Reproduction: EFVAE: Efficient Federated Variational Autoencoder for Collaborative Filtering

Abstract

Federated recommender systems address privacy concerns. FedVAE, an extension of MultVAE, suffers from high communication costs due to its item-dependent first/last layers.EFVAE (Efficient Federated Variational AutoEncoder),leveraging Federated Collaborative Importance Sampling (FCIS) to reduce communication through client-server collaborative sampling and dynamic multi-stage decoding.Experiments show EFVAE reduces communication by up to 94.51% while maintaining or even improving recommendation performance (up to 13.79% on sparse datasets).

Keywords: Collaborative Filtering; Federated Learning; Importance Sampling;Variational Autoencoder

1 Introduction

Recommender systems (RS) address information overload. Traditional RS rely on centralized data, posing privacy risks [20] and facing regulations like GDPR. Federated Learning (FL) [10]offers privacy-preserving model training on decentralized data. Federated Recommender Systems (FRS) [1, 2, 4, 6, 11, 13, 15], based on FL, allow multi-client collaborative model training with a server without sharing raw data. However, existing FRS methods like FedVAE [14], based on MultVAE [8], suffer from high communication costs with large item catalogs. MultVAE's parameter count is directly tied to the number of items, causing large model transmission overhead.

To solve this, Efficient Federated Variational AutoEncoders (EFVAE) is proposed, with Federated Collaborative Importance Sampling (FCIS). FCIS reduces communication via client-server collaborative sampling and dynamic multi-stage approximation of the decoding distribution. Specifically, the server pre-clusters items; users receive clusters, sample within them, and send back cluster indices. The server then helps users sample items from those clusters and sends them back. This only transmits some item parameters, greatly reducing communication and maintaining good recommendation performance. EFVAE works with other AutoEncoder (AE) methods [8, 18] and privacy technologies like Local Differential Privacy (LDP) [5] and secret sharing [17, 19].

2 Related works

In this section, we review federated recommender systems and centralized AutoEncoder-based Collaborative Filtering (AE-CF).

2.1 Federated Recommender system

Federated recommender systems (FRS) have emerged to address privacy concerns in recommendation by collaboratively training models between clients and servers [1, 6, 9, 12, 14, 16]. Examples include MetaMF [9], which uses a meta-network for generating embeddings and sends gradient data to the server; FCF [1], which adapts collaborative filtering for the federated setting; FedFast [11], which accelerates convergence; FedPOIRec [12], which incorporates encrypted social connections for POI recommendations; SemiDFEGL [16], which uses semi-decentralized federated ego graph learning; and FedVAE [14], a federated version of MultVAE. However, these systems often require model components (e.g., user embeddings) to reside on user devices [21], thus excluding non-participating users from receiving recommendations.

2.2 Autoencoder for Collaborative Filtering

Autoencoder (AE)-based recommender systems [8, 18]learn low-dimensional user and item representations for efficient recommendations. Variational Autoencoders (VAEs) [7] are a common approach, with MultVAE [8] being a prominent example.

MultVAE encodes a user u's interactions x_u into a latent vector z_u that follows a Gaussian distribution and decodes it back to $\hat{x}_u$, a probability distribution over items. It maximizes the enhanced Evidence Lower Bound (ELBO), consisting of reconstruction loss and KL regularization,where a parameter $bata$ controls the strength of regularization:

$$\mathcal{L}_\beta(x_u;\theta;\phi) = \mathbb{E}_{q_\phi(z_u|x_u)}[log_{p_\theta(x_u|z_u)}] - \beta \cdot KL(q_\phi(z_u|x_u)||p(z_u)), \tag{1}$$

where the variational posterior $q_\phi(z_u|x_u)$ approximates the true distribution, the prior $p_\theta(z_u)$ is standard Gaussian, and the generative distribution of x_u given z_u is multinomial. A dropout layer at the input enhances training.

MultDAE [8] is a denoising autoencoder applying Bernoulli noise without the VAE structure, essentially a variant of MultVAE without the KL divergence. RecVAE [18] enhances recommendation performance through a composite prior, dynamic adjustment of the KL divergence hyperparameter $bata$ based on user feedback, and an alternating update strategy prioritizing encoder updates.

Like MultVAE, other methods can suffer from high communication overhead in federated settings due to the large number of encoder/decoder parameters when dealing with many items.

3 Method

In this section, we first introduce Federated Collaboration Importance Sampling(FCIS). Then, we summarize the process of EFVAE.

3.1 Federated Collaboration Importance Sampling

To reduce training communication costs, we aim to minimize transmission by sending only essential parameters. Thus, we first analyze the communication cost of the encoder's first layer and the decoder's last layer to identify unnecessary parameters.

The encoder's first layer parameters used in forward computation are relevant only to interacted items, resulting in many zero gradients. Thus, we reduce communication by selecting first-layer parameters and gradients based on these items. We also sample fake items to protect user interaction history.

For the decoder's last layer, let o_i represent the output for item i. Then the preference of user u for item i can be expressed as $P(i) = \frac{exp(o_i)}{\sum_{j \in I} exp(o_j)}$(for simplification, u is omitted here), where I denotes the set of all items. Let w denote the model's parameters, and we get: $\bigtriangledown_w logP(i) = \bigtriangledown_w o_i - \sum_{j \in I} P(j) \bigtriangledown_w o_j$, we can see that its forward computation and gradient involve all items. We aim to further reduce the amount of communication.

An alternative probability distribution, Q, known as the proposal distribution, can be used. By sampling from Q distribution, a sample set I^s is obtained from I, forming the candidate set $I^c = \{i\} \cup I^s$. Then, on the candidate set I^c, the softmax computation can be performed as:$P(i) = \frac{exp(o_i)}{\sum_{j \in I^c} exp(o_j)}$.

Inspired by work [3], FCIS first divides the feature space of item vector $\mathbf{e}$ into A subspaces, that is, $\mathbf{e} = \mathbf{e^1}||\mathbf{e^2}||...||\mathbf{e^A}$.Then, a codebook is built on each subspace. This operation is realized by K-Means clustering method of item vector. Notably, the codebook constructed in the a-th subspace is represented as $C^a = \{\mathbf{c}_1^a, \mathbf{c}_2^a, ..., \mathbf{c}_K^a\}$, where K is the total number of codewords in the codebook C^a. Then there is:

$$f(\mathbf{e}) = f^1(\mathbf{e}^1)||f^2(\mathbf{e}^2)||...||f^A(\mathbf{e}^A) = \mathbf{c}^1||\mathbf{c}^2||...||\mathbf{c}^A. \tag{2}$$

Set A to 2, then

$$f(\mathbf{e}_i) = \mathbf{c}^1_{\alpha(i)}||\mathbf{c}^2_{\beta(i)} = \gamma(\alpha(i), (\beta(i)) = \hat{\mathbf{e}}_i \in \Omega_{\alpha(i),(\beta(i)}, \tag{3}$$

in which $\alpha(i)$ and $\beta(i)$ represent the indices of the two codewords obtained by quantizing the item i's vector $\mathbf{e}_i$, and the concatenating results are represented $\gamma(\alpha(i), (\beta(i))$ and $\hat{\mathbf{e}}_i$, also, $\Omega_{\alpha(i),(\beta(i)}$ denotes the set consisting of items whose codeword indices are $\mathbf{c}^1$ and $\mathbf{c}^2$. To express it more clearly, a residual vector notation is added, $\tilde{\mathbf{e}}_i = \mathbf{e}_i - \hat{\mathbf{e}}_i$. Then the softmax probability distribution Q can be decomposed as follows:

$$\begin{aligned} Q(i|\mathbf{h}_u) =& \frac{exp(\mathbf{h}_u^T \mathbf{e}_i + b_i)}{\sum_{j \in I} exp(\mathbf{h}_u^T \mathbf{e}_j + b_j)} \\ =& \frac{exp(\mathbf{h}_u^T (\hat{\mathbf{e}}_i + \tilde{\mathbf{e}}_i) + b_i)}{\sum_{j \in I} exp(\mathbf{h}_u^T (\hat{\mathbf{e}}_j + \tilde{\mathbf{e}}_j) + b_j)} \\ =& \frac{exp(\mathbf{h}_u^T \hat{\mathbf{e}}_i) \cdot exp(\mathbf{h}_u^T \tilde{\mathbf{e}}_i + b_i)}{\sum_{m=1}^K \sum_{n=1}^K [exp(\mathbf{h}_u^T \gamma(m,n)) \cdot \sum_{j \in \Omega_{m,n}} exp(\mathbf{h}_u^T \tilde{\mathbf{e}}_j + b_j)]}, \end{aligned} \tag{4}$$

where h_u is the hidden vector w.r.t. the user u, and b_i is the bias about item i. For simplicity and approximation, note that $\tilde{\mathbf{e}}_i = 0, b_i = 0$, then we can deduce from (4):

$$\begin{aligned} Q(i|\mathbf{h}_u) \approx& \frac{exp(\mathbf{h}_u^T \hat{\mathbf{e}}_i)}{\sum_{m=1}^K \sum_{n=1}^K |\Omega_{m,n}| exp(\mathbf{h}_u^T \gamma(m,n))} \\ =& \frac{exp(\mathbf{h}_u^T \gamma(c^1, c^2))}{\sum_{m=1}^K \sum_{n=1}^K |\Omega_{m,n}| exp(\mathbf{h}_u^T \gamma(m,n))} \\ =& \frac{exp(\mathbf{h}_u^T \gamma(c^1, c^2)) |\Omega_{c^1,c^2}|}{\sum_{m=1}^K \sum_{n=1}^K |\Omega_{m,n}| exp(\mathbf{h}_u^T \gamma(m,n))} \cdot \frac{1}{|\Omega_{c^1,c^2}|} \\ =& p(c^1, c^2|\mathbf{h}_u) \cdot p(i|c^1, c^2, \mathbf{h}_u). \end{aligned} \tag{5}$$

Finally, the final simplified form are as follows:

$$p(c^1, c^2|\mathbf{h}_u) = \frac{exp(\mathbf{h}_u^T \gamma(c^1, c^2))|\Omega_{c^1,c^2}|}{\sum_{m=1}^{K}\sum_{n=1}^{K}|\Omega_{m,n}|exp(\mathbf{h}_u^T\gamma(m,n))}. \tag{6}$$

$$p(i|c^1, c^2, \mathbf{h}_u) = \frac{1}{|\Omega_{c^1,c^2}|}. \tag{7}$$

The result of this derivation can be understood as follows: user u first sample class (or codeword index) from the set $C_1 \times C_2$ with the probability $p(c^1, c^2|\mathbf{h}_u)$, and then server sample items belonging to this class with probability $p(i|c^1, c^2, \mathbf{h}_u)$. The sampling process is shown in Fig.1.

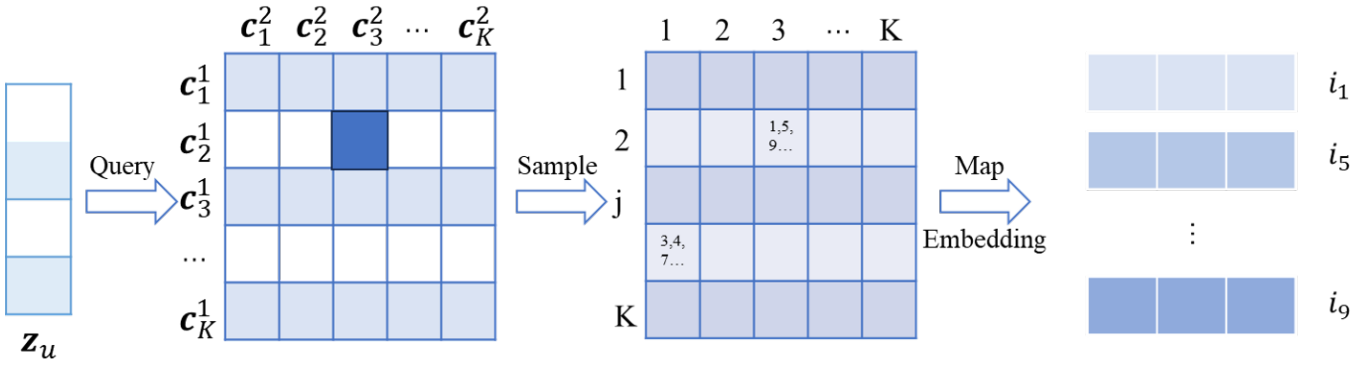

Figure 1. An illustration of sampling process

3.2 Efficient Federated Variation Autoencoder

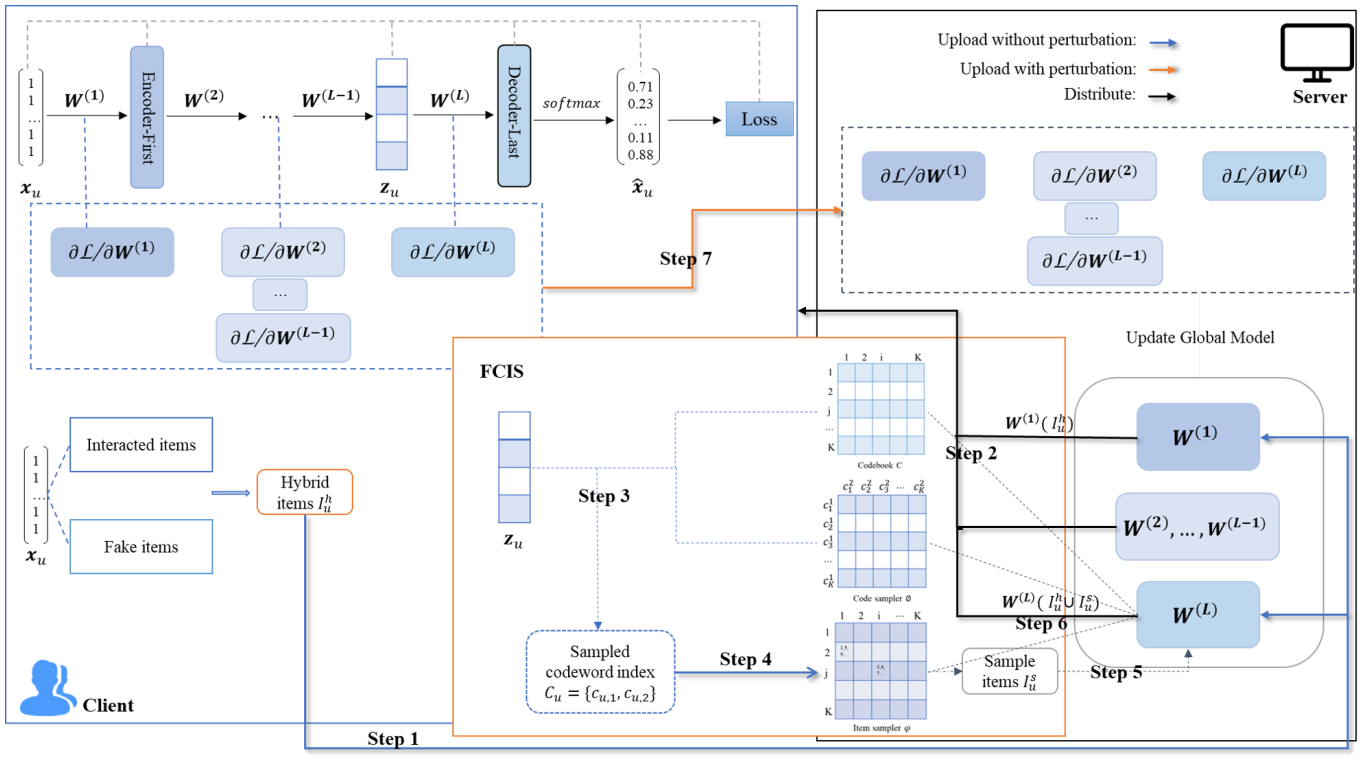

Figure 2. The overview of EFVAE

By introducing FCIS into FedVAE, EFVAE is proposed. As described in Fig.2, the process of EFVAE can be summarized in the following steps:

(1) The server initializes all parameters $\mathbf{W}$ and applies K-Means to each subspace of $\mathbf{W}^L$ separately to construct codebook C, code sampler ϕ and item sampler ψ.

(2) The server samples some clients $\mathbf{U}_t$ for training, then each selected user u samples $\rho|I_u|$ fake items from $I \setminus I_u$, forming a hybrid item set I_u^h, and uploads it to the server.

(3) The server sends corresponding first layer model parameters for I_u^h, other middle layers model parameters excluding the last layer along with codebook C and code sampler ϕ to the respective users.

(4) Each user performs forward computation and uploads a codeword index set C_u to the server according to FCIS.

(5) The server generates a sampled item set I_u^s for each user u on the uploaded C_u using item sampler ψ and then distributed corresponding last layer model parameters $\mathbf{W}^L(I_u^h \cup I_u^s)$ to client u.

(6) Each user u computes the local gradient and then uploaded perturbed gradients to the server using secret sharing techniques.

(7) The server aggregates all uploaded gradients to update the global model.

4 Experiment

In this section, the reproduction of the EFVAE model based on the original source code[1] is implemented, conducting an empirical investigation of its performance across three autoencoder architectures: Mult-DAE, Mult-VAE, and RecVAE. Furthermore, this study examines the impact of different item sampling strategies within the FCIS framework, comparing uniform sampling with popularity-based sampling for each of the aforementioned models.

4.1 Comparing with the released source codes

While the reproduction leverages the authors' publicly available code, several bugs were identified that hindered successful execution. Consequently, we implemented necessary modifications, which are primarily concentrated on the definition of the loss function and the gradient aggregation process guided by the original publication, to ensure correct compilation and runtime behavior.

[1] https://github.com/LukeZane118/EFVAE

4.2 Experimental environment setup

4.2.1 Datasets

Same as the setting of EFVAE, the reproduction uses LastFM[2], Citeulike-a[3], Film Trust[4] and Stream[5] datasets, the statistics of which are shown in Table 1. Users are divided into training, validation, and test sets. Models are trained using the complete positive feedback data of training users, while a portion of the positive feedback data (in EFVAE, 80%) from validation and test users is used for model learning, with the remainder used for evaluation.

Dataset	#User	#Item	#Interaction	Density	#Held-out
LastFM	1877	15270	90334	0.32%	200
Citeulike-a	5550	16865	210308	0.23%	800
FilmTrust	1119	1352	22952	1.50%	200
Stream	3757	4414	114149	0.69%	1000

Table 1. Dataset Statistics

4.2.2 Hyper-parameters Setting

Following EFVAE, the backbone architecture is $|I| \rightarrow 120 \rightarrow 40 \rightarrow 120 \rightarrow 40 \rightarrow |I|$, the model parameters are initialized using the Xavier method, using the Adam Optimizer, with the learning rate tuned among $\{0.01, 0.005, 0.002, 0.001, 0.0005, 0.0002, 0.0001\}$, the total number of codewords K is searched within $\{8, 16, 32, 64, 128\}$, and the sampling ratio (i.e., the number of sampled items as a percentage of the total number of items) is explored among $\{0.05, 1, 2, 5, 10, 20\}$. The ratio of fake items ρ is set to 1, and the number of shares for secret sharing is set to 3.

4.2.3 Metrics

EFVAE evaluates recommendation performance choosing Recall@k and NDCG@k (in paper, k=5), additionally, EFVAE explores efficiency selecting communication cost which is assessed by measuring the mean data volume exchanged by each client during each training round, encompassing both uploads and downloads.

4.3 Results and Analysis

In Table 2 and Table 3, we report the model performance of different models on four datasets. We can observe that:

- Our experiments, replicating the original work across four datasets, yielded results consistent with those presented by the authors. This consistency underscores the effectiveness of EFVAE in achieving satis-

[2]https://grouplens.org/datasets/hetrec-2011/
[3]https://github.com/js05212/citeulike-a
[4]https://www.kaggle.com/datasets/abdelhakaissat/film-trust
[5]https://www.kaggle.com/datasets/tamber/steam-video-games

factory recommendation performance coupled with a notable improvement in communication efficiency, attributed to the considerably reduced communication overhead.

- Comparing uniform and popularity-based sampling showed no clear performance winner. While popularity-based sampling performed slightly better on FilmTrust, the opposite was true for Citeu-like. Generally, performance was comparable, but popularity-based sampling's higher computational cost and communication overhead make uniform sampling a more efficient choice.
- EFMult-VAE generally outperforms EFMult-DAE, except possibly for the most active users. EFRec-VAE, while achieving better performance due to its composite prior, has higher communication costs due to storing previous encoder parameters.

	Lastfm			Citeu-like		
	Re@5	NDCG@5	comm.cost	Re@5	NDCG@5	comm.cost
efmult-dae-Uni	0.1461	0.2844	6.9119MB	0.1174	0.1626	8.4327MB
efmult-dae-Pop	0.1423	0.2778	7.4224MB	0.1113	0.1603	8.6281MB
efmult-vae-Uni	0.1451	0.274	6.5442MB	0.1244	0.1706	8.3159MB
efmult-vae-Pop	0.1419	0.2755	6.9368MB	0.1153	0.1653	8.4954MB
efrecvae-Uni	0.1547	0.2894	12.6562MB	0.1374	0.1892	13.6366MB
efrecvae-Pop	0.1534	0.2847	12.7795MB	0.1307	0.1898	13.9339MB

Table 2. Performance comparison of different models (Part 1: Lastfm and Citeu-like)

	filmtrust			steam		
	Re@5	NDCG@5	comm.cost	Re@5	NDCG@5	comm.cost
efmult-dae-Uni	0.3409	0.3139	1.3271MB	0.3121	0.3335	2.5809MB
efmult-dae-Pop	0.3378	0.3099	1.3233MB	0.3252	0.3418	2.5922MB
efmult-vae-Uni	0.3414	0.3383	1.2537MB	0.3176	0.3309	2.4004MB
efmult-vae-Pop	0.3379	0.3225	1.2541MB	0.3222	0.3422	2.4824MB
efrecvae-Uni	0.3572	0.3473	2.1668MB	0.3459	0.3549	4.3433MB
efrecvae-Pop	0.3513	0.3495	2.1483MB	0.3449	0.354	4.4266MB

Table 3. Performance comparison of different models (Part 2: filmtrust and steam)

5 Conclusion and future work

In this paper, we reproduce an Efficient Federated Variational AutoEncoder framework for recommendation, termed EFVAE. It explores the communication cost bottleneck of FedVAE and then reduces communication costs through a client-to-server collaborative sampling mechanism and satisfactory recommendation

performance with dynamic multi-stage approximation of the decoding distribution. Extensive experiments on four public datasets confirm the effectiveness and efficiency of EFVAE.Considering that federated autoencoder recommender systems can provide services to non-participating users in training, they are likely more suitable for federated recommendation scenarios, offering better privacy-preserving recommendation services to a broader range of users.

While EFVAE employs average communication cost as a metric for communication overhead, it is crucial to acknowledge the interplay between communication cost and model convergence. This observation motivates the consideration of both the total number of training iterations and the average communication cost when assessing communication efficiency. Additionally, the item sampling strategy in FCIS, which utilizes a clustering algorithm, employs cluster centers instead of true item embeddings for codeword selection. This approximation may introduce a degree of information loss. Consequently, this motivates our investigation into alternative, more lossless approaches to mitigating communication costs.

Conversely, the practical deployment of VAE-based federated recommendation systems in more intricate real-world settings encounters a range of significant challenges. In contrast to controlled experimental environments, real-world data is characterized by increased scale, greater distributional complexity, heightened privacy concerns, and more demanding real-time performance requirements. For instance, within e-commerce and social media ecosystems, users often demonstrate heterogeneous privacy preferences. While some users may exhibit a lower sensitivity to privacy, potentially accepting some trade-offs for enhanced recommendation quality, others place paramount importance on data protection. This heterogeneity necessitates the storage of sensitive user information across both server and client infrastructures. A critical challenge thus lies in how platforms can effectively accommodate this diversity of privacy needs, coupled with the increased complexity of data distribution across edge devices and cloud infrastructure, while simultaneously delivering superior recommendation experiences to users with lower privacy concerns and rigorously safeguarding the data of privacy-sensitive users. Moreover, the inherent dynamism and temporal sensitivity of user behavior data necessitate real-time model updates to ensure rapid adaptation to evolving user interactions.

References

[1] Muhammad Ammad-Ud-Din, Elena Ivannikova, Suleiman A Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888*, 2019.

[2] Di Chai, Leye Wang, Kai Chen, and Qiang Yang. Secure federated matrix factorization. *IEEE Intelligent Systems*, 36(5):11–20, 2020.

[3] Jin Chen, Defu Lian, Binbin Jin, Xu Huang, Kai Zheng, and Enhong Chen. Fast variational autoencoder with inverted multi-index for collaborative filtering. In *Proceedings of the ACM Web Conference 2022*, pages 1944–1954, 2022.

[4] Yucheng Chen, Chenyuan Feng, and Daquan Feng. Privacy-preserving hierarchical federated recommendation systems. *IEEE Communications Letters*, 27(5):1312–1316, 2023.

[5] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.

[6] Adrian Flanagan, Were Oyomno, Alexander Grigorievskiy, Kuan E Tan, Suleiman A Khan, and Muhammad Ammad-Ud-Din. Federated multi-view matrix factorization for personalized recommendations. In *Machine learning and knowledge discovery in databases: European conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part II*, pages 324–347. Springer, 2021.

[7] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[8] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*, pages 689–698, 2018.

[9] Yujie Lin, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Dongxiao Yu, Jun Ma, Maarten de Rijke, and Xiuzhen Cheng. Meta matrix factorization for federated rating predictions. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 981–990, 2020.

[10] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[11] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. Fedfast: Going beyond average for faster training of federated recommender systems. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1234–1242, 2020.

[12] Vasileios Perifanis, George Drosatos, Giorgos Stamatelatos, and Pavlos S Efraimidis. Fedpoirec: Privacy-preserving federated poi recommendation with social influence. *Information Sciences*, 623:767–790, 2023.

[13] Vasileios Perifanis and Pavlos S Efraimidis. Federated neural collaborative filtering. *Knowledge-Based Systems*, 242:108441, 2022.

[14] Mirko Polato. Federated variational autoencoder for collaborative filtering. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[15] Tao Qi, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. Privacy-preserving news recommendation model learning. *arXiv preprint arXiv:2003.09592*, 2020.

[16] Liang Qu, Ningzhi Tang, Ruiqi Zheng, Quoc Viet Hung Nguyen, Zi Huang, Yuhui Shi, and Hongzhi Yin. Semi-decentralized federated ego graph learning for recommendation. In *Proceedings of the ACM Web Conference 2023*, pages 339–348, 2023.

[17] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[18] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I Nikolenko. Recvae: A new variational autoencoder for top-n recommendations with implicit feedback. In *Proceedings of the 13th international conference on web search and data mining*, pages 528–536, 2020.

[19] Senci Ying. Shared mf: A privacy-preserving recommendation system. *arXiv preprint arXiv:2008.07759*, 2020.

[20] Sergej Zerr, Stefan Siersdorfer, Jonathon Hare, and Elena Demidova. Privacy-aware image classification and search. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, page 35–44, New York, NY, USA, 2012. Association for Computing Machinery.

[21] Lu Zhang, Guohui Li, Ling Yuan, Xuanang Ding, and Qian Rong. Hn3s: A federated autoencoder framework for collaborative filtering via hybrid negative sampling and secret sharing. *Information Processing & Management*, 61(2):103580, 2024.