

Article

LSH Models in Federated Recommendation

Huijun Dai ^{1,2,*} , Min Zhu ^{1,3} and Xiaolin Gui ^{1,2}

¹ School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China; minzhu@xjtus.cn (M.Z.); xlgui@xjtu.edu.cn (X.G.)

² Shanxi Province Key Laboratory of Computer Network, Xi'an Jiaotong University, Xi'an 710049, China

³ Suzhou Academy, Xi'an Jiaotong University, Suzhou 215123, China

* Correspondence: daihuijun@xjtu.edu.cn

Abstract: Given the challenges in recommendation effectiveness, communication costs, and privacy issues associated with federated learning, the current algorithm amalgamates locality sensitive hash (LSH) with three federated recommendation models: Generalized Matrix Factorization, Multilayer Perceptions, and Neural Matrix Factorization. First, the participation weights of the model are determined based on the participation degree of the federated learning clients to improve the efficiency of joint learning. Second, the local parameters of the federated aggregation model are divided into two groups to protect user embedding. Finally, rapid mapping and similarity retrieval of the upload parameters are performed using LSH to protect user privacy and shorten training time. We conducted experiments to compare the performance differences between LSH-based and Laplace noise-based differential privacy methods in terms of recommendation effectiveness, communication costs, and privacy preservation. Experimental results demonstrate that LSH models achieved a favorable balance between recommendation effectiveness and privacy protection, with improved time performance.

Keywords: federated learning; locality sensitive hash; neural collaborative filtering; differential privacy; model weight; local parameter



Citation: Dai, H.; Zhu, M.; Gui, X. LSH Models in Federated Recommendation. *Appl. Sci.* **2024**, *14*, 4423. <https://doi.org/10.3390/app14114423>

Academic Editor: Giacomo Fiumara

Received: 30 January 2024

Revised: 10 April 2024

Accepted: 11 May 2024

Published: 23 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Deep learning-based recommendation systems usually take data related to various kinds of users or items as model input, represent the hidden mapping relationship between the users and items, and generate the recommendation results. Three models named Generalized Matrix Factorization (GMF), Multilayer Perceptron (MLP), and Neural Matrix Factorization (NeuMF) are proposed [1]. One of the most common forms for constructing deep learning recommendation is MLP [2]. By increasing the number of layers and complexity of the deep neural network structure, Autoencoder is proposed [3]. Neural collaborative filtering (NCF) [4] has evolved through a rich feature crossover mode. In NCF, the simple inner product operations in the matrix decomposition have been dropped, more nonlinear features are introduced, which improves the learning ability of the sparse features. In addition, the deep interest network [5], as well as the deep interest evolution network [6], and the dynamic path selection multi-interest network [7] using the capsule network to extract the diverse interests of users all represent the current optimal deep learning recommendation model.

Federated learning is a distributed machine learning environment. Under the coordination of central servers, multiple entities (clients) collaborate to complete the learning problems. Each client's raw data are stored locally, not exchanged or transmitted, and only in the aggregated update model. With the public attention on personal data privacy, it becomes increasingly important how the recommendation system to seek the optimal solution between personalization and privacy protection and data perturbation technology have been widely used in recommendation. Dwork [8] has proposed differential privacy

(DP, including local DP and global DP) by adding Laplacian noise, which is used for the collaborative filtering algorithm. A localized DP-based perturbations on federated recommendation is proposed recently [9]. The traditional recommendation system is designed to protect the privacy of individual users by preventing the leakage of their original data and by encrypting the transmission of recommended original inputs. The federated recommendation system employs encryption of the gradient update of the recommendation model to safeguard privacy [10]. It also avoids the significant computational overhead of directly training the global model on the original data cipheric domain. Furthermore, it effectively reduces the numerous privacy risks associated with data aggregation in traditional machine learning sources. The current federal recommendation system is still afflicted by two significant shortcomings: firstly, the addition of LDP has resulted in a reduction in the accuracy of recommendations, and secondly, the incorporation of Laplace noise has led to an increase in communication overhead [10].

Locality sensitive hash (LSH) is an approximate nearest neighbor (ANN) search and mapping technique. The locality hashing technique is employed to protect the client-uploaded model parameters throughout the training process of federated learning. The LSH primarily safeguards the local parameters of the model, which comprise sensitive information from the user. By hashing these local parameters, namely converting them to hash values, it is possible to ensure that when uploaded to the central server, even if the parameters are intercepted or leaked, the user-specific information cannot be directly obtained due to the unidirectional nature of the hash function. Consequently, LSH can safeguard user privacy and model security, even when federated learning clients are not deemed trustworthy or at risk.

This paper presents three joint locality sensitive hashing (LSH) models combined with recommendation systems: GMF, MLP, and NeuMF. The LSH-enhanced federated learning model is the result of this combination. Firstly, the participation weight of the model is determined according to the participation degree of joint learning customers, with the objective of improving the efficiency of joint learning. Secondly, the local parameters of the joint aggregation model are divided into two groups in order to protect the user embedding. Finally, rapid mapping and similarity retrieval of the upload parameters are performed using LSH in order to protect user privacy and to reduce the training time.

2. Related Works

2.1. Neural Cooperative Filtering and Differential Privacy

Deep neural networks are widely used to represent the relationship between the mapping functions and the hidden factor of users and items. Three models named Generalized Matrix Factorization (GMF), Multilayer Perceptron (MLP), and Neural Matrix Factorization (NeuMF) are proposed, in which, GMF can learn the linear characteristics in the recommendation model, while MLP excels at getting more detailed and nonlinear characteristics. Neural network-based collaborative filtering (NCF) utilizes implicit feedback, meaning the historical interaction of users with the items. This approach avoids revealing user rating, but the historical interaction records of users are necessary during training. The deep interest network (DIN) [5] is evolved by importing the attention layer between the embedded layer and the perceptron layer. The deep interest evolution network (DIEN) [6] incorporates sequential models to simulate the evolution of users' preferences. In contrast, capsule networks are explored to extract users' diverse interests, and label-based attention mechanisms are also proposed in multi-interest network with dynamic routing (MIND) [7].

The parameters of the federated recommender system models are encrypted by privacy protection methods during the gradient updates [10]. It is unnecessary to exchange local parameters of different recommendation services respectively and to train the global model directly in plain text, which effectively reduces the privacy risks during the aggregation process. During the aggregation of federated learning, users' information can also be restored based on model parameters or gradient information. Orekondy [10] and Wang proposed that simple federated learning without privacy-preserving mechanisms still

suffers from gradient information attacks, which can leak private information. To avoid this kind of attack, differential privacy (DP) is combined into the model via adding noise [11]. The DPGAN framework proposed by Xu [12] makes use of DP to invalidate GAN-based attacks when inferring the training data of users in deep learning networks. McSherry [13] proposed improved collaborative filtering algorithms applying DP and obtained a blinded item–item covariance matrix by adding Laplace noise to count and sum up the user rating data. And K-Nearest Neighbor (KNN) and Singular Value Decomposition (SVD) algorithms have been implemented based on the above scheme. The DPFedAvgGAN [14] framework also has been proposed in federated learning. The input layer data of the model are divided into two parts, the matrix factorization (MF) and the multi-layer perceptron (MLP). In the MF part, the input users' data and items' data are mapped into the user hidden vectors and item hidden vectors. In the MLP part, the input user data and item data also do the same mapping. The MF part and the MLP parts share the same embedding layer and learn the embedding layers separately with two different models providing better flexibility.

2.2. Neural Cooperative Filtering Framework and Process

The overall framework of the NCF is shown in Figure 1; the first layer of the model accepts two inputs, namely, the user input layer and the item input layer.

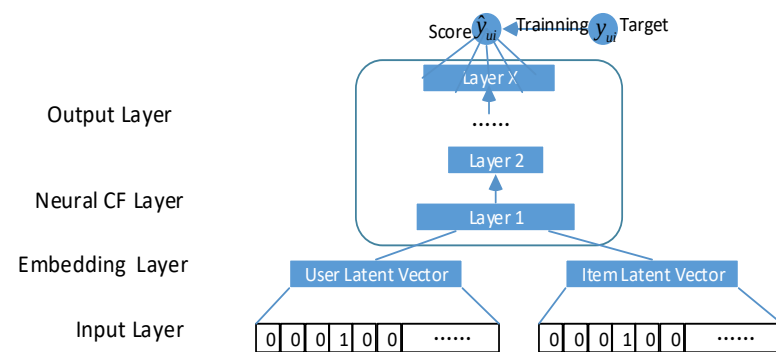


Figure 1. NCF network framework.

The input layers require only binary labelling (0 or 1) to reflect the historical interaction of users and items. Therefore, one-hot encoding is sufficient. The user and item embedding layer is the second layer.

The input layer is mapped to the embedding layer to generate hidden vectors for both the user and item. These vectors are then used as inputs to the NCF network, resulting in a prediction score. The model recommendation effect is determined by the dimension and number of the network layers. The NCF model can be formulated as follows:

$$f(P^T v_u^U, Q^T v_i^I) = \varphi_{out}(\varphi_X(\dots \varphi_2(\varphi_1(P^T v_u^U, Q^T v_i^I)) \dots)) \quad (1)$$

where φ_i represents the i -th layer in the NCF, with a total layers of X ; φ_{out} represents the output layer. P and Q are the user embedding layer parameters and the item embedding layer parameters, respectively. v_u^U and v_i^I represent the user input vectors and the item input vectors after one-hot encoding, respectively. The model uses cross-entropy as a loss function, as follows:

$$C = - \sum_{(u,i) \in y \cup y^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui}) \quad (2)$$

where $\hat{y}_{ui}$ represents the historical interaction results of user u for item i ; if user u has a scoring record for item i in the dataset, $\hat{y}_{ui}$ is equal to 1, otherwise it is 0. The loss function can obtain the optimal results through random gradient descent or the Adam algorithm. In the training process, a positive sample is created using the interaction record between the

user and the item, labeled as 1. If there is no historical interaction between the user and the item, it is considered a negative sample and labeled as 0.

Horizontal federated learning is used in NCF to aggregate user privacy data on distributed sides and integrate the model on the server side. A round of federated learning training includes four steps: (1) model initialization, where a model and various parameters are initialized by the central server and sent to each client; (2) local model training, where all online clients train the model delivered by the server on their own local dataset; (3) model aggregation, where after each client has finished their training and uploading, the central server aggregates the model based on the level of client involvement; (4) model update, where the central server distributes the aggregated model to each client, updates the client's model, and then repeats step 2. The formal performance is shown in Equation (3).

$$w^* = \operatorname{argmin}_w \sum_{i=1}^N p_i F_i(w, D_i) \quad (3)$$

where $F_i(\cdot)$ is the local loss function for the i -th client; usually, $F_i(\cdot)$ is set by local experience; p_i is the aggregation weight of the model for the i -th client.

2.3. Introduction of Locality Sensitive Hashing (LSH)

LSH is a randomization algorithm for the nearest neighbour problem, which is significantly faster than the other existing methods, especially in high-dimensional space. It uses the hash method to hash the data from the original space into a new space, so that similar data in the original space have a large probability in the new space, while dissimilar data in the original space have a small probability of being similar in the new space.

The basic idea of LSH is the following:

1. To randomly select the hash function from the family of LSH functions and hash each data point into the hash table;
2. When finding the nearest neighbor, LSH only scans the points with the same hash index as the query point.

Definition and Algorithmic Process are as follows:

Definition 1. For $r_1 \leq r_2$ and $p_2 \leq p_1$, a hash family H is (r_1, r_2, p_1, p_2) -sensitive if for all $x, y \in S^{d-1}$.

$$\text{if } \|x - y\|_2 \leq r_1, \text{ then } \Pr_H[h(x) = h(y)] \geq p_1$$

$$\text{if } \|x - y\|_2 \geq r_2, \text{ then } \Pr_H[h(x) = h(y)] \leq p_2$$

If the hash collision probability of a point that is close is greater than that of a point that is far away, it is a locality sensitive hash, that is, the hash function is sensitive to the distance between data points, locally close points can be hashed into a bucket as much as possible. According to the above definition, two points are considered close if the distance is not greater than r_1 ; if the minimum distance is $r_2 = r_1 * c$ ($c > 1$), c is the distance coefficient between the far and near points.

The quality of the hash function is defined by two key parameters: p_1 represents the hash conflict probability of proximity points, while p_2 represents the hash conflict probability of distance points. The difference between these two probabilities determines the sensitivity of the hash function to distance in changes and can usually be expressed as a function of the distance gap.

3. NCF Federated Recommendation Based on DP Results

3.1. NCF Recommendations Based on Federated Learning

The NCF recommendation system based on federated learning can pre-train GMF and MLP to obtain the NeuMF model, that is, the trained GMF and MLP can be applied to initialize and train NeuMF. Firstly, GMF and MLP are randomly initialized. Subsequently, the

two models are trained by using idle online clients to acquire the client model parameters. Lastly, the models are merged and submitted to the central server; the connection between the two parts can be established using the following formula:

$$h \leftarrow \begin{bmatrix} \alpha h^{GMF} \\ (1 - \alpha) h^{MLP} \end{bmatrix} \quad (4)$$

where h represents the parameters of the NCF, h^{GMF} represents the parameters of the GMF part, and h^{MLP} represents the parameters of the MLP part. α is the proportion of the parameters of the GMF part in the final NeuMF model.

3.2. Model Weight Setting Based on Participation

In the parameter aggregation of the federated recommendation system, the weight of the model parameters uploaded by all clients is equal. However the quality of the uploaded models varies due to the randomness of the federated learning training parameters. The quality of client-side data, the level of participation in federated learning, user hardware equipment, and other factors can impact the quality of the aggregated model. To enhance the recommendation effect, it becomes necessary to elevate the weight of the optimal model and diminish the weight of other models. In each round of federated learning, a portion of users is randomly selected as samples to participate in the training. Due to the random nature of the training process, some clients have participated in multiple rounds of training while others have only recently started. Consequently, models with low engagement are likely to underfit their own data. Therefore, during the aggregation stage of federated learning, weight parameters can be assigned to the model uploaded by the client based on their level of training participation. The core idea of the model weight setting based on participation is to adjust the weight w_i of each client by recording the participation number x_i . Then, the function $w_i = f(x_i)$ is explored to determine the weight value. The specific process is as follows:

- The recommendation system server initializes the participation number of the client side;
- Calculates the aggregate weight $w_i = f(x_i)$ of each client and saves it on the server side;
- The participation number of all clients in this round is increased once, $x'_i = x_i + 1$.
- The recommendation system recalculates the aggregation weights $w'_i = f(x'_i)$ according to the participation number and updates the weights on the server side.

The choice of function determines the convergence effect of this federated learning algorithm, and a better function is conducive to the training of the federated recommendation model; conversely, a poorer function may hinder the training process. In this paper, a simple liner function is conducive to speeding up the training of the federated recommendation model; the specific formula is as follows:

$$w_i = f(x_i) = \begin{cases} x_i & x_i < T \\ T & x_i \geq T \end{cases} \quad (5)$$

where T is the threshold of the weight to prevent excessive growth of a user's weight. According to Formula (5), the clients' aggregation weights are linearly positively correlated with the number of users participating in federated training. In other words, the more times a user participates, the greater the weight. Once the weight reaches the threshold T , it is set to T and remains constant.

3.3. Local Parameter NCF Recommendation

In order to prevent getting the user's rating or other data from the historical interaction by back propagation attacks, a local parameter training method is proposed for NCF federated recommendation. This is as shown in Equation (6):

$$\min_{x \in R^d} F(x) = E_{i \sim p}[f_i(x)] \quad (6)$$

where $f_i(x)$ is the local optimization target for user i ; x is the vector of model parameters with dimension d ; and p is the probability distribution for the clients. Some parameters in this model are stored locally, not transmitted, and do not participate in the aggregation process of federated learning. These parameters are referred to as the local parameters of the federated learning. In the NCF model, the embedding layer maps the user ID and the item ID to the hidden layer using one-hot encoding. Therefore, its parameters can be considered as local parameters that do not participate in the aggregation and the issue process of the federated learning. Each user's or item's ID is independent during both the training and prediction processes. Only the correct user hidden feature vector is required as input. As a result, the user embedding layer can serve as a local model parameter, saved and updated locally on each client. The training process is shown in Figure 2 below:

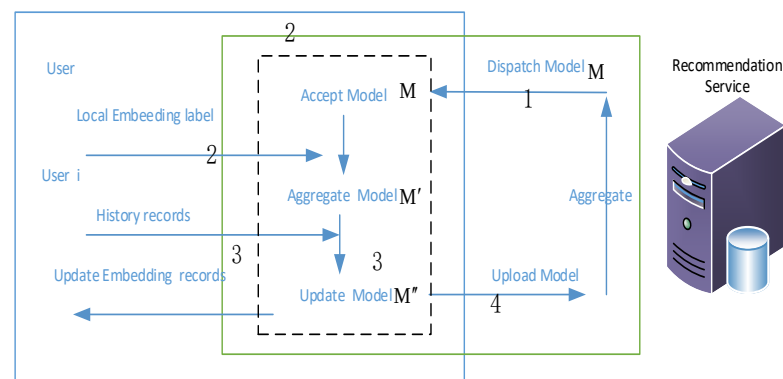


Figure 2. Local parameter NCF recommendation model.

As depicted in Figure 3, the process unfolds as follows: firstly, the recommendation system server dispatches the initialized NeuMF model M , excluding the embedding layer, to each online client. Secondly, each client initializes its own embedding layer and combines model M into a complete recommendation model M' . Thirdly, the newly trained model M'' is obtained by training local data on the client side based on M' , and the embedding layer parameters of M' will be replaced by those of the model M'' . Fourthly, the model M'' excluding the user embedding layer is uploaded to the recommendation system server. Finally, the recommendation system server aggregates the uploaded parameters according to each client's weight.

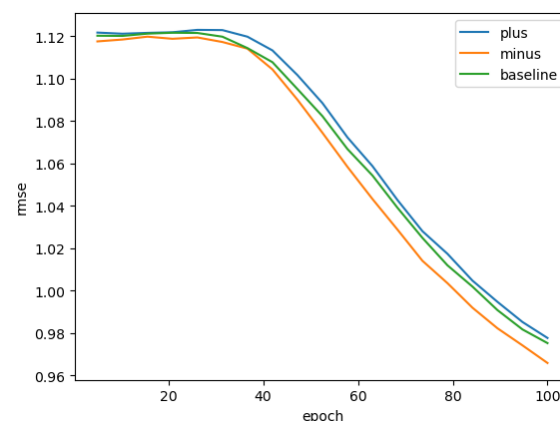


Figure 3. Different weight setting in federated recommendation.

In this local parameter NCF federated recommendation, the local model M' needs to be updated, and the model M issued by the recommendation system server does not contain the user embedding layer. After updating and merging with the embedding layer, the model M'

has been evolved into the model M'' . The local parameter NCF federated recommendation reduces the transmission or communication cost, but improves the data privacy.

3.4. NCF Federated Recommendation with DP

To further prevent the attacks on the uploaded model, the algorithm uses the Laplace mechanism in DP to safeguard the model's parameters during the federated learning. In the Laplace mechanism, the noise size is $Laplace(f/\epsilon)$, where f is the sensitivity, and ϵ can gauge the level of privacy protection. The Laplace machine satisfies ϵ -differential privacy; with smaller ϵ , the greater the noise and the higher the security of the data. Given that the sensitivity impacts the effect of the parameters ϵ in Laplace mechanism, the model incorporates two schemes, namely, overall sensitivity and hierarchical sensitivity. There is no requirement to safeguard the embedding layer and local parameters in the NCF recommendation within the privacy schemes for these two sensitivities, as these parameters are not uploaded.

Stratified Sensitivity Scheme

The models uploaded contain different datasets according to different layers and calculate f in Laplace noise respectively.

$$f_i = \max(grad_i) - \min(grad_i) \quad (7)$$

where f_i and $grad_i$ correspond to the i -th layer of the model, adding noise of size $Laplace(f_i/\epsilon)$ to the i -th layer. In contrast to the overall sensitivity scheme, the stratified sensitivity scheme takes into account the distinction of each layer's parameters, resulting in the addition of relatively less noise to individual layers. This scheme can achieve equivalent privacy protection, and introducing varying levels of noise to different layers provides greater flexibility and enhances training effectiveness.

3.5. LSH Models in Federated Recommendation

To improve the efficiency and level of privacy protection, we recommend combining NCF with LSH. The NCF can be used to make personalized recommendations. Meanwhile, LSH can be used to accelerate the approximate nearest neighbor search, thereby reducing the computational complexity of the recommendation system and improving the efficiency of the recommendation.

The specific implementation method is as follows:

The NCF model is used to represent the data: first, the data are analyzed using the neural network model. These represent the complex relationships therein, thereby improving the accuracy of the recommendations.

Accelerate nearest neighbor search with LSH: LSH hash the resulting data processed, mapping similar users and items to the same hash bucket. In the recommendation process, only the users or items in the same hash bucket need to be considered, thus reducing the size of the candidate set and improving the recommendation efficiency.

Protecting user privacy: In federated learning, use local hash technology to protect the model parameters uploaded by clients. By hashing the local parameters of the model, we ensure that even if the parameters are intercepted or leaked, the user's specific information cannot be directly obtained, so as to protect the user's privacy.

Comparative differential privacy techniques: Local hash techniques may be more suitable to solve communication costs and privacy issues than the differential privacy technologies using Laplace noise. Partial hashing technology can reduce traffic, reduce communication costs, and provide stronger privacy protection.

By combining NCF with LSH and adopting local hashing technology to protect user privacy, we can improve the efficiency and privacy protection level of the recommendation system at the same time, so as to provide a better recommendation experience for users' hash family of functions.

4. Comparative Analysis of Experiment

This paper utilizes the MovieLens 1M dataset, compiled by the GroupLens team at the University of Minnesota, to gather users' movie ratings without collecting personal information. The MovieLens 1M dataset consists of rating data from one million users for movies, with each user providing ratings for a minimum of 20 movies on a scale of 1 to 5. The data processing procedure involves the following steps: firstly, 90% of the data are randomly allocated as the training set, while the remaining 10% are assigned to the test set. Secondly, the NCF model handles implicit interactions and does not require explicit ratings. Therefore, all ratings in the MovieLens dataset are standardized to 1 as labels to indicate user interaction with or rating of those movies. Moreover, for the training set, unrated data points are chosen, assuming that the user has not seen the movie, and are assigned a value of 0. These two sets of data are then combined to form the training set, while the remaining unrated data are used to create the test set.

4.1. Evaluation Metrics

For the traditional Matrix Factorization (MF) model, the Root Mean Square Error (RMSE) is used as the evaluation index, and the smaller the RMSE, the more accurate the prediction result. The RMSE is as follows:

$$RMSE = \frac{\sqrt{\sum_R (r_{ui} - \tilde{r}_{ui})^2}}{|R|} \quad (8)$$

where $|R|$ denotes the number of valid ratings in the dataset.

For the NCF model, the prediction output is a list of recommended rankings. This paper employs Leave-One-Out Cross-Validation (LOOCV) as the evaluation metric. The rankings are determined based on the ratings, and these rankings are then used to calculate the Hit Rate (HR) for positive samples. For instance, with a Top-K value of 10, the top ten recommendations are examined to identify the presence of positive samples. If a positive sample is found, it is considered a hit; otherwise, it is classified as a miss. A higher HR value indicates a more accurate recommendation performance. Additionally, based on the ranking outcomes, the Normalized Discounted Cumulative Gain (NDCG) is computed to assess the predictive effectiveness. A higher NDCG value signifies an enhanced recommendation performance. The NDCG is as follows:

$$NDCG(k) = Z_k \sum_{j=1}^k \frac{2^{r(j)} - 1}{\log_2(j + 1)} \quad (9)$$

where k represents the number of top k ; Z_k is the normalization parameter.

The F_1 score is the harmonic average of *Precision* and *Recall*, and it is particularly useful in binary classification problems. When the positive and negative sample categories in the dataset are not balanced, the F_1 score can evaluate the model performance more comprehensively, because it considers the effects of both false positive (False Positives, *FP*) and false negative (False Negatives, *FN*). The F_1 score is calculated as follows:

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (10)$$

where *Precision* represents the proportion of all samples predicted to be positive, using the formula $Precision = \frac{TP}{TP+FP}$. *Recall* represents the proportion of all samples that are actually positive, according to the formula $Recall = \frac{TP}{TP+FN}$. *TP* is the number of true cases, the number the model correctly predicts as a positive class.

F_1 scores range from 0 to 1, with higher values indicating better performance of the model. When the F_1 score approaches 1, the representation model achieves a good balance between *Precision* and *Recall*.

4.2. The Influence of Federated Learning on Traditional Matrix Decomposition

Federated learning was separately incorporated into the traditional matrix decomposition model, MLP, GMF, and NeuMF, and the hidden vector dimension was set to 8. The federated matrix factorization model and the traditional models are compared. Each round of federated learning involved five participating samples, and the RMSE was used to evaluate the recommendation performance. The experimental results are shown in Table 1 below:

Table 1. The impact of federated learning on different models.

Model	RMSE of Model	
	No Federated Learning	Federated Learning
MF	0.8956	0.9433
GMF	0.7744	0.8773
MLP	0.8333	0.9014
NeuMF	0.8011	0.8457

The results show that the RMSE of matrix factorization (MF) and other models increases after using federated learning, which implies a decrease in recommendation performance. Essentially, all the models sacrifice the validity of the recommendation in the context of federated learning, highlighting the trade-off between data privacy and accurate recommendation. Maintaining precise recommendations while protecting data privacy becomes very difficult. At the same time, the RMSE values of mlp, gmf, and neuMF are less than the matrix factorization model, indicating that their recommended effect is better than that of matrix factorization.

4.3. NCF Federated Recommendation

The NCF federated recommendation framework comprises three models: GMF, MLP, and NeuMF. The ratio of negative samples to positive samples in the training sets of these models is 4:1, with K set to 10 for top-K evaluation. In the GMF model, user and item embedding layers are mapped to hidden vectors of length 8 using one-hot encoding. For the MLP model, the hidden vector length is also eight, incorporating a hidden perceptron layer with a dimension of 10. In the NeuMF model, embedding layers of length 8 are established in both the MLP and GMF parts. Additionally, a hidden layer is introduced in the GMF component. NeuMF integrates these two parts and employs federated learning. Each round of federated learning involves five participating users, and the experimental results are detailed in Table 2:

Table 2. Three models in NCF federated recommendation.

Model	No Federated Learning		Federated Learning	
	HR	NDCG	HR	NDCG
GMF	0.4788	0.2827	0.4331	0.2216
MLP	0.4093	0.2414	0.4029	0.2408
NeuMF	0.4887	0.3030	0.4411	0.2529

The results indicate that while federated learning offers some level of data privacy protection, it significantly diminishes the recommendation accuracy of the GMF and NeuMF models. On the other hand, the recommendation accuracy of the MLP model with a single hidden layer remains relatively stable, demonstrates minimal impact from federated learning.

4.4. The Choice of Sample Number

In the context of the NCF federated recommendation framework, the choice of sample number plays a crucial role in the speed of federated learning. The experiment results showing in Table 3 involves traditional MF federated recommendation with varying sample

numbers randomly selected for each round of federated learning. As the number of samples per round increases, the duration of each round also lengthens, leading to potential delays in client model uploads in real-world scenarios. To address this, the product of the sample number per round and the number of federated learning training rounds are kept constant to maintain consistency in the total number of samples used for training.

Table 3. Traditional MF federated recommendation with varying sample numbers.

Sample Number	Training Rounds	RMSE
5	1000	0.9752
10	500	1.0912
20	250	1.1183
50	100	1.1196
100	50	1.1175

The experimental findings demonstrate that, with a consistent total dataset size for training, lower sample numbers per round correspond to reduced RMSE values and improved recommendation accuracy. While fewer samples enhance accuracy, they also increase communication frequency between clients and the server in practical applications. Therefore, in a federated recommendation model, striking a balance between communication costs and recommendation effectiveness is essential. Opting for a smaller sample size in a conducive communication environment can enhance recommendation accuracy.

4.5. Weight Setting Based on Participation

This experiment validates the model weighting algorithm based on participation. In Figure 3, the ablation experiment utilized a traditional MF model with a hidden vector dimension of eight and five samples per round. For the selection of $f(x_i)$, when Equation (5) is chosen, the federated recommendation based on participation performs worse than that of the random parameter, resulting in a higher RMSE after 100 iterations. This indicates a decrease in recommendation accuracy, suggesting that the weight setting function hampers federated learning. On the other hand, when Formula (6) is utilized, the federated recommendation based on participation improves, achieving the lowest RMSE after 100 iterations. In conclusion, Formula (6) represents a weight setting algorithm that enhances federated recommendation.

4.6. Local Parameter NCF Recommendation

In this experiment, GMF, MLP, and NeuMF are individually trained using local parameters in federated learning. The ratio of negative to positive training samples for all three models is set at 4:1, with a value of K set to 10. The model incorporates a hidden vector layer of length 8, while the MLP component includes a hidden layer with a dimensionality of 10. Each round involves the participation of five entities. The experiment compares the HR and NDCG metrics across different models, and the results are presented in Table 4.

Table 4. Three models in NCF federated recommendation using local parameter.

Model	No Federated Learning		Federated Learning	
	HR	NDCG	HR	NDCG
GMF	0.4777	0.2816	0.4384	0.2399
MLP	0.4318	0.2366	0.4300	0.2337
NeuMF	0.5169	0.3017	0.4391	0.2487

The results indicate a decrease in the recommendation effectiveness of the local parameter NCF federated recommendation. Specifically, the embedding layer, serving as the local

parameter, exhibits some adverse effects. However, the decrease in the recommendation effectiveness of MLP is less pronounced, suggesting that the local parameter scheme has a relatively minor impact on MLP.

4.7. NCF Federated Recommendation with DP

Experiments were conducted on NCF federated recommendation with DP, employing overall and stratified sensitivity schemes with various parameters. The federated recommendation performance of three models (GMF, MLP, and NeuMF) was compared under different levels of noise.

The ratio of negative to positive training samples for all three models was set to 4:1, with a value of K equal to 10. Each model utilized a hidden vector layer of length 8, and the MLP component included a hidden layer with a dimensionality of 10. Five participants were involved in each round. In this experiment, we explored the joint recommendation performance of the NCF model with different differential privacy (DP) parameters. The experiments involved three models, GMF, MLP, and NeuMF, and Laplace noise was introduced to achieve DP. We recorded the HR, NDCG, and F1 scores under different DP parameters ϵ . The following results in Table 5 is a summary of the experimental data.

Table 5. NCF federated recommendation with DP.

Model	$\epsilon = 10$			$\epsilon = 20$			$\epsilon = 40$			$\epsilon = 60$		
	HR	NDCG	F1	HR	NDCG	F1	HR	NDCG	F1	HR	NDCG	F1
GMF	0.0028	0.0011	0.0014	0.2228	0.1213	0.1506	0.3787	0.2511	0.2879	0.4401	0.2411	0.3221
MLP	0.4389	0.2511	0.3459	0.4489	0.2677	0.3726	0.4458	0.2611	0.3785	0.4489	0.2622	0.3797
NeuMF	0.3020	0.2009	0.2459	0.33777	0.2255	0.2785	0.4626	0.2418	0.3465	0.4837	0.2590	0.3680

The results indicate that the three NCF models under the overall sensitivity scheme can maintain a certain level of recommendation accuracy only when ϵ is large, such as setting ϵ to 60. At small ϵ , the recommendation accuracy is far below the requirements. However, as ϵ increases, the HR, NDCG, and F1 scores of all the three models improved, but the increase gradually decreased; not only is privacy protection not guaranteed, but the recommendation effectiveness also deteriorates under the overall sensitivity scheme.

Under the stratified sensitivity scheme, the recommendation effectiveness of GMF improves relatively as ϵ increases. However, it remains less effective than the model without DP. It shows that DP has little effect on the MLP recommendation. And the recommendation effectiveness of NeuCF with DP noise is initially relatively poor, but as ϵ increases, its recommendation effectiveness improves. Overall, as ϵ increases, the recommendation effectiveness of GMF and NeuMF also improves relatively, but this comes at the cost of deteriorating privacy protection. Therefore, to better protect privacy while meeting the recommendation requirements, it is advisable to choose a smaller ϵ if possible, even though recommendations may be less effective. This ensures that models uploaded by clients are well protected.

4.8. NCF Federated Recommendation with LSH

The experiment was conducted on federated recommendation using locality sensitive hashing (LSH) with different dimensions of hash vectors. Three models (GMF, MLP, and NeuMF) were compared across various vector dimension levels to evaluate their federated recommendation performance.

The negative–positive training sample ratio was set to 4:1 with a K value equal to 10. Each model uses a hidden vector layer of length 8, while the MLP component includes a hidden layer of dimension 10. Each round of the survey had five participants. The specific structure mainly follows the framework of experiment 3.6, where the noise addition mechanism is replaced by the hash vector, and the results are shown in Table 6.

Table 6. NCF federated recommendation with LSH.

Model	Dim = 8			Dim = 64		
	HR	NDCG	F1	HR	NDCG	F1
GMF	0.4128	0.2387	0.3001	0.4729	0.2800	0.3549
MLP	0.3562	0.2211	0.2829	0.4699	0.2705	0.3526
NeuMF	0.3721	0.2397	0.3052	0.5123	0.2997	0.4064

The experimental results indicate that reducing the dimensionality of hash vectors leads to decreases in both HR and NDCG, resulting in degraded model performance and reduced recommendation accuracy. However, it may also contribute to reducing overfitting and improving computational efficiency. In practical applications, the choice of hash vector dimensionality needs to be balanced according to specific circumstances to achieve the optimal recommendation effectiveness.

4.9. Comparison of the LSH and the DP

In this section, we compare the recommended performance of using LSH and DP methods in a federated learning framework. By analyzing the data in Tables 5 and 6, the following is generated: Table 7, showing the recommended performance comparison of the LSH and DP methods under a federated learning framework.

Table 7. Comparison of LSH and DP.

Model	Dim = 64			$\epsilon = 60$			Increase/Decrease
	HR	NDCG	F1	HR	NDCG	F1	
GMF	0.4729	0.2800	0.3549	0.4401	0.2411	0.3221	HR increased by 0.0318, NDCG decreased by 0.0389, and F1 increased by 0.0328
MLP	0.4699	0.2705	0.3526	0.4489	0.2622	0.3797	HR increased by 0.0210, NDCG increased by 0.0083, and F1 decreased by 0.0271
NeuMF	0.5123	0.2997	0.4064	0.4837	0.2590	0.3680	HR increased by 0.0286, NDCG by 0.0407, and F1 by 0.0384

Note: The increase/decrease column shows the change in HR, NDCG, and F1 scores at LSH (dim = 64) compared to DP ($\epsilon = 60$). Positive values indicate that the LSH method outperformed the DP method on this index, and negative values indicate a decrease. It can be seen from the table that for the GMF and NeuMF models, the LSH method improves compared to DP in HR and F1 scores, but the GMF model decreased while the NeuMF model improved in NDCG. For the MLP model, the LSH method improved in HR, while it was close to the DP method, or even decreased slightly, in the NDCG and F1 scores.

These results suggest that the LSH method may have some advantages in recommendation performance, especially when using the hash vectors of higher dimensions, to improve the discrimination of recommendation systems. However, this promotion may come at the expense of privacy protection, as the LSH method does not provide quantifiable privacy protection as the DP method does. The DP method protects user privacy by introducing noise, but this noise reduces the learning ability of the model and thus affects the recommendation performance. However, with appropriate selection of the privacy protection parameter ϵ , a balance is achieved between protecting privacy and maintaining recommendation performance. In practical application, the choice of the LSH or DP method, or how to combine the two, should be comprehensively considered according to the specific requirements of the recommendation system and the requirements of privacy protection, so as to achieve the optimal recommendation effect and privacy protection level.

5. Conclusions

In this paper, we have explored the application of federated learning to the neural collaborative filtering (NCF) recommendation model. Our study began by comparing

the federated recommendation model based on NCF with traditional recommendation systems, highlighting the trade-offs between privacy and recommendation accuracy. We then examined the impact of participant numbers on federated learning and demonstrated the benefits of pre-training, particularly with the NeuMF model. Secondly, we proposed a model weight setting scheme based on participation levels and investigated the effects of different weight-setting functions on the federated learning process. Thirdly, we introduced a local parameter NCF recommendation scheme to study the influence of embedding layer local parameters on federated recommendations. Fourthly, we proposed an NCF federated recommendation model with a differential privacy (DP) scheme to enhance model privacy, achieving a balance between privacy and recommendation effectiveness through parameter tuning. Our fifth contribution involved the introduction of locality sensitive hashing (LSH) to the NCF collaborative recommendation model, aiming to improve privacy protection. By adjusting the dimensionality of hash vectors, we were able to enhance the system's performance. In summary, our research combines pre-training, local parameters, DP, and LSH techniques to improve federated learning in recommender systems. These enhancements have not only improved the federated learning process, efficiency, and recommendation accuracy but also bolstered data privacy and protected client data. Comparative experiments have validated the proposed models and assessed their recommendation effectiveness and privacy protection capabilities. Despite the progress made, there are several limitations to this study that warrant further investigation:

1. The model weight setting function is linear and may not capture the complexity of real-world user participation patterns. Non-linear or adaptive functions could be explored in future work.
2. Our experiments were conducted using the MovieLens 1M dataset, which may not be representative of all recommendation scenarios. A broader range of datasets should be considered to generalize the findings.
3. The MLP and the MLP component within the NeuMF model were limited to a single hidden layer with a fixed vector length of eight. The impact of varying the layer structure and vector dimensions requires further study.
4. The ratio of positive to negative samples was fixed at 1:4 in our experiments. Investigating the effects of different ratios could provide insights into model robustness.
5. The total number of training participations was constrained to ensure timely experiment completion. Real-world federated learning scenarios may involve more dynamic participation patterns.
6. The optimal dimensionality of hash vectors for LSH, which ensures both system recommendation effectiveness and privacy protection, remains to be determined. Future research could focus on finding this balance through a more rigorous analysis.

Author Contributions: Conceptualization, software, methodology, writing, H.D.; validation, formal analysis, M.Z.; resources, supervision, X.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities, No. xjj2021023, the Natural Science Basic Research Programme Project of Ningxia Province (2023AAC02086), and Special Project of Grassroots Teaching Organisation 'Exploration of Modular Teaching Reform of Introduction to Information Security'.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Peng, Y. A Survey on Modern Recommendation System based on Big Data. *arXiv* **2022**, arXiv:2206.02631.
2. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [\[CrossRef\]](#)

3. AbdulRahman, S.; Tout, H.; Ould-Slimane, H.; Mourad, A.; Talhi, C.; Guizani, M. A Survey on Federated Learning: The Journey From Centralized to Distributed On-Site Learning and Beyond. *IEEE Internet Things J.* **2021**, *8*, 5476–5497. [[CrossRef](#)]
4. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.S. Neural collaborative filtering. In Proceedings of the 26th International World Wide Web Conference, Perth, Australia, 3–7 April 2017; International World Wide Web Conferences Steering Committee: Geneva, Switzerland, 2017.
5. Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; Gai, K. Deep interest network for click-through rate prediction. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; ACM Press: New York, NY, USA, 2018; pp. 1059–1068.
6. Zhou, G.; Mou, N.; Fan, Y.; Pi, Q.; Bian, W.; Zhou, C.; Zhu, X.; Gai, K. Deep interest evolution network for clickthrough rate prediction. In Proceedings of the 2019 AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; AAAI Press: Palo Alto, CA, USA, 2019; pp. 5941–5948.
7. Li, C.; Liu, Z.; Wu, M.; Xu, Y.; Zhao, H.; Huang, P.; Kang, G.; Chen, Q.; Li, W.; Lee, D.L. Multiinterest network with dynamic routing for recommendation at Tmall. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; ACM Press: New York, NY, USA, 2019; pp. 2615–2623.
8. Dwork, C. Differential Privacy: A Survey of Results. In Proceedings of the International Conference on Theory and Applications of Models of Computation, Xi'an, China, 25–29 April 2008; Springer: Berlin/Heidelberg, Germany, 2008. [[CrossRef](#)]
9. Zhou, J.; Dong, X.L.; Cao, Z.F. Research advances on privacy preserving in recommender systems. *J. Comput. Res. Dev.* **2019**, *56*, 2033–2048.
10. Orekondy, T.; Oh, S.J.; Zhang, Y.; Schiele, B.; Fritz, M. Gradient-leaks: Understanding and controlling deanonymization in federated learning. *arXiv* **2018**, arXiv:180505838.
11. Berlioz, A.; Friedman, A.; Kaafar, M.A.; Boreli, R.; Berkovsky, S. Applying differential privacy to matrix factorization. In Proceedings of the 9th ACM Conference on Recommender Systems, Vienna, Austria, 16–20 September 2015; ACM Press: New York, NY, USA, 2015; pp. 107–114.
12. Xu, J.; Ren, X.; Lin, J.; Sun, X. Dp-gan: Diversity-promoting generative adversarial network for generating informative and diversified text. *arXiv* **2018**, arXiv:1802.01345.
13. McSherry, F.; Mahajan, R. Differentially-private network trace analysis. *ACM SIGCOMM Comput. Commun. Rev.* **2010**, *40*, 123–134. [[CrossRef](#)]
14. Das, R.; Hashemi, A.; Sanghavi, S.; Dhillon, I.S. DP-NormFedAvg: Normalizing Client Updates for Privacy-Preserving Federated Learning. *arXiv* **2021**, arXiv:2106.07094.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.