A Value-Preserving and Efficiency-Aware Split Learning Framework for Recommender System: Rebuttal

To Reviewer a4UN

Thanks very much for your precious time and careful review. Here we provide answers for some of the questions mentioned hoping to clarify your confusion.

Q1 There is no clear discussion on how the three components collaborate to solve the problem, and the connection between them should be more logically structured.

R1 Sorry for the confusion caused by the unclear explanation of the motivation in our paper. We would like to clarify the connection between the three parts of our main body.

The initial and first purpose of our paper is to propose a cohesive split learning framework to enable the usage of more user or item-related features (not available or have serious safety concerns in a centralized setting) from different participants for a potentially better overall recommendation performance (both training & inference).

FedAds [1] is a previous SOTA (2023) that pioneers the usage of features from multiple parts in a vertically split scenario. However, the model structure used in FedAds is naive with only fully connected layers. In this paper, we take a step forward and adapt DeepFM to such a multiple-party-involved, vertical federated setting. Compared to naive DNN, DeepFM is a classic but effective structure and is still widely used in real-world business. However, adapting DeepFM to such a vFL setting is not straightforward because the second-order feature (e.g. $x_1 \times x_2$) in the Factorized Machine (FM) requires direct interaction in the early stage, which leads to a serious threat in the above-mentioned decentralized paradigm. Therefore, we design a safe FM-based splitting with secure interaction modeling in section 3.2 to avoid direct input exposure from the follower(s) to the leader.

Even with the secure interaction above, which focuses solely on the FM part, such a decentralized RS paradigm may still have potential risks of privacy leakage (the followers' data value), which is not considered in [1] but discussed in [2,3]. However, the measurement of such data leakage is still unclear. So in section 3.3, we first define the data value leakage γ between the primary task A to the unknown attack task B, which measures how much unnecessary information is leaked from the follower to the leader. In our split DeepFM, it measures the information leakage on h_{p3}(the output of the Deep part). Since $h_{p_3}$is also a vital part contributing to the final result, we cannot fully ban the information carried on this feature but use a VIB-based method to allow only the minimal information necessary (related to the main task) from the follower to the leader.

Another main concern raised by such a multi-party paradigm is the increase in communication overhead. Though additional cost is unavoidable, we discuss different efficiency optimization strategies (Training Time and Communication Cost) and compare the efficiency under different real-world settings (follower numbers & bandwidth) to prove the feasibility of our split DeepFM framework.

Overall, the three parts 3.2 (FM-based Secure Modeling), 3.3 (Reducing Data Leakage), and 3.4 (Efficiency Optimization) do not have a sequential relationship; rather, they serve the overall framework aimed at leveraging features from more participants for a better recommendation system. (1) Compared to naive DNN ([1]), it achieves better recommendation performance. - 3.2; (2) It explores reducing data value leakage when more participants are introduced. - 3.3; (3) It enhances the efficiency of this framework's operation. - 3.4.

Q2 Given that DeepFM is from 2017, why was it selected for this study? Is there a specific reason it is well-suited for this framework, or would newer models have better results? Maybe the authors should compare DeepFM with other more recent models or explain why DeepFM is particularly suited for this work.

R2 Because of the purpose mentioned above in A1, our paper navigates the possibility of utilizing more features from different parties for a better overall result and makes a step forward than FedAds[1] by adapting DeepFM to such a split decentralized setting rather than only fully connected layers.

The results in Table 1 of our paper are actually meant to support two conclusions:

(1) Using more input features (extra item-related features here) can indeed improve the performance of the recommendation model. Both results under split @ 2 parties are generally higher than centralized methods with fewer input features.

(2) Compared with FedAds[1]'s naive structure, our split DeepFM can achieve better performance.

Q3 The choice of benchmark models in 4.2.1 are too old. These models(Wide&Deep, DeepFM, NFM, AFM,DCN) are works being 7-8 years old with only one(DCNv2) in 2021. Why were they chosen, and how do they reflect the current state of the art?

R3 Following A2, the reason we compare our split DeepFM with benchmarks like Wide&Deep, DeepFM, NFM, DCN, etc. is that the purpose of this comparison is to demonstrate that using more input features does help the RS performance. The comparison is only meaningful when the base model under the split paradigm (DNN in FedAds) and (DeepFM in our paper) is of a similar age or similar complexity. The author does believe that with a more complex model structure, the recommendation performance may be significantly better than aged models like Wide&Deep or DeepFM, or even higher than FedAds or our split DeepFM with even more input features. However, this is not the core of our part of this experiment.

Q4 Why does the paper compare only with FedAds for split and DRAVF for value-preserving? Would it be possible to compare with other more recent vFL-based RSs or data protection methods to better demonstrate the advantages of this framework?

R4 (1) Following A1 & A3, the main focus in section 3.2 is to make a step forward and adapt the DeepFM structure into a decentralized paradigm and achieve a better result than previous SOTA FedAds[1].

(2) Information leakage in a vertical-federated setting has been discussed in previous work [2,3]. The attack and defense setting in [3] (leader label obtained by follower) is different from the setting in our paper. So we use DRAVF [2] as our benchmark to demonstrate the advantage of our VIB-based data protection.

Q5 In Section 4.2.2, the authors don't provide adequate explanations for why the performance of their method in certain tasks is suboptimal. Anomalies in the experimental results are not well analyzed, leaving me without a clear understanding of the strengths and weaknesses of the proposed framework.

R5 We believe that the suboptimal result referred to by the reviewer is the recommendation result of task 'Share' in Table 1, and data value leakage from the primary task to attack task 'Share' in Table 2. We provide a clearer explanation here:

(1) As explained in A3, the results in Table 1 are provided to demonstrate 2 things: (i) More input features are better. (ii) Our split DeepFM better utilizes features than FedAds to give a better overall performance. However, in a real-world recommendation task, conclusion (i) does not always work, in which only basic user and item features (even only IDs) are adequate for the final recommendation decision. In this task 'Share', we find that adding more item-related features makes little contribution to the final result. Thus, the centralized method like AFM, and DCNv2 can provide a better performance than both FedAds and split DeepFM (ours) with more input features.

(2) The theory of our proposed VIB to prevent data value leakage is to pass only the information necessary to the primary task ('Read' here) but reduce the extra information that can be potentially used by the leader for another secret task. Just like we mentioned above, task 'Share' only uses a minimal amount of input features, or in other words, there is merely no extra abundant information in h_{p3}if we want to keep the recommendation performance of task 'Read' unaffected.

Q6 & 7 There is no detailed discussion on the trade-off between performance and training time when using different compression strategies. The impact of compressors on both performance and efficiency should be discussed more thoroughly. The logical flow lacks clarity and coherence. For example, the relationship between bandwidth limitations and training efficiency is not clearly established in the paper. The pipeline-parallel strategy is mentioned in the introduction but does not align clearly with the results presented in the efficiency experiments.

R6 & 7 We use two different methods to reduce the communication overhead and increase the training efficiency in section 3.4 (results in 4.2.3).

In an ideal experimental environment, the bandwidth between two (or more) parties involved is usually big. Here, we use $\geq$1000 to represent a physical connection. In this situation, the original training time itself is not long 1851 seconds. However, real-world joint training of multiple parties faces a more complicated training environment with smaller bandwidth and higher latency.

With a bandwidth of 5Mbps, the training time has increased to 10247 seconds, showing the importance of efficiency optimization methods.

Results in Tables 3 and 4 show that (i) Both methods achieve a significant optimization in training efficiency under a condition of smaller bandwidth. (ii) A more aggressive compressing strategy will increase training efficiency at the cost of more overall performance degradation (Sparse 15% lower than 25% in Quantization). (iii) With more pipelines used, the training time decreases, especially when the bandwidth between parties is small. This is because a smaller bandwidth may lead to more 'time bubbles' in a decentralized training setting. We will provide a more thorough explanation in the edited version.

Q8 The paper contains several grammatical issues, such as line 103 and line 398. In addition, the tenses should be consistent throughout the paper—there is a mix of past and present tense.

R8 We would like to thank the reviewer once again for their careful reading. We will thoroughly review the entire text again and correct all the grammatical errors and typos in the edited version.

[1] FedAds: A Benchmark for Privacy-Preserving CVR Estimation with Vertical Federated Learning, https://arxiv.org/pdf/2305.08328

[2] Defending against Reconstruction Attack in Vertical Federated Learning, https://arxiv.org/abs/2107.09898

[3] Label Leakage and Protection from Forward Embedding in Vertical Federated Learning, https://arxiv.org/pdf/2203.01451