PIGR-VFL: PARTITION-INDUCED-GAP REDUCTION VERTICAL FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vertical Federated Learning (VFL) enables K clients to collaboratively exploit partitioned feature blocks without sharing their raw features. However, when each client holds an increasingly fine-grained partitioned feature block, aggregating local representations obtained by local extraction models may cause information loss and degrade predictive performance. To address this issue, we first derive a worst-case bound on the population risk to identify the factors responsible for this VFL-specific partition-induced performance degradation. Decomposing this bound suggests that combining model architectural designs (coordinate-preserving embedding, participant-mean aggregation) with optional training objectives (extraction-curvature regularization, proximal extraction-output regularization) reduces the identified factors without unduly restricting the extraction model's expressiveness; we refer to the resulting method as Partition-Induced-Gap Reduction VFL (PIGR-VFL). Through numerical experiments, we confirm the effectiveness of PIGR-VFL, particularly on image classification tasks.

1 INTRODUCTION

Vertical Federated Learning (VFL) is a collaborative learning paradigm in which features are vertically partitioned into K feature blocks. It enables the clients to jointly train a global predictive model that exploits information from all feature blocks without directly sharing their raw features. For example, personal information distributed across hospitals, insurance companies, and banks—including health, insurance, and financial records—may be collaboratively utilized for credit evaluation and insurance risk assessment, subject to applicable privacy and regulatory requirements (Poirot et al., 2019; Zheng et al., 2020; Huang et al., 2023; Yin et al., 2023; Paramarthalingam et al., 2025). Motivated by such applications, our goal is to develop a scalable VFL that effectively exploits complementary information distributed across a potentially large number of organizations.

Vanilla VFL commonly employs a split neural-network architecture, in which clients transmit locally extracted representations to a central server for concatenation and prediction (Vepakomma et al., 2018; Gupta & Raskar, 2018; Singh et al., 2019; Romanini et al., 2021; Sun et al., 2023; Thapa et al., 2022). Practical frameworks such as FATE (Liu et al., 2021) and VFLAIR (Zou et al., 2024) support distributed VFL training, while extensions such as FedBCD (Liu et al., 2019), several communication-efficient variants Fu et al. (2022); Castiglia et al. (2022); Valdeira et al. (2025b), FedCVT Kang et al. (2022), and One-shot VFL (Sun et al., 2023) address communication costs and limited data availability. However, concatenation makes the input dimension of the fusion model dependent on the number of clients K, thereby requiring a fixed set of participating clients.

To eliminate this dependence of the fusion model's input dimension on the number of clients, LASER-VFL (Valdeira et al., 2025a) aggregates the available client representations by participant-mean aggregation rather than concatenation, making the aggregated dimension independent of the number of participating clients and thereby accommodating partial client participation. However, two issues remain: (i) the use of participant-mean aggregation is not theoretically justified, as other aggregation rules (e.g., full-client summation) equally satisfy the dimension-independence property, and (ii) empirical evaluations have been primarily limited to federations with a small number of clients. We conjecture that the choice of aggregation rule, together with increasingly fine-grained feature blocks (for large K), may cause information loss and degrade predictive performance. However, this has not been investigated.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

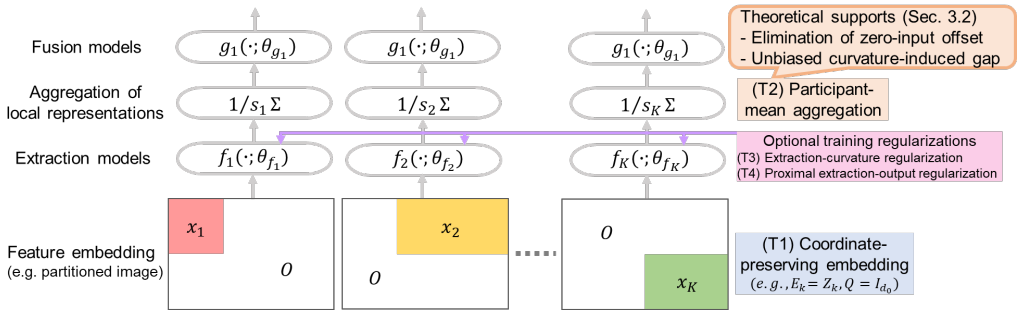

Figure 1: PIGR-VFL integrates two model architectural designs—(T1) coordinate-preserving embedding and (T2) participant-mean aggregation—together with two optional training regularizations—(T3) extraction-curvature regularization and (T4) proximal extraction-output regularization.

In this paper, we propose Partition-Induced-Gap Reduction VFL (PIGR-VFL). Our contributions are summarized as follows.

Identification of Partition-Induced Gap (Section 3.1). Our key idea is to quantitatively evaluate how the VFL-specific partitioning structure—namely, the feature embedding, the local extraction models, and the aggregation of local representations, all of which are architectural components unique to VFL—degrades predictive performance relative to a reference model that operates on the complete, unpartitioned feature vector. We found that, by fixing the input and output dimensions of the local extraction models, this degradation, which we refer to as the *partition-induced gap*, can be quantified. In Proposition 1, we derive a worst-case bound on the population risk and thereby identify five factors responsible for partition-induced performance degradation: (a) missing-feature gap, (b) embedding-reconstruction gap, (c) zero-input offset, (d) extraction-curvature-induced gap, and (e) extraction-output heterogeneity gap.

Partition-Induced-Gap Reduction VFL (PIGR-VFL) (Section 3.2). To reduce this partition-induced gap, we propose PIGR-VFL. Since the client-sampling rate is uncontrollable in practical VFL settings, (a) missing-feature gap cannot be controlled either, but the remaining four factors can be directly controlled through model design and training. We introduce two model architectural designs that eliminate (b) embedding-reconstruction gap and (c) zero-input offset: (T1) coordinate-preserving embedding, which retains each feature block's original coordinates via zero-padding (Figure 1), and (T2) participant-mean aggregation, which also provides a theoretical justification for the aggregation rule LASER-VFL has so far used only empirically. The remaining factors, (d) extraction-curvature-induced gap and (e) extraction-output heterogeneity gap, can instead be mitigated via two optional training regularizations—(T3) extraction-curvature regularization and (T4) proximal extraction-output regularization—so as not to unduly restrict model expressiveness. The resulting worst-case population risk bound of PIGR-VFL is given in Proposition 2.

Empirical Validation (Section 4). We evaluate PIGR-VFL on image and tabular classification tasks. We empirically confirm that PIGR-VFL achieves a substantial improvement, particularly on the image classification tasks, which critically require the global position/coordinate information of fine-grained feature blocks for accurate classification.

2 Problem Formulation and Related Works

Section 2.1 introduces the VFL problem setting, and Section 2.2 describes LASER-VFL.

2.1 VFL Problem Setting

Suppose that N data samples are vertically partitioned across K clients. Each complete sample is denoted by $x^n \in \mathbb{R}^{d_0}$, $n = 1, \ldots, N$. Although no client is allowed to access the complete sample x^n, each client k holds the local feature block $x_k^n \in \mathbb{R}^{d_k}$. The vertical feature partition can be represented using canonical zero-padding matrices.

Definition 1 (Vertical Feature Partition). *For each client k, let $Z_k \in \{0,1\}^{d_0 \times d_k}$ denote the fixed zero-padding matrix that places the local feature block x_k^n into its original coordinates in the complete feature space. Thus, $Z_k x_k^n \in \mathbb{R}^{d_0}$ contains x_k^n in the coordinates assigned to client k and zeros elsewhere. The complete sample can be additively reconstructed as $x^n = \sum_{k=1}^{K} Z_k x_k^n$.*

Client k employs a learnable extraction model f_k, parameterized by θ_{f_k}, to generate an intermediate representation $f_k(x_k^n;\theta_{f_k}) \in \mathbb{R}^{r_k}$. The local representations generated by all clients are concatenated (yielding a representation of dimension $\sum_{k=1}^K r_k$) and subsequently fed into a fusion model g. The resulting predictor is given by

$$h_{\text{vanilla}}(\{x_k^n\}_{k=1}^K;\theta) = g\Big(\text{Concat}\left(f_1(x_1^n;\theta_{f_1}),\ldots,f_K(x_K^n;\theta_{f_K})\right);\theta_g\Big), \tag{1}$$

where $\{f_k\}_{k=1}^K$ and g are learnable models parameterized by $\{\theta_{f_k}\}_{k=1}^K$ and θ_g, respectively. The collection of all learnable parameters is denoted by $\theta = (\{\theta_{f_k}\}_{k=1}^K, \theta_g)$.

In the vanilla VFL setting considered here, the fusion model g and the labels $\{t^n\}_{n=1}^N$ are held by a central server. The clients and the server collaboratively train the model parameters by solving

$$\min_\theta \Big(\mathcal{L}_{\text{vanilla}}(\theta) := \frac{1}{N}\sum_{n=1}^N \ell\left(h_{\text{vanilla}}\left(\{x_k^n\}_{k=1}^K;\theta\right), t^n\right)\Big), \tag{2}$$

where ℓ denotes a task-specific loss function, such as the cross-entropy loss for classification tasks.

2.2 LASER-VFL

We next describe LASER-VFL, a state-of-the-art VFL method. A limitation of the vanilla VFL architecture in (1) is that the input dimension of the fusion model, $\sum_{k=1}^K r_k$, depends on the number of concatenated representations. Consequently, the trained predictor generally requires the participation of all clients and cannot directly handle an arbitrary subset of available feature blocks.

LASER-VFL addresses this limitation by aggregating the available representations via participant-mean aggregation, rather than concatenation. To make this aggregation well-defined, all local extraction models produce representations of the same dimension: $f_k(x_k^n;\theta_{f_k}) \in \mathbb{R}^{r_0}$. Since the dimension of the resulting averaged representation is independent of the number of participating clients, LASER-VFL can process different numbers and combinations of available feature blocks. More specifically, LASER-VFL considers decentralized inference to solve local tasks, in which each client k maintains its own fusion model g_k, parameterized by θ_{g_k}. Given a participating-client subset $\mathcal{S}_k$ that contains client k, and $s_k = |\mathcal{S}_k|$, client k forms the predictor

$$h_{\text{LASER},k,\mathcal{S}_k}\left(\{x_i^n\}_{i\in\mathcal{S}_k};\theta\right) = g_k\left(\frac{1}{s_k}\sum_{i\in\mathcal{S}_k} f_i(x_i^n;\theta_{f_i});\theta_{g_k}\right). \tag{3}$$

Let q_k denote a distribution over $\mathcal{S}_k$. A simplified expectation-based form of the LASER-VFL training problem can be written as

$$\min_\theta \Big(\mathcal{L}_{\text{LASER}}(\theta) := \frac{1}{K}\sum_{k=1}^K \frac{1}{N_k}\sum_{n=1}^{N_k} \mathbb{E}_{\mathcal{S}_k\sim q_k}\left[\frac{1}{s_k}\ell_k\left(h_{\text{LASER},k,\mathcal{S}_k}\left(\{x_i^n\}_{i\in\mathcal{S}_k};\theta\right), t_k^n\right)\right]\Big), \tag{4}$$

where θ is augmented as $\theta := (\{\theta_{f_k}\}_{k=1}^K, \{\theta_{g_k}\}_{k=1}^K)$, and loss function ℓ_k and supervisor $\{t_k^n\}_{n=1}^{N_k}$ are prepared for each client k. The distribution q_k and the associated weighting recover the weighted objective over all subsets considered by LASER-VFL. Conceptually, this objective encompasses all possible subsets containing each client, whose explicit evaluation would require an exponential number of predictions. LASER-VFL therefore employs task sampling to obtain a stochastic estimate of the full objective while evaluating only a limited number of subsets at each training iteration.

However, as noted in the Introduction, two issues remain in LASER-VFL. First, although participant-mean aggregation is used in (3), its use is not theoretically justified: other aggregation rules are equally available, e.g., normalizing local representations by 1 (full-client summation). While LASER-VFL empirically demonstrates robustness to client sampling, no theoretical justification for this choice has been provided. Second, LASER-VFL has primarily been evaluated with a relatively small number of clients K. More generally, across conventional VFL methods, the VFL-specific partitioning structure—feature embedding, local extraction models, and aggregation of local representations—may cause information loss and degrade predictive performance, particularly with increasingly fine-grained feature blocks (for large K); however, this has not been investigated.

3 PROPOSED METHOD

In Section 3.1, we theoretically investigate how VFL's unique architectural structure—partitioned embeddings, local extraction models, and representation aggregation—affects predictive performance (the population risk of the predictor), and identify the resulting partition-induced gap factors. Guided by this analysis, Section 3.2 introduces PIGR-VFL, which integrates dedicated model architectural designs with optional training regularizations to reduce these partition-induced gap factors.

3.1 PARTITION-INDUCED-GAP IN POPULATION-RISK BOUNDS

We first formulate the discrepancy between the oracle population risk of a VFL predictor constructed from a subset of vertically partitioned feature blocks and that of an oracle reference predictor constructed from the corresponding complete sample. To compare their intermediate representations, the local extraction model and the reference extraction model must operate on a common input space and produce representations in a common output space.

To this end, let $Q \in \mathbb{R}^{\tilde{d}_0 \times d_0}$ be an arbitrary fixed linear map that defines the input space of the oracle reference extraction model. Each client i also applies a fixed, non-trainable embedding matrix $E_i \in \mathbb{R}^{\tilde{d}_0 \times d_i}$ to map its local feature block $x_i^n \in \mathbb{R}^{d_i}$ into the common space $\mathbb{R}^{\tilde{d}_0}$. At this stage, we do not impose any compatibility condition between Q and $\mathcal{E} := \{E_k\}_{k=1}^K$; consequently, the discrepancy between the embedded partial input and the corresponding reference input remains explicit in our analysis. The particular choices of Q, $\tilde{d}_0$, and $\mathcal{E}$ are specified in the next subsection.

We further introduce normalization coefficients $m := \{m_k\}_{k=1}^K$, where $1 \le m_k \le K$: $m_k = 1$ recovers summation aggregation, and $m_k = s_k$ gives participant-mean aggregation. Although LASER-VFL uses participant-mean aggregation, the choice $m_k = s_k$ has not been fully characterized, as noted in Section 2.2. We therefore retain a general m_k in this subsection and identify an appropriate choice in the next subsection.

Then, the VFL predictor associated with client k is defined as

$$h_{\mathrm{VFL},k,\mathcal{S}_k}(\{x_i^n\}_{i\in\mathcal{S}_k}, \mathcal{E}, m; \theta) := g_k\left(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n; \theta_{f_i}); \theta_{g_k}\right), \tag{5}$$

where each local extraction model $f_i(\cdot;\theta_{f_i}) : \mathbb{R}^{\tilde{d}_0} \to \mathbb{R}^{r_0}$ may have a client-specific model architecture, provided that its input and output dimensions agree with those of the reference extraction model $f(\cdot;\theta_f) : \mathbb{R}^{\tilde{d}_0} \to \mathbb{R}^{r_0}$. For theoretical comparison, we define the oracle reference predictor as

$$h_{\mathrm{REF},k}(x^n, Q; \theta_{\mathrm{REF}}) := g_k\left(f\Big(\frac{Qx^n}{K}; \theta_f\Big); \theta_{g_k}\right), \tag{6}$$

where $\theta_{\mathrm{REF}} = \{\theta_f, \{\theta_{g_k}\}_{k=1}^K\}$, and the oracle reference predictor is introduced solely for theoretical comparison; the complete sample x^n is not accessible to any participating entity during VFL training or inference. Averaging over the client-subsampling distribution $\mathcal{S}_k \sim q_k$ and the task-specific data distribution $(x, t_k) \sim \mathcal{D}_k$, we define oracle population risk for each case as

$$R_{\mathrm{VFL}}(\theta) := \frac{1}{K}\sum_{k=1}^K \mathbb{E}_{(x,t_k)\sim\mathcal{D}_k}\mathbb{E}_{\mathcal{S}_k\sim q_k}\left[\ell_k\left(h_{\mathrm{VFL},k,\mathcal{S}_k}(\{x_i\}_{i\in\mathcal{S}_k}, \mathcal{E}, m; \theta), t_k\right)\right], \tag{7}$$

$$R_{\mathrm{REF}}(\theta_{\mathrm{REF}}) := \frac{1}{K}\sum_{k=1}^K \mathbb{E}_{(x,t_k)\sim\mathcal{D}_k}\left[\ell_k\left(h_{\mathrm{REF},k}(x, Q; \theta_{\mathrm{REF}}), t_k\right)\right]. \tag{8}$$

We next bound the discrepancy between these population risks using the following assumptions.

Assumption 1. *The reference extraction model $f(\cdot;\theta_f) : \mathbb{R}^{\tilde{d}_0} \to \mathbb{R}^{r_0}$ is continuously differentiable and has β_f-Lipschitz Jacobian: $\|\nabla f(a;\theta_f) - \nabla f(b;\theta_f)\|_{\mathrm{op}} \le \beta_f \|a-b\|_2$ for all $a, b \in \mathbb{R}^{\tilde{d}_0}$.*

Assumption 2. *For every task k, the loss $\ell_k(\cdot, t_k)$ and the fusion model $g_k(\cdot;\theta_{g_k})$ are β_{ℓ_k}- and β_{g_k}-Lipschitz continuous in their first arguments, respectively.*

Proposition 1 (Worst-Case Population Risk Bound)**.** *Under Assumptions 1 and 2, for any fixed linear reference map $Q \in \mathbb{R}^{\tilde{d}_0 \times d_0}$, embedding matrices $\mathcal{E} := \{E_i \in \mathbb{R}^{\tilde{d}_0 \times d_i}\}_{i=1}^K$, and positive normalization $m := \{m_k\}_{k=1}^K$, the population-risk discrepancy satisfies*

$$|R_{\mathrm{VFL}}(\theta) - R_{\mathrm{REF}}(\theta_{\mathrm{REF}})| \;\le\; \Delta^{(Q,\mathcal{E},m)}(\theta_f),$$

where the partition-induced-gap $\Delta^{(Q,\mathcal{E},m)}(\theta_f) := \Delta^{(Q,m)}_{\mathrm{miss}}(\theta_f) + \Delta^{(Q,\mathcal{E},m)}_{\mathrm{rec}}(\theta_f) + \Delta^{(m)}_{\mathrm{offset}}(\theta_f) + \Delta^{(\mathcal{E},m)}_{\mathrm{curv}}(\theta_f) + \Delta^{(\mathcal{E},m)}_{\mathrm{het}}(\theta_f)$ consists of (a) missing-feature gap $\Delta^{(Q,m)}_{\mathrm{miss}}$, (b) embedding-reconstruction gap $\Delta^{(Q,\mathcal{E},m)}_{\mathrm{rec}}$, (c) zero-input offset $\Delta^{(m)}_{\mathrm{offset}}$, (d) curvature-induced gap $\Delta^{(\mathcal{E},m)}_{\mathrm{curv}}$, and (e) extraction-output heterogeneity gap $\Delta^{(\mathcal{E},m)}_{\mathrm{het}}$, given by

$$\text{(a) } \Delta^{(Q,m)}_{\mathrm{miss}}(\theta_f) := \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\, \mathbb{E}_{x,\mathcal{S}_k}\left[\left\| f\left(\frac{Qx_{\mathcal{S}_k}}{m_k};\theta_f\right) - f\left(\frac{Qx}{K};\theta_f\right)\right\|_2\right],$$

$$\text{(b) } \Delta^{(Q,\mathcal{E},m)}_{\mathrm{rec}}(\theta_f) := \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\, \mathbb{E}_{x,\mathcal{S}_k}\left[\left\| f\left(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i;\theta_f\right) - f\left(\frac{Qx_{\mathcal{S}_k}}{m_k};\theta_f\right)\right\|_2\right],$$

$$\text{(c) } \Delta^{(m)}_{\mathrm{offset}}(\theta_f) := \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\, \mathbb{E}_{\mathcal{S}_k}\left[\left|1 - \frac{s_k}{m_k}\right|\right] \|f(0;\theta_f)\|_2,$$

$$\text{(d) } \Delta^{(\mathcal{E},m)}_{\mathrm{curv}}(\theta_f) := \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\, \mathbb{E}_{x,\mathcal{S}_k}\left[\frac{\beta_f}{2}\left(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k}\left\|E_i x_i - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i\right\|_2^2 + \left|1-\frac{s_k}{m_k}\right|\left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i\right\|_2^2\right)\right],$$

$$\text{(e) } \Delta^{(\mathcal{E},m)}_{\mathrm{het}}(\theta_f) := \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\, \mathbb{E}_{x,\mathcal{S}_k}\left[\frac{1}{m_k}\sum_{i\in\mathcal{S}_k}\left\|f_i(E_i x_i;\theta_{f_i}) - f(E_i x_i;\theta_f)\right\|_2\right],$$

with $x_{\mathcal{S}_k} := \sum_{i\in\mathcal{S}_k} Z_i x_i$.

The proof is given in Appendix A. Proposition 1 is a two-sided bound: reducing the partition-induced gap $\Delta^{(Q,\mathcal{E},m)}(\theta_f)$ does not guarantee an improvement in $R_{\mathrm{VFL}}(\theta)$. Nonetheless, such worst-case guarantees are standard in theoretical analysis, analogous to robustness certificates in adversarial or distributionally robust learning: shrinking $\Delta^{(Q,\mathcal{E},m)}(\theta_f)$ provably lowers the worst-case value $R_{\mathrm{REF}}(\theta_{\mathrm{REF}}) + \Delta^{(Q,\mathcal{E},m)}(\theta_f)$, regardless of the sign of the underlying discrepancy. Furthermore, indiscriminately reducing all gap factors is not necessarily desirable: doing so generally requires restricting the expressive capacity of the extraction models, which can increase $R_{\mathrm{REF}}(\theta_{\mathrm{REF}})$ itself and thereby $R_{\mathrm{VFL}}(\theta)$. In the next subsection, we introduce techniques that reduce these gap factors without unduly restricting the extraction model expressiveness.

3.2 Partition-Induced-Gap Reduction VFL (PIGR-VFL)

In Proposition 1, the partition-induced gap is decomposed into five factors. Among these, (a) missing-feature gap cannot be controlled either, since the client-sampling rate is uncontrollable in practical VFL settings. The remaining four factors, (b)–(e), split into two qualitatively different groups. As detailed later in this subsection, a specific, appropriate choice of $(Q, \mathcal{E}, m)$ makes (b) embedding-reconstruction gap $\Delta^{(Q,\mathcal{E},m)}_{\mathrm{rec}}$ and (c) zero-input offset $\Delta^{(m)}_{\mathrm{offset}}$ vanish, without imposing any restriction on the extraction model parameters θ_f and $\{\theta_{f_k}\}_{k=1}^K$. In contrast, (d) curvature-induced gap $\Delta^{(\mathcal{E},m)}_{\mathrm{curv}}$ and (e) extraction-output heterogeneity gap $\Delta^{(\mathcal{E},m)}_{\mathrm{het}}$ depend on the extraction model parameters, and reducing them can in turn increase the achievable reference risk $\inf_\theta R_{\mathrm{REF}}(\theta_{\mathrm{REF}})$. We therefore control these two factors only softly, via optional training regularizations whose strength can be tuned to balance gap reduction against the extraction model's expressiveness.

Guided by this distinction, our PIGR-VFL introduces four techniques: (T1) coordinate-preserving embedding and (T2) participant-mean aggregation, which unconditionally eliminate (b) embedding-reconstruction gap and (c) zero-input offset, and additionally remove the bias term in (d) curvature-induced gap; and (T3) extraction-curvature regularization and (T4) proximal extraction-output regularization, which softly control the remaining curvature-induced variance term and (e) extraction-output heterogeneity gap. Each technique is explained below.

(T1) Coordinate-Preserving Embedding. To eliminate (b) embedding-reconstruction gap, we require embedding matrices $\{E_i\}$ that are compatible with Q in the sense that, for every participating-

client subset $\mathcal{S}_k$, $\sum_{i\in\mathcal{S}_k} E_i x_i^n = Q x_{\mathcal{S}_k}^n$. A sufficient condition is to set

$$E_k = QZ_k, \qquad \forall k, \tag{9}$$

since this embeds each client's feature block together with its coordinate position within the complete feature vector, regardless of the choice of Q; we refer to this as coordinate-preserving embedding. Since $\sum_{i\in\mathcal{S}_k} E_i x_i^n = Q\sum_{i\in\mathcal{S}_k} Z_i x_i^n = Q x_{\mathcal{S}_k}^n$ for every $\mathcal{S}_k$, we have $f(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f) - f(\frac{Q x_{\mathcal{S}_k}^n}{m_k};\theta_f) = 0$. Then, (b) the embedding-reconstruction gap vanishes ($\Delta_{\mathrm{rec}}^{(Q,\mathcal{E},m)}(\theta_f) = 0$) unconditionally, regardless of the extraction model's architecture and parameters θ_f, the choice of Q, and the participating-client subset $\mathcal{S}_k$.

Although Q retains this freedom, we use $Q = I_{d_0}$ as our choice (so that $\tilde{d}_0 = d_0$ and $E_k = Z_k$), since this choice guarantees that no information of the complete feature vector is lost through Q. Moreover, we interpret (9) with $Q = I_{d_0}$ as embedding the global position/coordinate, of each client's feature block within the complete feature vector. We expect this embedding to be particularly informative for tasks in which the position of a client's feature block carries meaningful information, such as the classification of a partitioned image.

(T2) Participant-Mean Aggregation. (c) The zero-input offset reflects the extent to which the extraction model's output deviates from zero at the zero-input vector; it arises from expanding (d) the curvature-induced gap via a Taylor expansion of the nonlinear extraction model f around the origin. To eliminate this, we set the normalization factor to the number of participating clients as

$$m_k = s_k, \qquad \forall k. \tag{10}$$

Since $1 - \frac{s_k}{m_k} = 0$, (c) the zero-input offset vanishes ($\Delta_{\mathrm{offset}}^{(m)}(\theta_f) = 0$), regardless of the extraction model's architecture and parameters θ_f and the participating-client subset $\mathcal{S}_k$. Although LASER-VFL uses participant-mean aggregation empirically, this result provides its theoretical justification.

Furthermore, (T2) participant-mean aggregation provides an additional benefit: it renders (d) the curvature-induced gap unbiased, since the second term $|1-\frac{s_k}{m_k}|\,\|\frac{\widetilde{x}_{\mathcal{S}_k}}{m_k}\|_2^2$ in the definition of $\Delta_{\mathrm{curv}}^{(\mathcal{E},m)}(\theta_f)$ vanishes. Owing to (T1) coordinate-preserving embedding with $Q = I_{d_0}$, the remaining first term in (d) the curvature-induced gap becomes interpretable: the variance of the participating clients' feature blocks is scaled by the extraction curvature β_f as follows:

$$\Delta_{\mathrm{curv}}^{(\mathcal{E},m)}(\theta_f) = \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\,\mathbb{E}_{x,\mathcal{S}_k}\left[\frac{\beta_f}{2s_k}\sum_{i\in\mathcal{S}_k}\left\|Z_i x_i - \frac{1}{s_k}\sum_{j\in\mathcal{S}_k} Z_j x_j\right\|_2^2\right] \propto \beta_f \cdot \mathrm{Var}\big(\{Z_i x_i\}_{i\in\mathcal{S}_k}\big). \tag{11}$$

While $\mathrm{Var}(\{Z_i x_i\}_{i\in\mathcal{S}_k})$ is determined solely by the feature partition, independent of the extraction model's architecture and parameters θ_f, β_f can be reduced through training. However, this reduction can restrict the extraction model's expressiveness. Thus, PIGR-VFL requires (T1) and (T2), whereas the remaining regularizations, (T3) and (T4), are optional components that trade off gap reduction against expressiveness. We next introduce a training regularization to relax β_f.

(T3) Extraction-Curvature Regularization. A training regularization to relax β_f is introduced for nonlinear extraction models. The Jacobian-Lipschitz constant β_f can be controlled by the magnitude of the input Hessian. Because minimizing its operator norm is generally difficult, we use the Frobenius norm as a tractable upper-bounding surrogate: $\|\nabla_y^2 f_k(y;\theta_{f_k})\|_{\mathrm{op}} \le \|\nabla_y^2 f_k(y;\theta_{f_k})\|_F$, where $y = Z_k x_k^n \in \mathbb{R}^{d_0}$ denotes the extraction model input. Accordingly, we define the extraction-curvature regularization as

$$\mathcal{L}_{\mathrm{curv}}(\theta) := \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{n=1}^{N_k}\left\|\nabla_y^2 f_k(y;\theta_{f_k})\big|_{y=Z_k x_k^n}\right\|_F^2. \tag{12}$$

Because $f_k : \mathbb{R}^{d_0} \to \mathbb{R}^{r_0}$ is vector-valued, its input Hessian is a third-order tensor, and computing its squared Frobenius norm directly requires constructing the $d_0 \times d_0$ input Hessian matrix for each of the r_0 output dimensions. This computation and the memory it requires become prohibitively expensive as d_0 and r_0 grow.

We therefore estimate the squared Frobenius norm without explicitly constructing the full input-Hessian tensor. Let $u, v \in \mathbb{R}^{d_0}$ be independent random vectors satisfying $\mathbb{E}[uu^\top] = \mathbb{E}[vv^\top] = I_{d_0}$.

For $a, b \in \mathbb{R}^{d_0}$ and step size $\mu > 0$, define the finite-difference operator

$$D_\mu f_k(y; a, b) := \frac{f_k(y+\mu a+\mu b) - f_k(y+\mu a-\mu b) - f_k(y-\mu a+\mu b) + f_k(y-\mu a-\mu b)}{4\mu^2} \in \mathbb{R}^{r_0},$$

which approximates the second-order directional derivative of f_k along a and b as $\mu \to 0$, i.e., the contraction of the input-Hessian tensor with a and b along its last two axes. By combining two randomized projections along the input dimension, the squared Frobenius norm can be expressed as $\|\nabla_y^2 f_k(y; \theta_{f_k})\|_F^2 = \lim_{\mu\to 0} \mathbb{E}_{u,v}[\|D_\mu f_k(y; u, v)\|_2^2]$. Using J independently sampled probe pairs $\{(u_j, v_j)\}_{j=1}^J$ and a fixed, small step size μ, we estimate as

$$\widehat{\mathcal{L}}_{\mathrm{curv}}(\theta) := \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{n=1}^{N_k}\frac{1}{J}\sum_{j=1}^{J}\|D_\mu f_k(Z_k x_k^n; u_j, v_j)\|_2^2 . \tag{13}$$

The entries of u_j and v_j are independently sampled from the Rademacher distribution, taking values in $\{-1, +1\}$ with equal probability. Unlike an automatic-differentiation-based Hessian-vector product, this finite-difference formulation requires only forward evaluations of f_k (four per probe pair), avoiding the memory overhead of a double-backward pass, at the cost of an approximation error controlled by the step size μ. The algorithm is given in Appendix B.

(T4) Proximal Extraction-Output Regularization. To mitigate (e) extraction-output heterogeneity gap, we introduce a training regularization. The extraction-output heterogeneity gap measures how far each local extraction model's output $f_i(Z_i x_i; \theta_{f_i})$ deviates from a common reference extraction output $f(Z_i x_i; \theta_f)$. However, f is introduced conceptually for theoretical analysis and is inaccessible, and thus cannot be directly used in a training regularization.

A natural relaxation arises under the view that each local extraction model is a noisy perturbation of f, averaging the local extraction models' outputs across clients cancels out the independent noise terms and thus approaches f itself. We therefore encourage each local extraction output to approach the average of the local extraction models' outputs, $\bar{h}^n := \frac{1}{K}\sum_{j=1}^K f_j(Z_j x_j^n; \theta_{f_j})$, which can be computed on the server. The proximal extraction-output regularization is then given by

$$\mathcal{L}_{\mathrm{prox}}(\theta) := \frac{1}{K r_0}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{n=1}^{N_k}\left\|f_k(Z_k x_k^n; \theta_{f_k}) - \mathrm{StopGrad}(\bar{h}^n)\right\|_2^2 , \tag{14}$$

where $\mathrm{StopGrad}(\cdot)$ denotes a stop-gradient operator.

PIGR-VFL. We now integrate the four techniques (T1)–(T4) to construct PIGR-VFL. By incorporating (T1) coordinate-preserving embedding and (T2) participant-mean aggregation into (5), we define the PIGR-VFL predictor at client k as

$$h_{\mathrm{PIGR},k,\mathcal{S}_k}\left(\{x_i^n\}_{i\in\mathcal{S}_k}; \theta\right) = g_k\Big(\frac{1}{s_k}\sum_{i\in\mathcal{S}_k} f_i(Z_i x_i^n; \theta_{f_i}); \theta_{g_k}\Big). \tag{15}$$

The main objective of PIGR-VFL is defined as

$$\mathcal{L}_{\mathrm{PIGR}}(\theta) := \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{n=1}^{N_k}\mathbb{E}_{\mathcal{S}_k\sim q_k}\left[\ell\left(h_{\mathrm{PIGR},k,\mathcal{S}_k}\left(\{x_i^n\}_{i\in\mathcal{S}_k}; \theta\right), t_k^n\right)\right]. \tag{16}$$

By incorporating (T3) extraction-curvature regularization and (T4) proximal extraction-output regularization, the PIGR-VFL training problem is formulated as

$$\min_\theta \ \mathcal{L}_{\mathrm{PIGR}}(\theta) + \gamma_{\mathrm{curv}}\widehat{\mathcal{L}}_{\mathrm{curv}}(\theta) + \gamma_{\mathrm{prox}}\mathcal{L}_{\mathrm{prox}}(\theta), \tag{17}$$

where the weight coefficients $\gamma_{\mathrm{curv}}(\geq 0)$ and $\gamma_{\mathrm{prox}}(\geq 0)$ are available, since (T3) and (T4) are optional components in PIGR-VFL.

Finally, integrating PIGR-VFL with Proposition 1 yields the following simplified worst-case population risk bound between PIGR-VFL and the oracle reference model.

Proposition 2 (Worst-Case Population Risk Bound for PIGR-VFL)**.** *Under Assumptions 1 and 2, the PIGR-VFL predictor $h_{\mathrm{PIGR},k,\mathcal{S}_k}$ in (15) satisfies*

$$|R_{\mathrm{PIGR}}(\theta) - R_{\mathrm{REF}}(\theta_{\mathrm{REF}})| \ \leq \ \Delta_{\mathrm{miss}}^{(Q,m)}(\theta_f) + \Delta_{\mathrm{curv}}^{(\mathcal{E},m)}(\theta_f) + \Delta_{\mathrm{het}}^{(\mathcal{E},m)}(\theta_f),$$

where $R_{\mathrm{PIGR}}(\theta) := R_{\mathrm{VFL}}(\theta)|_{E_k=Z_k, m_k=s_k, Q=I_{d_0}}$ denotes the population risk of the PIGR-VFL.

The proof is given in Appendix A. Proposition 2 shows that the worst-case discrepancy between $R_{\mathrm{PIGR}}(\theta)$ and $R_{\mathrm{REF}}(\theta)$ is governed only by the missing-feature, extraction-output heterogeneity, and curvature-induced gaps. As discussed below Proposition 1, however, this bound is two-sided; consequently, indiscriminately shrinking the right-hand-side terms does not necessarily improve $R_{\mathrm{PIGR}}(\theta)$, since doing so may restrict the extraction model's expressiveness and thereby fail to reduce $R_{\mathrm{PIGR}}(\theta)$ itself. For this reason, while (T1) and (T2) are required, we regard (T3) and (T4) as optional regularizations to be introduced with weight coefficients $\gamma_{\mathrm{curv}}, \gamma_{\mathrm{prox}}$, which may yield better performance; we investigate this empirically in the next section.

4 Numerical Experiments

We evaluate PIGR-VFL on image and tabular classification tasks commonly used in the VFL literature. Section 4.1 describes the experimental setups, and Section 4.2 presents the results.

4.1 Experimental Setups

Tasks and Models. We use three benchmark tests comprising two image-classification tasks (B1)–(B2) and a tabular-classification task (B3). (B1) CIFAR-10 and (B2) CIFAR-100 contain 32×32 color images ($d_0{=}32\times32\times3$) for 10- and 100-class object classification, respectively (Krizhevsky et al., 2009). For both tasks, we use a 3-layer CNN (with ReLU) extraction model with a ResNet fusion model (see Appendix C). Since we are primarily interested in the large-K regime, we aim to examine and improve robustness to fine-grained feature blocks; specifically, each image is partitioned into equal-sized blocks across $K \in \{4, 8, 16\}$ clients: a 2×2 grid of $d_k{=}16\times16\times3$-dimensional blocks for $K{=}4$, a 2×4 grid of $d_k{=}16\times8\times3$-dimensional blocks for $K{=}8$, and a 4×4 grid of $d_k{=}8\times8\times3$-dimensional blocks for $K{=}16$. (B3) HAPT is a smartphone-sensor dataset in tabular form for recognizing human activities and postural transitions ($d_0{=}560$) (Reyes-Ortiz et al., 2016). For this task, we use a 3-layer MLP extraction model with a 2-layer MLP fusion model. Each client holds a partitioned block for $K{\in}\{20, 40\}$, with dimension $d_k{=}560/K$. For all three benchmarks, every client is provided with the same supervisor label; that is, all clients collaboratively solve common classification task.

We report results under the full-participation setting ($s_k{=}K$) for two reasons. First, our theoretical contributions (Propositions 1 and 2) concern the extraction model's architecture and training regularization, and are agnostic to the specific choice of the client-sampling distribution q_k; evaluating under partial participation is therefore not essential for validating our main theoretical claims. Second, no well-established protocol currently exists for training and evaluating under partial participation in the large-K regime we target. Although our implementation is based on the LASER-VFL source code[1], it requires enumerating all client-availability patterns ($2^K{-}1$ of them). In the large-K regime, this becomes computationally exhaustive. We instead allocate our limited computational resources to a hyperparameter search over the learning rate and the training regularization weights $(\gamma_{\mathrm{curv}}, \gamma_{\mathrm{prox}})$.

Training. We use Adam (Kingma & Ba, 2014) for training, and the learning rate is tuned separately for each combination of benchmark, number of clients K, and VFL method. The candidate learning rates and the selected values yielding the highest test accuracy are summarized in Appendix C.

Comparison Methods. We prepare baseline methods (M1)–(M4) and four variants of the proposed method, (P1)–(P4). Additional information is summarized in Appendix C.

(M1) Vanilla VFL: the d_k-dimensional raw feature block is mapped to a local representation by each client's extraction model, and the concatenation of K representations is used as the fusion model's input, as in (1); however, we extend this so that each client holds its own fusion model, for a fair comparison with the other decentralized VFL methods.

(M2) LASER-VFL: d_k-dimensional raw feature block is mapped to a local representation by each client's extraction model, and the participant-mean aggregation of these representations is used as each client's fusion model input, as in (3).

(M3) PlugVFL (Sun et al., 2024): the model architecture is almost identical to that of (M2) LASER-VFL; however, it additionally employs party-wise dropout during representation aggregation.

(M4) Local: the d_k-dimensional raw feature block is mapped to a local representation by each client's extraction model; without aggregating the K clients' representations, each client independently makes a prediction in a decentralized manner.

[1] https://github.com/Valdeira/LASER-VFL

Table 1: Highest test accuracy for each benchmark test. Since each method is extended so that each client holds its own fusion model, K prediction results are obtained; their mean and standard deviation are reported.

Benchmark test	(B1) CIFAR-10			(B2) CIFAR-100			(B3) HAPT	
# of clients K	4	8	16	4	8	16	20	40
(M1) Vanilla VFL	$0.8641_{\pm 0.0008}$	$0.8422_{\pm 0.0016}$	$0.8146_{\pm 0.0028}$	$0.5394_{\pm 0.0054}$	$0.5530_{\pm 0.0046}$	$0.5378_{\pm 0.0044}$	$0.9555_{\pm 0.0022}$	$0.9560_{\pm 0.0027}$
(M2) LASER-VFL	$0.8611_{\pm 0.0013}$	$0.8398_{\pm 0.0014}$	$0.8113_{\pm 0.0014}$	$0.5268_{\pm 0.0023}$	$0.5495_{\pm 0.0032}$	$0.5380_{\pm 0.0027}$	$\mathbf{0.9572}_{\pm 0.0021}$	$0.9543_{\pm 0.0015}$
(M3) PlugVFL	$0.8698_{\pm 0.0006}$	$0.8557_{\pm 0.0023}$	$0.8479_{\pm 0.0024}$	$0.5170_{\pm 0.0011}$	$0.5030_{\pm 0.0013}$	$0.4747_{\pm 0.0041}$	$0.9500_{\pm 0.0022}$	$0.9523_{\pm 0.0017}$
(M4) Local	$0.7472_{\pm 0.0054}$	$0.6319_{\pm 0.0525}$	$0.5109_{\pm 0.0636}$	$0.4288_{\pm 0.0042}$	$0.3379_{\pm 0.0431}$	$0.2511_{\pm 0.0504}$	$0.7710_{\pm 0.0620}$	$0.6862_{\pm 0.0700}$
(P1) PIGR-VFL ($\gamma_{\text{curv}}=0, \gamma_{\text{prox}}=0$)	$0.8994_{\pm 0.0029}$	$0.8806_{\pm 0.0029}$	$0.8830_{\pm 0.0023}$	$0.6329_{\pm 0.0020}$	$\mathbf{0.6407}_{\pm 0.0026}$	$0.6145_{\pm 0.0029}$	$0.9528_{\pm 0.0020}$	$\mathbf{0.9575}_{\pm 0.0013}$
(P2) PIGR-VFL ($\gamma_{\text{curv}}=0, \gamma_{\text{prox}}>0$)	$\mathbf{0.9003}_{\pm 0.0019}$	$0.8800_{\pm 0.0031}$	$0.8835_{\pm 0.0016}$	$0.6334_{\pm 0.0029}$	$0.6391_{\pm 0.0048}$	$0.6143_{\pm 0.0029}$	$0.9528_{\pm 0.0020}$	$\mathbf{0.9575}_{\pm 0.0013}$
(P3) PIGR-VFL ($\gamma_{\text{curv}}>0, \gamma_{\text{prox}}=0$)	$0.8993_{\pm 0.0022}$	$0.8810_{\pm 0.0028}$	$0.8838_{\pm 0.0029}$	$\mathbf{0.6367}_{\pm 0.0038}$	$0.6379_{\pm 0.0061}$	$0.6154_{\pm 0.0039}$	$0.9528_{\pm 0.0011}$	$0.9564_{\pm 0.0016}$
(P4) PIGR-VFL ($\gamma_{\text{curv}}>0, \gamma_{\text{prox}}>0$)	$0.8987_{\pm 0.0013}$	$\mathbf{0.8815}_{\pm 0.0015}$	$\mathbf{0.8843}_{\pm 0.0028}$	$0.6332_{\pm 0.0028}$	$0.6384_{\pm 0.0025}$	$\mathbf{0.6167}_{\pm 0.0031}$	$0.9528_{\pm 0.0011}$	$0.9564_{\pm 0.0016}$

(P1)–(P4) PIGR-VFL: the d_k-dimensional raw feature block is first augmented, via (T1) coordinate-preserving embedding, into the d_0-dimensional space with zero-padding ($E_k{=}Z_k$, $Q{=}I_{d_0}$), and mapped to a local representation by each client's extraction model; at the server, (T2) participant-mean aggregation of the K local representations is applied. For (T3) extraction-curvature regularization, we use $J{=}1$ probe pair and step size $\mu{=}1{\times}10^{-2}$. Nine variants are prepared using all combinations of the two optional training regularizations (T3)–(T4), each with three candidate weights (e.g., for CIFAR-10, $\gamma_{\text{curv}}{\in}\{0.0, 1{\times}10^{-3}, 5{\times}10^{-3}\}$ and $\gamma_{\text{prox}}{\in}\{0.0, 0.5, 1.0\}$); the full set is summarized in Appendix C. In the main paper, we report four variants, each selected as the highest-accuracy variant within one of the following categories—i.e., with and without each regularization: (P1) without either regularization ($\gamma_{\text{curv}}{=}0$, $\gamma_{\text{prox}}{=}0$), (P2) with only (T4) proximal extraction-output regularization ($\gamma_{\text{curv}}{=}0$, $\gamma_{\text{prox}}{>}0$), (P3) with only (T3) extraction-curvature regularization ($\gamma_{\text{curv}}{>}0$, $\gamma_{\text{prox}}{=}0$), (P4) with both regularizations ($\gamma_{\text{curv}}{>}0$, $\gamma_{\text{prox}}{>}0$).

4.2 Experimental Results

Table 1 summarizes the highest test accuracy. PIGR-VFL (P1)–(P4) demonstrates its effectiveness, particularly on the image classification tasks (B1)–(B2) even under large K regimes. We focus on the difference between (M2) LASER-VFL and (P1) PIGR-VFL ($\gamma_{\text{curv}}{=}0$, $\gamma_{\text{prox}}{=}0$), since the two differ only in whether (T1) coordinate-preserving embedding is employed, yet (P1) PIGR-VFL achieves a substantial improvement on the image classification tasks (B1)–(B2). This indicates that (T1) coordinate-preserving embedding encodes the global position/coordinate of fine-grained feature blocks, which is critically effective for solving image classification tasks, but considerably less so for the tabular classification task (B3), since which client holds a given feature block would not substantially affect predictive performance on tabular data. For the optional training regularizations (T3)–(T4), small coefficients $\gamma_{\text{curv}}, \gamma_{\text{prox}}$ slightly improve predictive performance; however, the computational cost, particularly for (T3) extraction-curvature regularization, is substantial (see Appendix C). While (T3) is theoretically motivated by Proposition 1, its unfavorable cost-benefit trade-off suggests that it need not be used by default. These results highlight that (P1) PIGR-VFL ($\gamma_{\text{curv}}{=}0$, $\gamma_{\text{prox}}{=}0$) and (P2) PIGR-VFL ($\gamma_{\text{curv}}{=}0$, $\gamma_{\text{prox}}{>}0$) each offer a strong and practical choice.

In Appendix C, we further present additional ablation studies comparing summation-based aggregation ($m_k{=}1$) with (T2) participant-mean aggregation ($m_k{=}s_k$). The difference in predictive performance was slight, likely because ReLU-based models tend to output values close to zero at the zero input. As suggested by Proposition 2, the two aggregation rules are unlikely to differ significantly when this zero-input offset—i.e., the norm of the extraction model's output at the zero-vector input—happens to be small. Since this depends on the model architecture, however, (T2) participant-mean aggregation remains important for consistently achieving high predictive performance.

5 Conclusion

We propose PIGR-VFL to achieve high predictive accuracy in VFL. In Proposition 1, we quantify the discrepancy between the population risk of a VFL model incorporating K partitioned feature blocks and that of a reference model incorporating the complete, unpartitioned feature vector, and show that this discrepancy is bounded by five factors in the worst case. Since four of them can be directly controlled through model architectural design and training objectives, we introduce two model architectural designs—(T1) coordinate-preserving embedding and (T2) participant-mean aggregation—together with two optional training regularizations—(T3) extraction-curvature regularization and (T4) proximal extraction-output regularization. Experimental results show that PIGR-VFL is particularly effective in partitioned image classification tasks even with large-K regimes, since the global position/coordinate of fine-grained feature blocks is critical for solving them.

AI Use Statement. Generative AI tools (LLMs) were primarily used for text refinement throughout the manuscript. They were also used to verify the correctness of our proofs of the Propositions.

Ethics Statement. This work is a theoretical and methodological contribution to vertical federated learning, evaluated on publicly available benchmark datasets (CIFAR-10, CIFAR-100, HAPT). While our motivating applications may involve privacy-sensitive data (e.g., healthcare, finance), this paper does not use or release any such data, and we do not foresee ethical concerns beyond those inherent to the broader vertical federated learning literature.

Reproducibility Statement. We derive Propositions 1 and 2, and the assumptions used are clearly stated (Assumptions 1–2); their proofs are given in Appendix A. In addition, the details of the experimental setup (models, hyperparameter tuning, and additional results) are provided in Appendix C, and the source code used in the experiments is included as supplementary material.

REFERENCES

Timothy J Castiglia, Anirban Das, Shiqiang Wang, and Stacy Patterson. Compressed-vfl: Communication-efficient learning with vertically partitioned data. In *International conference on machine learning*, pp. 2738–2766. PMLR, 2022.

Fangcheng Fu, Xupeng Miao, Jiawei Jiang, Huanran Xue, and Bin Cui. Towards communication-efficient vertical federated learning training via cache-enabled local updates. *arXiv preprint arXiv:2207.14628*, 2022.

Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

Chung-ju Huang, Leye Wang, and Xiao Han. Vertical federated knowledge transfer via representation distillation for healthcare collaboration networks. In *Proceedings of the ACM Web Conference 2023*, pp. 4188–4199, 2023.

Yan Kang, Yang Liu, and Xinle Liang. Fedcvt: Semi-supervised vertical federated learning with cross-view training. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–16, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yang Liu, Yan Kang, Xinwei Zhang, Liping Li, Yong Cheng, Tianjian Chen, Mingyi Hong, and Qiang Yang. A communication efficient collaborative learning framework for distributed features. *arXiv preprint arXiv:1912.11187*, 2019.

Yang Liu, Tao Fan, Tianjian Chen, Qian Xu, and Qiang Yang. Fate: An industrial grade platform for collaborative learning with data protection. *Journal of Machine Learning Research*, 22(226):1–6, 2021.

Arjun Paramarthalingam, Harini Kanthan, Mahalakshmi Karthiban, et al. A privacy-preserving approach to health insurance fraud detection using vertical federated learning. *Sensors*, 25(23): 7354, 2025.

Maarten G Poirot, Praneeth Vepakomma, Ken Chang, Jayashree Kalpathy-Cramer, Rajiv Gupta, and Ramesh Raskar. Split learning for collaborative deep learning in healthcare. *arXiv preprint arXiv:1912.12115*, 2019.

Jorge-L Reyes-Ortiz, Luca Oneto, Albert Samà, Xavier Parra, and Davide Anguita. Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767, 2016.

Daniele Romanini, Adam James Hall, Pavlos Papadopoulos, Tom Titcombe, Abbas Ismail, Tudor Cebere, Robert Sandmann, Robin Roehm, and Michael A Hoeh. Pyvertical: A vertical federated learning framework for multi-headed splitnn. *arXiv preprint arXiv:2104.00489*, 2021.

Abhishek Singh, Praneeth Vepakomma, Otkrist Gupta, and Ramesh Raskar. Detailed comparison of communication efficiency of split learning and federated learning. *arXiv preprint arXiv:1909.09145*, 2019.

Jingwei Sun, Ziyue Xu, Dong Yang, Vishwesh Nath, Wenqi Li, Can Zhao, Daguang Xu, Yiran Chen, and Holger R Roth. Communication-efficient vertical federated learning with limited overlapping samples. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5180–5189. IEEE, 2023.

Jingwei Sun, Zhixu Du, Anna Dai, Saleh Baghersalimi, Alireza Amirshahi, David Atienza, and Yiran Chen. Plugvfl: Robust and ip-protecting vertical federated learning against unexpected quitting of parties. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 1124–1133. IEEE, 2024.

Chandra Thapa, Pathum Chamikara Mahawaga Arachchige, Seyit Camtepe, and Lichao Sun. Splitfed: When federated learning meets split learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 8485–8493, 2022.

Pedro Valdeira, Shiqiang Wang, and Yuejie Chi. Vertical federated learning with missing features during training and inference. In *International Conference on Learning Representations*, volume 2025, pp. 72304–72323, 2025a.

Pedro Valdeira, João Xavier, Cláudia Soares, and Yuejie Chi. Communication-efficient vertical federated learning via compressed error feedback. *IEEE transactions on signal processing*, 73: 1065–1080, 2025b.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

Tao Yin, Changgen Peng, Weijie Tan, Dequan Xu, and Hanlin Tang. Federated learning model for auto insurance rate setting based on tweedie distribution. *CMES-Computer Modeling in Engineering and Sciences*, 138(1):827–843, 2023.

Fanglan Zheng, Kun Li, Jiang Tian, Xiaojia Xiang, et al. A vertical federated learning method for interpretable scorecard and its application in credit scoring. *arXiv preprint arXiv:2009.06218*, 2020.

Tianyuan Zou, Zixuan Gu, Yu He, Hideaki Takahashi, Yang Liu, and Ya-Qin Zhang. Vflair: A research library and benchmark for vertical federated learning. In *International Conference on Learning Representations*, volume 2024, pp. 34157–34195, 2024.

A Proofs for Propositions in Section 3

A.1 Proof of Proposition 1

Proof. First, $|R_{\mathrm{VFL}}(\theta) - R_{\mathrm{REF}}(\theta_{\mathrm{REF}})|$ can be bounded under Assumptions 1 and 2:

$$|R_{\mathrm{VFL}}(\theta) - R_{\mathrm{REF}}(\theta_{\mathrm{REF}})| \le \beta_{\ell_k}\beta_{g_k}\left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n;\theta_{f_i}) - f\Big(\frac{Qx^n}{K};\theta_f\Big)\right\|_2. \tag{18}$$

The component in the right-hand-side norm term of (18) is decomposed as follows:

$$\begin{aligned}
&\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n;\theta_{f_i}) - f\Big(\frac{Qx^n}{K};\theta_f\Big)\\
&= \underbrace{\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n;\theta_{f_i}) - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f)}_{T_1} + \underbrace{\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)}_{T_2}\\
&\quad + \underbrace{f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big) - f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big)}_{T_3} + \underbrace{f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big) - f\Big(\frac{Qx^n}{K};\theta_f\Big)}_{T_4}.
\end{aligned} \tag{19}$$

Taking the L_2 norm of (19) and applying the triangle inequality, we obtain

$$\begin{aligned}
&\left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n;\theta_{f_i}) - f\Big(\frac{Qx^n}{K};\theta_f\Big)\right\|_2\\
&\le \left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f_i(E_i x_i^n;\theta_{f_i}) - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f)\right\|_2 + \left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\right\|_2\\
&\quad + \left\|f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big) - f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big)\right\|_2 + \left\|f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big) - f\Big(\frac{Qx^n}{K};\theta_f\Big)\right\|_2\\
&\le \underbrace{\frac{1}{m_k}\sum_{i\in\mathcal{S}_k}\left\|f_i(E_i x_i^n;\theta_{f_i}) - f(E_i x_i^n;\theta_f)\right\|_2}_{\text{(e) extraction-output heterogeneity gap}} + \underbrace{\left\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\right\|_2}_{\tau_1}\\
&\quad + \underbrace{\left\|f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big) - f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big)\right\|_2}_{\text{(b) embedding-reconstruction gap}} + \underbrace{\left\|f\Big(\frac{Qx^n_{\mathcal{S}_k}}{m_k};\theta_f\Big) - f\Big(\frac{Qx^n}{K};\theta_f\Big)\right\|_2}_{\text{(a) missing-feature gap}}.
\end{aligned} \tag{20}$$

Next, we bound τ_1 in (20), which we do in two steps below. First, we record a first-order Taylor expansion of f with an integral remainder term. For any $a, b \in \mathbb{R}^{\tilde{d}_0}$, we get

$$f(b;\theta_f) = f(a;\theta_f) + \nabla f(a;\theta_f)(b-a) + r(a,b), \tag{21}$$

where $r(a,b) := \int_0^1 \left[\nabla f\big(a + t(b-a);\theta_f\big) - \nabla f(a;\theta_f)\right](b-a)\,\mathrm{d}t$. Taking the norm and applying the triangle inequality inside the integral, we get

$$\|r(a,b)\|_2 \le \int_0^1 \left\|\nabla f\big(a + t(b-a);\theta_f\big) - \nabla f(a;\theta_f)\right\|_{\mathrm{op}} \|b-a\|_2\,\mathrm{d}t. \tag{22}$$

Under Assumption 1, $\left\|\nabla f\big(a_1;\theta_f\big) - \nabla f(a_2;\theta_f)\right\|_{\mathrm{op}} \le \beta_f\|a_1 - a_2\|_2$ for all $a_1, a_2 \in \mathbb{R}^{\tilde{d}_0}$; setting $a_1 = a + t(b-a)$ and $a_2 = a$, we get $\left\|\nabla f\big(a + t(b-a);\theta_f\big) - \nabla f(a;\theta_f)\right\|_{\mathrm{op}} \le \beta_f\|t(b-a)\|_2 = \beta_f t\|b-a\|_2$. Substituting this bound into (22) and evaluating the integral, we get

$$\|r(a,b)\|_2 \le \int_0^1 \beta_f t\|b-a\|_2^2\,\mathrm{d}t = \frac{\beta_f}{2}\|b-a\|_2^2. \tag{23}$$

Second, we relate $f(E_i x_i^n;\theta_f)$ and $f\big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\big)$, which is included in τ_1, by a Taylor expansion around each other. However, doing so alone leaves a first-order gradient term that does not vanish, so the relation cannot be closed using only these two points. We therefore additionally expand f around 0, which lets us eliminate the first-order term. Applying (21) with $a = \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n$ and $b = E_i x_i^n$ for each $i \in \mathcal{S}_k$, and separately with $a = \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n$ and $b = 0$, we have

$$f(E_i x_i^n;\theta_f) = f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big) + \nabla f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\Big(E_i x_i^n - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big) + r\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n, E_i x_i^n\Big), \tag{24}$$

$$f(0;\theta_f) = f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big) + \nabla f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\Big(0 - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big) + r\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n, 0\Big). \tag{25}$$

We choose the weights $w_0 := 1 - \frac{s_k}{m_k}$ and $w_i := \frac{1}{m_k}$ for $i \in \mathcal{S}_k$ specifically so that $\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n$ is recovered as their weighted combination of 0 and $\{E_i x_i^n\}_{i\in\mathcal{S}_k}$:

$$w_0 + \sum_{i\in\mathcal{S}_k} w_i = 1, \qquad w_0 \cdot 0 + \sum_{i\in\mathcal{S}_k} w_i E_i x_i^n = \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n.$$

This is precisely what makes the first-order term vanish when we combine (25) and (24) with these same weights: taking $w_0 \times (25) + \sum_{i\in\mathcal{S}_k} w_i \times (24)$, the first-order term on the right-hand side becomes

$$\nabla f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\left[w_0\Big(0 - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big) + \sum_{i\in\mathcal{S}_k} w_i\Big(E_i x_i^n - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big)\right] = 0,$$

where the bracketed term vanishes because it is exactly $w_0 \cdot 0 + \sum_{i\in\mathcal{S}_k} w_i E_i x_i^n$ minus $\big(w_0 + \sum_{i\in\mathcal{S}_k} w_i\big) \cdot \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n$, i.e., $\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n$ minus itself. Thus, the weighted combination of (25) and (24) reduces to

$$\begin{aligned}&\Big(1 - \frac{s_k}{m_k}\Big) f(0;\theta_f) + \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\\ &= \Big(1 - \frac{s_k}{m_k}\Big) r\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n, 0\Big) + \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} r\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n, E_i x_i^n\Big).\end{aligned} \tag{26}$$

Taking L_2 norms of (26) and applying (23) together with the triangle inequality (using $|w_0| = |1 - \frac{s_k}{m_k}|$ for the possibly negative weight w_0), we get

$$\begin{aligned}&\left\|\Big(1 - \frac{s_k}{m_k}\Big) f(0;\theta_f) + \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\right\|_2\\ &\le \frac{\beta_f}{2}\left[\Big|1 - \frac{s_k}{m_k}\Big| \Big\|\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big\|_2^2 + \frac{1}{m_k}\sum_{i\in\mathcal{S}_k}\Big\|E_i x_i^n - \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n\Big\|_2^2\right].\end{aligned} \tag{27}$$

Writing

$$\begin{aligned}&\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\\ &= \left[\Big(1 - \frac{s_k}{m_k}\Big) f(0;\theta_f) + \frac{1}{m_k}\sum_{i\in\mathcal{S}_k} f(E_i x_i^n;\theta_f) - f\Big(\frac{1}{m_k}\sum_{i\in\mathcal{S}_k} E_i x_i^n;\theta_f\Big)\right] - \Big(1 - \frac{s_k}{m_k}\Big) f(0;\theta_f)\end{aligned}$$

and applying the triangle inequality together with (27), we obtain

$$\tau_1 = \left\| \frac{1}{m_k} \sum_{i \in \mathcal{S}_k} f(E_i x_i^n; \theta_f) - f\Big(\frac{1}{m_k} \sum_{i \in \mathcal{S}_k} E_i x_i^n; \theta_f \Big) \right\|_2$$
$$\leq \underbrace{\left| 1 - \frac{s_k}{m_k} \right| \|f(0; \theta_f)\|_2}_{\text{(c) zero-input offset}} + \underbrace{\frac{\beta_f}{2} \left[\frac{1}{m_k} \sum_{i \in \mathcal{S}_k} \left\| E_i x_i^n - \frac{1}{m_k} \sum_{i \in \mathcal{S}_k} E_i x_i^n \right\|_2^2 + \left| 1 - \frac{s_k}{m_k} \right| \left\| \frac{1}{m_k} \sum_{i \in \mathcal{S}_k} E_i x_i^n \right\|_2^2 \right]}_{\text{(d) curvature-induced gap}}. \tag{28}$$

Substituting (28) into (20) completes the decomposition of the norm in (18) into (a)–(e). Taking expectations over $\mathcal{S}_k \sim q_k$ and $(x, t_k) \sim \mathcal{D}_k$, averaging over clients, and summing the resulting five terms yields $\Delta^{(Q,\mathcal{E},m)}(\theta_f)$, which proves Proposition 1. □

A.2 Proof of Proposition 2

Proof. Substituting $Q = I_{d_0}$, $E_k = Z_k$, and $m_k = s_k$ into Proposition 1 makes $\Delta_{\text{rec}}^{(Q,\mathcal{E},m)}(\theta_f)$ and $\Delta_{\text{offset}}^{(m)}(\theta_f)$ vanish identically, for every θ_f (as explained in (T1) and (T2), Section 3.2), which proves Proposition 2. □

B Computation of (T3) Extraction-Curvature Regularization

Algorithm 1 details how the estimator $\widehat{\mathcal{L}}_{\text{curv}}(\theta)$ in (13) is computed via finite differences, without explicitly constructing the input-Hessian tensor.

Algorithm 1 Approximated estimation of (T3) extraction-curvature regularizer $\widehat{\mathcal{L}}_{\text{curv}}(\theta)$

```
Initialization L̂_curv ← 0
Set minibatch sampling index B and step size μ
for k = 1, ..., K do
    for n ∈ B do          /* minibatch sampled feature blocks */
        y ← Z_k x_k^n
        for j = 1, ..., J do
            Set Rademacher vectors u_j, v_j ∈ {−1, +1}^{d_0}
            φ ← [f_k(y+μu_j+μv_j) − f_k(y+μu_j−μv_j) − f_k(y−μu_j+μv_j)
                  +f_k(y−μu_j−μv_j)]/(4μ²)     /* four forward passes through f_k */
            L̂_curv ← L̂_curv + ‖φ‖_2^2
        end for
    end for
end for
Return 1/(K|B|J) L̂_curv
```

C Additional Experiments

C.1 Model Architecture Information

(B1) CIFAR-10 Classification

Extraction model (3-layer CNN):

- Conv2d(3, 24, k3, s1, p1) $\rightarrow$ ReLU
- Conv2d(24, 48, k3, s1, p1) $\rightarrow$ ReLU $\rightarrow$ Dropout(p=0.3)
- Conv2d(48, 64, k3, s1, p1)

Fusion model (ResNet-18-base):

- Conv2d(64, 64, k7, s2, p2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPool(k3, s2, p1)
- Layer1: [Conv2d(64,64,k3,s1) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(64,64,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer2: [Conv2d(c_{in}=64,128,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(128,128,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer3: [Conv2d(c_{in}=128,256,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(256,256,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer4: [Conv2d(c_{in}=256,512,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(512,512,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- AvgPool2d(1,1) $\rightarrow$ Dropout(p=0.7) $\rightarrow$ Linear(512, 10)

(B2) CIFAR-100 Classification

Extraction model (3-layer CNN):

- Conv2d(3, 24, k3, s1, p1) $\rightarrow$ ReLU
- Conv2d(24, 48, k3, s1, p1) $\rightarrow$ ReLU $\rightarrow$ Dropout(p=0.3)
- Conv2d(48, 64, k3, s1, p1)

Fusion model (ResNet-18-base):

- Conv2d(64, 64, k7, s2, p2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPool(k3, s2, p1)
- Layer1: [Conv2d(64,64,k3,s1) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(64,64,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer2: [Conv2d(c_{in}=64,128,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(128,128,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer3: [Conv2d(c_{in}=128,256,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(256,256,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- Layer4: [Conv2d(c_{in}=256,512,k3,s2) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ Conv2d(512,512,k3,s1) $\rightarrow$ BatchNorm] $\times 2$
- AvgPool2d(1,1) $\rightarrow$ Dropout(p=0.7) $\rightarrow$ Linear(512, 100)

(B3) HAPT Classification

Extraction model (2-layer MLP):

- Linear(input size, 128) $\rightarrow$ ReLU
- Linear(128, 13)

Fusion model (3-layer MLP):

- Linear(130, 130) $\rightarrow$ ReLU
- Linear(130, 130) $\rightarrow$ ReLU $\rightarrow$ Dropout(p=0.7)
- Linear(130, 12)

C.2 HYPERPARAMETER TUNING

(B1) CIFAR-10 Classification

Common:

- Number of clients: $K \in \{4, 8, 16\}$

Optimizer (AdamW):

- Initial learning rate: $\{1\times10^{-2}, 1\times10^{-3}, 1\times10^{-4}\}$
- Learning schedule: cosine annealing to $1/10$ of the initial learning rate
- Number of iterations: 400
- First-order moment (β_1): 0.9
- Second-order moment (β_2): 0.999
- Weight decay: 1×10^{-3}

PIGR-VFL:

- Weight of (T3) extraction-curvature regularization: $\gamma_{\text{curv}} \in \{0.0, 1\times10^{-3}, 5\times10^{-3}\}$
- Weight of (T4) proximal extraction-output regularization: $\gamma_{\text{prox}} \in \{0.0, 0.5, 1.0\}$
- Number of probe pairs: J=1
- Step size μ=1×10^{-2}

Table 2 shows the highest test accuracy with the selected hyperparameters.

Table 2: Highest test accuracy and selected hyperparameters. Since each method is extended so that each client holds its own fusion model, K prediction results are obtained; their mean and standard deviation are reported.

Benchmark test	(B1) CIFAR-10		
# of clients K	4	8	16
(M1) Vanilla VFL	$0.8641_{\pm0.0008}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8422_{\pm0.0016}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8146_{\pm0.0028}$ (lr: 1×10^{-4}, wd: 1×10^{-3})
(M2) LASER-VFL	$0.8611_{\pm0.0013}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8398_{\pm0.0014}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8113_{\pm0.0014}$ (lr: 1×10^{-3}, wd: 1×10^{-3})
(M3) PlugVFL	$0.8698_{\pm0.0006}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8557_{\pm0.0023}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.8479_{\pm0.0024}$ (lr: 1×10^{-3}, wd: 1×10^{-3})
(M4) Local	$0.7472_{\pm0.0054}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.6319_{\pm0.0525}$ (lr: 1×10^{-3}, wd: 1×10^{-3})	$0.5109_{\pm0.0636}$ (lr: 1×10^{-3}, wd: 1×10^{-3})
(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0)	$0.8994_{\pm0.0029}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)	$0.8806_{\pm0.0029}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)	$0.8830_{\pm0.0023}$ (lr: 1×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)
(P2) PIGR-VFL (γ_{curv}=0, γ_{prox}>0)	**0.9003**$_{\pm0.0019}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 1.0)	$0.8800_{\pm0.0031}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 1.0)	$0.8835_{\pm0.0016}$ (lr: 1×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 1.0)
(P3) PIGR-VFL (γ_{curv}>0, γ_{prox}=0)	$0.8993_{\pm0.0022}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 5×10^{-3}, γ_{prox} : 0.0)	$0.8810_{\pm0.0028}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)	$0.8838_{\pm0.0029}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)
(P4) PIGR-VFL (γ_{curv}>0, γ_{prox}>0)	$0.8987_{\pm0.0013}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 1.0)	**0.8815**$_{\pm0.0015}$ (lr: 1×10^{-3}, wd: 1×10^{-3}, γ_{curv} : 5×10^{-3}, γ_{prox} : 0.5)	**0.8843**$_{\pm0.0028}$ (lr: 1×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.5)

(B2) CIFAR-100 Classification

Common:

- Number of clients: $K \in \{4, 8, 16\}$

Optimizer (AdamW):

- Initial learning rate: $\{5\times10^{-3}, 5\times10^{-4}, 5\times10^{-5}\}$
- Learning schedule: cosine annealing to $1/10$ of the initial learning rate
- Number of iterations: 500
- First-order moment (β_1): 0.9
- Second-order moment (β_2): 0.999
- Weight decay: 1×10^{-3}

PIGR-VFL:

- Weight of (T3) extraction-curvature regularization: $\gamma_{\text{curv}} \in \{0.0, 1\times10^{-3}, 5\times10^{-3}\}$
- Weight of (T4) proximal extraction-output regularization: $\gamma_{\text{prox}} \in \{0.0, 0.5, 1.0\}$
- Number of probe pairs: J=1
- Step size μ=1×10^{-2}

Table 3 shows the highest test accuracy with the selected hyperparameters.

Table 3: Highest test accuracy and selected hyperparameters. Since each method is extended so that each client holds its own fusion model, K prediction results are obtained; their mean and standard deviation are reported.

Benchmark test	(B2) CIFAR-100		
# of clients K	4	8	16
(M1) Vanilla VFL	$0.5394_{\pm 0.0054}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.5530_{\pm 0.0046}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.5378_{\pm 0.0044}$ (lr: 5×10^{-4}, wd: 1×10^{-3})
(M2) LASER-VFL	$0.5268_{\pm 0.0023}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.5495_{\pm 0.0032}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.5380_{\pm 0.0027}$ (lr: 5×10^{-4}, wd: 1×10^{-3})
(M3) PlugVFL	$0.5170_{\pm 0.0011}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.5030_{\pm 0.0013}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.4747_{\pm 0.0041}$ (lr: 5×10^{-4}, wd: 1×10^{-3})
(M4) Local	$0.4288_{\pm 0.0042}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.3379_{\pm 0.0431}$ (lr: 5×10^{-4}, wd: 1×10^{-3})	$0.2511_{\pm 0.0504}$ (lr: 5×10^{-4}, wd: 1×10^{-3})
(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0)	$0.6329_{\pm 0.0020}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)	**0.6407**$_{\pm 0.0026}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)	$0.6145_{\pm 0.0029}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)
(P2) PIGR-VFL (γ_{curv}=0, γ_{prox}>0)	$0.6334_{\pm 0.0029}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.5)	$0.6391_{\pm 0.0048}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.5)	$0.6143_{\pm 0.0029}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 1.0)
(P3) PIGR-VFL (γ_{curv}>0, γ_{prox}=0)	**0.6367**$_{\pm 0.0038}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)	$0.6379_{\pm 0.0061}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 5×10^{-3}, γ_{prox} : 0.0)	$0.6154_{\pm 0.0039}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)
(P4) PIGR-VFL (γ_{curv}>0, γ_{prox}>0)	$0.6332_{\pm 0.0028}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.5)	$0.6384_{\pm 0.0025}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.5)	**0.6167**$_{\pm 0.0031}$ (lr: 5×10^{-4}, wd: 1×10^{-3}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.5)

(B3) HAPT Classification

Common:

- Number of clients: $K \in \{20, 40\}$

Optimizer (Adam):

- Initial learning rate: $\{1\times10^{-3}, 7.5\times10^{-4}, 5\times10^{-4}, 2.5\times10^{-4}, 1\times10^{-4}\}$
- Learning schedule: cosine annealing to $1/10$ of the initial learning rate
- Number of iterations: 2,000
- First-order moment (β_1): 0.9
- Second-order moment (β_2): 0.999

PIGR-VFL:

- Weight of (T3) extraction-curvature regularization: $\gamma_{\text{curv}} \in \{0.0, 1\times10^{-3}, 5\times10^{-3}\}$
- Weight of (T4) proximal extraction-output regularization: $\gamma_{\text{prox}} \in \{0.0, 0.5, 1.0\}$
- Number of probe pairs: J=1
- Step size μ=1×10^{-2}

Table 4 shows the highest test accuracy with the selected hyperparameters.

Table 4: Highest test accuracy and selected hyperparameters. Since each method is extended so that each client holds its own fusion model, K prediction results are obtained; their mean and standard deviation are reported.

Benchmark test	(B3) HAPT	
# of clients K	20	40
(M1) Vanilla VFL	$0.9555_{\pm0.0022}$ (lr: 2.5×10^{-4})	$0.9560_{\pm0.0027}$ (lr: 5×10^{-4})
(M2) LASER-VFL	$\mathbf{0.9572}_{\pm0.0021}$ (lr: 1×10^{-3})	$0.9543_{\pm0.0015}$ (lr: 5×10^{-4})
(M3) PlugVFL	$0.9500_{\pm0.0022}$ (lr: 5×10^{-4})	$0.9523_{\pm0.0017}$ (lr: 5×10^{-4})
(M4) Local	$0.7710_{\pm0.0620}$ (lr: 1×10^{-4})	$0.6862_{\pm0.0700}$ (lr: 1×10^{-4})
(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0)	$0.9528_{\pm0.0020}$ (lr: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 0.0)	$\mathbf{0.9575}_{\pm0.0013}$ (lr: 5×10^{-4}, γ_{curv} : 0.0, γ_{prox} : 0.0)
(P2) PIGR-VFL (γ_{curv}=0, γ_{prox}>0)	$0.9528_{\pm0.0020}$ (lr: 1×10^{-3}, γ_{curv} : 0.0, γ_{prox} : 1.0)	$\mathbf{0.9575}_{\pm0.0013}$ (lr: 5×10^{-4}, γ_{curv} : 0.0, γ_{prox} : 1.0)
(P3) PIGR-VFL (γ_{curv}>0, γ_{prox}=0)	$0.9528_{\pm0.0011}$ (lr: 7.5×10^{-4}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)	$0.9564_{\pm0.0016}$ (lr: 5×10^{-4}, γ_{curv} : 1×10^{-3}, γ_{prox} : 0.0)
(P4) PIGR-VFL (γ_{curv}>0, γ_{prox}>0)	$0.9528_{\pm0.0011}$ (lr: 7.5×10^{-4}, γ_{curv} : 5×10^{-3}, γ_{prox} : 1.0)	$0.9564_{\pm0.0016}$ (lr: 2.5×10^{-4}, γ_{curv} : 5×10^{-3}, γ_{prox} : 1.0)

C.3 Ablation study to investigate partial-mean aggregation v.s. summation aggregation.

As for an ablation study, we empirically investigate the effectiveness of the choice of m_k. In Proposition 1, we show that (c) the zero-input offset $\Delta^{(m)}_{\text{offset}}(\theta_f) = \frac{1}{K}\sum_{k=1}^{K} \beta_{\ell_k}\beta_{g_k}\,\mathbb{E}_{\mathcal{S}_k}[|1-\frac{s_k}{m_k}|]\|f(0;\theta_f)\|_2$ is one of the factors contributing to the partition-induced gap. In Section 3.2, we select (T2) participant-mean aggregation ($m_k = s_k$) to eliminate (c) the zero-input offset. Without the training regularizations (T3)–(T4), PIGR-VFL with the minimal requirements is as follows:

(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0): the d_k-dimensional raw feature block is first augmented, via (T1) coordinate-preserving embedding, into the d_0-dimensional space with zero-padding (E_k=Z_k, Q=I_{d_0}), and is then mapped to a local representation by each client's extraction model. At the server, (T2) participant-mean aggregation of the K local representations (m_k=s_k) is applied. Optional training regularizations are not utilized (γ_{curv}=0, γ_{prox}=0).

On the other hand, if we instead select summation aggregation ($m_k = 1$) in place of (T2) participant-mean aggregation ($m_k = s_k$), (c) the zero-input offset cannot be eliminated, and (d) the curvature-induced gap is no longer unbiased. To empirically investigate this difference in aggregation choice, we prepare the following ablation method.

(A1) Ablation 1 (γ_{curv}=0, γ_{prox}=0): The only difference from (P1) is that summation aggregation (m_k=1) is applied, instead of (T2) participant-mean aggregation of the K local representations (m_k=s_k).

We prepare two evaluation metrics: the first is test accuracy, and the second is the zero-input-output norm, defined by

$$\zeta := \frac{1}{K}\sum_{k=1}^{K} \|f_k(0;\theta_{f_k})\|_2.$$

If ζ is small, the L_2-norm of the extraction model's output at the zero input is suppressed; namely, predictive performance may not be affected by the choice of m_k.

Table 5 shows the test accuracy and the zero-input-output norm for the image classification tasks. In terms of test accuracy, the difference between (P1) and (A1) is slight; that is, there is little difference between participant-mean aggregation (m_k=s_k) and summation aggregation (m_k=1). This can be attributed to ζ taking a small value, which is likely due to the general tendency of neural networks with ReLU-based activation functions to be trained so as to produce outputs close to zero near the origin. Note, however, that this is merely incidental to the present setting: depending on the model architecture and the task being solved, there may well be settings in which ζ takes a large value, and under such scenarios, the difference between participant-mean aggregation (m_k=s_k) and summation aggregation (m_k=1) could substantially affect the predictive results. As theoretically suggested by Proposition 1, we nonetheless recommend choosing participant-mean aggregation (m_k=s_k), as it eliminates (c) the zero-input offset and renders (d) curvature-induced gap unbiased.

Table 5: Highest test accuracy and zero-input-output norm

	(B1) CIFAR-10			(B2) CIFAR-100		
# of clients K	4	8	16	4	8	16
(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0)	$0.8994_{\pm 0.0029}$ (ζ=0.45)	$0.8806_{\pm 0.0029}$ (ζ=0.22)	$0.8830_{\pm 0.0023}$ (ζ=0.15)	$0.6329_{\pm 0.0020}$ (ζ=0.63)	$0.6407_{\pm 0.0026}$ (ζ=0.35)	$0.6145_{\pm 0.0029}$ (ζ=0.16)
(A1) Ablation 1 (γ_{curv}=0, γ_{prox}=0)	$0.8993_{\pm 0.0009}$ (ζ=0.46)	$0.8760_{\pm 0.0025}$ (ζ=0.19)	$0.8535_{\pm 0.0025}$ (ζ=0.15)	$0.6360_{\pm 0.0010}$ (ζ=0.51)	$0.6390_{\pm 0.0055}$ (ζ=0.30)	$0.6154_{\pm 0.0038}$ (ζ=0.24)

C.4 COMPUTATION RESOURCES.

We used the following machine for (B1) CIFAR-10 and (B2) CIFAR-100 classification tasks:

- GPU: NVIDIA RTX 6000 Ada $\times$ 8
- CPU: AMD EPYC 9354 32-Core Processor
- Memory: 1.5 TiB

We used the following machine for the (B3) HAPT classification task:

- GPU: NVIDIA GeForce RTX 2080 Ti $\times$ 10
- CPU: Intel(R) Xeon(R) Gold 5215 CPU @ 2.50GHz
- Memory: 0.19 TiB

C.5 COMPUTATIONAL RUNTIME INVESTIGATION.

For the (B1) CIFAR-10 classification task, we investigated the computational runtime. As shown in Table 6, incorporating (T1) coordinate-preserving embedding increases the runtime by approximately $1.5\times$, and incorporating (T3) extraction-curvature regularization increases it by nearly $3\times$, whereas the effect of (T4) proximal extraction-output regularization on runtime is negligible. Considering both the predictive performance in Table 1 and the runtime in Table 6, (T3) offers only a marginal accuracy gain at a substantially higher computational cost than (T4). We therefore empirically found that, for practical use, forgoing (T3) extraction-curvature regularization may be a reasonable choice, and recommend enabling it only when the modest additional gain is worth the added cost for the application at hand.

Table 6: Training time (in seconds) for each method on (B1) CIFAR-10.

# of clients K	4	8	16
(M1) Vanilla VFL	7021	13642	28129
(M2) LASER-VFL	7213	13391	27105
(P1) PIGR-VFL (γ_{curv}=0, γ_{prox}=0)	10359	23184	55517
(P2) PIGR-VFL (γ_{curv}=0, γ_{prox}>0)	10934	22937	55253
(P3) PIGR-VFL (γ_{curv}>0, γ_{prox}=0)	19429	40615	90782
(P4) PIGR-VFL (γ_{curv}>0, γ_{prox}>0)	20161	42291	91848

D LIMITATIONS AND FUTURE WORK

The first limitation of our theoretical analysis is that Propositions 1 and 2 are two-sided bounds: reducing the partition-induced gap does not guarantee an improvement in population risk. As discussed below Proposition 1, such worst-case guarantees are nonetheless standard in non-convex extraction/fusion model settings. Empirically, however, PIGR-VFL—which incorporates (T1) coordinate-preserving embedding and (T2) participant-mean aggregation as required components—consistently outperforms baseline methods that adopt only a subset of these techniques (e.g., LASER-VFL, which employs only (T2)), particularly on the image classification tasks.

The second limitation concerns the approximate computation of (T3) extraction-curvature regularization. As explained in Section 3.2, we approximate it via the Frobenius norm instead of the operator norm, and use a finite-difference estimator with a single probe pair ($J=1$) to reduce computational cost. These approximations may prevent (T3) from achieving the intended regularization effect—constraining curvature without unduly restricting the extraction model's expressiveness—and developing a better approximation without using heavy computational cost is left as future work.

For future work, extending our analysis to a broader class of aggregation rules, such as attention-based aggregation, is a promising direction.