Decentralized Neural Network Based Collaborative Filtering For Privacy Concern Recommendation Systems

Kazi Omar Faruk , Anika Rahman , Sanjida Ali Shusmita , Md Sifat Ibn Awlad , Prasenjit Das , Md Humaion Kabir Mehedi , Shadab Iqbal , and Annajiat Alim Rasel

Department of Computer Science and Engineering
Brac University
66 Mohakhali, Dhaka - 1212, Bangladesh
{kazi.omar.faruk, anika.rahman, sanjida.ali.shusmita, md.sifat.ibn.awlad, prasenjit.das, md.sabbir.hossain1, humaion.kabir.mehedi}@g.bracu.ac.bd
annajiat@bracu.ac.bd

***Abstract*—The growing concern about the privacy of user data is inspiring the development of new privacy preserving machine learning approaches. Decentralized federated learning is such a method that can handle privacy concerns effectively. It consists of servers and clients. Here a machine learning model is distributed to a number of clients and the clients use their locally stored data to train the model and send the model to the server for aggregation. Here the client only shares the model parameters with the server. The server receives thousands of locally trained models from clients and performs aggregation to define a global model. Only the model parameters such as weights and biases are being shared between the server and the client. In this paper we have proposed a decentralized privacy preserving technique for neural collaborative filtering which is widely used in rating prediction for recommendation systems. Here each client receives an initial neural network based collaborative filtering model from the server and trains the model locally with its own data and only sends the model and parameters back to the server for aggregation. This method will eliminate privacy concerns in modern recommendation systems.**

***Index Terms*—Federated learning, neural network, federated collaborative filtering, neural collaborative filtering, federated neural collaborative filtering etc.**

I. Introduction

With the advancement of machine learning and deep learning algorithms more data driven applications have emerged. The increasing use of data driven applications leads to a variety of privacy issues. The concern for privacy is driving to the development of new privacy-preserving algorithms. To handle privacy concerns, federated learning techniques have been introduced in a variety of applications. Here the clients use their locally stored data to train a local model [1] after that model parameters are sent to the server, which aggregates them to define the master model, further the master model sent back to the clients. Only the model parameters are being shared. In contrast to the common practice of collecting, storing, and processing user data on a central server beyond user's control, user data in this situation never shared with the server, significantly improving user privacy [2]. In brief, the local dataset will be trained on the local device, with each client having their model to train their dataset. To define a global model, clients merely provide their local model and model parameters to the aggregate server. The privacy-preserving decentralized federated learning technique is utilized in a range of applications where privacy is a concern [2]. Recommendation systems are an example of such an application. Data is at the heart of recommender systems. A recommendation systems trained with a huge data can be more accurate in generating proper recommendation in general. Directly exchanging user data, however, is not desirable due to privacy and security concerns. Such privacy concerns are widespread in recommender systems [3]. Decentralized privacy-preserving federated learning approaches are now possible thanks to advances in modern computing hardware and network technologies. Many applications allow federated learning, one of which is recommendation systems. Neural CF [4] is a cutting-edge algorithm commonly utilized in recommendation systems. It works by anticipating how a user will rate unknown goods and then recommending products based on the highest projected ratings.

II. **Literature review**

R. Shokri et al. proposed a deep learning approach that protects data privacy in 2015. [1] Here, the author has proposed a practical solution that enables multiple parties to derive an accurate neural network based model for a specific goal without sharing their input datasets. They make use of the fact that modern deep learning optimization algorithms, such as stochastic gradient descent, may be parallelized and run asynchronously. Their strategy allows participants to train independently on their datasets while selectively sharing tiny portions of their model's key parameters during training. In 2018, K. Bonawitz published a paper titled "Practical Secure Aggregation for Privacy-Preserving Machine Learning" and proposed a secure federated machine learning model. The author devised a unique, communication-efficient, and failure-tolerant protocol for the secure aggregation of high-dimensional data in this paper. [2]. This protocol enables a server to securely compute the total of huge, user-held data vectors from mobile devices (i.e. without learning each user's specific input), and may be used, for example, in a federated learning environment to aggregate user-provided deep neural network model updates. M. Ammad-ud-din et al. [9] proposed a federated version of collaborative filtering where a model is distributed to clients and client trains that model with their local data. Then the local models transmitted to the server for aggregation to generate a master model which is then sent back to the clients. Although their method just requires each user to update the item profile, raw gradient data might result in data breaches [10]. Furthermore, before the update process begins, the system's Coordination Server waits for updates from all available participants. Linetal et al. are a group of researchers that have been working on a project [11] has looked towards federated MF systems with clear feedback. FedRec, their system, makes use of the parameter.$\rho \epsilon$ {0, 1, 2, 3} to select unrated objects at random, issue virtual ratings, obscure user activity, and enhance privacy Item profile changes are completed when all clients make updates, as their job is an extension of [9] Recently, Zheng et al. published an article [12] where the author proposed a neural auto-regressive approach for CF.

III. **Working Method of Traditional Recommendation Systems**

In real-world applications, the recommender system is critical. It has evolved into a critical tool for coping with information overload and has become a substantial source of revenue for many online businesses throughout the world. RecSys' recommendation performance improves with the amount of data it has access to. RecSys requires as much information from the user as possible to provide a fair suggestion. They gather personal information from users, such as behavioral data, contextual data, domain knowledge, item metadata, purchase history, user feedback etc. Some recommender systems use numerous data sources from other companies in the search of improved suggestions. All of this useful user information is centralized in each organization's database to facilitate various types of recommendation services. Traditional decision support systems, such as content-based [5] and collaborative filtering [6] systems make suggestions to users based on previous interactions. However, with RecSys, data concentration might pose major privacy and security problems. [3]. Recommenders, for example, may acquire personal information from users without their permission and sell it to third parties for a profit. Furthermore, data transfer may expose user privacy. In addition, various privacy and security-related laws have been passed in recent years committed to protecting privacy and security and pilot tests have been conducted to safeguard user data security and privacy. Obfuscation or cryptography techniques are commonly used in these methods. Some of these include sounds in various suggestion methods. Others encrypt the information before sending it to a recommender. Traditional data recommendations [7] rely mostly on user-item and profile attributes, whereas collaborative filtering suggests things to a user based on multiple user-to-user and item-to-item similarity metrics. Memory-based and model-based approaches are two types of collaborative filtering algorithms. There are also hybrid techniques that mix content-based and collaborative filtering. The model-based method employs statistical machine learning techniques, whereas the memory-based method employs closest neighbor classification methods. These systems, on the other hand, have their limitations, such as the cold start problem, which involves inconsequential proposals for a new user who has only performed a few or no system interactions, and the data sparsity problem, which is caused by a small number of actual full response per typical user. The most significant disadvantage of these approaches is that they require centralized user-item relevant information to function. This traditional strategy falls short when it comes to privacy concerns about sharing personal data. Federated learning-based recommendation systems can be a lifesaver when it comes to dealing with privacy concerns. In recommendation systems, neural clustering is a cutting-edge technique, and a federated version can easily deal with privacy problems.

IV. Traditional Neural Collaborative Filtering

In traditional neural networks based methods various inputs are embedded into a different dimension with unique weights and the embedded vectors are then feedforward into a deep neural network for rating prediction. Here, user item interactions are embedded into different dimensions. The embedding is simply a matrix dot product of one hot encoding of a user, item interactions and the embedding weights. This dot product between embeddings is also known as matrix factorization [8]. In matrix factorization the rating prediction of an unknown item is done by performing the dot product between the user and item's latent vectors. Neural network is introduced at the end of the embedding layers to make it nonlinear. For that some hidden dense layers are added. To provide non-linearity, different activation functions such as sigmoid, hyperbolic tangent, and rectified linear units are used in the hidden nodes. Further, the weights are learned by optimizing the network. Here, the training is done by comparing the given true rating with the predicted rating by the network. If there are differences between the true and predicted ratings then the difference is optimized. For the optimization different optimizers such as adam or sgd can be used along with a variety of loss functions such as mean squared error, mean absolute error etc.

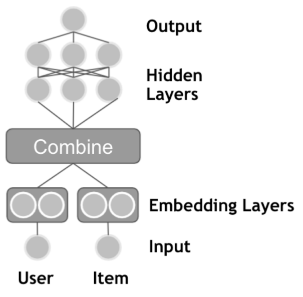


Fig. 1. An example neural network with embedding and fully connected layers

In the feed forwarded layer along with user id and product id additional user information such as age, gender, address, last interactions, payment information etc and item information such as category, brand, price, quantity sold etc can also be embedded and feed forwarded into the neural network.

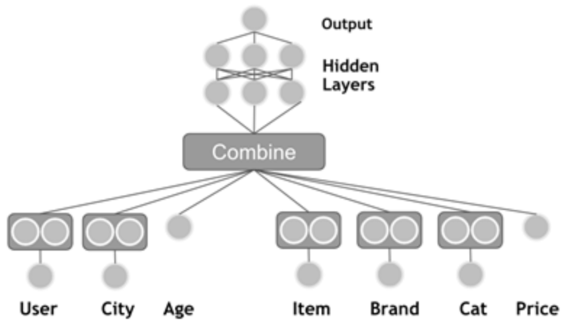


Fig. 2. Multiple categorical and continuous input neural network with embedding and fully connected layers

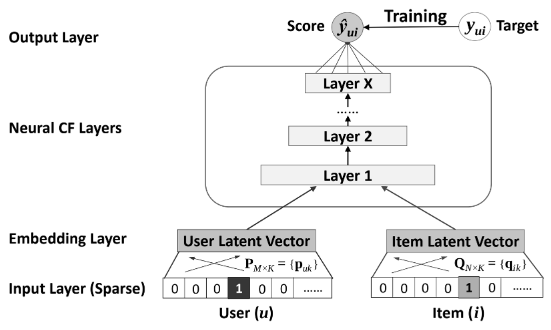


Fig. 3. Neural collaborative filtering model

V. Usefulness of federated learning in recommendation systems

Federated learning is a machine learning system that is still in its early stages of development. The cooperative training of machine learning models safeguards the privacy of users' data. Each party stores private user information locally. To interact with other parties, only intermediate outcomes like parameter changes are utilized. It enables many parties to share information without compromising user data security and privacy. Federated learning minimizes both privacy issues and costs, when compared to traditional data-centralized machine learning methodologies.

VI. Working Method of Federated Learning

Traditional machine learning models have both model and data at the same device to train which is known as decentralized machine learning which creates privacy related issues as discussed earlier. Federated Learning allows users sensitive data to be used to train models without having to send it to servers. It's a distributed training approach in which training and testing take place locally (at the edge), with just metadata transmitted to a central server, which merges multiple model updates (from many clients) into a new model.

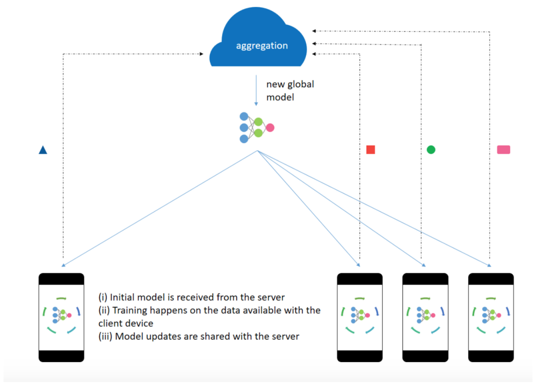

aggregation
new global model
(i) Initial model is received from the server
(ii) Training happens on the data available with the client device
(iii) Model updates are shared with the server

Fig. 4. Decentralized federated learning

Fig. 4 shows a traditional federated learning model. Here the client or edge devices receives an initial model from the server and then trains the model with the data available in it which is local training. Finally after local training it sends its locally trained model to the aggregation or global server which finally generates a combined model after receiving thousands of locally trained models from clients. In privacy constraints applications only the model parameters are shared between the global and local servers.

VII. **Federated neural collaborative filtering**

Federated neural collaborative filtering is a decentralized neural network based architecture. It consists of a number of clients and an aggregation server. Initially the server transmits an initial neural networks based model to the clients. The client receives the model and trains the model with their own data and sends it back to the aggregation server. The aggregation server then collects the locally trained model from a number of clients and performs aggregation. Here only the model parameters such as weights and biases are being shared between the clients and the server.

VIII. Experiment

In our experiment, we have used the MovieLens [13] ML100k dataset. MovieLens 100K movie ratings is a stable benchmark dataset. 100,000 ratings from 943 users on 1,682 movies. It was released in april 1998. Here, each user has rated at least 20 movies. The structure of the dataset is shown below,

We organized the dataset to make it usable for our federated neural collaborative filtering. We have 20% data to test our model. There are 3 steps in our experiment. Which are discussed below.

Initial model generation: The server will generate an initial model with 30% of the data choosing

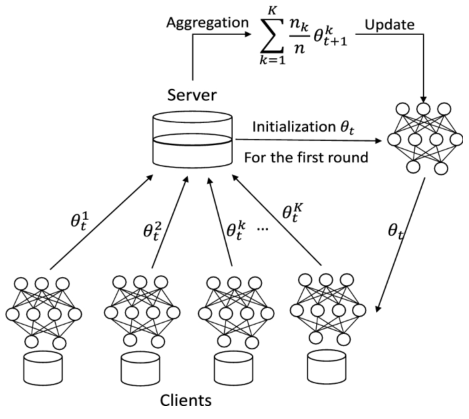
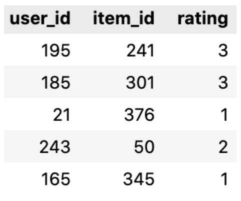


Fig. 5. Federated architecture for neural network



user_id	item_id	rating
195	241	3
185	301	3
21	376	1
243	50	2
165	345	1

Fig. 6. Overview of the dataset

randomly except the testing set. The following fig. 7 explains the initial model.

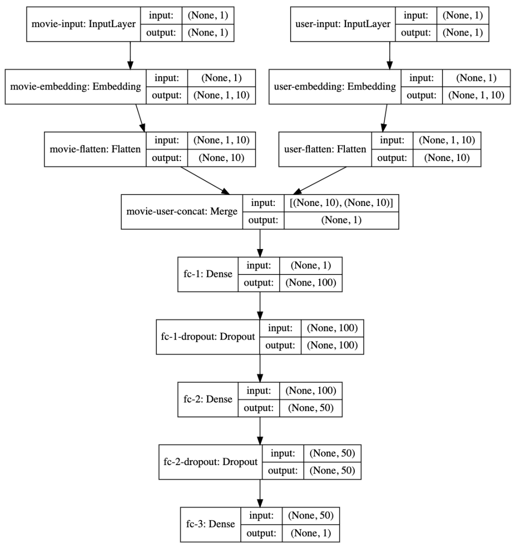

movie-input: InputLayer | input: (None, 1) | output: (None, 1)
user-input: InputLayer | input: (None, 1) | output: (None, 1)
movie-embedding: Embedding | input: (None, 1) | output: (None, 1, 10)
user-embedding: Embedding | input: (None, 1) | output: (None, 1, 10)
movie-flatten: Flatten | input: (None, 1, 10) | output: (None, 10)
user-flatten: Flatten | input: (None, 1, 10) | output: (None, 10)
movie-user-concat: Merge | input: [(None, 10), (None, 10)] | output: (None, 1)
fc-1: Dense | input: (None, 1) | output: (None, 100)
fc-1-dropout: Dropout | input: (None, 100) | output: (None, 100)
fc-2: Dense | input: (None, 100) | output: (None, 50)
fc-2-dropout: Dropout | input: (None, 50) | output: (None, 50)
fc-3: Dense | input: (None, 50) | output: (None, 1)

Fig. 7. Initial neural collaborative filtering model

Here, the input layer of the initial model takes the user id, item id and embedded these into a 10 dimensional embedding vector with the help of the embedding layer. Then the embeddings are flattened and concatenated and then feed forwarded this concatenated vector into the neural network model which consists of few dense and dropout layers. Here, the final dense layer provides a single value output which is the predicted rating. Here the hyper parameter we have used is number a epoch size of 6, adam as optimizer and mean absolute error as loss function. Further we evaluated this model with the help of mean absolute error, mean squared error and root mean squared error loss functions. Further this initial model will be sent to the clients to be trained with client data.

Clients local model training: Here the initial model will be sent to the clients for local training with clients data. In our experiment there are 943 users. Here each user will receive the initial model and then train that model with their own user-item interaction dataset with a number of epochs. In our experiment, we train the client's model to 6 epochs.

$$y = f(\sum_{i=0}^{n} W_i * Xi + B) \tag{1}$$

$$w_t = w_{t-1} - \eta \sum_{a \in S_t} \frac{n_k}{n}(w_t - w_{t,k}) \tag{2}$$

Global model generation: Finally the global aggregation server will receive all the locally trained models by the clients and will aggregate them to generate a global model. Here the aggregate server will receive all the weights and biases of the locally trained model and aggregate them by taking the average of these parameters.

Model evaluation: We evaluated our model in terms of mean absolute error, mean squared error, root mean squared error by comparing the prediction with the true rating. Also we evaluated the final global model using hit rate, cumulative hit rate, coverage and novelty. The hit rate is the count of hits divided by the test user count. It measures how often we are able to recommend a leave one out rating. Hit rate is higher is better. cumulative hit rate is calculated by ignoring the the predicted ratings lower than a certain threshold. Coverage determines what percentage of users have at least one good recommendation. In this case we considered a rating threshold of 4.0. Novelty is a measure of how popular the items system recommending or the ability of the recommendation systems to recommend popular items.

IX. Result Analysis

In this section, we will discuss and analysis the result of our model. First of all the, training and validation loss for 4 epochs of training for the initial model is shown the fig. 8. The performance of the initial model on the test set in terms of mean absolute error is 0.761, mean squared error is 1.021 and root mean squared error is 1.01.

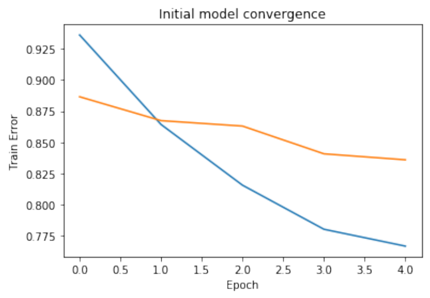
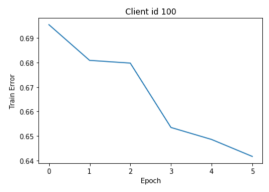
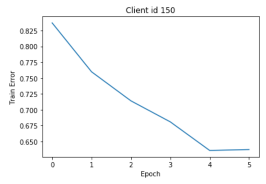


Fig. 8. Training loss convergence of initial model

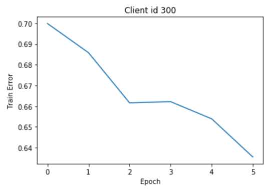
Result analysis of local model by client: This following figures shows the training loss convergence and the model evaluation of the randomly selected locally trained models.



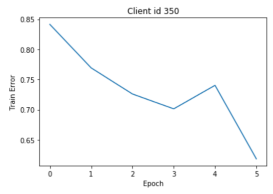
```
mae:  0.734682993721962
mse:  0.994757726026331
rmse:  0.9973754187999276
```



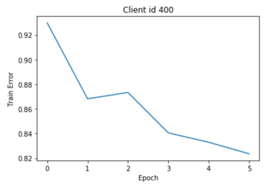
```
mae:  0.7525530477404594
mse:  1.0332228064015114
rmse:  1.0164756791982341
```



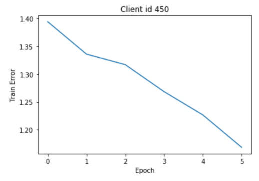
```
mae:  0.7648259829878807
mse:  1.072955140027623
rmse:  1.035835479228059
```



```
mae:  0.876840044927597
mse:  1.3166090092063778
rmse:  1.1474358409978214
```



```
mae:  0.7798394654035569
mse:  1.0511718315337062
rmse:  1.0252667123893695
```



```
mae:  0.8108160196661949
mse:  1.1185009884946424
rmse:  1.0575920709303008
```

Result analysis of global model by the aggregation server: We evaluated the final global model in terms of mean absolute error, mean squared error, root mean squared error global model using hit rate, cumulative hit rate, coverage and novelty which are shown in the table I.

TABLE I
RESULT EVALUATION OF THE GLOBAL MODEL

Evaluation metrics	**Result**
Mean absolute error	0.7741
Mean squared error	1.046
Root mean squared error	1.022
Hit rate	0.036
Cumulative hit rate	0.00954
Coverage	0.9968
Novelty	4.9957

X. Conclusion

In traditional machine learning technique central collection and storage of private data increases the risk of security and privacy. By using decentralized techniques the security and privacy concerns can be solved. To train a model in decentralized systems sharing model parameters can be useful. In this paper, we have introduced a federated version of neural collaborative filtering algorithm. It is a privacy preserving decentralized technique. With the help of our proposed model privacy concern in recommendation systems can be eliminated. Here multiple users could collaboratively train neural network based collaborative filtering model with their private data. We conclude that federated learning is a promising learning method because it no longer requires sharing of private data. Therefore, training your model with previously unavailable data will ultimately lead to smarter and more efficient systems. On the other hand, there are some open points that need to be carefully considered. It can be choosing the best number of participants for your training round and dealing with network failure events. There are some disadvantages of the proposed model. Here, if the server fails to receive the user model properly or if the user fails to train its local model, then it will not be able to define a global model properly. Proper and effective networking structure should be used to gain the maximum output of our proposed model.

References

[1] Reza Shokr, Vitaly Shmatikov: Privacy Preserving Deep Learning,shmat 2015

[2] Kallista Bonawitz: Practical Secure Aggregation for Privacy Preserving Machine Learning, 2018

[3] Naren Ramakrishnan, Benjamin James Keller, Batul J. Mirza, Ananth Y. Grama, George Karypis (2001). Privacy Risks in Recommender Systems. IEEE Internet Computing. 5. Piscataway, NJ: IEEE Educational Activities Department. pp. 54–62.

[4] Xiangnan He, Lizi Liao, Hanwang Zhang, Xia Hu, Liqiang Nie, Tat-Seng Chua ”Neural Collaborative Filtering”. Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee: 173–182. arXiv:1708.05031. doi:10.1145/3038912.3052569.

[5] Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011, pp. 1-35

[6] Aggarwal, Charu C. (2016). Recommender Systems: The Textbook. Springer. ISBN 9783319296579.

[7] Donghui Wang, Yanchun Liang, Dong Xu, Xiaoyue Feng,Renchu Guan. ”A content-based recommender system for computer science publications”. Knowledge-Based Systems. 157: 1–9.

[8] Yehuda Koren, Chris Volinsky, Robert Bell. ”Matrix Factorization Techniques for Recommender Systems”. Computer. 42 (8): 30–37.

[9] Muhammad Ammad-ud-din, Elena Ivannikova, Suleiman A. Khan, Qiang Fu. Federated Collaborative Filtering for Privacy-Preserving Personalized Recommendation System.

[10] Di Chai, Leye Wang, Kai Chen, Qiang Yang. Secure federated matrix factorization. IEEE Intelligent Systems.

[11] Guanyu Lin, Feng Liang, Weike Pan. FedRec: Federated Recommendation with Explicit Feedback. IEEE Intelligent Systems.

[12] Yin Zheng, Bangsheng Tang, Hanning Zhou, Wenkui Ding : A neural autoregressive approach to collaborative filtering. In ICML, pages 764–773, 2016.

[13] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. https://grouplens.org/datasets/movielens/100k/