MetaFRS- Federated Learning based Cold Start Recommendation System using Meta-Learning

1st Ujjwal Kumar
Department of Computer Engineering
Somaiya Vidyavihar University
Mumbai, India
ujjwal.kumar@somaiya.edu

2nd Dhairya Ameria
Department of Computer Engineering
Somaiya Vidyavihar University
Mumbai, India
dhairya.a@somaiya.edu

3rd Pragya Gupta
Department of Computer Engineering
Somaiya Vidyavihar University
Mumbai, India
pragya.g@somaiya.edu

2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT) | 979-8-3503-1341-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/EASCT59475.2023.10392700

***Abstract*—The field of recommender systems has seen significant advancements recently, but several challenges remain. In this paper, we address two challenges in recommendation systems. Firstly, conventional recommendation systems require uploading private data to a central server for training, which inevitably impacts user privacy. To tackle this issue, we use Graph Federated Learning (GFL), a novel paradigm for distributed learning that ensures privacy preservation by enabling training on distributed data and eliminating direct data sharing. However, distributed recommender systems have a performance gap compared to non-distributed ones due to incomplete user-item interaction graphs. As a result, these systems struggle to utilize indirect user-item interactions effectively. Secondly, the cold start scenario, where a recommender system lacks sufficient data to make accurate recommendations for new users or items. Therefore, we propose MetaFRS - Federated Learning based Cold Start Recommendation System using Meta-Learning to overcome these limitations. Our system incorporates a graph neural network that uses attention mechanisms and an aggregation layer to summarize various orders of indirect user-item and user-user interactions. Meta-learning algorithm is employed to address the issue of sparse interactions in cold start scenarios and incomplete user-item graphs in a distributed setup.**

Keywords—Federated Learning, meta-learning, cold-start recommendations, Graph Neural Networks

I. INTRODUCTION

Recommender systems are sophisticated algorithms developed to provide recommendations on items or content by analyzing user preferences and behaviors. They have widely attracted interest from industries such as streaming platforms, e-commerce platforms and online content recommendations to enhance the user experience and increase engagement by providing personalized recommendations. There are various types of recommender systems, which include Collaborative filtering systems that recommend items based on the preferences of similar users [1], Content-based systems that recommend items based on their attributes and similarity to items previously liked by the user [2] and Hybrid systems that combine multiple approaches to provide more accurate recommendations [3]. These systems are highly dependent on collecting user behavior data. Gathering such information undermines the privacy of the user.

Graph-federated learning (GFL) is a promising approach to addressing privacy concerns in recommender systems. By leveraging the principles of federated learning and the structure of graph data, it allows collaborative training of recommendation models without the need for centralized data aggregation [4]. In GFL, user data is represented as a graph, where nodes represent users, items, or other entities, and edges represent relationships or interactions between them. This graph structure preserves the privacy of individual users by avoiding the direct sharing of their raw data. Instead, only aggregated or anonymized information is exchanged between the central server and the participating devices. However, the issue of an incomplete user-item graph or limited interactions between users and items still poses a challenge in the federated setting. Graph federated learning approach needs to address this limitation in scenarios where the user-item graph is sparse or incomplete.

The recent approach FedGNN [5] addresses the limitations of federated recommender systems by adapting graph neural network(GNN) models to the federated learning (FL) setting. However, FedGNN only utilizes first-order user-item interactions and relies on pseudo-item sampling for incomplete user-item graphs. This can result in suboptimal recommendations with limited item diversity and poor generalization to unseen scenarios. To overcome these limitations, [6] proposes explicitly storing latent embeddings of users and items to fully exploit indirect user-item interactions. However, storing embeddings introduces challenges such as maintenance and update overhead and difficulty capturing dynamic interactions. As the number of users and items increases, the storage and retrieval of latent embeddings can become computationally challenging. These factors can hinder the system's performance and its ability to provide timely and accurate recommendations in evolving or dynamic environments. Moreover, both of these systems do not consider the "cold-start scenario". The cold start scenario arises when there is limited or no interaction data available, and the stored embeddings may not provide effective recommendations for new or unseen users and items. To address the problem, techniques such as content-based recommendations, popularity-based recommendations [7], and utilizing demographic information [8] have been employed in non-distributed GNN-based recommender systems to make initial recommendations until sufficient user feedback is available. These, however, only exacerbate the issue of privacy. Therefore, in our system, we use Meta-learning, an approach that can be leveraged to tackle these challenges. Meta-learning focuses on learning the learning process itself, enabling models to quickly adapt to new tasks or scenarios with limited data.

In this research paper, we propose a novel approach called MetaFRS that integrates meta-learning techniques to handle cold start scenarios and improve the recommendation accuracy for both new users and new items. By utilizing meta-learning, the system can adapt quickly to new data and make accurate predictions even with limited user-item interactions. To achieve this, each client in our proposed system stores the user interactions with items as well as interactions with their neighbors. We employ a Graph Attention Network(GAT)[9] to compute attention weights for user-item and user-user

[neighbor] pairs, capturing the importance of each interaction. The item embeddings of the user are utilized as the support set for the meta-learning process. By calculating the loss on the support set, the model's attention weights and forward layer are updated locally. The updated model is then tested on a query set, which consists of new items that the user has not interacted with. The loss of the query set is aggregated at the server to update the model globally. Additionally, privacy techniques are applied to ensure the confidentiality and security of user interactions. This ensures that user data remains protected while still enabling effective recommendation generation.

The major contributions of this paper are outlined as follows.

- We propose MetaFRS, a novel federated learning-based cold start recommendation system that leverages meta-learning to effectively address cold start scenarios.
- Our approach overcomes the challenge of incomplete user-item interactions in a federated learning setting and successfully handles the cold start scenarios for both new users and new items.
- Privacy-preserving techniques, integrated within the federated learning framework, are employed to safeguard the confidentiality and security of user interactions.
- We evaluate the effectiveness of MetaFRS in enhancing recommendation accuracy by comparing it to multiple baseline models and its performance on a range of cold start scenarios.

II. PRELIMINARIES

A. Graph Federated Learning

In the context of Graph Federated Learning, the recommendation system relies on a user-item graph. This graph consists of a user set $\mu^{(n)} = \{u_1^{(n)}, u_2^{(n)} ... u_N^{(N)}\}$ and an item set $\tau^{(n)} = \{t_1^{(n)}, t_2^{(n)} ... t_N^{(N)}\}$ where the number of users is denoted as 'n' and the number of items is denoted as 'm'. The interactions between users and items are represented by a 2D rating matrix, denoted as R, which has dimensions n x m and contains the ratings given by users to items. In the federated setting, different models such as Graph Neural Networks(GNNs)[10], Graph Convolutional Networks(GCNs)[11] and Graph Attention Networks(GATs)[12] are commonly employed for recommendation tasks. FedGraphNN[13], is a benchmark for federated learning with graph neural networks. FedPerGNN[5] is the first federated GNN-based recommender system on user-item graphs. In FedGNN[5], the user-item interaction data is utilized to train GNN-based recommendation models and pseudo-item interactions are used to create higher-order interactions among nearby users while still protecting their privacy. FedGR[14] employs a GAT network to represent user learnings from social relationship graphs and past item interactions while utilizing cryptographic methods and noise injection for data protection. In FedGRec[6], latent embeddings are employed as pseudo-interactions for absent neighbors during local training that are stored to represent indirect user-item interactions. FeSoG[15] uses dynamic noise to maintain privacy while using attention layer, relational graph aggregation layers, and rating prediction layers for recommendations.

B. Cold Start Recommendation

In a recommendation system, a cold start scenario occurs when there is little to no knowledge about new users or items, which makes it challenging to provide reliable recommendations. There are majorly 4 different types of cold start scenarios, including user cold start (lack of user-specific data), item cold start (limited information about new items), context cold start (insufficient contextual information), and hybrid cold start (a combination of multiple types). Strategies like content-based recommendations, popularity-based recommendations, collaborative filtering, and hybrid approaches can be used to address these challenges. Some approaches such as [16] [17] have also used meta-learning to deal with this issue.

C. Meta Learning

With a limited number of training samples, Meta-Learning successfully applies prior knowledge to new tasks. It can be categorized into memory-based, metric-based and optimization-based approaches. Memory-based methods store key knowledge using memory architectures or specialized training processes, but they often come with a large number of tunable parameters. Metric-based methods learn a distance metric for instance similarity but are more suitable for classification tasks. Optimization-based methods learn parameters that enable quick adaptation to new tasks, resulting in cutting-edge performance.

Researchers have done great work in non-distributed environments by leveraging meta-learning techniques for cold-start recommendations, such as MAML[18], MetaKG [19], CLOVER[20] and MELU[21]. These approaches have demonstrated strong generalization performance and proposed candidate selection strategies for personalized preference estimation. AMeLU[22] seeks to enhance Cold-Start Recommendation performance using the combination of attention network and meta-learning. [23] uses a meta network that consists of users' characteristic embeddings to generate personalized bridge functions so that we can have personalized transfer of preferences for each user. Recently, there has been a small amount of research on combining federated learning with meta-learning techniques for recommendation systems such as[25].

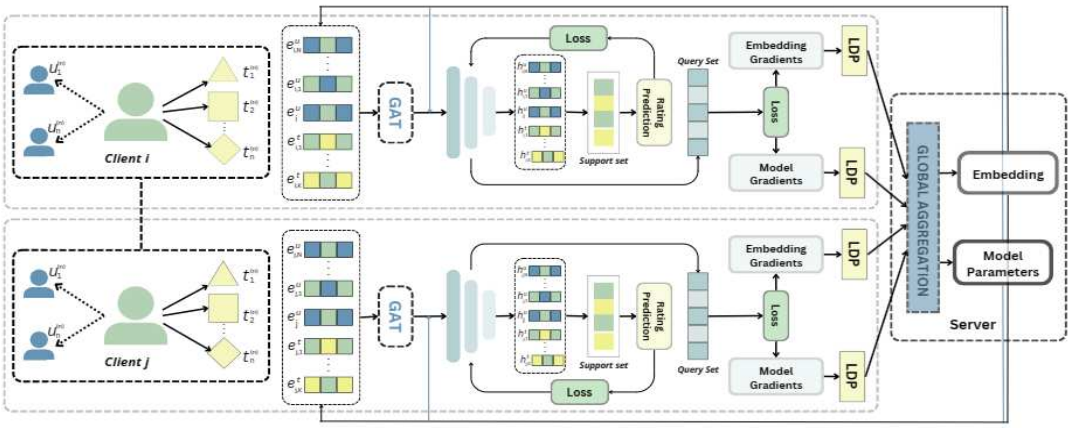

Fig. 1. MetaFRS Architecture Diagram.

III. METAFRS : A NOVEL APPROACH

A. Problem Formulation

The objective of Federated Learning based Cold Start Recommendation System using Meta-Learning [MetaFRS] is to improve the recommendation system in cold start scenarios by leveraging meta-learning. In this context, each client Ci in the system maintains a local graph Gn, which includes the user's first-order neighbors and their interactions with items. The local graph is constructed from partially observed privacy data. The client's graph consists of two types of edges: user-item edges and user-user edges representing neighbor interactions with the user. For each client Ci, we denote its rated items as $\tau^{(n)} = \{t_1^{(n)}, t_2^{(n)} ... t_N^{(n)}\}$ and the user interactions as $\mu^{(n)} = \{u_1^{(n)}, u_2^{(n)} ... u_N^{(n)}\}$. The goal is to predict the ratings γ_i ,of unobserved items or enhance recommendations for new users. By utilizing meta-learning, MetaFRS aims to effectively handle the challenges posed by cold start scenarios and improve the overall performance of the recommendation system.

B. Notations

Notation	Description
T(n) U(n) d θ1 θ2 W1 W2 Yi Ci	Set of rated items for a client Set of neighbor user of a client Dimension of the embedding vector Embedding vector for user and items i Model parameters u Linear Weight Matrix for user-user Linear Weight Matrix for user-item Predicted rating of user u on item i Client i of total clients n.

C. Methodology

In this section, we will explain the MetaFRS framework as shown in Fig.1. We have five important components: node embeddings, Graph Neural Network, Meta learner, Aggregation Layer and Privacy.

1) Node Embeddings: Node embeddings are used as important components to represent graph structural information. Our framework, MetaFRS, incorporates embedding layers for both user and item nodes. We represent these embeddings as $E_u \in R^{dxN}$ for user nodes and $E_t \in R^{dxN}$ for item nodes. These embeddings are centrally maintained and can be downloaded from the server by each client. The client retrieves the complete embedding table from the server, enabling it to associate user/item interactions with their corresponding embeddings. To provide a more detailed explanation, for each client Ci, the items that the client has rated are represented as $\tau^{(n)} = \{t_1^{(n)}, t_2^{(n)} ... t_K^{(n)}\}$, and the user interactions are represented as $\mu^{(n)} = \{u_1^{(n)}, u_2^{(n)} ... u_N^{(n)}\}$. These items and user interactions are converted into their respective embeddings using the embedding table retrieved from the server. Specifically, we denote the e_i^t embeddings for client Ci as $[e_{i,1}, e_{i,2}, ..., e_{1,K}]$ and $[e_{i,1}, e_{i,2}, ..., e_{1,N}]$ where K represents the total number of item neighbors and N represents the total number of user neighbors for the specific user.

2) Graph Neural Network: In our framework, we have used a Graph attention network which is designed by using the self-attention mechanism to learn the node embeddings for both the user-item and user-neighbor embeddings. However, as each neighbor of the user has an unequal contribution, to determine its importance we first learn the weight of each neighbor by passing through an attention score which is formulated as in Eq

$$O_n = Attention(W_{1e_{u_n}}, W_{1e_{u_p}}) \tag{1}$$

Where $W_1 \in R^{dxN}$ is a linear mapping matrix that is applied to every node. To make the nodes comparable across all the nodes we normalize them across all the neighbors of the center node using a softmax function separately for user-item and user-user embeddings in Eq 2:

$$\gamma_{rq} = softmax_q(o_{rq}) = \frac{\exp(e_{rq})}{\sum_{i=1}^{P} \exp(o_{rq})} \tag{2}$$

Where $\gamma_{rq} = \alpha_{np} = \beta_{nk}$. We have the attention layer as a single layer feed-forward network which is parameterized with a weight vector: $\gamma \in R^{2d}$ and when applied a LeakyReLU[31] activation the overall expansion of the attention mechanism is expressed as for both user-user and user-item.

3) Aggregation Layer: Now that the prediction has been made, the user embeddings must be aggregated. To do this, neighbor item nodes and neighbor user nodes with their

associated attention weights must be aggregated as shown in Eq 3.

$$h_t^{(n)} = \sum_{k=1}^{P} \gamma_{rq} W_{h^{e_{lk}}} \quad (3)$$

This gives us the hidden embeddings of both the user and item neighbors for aggregating the user and item neighbors. Where, W_h is the linear weight matrix. This updates the weights of the model and in turn, the hidden embeddings locally which is then used by the meta learner.

4) Meta-learner: In the meta-learner framework, we utilize the support set, which consists of the items that the user has interacted with. We extract the item embeddings from this set and employ a linear weight matrix that encompasses the weights of each layer. By employing the matching network algorithm [26] of meta-learning, we update the weights of both the output layer and the hidden embeddings. In essence, we leverage the user-item history represented by the support set to refine the model parameters. This approach effectively addresses the cold start scenario, enabling us to handle situations where there is limited or no prior interaction information for clients. Additionally, it facilitates the resolution of incomplete graphs, as it enables capturing higher-order interactions. Subsequently, we use the query set, which consists of new items that the user has not interacted with previously, and we retrieve the embeddings of these items from the server as well. On the calculation of the loss of that, we update the overall model parameters and user item embeddings.

The presented Algorithm 1 showcases the utilization of meta learning in the context of initializing model parameters and retrieving user-item and user-user embeddings from the server. Subsequently, attention weights are computed, and user and item embeddings are aggregated using Equation 1. The model parameters are then combined with the support set, consisting of user-interacted items and corresponding embeddings. By considering the original ratings, the loss L is computed. To optimize the learning process, an Adam optimizer is employed. The algorithm iteratively updates the model parameters and minimizes the loss L as mentioned in Eq 4.

$$L_i = \frac{1}{|H_i|} \sum_{j \in H_i} (y_{ij} - \hat{y}_{ij})^2 \quad (4)$$

Here, Hi is a set of items consumed by user i and j represents item consumed by user i, i.e. Through this iterative process, the local model parameters are updated to achieve user-level personalization. This update is performed for a certain number of epochs to enhance the model's ability to capture user preferences. Subsequently, the user is presented with a query set, and the loss is calculated based on the model gradients, as well as the gradients of user-item and user-user interactions. These gradients are sent to the server for aggregation, allowing the server to combine and aggregate the losses. By aggregating the loss at the server, the model parameters, as well as the user-item and user-user embeddings, are updated accordingly. This aggregation mechanism enables the system to leverage collective knowledge and improve the overall recommendation quality.

5) Privacy: To ensure that we have a secure aggregation of the user, item gradients and the model parameters we use LDP (Local Differential Privacy)[27][28] in which we add dynamic Laplacian noise instead of static as constant noise isn't appropriate if we are dealing with gradients at different magnitudes

$$\bar{g}^{(n)} = \text{clip}(g^{(n)}, \delta) + \text{Laplacian}(0, \lambda.\text{mean}(g^{(n)})) \quad (5)$$

As seen in Eq 5, g(n) is the combining the gradients of all the trainable parameter $g^{(n)} = \theta_1 t^{(n)}, \theta_{1_u}^{(n)}, \theta_{2_m}^{(n)} = \frac{\partial L_u}{\partial \theta}$ where θ denotes all trainable parameters.

D. Algorithm

Algorithm 1: MetaFRS - Federated Learning based Cold Start Recommendation System using Meta-Learning

	Input:	Embedding size, learning rate: d, η Total number of items and clients: N, M, T Clients local graph: $\{G_n \mid_{n=1}^{N}\}$ LDP parameters: λ	
	Output:	Model parameters and embeddings θ Local client embeddings $\{e_{u_n}^{*} \mid_{n=1}^{N}\}$	
1.	Initializing θ		
2.	while not converge do		
3.		Function Server:	
4.			Pick random set of Clients(n)
5.			Aggregating loss: $f\{\theta'\} \leftarrow \beta \sum_{i \in B} \nabla_{\theta_i} L_i^{'}(f_{\theta_1, \theta_2^i})$
6.			Global update: $\theta_1 \leftarrow \theta_1 - f\{\theta'\}$
7.			Global update: $\theta_2 \leftarrow \theta_2 - f\{\theta'\}$
8.	Function Client(i, θ):		
9.		Download θ item embedding and model parameters from Server.	
10.		Calculate attention weight $\gamma_{rq} = softmax(e_{rq}) = \frac{e^{e_{rq}}}{\sum_{k=N}^{n} e^{o_{rk}}}$	
11.		M epochs update $\theta_2^i = \theta_2^{i'}$	
12.		Evaluate $\nabla_{\theta_2^i} L_i^{'}(f_{\theta_1, \theta_2^i})$	
13.		Local update $\theta_2^i = \theta_2^i - \alpha \nabla_{\theta_2^i} L_i^{'}(f_{\theta_1, \theta_2^i})$	
14.		LDP $(\theta_1, \theta_2) \rightarrow G_n$	
15.		Return G_n	

IV. EXPERIMENTS

A. Datasets

In our experiment we use three widely benchmarked dataset for recommendation which includes MovieLens Ml-100k, Ml-1M and ML-10M the preprocessed version. The dataset provides us with basic user item information . The user content consists of Gender, Age, Occupation, Zip Code

whereas the item content consists of the Rating, Genre, Director, Actor. We have the statistics in Table I.

TABLE I. STATISTICS OF THE DATASET

Dataset	Users	Items	Ratings	Rating Level
100K	943	1,682	100,000	1,2...5
M	6,040	3,706	1,000,209	1,2...5
10M	69,878	10,677	10,000,054	0.5,1...5

To commence, we need to categorize the dataset into two categories: "Pre-1997" and "Post-1998" movies. We designated movies released before 1997 as "Present items" and those released after 1998 as "new items." Following this categorization, we further split the dataset into four segments, each tailored for use in various cold-start scenario. Present items for present users(Scenario 1), Present items for current users(Scenario 2), Current items for present users.(Scenario 3) and Current items for current users(Scenario 4). For every user, we selected ten items at random from their item history and designated them as the query set Hi'. The remaining items were then allocated to the support set Hi.

B. Parameter Settings

Here are the algorithm's parameter settings: We use a GAT network to calculate attention weights for interactions between users, both user-to-user and user-to-item. The dataset is divided randomly into three parts: training 70%, validation 15%, and test 15%. We fine-tune the hyperparameters based on how well the model performs on the validation set, with RMSE and MAE as our evaluation metrics. We set the gradient clipping threshold to 0.4 and use a strength of 0.1 for Laplacian noise in the LDP technique of MetaFRS. To optimize other hyperparameters, we employ a grid search approach. The step sizes α and β are specifically set to 6×10^{-1} and 6×10^{-5}, respectively. We experiment with local epochs, exploring values between 5 and 10. The embedding size is varied from 16, 32, 64, 128, with 32 being identified as the best choice. We search for the learning rate within 0.1, 0.05 and 0.01. We investigate the user batch size in each training round from16, 32, 64, 128. Lastly, we halt the training process if the RMSE value remains unchanged for 5 consecutive validation rounds.

C. Baselines

We compare our model with non-distributed recommender systems and a Federated recommender system as well. We use [29], which is a collaborative filtering method based on variational autoencoder and is considered a state-of-the-art method. Additionally, we compare our model with NGCF[30] and LightGCN[31], which are two other graph-based recommender systems. For the federated baseline, we compare FedMF[32] and FedGNN. FedMF is a matrix-based federated recommendation system, while FedGNN is a graph-based recommendation system. We test these baselines using our MetaFRS framework and compare their performance.

D. Performance Evaluation

In the evaluation of our research as we can see in Table II, we find that present user and present item scenarios demonstrate better results with non-distributed traditional recommender systems, compared to our proposed approach. However, our approach, leveraging meta learning techniques, excels in addressing cold start scenarios such as current users, current items, and current items for present users, offering substantial improvements over traditional methods. Furthermore, our model outperforms other federated learning algorithms, particularly in terms of capturing higher order interactions within the graph, resulting in superior performance on the present user and present item dataset.

TABLE II. COMPARISON OF METAFRS WITH VARIOUS BASELINE MODELS

Type	Methods	MovieLens					
		ML-100K		*ML-1M*		*ML-10M*	
		RMSE	*MAE*	*RMSE*	*MAE*	*RMSE*	*MAE*
Scenario 1	MultiVae	0.164	0.12	0.15	0.11	0.13	0.102
	LightGCN	0.15	0.08	0.13	0.09	0.11	0.06
	NGCF	0.17	0.13	0.12	0.093	0.08	0.05
	FedMF	1.08	0.99	1.02	0.92	0.99	0.89
	FedGNN	0.92	0.88	0.848	0.79	0.803	0.71
Scenario 2	MultiVae	1.0784	0.8299	0.987	0.78	0.95	0.821
	LightGCN	1.069	0.8599	0.97	0.672	1.045	0.945
	NGCF	1.087	0.8779	0.92	0.87	1.32	1.02
	FedMF	1.42	1.23	1.21	1.09	1.23	1.02
	FedGNN	1.56	1.14	1.31	1.09	1.21	1.03
Scenario 3	MultiVae	1.2441	1.07	1.032	0.876	1.002	0.876
	LightGCN	1.26	1.02	1.021	0.96	1.01	0.867
	NGCF	0.9371	0.78	0.879	0.798	0.786	0.754
	FedMF	1.89	1.32	1.67	1.43	1.567	1.324
	FedGNN	2.02	1.78	1.67	1.43	1.56	1.236
Scenario 4	MultiVae	1.2441	1.07	1.032	0.876	1.002	0.876
	LightGCN	1.2665	1.02	1.021	0.96	1.01	0.867
	NGCF	0.929	0.876	0.789	0.654	0.678	0.56
	FedMF	1.902	1.456	1.79	1.54	1.65	1.32
	FedGNN	1.87	1.345	1.67	1.34	1.43	1.21

V. CONCLUSION

In conclusion, we have introduced MetaFRS, a Cold Start Recommendation System based on Federated Learning that overcomes significant challenges in the field of recommender systems. We have used Graph Federated Learning (GFL), a novel distributed learning paradigm, to address the privacy concerns in recommender systems. GFL ensures the preservation of privacy by enabling training on distributed data eliminating the need for direct data sharing. This allows us to effectively capture indirect user-item and user-user interactions, which are often overlooked in distributed recommender systems with incomplete user-item interaction graphs. To address the issue of cold start scenario, we have utilized a meta-learning algorithm that leverages the sparse interactions in cold start scenarios and incomplete user-item graphs. This enables our system to make accurate recommendations even in situations where limited data is available for new users or items. Through extensive evaluations, we have compared our MetaFRS framework with non-distributed recommender systems as well as with federated baseline models. Our results show the effectiveness of MetaFRS in handling the cold start scenario and achieving competitive performance in a distributed and privacy-preserving setting. Overall, our proposed MetaFRS framework not only addresses privacy concerns and performance limitations but also offers a promising solution for effective recommendations in challenging scenarios.

REFERENCES

[1] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," in Proceedings of the 10th International Conference on World Wide Web, in WWW '01. New York, NY, USA: Association for Computing Machinery, 2001, pp. 285–295. doi: 10.1145/371920.372071.

[2] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," Recommender systems handbook, pp. 73–105, 2011.

[3] B. Walek and P. Fajmon, “A hybrid recommender system for an online store using a fuzzy expert system,” Expert Syst Appl, vol. 212, p. 118565, 2023.

[4] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, “Peer-to-peer federated learning on graphs,” arXiv preprint arXiv:1901.11173, 2019.

[5] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, “FedGNN: Federated Graph Neural Network for Privacy-Preserving Recommendation,” ArXiv, vol. abs/2102.04925, 2021.

[6] J. Li and H. Huang, “FedGRec: Federated Graph Recommender System with Lazy Update of Latent Embeddings,” arXiv e-prints, p. arXiv:2210.13686, Oct. 2022, doi: 10.48550/arXiv.2210.13686.

[7] F. Islam, M. S. Arman, N. Jahan, M. H. Sammak, N. Tasnim, and I. Mahmud, “Model and Popularity Based Recommendation System- A Collaborative Filtering Approach,” in 2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2022, pp. 1–5. doi: 10.1109/ICCCNT54827.2022.9984348.

[8] M. Y. H. Al-Shamri, “User profiling approaches for demographic recommender systems,” Knowl Based Syst, vol. 100, pp. 175–187, 2016, doi: https://doi.org/10.1016/j.knosys.2016.03.006.

[9] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph Attention Networks.” 2018.

[10] R. Liu, P. Xing, Z. Deng, A. Li, C. Guan, and H. Yu, “Federated Graph Neural Networks: Overview, Techniques and Challenges.” 2022.

[11] Y. Yin, Y. Li, H. Gao, T. Liang, and Q. Pan, “FGC: GCN-Based Federated Learning Approach for Trust Industrial Service Recommendation,” IEEE Trans Industr Inform, vol. 19, no. 3, pp. 3240–3250, 2023, doi: 10.1109/TII.2022.3214308.

[12] H. Wang, C. Bai, and J. Yao, “Federated Graph Attention Network for Rumor Detection.” 2022.

[13] C. He et al., “FedGraphNN: A Federated Learning System and Benchmark for Graph Neural Networks.” 2021.

[14] C. Ma, X. Ren, G. Xu, and B. He, “FedGR: Federated Graph Neural Network for Recommendation Systems,” Axioms, vol. 12, no. 2, 2023, [Online]. Available: https://www.mdpi.com/2075-1680/12/2/170

[15] Z. Liu, L. Yang, Z. Fan, H. Peng, and P. S. Yu, “Federated Social Recommendation with Graph Neural Network,” vol. 13, no. 4, Aug. 2022, doi: 10.1145/3501815.

[16] Y. Zheng, S. Liu, Z. Li, and S. Wu, “Cold-start Sequential Recommendation via Meta Learner,” CoRR, vol. abs/2012.05462, 2020, [Online]. Available: https://arxiv.org/abs/2012.05462

[17] Y. Lu, Y. Fang, and C. Shi, “Meta-Learning on Heterogeneous Information Networks for Cold-Start Recommendation,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in KDD ’20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1563–1573. doi: 10.1145/3394486.3403207.

[18] C. Finn, P. Abbeel, and S. Levine, “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.” 2017.

[19] Y. Du, X. Zhu, L. Chen, Z. Fang, and Y. Gao, “MetaKG: Meta-learning on Knowledge Graph for Cold-start Recommendation,” IEEE Trans Knowl Data Eng, 2022, doi: 10.1109/tkde.2022.3168775.

[20] T. Wei and J. He, “Comprehensive Fair Meta-learned Recommender System,” in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM , Aug. 2022. doi: 10.1145/3534678.3539269.

[21] H. Lee, J. Im, S. Jang, H. Cho, and S. Chung, “MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in KDD ’19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1073–1082. doi: 10.1145/3292500.3330859.

[22] S. Liu, Y. Liu, X. Zhang, C. Xu, J. He, and Y. Qi, “Improving the Performance of Cold-Start Recommendation by Fusion of Attention Network and Meta-Learning,” Electronics (Basel), vol. 12, no. 2, 2023, doi: 10.3390/electronics12020376.

[23] [Y. Zhu et al., “Personalized Transfer of User Preferences for Cross-Domain Recommendation,” in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, in WSDM ’22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 1507–1515. doi: 10.1145/3488560.3498392.

[24] G. and W. X. and Q. Z. and W. Y. Ai Zhengyang and Wu, “Towards Better Personalization: A Meta-Learning Approach for Federated Recommender Systems,” in Knowledge Science, Engineering and Management, B. and K. L. and Z. T. and Q. M. Memmi Gerard and Yang, Ed., Cham: Springer International Publishing, 2022, pp. 520–533.

[25] Y. Di and Y. Liu, “MFPCDR: A Meta-Learning-Based Model for Federated Personalized Cross-Domain Recommendation,” Applied Sciences, vol. 13, no. 7, 2023, doi: 10.3390/app13074407.

[26] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, “Matching Networks for One Shot Learning.” 2017.

[27] X. Ren, L. Shi, W. Yu, S. Yang, C. Zhao, and Z. Xu, “LDP-IDS: Local Differential Privacy for Infinite Data Streams,” in Proceedings of the 2022 International Conference on Management of Data, ACM , Jun. 2022. doi: 10.1145/3514221.3526190.

[28] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, “LDP-Fed: Federated Learning with Local Differential Privacy.” 2020.

[29] J. Xu et al., “Multi-VAE: Learning Disentangled View-common and View-peculiar Visual Representations for Multi-view Clustering.” 2021.

[30] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, “Neural Graph Collaborative Filtering,” in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM , Jul. 2019. doi: 10.1145/3331184.3331267.

[31] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, “LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation.” 2020.

[32] S. Wang and T.-H. Chang, “Federated Matrix Factorization: Algorithm Design and Application to Data Clustering.” 2020 .