# Neural Network Based Prediction in Recommender system

Karishma Nanda
Department of Computing and Informatics
Bournemouth University, Poole BH12 5BB, U.K.
ID https://orcid.org/0000-0002-8955-9956
https://www.researchgate.net/publication/346787617_Neural_Network_Based_Prediction_in_Recommender_system

**ABSTRACT- This paper aims to solve the cold start problem in recommender system with Neural Network based approach. There are several attempts in academia and in the industry to improve the recommender system. For instance, latent matrix factorization is an algorithm that solves the recommendation problem, it produces efficient outcomes from the core problem. Latent factors are not directly observed but are inferred from other factors. It can be computed by assuming a specific number of such factors and then transforming the large user-item matrix into a smaller matrix based on previously assumed factors. These smaller matrices can be multiplied to reproduce a close approximation to the original user-item matrix using a technique called matrix factorization. Assuming that the matrix can be written as the product of two low-rank matrices, matrix factorization techniques seek to retrieve missing or corrupted entries. Matrix factorization approximates the matrix entries by a simple fixed-function — namely, the inner product — acting on the corresponding row and column latent feature vectors. Substituting a neural architecture for the inner product that learns from the data, improves recommendation problem and deals with the cold start problem.**

***KEYWORDS - Recommender system, collaborative filtering, latent factor, content-based, matrix factorization, inner product, cold start problem, neural network.***

## I. Introduction

Prediction is important in decision making and optimum decision making is an important virtue of an intelligent system. Designing an intelligent system (like recommender system), which can make an optimum prediction for a given problem is an interesting question to research, because decision making depends on it. Artificial Intelligence (AI) provides a framework where a designed model can solve problems without much human interaction similarly a model in Data Science can effectively read and analyse patterns in a data set.

A recommender system is a subclass of information filtering system that deals with the problem of digital data overload to filter items or information according to the user's preferences, interests, and behaviour. There are two popular kinds of recommender systems, content-based [1]and collaborative filtering [2]. Several techniques are used for creating suggestions for both content-based and collaborative filtering. BM25 weighting, for instance, to help compare products and users with vastly different behaviours, effective nearest neighbour search to find the highest scoring recommendations, and matrix factorization to predict the preference of a user for each item. Prediction techniques can be classified into three general categories: Expert judgement, Algorithm models and Machine learning. Prediction based on Machine learning includes fuzzy logic [3], regression trees [4], artificial neural networks [5], and case-based reasoning [6]. These techniques have been used in the last few years on predictive models. Artificial neural networks (ANN) get intuition from how the real brain works, making it a fascinating topic for research. It has proven advantages over another machine learning algorithm [7].

There are multiple use cases where prediction-based AI models can be utilised to enhance the decisioning process. For instance, a churn risk model can be used to reduce employee attrition, enhance customer retention. A next best action process can be built around multiple attributes for cross-selling or up selling on an e-commerce website. End-user experience can be designed using these models which will enhance the seamless browsing experience on a normal website by filtering information.

Other user presumptions about items are not considered in content-based prediction. Finding items (entities a system recommends) is not tailored with a specific user. Collaborative filtering works on "What is popular among my peers" [8]. Interaction between item and user information are key factors in modelling collaborative filtering. Collaborative filtering utilizes matrix factorization and applied an inner product on the latent features of items and users. By replacing the inner product with a neural network. A neural network learns an arbitrary function from data, and this property of neural network enhance model performance over state-of-the-art methods [9].

The paper analyses a design and mathematical view of how Neural Network (NN) based prediction can enhance decision making in recommender systems and consists of six sections. Section 2 explains different methodologies used to solve the recommendation problem, its advantages and disadvantages; Section 3 formulates the problem; Section 4 analyses neural network-based solution and explains why and how it enhances the system; Section 5 implements (only at design level) the concepts of neural network based embedding layers with Keras (Python library) and MovieLens dataset[1]. Finally, Section 6 concludes this paper with possible future scope.

## II. Methodology

This section evaluates different approaches which have been used to solve the recommendation problem and examine its advantages and disadvantages.

### A. *Content-based*

There are different methodologies to implement a recommendation in recommender system in this section content-based method is explained with its advantages and disadvantages.

A content-based filtering method treat recommendation as a user-specific classification problem. In an online shop setting, this would be recommending future items based on what items the user had previously purchased or rated. This method can capture the specific interests of a user and can recommend niche items that no other users are interested in. Items' description and title are converted into a vector of numbers with a technique called "bag-of-words" (BOW). BOW equally weights all words, two items will be considered similar even if they use quite common words. Term Frequency-Inverse Document Frequency (TF-ID) technique is used to penalize the frequently occurring words. Bayesian classifier or machine learning approaches are used to predict user's preference about an item. Content-based recommendation works well for new users who do not have a purchase history. The disadvantages of this method are that it only makes predictions based on the existing interest of users buying and reviewing items. Other users review makes no difference.

### B. *Collaborative filtering*

In this section, the collaborative filtering-based prediction method is explained with their advantages and disadvantages.

Collaborative filtering is the method of making automatic predictions about the preferences of a user by collaborating information from many users. It assumes that users will have similar preferences. Collaborative filtering can predict and recommend items to a user based on how similar users have acted to the item. It uses three methods to address the recommendation problem. Firstly, the neighbourhood-based model, which uses the most similar user interaction information to predict items for the current user (the user who is currently purchasing). This approach has a scalability problem. Other models are Classification and Matrix Factorization (MF) based [10]. There are two major collaborative filtering algorithms; User-user collaborative filtering; Item-item collaborative filtering. Similar users' preferences are weighted more and weights are generally calculated by Pearson correlation coefficient or Cosine similarities [11]. It works on "find user which is similar to the current user" (same row element figure2). For the latter prediction is based on the similarity between items.

For a movie recommendation system feedback about movies is categorised in two types: Explicit and Implicit. In explicit feedback, users make a specific effort to provide information. With explicit ratings, a low rating can indicate that a user does not prefer the item(product) and these negative ratings can help the recommendation system filter out poor recommendations. Implicit feedback makes sparse matrix (see figure1 below) and comprises likes or even mouse movement [12]. With implicit feedback, there is no way to model negative feedback [8]. The collaborative nature of this approach is evident when the model learns the embeddings [12]. Backpropagation uses gradient descent or one of its variants to solve the problem.

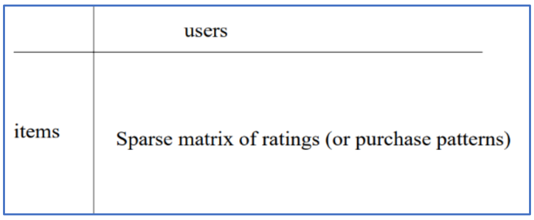



*Figure 1. User item sparse matrix.*

### C. *Matrix Factorization*

Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. This approach arose in popularity during Netflix Prize Challenge. Matrix factorization exploits the similarities in user's preferences to solve the recommendation problem. It decomposes the user-item matrix into the product of two low dimension matrices. The original user-item matrix is decomposed into a user matrix and an item matrix in such a way that it approximates the original matrix. Approximate matrix of R can be defined as $R^{\wedge}=U^{T}\,P$ where $R^{\wedge}$ is an approximation matrix of user-item matrix R, $U^{T}$ is the transpose of the user matrix and P is the item matrix. Each column of U(P) holds the latent factors of the respective user(item) Fig2(see below left). $r^{\wedge}_{i,j}$ means to predict what user i might rate item j.

For a specific user, i and item, j, the rating are defined as $r^{\wedge}= u_i^{T} p_j$ where i (specific user) is represented by vector $u_i$ and j (specific item) is represented as vector $p_j$. To minimize the (sum square error) difference between original user-item rating and predicted user-item rating objective function can be written as:

[1] Codes are available at https://github.com/Karishma73/Research-Method/blob/main/RM_project.ipynb

$\min_{u,p} \Sigma(r_{i,j}-u_i^T p_j)^2 + \lambda(||u_i||^2+||p_j||^2)$ where $||u_i||$ and $||p_j||$ is distance from the origin, $r_{ij}$ original rating matrix.
Here u and p represent the vectors of U and P, $\lambda(||u_i||^2+||p_j||^2)$ is regularization parameter. λ is a free parameter. This approach struggle with the problem of negative feedback in user-item interaction.

## III. Problem Formulation

This section analyses the limitation of the inner product (matrix multiplication) of the user-item matrix and formulates the problem.

Let there are M users, U= $\{u_1, u_2, u_3, \ldots u_M\}$ and N items, P= $\{p_1, p_2, p_3, \ldots p_N\}$ where M denote the number of users and N denote the number of items. The user-item interaction matrix can be expressed as Fig2 (see below). R denotes the rating matrix and $R_{ij}$ is the rating of user i on item j. Let $U_i$ and $P_j$ represent the latent vector for user i and item j.

| | Item 1 | Item 2 | Item 3 | Item 4 | Item 5 |
|---|---|---|---|---|---|
| User 1 | 1 | 1 | 1 | 0 | 1 |
| User 2 | 0 | 1 | 1 | 0 | 0 |
| User 3 | 0 | 1 | 1 | 1 | 0 |
| User 4 | 1 | 0 | 1 | 1 | 1 |

*Figure 2. User item interaction.*

$R \in \mathbb{R}^{M\times N}$ and $y_{ij}$ is the predicted score for interaction between user i and item j.

$$y_{ij} = \begin{cases} 1, if\ interaction == True \\ 0, else \end{cases} \qquad (1)$$

Matrix generated by this prediction score will look like figure2 (see above).
If a user has interacted with an item it is considered as positive feedback. But in real cases, 1 does not explicitly mean that the user liked the item. Likewise, 0 does not imply users dislike the item. It is possible there was no interaction between that user and the item. Thus, it is a negative feedback problem. Assume S{i,j}, represent cosine similarity between user i and j.

S{i, j} =Cosine{i, j} assume following similarity were observed.
S{23}>S{12}>S{13} Here each vector P in latent space, represents a corresponding user from the user-item matrix. User-item matrix is similar to term-document matrix [13]. A single document is taken at a time in the pointwise loss function. Pair of documents is taken at a time in a pairwise loss function. To work this approach effectively, it is required for an item to be rated by a substantial number of users. For a new user, say $U_4$, latent space can be represented as shown below in Fig3. $P_4$ will give the wrong recommendation since it weighs more $P_2$ than $P_3$. So, for new users, this approach is not efficient unless the whole model is retrained. This is called "cold-start problem", the system cannot deal efficiently with new users [14]. This problem can be addressed by neural network-based prediction.

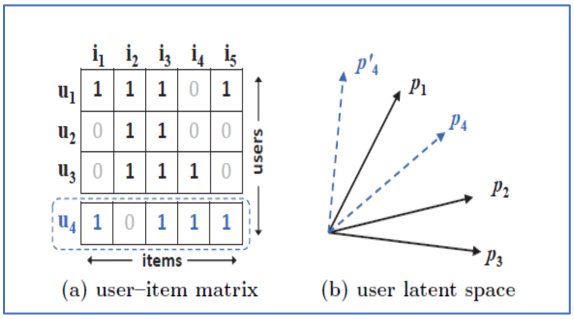



*Figure 3.Latent factor*

## IV. Neural Network-Based Solution

This section examines why and how a neural network-based approach enhances the user-item interaction function in recommender system.

### *Interaction Function*

User item interaction function can be expressed as: $\hat{y}_{ij}=f(i, j|\theta)$ where $\hat{y}_{ij}$ represent the predicted score between user i and item j. f is an interaction function and θ denotes model parameter. The parameters of the model are a component of users and item embedding matrix. If we have m users and n items, then the model will learn (mxk+nxk) parameters. Usually, a machine learning approach is used to learn this parameter, which optimizes the cost function. There are two types of cost function—pointwise loss and pairwise loss [15]. A neural network-based approach (implementing a neural network for learning interaction function) for user-item interaction gives a better result than machine learning approach [16]. The user-item interaction, $U_{MK} \odot P_{NK}$ ($\odot$ denotes element-wise product), where $U_{MK}$ is the users embedding vector and $P_{N,K}$ is the item embedding vector, is feed into multilayer fully connected neural network to learn the user-item interaction. Assume $p_i$ and $q_j$ represent the latent vector for user i and item j. Interaction function can be expressed as:

$$\hat{y}_{ij}=f(i,j|p_i,q_j)=p_i^T q_j=\sum_{k=1}^{k} p_{ik} q_{jk} \qquad (2)$$

Where k is the desired number of latent factors. User latent vectors and item latent vectors are feed into a neural network to make the prediction.

### *Neural Network*

Neural networks can be modified to add biases for users and items. Figure4 represents a multilayer approach to model user-item interaction function $y_{ij}$, where the output of the previous layer serves as the input of the next layer. For instance, the output of layer1 becomes the input for layer2. The bottom input layer Fig4 (see right) defines user i and item j. For instance,
The input layer is a feature vector say $X_0=U \odot P$

$X_1 = f\,(w_1\, X0 + b_1)$
$X_2 = f\,(w_2 X_1 + b_2)$
.
.
$X_n = f\,(w_3 X_{n-1} + b_n)$ where $f$ is activation function, w is the weight matrix(variable) and b is the biased matrix(variable). The binarized sparse representation of the user-item is converted into a dense vector by embedding layers. The obtained user(item) embedding is treated as a latent vector for the user(item). In the neural network layer, these latent vectors are fed in a multilayer perceptron to predict the score. Output layer will give a prediction score for $y^{\wedge}_{ij}$. The cost function of the model is to minimize the gap between target value($y_{ij}$) and scored value ($y^{\wedge}_{ij}$). The problem when $y_{ij}=1$ is, it depicts the user liked the item as mentioned above (1). With the help of probability distribution function in an output layer, this problem can be transferred into how likely i is relevant to j. A logistic function can bound $y^{\wedge}_{ij}$ in range of [0 1]. And the required equation can be expressed as:

$$P\,(y, y^{-}|U, P, \theta f) = \prod_{(i,\,j)\,\epsilon\, y} y^{\wedge}_{ij} \quad \prod_{(i,\,k)\,\epsilon\, y^{-}} (1 - y^{\wedge}_{jk}) \quad (3)$$

Where y denotes the set of observed interaction and y- denotes set of unobserved interaction. For the purpose of minimization

$$L = -\sum_{i,j\epsilon y} log y^{\wedge}_{i,j} - \sum_{i,j\epsilon \ddot{y}} \log\,(1 - y^{\wedge}_{i,j}) \qquad (4)$$

Equation 4 (cross-entropy loss) is a mathematically more convenient form of (3). Negative sign shows it is a minimization problem.
Output function (sigmoid function) can be written as:

$$y^{\wedge} = \frac{1}{1+e^{-(wX+b)}} \qquad (5)$$

Here w (means $w^T$). Backpropagation technique is used to learn these weights and biases.

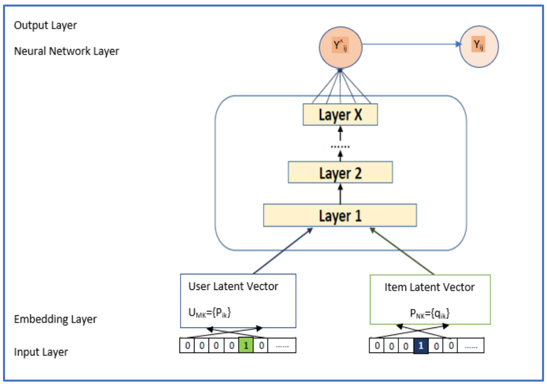



*Figure 4. Neural network-based approach*

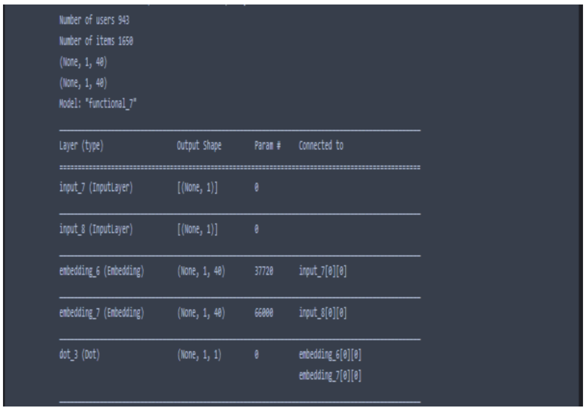
For the regression problem, square errors are generally used since the model predicts some value and check how far it is from the real value. For neural network classification is a probability distribution, error function requires to calculate the difference between true probability distribution and predicted probability distribution. Substituting neural network by inner product model will learn the parameters, which will result in better performance.

## V. Dataset and Model Design

This section implements the neural network-based design approach on MovieLens dataset with Python library Keras.[2] Keras is a neural network-based framework provided by python.

MovieLens dataset is used to design the neural network-based model. This dataset was collected for the GroupLens research project at the University of Minnesota. This dataset consists of 943 users and 1650 items (movies). At least 20 movies are rated by a single user. Embedding-6 represents the embedding layer for users and embedding-7 represents the embedding layers for items (see below) Fig 5. Dot_3 performs the dot operation between user embedding vector and item embedding vector, which will be the first layer of neural network (see above) Fig4.

[2] Codes are available at https://github.com/Karishma73/Research-Method/blob/main/RM_project.ipynb



```
Number of users 943
Number of items 1650
(None, 1, 40)
(None, 1, 40)
Model: "functional_7"
```

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_7 (InputLayer) | [(None, 1)] | 0 | |
| input_8 (InputLayer) | [(None, 1)] | 0 | |
| embedding_6 (Embedding) | (None, 1, 40) | 37720 | input_7[0][0] |
| embedding_7 (Embedding) | (None, 1, 40) | 66000 | input_8[0][0] |
| dot_3 (Dot) | (None, 1, 1) | 0 | embedding_6[0][0]<br>embedding_7[0][0] |

*Figure 5. Keras embedding layer*

## VI. CONCLUSION AND FUTURE WORK

Recommender systems are a leading technology used to provide recommendations to increase market value through the extraction of useful items from databases for clients. Such schemes allow consumers to get the items they would like to purchase from an organization. This system also supports the organisation by producing further profits. Neuronal network-based collaborative filtering outperforms most of state-of-the-art collaborative filtering techniques and deals with the cold start problem effectively [17]. Greater computational burden and proneness to overfitting are the problems with neural networks, which needs to be addressed carefully. The problem of overfitting was found with a large factor [18]. Hyperparameter and number of hidden layers seem to play an important role in model performance. Stochastic gradient descent optimization with learning rate appears to works better than Adam optimization in most of the cases. Overall, Neural network-based models outperform base-line methods [19]. Neural networks are "black box" in nature. It is difficult to find out what computations are going inside those perceptron layers. It develops models based on empirical analysis. Future research could be to determine exactly why neuronal network-based ML fits well with certain recommended systems and not with others. And implementation of the neural network-based latent factor model in real life.

# References


[1] P. e. al., "https://link.springer.com/," 2007. [Online]. Available: https://link.springer.com/. [Accessed 17 11 2020].

[2] 2. Victoria Bibaeva, "https://www.ieee.org/," 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8516989. [Accessed 06 12 2020].

[3] L. Zadeh, "https://www.ieee.org/," 1988. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/53. [Accessed 15 11 2020].

[4] M. S. Acharya, "https://www.ieee.org/," 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8862140. [Accessed 05 12 2020].

[5] X. Peng, "https://www.ieee.org/," 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5554740. [Accessed 05 12 2020].

[6] A. l. o. o. BrunoPerez, "https://www.sciencedirect.com/," 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0010482520303711. [Accessed 06 12 2020].

[7] Albrecht, "https://www.computer.org/csdl/home," 1993. [Online]. Available: https://www.computer.org/csdl/journal/ts/1983/06/01703110/13rRUNvya2z. [Accessed 05 12 2020].

[8] 2. Lifang chan, "https://link.springer.com/," 2019. [Online]. Available: https://link.springer.com/article/10.1007/s40747-019-00123-5. [Accessed 04 12 2020].

[9] G. C. V. A. L. A. R. A. S. R. C. C. D. S. M. J.L. Adeodatol, "https://www.sciencedirect.com/," 2004. [Online]. Available: https://www.researchgate.net/. [Accessed 05 12 2020].

[10] Y. a. Bao, "https://www.ieee.org/," 2012. [Online]. Available:

https://ieeexplore.ieee.org/document/6243081. [Accessed 05 12 2020].

[11] A. Dai, Y. Xia and Y. Gui, "https://www.computer.org/csdl/home," 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8322926. [Accessed 01 12 2020].

[12] F. Yang and Y. Lu, "https://www.ieee.org/," 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8622127. [Accessed 09 12 2020].

[13] B. Yi, X. Shen, H. Liu, Z. Zhang, W. Zhang, S. Liu and N. Xiong, "https://www.ieee.org/," 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8616805. [Accessed 29 11 2020].

[14] S. H. H. M. W. M. Wanas, "https://www.researchgate.net/," 2011. [Online]. Available: https://www.researchgate.net/publication/220183352_PUT-Tag_Personalized_User-Centric_Tag_Recommendation_for_Social_Bookmarking_Systems_Social_Network_Analysis_and_Mining_14_377-385. [Accessed 05 12 2020].

[15] Kostas, "https://www.sciencedirect.com/science/article/pii/S0957417413007240," 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417413007240. [Accessed 28 11 2020].

[16] 2. Saratchandra Indrakanti et.al, "Influence of Neighborhood on the Preference of an Item in eCommerce Search," in *IEEE https://ieeexplore.ieee.org/document/9006172*, Los Angeles, CA, USA, USA, 2019.

[17] 2. Aanchal Mongia et. al, "https://www.cornell.edu/," 2015. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1912/1912.04754.pdf. [Accessed 07 12 2020].

[18] K. Davagdorj, "www.researchgate.net," 2020. [Online]. Available: https://www.researchgate.net/publication/334371996_A_Collaborative_Filtering_Recommendation_System_for_Rating_Prediction. [Accessed 28 11 2020].

[19] 2. Joris Voerman et al, "https://www.cornell.edu/," [Online]. Available: https://arxiv.org/abs/2007.07547. [Accessed 03 12 2020].

[20] K. Makantasis, "https://ieeexplore.ieee.org/document/7326945," 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7326945. [Accessed 07 12 2020].