1 APPENDIX

1.1 MORE DETAILS OF THE DATASETS

Table 1 shows the characteristics of the four datasets used in experiments.

Dataset	Interactions	Users	Items	Sparsity
MovieLens-100K	100,000	943	1,682	93.70%
MovieLens-1M	1,000,209	6,040	3,706	95.53%
Lastfm-2K	185,650	1,600	12,454	99.07%
Amazon-Video	63,836	8,072	11,830	99.93%

Table 1: Datasets statistics.

1.2 EVALUATION PROTOCOLS

For dataset split, we adopt prevalent leave-one-out evaluation, following He et al. (2017). For each user, we take her last interaction as the test data and remain other interactions for training. Additionally, we regard the latest raction in the training set as the validation data to find hyper-parameters. For the final evaluation, we follow the regular strategy Koren (2008); He et al. (2017) that randomly samples 99 unobserved items for each user, ranking the test item among the 100 items. We evaluate the ranked list with Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) metrics. In particular, HR measures whether the test data is in the top-K list, while NDCG considers the test data's position in the list. The two evaluation metrics could be formulated as follows,

$$HR = \frac{1}{N}\sum_{u=1}^{N} hits(u) \tag{1}$$

where N is the number of users and $hits(u) = 1$ indicates that the test data of user u is in the top-K recommendation list, otherwise $hits(u) = 0$.

$$NDCG = \frac{1}{N}\sum_{u=1}^{N} \frac{\log 2}{\log(p_u + 1)} \tag{2}$$

where p_u denotes the position of test data of user u in the recommendation list and $p_u \to \infty$ when the test data is not in the top-K recommendation list. Here we set $K = 10$ for all experiments.

Method	MovieLens-100K			MovieLens-1M			Lastfm-2K			Amazon-Video		
	parameters	epochs	time (s)	parameters	epochs	time (s)	parameters	epochs	time (s)	parameters	epochs	time (s)
FedNCF	86,753	90	1,800	314,625	98	18,718	452,481	95	4,750	639,617	60	5,820
w/ finetune	86,753	90	1,876	314,625	98	19213	452,481	83	4,680	639,617	32	4,120
FedRecon	53,856	465	9,486	118,624	470	123,892	398,560	98	7,154	378,592	57	3,819
w/ finetune	53,856	444	9,380	118,624	476	125,230	398,560	87	6,740	378,592	91	4,760
FedMF	53,856	72	936	118,624	93	12,927	398,560	82	3,280	378,592	56	3,920
PFedRec	53,857	61	854	118,625	95	13,585	398,561	60	2,340	378,593	75	5,100

Table 2: Efficiency comparison results on four datasets, including model parameter volume, training epochs and running time.

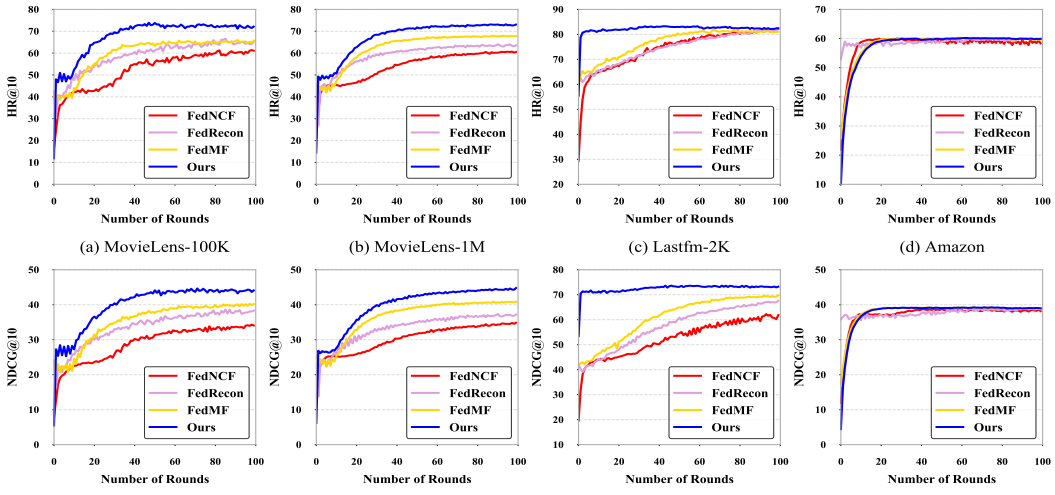
1.3 EFFICIENCY COMPARISON

In federated learning, space and time efficiency are prominent factors for application. We compare the model's efficiency, including parameter volume, training epochs and running time, as shown in Table 2. According to the results, *(1)* FedNCF has the largest volumes of parameters. Parameters of recommendation systems consist of several parts, *i.e.,* user and item embedding and score function. The embedding size is the same for all methods in each dataset. For score function, FedNCF employs a three-layers MLP, which leads to much more parameters than one-dimensional embedding in other methods. *(2)* FedRecon takes the most training epochs to converge. The reconstruction mechanism in FedRecon demands retraining the local module from scratch in each round, which results in

more training epochs. Taking MovieLens-100K as an example, our method converges in 61 training epochs, while FedRecon requires 465 training epochs, approaching 8 times ours. FedNCF takes the second longest training epochs and training time. On MovieLens-100K, the total training time of FedNCF is about 2 times as long as ours. *(3)* The space and time efficiency of FedMF is at the same level as our method. Our method enhances the personalization modeling and achieves the best performance without extra computational complexity. In our experiments, we found that just one local gradient descent step to learn personalized item embedding can yield advanced performance.

1.4 Convergence Analysis

We show the convergence curves of the two evaluation metrics for our method and baselines on four datasets. As shown in Figure 1, our method achieves the fastest convergence on all datasets and metrics almost all the time, followed by FedMF and FedRecon. FedNCF has the slowest convergence speed because it has the most parameters and requires a longer number of iterations.



(a) MovieLens-100K (b) MovieLens-1M (c) Lastfm-2K (d) Amazon

(e) MovieLens-100K (f) MovieLens-1M (g) Lastfm-2K (h) Amazon

Figure 1: Model convergence comparison. The horizontal axis is the number of federated optimization rounds, and the vertical axis is the model performance, where (a)-(d) are HR@10 metric and (e)-(h) are NDCG@10 metric.

1.5 Hyper-Parameters Study

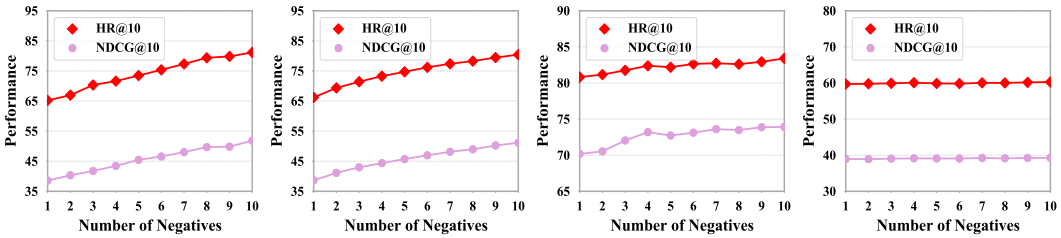
Effect of latent embedding size We tune the latent embedding size from $\{16, 32, 64, 128\}$. Accordingly, the architecture of the score function, *i.e.,* one-layer MLP, is $16 \rightarrow 1$, $32 \rightarrow 1$, $64 \rightarrow 1$ and $128 \rightarrow 1$, respectively. Experimental results are shown in Table 3. Almost all four datasets achieve the best results when the latent embedding size is 32. Due to the limited data of a single user, increasing the model parameters did not obtain further performance improvement. Generally, setting the dimension to 16 or 32 can achieve advanced performance. A small volume of parameters helps to build a lightweight on-device recommendation model.

Effect of negative sample size We set the negative sample size from 1 to 10 and observe the effect on model performance. Experimental results are shown in Figure 2. For two MovieLens datasets, model performance improves significantly as the number of negative samples increases. Since these two datasets have more user data than Lastfm-2K and Amazon, they contain richer user preference information to learn. Sampling more negative instances help the model to further identify user preferences. On the other two datasets, 4 negative samples are enough to obtain ideal model performance.

Effect of client samples participating in each round There is a trade-off between client sampling ratio and communication efficiency in federated optimization. Generally, the more clients are

Dataset	Metrics	16	32	64	128
Movielens-100K	HR@10	$\mathbf{72.81 \pm 0.90}$	71.62 ± 0.83	71.64 ± 0.44	71.75 ± 0.80
	NDCG@10	43.32 ± 0.43	43.44 ± 0.89	44.70 ± 1.01	$\mathbf{45.42 \pm 0.88}$
Movielens-1M	HR@10	72.70 ± 0.18	$\mathbf{73.26 \pm 0.20}$	71.91 ± 0.41	70.67 ± 0.15
	NDCG@10	43.04 ± 0.19	$\mathbf{44.36 \pm 0.16}$	44.19 ± 0.15	43.74 ± 0.36
Lastfm-2K	HR@10	81.93 ± 0.80	$\mathbf{82.38 \pm 0.92}$	81.93 ± 0.41	82.01 ± 0.64
	NDCG@10	72.47 ± 0.65	$\mathbf{73.19 \pm 0.38}$	72.41 ± 0.88	72.63 ± 0.29
Amazon-Video	HR@10	59.96 ± 0.18	$\mathbf{60.08 \pm 0.08}$	59.75 ± 0.15	59.60 ± 0.16
	NDCG@10	$\mathbf{39.14 \pm 0.07}$	39.12 ± 0.09	39.07 ± 0.14	38.99 ± 0.11

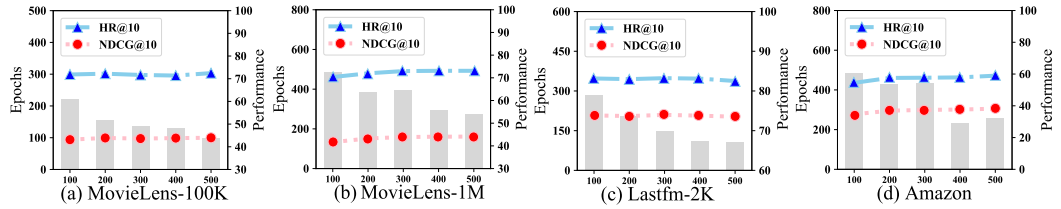
Table 3: Performance of different latent embedding sizes on four datasets. The results are the mean and standard deviation of the five repeated trials. Each number has an order of magnitude of 1e-2.



(a) MovieLens-100K (b) MovieLens-1M (c) Lastfm-2K (d) Amazon

Figure 2: Performance under different negative sample numbers for each positive sample.

selected to participate in the global aggregation, the faster the model converges in each round. However, in the physical scenario, it is difficult for the server to collect the complete model information from all the clients. Particularly, there are a large number of user clients in the recommendation scenario, which further increases the difficulty. To verify the relationship between the model's convergence and the clients' participation in each round, we conduct experiments on four datasets with various client samples. To create a consistent validation environment for all datasets, we set the number of users selected in each round as 100, 200, 300, 400 and 500, respectively. Experimental results are represented in Figure 3.



(a) MovieLens-100K (b) MovieLens-1M (c) Lastfm-2K (d) Amazon

Figure 3: Performance under different client numbers participating in each round.

We run the model until convergence and report the best validation performance with the corresponding epoch. According to the experimental results, we can observe that PFedRec could reach consistently advanced performance in all settings on all datasets, even only with 100 clients selected in each round during model training. On the other hand, it is obvious that more clients participating in each round of training lead to a quicker convergence. PFedRec supports the server to update with insufficient clients accessible, which is ubiquitous in physical circumstances.

REFERENCES

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 426–434, 2008.