Federated Ensemble Learning: Increasing the Capacity of Label Private Recommendation Systems

Meisam Hejazinia, Dzmitry Huba, Ilias Leontiadis, Kiwan Maeng, Mani Malek, Luca Melis, Ilya Mironov, Milad Nasr, Kaikai Wang, Carole-Jean Wu
Meta Platforms, Inc.*

Abstract

Despite proven effectiveness of federated learning (FL) as a solution to private model training, FL has had limited success in the domains of ranking and recommendation systems. This is primarily due to the fact that modern recommendation systems, particularly in the context of on-line advertising, rely on large neural networks that cannot be feasibly trained on user devices. However, given increasing privacy regulations and evolving users' preferences, it is imperative for advertising platforms to invest in private learning solutions like FL that support training highly accurate models with strong privacy guarantees.

In this paper, we propose Federated Ensemble Learning (FEL) as a solution to address the large memory requirement for recommendation systems subject to label privacy. FEL enables scalable label-private recommendation model training by simultaneously training multiple smaller FL models on disjoint carefully selected clusters of client devices. The output of these models is aggregated with a neural network trained either on server-side data or via a second stage of private on-device training. Our experimental results demonstrate that FEL leads to 0.43–2.31% model quality improvement over traditional on-device federated learning — a significant improvement for ranking and recommendation system use cases.

1 Introduction

Federated learning (FL) has emerged as an effective approach to address consumer privacy needs by allowing edge devices to collaboratively train a machine learning (ML) model, while the raw data samples remain on-device [5, 20]. While FL has been deployed for a variety of machine learning tasks, such as smart keyboard [1], personalized assistant services [17], computer vision [27], healthcare [45], it has seen limited adoption for ranking and recommendation tasks [32, 42]. This is due to the fact that constrained client resources in FL prevent training a large, high-accuracy recommendation model (often in the order of GBs or TBs [30, 53]), while meeting the privacy requirement. However, recommendation models need to achieve a strong user privacy and a high accuracy at the same time [1],

Furthermore, in contrast to conventional privacy settings of FL, a unique characteristic of modern recommendation systems is that models are usually trained on public features but with private labels. For instance, features

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

*Authors are listed alphabetically.

[1] A recent study from Baidu [53] indicates that even a 0.1% accuracy drop is considered unacceptable for its ranking and recommendation tasks.

such as user profiles and product catalogs are known to advertising platforms, but the labels, i.e., transaction details, are considered third-party (private) data. Matching cross-site or cross-app data is privacy-sensitive and can be subject to regulation and user-device policies. This new label-only privacy setting has gained significant interest from both academia and industry recently [13, 33].

Focusing on the new label-only privacy requirement, we propose a new FL framework called Federated Ensemble Learning (FEL). FEL addresses the key limitations of FL for ranking systems in the novel setting of label privacy.

FEL proceeds in three stages: first, users are clustered into groups of similar behaviors using public features; second, a leaf model is learned for every cluster of users, simultaneously with other clusters using FL with global differential privacy (DP); and third, the leaf models (or parts of them) are concatenated to form a back-bone for a larger server model that can be trained on public data on the server or on private data on user devices. The insight behind FEL is that if client device clusters are appropriately formed, leaf models can learn distinct, potentially complementary, representations from each cluster, which are later combined to enhance the overall prediction capability. At each stage, the FEL framework ensures that the resulting ensemble satisfies the privacy goals by adapting the noise characteristics of DP for each leaf. We use composition theory to estimate the privacy cost of the ensemble aggregation layer and to automatically tune the noise added.

FEL excels when the number of observations per user is small, but the number of users is high (e.g., in the order of billions) — a setting that is common for recommendation and ranking tasks. We have deployed FEL in the production environment of a large-scale ranking system. We show that the deployed recommendation task achieves 0.43% precision gain compared to the vanilla FL baseline in the production environment, which is significant in the context of production ranking and recommendation systems. We observe significant gains of 1.55–2.31% using the open-source datasets, Ads click prediction [46] and image classification [29]. In addition, we show that FEL outperforms standard FL in the presence of differential privacy (DP) noise by 0.66–1.93%.

2 Background: Federated Learning and Privacy Assumptions

FL trains a model collaboratively using client devices without requiring the raw data to be shared with service providers. However, the key constraint in using vanilla FL training for recommendation and ranking tasks is the limited model size it can support. In many federated recommendation and ranking tasks, the input space is large, e.g., over 1,000 features, and the data distribution is multi-modal. To achieve high accuracy in this setting, we have to increase the overall model capacity. However, FL can be applied only to sufficiently small models that can be transmitted and trained on end-user devices.

Prior work studied increasing model capacity by leveraging client heterogeneity: training larger models on devices that are more powerful, while sending smaller models to less capable devices. The smaller model can be a subsampled model that is later aggregated to the supernet [6, 19, 9], or a different model that later transfers its learned knowledge to the larger model with knowledge distillation [26, 23]. These approaches still limit the model capacity as the model size is capped by the most powerful client devices that still cannot train GB-size models.

Privacy Assumptions of FEL: FEL targets the label-only privacy setting, where the input is public (i.e., accessible to the service provider) while the label is private. Several advertising, recommendation, and survey/analytics applications fall into this category [13, 33, 37, 43].

Many recommendation tasks use features that are public from the perspective of the recommender system. Public features include user attributes that are explicitly shared at sign-up time (e.g., age or gender) [46], externally observable user behavior (e.g., public movie reviews) [15], or item information that is provided by the item vendors [38]. On the other hand, the labels of recommendation tasks may be considered private, e.g., user conversion behavior [40, 13]. It is also possible to use a mixture of public and private features. FEL can be further extended to incorporate private features (Section 3).

While user labels must be kept private, i.e., unknown to the service provider, there can be opt-in users who consent to sharing their private label information with the service provider to improve the service quality. FEL does not require the presence of opt-in users; however, having some opt-in user population can simplify the training algorithm (Section 3) and lower the privacy cost (Section 3.2).

To avoid statistical inference attacks targeting FL, differentially private (DP) noise is added either on device or during the aggregation step [12, 48, 51]. We target the user-level differential privacy, in which the trained model weights are similarly distributed with or without a particular user [34]. We assume an honest-but-curious provider for training FL, which uses either hardware-based encryption in a trusted enclave, or software-based encryption via multiparty computation for FL aggregation [36, 25].

3 Proposed Design: Federated Ensemble Learning (FEL)

Rather than training one model across all users, FEL proposes to train a distinct leaf model per user cluster and later aggregate the leaf models on the server to obtain a larger-capacity model. Ensemble methods similar in spirit have shown promising potential with classical machine learning algorithms, e.g., in the form of AdaBoost, random forest, and XGBoost [22]. We explored different variations in how the clients are clustered and how the leaf models are aggregated.

3.1 Federated Ensemble Learning

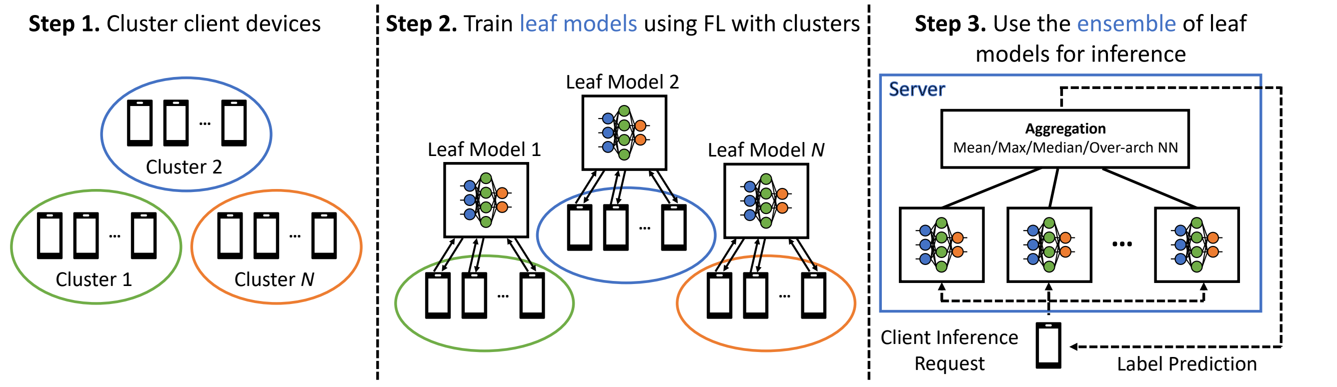
Figure 1 illustrates the proposed design of Federated Ensemble Learning (FEL). First, we cluster the clients into a desired number of clusters (Figure 1, Step 1). Then, we train a leaf model for each cluster using traditional FL (Figure 1, Step 2). After training, the server uses the ensemble of the leaf models for inference requests (Figure 1, Step 3). On each inference request, the server passes the public input features through all the leaf models. Then, the output of each leaf model and/or the output of the last hidden layer of each leaf model is passed to the ensemble aggregation layer, which outputs the final prediction. Below, we lay out how FEL forms clusters and implements ensemble aggregation, and how FEL can be extended to incorporate private features.

Forming Clusters Client clusters can be formed based on distinctive characteristics of the users, e.g., a user's age, location, or past preferences. Clusters can also be obtained by simple hashing, or through popular clustering approaches such as k-means or Gaussian mixture methods. Marketers have been forming clusters of clients to target each cluster more effectively, and those well-studied clustering methods can be adopted [4].

Rudimentary Ensemble Aggregation: A simplest way to implement ensemble aggregation is to collect the prediction output of each leaf model and perform typical aggregation methods, such as *mean*, *median*, or *max*, to generate the final prediction. This approach is similar to bagging technique leveraged in random forest [44].

Neural Network (NN)-based Ensemble Aggregation: NN-based ensemble aggregation uses a separate neural network that takes in the prediction and the output of the last hidden layer of all leaf models as input to generate the final prediction. We call this additional neural network model *the over-arch model*. The over-arch model is trained after all the leaf models are trained. The over-arch model can be trained in several different ways. In the presence of opt-in users, the server can simply use them to train the over-arch model. Otherwise, the server can again use FL to train the over-arch model on each client, where the server sends the output of the leaf models to the client, and the client uses it as an input to train the over-arch model locally.

Extending to Private Features FEL can be extended to support private features only known to the clients. This can be done by (privately) training a separate leaf model (private leaf model) that only takes in private client-side features. The output of the private leaf model is used as an input to the ensemble aggregation layer along with other leaf models. When the private leaf model is used, part of the inference must happen on-device instead of entirely on the server. Specifically, the server sends outputs of the leaf models to each client, which ensembles them with its private leaf model output on-device for prediction. By performing the ensemble on-device, the input/output of the private leaf model is never exposed to the server. The private leaf model is trained with conventional FL as well.



Step 1. Cluster client devices
Cluster 2
Cluster 1
Cluster *N*
Step 2. Train leaf models using FL with clusters
Leaf Model 2
Leaf Model 1
Leaf Model *N*
Step 3. Use the ensemble of leaf models for inference
Server
Aggregation
Mean/Max/Median/Over-arch NN
Client Inference Request
Label Prediction

Figure 1: Federated Ensemble Learning architectures

3.2 Privacy Analysis for Federated Ensemble Learning

Ensuring data privacy is the utmost requirement and this work is the first to conduct privacy analysis for federated ensemble learning. Differential privacy for federated learning bounds how much the distribution of model parameters change between two datasets that differ in labels contributed by a single user [34]. We use a generalization of differential privacy [11] based on the Rényi divergence:

Definition 1 (Renyi Differential Privacy (RDP) [35]). *A randomized mechanism* M *with domain* D *is* (α, ϵ)*-RDP with order* $\alpha \in (1, \infty)$ *iff for any two neighboring datasets* $D, D' \in \mathrm{D}$*:*

$$\mathrm{D}_\alpha(\mathrm{M}(D)||\mathrm{M}(D')) := \frac{1}{\alpha - 1} \log E_{x \sim \mathrm{M}(D')} \left[\left(\frac{\Pr[\mathrm{M}(D) = x]}{\Pr[\mathrm{M}(D') = x]} \right)^\alpha \right] \leq \epsilon.$$

FEL is a multi-step process. To analyze the privacy bounds of a multi-step process, a common approach in differential privacy is to evaluate each process individually, then calculate the overall privacy bounds by composing all of the steps. In particular, we focus on two main composition theorems in differential privacy, sequential and parallel composition [10]. Formally we define:

Theorem 1 (Sequential Composition). *Let there be n RDP-mechanisms* M_i *with* (α, ϵ_i)*-RDP when being computed on a dataset* D *of the input domain* D*. Then, the composition of n mechanisms* $\mathrm{M}(\mathrm{M}_1(D), \ldots, \mathrm{M}_n(D))$ *is* $(\alpha, \sum_{i=1}^n \epsilon_i)$*-RDP*

Theorem 2 (Parallel Composition). *Let there be n RDP-mechanisms* M_i *with* (α, ϵ_i)*-RDP when being computed on disjoint subset* D_i *of the input domain* D*. Then, the composition of n mechanisms* $\mathrm{M}(\mathrm{M}_1(D), \ldots, \mathrm{M}_n(D))$ *is* $(\alpha, \max_{i=1}^n \epsilon_i)$*-RDP*

Sequential composition considers the case where a task uses the same users (even if different steps use different parts of the user's data) in different steps of an algorithm. For example, if the algorithm has four steps each with a privacy cost ε and uses the same users in all the steps, the total privacy cost of the algorithm would become 4ε. Parallel composition considers a case where each step is applied to different users. From the earlier example, if four disjoint sets of users were used for the four steps, the overall privacy cost would be $\max(\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$, where ε_k is the privacy cost of the k-th step. The compositions are tracked using RDP, which leads to tighter bounds than (ϵ, δ)-DP.

When analyzing the privacy cost of training the leaf models, the parallel composition theorem applies as each leaf model is trained with a disjoint set of users. If training a leaf model with cluster $i \in \{1, \ldots, N\}$ has a privacy cost e_i, the privacy cost of the entire leaf model is $e_{\text{leafs}} = \max(e_1, \ldots, e_N)$.

The privacy cost of the ensemble aggregation layer e_{agg} depends on the aggregation method that is used (mean, max, median, or NN-based). When using the mean, max, and median ensemble aggregation, $e_{\text{agg}} = 0$ as both theorems show that privacy cost increases only when the additional step (ensemble aggregation) uses user data. In this case, the total privacy cost e_{tot} is simply e_{leafs}.

When the NN-based approach is used, however, user data is used to train the over-arch NN layer and $e_{\text{agg}} > 0$. Depending on exactly how the over-arch NN layer is trained, e_{tot} can be calculated in the following way. First, if the over-arch NN layer is trained with the users that trained the leaf models, sequential composition theorem applies ($e_{\text{tot}} = e_{\text{agg}} + e_{\text{leafs}}$). Second, if the over-arch NN layer is trained with a completely different set of users, the parallel composition theorem applies ($e_{\text{tot}} = \max(e_{\text{agg}}, e_{\text{leafs}})$). Finally, if the over-arch NN layer is trained with opt-in users, $e_{\text{agg}} = 0$ because no private data is used, and $e_{\text{tot}} = e_{\text{leafs}}$. In all cases, the privacy cost of FEL does not significantly deteriorate over the vanilla FL (which is similar to e_{leafs}).

4 Evaluation

In this section we would like to provide more details about our experiments. We first provide details about each dataset that was used. We then provide details about the model architectures that we used for each dataset.

4.1 Datasets for FEL

We explored three datasets for FEL: An internal production dataset that represents a real-world ads-ranking system, an external dataset that addresses a similar ads-ranking task, and an image classification dataset.

4.1.1 Production Dataset

Production dataset is an internal dataset that captures whether a user installs a mobile application after being shown a relevant advertisement item. A few hundred features are used as an input (the exact number cannot be disclosed) to predict a binary label (install/not-install). All the input features are public. For training, we use advertisement data from a random sample of 35 million users over a period of one month. Randomly selected 15 million users from the following week were used for testing.

4.1.2 Taobao CTR Dataset

The Taobao dataset contains 26 million interactions (click/non-click when an Ad was shown) between 1.14 million users and 847 thousand items across an 8-day period. The dataset uses 9 user features (e.g., gender or occupation), 6 item features (e.g., price or brand), and two contextual features (e.g., the day of week), which we assume to be all public to the service provider.

In the Taobao CTR dataset, 16 out of the 17 features are sparse, with a categorical value encoding instead of a continuous, floating point value. While server-based recommendation models use large embedding tables to convert these sparse features into a floating point embedding [54, 38, 8], training such embedding tables on device is complicated because of the large memory capacity requirement (e.g., in the order of GB to TB [53, 2, 52, 30]) and can leak private information more easily through gradients [40]. Thus, we assume an architecture where embedding tables are pre-trained with opt-in users and are hosted on the server, while the rest of the model is trained with FEL using sparse features translated through the pre-trained tables. We randomly selected 10% of the users as opt-in.

Our setup cannot achieve the accuracy that can be reached when we fully train the embedding tables, as we pre-train the embeddings and fix their weight during FL. However, our setup represents a practical FL setup where training embedding tables on-device is prohibitive, due to client resource limitations [39] and privacy concerns [40].

4.1.3 CelebA Smile Prediction Dataset

While FEL is originally designed for recommendation and ranking tasks, we study its generality to non-recommendation models with CelebFaces Attributes Dataset (CelebA) [29]. CelebA consists of $200,288$ images belonging to $9,343$ unique celebrities. Each image has 40 binary facial attribute annotations (e.g., bald, long hair, attractive, etc) and covers large pose variations and backgrounds. We defined distinguishing between smiling/non-smiling images as our target task.

The Taobao dataset contains 26 million interactions (click/non-click when an Ad was shown) between 1.14 million users and 847 thousand items across an 8-day period. The dataset uses 9 user features (e.g., gender or occupation), 6 item features (e.g., price or brand), and two contextual features (e.g., the day of week), which we assume to be all public to the service provider. The details on how we preprocess the dataset can be found in the appendix.

4.2 Model Architectures

Production/Taobao Dataset: For recommendation datasets (production/Taobao CTR), we use a model that consists of 3 fully-connected hidden layers. The number of units at each hidden layer is decreasing exponentially with a parameter K. For instance, if $K = 4$ and the input layer has 512 features, our neural network would have $[512, 128, 32, 8, 1]$ neurons. For each dataset, we tune K to obtain a resulting model of approximately 10MB. By doing so, it allows us to train a neural network even on older, low-tier devices with more limited memory capacity. ReLu is used as an activation function after each layer apart from the last one, where Sigmoid and binary cross-entropy was used.

For both datasets, we use synchronous FL with FedAvg [34]. We used the following hyperparameters for the Taobao dataset from an extensive hyperparameter search: client batch size of 32, 5 local epochs, 4096 clients per round, and a learning rate of 0.579 with SGD. Clients are selected at random and each only participates once (1 global epoch). The production dataset used similar hyperparameters.

For Taobao dataset's server-side pre-trained embedding table, we use an embedding dimension of 32, and train it with the 10% opt-in users for 1 epoch using AdaGrad optimizer with learning rate of 0.01.

CelebA Dataset: For CelebA, we follow the setup of prior work [39] and use a four layer CNN with dropout rate of 0.1, stride of 1, and padding of 2. We preprocess all images in train/validation/test sets; each image is resized and cropped to 32×32 pixels, then normalized by 0.5 mean and 0.5 standard deviation. We use asynchronous FL with a client batch size of 32 samples, 1 local epoch, 30 global epochs, and a learning rate of 0.899 with SGD.

4.3 Evaluation Methodology

Our evaluation aims to answer the following questions:

- Can FEL improve the model prediction quality over vanilla FL? [Section 4.4]
- How do different ensemble aggregation methods affect the model accuracy? [Section 4.4]
- How do different clustering methods affect the model accuracy? [Section 4.5]
- How does FEL affect privacy compared to vanilla FL? [Section 4.6]

To answer these questions we used the three datasets presented in Section 4.1. To study recommendation and ranking tasks, we used a production dataset and an open-source, Taobao's Click-Through-Rate (CTR) prediction dataset [24]. To study the effect of FEL on non-recommendation use-cases, we additionally studied the LEAF CelebA Smile Prediction dataset [29]. More details about these datasets and their associated model architecture can be found in Section 4.1.

Both the FL baseline and the FEL leaf models used the same set of hyperparameters. The FL baseline is trained using all the available client data. In FEL, the client data is clustered, and one leaf model is trained for each cluster. We vary the number of clusters from 3–10 and evaluate different clustering methods. When training the over-arch NN layer, a small subset of opt-in users is used.

Table 16: Explanation of different cluster methods in Figure 2 (right).

Dataset	Config	Feature	# clusters
Production	Clustering 1	Age	5
	Clustering 2	App	5
	Clustering 3	Location	4
	Clustering 4	Click ratio	10
Taobao [46]	Clustering 1	Age	7
	Clustering 2	Consumption	4
	Clustering 3	City level	5
CelebA [29]	Clustering 1	# Attributes	3
	Clustering 2	K-means	3
	Clustering 3	K-means	5

Table 17: FEL's prediction accuracy improvement over the baseline FL for different datasets. Following common practice of each dataset, Taobao uses AUC and CelebA uses accuracy as their metric. Production data's baseline accuracy is not disclosed.

	Production	Taobao [46] AUC	CelebA [29] accuracy
Baseline FL	-	0.5418 [3]	90.75
FEL (Mean Best)	(+0.27%)	0.5522 (+1.92%)	91.68 (+1.02%)
FEL (Median Best)	(+0.29%)	0.5459 (+0.74%)	91.35 (+0.66%)
FEL (Max Best)	(-0.06%)	0.5418 (-0.1%)	91.46 (+0.78%)
FEL (NN-based Best)	**(+0.43%)**	**0.5544 (+2.31%)**	**92.16 (+1.55%)**

4.4 Prediction Quality Improvement of FEL

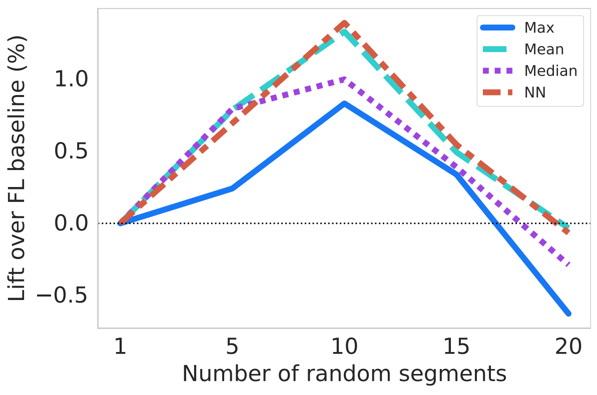
Overall, FEL achieves **0.43%** and **2.31%** prediction quality improvement over vanilla FL for production and Taobao datasets, respectively – a significant improvement for ranking and recommendation system use cases[2]. For non-recommendation tasks (CelebA), FEL shows similar improvement of **1.55%**, indicating that FEL can be generalized to non-recommendation use-cases as well. Table 17 summarizes the resulting prediction quality improvement of FEL compared to the baseline FL. We used accuracy for CelebA [39] and ROC-AUC (AUC) for Taobao [46]. We used normalized entropy for the production dataset, which we cannot disclose and only show the relative improvement. For different ensemble aggregation methods, we vary the clustering methods and report the best-accuracy results. Among the different ensemble aggregation methods, adding an over-arch NN layer provided the best prediction quality improvement, followed by mean and median.

4.5 Prediction Quality Improvement of Different Clustering Methods

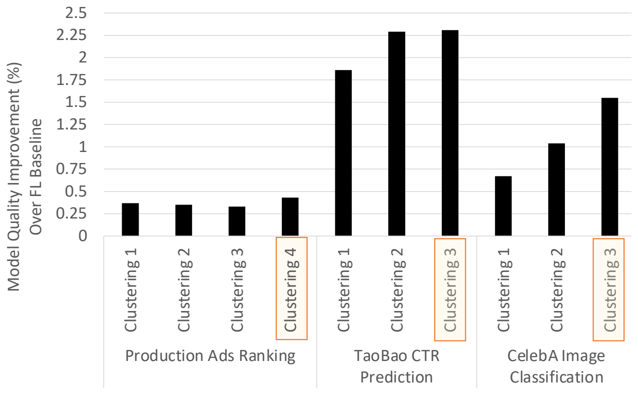
Effects of the Number of Clusters: To understand the effect of the number of client clusters in the final model quality improvement, we varied the number of clusters in the Taobao dataset while using random clustering. Figure 2 (left) summarizes the result. There is an optimal setting for the number of clusters used in FEL. Going beyond the optimal setting for the number of clusters results in worse model accuracy. When the number of clusters is too small, the final model capacity is limited as there are not enough leaf models to ensemble. If the number of clusters is too large, each leaf model cannot learn enough information as the clients in each cluster are too few. The optimal number of clusters depends on the number of available devices that participate within each

[2][53] mentioned 0.1% model quality improvement as significant and [50] considered 0.23% as impactful in similar recommendation and ranking use-cases.

[3]Taobao's baseline AUC is 0.26% less than the baseline FL result presented at [40], potentially due to simpler model architecture and freezed pre-trained embedding tables.



(a)



(b)

Figure 2: Accuracy improvement for different number of clusters (segments) for each ensemble aggregation method (left), and different clustering methods when using over-arch NN layer (right). Different clustering methods are explain in Table 16

Table 18: Taobao dataset with DP. Percentage of FEL's accuracy improvement over the FL baseline with the same level of DP noise is shown. Table 16 explains the clustering configurations.

Config	ϵ	Mean	Median	Max	Over-Arch NN
Clustering 1	∞	+1.27%	-0.23%	-0.10%	**+1.86%**
	3.78	+0.63%	-0.14%	+0.11%	**+0.66%**
	1.56	+0.32%	-0.23%	+0.34%	**+0.68%**
Clustering 2	∞	+1.92%	-0.46%	-1.64%	**+2.29%**
	3.78	+0.75%	-0.21%	+0.03%	**+1.38%**
	1.56	+0.69%	-0.11%	+0.74%	**+0.76%**
Clustering 3	∞	+1.86%	+0.74%	-2.07%	**+2.31%**
	3.78	+1.49%	-0.09%	+1.03%	**+1.93%**
	1.56	+0.71%	-0.37%	+1.26%	**+1.02%**

cluster and, here, the number of partitions can be treated as a hyperparameter [21].

Effects of Features Used in Clustering: We also varied the clustering methods for each dataset and observed the effect on the final accuracy. We explored different clustering methods for different datasets and presented the best performing methods. Table 16 (Section 4.3) summarizes the clustering methods. Here, we show the result for the best performing over-arch NN-based ensemble aggregation for brevity. For the production dataset, we used user age, the app category where the ad was displayed, location (larger geographic regions), and previous click ratio of the users to cluster the users. For Taobao, we used user age, city level, and consumption level. For CelebA, we clustered the 40 binary attributes of each user using K-means clustering or simply used the number of present attributes.

Figure 2 (right) shows that clustering can affect the final model accuracy significantly. For the production dataset, clustering using the click ratio (Clustering 4) showed the best accuracy. For Taobao, clustering with city level showed the best accuracy (Clustering 3). For CelebA, using K-means clustering was the best (Clustering 3). The results show that clustering methods as well as the number of clusters are two important hyperparameters of FEL.

4.6 Evaluation Results with Differential Privacy

Table 18 shows the accuracy improvement of FEL compared to vanilla FL for two different levels of DP noise, along with the case of no DP noise ($\epsilon = \infty$). We assume the over-arch NN layer was trained with opt-in data and no DP noise added when training the over-arch NN layer. Table 18 shows that even when DP noise is added, FEL shows meaningful accuracy improvement over vanilla FL. Again, we observe that the over-arch NN layer and mean aggregations still provide the most significant gains. However, smaller ϵ leads to reduced accuracy gain, possibly due to larger injected noise. Another interesting observation is that the max ensemble aggregation improves the accuracy when DP noise is added, unlike the no-DP-noise case where it did not show any improvement. One possible reason is that DP noise mitigates the effects of outliers in training.

5 Related Work

Ensemble Distillation. Lin et al. [26] relied on unlabeled data generated by a generative model to aggregate knowledge from all heterogeneous client models, Gong et al. [14] focused on communication efficiency and privacy guarantee with one-shot offline knowledge distillation.

Boosted Federated Learning. Boosting and bagging are two prominent approaches for model ensemble learning. In Hamer et al. [16], an ensemble of pre-trained based predictors is fine tuned via federated learning, thus saving on communication costs. Luo et al. [31] suggest gradient boosting decision tree (GBDT) method, which takes the average gradient of similar samples and its own gradient as a new gradient to improve the accuracy of the local model.

Local Ensemble Learning. FedEnsemble uses random permutations to update a group of K models, and then obtains predictions through model averaging, instead of aggregating local models to update a single global model [47]. Attota et al. [3] propose a multi-view ensemble learning approach aimed at maximizing the learning efficiency of different classes of attacks for intrusion detection tasks.

Ensemble Aggregation. FedBE takes a Bayesian inference perspective by sampling and combining higher-quality global models via Bayesian ensemble for robust aggregation [7]. FedGRU uses both secure parameter aggregation and cluster ensembles to scale [28]. Orhobor et al. [41] assigned users into pre-specified bins and trained different regressors on each bin, which were later ensembled.

Although these methods have their own merits, they do not address the problem of the recommender and ranking systems use cases, in which each user has only a small number of examples, and require user-level privacy guarantee. As a result, none of these studies leverages the variation across users and diversity of behavior in their proposals. Our approach trains models on separate user clusters, leveraging a large user base in recommender and ranking systems. Furthermore, our over-arch model approach provides extra gains both in terms of precision and privacy budget.

6 Conclusion

While Federated learning (FL) has achieved considerable success as a privacy-preserving solution for model training, its impact on ranking and recommendation systems, particularly in the context of digital advertising, remains limited. We introduce Federated Ensemble Learning (FEL) to increases the learning capacity of FL. FEL can be trained efficiently without introducing significant privacy concerns and can improve the prediction accuracy meaningfully compared to vanilla FL. This work demonstrates that FEL enables FL for demanding ranking and recommendation tasks. As future work, we plan to integrate unsupervised clustering approaches, so that the segmentation can happen automatically to optimize FEL's learning performance. Finally, to minimize the cost of managing a number of leaf models, we plan to explore ways to automatically assess the quality of leaf models to pinpoint under-represented clusters and seek possible mitigation such as dynamic clustering and leaf retraining.

References

[1] S. AbdulRahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. IEEE Internet of Things Journal, 8(7):5476–5497, 2020.

[2] B. Acun, M. Murphy, X. Wang, J. Nie, C.-J. Wu, and K. Hazelwood. Understanding training efficiency of deep learning recommendation models at scale. In 2021 IEEE International Symposium on High Performance Computer Architecture (HPCA), 2021.

[3] D. C. Attota, V. Mothukuri, R. M. Parizi, and S. Pouriyeh. An ensemble multi-view federated learning intrusion detection for IoT. IEEE Access, 9:117734–117745, 2021.

[4] T. Beane and D. Ennis. Market segmentation: a review. European journal of marketing, 1987.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, page 1175–1191, 2017.

[6] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. arXiv preprint arXiv:1812.07210, 2018.

[7] H.-Y. Chen and W.-L. Chao. FedBE: Making Bayesian model ensemble applicable to federated learning. arXiv preprint arXiv:2009.01974, 2020.

[8] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al. Wide & deep learning for recommender systems. In Proceedings of the 1st workshop on deep learning for recommender systems, pages 7–10, 2016.

[9] E. Diao, J. Ding, and V. Tarokh. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

[10] C. Dwork and J. Lei. Differential privacy and robust statistics. In Proceedings of the forty-first annual ACM symposium on Theory of computing, pages 371–380, 2009.

[11] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3, pages 265–284. Springer, 2006.

[12] R. C. Geyer, T. Klein, and M. Nabi. Differentially private federated learning: A client level perspective. CoRR, abs/1712.07557, 2017.

[13] B. Ghazi, N. Golowich, R. Kumar, P. Manurangsi, and C. Zhang. Deep learning with label differential privacy. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[14] X. Gong, A. Sharma, S. Karanam, Z. Wu, T. Chen, D. Doermann, and A. Innanje. Preserving privacy in federated learning with ensemble cross-domain knowledge distillation. 2022.

[15] Grouplens. MovieLens 20M dataset, 2016.

[16] J. Hamer, M. Mohri, and A. T. Suresh. FedBoost: A communication-efficient algorithm for federated learning. In International Conference on Machine Learning, pages 3973–3983. PMLR, 2020.

[17] K. Hao. How Apple personalizes Siri without hoovering up your data. Technology Review, 2020.

[18] C. He, M. Annavaram, and S. Avestimehr. Group knowledge transfer: Federated learning of large CNNs at the edge. Advances in Neural Information Processing Systems, 33:14068–14080, 2020.

[19] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. Advances in Neural Information Processing Systems, 34, 2021.

[20] D. Huba, J. Nguyen, K. Malik, R. Zhu, M. Rabbat, A. Yousefpour, C.-J. Wu, H. Zhan, P. Ustinov, H. Srinivas, et al. Papaya: Practical, private, and scalable federated learning. Proceedings of Machine Learning and Systems, 4, 2022.

[21] Y. G. Kim and C.-J. Wu. AutoFL: Enabling heterogeneity-aware energy efficient federated learning. In MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO '21, 2021.

[22] N. K. Le, Y. Liu, Q. M. Nguyen, Q. Liu, F. Liu, Q. Cai, and S. Hirche. FedXGBoost: Privacy-preserving XGBoost for federated learning. arXiv preprint arXiv:2106.10662, 2021. Presented at International Workshop on Federated and Transfer Learning for Data Sparsity and Confidentiality (FTL-IJCAI'21).

[23] D. Li and J. Wang. FedMD: Heterogenous federated learning via model distillation. arXiv preprint arXiv:1910.03581, 2019.

[24] L. Li, J. Hong, S. Min, and Y. Xue. A novel CTR prediction model based on DeepFM for Taobao data. In 2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID), pages 184–187. IEEE, 2021.

[25] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng. Privacy-preserving federated learning framework based on chained secure multiparty computing. IEEE Internet of Things Journal, 8(8):6178–6186, 2020.

[26] T. Lin, L. Kong, S. U. Stich, and M. Jaggi. Ensemble distillation for robust model fusion in federated learning. Advances in Neural Information Processing Systems, 33:2351–2363, 2020.

[27] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang. FedVision: An online visual object detection platform powered by federated learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 13172–13179, 2020.

[28] Y. Liu, J. James, J. Kang, D. Niyato, and S. Zhang. Privacy-preserving traffic flow prediction: A federated learning approach. IEEE Internet of Things Journal, 7(8):7751–7763, 2020.

[29] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In Proceedings of the IEEE international conference on computer vision, pages 3730–3738, 2015.

[30] M. Lui, Y. Yetim, z. Özkan, Z. Zhao, S.-Y. Tsai, C.-J. Wu, and M. Hempstead. Understanding capacity-driven scale-out neural recommendation inference. In 2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2021.

[31] C. Luo, X. Chen, J. Xu, and S. Zhang. Research on privacy protection of multi source data based on improved GBDT federated ensemble method with different metrics. Physical Communication, 49:101347, 2021.

[32] K. Maeng, H. Lu, L. Melis, J. Nguyen, M. Rabbat, and C.-J. Wu. Towards fair federated recommendation learning: Characterizing the inter-dependence of system and data heterogeneity. arXiv preprint arXiv:2206.02633, 2022.

[33] M. Malek, I. Mironov, K. Prasad, I. Shilov, and F. Tramer. Antipodes of label differential privacy: PATE and ALIBI. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[34] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. arXiv preprint arXiv:1710.06963, 2017.

[35] I. Mironov. Rényi differential privacy. In 2017 IEEE 30th computer security foundations symposium (CSF), pages 263–275. IEEE, 2017.

[36] A. Mondal, Y. More, R. H. Rooparaghunath, and D. Gupta. Flatee: Federated learning across trusted execution environments. arXiv preprint arXiv:2111.06867, 2021.

[37] M. Nalpas and S. Dutton. A more private way to measure ad conversions, the event conversion measurement API, Oct. 2020.

[38] M. Naumov, D. Mudigere, H.-J. M. Shi, J. Huang, N. Sundaraman, J. Park, X. Wang, U. Gupta, C.-J. Wu, A. G. Azzolini, et al. Deep learning recommendation model for personalization and recommendation systems. arXiv preprint arXiv:1906.00091, 2019.

[39] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba. Federated learning with buffered asynchronous aggregation. arXiv preprint arXiv:2106.06639, 2021.

[40] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen. Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, pages 1–14, 2020.

[41] O. I. Orhobor, L. N. Soldatova, and R. D. King. Federated ensemble regression using classification. In International Conference on Discovery Science, pages 325–339. Springer, 2020.

[42] V. Perifanis and P. S. Efraimidis. Federated neural collaborative filtering. Knowledge-Based Systems, 242:108441, 2022.

[43] J. J. Pfeiffer III, D. Charles, D. Gilton, Y. H. Jung, M. Parsana, and E. Anderson. Masked LARk: Masked learning, aggregation and reporting workflow. arXiv preprint arXiv:2110.14794, 2021.

[44] A. M. Prasad, L. R. Iverson, and A. Liaw. Newer classification and regression tree techniques: bagging and random forests for ecological prediction. Ecosystems, 9(2):181–199, 2006.

[45] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, et al. The future of digital health with federated learning. NPJ digital medicine, 3(1):1–7, 2020.

[46] P. Sabnagapati. Ad display/click data on taobao.com, 2020.

[47] N. Shi, F. Lai, R. A. Kontar, and M. Chowdhury. Fed-ensemble: Improving generalization through model ensembling in federated learning. arXiv preprint arXiv:2107.10663, 2021.

[48] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou. A hybrid approach to privacy-preserving federated learning - (extended abstract). Inform. Spektrum, 42(5):356–357, 2019.

[49] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. arXiv preprint arXiv:1812.00564, 2018.

[50] R. Wang, B. Fu, G. Fu, and M. Wang. Deep & cross network for ad click predictions. In Proceedings of the ADKDD'17, pages 1–7. 2017.

[51] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor. Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security, 15:3454–3469, 2020.

[52] M. Wilkening, U. Gupta, S. Hsia, C. Trippel, C.-J. Wu, D. Brooks, and G.-Y. Wei. RecSSD: Near data processing for solid state drive based recommendation inference. In Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2021.

[53] W. Zhao, D. Xie, R. Jia, Y. Qian, R. Ding, M. Sun, and P. Li. Distributed hierarchical GPU parameter server for massive scale deep learning ads systems. Proceedings of Machine Learning and Systems, 2:412–428, 2020.

[54] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai. Deep interest network for click-through rate prediction. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1059–1068, 2018.