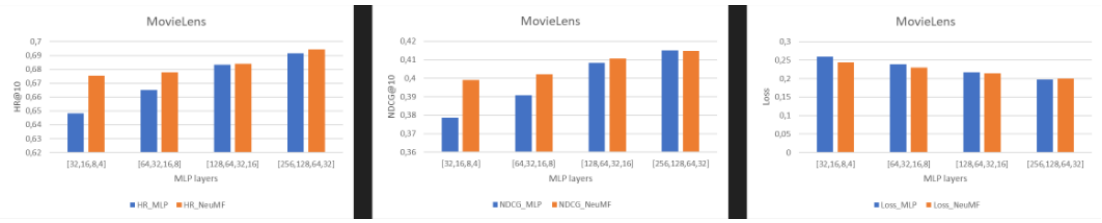

Fig. 4. Experiment 1.3 Different sizes of MLP layers

7.1.4 Comparison of results with original experiments Adding more neurons in the MLP layers allowed for a lower Loss value than in previous experiments (< 0.2). The best results of HR and NDCG metrics in the modified experiment unfortunately did not exceed the highest results from previous experiments.

Conclusion 1: Modifications to the learning rate and optimizer parameters did not produce better results (default values are the best).

Conclusion 2: Increasing the number of neurons in MLP layers using the MLP layers parameter allowed for better results.

7.2 Experiment 2 - Regularization

As part of the experiment, the MovieLens dataset was utilized, conducting a series of tests regarding model parameters. Specifically, changes were made to the reg_layers parameters (defaulted to '[0,0,0,0]') and reg_mf (defaulted to 0). These changes involved converting these parameters into float numbers ranging from 0 to 1. Additionally, for reg_layers, it was possible to select a specific layer for L2 Lasso regularization.

Tests were conducted for various regularization parameter values, namely "1, 0.1, 0.01, 0". The results obtained from these experiments were carefully compared and presented in the form of graphs. It is worth noting that initially, regularization values were tested with excessively high values, prompting further modifications by reducing values to, for example, 0.1 or 0.01.

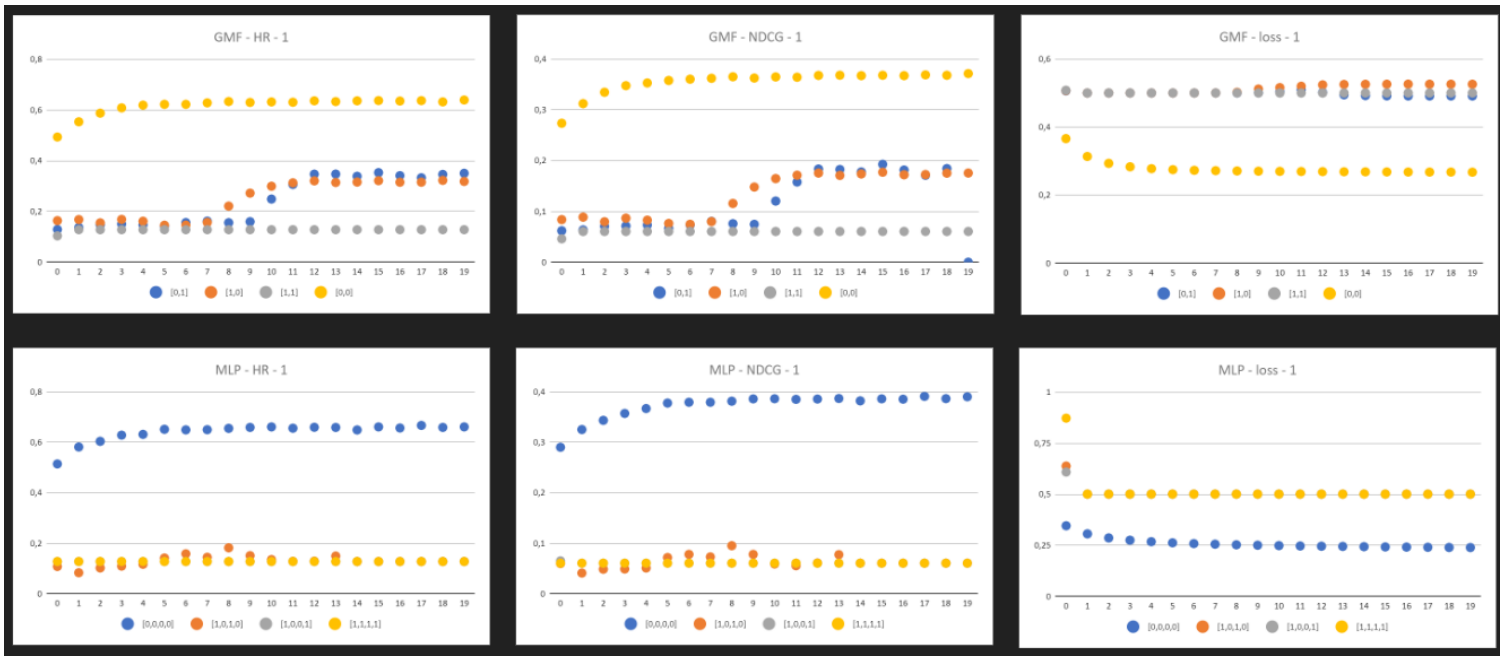

Fig. 5. Experiment 2.1 GMF, MLP regularization = 1

Fig. 6. Neural collaborative filtering framework

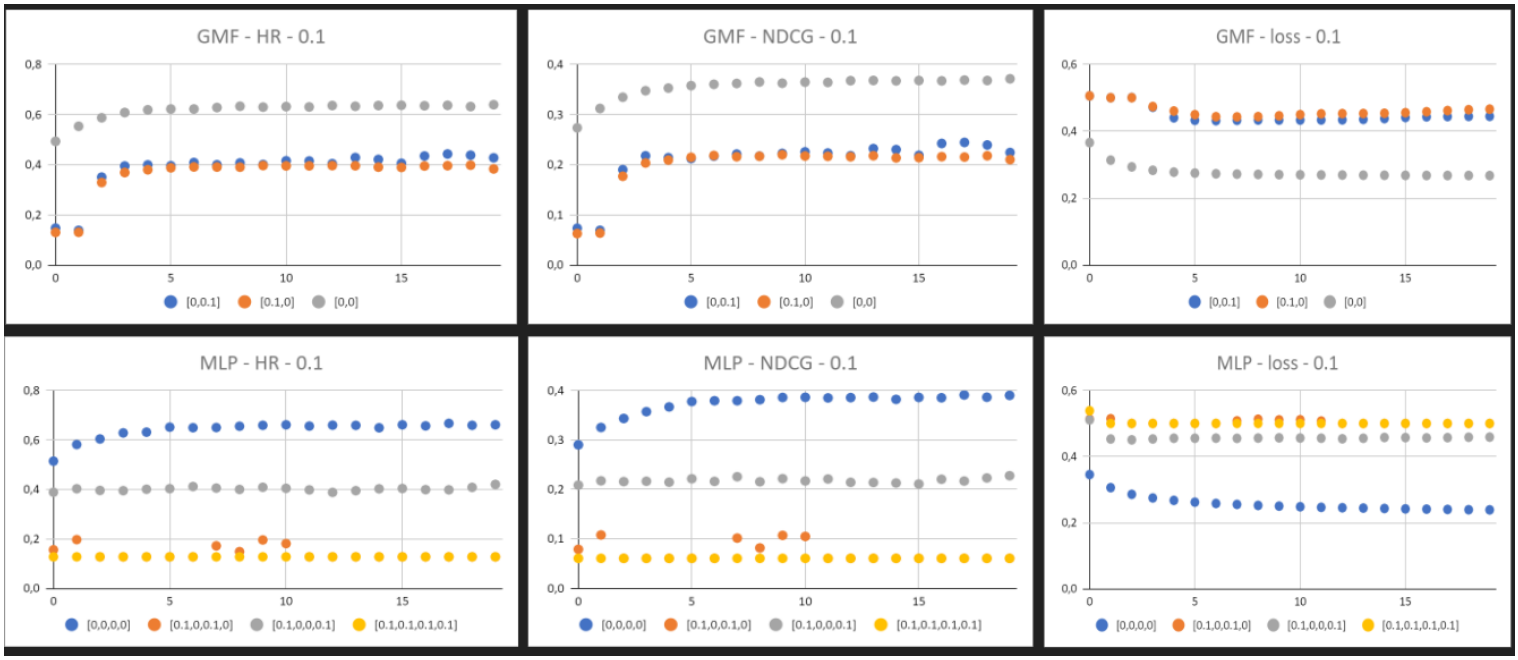

Fig. 7. Experiment 2.2 GMF, MLP regularization = 0.1

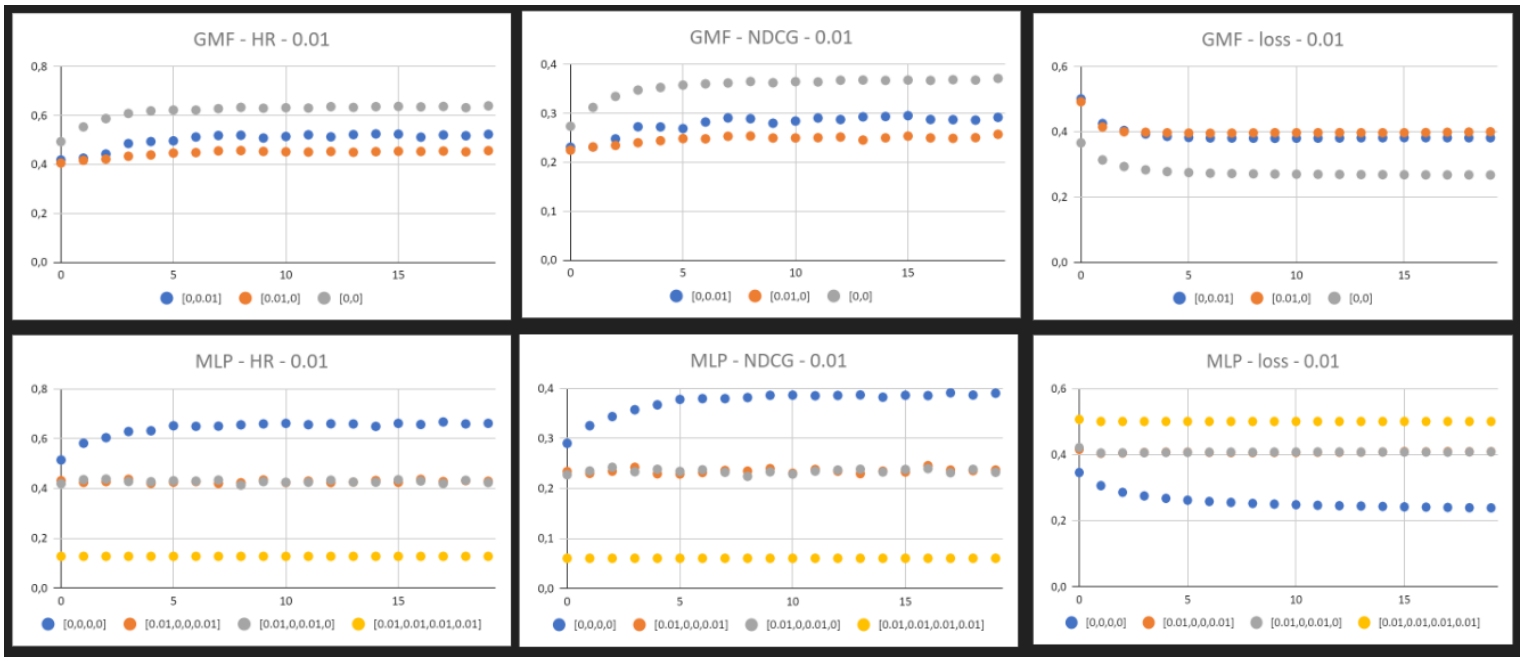

Fig. 8. Experiment 2.3 GMF, MLP regularization = 0.01

7.2.1 Comparison of results with original experiments Unfortunately, the results of experiments with regularization were inferior to those obtained without its application. It was observed that the smaller the regularization value, the less difference there was between the obtained values for NDCG, HR, and loss metrics.

Particularly significant was the finding that every model achieved the best results without regularization, and the introduction of regularization had practically no significant impact on improving the values. In the context of further research, it was suggested to explore values smaller than 0.01, such as 0.0001, to see if there exists a range of values that could positively influence the results.

8 Conclusions

Wprowadzone modyfikacje parametrów, takich jak learning rate oraz optimizer, nie przyniosły oczekiwanych rezultatów. Zwiększenie liczby neuronów w warstwach MLP pozytywnie wpłynęło na osiągane wyniki, sugerując, że rozszerzenie architektury sieci może być kluczowym elementem optymalizacji. Nawet minimalne wartości parametru regularyzacji nie przyczyniły się do znaczącej poprawy efektywności modelu. Dla obszernych zbiorów danych zaleca się stosowanie niewielkich wartości parametrów regularyzacji w okolicach 0.01 oraz niższych, co może stanowić istotny element optymalizacji dla dużej ilości danych. Przyjęta regularyzacja nie zawsze przyniosła korzystne efekty, co sugeruje, że wpływ na wyniki może być zmienny w zależności od specyfiki zbioru danych. Dodatkowo, eksperymenty z dodaną regularyzacją nie przyniosły oczekiwanych korzyści, co sugeruje, że w tym konkretnym kontekście to podejście może być mniej efektywne. Warto również rozważyć alternatywne strategie modyfikacji eksperymentów, takie jak zwiększenie liczby epok, co prawdopodobnie mogłoby prowadzić do uzyskania lepszych rezultatów.

References

1. Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web (WWW '17). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569
2. https://github.com/hexiangnan/neural_collaborative_filtering
3. https://medium.com/data-science-in-your-pocket/recommendation-systems-using-neural-collaborative-filtering-ncf-explained-with-code
4. https://d2l.ai/chapter$_{r}ecommender - systems/neumf.html$
5. https://www.width.ai/post/neural-collaborative-filtering
6. https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/recommender/neural_collaborative_filterings