SHARED YET DISTINCT: RETHINKING LABEL-SCARCE VERTICAL FEDERATED LEARNING WITH HETEROGENEITY-AWARE META-PROTOTYPE

Anonymous authors
Paper under double-blind review

ABSTRACT

Label scarcity and aligned-sample sparsity have emerged as major barriers to the practical deployment of supervised Vertical Federated Learning (VFL). Existing self-supervised VFL frameworks designed for label-scarce settings primarily rely on instance-discriminative representation learning. However, this learning paradigm tends to enforce rigid alignment of representations in the latent space, thereby inducing severe representation collapse in VFL. Moreover, these self-supervised VFL frameworks remain heavily dependent on frequent inter-participant representation exchange and knowledge-transfer schemes, further limiting scalability in heterogeneous settings while introducing computational and security overhead. To overcome these fundamental limitations, we propose the Heterogeneity-Aware Meta-Prototype Learning (HAMPL) framework, which establishes a new paradigm for self-supervised VFL by introducing meta-prototypes as safer intermediaries. By accounting for inter-participant heterogeneity, HAMPL enables each participant model to learn shared and generalizable knowledge while preserving its distinctive participant-specific information, thereby exploiting the complementarity of heterogeneous feature subsets across participants. Furthermore, meta-prototypes act as a knowledge bridge across the two pre-training stages, naturally mitigating catastrophic forgetting that can arise from the two-stage learning paradigm. We further provide a theoretical characterization and a security analysis of HAMPL. Extensive experiments indicate that HAMPL achieves superior performance in label-scarce VFL and remains effective under heterogeneous settings.

1 INTRODUCTION

Vertical federated learning (VFL) enables participants holding complementary feature subsets to jointly train a model without sharing raw data. Its practical deployment, however, is constrained by scarce labels and limited sample overlap across participants (Ye et al., 2025).Self-supervised VFL (SSVFL) alleviates these constraints by exploiting aligned labeled, aligned unlabeled, and unaligned samples for representation learning (Liu et al., 2024).

Existing SSVFL frameworks (He et al., 2025; Chen et al., 2026) largely inherit instance-discriminative representation learning (He et al., 2020; Chen & He, 2021) from centralized settings, enforcing rigid alignment between participant representations and shared anchors constructed by fusing heterogeneous views. This overlooks a fundamental characteristic of VFL: different participants hold distinct and complementary sets of real-world features corresponding to the same entity, rather than artificially augmented views of the same instance.

As shown in Figure 1, centralized instance-discriminative methods learn augmentation invariance by aligning two augmented views of an instance within a common latent space, typically using online and target encoders with identical architectures and initial parameters. In VFL, however, participants process complementary feature subsets using local encoders that may differ in architecture and initialization. Even with the same output dimensionality, these encoders can map the feature subsets of the same entity to markedly different locations in independently learned latent spaces, particularly during the early stages of training. Given these discrepancies, the shared anchor may conflate hetero-

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.

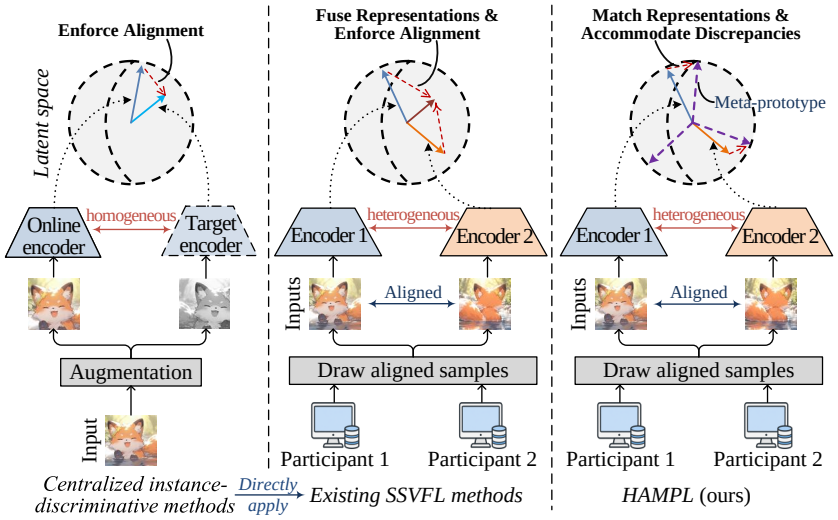


Figure 1: Conceptual overview of the HAMPL framework. Projectors are omitted for clarity.

geneous participant-specific semantics rather than faithfully capture the complementary information encoded by different feature subsets. Enforcing alignment with such an anchor can impede convergence and induce representation collapse, thereby eroding valuable participant-specific semantic information.

Some Horizontal Federated Learning (HFL) methods (Zhang et al., 2026; D'Acunto et al., 2026) address discrepancies between independently learned latent spaces using orthogonal transformations or isometric mappings, treating them as coordinate-system mismatches. In VFL, however, these discrepancies arise not only from coordinate misalignment but also from intrinsic participant-specific semantics encoded by complementary feature subsets. These intrinsic semantic differences cannot be reconciled by such transformations alone. Moreover, representation exchange, partial model aggregation, and separate models for knowledge transfer in existing SSVFL frameworks (He et al., 2025; Chen et al., 2026) collectively increase privacy risks, limit architectural flexibility in heterogeneous VFL settings, and introduce additional computational overhead.

To overcome these limitations, we fundamentally rethink the learning paradigm for label-scarce VFL and propose the Heterogeneity-Aware Meta-Prototype Learning (HAMPL) framework, which introduces meta-prototypes to establish a new SSVFL paradigm without relying on any of the aforementioned mechanisms. Specifically, unlike conventional FL prototypes constructed through forward aggregation, The meta-prototypes in HAMPL are optimized via backpropagation, allowing them to absorb representational knowledge from individual participants. These meta-prototypes serve as intermediaries across participants, linking the two alternating stages in the pre-training of their local models. Rather than forcing participant-specific representations into alignment, HAMPL employs heterogeneity-aware mechanism to match between these representations and the meta-prototypes, enabling bidirectional learning between the meta-prototypes and participant models while accommodating cross-participant semantic discrepancies. By explicitly accounting for both cross-participant heterogeneity and long-tailed data distributions, HAMPL enables each local model to acquire shared, generalizable knowledge while preserving participant-specific information, thereby fully exploiting the complementarity of heterogeneous feature subsets.

Our contributions are as follows:

- We identify a fundamental incompatibility between the centralized instance-discriminative representation learning paradigm and VFL, and propose HAMPL framework that enables effective knowledge sharing while preserving distinct participant-specific semantics.
- We develop a two-stage alternating pre-training scheme in which paticipants interacts exclusively with the meta-prototypes and assignment matrices rather than exposing intermediate representations and model parameters. Meanwhile, the meta-prototypes serve as a

knowledge bridge between the two stages, naturally mitigating catastrophic forgetting that can arise from the two-stage learning paradigm.

- We further provide a mutual-information characterization and analyze the security threats to HAMPL and its privacy boundaries. Performance and privacy evaluations across four benchmarks demonstrate its robustness and superiority.

We defer a detailed discussion of related work to Appendix A.3, while retaining the discussion and analysis most relevant to our core motivation in this introduction.

2 METHODOLOGY

2.1 PROBLEM DEFINITION

We consider P participants indexed by $p \in \mathcal{P} = \{1, \ldots, P\}$. Participant p holds $\mathcal{D}^p = \{(i, \mathbf{x}_i^p) \mid i \in \mathcal{I}^p\}$, where $\mathcal{I}^p$ is its entity-ID set and $\mathbf{x}_i^p \in \mathbb{R}^{d_p}$ is its feature vector for entity i. Different participants hold distinct feature subsets. Participant 1 is the active participant and owns labels for only a small subset of shared entities, while the remaining participants are passive. Let $\mathcal{I}_\mathrm{a} = \bigcap_{p=1}^{P} \mathcal{I}^p$ be the aligned ID set, partitioned as $\mathcal{I}_\mathrm{a} = \mathcal{I}_\mathrm{al} \dot\cup \mathcal{I}_\mathrm{au}$ into labeled and unlabeled subsets. Participant p additionally holds private IDs $\mathcal{I}_\mathrm{u}^p = \mathcal{I}^p \setminus \mathcal{I}_\mathrm{a}$.

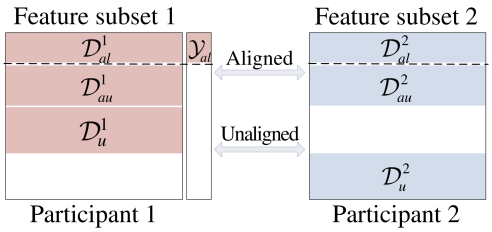


Figure 2: Sample composition in a two-participant label-scarce VFL scenario.

Using these ID sets, we define the three data subsets of each participant $p \in \mathcal{P}$ and the label set held by the active participant as follows:

$$\mathcal{D}_\mathrm{al}^p = \{(i, \mathbf{x}_i^p) \mid i \in \mathcal{I}_\mathrm{al}\},\ \mathcal{D}_\mathrm{au}^p = \{(i, \mathbf{x}_i^p) \mid i \in \mathcal{I}_\mathrm{au}\},\ \mathcal{D}_\mathrm{u}^p = \{(i, \mathbf{x}_i^p) \mid i \in \mathcal{I}_\mathrm{u}^p\},\ \mathcal{Y}_\mathrm{al} = \{(i, y_i) \mid i \in \mathcal{I}_\mathrm{al}\}. \quad (1)$$

Thus, $\mathcal{D}^p = \mathcal{D}_\mathrm{al}^p \dot\cup \mathcal{D}_\mathrm{au}^p \dot\cup \mathcal{D}_\mathrm{u}^p$, and $\mathcal{Y}_\mathrm{al}$ is held only by the active participant. We provide a visual illustration of the sample composition in a two-participant label-scarce VFL scenario in Figure 2.

2.2 HETEROGENEITY-AWARE META PROTOTYPE LEARNING FRAMEWORK

Consistent with conventional SSVFL frameworks, HAMPL follows a two-phase pipeline comprising self-supervised pre-training and downstream task-specific fine-tuning. We focus on the former and establish a novel pre-training paradigm centered on learnable heterogeneity-aware meta-prototypes, as illustrated in Figure 3. Specifically, the pre-training procedure alternates between a cross training stage and a local training stage, as summarized in Algorithm 4.

2.2.1 CROSS TRAINING STAGE

Each participant p maintains a local model with parameters θ^p, comprising an encoder $f^p : \mathbb{R}^{d_p} \to \mathbb{R}^{h_p}$ and a projector $\varphi^p : \mathbb{R}^{h_p} \to \mathbb{R}^R$, where h_p is the encoder output dimension and R is the shared projection dimension. The forward mapping is therefore $\varphi^p \circ f^p$. At the beginning of pre-training epoch t, the server broadcasts the global meta-prototype bank $\mathcal{C}_t^G = [c_t^1; \ldots; c_t^{N_C}] \in \mathbb{R}^{N_C \times R}$ (communication ① in Figure 3), where N_C is the number of prototypes and c_t^k is its k-th prototype row. Throughout, semicolons within brackets denote vertical stacking of row vectors. The current cross-stage bank is $\mathcal{C} = [c^1; \ldots; c^{N_C}]$, where c^k is its k-th prototype row. It is initialized as $\mathcal{C} \leftarrow \mathcal{C}_t^G$ and updated after each aligned mini-batch. Let $\mathcal{N}$ denote rowwise ℓ_2 normalization. Both projected representations and prototypes are normalized before computing their similarities.

For an aligned mini-batch $\mathcal{B}_\mathrm{a}$ of m_a entities, all participants use the same ordered IDs $i_1, \ldots, i_{m_\mathrm{a}} \in \mathcal{I}_\mathrm{a}$. Participant p holds $\mathcal{B}_\mathrm{a}^p = \{(i_b, \mathbf{x}_{i_b}^p)\}_{b=1}^{m_\mathrm{a}} \subseteq \mathcal{D}_\mathrm{al}^p \dot\cup \mathcal{D}_\mathrm{au}^p$, where b indexes an entity within the batch and $m_\mathrm{a} = |\mathcal{B}_\mathrm{a}^p|$. Passing this batch through the encoder f^p and projector φ^p, followed by rowwise normalization $\mathcal{N}$, yields the projected representation matrix $\mathbf{Z}_\mathrm{a}^p \in \mathbb{R}^{m_\mathrm{a} \times R}$. The participant then computes the prototype similarity matrix $\mathbf{S}_\mathrm{a}^p = \mathbf{Z}_\mathrm{a}^p \mathcal{C}^\top \in \mathbb{R}^{m_\mathrm{a} \times N_C}$.

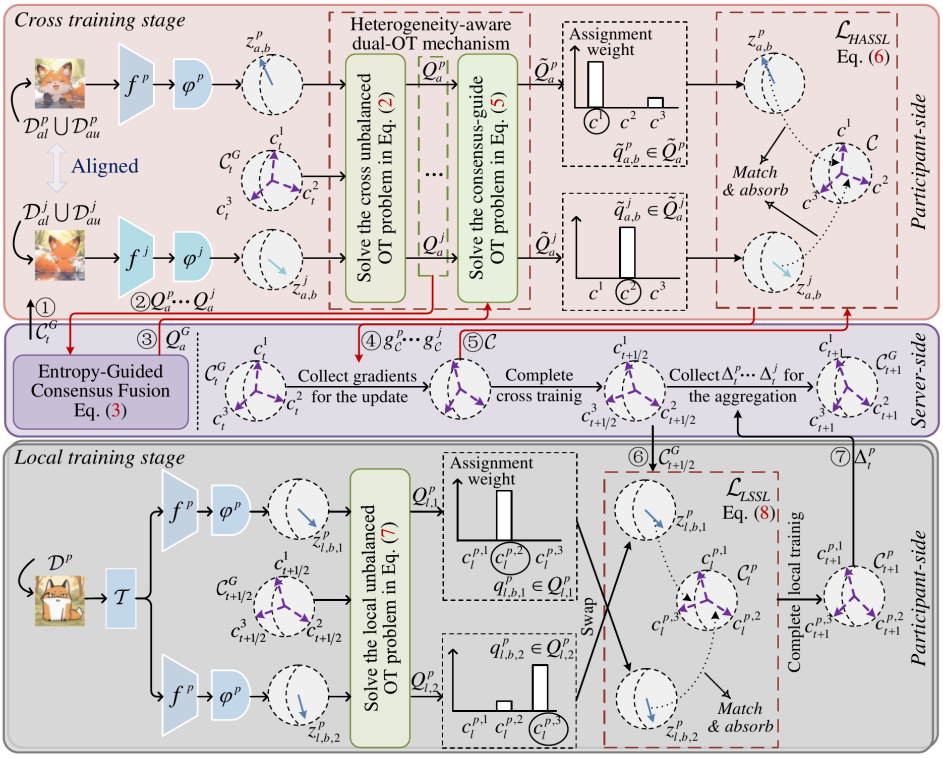


Figure 3: Overview of the HAMPL framework. HAMPL uses meta-prototypes to bridge the cross-training and local-training stages, enabling bidirectional learning between the meta-prototypes and participant models while accommodating cross-participant semantic discrepancies. During cross-training, participants learn from aligned samples and communicate with the server between consecutive mini-batches; during the subsequent local-training stage, they learn from private samples and communicate only at stage boundaries. Red and black arrows denote these two communication patterns, respectively.

Motivated by the limitations discussed in Appendix A.3, we propose a heterogeneity-aware dual-OT mechanism that combines cross unbalanced OT with consensus-guided OT to enable effective knowledge sharing while preserving distinct participant-specific semantics.

Cross unbalanced OT. Let $\mathbf{r} = \mathbf{1}_{N_C}/N_C$ be the prototype reference marginal, and $\mathbf{s}_{m_\mathrm{a}} = \mathbf{1}_{m_\mathrm{a}}/m_\mathrm{a}$ the sample reference marginal, where $\mathbf{1}_d$ is the length-d all-ones column vector. Rather than enforcing either marginal exactly in balanced OT (Caron et al., 2020), we formulate cross unbalanced OT problem as

$$\max_{\mathbf{Q}\in\mathbb{R}_+^{m_\mathrm{a}\times N_C}} \Big\{ \langle \mathbf{Q}, \mathbf{S}_\mathrm{a}^p \rangle_F + \varepsilon_\mathrm{a}\mathcal{H}(\mathbf{Q}) - \rho_\mathrm{a}\mathrm{KL}(\mathbf{Q}^\top \mathbf{1}_{m_\mathrm{a}} \| \mathbf{r}) - \nu_\mathrm{a}\mathrm{KL}(\mathbf{Q}\mathbf{1}_{N_C} \| \mathbf{s}_{m_\mathrm{a}}) \Big\}, \tag{2}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product, $\mathcal{H}$ is generalized entropy, and KL is the generalized Kullback-Leibler divergence. The positive coefficients ε_a, ρ_a, and ν_a are the regularization. In contrast to the squared ℓ_2 marginal penalties used by Liao et al. (2024) for UOT in HFL, we adopt generalized KL marginal penalties. We therefore obtain the assignment matrix $\mathbf{Q}_\mathrm{a}^p$ using the Sinkhorn–Knopp solver in Algorithm 1 (detailed in Appendix C.1), rather than the projected gradient descent used in their method.

Entropy-guided consensus fusion. We use the entropy of $\mathbf{Q}_\mathrm{a}^p$ as a label-free proxy for the uncertainty in the matching. Lower-entropy assignments are more decisive and therefore receive greater fusion weights, allowing clearer matching signals to contribute more strongly to the consensus. Thus, participants upload $\mathbf{Q}_\mathrm{a}^p$ to the server (communication ② in Figure 3), which row-normalizes

them as $\bar{\mathbf{Q}}_{\mathrm{a}}^{p}$ and computes entropy-based confidence $\mathbf{w}_{\mathrm{a}}^{p} \in [0,1]^{m_{\mathrm{a}}\times 1}$ for fusion:

$$\mathbf{w}_{\mathrm{a}}^{p} = \mathbf{1}_{m_{\mathrm{a}}} - \frac{\mathcal{H}_{\mathrm{row}}(\bar{\mathbf{Q}}_{\mathrm{a}}^{p})}{\log N_C}, \quad \mathcal{H}_{\mathrm{row}}(\mathbf{Q}) = -(\mathbf{Q}\odot\log\mathbf{Q})\mathbf{1}_{N_C}, \quad \mathbf{Q}_{\mathrm{a}}^{G} = \sum_{p=1}^{P} \frac{\mathbf{w}_{\mathrm{a}}^{p}}{\sum_{p'=1}^{P}\mathbf{w}_{\mathrm{a}}^{p'}} \odot \bar{\mathbf{Q}}_{\mathrm{a}}^{p}, \quad (3)$$

where $\mathcal{H}_{\mathrm{row}}$ denotes rowwise Shannon entropy and $\odot$ denotes the Hadamard product, with row weights broadcast across prototype columns. The consensus $\mathbf{Q}_{\mathrm{a}}^{G} \in \mathbb{R}_{+}^{m_{\mathrm{a}}\times N_C}$ satisfies $\mathbf{Q}_{\mathrm{a}}^{G}\mathbf{1}_{N_C} = \mathbf{1}_{m_{\mathrm{a}}}$ and is broadcast to all participants (communication ③ in Figure 3).

Consensus-guided OT. Each participant then utilizes the consensus as a soft prior to conduct Consensus-guided OT. Define the Consensus-guided OT set

$$\mathcal{U}_{m_{\mathrm{a}}} = \{\mathbf{Q} \in \mathbb{R}_{+}^{m_{\mathrm{a}}\times N_C} : \mathbf{Q}\mathbf{1}_{N_C} = \mathbf{s}_{m_{\mathrm{a}}},\ \mathbf{Q}^{\top}\mathbf{1}_{m_{\mathrm{a}}} = \mathbf{r}\}. \quad (4)$$

The refined assignment matrix is obtained by solving the following consensus-guided OT problem:

$$\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p} = m_{\mathrm{a}} \operatorname*{arg\,max}_{\mathbf{Q}\in\mathcal{U}_{m_{\mathrm{a}}}} \left\{ \langle \mathbf{Q}, \mathbf{S}_{\mathrm{a}}^{p}\rangle_F + \varepsilon_{\mathrm{g}}\mathcal{H}(\mathbf{Q}) - \lambda \mathrm{KL}\left(\mathbf{Q} \,\middle\|\, \frac{\mathbf{Q}_{\mathrm{a}}^{G}}{m_{\mathrm{a}}}\right)\right\}, \quad (5)$$

where $\varepsilon_{\mathrm{g}} > 0$ controls entropy and $\lambda \geq 0$ controls the consensus prior. The solver details are given in Appendix C.2 (Chizat et al., 2018). Rather than forcing all participants to use the consensus assignment matrix directly as a common target for consistency learning, this refinement uses the consensus as a soft prior to rematch participant representations to the meta-prototypes. This allows different meta-prototypes to absorb representational knowledge from different participants through subsequent bidirectional learning, capturing shared knowledge while preserving distinct participant-specific semantics and thereby accommodating cross-participant semantic discrepancies. Additionally, Appendix C.2 explains why we retain balanced marginals rather than introduce soft marginal constraints in consensus-guided OT.

Bidirectional cross learning. Let $\mathbf{z}_{\mathrm{a},b}^{p}$ and $\widetilde{\mathbf{q}}_{\mathrm{a},b}^{p}$ denote the b-th rows of $\mathbf{Z}_{\mathrm{a}}^{p}$ and $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p}$, respectively. Each participant p enables bidirectional learning between its local model θ^p and the meta-prototypes in $\mathcal{C}$ by minimizing the heterogeneity-aware self-supervised loss $\mathcal{L}_{\mathrm{HASSL}}^{p}$, defined as follows:

$$\mathcal{L}_{\mathrm{HASSL}}^{p} = -\frac{1}{m_{\mathrm{a}}}\sum_{b=1}^{m_{\mathrm{a}}}\sum_{k=1}^{N_C} \widetilde{q}_{\mathrm{a},b,k}^{p}\log\pi_k(\mathbf{z}_{\mathrm{a},b}^{p};\mathcal{C}), \quad \pi_k(\mathbf{z};\mathcal{C}) = \frac{\exp(\mathbf{z}(c^k)^{\top}/\tau)}{\sum_{k'=1}^{N_C}\exp(\mathbf{z}(c^{k'})^{\top}/\tau)}, \quad (6)$$

where $\pi_k(\mathbf{z};\mathcal{C})$ is the prediction probability of prototype k for a projected row representation $\mathbf{z}$, c^k is the k-th row of $\mathcal{C}$, $\widetilde{q}_{\mathrm{a},b,k}^{p}$ is component k of $\widetilde{\mathbf{q}}_{\mathrm{a},b}^{p}$, and τ is the prediction temperature.

In this federated setting, the meta-prototype bank $\mathcal{C}$ is shared across participants and can not update it independently. Therefore, for each aligned mini-batch, each participant uses backpropagation to update θ^p locally and compute the prototype gradient $\mathbf{g}_{\mathcal{C}}^{p} = \nabla_{\mathcal{C}}\mathcal{L}_{\mathrm{HASSL}}^{p}$. The server then collects and averages these gradients, updates the global meta-prototypes using stochastic gradient descent (SGD), normalizes each prototype to unit length, and broadcasts the updated $\mathcal{C}$ for the next mini-batch.. After completing cross training, the final meta-prototypes is recorded as $\mathcal{C}_{t+1/2}^{G}$. These operations correspond to communications ④ and ⑤ in Figure 3.

2.2.2 Local Training Stage

At the start of local training in pre-training epoch t, the server broadcasts $\mathcal{C}_{t+1/2}^{G} = [c_{t+1/2}^{1};\ldots;c_{t+1/2}^{N_C}] \in \mathbb{R}^{N_C\times R}$ to initialize each participant's meta-prototypes (communication ⑥ in Figure 3). Participant p continues training the same local model θ^p on its complete private dataset $\mathcal{D}^p$, including aligned and unaligned samples. For simplicity, we denote participant p's current prototype bank by $\mathcal{C}_l^p = [c_l^{p,1};\ldots;c_l^{p,N_C}] \in \mathbb{R}^{N_C\times R}$, where $c_l^{p,k}$ is its k-th local prototype row.

Given a local mini-batch $\mathcal{B}_l^p = \{(i_b, \mathbf{x}_{i_b}^p)\}_{b=1}^{m_l^p} \subseteq \mathcal{D}^p$, where $i_b \in \mathcal{I}^p$ and $m_l^p = |\mathcal{B}_l^p|$, participant p independently applies the augmentation strategy $\mathcal{T}$ to generate two view $\mathcal{V}_{l,1}^p$ and $\mathcal{V}_{l,2}^p$. For each view $j \in \{1,2\}$, passing $\mathcal{V}_{l,j}^p$ through f^p and φ^p, followed by rowwise normalization $\mathcal{N}$, yields the

projected representation matrix $\mathbf{Z}_{l,j}^p \in \mathbb{R}^{m_l^p \times R}$. The corresponding prototype similarity matrix is $\mathbf{S}_{l,j}^p = \mathbf{Z}_{l,j}^p (\mathcal{C}_l^p)^\top \in \mathbb{R}^{m_l^p \times N_C}$.

Local unbalanced OT. Although both views now originate from the same local instance, their assignment distribution can still be long-tailed. We therefore retain marginal relaxation when matching each view to the local meta-prototypes. Let $\mathbf{s}_{m_l^p} = \mathbf{1}_{m_l^p}/m_l^p$ be the local sample reference marginal. For each $j \in \{1, 2\}$, we formulate local unbalanced OT problem as

$$\max_{\mathbf{Q} \in \mathbb{R}_+^{m_l^p \times N_C}} \Big\{ \langle \mathbf{Q}, \mathbf{S}_{l,j}^p \rangle_F + \varepsilon_l \mathcal{H}(\mathbf{Q}) - \rho_l \mathrm{KL}(\mathbf{Q}^\top \mathbf{1}_{m_l^p} \| \mathbf{r}) - \nu_l \mathrm{KL}(\mathbf{Q}\mathbf{1}_{N_C} \| \mathbf{s}_{m_l^p}) \Big\}, \tag{7}$$

where $\varepsilon_l, \rho_l, \nu_l > 0$ are the regularization coefficients. The solver produces $\mathbf{Q}_{l,j}^p$, whose b-th row is denoted by $\mathbf{q}_{l,b,j}^p$. Implementation details are provided in Appendix C.3.

Bidirectional local learning. Subsequently, following SwAV (Caron et al., 2020), the assignments are swapped for the self-supervision. Each participant p enables bidirectional local learning between its local model θ^p and the meta-prototypes in $\mathcal{C}_l^p$ by minimizing the local self-supervised loss $\mathcal{L}_{\mathrm{LSSL}}^p$, defined as follows:

$$\mathcal{L}_{\mathrm{LSSL}}^p = -\frac{1}{2m_l^p} \sum_{b=1}^{m_l^p} \sum_{k=1}^{N_C} \Big[q_{l,b,2,k}^p \log \pi_k(\mathbf{z}_{l,b,1}^p; \mathcal{C}_l^p) + q_{l,b,1,k}^p \log \pi_k(\mathbf{z}_{l,b,2}^p; \mathcal{C}_l^p) \Big], \tag{8}$$

where $\mathbf{z}_{l,b,j}^p$ denotes the b-th row of $\mathbf{Z}_{l,j}^p$ and $q_{l,b,j,k}^p = (\mathbf{Q}_{l,j}^p)_{b,k}$ denotes component k of $\mathbf{q}_{l,b,j}^p$. The function π_k follows Eq. (6), using $c_l^{p,k}$ as the prototype row. Both assignments are treated as fixed targets. This loss updates θ^p and its local bank $\mathcal{C}_l^p$ jointly, with prototype rows normalized before each forward pass. No server communication is needed between local mini-batches.

After local training, participant p records its final prototype bank as $\mathcal{C}_{t+1}^p = [c_{t+1}^{p,1}; \ldots; c_{t+1}^{p,N_C}]$ and uploads its prototype increment $\Delta_t^p = \mathcal{C}_{t+1}^p - \mathcal{C}_{t+1/2}^G$ to the server (communication ⑦ in Figure 3). The server averages the increments across participants and uses the update $\mathcal{C}_{t+1}^G = \mathcal{N}(\mathcal{C}_{t+1/2}^G + \frac{\eta_{\mathrm{meta}}}{P} \sum_{p=1}^P \Delta_t^p)$ for aggregation, where η_{meta} is learning rate. The resulting $\mathcal{C}_{t+1}^G$ is then used for the next pre-training epoch.

Notably, the meta-prototypes serve as a knowledge bridge between the two pre-training stages, naturally mitigating catastrophic forgetting that can arise from alternating between cross and local training.

Downstream task-specific fine-tuning phase. After pre-training, each participant p obtains a pretrained local encoder f^p. We then follow the standard supervised VFL protocol (Liu et al., 2024) to fine-tune the pretrained encoders and train a joint predictor using aligned and labeled samples (see Appendix A.1 for details). We assess HAMPL based on the downstream task performance of this joint VFL model.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Datasets and Models. We evaluate HAMPL on four public benchmarks: NUS-WIDE (Chua et al., 2009) for multimodal classification, Avazu[1] for click-through rate (CTR) prediction, Breast Histopathology Images (BHI)[2] for breast cancer identification, and ModelNet (Wu et al., 2015) for multi-view object classification. The first three datasets use two participants, while ModelNet uses four. We consider both homogeneous and heterogeneous encoder settings: participants use the same encoder architecture in the former, whereas the active participant uses a deeper encoder in the latter. Dataset statistics, feature partitions, and model architectures are detailed in Appendix A.2, with Tables 4 and 5.

[1] https://www.kaggle.com/c/avazu-ctr-prediction/data

[2] https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images

Baselines. We compare HAMPL with supervised baselines LightGBM (LGB) (Ke et al., 2017) and Vanilla VFL, the semi-supervised framework FedCVT (Kang et al., 2022), and the most advanced SSVFL frameworks FedLocal (He et al., 2025), FedHSSL (He et al., 2025), ISSVFL (Chen et al., 2025), and DACLSC (Chen et al., 2026). For FedHSSL and ISSVFL, we adopt their MoCo-based variants, FedHSSL-M and ISSVFL-M, for comparison due to their superior overall performance relative to the other variants. All methods use the same budget of aligned and labeled samples; baseline descriptions and their applicable settings are provided in Appendix A.2.

Evaluation Metrics. We assess the downstream joint VFL model using Top-1 accuracy on NUS-WIDE and ModelNet, the area under the receiver operating characteristic curve (AUC) on Avazu, and F1-score on BHI. Higher values indicate better performance for all metrics.

Data Augmentation. We use feature-level corruption for NUS-WIDE (Bahri et al., 2022), numerical-feature corruption and categorical-feature masking for Avazu following (Yao et al., 2021), and image augmentations following Chen & He (2021) for BHI and ModelNet. The dataset-specific augmentation strategies are detailed in Appendix A.2.

Implementation Details. Within each dataset and model setting, HAMPL and all baselines use the same encoder architectures. We designate 40% of the training samples as aligned, with the remaining samples treated as unaligned. For downstream task-specific fine-tuning, we test learning rates of 0.025 and 0.01 and report the best-performing results. Pre-training lasts 40 epochs with a batch size of 512 on all datasets. Additional hyperparameters, prototype counts, queue lengths, and prototype-freezing schedules are specified in Appendix A.2.

3.2 Experimental Results under Homogeneous Settings

For a fair comparison, we leave the partial model aggregation step of the existing SSVFL baselines intact. Because this step requires compatible model components across participants, we conduct the primary comparison under the homogeneous encoder setting.

As shown in Table 1, label scarcity substantially limits the performance of the supervised methods LGB and Vanilla VFL, as well as the semi-supervised method FedCVT. HAMPL's gains over the local-only baseline FedLocal highlight the benefits of cross-participant collaboration. HAMPL achieves the best results in most settings. With 200 labeled samples, its relative improvements over the strongest SSVFL baseline on each dataset are 1.7% on NUS-WIDE, 1.5% on Avazu, 1.6% on BHI, and 4.9% on ModelNet. Its advantage on the four-participant ModelNet benchmark further demonstrates the effectiveness of prototype-based collaboration with multiple participants. For a more comprehensive evaluation, we investigate the impact of the aligned-sample ratio on HAMPL in Appendix D.1.

In contrast to existing SSVFL methods that directly exchange intermediate representations and partial model parameters, HAMPL communicates assignment matrices and prototype updates during pre-training. This avoids directly disclosing local encoder representations and parameters while maintaining strong downstream performance. We also provide a comprehensive security analysis in Appendices B.1 and discuss the privacy boundary of HAMPL in D.2. We further evaluate HAMPL's scalability to heterogeneous encoders in Section 3.3. In addition, we visualize the representations learned by each method after pre-training to provide an intuitive assessment of HAMPL's ability to mitigate severe representation collapse, as detailed in Appendix D.4 with Figure 5.

3.3 Scalability for Heterogeneous Settings

Table 2 evaluates HAMPL under heterogeneous encoders across the five label budgets. HAMPL retains performance close to that obtained with homogeneous encoders. Averaged across all datasets and label budgets, the relative performance decrease is only 0.8%. The results indicate that HAMPL remains effective when participants use different encoder architectures. The modest degradation is expected because heterogeneous encoders induce representation spaces with inherently different geometric structures, making cross-participant consensus construction and prototype matching more challenging. Moreover, after replacing the active participant's encoder with a deeper architecture, we retain its learning rate, number of training epochs, and other optimization hyperparameters from the homogeneous setting without architecture-specific tuning. Despite these additional difficulties,

Table 1: Performance evaluation of various methods.

Number of labeled samples		200	400	600	800	1000
Dataset	Method			Performance		
NUS-WIDE (Top-1 Acc)	LGB	0.415±0.020	0.460±0.017	0.525±0.021	0.559±0.009	0.589±0.014
	Vanilla VFL	0.501±0.017	0.544±0.011	0.561±0.013	0.590±0.012	0.614±0.012
	FedCVT	0.534±0.013	0.566±0.014	0.601±0.015	0.625±0.007	0.633±0.007
	FedLocal	0.597±0.012	0.620±0.008	0.632±0.008	0.639±0.011	0.655±0.010
	FedHSSL-M	0.645±0.006	0.670±0.004	0.679±0.003	0.685±0.003	0.693±0.004
	ISSVFL-M	0.653±0.009	0.677±0.003	0.683±0.008	0.691±0.006	0.696±0.008
	DACLSC	0.661±0.007	0.680±0.010	0.690±0.004	0.700±0.007	0.705±0.005
	HAMPL	**0.672**±0.009	**0.682**±0.014	**0.699**±0.004	**0.704**±0.005	**0.714**±0.007
Avazu (AUC)	LGB	0.577±0.012	0.573±0.020	0.592±0.013	0.613±0.010	0.615±0.016
	Vanilla VFL	0.592±0.020	0.591±0.009	0.602±0.005	0.606±0.009	0.620±0.006
	FedCVT	0.598±0.023	0.611±0.019	0.614±0.010	0.630±0.017	0.632±0.012
	FedLocal	0.573±0.015	0.600±0.008	0.610±0.005	0.611±0.014	0.616±0.008
	FedHSSL-M	0.635±0.006	0.644±0.006	0.642±0.011	0.646±0.011	0.658±0.006
	ISSVFL-M	0.640±0.009	0.643±0.004	0.652±0.007	0.659±0.007	0.670±0.005
	DACLSC	0.648±0.017	0.657±0.009	0.671±0.006	0.670±0.005	0.677±0.005
	HAMPL	**0.658**±0.008	**0.662**±0.002	0.665±0.013	**0.675**±0.007	**0.683**±0.012
BHI (F1-Score)	Vanilla VFL	0.704±0.009	0.713±0.005	0.729±0.006	0.731±0.010	0.738±0.011
	FedCVT	0.717±0.013	0.720±0.012	0.728±0.007	0.741±0.010	0.753±0.011
	FedLocal	0.729±0.006	0.733±0.005	0.742±0.010	0.758±0.005	0.771±0.009
	FedHSSL-M	0.799±0.002	0.798±0.002	0.801±0.002	0.807±0.003	0.810±0.002
	ISSVFL-M	0.797±0.003	0.798±0.003	0.795±0.001	0.809±0.005	0.811±0.001
	DACLSC	0.808±0.002	0.805±0.006	0.812±0.004	0.817±0.001	0.822±0.004
	HAMPL	**0.821**±0.004	**0.822**±0.001	**0.831**±0.001	**0.838**±0.001	**0.839**±0.001
ModelNet (Top-1 Acc)	Vanilla VFL	0.601±0.022	0.642±0.014	0.665±0.013	0.706±0.008	0.720±0.010
	FedLocal	0.643±0.018	0.681±0.009	0.724±0.008	0.740±0.007	0.755±0.005
	FedHSSL-M	0.690±0.008	0.734±0.004	0.765±0.006	0.779±0.009	0.780±0.007
	ISSVFL-M	0.696±0.005	0.731±0.005	0.783±0.002	0.798±0.004	0.803±0.002
	DACLSC	0.701±0.004	0.738±0.005	0.781±0.004	0.790±0.002	0.805±0.003
	HAMPL	**0.735**±0.010	**0.753**±0.008	**0.801**±0.002	**0.811**±0.001	**0.818**±0.002

Note: For each dataset, the best-performing value in each column is marked in bold.

Table 2: Performance evaluation under heterogeneous settings.

Dataset	Number of labeled samples				
	200	400	600	800	1000
NUS-WIDE	0.664±0.004	0.685±0.004	0.693±0.003	0.703±0.006	0.708±0.006
Avazu	0.652±0.006	0.656±0.008	0.658±0.011	0.665±0.004	0.675±0.011
BHI	0.818±0.003	0.821±0.002	0.829±0.002	0.830±0.002	0.835±0.001
ModelNet	0.727±0.002	0.729±0.003	0.797±0.002	0.811±0.002	0.816±0.001

HAMPL maintains strong performance across all four datasets, demonstrating its scalability to heterogeneous VFL settings.

3.4 Ablation Study

We construct six HAMPL variants to evaluate the contribution of its principal components. **HAMPL w/o CU** and **HAMPL w/o LU** replace cross and local unbalanced OT, respectively, with balanced OT. **HAMPL w/o UOT** replaces both with balanced OT. **HAMPL w/o EF** replaces entropy-guided consensus fusion with uniform averaging. **HAMPL w/o CG** removes consensus-guided OT and entropy-guided consensus fusion, and instead learns directly from the uniformly averaged consensus assignments. **HAMPL w/o PA** removes meta-prototype aggregation after local training. We conduct the ablation study under the heterogeneous setting. For each dataset, 200 aligned and labeled samples are used, and 40% of the training samples are treated as aligned. Table 3 reports the performance comparison across the variants.

Table 3: Performance comparison across the variants.

Method	NUS-WIDE	Avazu	BHI	ModelNet
	Top-1 Acc	AUC	F1-Score	Top-1 Acc
FedLocal	0.588±0.014	0.568±0.010	0.721±0.006	0.631±0.012
HAMPL w/o EF	0.658±0.003	0.649±0.004	0.815±0.002	0.722±0.006
HAMPL w/o CU	0.651±0.006	0.643±0.010	0.809±0.004	0.718±0.008
HAMPL w/o LU	0.655±0.004	0.650±0.007	0.812±0.002	0.719±0.003
HAMPL w/o UOT	0.643±0.006	0.639±0.005	0.806±0.003	0.710±0.011
HAMPL w/o PA	0.655±0.003	0.648±0.009	0.814±0.002	0.722±0.007
HAMPL w/o CG	0.638±0.008	0.633±0.013	0.802±0.003	0.707±0.010
HAMPL	**0.664**±0.004	**0.652**±0.006	**0.818**±0.003	**0.727**±0.002

Note: The best-performing value in each column is marked in bold.

It is shown that HAMPL obtains the highest performance on all four datasets. Removing either cross or local unbalanced OT reduces performance, with larger decreases for HAMPL w/o CU than for HAMPL w/o LU on every dataset. These results support retaining marginal relaxation in both stages to accommodate nonuniform matching preferences. The largest degradation among the six variants occurs for HAMPL w/o CG. To distinguish the role of consensus-guided rematching from that of entropy-guided consensus fusio, we compare it with HAMPL w/o EF: both use uniform fusion, but the latter retains participant-specific rematching and achieves higher performance. This supports using consensus as guidance for rematching rather than directly imposing a shared assignment target. In addtion, removing prototype aggregation also lowers all four scores, supporting the role of meta-prototypes in carrying locally learned information into subsequent collaborative training.

3.5 Security, Privacy, and Efficiency Analysis

We analyze HAMPL's privacy threats, including feature reconstruction, attribute inference, and label inference, and security threats posed by backdoor attacks. First, we identify a potential server-side attack chain in representation-sharing SSVFL, as detailed in Appendix B.1. In contrast, during pre-training, HAMPL only needs to exchange assignments and prototype information. We then discuss how participant-specific learning in HAMPL reduces encoder coupling and can limit an attacker's ability to infer the active participant's labels and sensitive attributes (Appendix B.1). We further evaluate this hypothesis through label-inference experiments in Appendix D.2. We further characterize privacy boundaries for information recoverable from assignments and prototype updates (Appendix D.3). For cooperative VFL, we focus on backdoors rather than deliberate convergence disruption. HAMPL supplies no task-label gradients and performs no aggregation of participant models during pre-training, thereby limiting opportunities for backdoor injection through these channels. During downstream supervised fine-tuning, it can seamlessly integrate existing VFL defenses (Zou et al., 2024; Liu et al., 2021), as detailed in Appendix B.1. We further compare the computational complexity of HAMPL and DACLSC in Appendix B.1.

4 Conclusion

This paper presents HAMPL, a self-supervised framework that rethinks collaboration in label-scarce VFL through learnable meta-prototypes. Rather than rigidly aligning representations from complementary feature subsets, HAMPL enables effective knowledge sharing while preserving distinct participant-specific semantics, thereby accommodating cross-participant semantic discrepancies. The meta-prototypes also bridge the two training stages, integrating collaborative and locally acquired knowledge. Extensive experiments demonstrate its superior downstream performance and continued effectiveness under heterogeneous encoder settings, while ablations support the contributions of the principal components. Our security and privacy analyses examine reconstruction, inference, and backdoor threats and characterize HAMPL's privacy boundaries. The privacy experiments further suggest that HAMPL encoder acquires less additional label information from collaboration than their counterparts in representation-sharing SSVFL frameworks.

AI USE STATEMENT

In preparing this manuscript, generative AI tools assisted with language, notation, formatting, and the drafting of the security and privacy discussion. These tools were not used to generate experimental measurements. The authors are responsible for verifying all AI-assisted content and for the final manuscript.

REPRODUCIBILITY STATEMENT

Our source code is available at `https://anonymous.4open.science/r/HAMPL-3E16`. The complete source code for HAMPL, together with the scripts required to reproduce the experiments, is provided in the accompanying source-code package. The package is structured as follows:

- `/HAMPL-HOMO/`: Contains the implementation for the homogeneous setting. `main_pretrain.py` and `main_cls.py` are the main entry points for pretraining and downstream evaluation, respectively; `prepare_experiments.py` configures the datasets and models; and `run_experiments.bat` provides the experiment commands.
- `/HAMPL-HETER/`: Contains the corresponding implementation and experiment scripts for the heterogeneous setting.
- `/LIA/`: Contains the label-inference evaluation code, including the model-completion entry points (`model_completion.py` and `model_completion_mixtext.py`), the vertical federated learning frameworks, and `run_model_completion.bat`.
- `/models/` and `/data/` (or `/datasets/` in `/LIA/`): Provide the model architectures, dataset loaders, and preprocessing utilities used by the experiment entry points.

Pretraining should be run before downstream classification and label-inference evaluation. Dataset locations should be configured in `prepare_experiments.py` or supplied through the applicable command-line arguments. Pretrained weights and generated experiment outputs are not included; they are produced locally by the corresponding training commands.

REFERENCES

Mido Assran, Randall Balestriero, Quentin Duval, Florian Bordes, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, and Nicolas Ballas. The hidden uniform cluster prior in self-supervised learning. In *Proceedings of the 11th International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=04K3PMtMckp.

Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CuV_qYkmKb3.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9912–9924, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/70feb62b69f16e0238f741fab228fec2-Paper.pdf.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021. doi: 10.1109/ICCV48922.2021.00951. URL https://doi.org/10.1109/ICCV48922.2021.00951.

Timothy Castiglia, Shiqiang Wang, and Stacy Patterson. Self-supervised vertical federated learning. In *NeurIPS Workshop on Federated Learning: Recent Advances and New Challenges*, 2022. URL https://openreview.net/forum?id=z2RNsvYZZTf.

Shuai Chen, Wenyu Zhang, Xiaoling Huang, Cheng Zhang, and Qingjun Mao. Improving self-supervised vertical federated learning with contrastive instance-wise similarity and dynamical

balance pool. *Future Generation Computer Systems*, 172:107884, 2025. doi: 10.1016/j.future.2025.107884. URL https://doi.org/10.1016/j.future.2025.107884.

Shuai Chen, Wenyu Zhang, Jiahao Nie, Zixuan Dai, and Qingjun Mao. Dual-aware collaboration and localized semantic calibration for label-scarce vertical federated learning. *Information Sciences*, 757:123909, 2026. ISSN 0020-0255. doi: 10.1016/j.ins.2026.123909. URL https://doi.org/10.1016/j.ins.2026.123909.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 1597–1607, 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the 2021 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*, pp. 15750–15758, 2021. doi: 10.1109/CVPR46437.2021.01549. URL https://doi.org/10.1109/CVPR46437.2021.01549.

Zihao Chen, Chi-Heng Lin, Ran Liu, Jingyun Xiao, and Eva L. Dyer. Your contrastive learning problem is secretly a distribution alignment problem. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 91597–91617, 2024. doi: 10.52202/079017-2907. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/a6a21421022d1da16eadbba533980530-Paper-Conference.pdf.

Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. Scaling algorithms for unbalanced optimal transport problems. *Mathematics of computation*, 87:2563–2609, 2018.

Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, New York, NY, USA, 2009. doi: 10.1145/1646396.1646452. URL https://doi.org/10.1145/1646396.1646452.

Gabriele D'Acunto, Enrico Grimaldi, Valeria Avino, Mario Edoardo Pandolfo, Leonardo Di Nino, Sergio Barbarossa, and Paolo Di Lorenzo. Sheaf-based federated representation learning. *arXiv preprint arXiv:2608.10016*, 2026.

Siwei Feng. Vertical federated learning-based feature selection with non-overlapping sample utilization. *Expert Systems with Applications*, 208:118097, 2022. doi: 10.1016/j.eswa.2022.118097. URL https://doi.org/10.1016/j.eswa.2022.118097.

Chong Fu, Xuhong Zhang, Shouling Ji, Jinyin Chen, Jingzheng Wu, Shanqing Guo, Jun Zhou, Alex X. Liu, and Ting Wang. Label inference attacks against vertical federated learning. In *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22)*, pp. 1397–1414, 2022. URL https://www.usenix.org/conference/usenixsecurity22/presentation/fu-chong.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/f3ada80d5c4ee70142b17b8192b2958e-Abstract.html.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 9726–9735, 2020. doi: 10.1109/CVPR42600.2020.00975. URL https://doi.org/10.1109/CVPR42600.2020.00975.

Yuanqin He, Yan Kang, Xinyuan Zhao, Jiahuan Luo, Lixin Fan, Yuxing Han, and Qiang Yang. A hybrid self-supervised learning framework for vertical federated learning. *IEEE Transactions on Big Data*, 11(5):2210–2223, 2025. doi: 10.1109/TBDATA.2024.3403386. URL https://doi.org/10.1109/TBDATA.2024.3403386.

Xiao Jin, Pin-Yu Chen, Chia-Yi Hsu, Chia-Mu Yu, and Tianyi Chen. CAFE: Catastrophic data leakage in vertical federated learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 994–1006, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/08040837089cdf46631a10aca5258e16-Abstract.html.

Yan Kang, Yang Liu, and Xinle Liang. FedCVT: Semi-supervised vertical federated learning with cross-view training. *ACM Transactions on Intelligent Systems and Technology*, 13(4):64:1–64:16, 2022. doi: 10.1145/3510031. URL https://doi.org/10.1145/3510031.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 3149–3157, 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cOtBRgsf2fO.

Xinting Liao, Weiming Liu, Chaochao Chen, Pengyang Zhou, Fengyuan Yu, Huabin Zhu, Binhui Yao, Tao Wang, Xiaolin Zheng, and Yanchao Tan. Rethinking the representation in federated unsupervised learning with non-iid data. In *Proceedings of 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2024*, pp. 22841–22850, 2024. doi: 10.1109/CVPR52733.2024.02155. URL https://doi.org/10.1109/CVPR52733.2024.02155.

Jing Liu, Chulin Xie, Krishnaram Kenthapadi, Oluwasanmi Koyejo, and Bo Li. RVFR: Robust vertical federated learning via feature subspace recovery. In *NeurIPS Workshop on New Frontiers in Federated Learning*, 2021. URL https://neurips2021workshopfl.github.io/NFFL-2021/papers/2021/Liu2021.pdf.

Yang Liu, Zhihao Yi, and Tianjian Chen. Backdoor attacks and defenses in feature-partitioned collaborative learning. In *ICML Workshop on Federated Learning for User Privacy and Data Confidentiality*, 2020. URL https://arxiv.org/abs/2007.03608.

Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. Vertical federated learning: Concepts, advances, and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3615–3634, 2024. doi: 10.1109/TKDE.2024.3352628.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2NlGyqNjns.

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 *Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5180–5189. IEEE, 2023. doi: 10.1109/ICCV51070.2023.00480. URL https://doi.org/10.1109/ICCV51070.2023.00480.

Yonglong Tian, Lijie Fan, Kaifeng Chen, Dina Katabi, Dilip Krishnan, and Phillip Isola. Learning vision from models rivals learning vision from data. In *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024*, pp. 15887–15898, June 2024. doi: 10.1109/CVPR52733.2024.01504. URL https://doi.org/10.1109/CVPR52733.2024.01504.

Liwen Wu, Zhizhi Liu, Bin Pu, Kang Wei, Hangcheng Cao, and Shaowen Yao. DGGI: Deep generative gradient inversion with diffusion model. *Information Fusion*, 113:102620, 2025. doi: 10.1016/j.inffus.2024.102620. URL https://doi.org/10.1016/j.inffus.2024.102620.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 1912–1920, 2015. URL https://openaccess.thecvf.com/content_cvpr_2015/html/Wu_3D_ShapeNets_A_2015_CVPR_paper.html.

Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H. Chi, Steve Tjoa, Jieqi Kang, and Evan Ettinger. Self-supervised learning for large-scale item recommendations. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, pp. 4321–4330, 2021. doi: 10.1145/3459637.3481952. URL https://doi.org/10.1145/3459637.3481952.

Mang Ye, Wei Shen, Bo Du, Eduard Snezhko, Vassili Kovalev, and Pong C Yuen. Vertical federated learning for effectiveness, security, applicability: A survey. *ACM Computing Surveys*, 57(9):1–32, April 2025. ISSN 0360-0300. doi: 10.1145/3720539. URL https://doi.org/10.1145/3720539.

Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *Proceedings of the 2022 European Conference on Computer Vision*, pp. 668–684, 2022. doi: 10.1007/978-3-031-19809-0_38. URL https://doi.org/10.1007/978-3-031-19809-0_38.

Haoran Zhang, Dongjun Kim, Seohyeon Cha, and Haris Vikalo. Fedrot-loRA: Mitigating rotational misalignment in federated loRA. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=2X8Qi3VdjA.

Lei Zhang, Lele Fu, Chen Liu, Zhao Yang, Jinghua Yang, Zibin Zheng, and Chuan Chen. Toward few-label vertical federated learning. *ACM Transactions on Knowledge Discovery from Data*, 18(7):176:1–176:21, 2024. doi: 10.1145/3656344. URL https://doi.org/10.1145/3656344.

Tianyuan Zou, Yang Liu, Yan Kang, Wenhan Liu, Yuanqin He, Zhihao Yi, Qiang Yang, and Ya-Qin Zhang. Defending batch-level label inference and replacement attacks in vertical federated learning. *IEEE Transactions on Big Data*, 10(6):1016–1027, 2024. doi: 10.1109/TBDATA.2022.3192121. URL https://doi.org/10.1109/TBDATA.2022.3192121.

A Appendix

A.1 Downstream Task-Specific Fine-Tuning Phase

After pre-training, each participant p obtains a pretrained local encoder f^p. We then follow the standard supervised VFL protocol (Liu et al., 2024) using the aligned and labeled samples indexed by $\mathcal{I}_{\text{al}}$. The active participant combines the local representations derived from local encoders of passive participants and feeds them into a task-specific joint predictor. The pretrained encoders and the joint predictor are jointly optimized by minimizing the cross-entropy loss using labels $\mathcal{Y}_{\text{al}}$, adapting the resulting joint VFL model to the downstream task. We assess HAMPL based on the downstream task performance of this joint VFL model.

A.2 Detailed Experimental Setup

Datasets. We follow the benchmark construction and preprocessing protocol used in FedHSSL (He et al., 2025), ISSVFL (Chen et al., 2025), and DACLSC (Chen et al., 2026). Table 4 summarizes the datasets used in this work.

- **NUS-WIDE** (Chua et al., 2009) contains Flickr images with associated text tags. We select ten semantic classes for multi-class classification. The two participants hold the 634-dimensional low-level image features and the corresponding 1,000-dimensional tag features, respectively, giving complementary descriptions of the same entities.
- **Avazu** is a binary CTR prediction benchmark containing numerical and categorical features. These features are partitioned between two participants, and each categorical feature is represented by a 32-dimensional embedding. We use a subset containing 800,000 training samples and 200,000 test samples.
- **BHI** is a binary breast cancer classification benchmark. We construct a two-participant VFL task by assigning two different histopathology images from the same patient to different participants, so that their observations are related but not identical.
- **ModelNet** (Wu et al., 2015) contains 3D objects from 40 categories. Following the benchmark protocol, we select 15 classes with a long-tailed class distribution. Each object is rendered from 12 viewpoints using Blender, and the resulting images are divided equally among four participants. A VFL sample is constructed by selecting one image of the same object from each participant's available viewpoints.

Table 4: Dataset details

Dataset	Classes	Training samples	Test samples	Metric	Data type
NUS-WIDE	10	58,619	38,955	Top-1 accuracy	Tabular
Avazu	2	800,000	200,000	AUC	Tabular
BHI	2	110,673	27,944	F1-score	Image
ModelNet	15	15,020	3,300	Top-1 accuracy	Image

Models and architectural settings. Table 5 specifies the encoder f^p used by each participant. In the homogeneous setting, all participants use the same architecture for a given dataset, although their input dimensions can differ with their feature subsets. These architectures follow the encoder configurations in Chen et al. (2026). In the heterogeneous setting, only the active participant's encoder is replaced: MLP2 becomes MLP3 on NUS-WIDE, the two Fully Connected (FC) layers becomes three FC layers on Avazu, and ResNet-18 becomes ResNet-34 on both BHI and ModelNet. Passive participants retain their homogeneous-setting architectures. Here MLP2 and MLP3 denote networks with two and three FC layers, respectively.

Table 5: Encoder architectures in the two settings. Passive participants retain the homogeneous encoder in the heterogeneous setting.

Dataset	Homogeneous: all participants	Heterogeneous: active participant
NUS-WIDE	MLP2 (2 FC layers)	MLP3 (3 FC layers)
Avazu	Embedding layer + 2 FC layers	Embedding layer + 3 FC layers
BHI	ResNet-18	ResNet-34
ModelNet	ResNet-18	ResNet-34

For every dataset, the active participant's downstream classifier is a single FC layer. This classifier is distinct from the SSL heads used during pre-training. The projector φ^p contains two FC layers, mapping a 512-dimensional encoder output through a 512-dimensional hidden layer to the shared space of dimension $R = 128$. The same encoder configurations are used for HAMPL and all encoder-based baselines within each experimental setting; LightGBM is a tree-based exception.

Baselines. The comparison includes supervised, semi-supervised, and self-supervised approaches:

- **LightGBM (LGB)** (Ke et al., 2017) is a gradient-boosted decision-tree baseline trained only on aligned labeled samples. It is included for the tabular datasets, NUS-WIDE and Avazu.
- **Vanilla VFL** (Liu et al., 2024) trains the local encoders and the active participant's classifier using aligned labeled samples and the supervised task loss.
- **FedCVT** (Kang et al., 2022) uses cross-view learning to estimate missing feature representations and generate pseudo-labels, allowing unlabeled and non-overlapping samples to contribute to training. Its original formulation is evaluated in the two-participant settings, rather than the four-participant ModelNet setting.
- **FedLocal** (He et al., 2025) independently pre-trains each participant's encoder on its private samples using MoCo (He et al., 2020), followed by supervised VFL fine-tuning. It does not use aligned samples for collaborative representation learning during pre-training.
- **FedHSSL** (He et al., 2025) combines cross-participant SSL, local SSL, and partial model aggregation to transfer knowledge between participants. We include its BYOL, SimSiam, and MoCo variants, denoted by FedHSSL-B, FedHSSL-S, and FedHSSL-M, respectively.
- **ISSVFL** (Chen et al., 2025) incorporates contrastive instance-wise similarity learning and a dynamical balance pool into self-supervised VFL. Its BYOL, SimSiam, and MoCo variants are denoted by ISSVFL-B, ISSVFL-S, and ISSVFL-M, respectively.
- **DACLSC** (Chen et al., 2026) combines dual-aware instance- and dimension-wise collaboration with multi-source adaptive semantic fusion and localized semantic calibration, improving collaborative representation learning and subsequent local knowledge transfer.

BYOL, SimSiam, and MoCo follow Grill et al. (2020), Chen & He (2021), and He et al. (2020), respectively. Baseline-specific hyperparameters follow their original references unless superseded by the shared experimental protocol. All methods receive the same aligned labeled samples for a given comparison.

Evaluation metrics. Top-1 accuracy measures the proportion of correctly classified samples on NUS-WIDE and ModelNet. AUC measures the ranking quality of predicted click probabilities on Avazu across classification thresholds. On BHI, F1-score is the harmonic mean of precision and recall for the positive cancer class. All metrics are computed from the downstream joint VFL model after supervised fine-tuning, rather than from the pre-training assignments.

Data augmentation. For NUS-WIDE, we use random feature corruption inspired by Bahri et al. (2022): 30% of each participant's feature values are selected and replaced by random values. The implementation samples each replacement uniformly within the corresponding feature's observed range. The original feature vector and its corrupted version form the two local views. For Avazu, we similarly corrupt 30% of the numerical features, sampling replacements in $[0, 1]$ after numerical-feature normalization. For categorical features, we select 30% of the fields and replace their values with reserved masking categories, providing the embedding-based feature corruption used in the reference protocol (Yao et al., 2021; Chen et al., 2026).

For BHI and ModelNet, we independently apply the image augmentation pipeline twice to obtain the two local views, following the augmentation family in Chen & He (2021). The implementation uses random resized cropping to 32×32 pixels with crop scale in $[0.3, 1.0]$, color jitter with probability 0.8, Gaussian blur with probability 0.5 and blur radius sampled from $[0.1, 2.0]$, and horizontal flipping with probability 0.5. The color-jitter strengths are 0.3 for brightness, contrast, and saturation and 0.1 for hue. The resulting images are converted to tensors. These augmentations are applied locally and require no exchange of raw samples.

OT hyperparameters. The prediction temperature is fixed at $\tau = 0.1$ on all datasets. Table 6 gives the hyperparameters of the three OT problems using exactly the notation of the main text and Appendix C. Cross UOT and local UOT use the same prototype- and sample-marginal penalties, but different entropy coefficients. All three solvers perform five iterations in the log domain, and the numerical floor in consensus-guided OT is $\delta = 10^{-12}$. The consensus coefficient λ is dataset- and setting-dependent, as shown in Table 7; all other OT parameters are shared across datasets and settings.

Meta-prototypes, queues, and warm-up. Table 7 lists the number of meta-prototypes N_C, the queue lengths, and the prototype-freezing thresholds. Queue length denotes the number of cached

Table 6: Hyperparameters of the three OT problems. A dash indicates that the corresponding penalty is not part of that problem.

OT problem	Entropy	Prototype KL	Sample KL	Consensus KL	Iterations
Cross UOT	$\varepsilon_{\mathrm{a}} = 0.01$	$\rho_{\mathrm{a}} = 0.5$	$\nu_{\mathrm{a}} = 1.0$	—	$N_{\mathrm{iter,a}} = 5$
Consensus-guided OT	$\varepsilon_{\mathrm{g}} = 0.01$	—	—	λ	$N_{\mathrm{iter,g}} = 5$
Local UOT	$\varepsilon_{l} = 0.02$	$\rho_{l} = 0.5$	$\nu_{l} = 1.0$	—	$N_{\mathrm{iter},l} = 5$

representations per participant; in the local stage, each of the two views has its own queue of the listed length. The cross and local queues augment the sample sets used to estimate first-pass cross-UOT and local-UOT assignments, respectively. Only current-mini-batch assignments enter the training losses, and consensus-guided OT operates on the current aligned mini-batch without appending queued samples.

Table 7: Dataset-specific HAMPL settings. Homo./Hetero. denote homogeneous/heterogeneous encoders. Queue and freeze settings are shared by both settings.

	λ			Queue length		prototype-freezing period
Dataset	Homo.	Hetero.	N_C	Cross	Local	(mini-bacth)
NUS-WIDE	1.0	1.5	1,024	3,072	3,072	1,536
Avazu	1.0	1.0	256	0	0	1,536
BHI	1.0	1.0	256	0	0	1,536
ModelNet	1.0	1.5	1,024	1,536	1,536	512

To prevent unstable representations at initialization from disrupting prototype learning, we freeze prototype updates for the first 1,536 training mini-batches on NUS-WIDE, Avazu, and BHI. Since ModelNet contains fewer training samples, its prototype-freezing period is shortened to 512 mini-batches. The warm-up counter accumulates training batches across both the cross and local stages, rather than restarting at a stage or epoch boundary; local encoders and projectors remain trainable during this period. For the alignment-ratio study in Appendix D.1, we further adjust the prototype-freezing period as the aligned-sample ratio decreases.

A.3 Related Work

Self-supervised learning. Instance-discriminative representation learning methods has been a major driver of SSL encouraging agreement between positive views while separating negative pairs (Chen et al., 2020). MoCo (He et al., 2020) uses a queue and a momentum encoder to maintain a large, consistent dictionary for contrastive learning. Subsequent refinements include DCL (Yeh et al., 2022), which improves optimization by removing the positive term from the loss denominator, and SynCLR (Tian et al., 2024), which treats synthetic images generated from the same caption as positive pairs. In parallel, BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2021) demonstrate that effective representations can be learned from positive pairs without explicit negatives. SwAV (Caron et al., 2020) promotes cross-view consistency through swapped prediction of online cluster assignments. Building on these advances, DINO (Caron et al., 2021) performs self-distillation by matching the output distributions of a momentum teacher and a student across augmented views. Oquab et al. (2024) advances this framework toward general-purpose visual pretraining by combining image-level self-distillation with masked patch prediction and feature regularization, while scaling training on a large curated dataset. Siméoni et al. (2026) further scales both data and model size and introduces Gram anchoring to preserve pairwise relationships among patch features, thereby preventing dense representations from degrading during prolonged training.

In addition, Assran et al. (2023) theoretically characterize the uniform cluster prior underlying SwAV and empirically show that this prior can impair downstream generalization on long-tailed data with pronounced semantic class imbalance. Chen et al. (2024) incorporate unbalanced optimal transport (UOT) into contrastive SSL by replacing hard constraints on both transport marginals with divergence penalties, improving robustness to noisy data and unreliable augmented views. UOT-

based regularization has also been incorporated into local SSL in HFL, matching local representations to non-learnable random reference samples on the unit sphere to mitigate representation collapse (Liao et al., 2024). However, both UOT-based approaches build on artificially augmented views of the same instance and do not directly address the cross-participant semantic discrepancies arising from complementary feature subsets in VFL.

Self (semi)-supervised vertical federated learning. Early semi-supervised VFL methods sought to estimate missing features and labels from limited aligned labeled samples. Kang et al. (2022) employed cross-view training to infer missing feature representations and pseudo-labels, whereas Sun et al. (2023) clustered gradients from different participants to derive pseudo-labels that support local training. Zhang et al. (2024) further combined supervised contrastive learning with pseudo-label-guided consistency to exploit unlabeled samples and mitigate cross-participant semantic discrepancies under domain shift. Nevertheless, these approaches remain dependent on true or inferred supervision, making them sensitive to label scarcity and error accumulation. Castiglia et al. (2022) and Feng (2022) introduced self-supervised objectives that enable each participant to train on its private unlabeled samples, but make limited use of aligned samples for cross-participant collaboration. He et al. (2025) subsequently proposed a knowledge-transfer framework that combines cross-party SSL on aligned samples with local SSL on private unaligned samples, thereby exploiting all available data. Chen et al. (2025) further improved cross-party self-supervision by introducing contrastive instance-wise similarity learning and a dynamical balance pool. More recently, Chen et al. (2026) enhanced both semantic fusion and knowledge transfer methods to further mitigate semantic dislocation. Despite this progress, existing frameworks remain largely rooted in the instance-discriminative representation learning paradigm and depend on frequent inter-participant representation exchange or costly knowledge-transfer schemes, limiting scalability in heterogeneous settings while introducing substantial computational overhead and security risks.

B Additional Analysis

B.1 Security, Privacy, and Efficiency Analysis

We distinguish privacy threats, including feature reconstruction, attribute inference, and label inference, from security threats that manipulate the behavior of the joint VFL model (Liu et al., 2024). For privacy analysis, we consider a non-colluding, honest-but-curious server or passive participant that follows the protocol but analyzes its observations. For security analysis, we focus on malicious participants seeking to implant backdoors through their local data or uploaded messages, with an honest coordinating server.

We first identify, from the server's perspective, a potential attack pathway in existing representation-sharing SSVFL frameworks.

Potential privacy threats on representation-sharing SSVFL. Representative frameworks such as FedHSSL and DACLSC exchange intermediate representations and aggregate partial models (He et al., 2025; Chen et al., 2026). A server therefore observes individual uploads before aggregation. Given these representations and the uploaded model components, a potential attack is

$$\mathbf{z}^p \xrightarrow{G} \widehat{\mathbf{x}}^p \xrightarrow{E} \widehat{\mathbf{z}}^p, \qquad \mathcal{L}_{\mathrm{cyc}} = \mathbb{E}\big\|E(G(\mathbf{z}^p)) - \mathbf{z}^p\big\|_2^2. \tag{B.1}$$

Here the generator G takes participant p's uploaded representation $\mathbf{z}^p$ as input and produces a surrogate raw input $\widehat{\mathbf{x}}^p$. The network E is a surrogate discriminator, with $E(\cdot)$ denoting its representation output. Its corresponding modules reproduce the architectures of the uploaded model components and are initialized with their exact parameter values; the undisclosed modules must be constructed by the attacker. Since these frameworks use instance-discriminative representation learning to enforce alignment across participants, the attacker could exploit the same principle through an instance-discriminative SSL objective that encourages the reconstructed representations to match the observed representation distribution. In addition to participant p's own representation, positive targets could include another participant's representation of the same entity or a fused representation of that entity.

Without the original feature schema or sufficient priors, $\widehat{\mathbf{x}}^p$ need not correspond to genuine features, and cycle consistency alone can admit uninformative solutions. Nevertheless, reconstructed representations may support attribute and label predictors, particularly when shadow data or a few

annotated examples provide semantic anchors. Given input-compatible shadow samples with task labels or sensitive-attribute annotations, the attacker could discard G, attach an appropriate prediction head to E, and fine-tune the retained network for label or attribute inference. Applying this network requires inputs compatible with E; uploaded representations alone are not automatically valid inputs to its raw-input interface. The reconstruction step itself reorganizes already exposed information rather than creating additional information. Since investigating this attack lies beyond the scope of this work, we present this chain as an attack hypothesis.

Privacy implications of HAMPL. During HAMPL pre-training, the communicated quantities are assignments $\mathbf{Q}_{\mathrm{a}}^{p}$ and $\mathbf{Q}_{\mathrm{G}}^{p}$, prototype gradients $\mathbf{g}_{C}^{p}$, prototype increments Δ_{t}^{p}, and prototype banks. Local encoder/projector parameters and their gradients are not uploaded, and sample representations are not directly transmitted. This distinction applies to pre-training: downstream standard supervised VFL exchanges representations and their loss gradients, as described in Appendix A.1. HAMPL avoids the direct representation-sharing and partial-model-aggregation channels above. Its heterogeneous encoders need not disclose their architectures or input dimensions, complicating attacks that require a known map from latent vectors to raw features. It also avoids rigid instance-level alignment across participants.

For feature reconstruction, gradient inversion attacks (Wu et al., 2025) and VFL reconstruction methods such as CAFE (Jin et al., 2021) exploit shared model gradients together with knowledge of the corresponding model. These attacks cannot be applied to HAMPL pre-training, which keeps the local encoder/projector parameters and gradients private. The uploaded prototype gradients provide aggregate constraints on private representations, corresponding to weighted mixtures of batch representations (Appendix D.3). Recovering raw features from such constraints requires additional model knowledge or suitable priors and identifiability conditions. The residual leakage through assignments and prototype updates is discussed below.

For label and attribute inference attacks based on fine-tuning a malicious passive participant's local model, attack success depends strongly on the representation capability acquired by that model during collaborative pre-training. Existing representation-sharing SSVFL frameworks combine instance-discriminative SSL, which aligns participants' representations, with partial model aggregation, which repeatedly synchronizes shared parameter components. These mechanisms strengthen coupling among local encoders and encourage their outputs to capture similar predictive information. The adversary's representations may consequently become closer to those of the active participant, allowing the adversary to infer some private labels or attributes by fine-tuning its own model using auxiliary annotated samples.

HAMPL instead emphasizes complementary participant-specific knowledge, so that local encoders contribute distinct views to downstream supervised learning. Avoiding encoder parameter aggregation and forced instance-level alignment reduces these sources of coupling between the adversary's and active participant's encoders. An adversary that fine-tunes only its own model must rely on its local view and the knowledge retained in that model, without jointly using the other participants' views. If predicting the target label or attribute requires complementary participant-specific information that is not captured by the adversary's model, this information gap can substantially limit inference. This motivates the hypothesis that a single HAMPL encoder acquires less additional label or attribute-predictive information from collaboration than in the representation-sharing SSVFL frameworks above, under comparable local features and auxiliary data. Appendix D.2 specifies a Model Completion (MC) evaluation (Fu et al., 2022) to test the label-inference hypothesis. Attribute inference requires a separate evaluation with sensitive-attribute annotations.

These architectural distinctions do not make HAMPL's messages information-free. Assignments expose sample-to-prototype relationships, and prototype gradients provide aggregate constraints on private representations. We characterize HAMPL's privacy boundaries by examining what information can be recovered from these communicated quantities. In particular, unperturbed Sinkhorn assignments can reveal centered projected representations when the server knows the assignment temperature ε_{a} and the public prototype bank satisfies the rank condition in Appendix D.3. These recovered representations describe each sample relative to the current-batch mean. Further reconstruction of the original data requires additional priors or identifiability assumptions; constructing the encoder-informed attack chain in Eq. (B.1) as described above additionally relies on knowledge of encoder architectures and parameters that HAMPL does not expose. These additional conditions are discussed in Appendix D.3. For the setting considered here, we therefore choose to retain the

base HAMPL protocol without additional perturbations such as isotropic Gaussian noise, avoiding their potential impact on training utility.

Backdoor threats. Unlike large-scale cross-device HFL, VFL typically involves a small group of organizations with complementary features and incentives to collaborate (Liu et al., 2024). We consider federations with two to four participants, consistent with our experimental settings. Because these participants benefit from a useful joint model, we regard deliberate Byzantine behavior aimed at disrupting convergence or degrading overall performance as less consistent with their incentives in the setting considered here. A malicious participant may nevertheless seek to implant a backdoor that induces attacker-chosen predictions on triggered inputs while preserving performance on clean data. Accordingly, our security analysis focuses on targeted backdoor attacks under these collaboration incentives.

During pre-training, HAMPL supplies no task-label gradients and performs no aggregation of participant models, removing those specific channels for label-guided trigger injection and the injection of backdoored models through aggregation. We therefore focus on backdoor threats during downstream supervised VFL fine-tuning. HAMPL can readily incorporate defenses developed for this stage, including feature-subspace recovery (Liu et al., 2021), as well as gradient sparsification and controlled noise on backward messages to mitigate representation-replacement attacks (Liu et al., 2020; Zou et al., 2024). These defenses act on the standard supervised VFL interface and can be integrated with HAMPL without changing its pre-training procedure.

Efficiency Analysis. In terms of GPU memory usage, HAMPL maintains one local encoder/projector per participant and uses meta-prototypes to connect the two pre-training stages, avoiding the additional memory required to maintain separate cross/local models for knowledge transfer. Following the analysis structure of Chen et al. (2026), we further quantify the computational overhead of pre-training on the participant and server sides. For both methods, we exclude architecture-dependent encoder/projector forward and backward passes and local updates of their parameters, while accounting for matching, fusion, and aggregation.

Computational complexity with queues. Let B_{a} and B_l^p denote the numbers of cross-training and local-training mini-batches in one epoch. Let L_{a}^p be the capacity of participant p's cross queue and L_l^p the capacity of *each* of its two local-view queues, excluding the current batch. The first-pass OT problems therefore contain at most $M_{\mathrm{a}}^p = m_{\mathrm{a}} + L_{\mathrm{a}}^p$ and $M_l^p = m_l^p + L_l^p$ representations, respectively. These give full-queue upper bounds; assignment estimation can cost less during queue filling. Cached representations are detached and used only for first-pass assignment estimation (Appendix A.2). Their similarities to the current prototypes are recomputed, whereas losses, consensus-guided OT, and backpropagation use only current-batch rows.

Before simplifying, we enumerate the participant-side costs. In the cross stage, score construction costs $O(M_{\mathrm{a}}^p N_C R)$, cross UOT costs $O(N_{\mathrm{iter,a}} M_{\mathrm{a}}^p N_C)$, and consensus-guided OT costs $O(N_{\mathrm{iter,g}} m_{\mathrm{a}} N_C)$ per batch. Prediction normalization and loss evaluation cost $O(m_{\mathrm{a}} N_C)$; gradients with respect to current representations and prototypes cost $O(m_{\mathrm{a}} N_C R)$. Representation/prototype normalization, prototype updates, and queue maintenance contribute at most $O((M_{\mathrm{a}}^p + N_C)R)$, including a possible shift or copy of the cached queue. The local stage incurs the corresponding costs for two views and their separate queues, without consensus-guided OT. Stage-boundary prototype copies or increment computation add $O(N_C R)$ per stage.

In our configuration, all three OT solvers use five iterations (Table 6). Since $m_{\mathrm{a}} \le M_{\mathrm{a}}^p$, the consensus-guided term satisfies $m_{\mathrm{a}} N_{\mathrm{iter,g}} \le M_{\mathrm{a}}^p N_{\mathrm{iter,a}}$ and is bounded by the first-pass OT term. With these fixed iteration budgets and nonempty training stages, all terms are bounded by the leading matching cost. The resulting per-epoch overheads T_{cross}^p and T_{local}^p are

$$\begin{aligned} T_{\mathrm{cross}}^p &= O(B_{\mathrm{a}}(m_{\mathrm{a}} + L_{\mathrm{a}}^p) N_C R)\,, \\ T_{\mathrm{local}}^p &= O(B_l^p (m_l^p + L_l^p) N_C R)\,. \end{aligned} \tag{B.2}$$

On the server, assignment normalization, entropy estimation, and weighted consensus construction together cost $O(P m_{\mathrm{a}} N_C)$ per cross batch. Prototype-gradient averaging costs $O(P N_C R)$, followed by an $O(N_C R)$ bank update and normalization. At the epoch boundary, local-increment aggregation and the meta-prototype update cost $O(P N_C R)$ and $O(N_C R)$, respectively. Thus, the server-side overhead per epoch is

$$T_{\mathrm{server}} = O(P B_{\mathrm{a}} N_C (m_{\mathrm{a}} + R))\,. \tag{B.3}$$

Comparison with DACLSC. Using the same batch notation and representation width R, the analysis of Chen et al. (2026) gives the participant-side overheads per epoch:

$$\begin{aligned} T^{p}_{\mathrm{DACLSC,cross}} &= O(B_{\mathrm{a}} m_{\mathrm{a}} R(m_{\mathrm{a}} + R)) \,, \\ T^{p}_{\mathrm{DACLSC,local}} &= O(B^{p}_{l} m^{p}_{l} L_Q R) \,, \qquad L_Q = 4096. \end{aligned} \tag{B.4}$$

Its cross stage includes sample-pair matching and dimension-wise correlation, while its local stage compares current samples with the negative queue. Server-side fusion costs $O(PB_{\mathrm{a}} m_{\mathrm{a}} R)$ per epoch, and partial model aggregation adds $O(PA_\theta)$, where A_θ is the number of scalar parameters aggregated per participant at the epoch boundary. HAMPL replaces these operations with sample-to-prototype matching and prototype updates. Its cross-stage dependence on batch size is linear for fixed queue capacity and prototype count, but its actual overhead also depends on N_C and the queue lengths. For example, on BHI and Avazu, HAMPL uses no queues ($L^p_{\mathrm{a}} = L^p_l = 0$) and $N_C = 256$. At a common batch size of 512 and representation width $R = 128$, its leading cross-stage and local-stage arithmetic terms are 0.4 and 0.0625 times those of DACLSC, respectively. These configurations therefore have substantially smaller leading computational terms, with constant factors omitted. Alongside this computational advantage, HAMPL provides stronger protection against direct disclosure of encoder representations, parameters, and gradients by keeping them local during pre-training.

B.2 Mutual Information Analysis for Cross Training

In this section, we analyze Eq. (6) from the perspective of mutual information. At a fixed training state, expectations are over a randomly drawn aligned mini-batch and a uniformly selected row b. We use Ω to denote the conditioning context: the current model parameters, prototype bank, detached queues, the selected index b, and the batch rows $\{\mathbf{z}^{p'}_{\mathrm{a},b'} : b' \neq b,\ p' \in \mathcal{P}\}$. In particular, Ω excludes the selected entity's representations $\{\mathbf{z}^{p'}_{\mathrm{a},b}\}_{p' \in \mathcal{P}}$ at every participant. These representations remain random. Conditioning on Ω fixes the other rows used by Sinkhorn, so each first-pass assignment for row b becomes a function of that participant's selected representation $\mathbf{z}^{p}_{\mathrm{a},b}$. Write $\mathbf{z}^{-p}_{\mathrm{a},b} = (\mathbf{z}^{p'}_{\mathrm{a},b})_{p' \neq p}$ for the selected entity's representations at the other participants. Throughout, $P, N_C \geq 2$, logarithms are natural, and H and I denote Shannon entropy and mutual information.

Ideal cross-participant prediction. Let $\bar{\mathbf{q}}^{p}_{\mathrm{a},b}$ and $\mathbf{q}^{G}_{\mathrm{a},b}$ be the corresponding rows of $\bar{\mathbf{Q}}^{p}_{\mathrm{a}}$ and $\mathbf{Q}^{G}_{\mathrm{a}}$, and let $w^{p}_{\mathrm{a},b}$ be the confidence in Eq. (3). The analysis uses these probability rows, obtained by normalizing the first-pass OT masses. To measure how well participant p can predict the other views' information from its own representation, we construct a reference target using only the other participants' normalized assignments:

$$\mathbf{q}^{-p}_{\mathrm{a},b} = \sum_{p' \neq p} \frac{w^{p'}_{\mathrm{a},b}}{\sum_{\ell \neq p} w^{\ell}_{\mathrm{a},b}} \bar{\mathbf{q}}^{p'}_{\mathrm{a},b}, \qquad \omega^{p}_{b} = \frac{w^{p}_{\mathrm{a},b}}{\sum_{\ell=1}^{P} w^{\ell}_{\mathrm{a},b}}. \tag{B.5}$$

Here $\mathbf{q}^{-p}_{\mathrm{a},b}$ summarizes only the other participants, while ω^{p}_{b} is participant p's weight in the actual all-participant consensus. This reference target isolates information supplied by the other views for the mutual-information bound. HAMPL itself still constructs $\mathbf{Q}^{G}_{\mathrm{a}}$ and refines it using Eq. (5).

If every other participant assigns equal probability to all prototypes, Eq. (3) gives zero confidence to all these rows. Their weighted average then has a zero denominator. In this case, we define the reference target as the same uniform distribution, $\mathbf{q}^{-p}_{\mathrm{a},b} = \mathbf{r}^{\top}$. This only completes the definition of the reference target. For the actual all-participant fusion, we assume $\sum_{\ell=1}^{P} w^{\ell}_{\mathrm{a},b} > 0$, as required by Eq. (3).

For the proof, introduce an auxiliary prototype index $A^{-p} \in \{1, \ldots, N_C\}$ with conditional probabilities

$$\Pr\Big(A^{-p} = k \,\Big|\, \mathbf{z}^{p}_{\mathrm{a},b}, \mathbf{z}^{-p}_{\mathrm{a},b}, \Omega\Big) = q^{-p}_{\mathrm{a},b,k}, \qquad k = 1, \ldots, N_C.$$

This is the categorical distribution represented by the soft assignment vector. Given Ω, its probabilities depend only on the other participants' selected representations, so A^{-p} is conditionally

independent of $\mathbf{z}^p_{\mathrm{a},b}$ given $\mathbf{z}^{-p}_{\mathrm{a},b}$. Averaging over the selected entity's representations gives its conditional marginal distribution and entropy deficit:

$$\begin{aligned} \boldsymbol{\mu}^{-p}(\Omega) = \mathbb{E}[\mathbf{q}^{-p}_{\mathrm{a},b} \mid \Omega], \qquad \eta^{-p} = \mathbb{E}_\Omega \mathrm{KL}\big(\boldsymbol{\mu}^{-p}(\Omega) \big\| \mathbf{r}^\top\big)\,, \\ H(A^{-p} \mid \Omega) = \log N_C - \eta^{-p}. \end{aligned} \tag{B.6}$$

The relevant entropy is that of the marginal prototype distribution. Neither low per-row entropy in confidence weighting nor balanced marginals of the refined targets implies $\eta^{-p} = 0$ for this ideal consensus. Define its prediction loss as

$$\mathcal{L}^p_{\mathrm{LOO}} = -\mathbb{E} \sum_{k=1}^{N_C} q^{-p}_{\mathrm{a},b,k} \log \pi_k(\mathbf{z}^p_{\mathrm{a},b}; \mathcal{C}). \tag{B.7}$$

Proposition 1 (Ideal mutual-information lower bound). For each participant p,

$$I(\mathbf{z}^p_{\mathrm{a},b}; \mathbf{z}^{-p}_{\mathrm{a},b} \mid \Omega) \geq I(\mathbf{z}^p_{\mathrm{a},b}; A^{-p} \mid \Omega) \geq \log N_C - \eta^{-p} - \mathcal{L}^p_{\mathrm{LOO}}. \tag{B.8}$$

Proof. We first derive the cross-entropy decomposition. By the definition of A^{-p} and the law of total expectation, its conditional probabilities given participant p's representation are

$$\rho_k := \Pr(A^{-p} = k \mid \mathbf{z}^p_{\mathrm{a},b}, \Omega) = \mathbb{E}\Big[q^{-p}_{\mathrm{a},b,k} \mid \mathbf{z}^p_{\mathrm{a},b}, \Omega\Big]\,, \qquad k = 1, \ldots, N_C.$$

Thus ρ_k averages the other participants' assignment probability over their representations, conditional on $\mathbf{z}^p_{\mathrm{a},b}$ and Ω. Within this proof, abbreviate $\pi_k = \pi_k(\mathbf{z}^p_{\mathrm{a},b}; \mathcal{C})$ and let $\boldsymbol{\rho}$ and $\boldsymbol{\pi}$ collect the corresponding probabilities. Since π_k is determined by $\mathbf{z}^p_{\mathrm{a},b}$ and Ω, conditioning Eq. (B.7) on these variables replaces $q^{-p}_{\mathrm{a},b,k}$ by ρ_k inside the expectation. Adding and subtracting the conditional-entropy term then yields

$$\begin{aligned} \mathcal{L}^p_{\mathrm{LOO}} &= -\mathbb{E} \sum_{k=1}^{N_C} \rho_k \log \pi_k \\ &= -\mathbb{E} \sum_{k=1}^{N_C} \rho_k \log \rho_k + \mathbb{E} \sum_{k=1}^{N_C} \rho_k \log \frac{\rho_k}{\pi_k} \\ &= H(A^{-p} \mid \mathbf{z}^p_{\mathrm{a},b}, \Omega) + \mathbb{E}\, \mathrm{KL}(\boldsymbol{\rho} \| \boldsymbol{\pi}). \end{aligned} \tag{B.9}$$

The conditional entropy is the irreducible uncertainty when predicting the other views' prototype code from participant p's representation; the KL term measures the discrepancy between its true conditional distribution and the predicted probabilities. Nonnegativity of KL divergence gives $H(A^{-p} \mid \mathbf{z}^p_{\mathrm{a},b}, \Omega) \leq \mathcal{L}^p_{\mathrm{LOO}}$. Combining this inequality with Eq. (B.6) and the definition of conditional mutual information gives

$$\begin{aligned} I(\mathbf{z}^p_{\mathrm{a},b}; A^{-p} \mid \Omega) &= H(A^{-p} \mid \Omega) - H(A^{-p} \mid \mathbf{z}^p_{\mathrm{a},b}, \Omega) \\ &\geq \log N_C - \eta^{-p} - \mathcal{L}^p_{\mathrm{LOO}}, \end{aligned}$$

which is the second inequality in Eq. (B.8). Finally, given $\mathbf{z}^{-p}_{\mathrm{a},b}$ and Ω, the sampling probabilities of A^{-p} do not depend on $\mathbf{z}^p_{\mathrm{a},b}$. This gives the conditional Markov chain

$$\mathbf{z}^p_{\mathrm{a},b} \longrightarrow \mathbf{z}^{-p}_{\mathrm{a},b} \longrightarrow A^{-p} \qquad \text{(conditional on } \Omega\text{)}.$$

The conditional data-processing inequality therefore gives $I(\mathbf{z}^p_{\mathrm{a},b}; \mathbf{z}^{-p}_{\mathrm{a},b} \mid \Omega) \geq I(\mathbf{z}^p_{\mathrm{a},b}; A^{-p} \mid \Omega)$, completing Eq. (B.8). Excluding participant p from the reference target ensures that this code is generated solely from the other views, as required for the data-processing argument. No identical feature spaces or representations are assumed.

Approximation by the implemented objective. HAMPL uses the refined target $\widetilde{\mathbf{q}}^p_{\mathrm{a},b}$ from Eq. (5), whose prior includes participant p. Equation (3) gives the exact decomposition

$$\mathbf{q}^G_{\mathrm{a},b} = \omega^p_b \bar{\mathbf{q}}^p_{\mathrm{a},b} + (1 - \omega^p_b) \mathbf{q}^{-p}_{\mathrm{a},b}. \tag{B.10}$$

The triangle inequality and Pinsker's inequality therefore give

$$\left\|\widetilde{\mathbf{q}}^{p}_{\mathrm{a},b} - \mathbf{q}^{-p}_{\mathrm{a},b}\right\|_1 \le \omega^{p}_{b}\left\|\bar{\mathbf{q}}^{p}_{\mathrm{a},b} - \mathbf{q}^{-p}_{\mathrm{a},b}\right\|_1 + \sqrt{2\,\mathrm{KL}\Big(\widetilde{\mathbf{q}}^{p}_{\mathrm{a},b}\Big\|\mathbf{q}^{G}_{\mathrm{a},b}\Big)}. \tag{B.11}$$

Define the two expected deviations by

$$\begin{aligned}\mathcal{E}^{p}_{\mathrm{self}} &= \mathbb{E}\Big[\omega^{p}_{b}\left\|\bar{\mathbf{q}}^{p}_{\mathrm{a},b} - \mathbf{q}^{-p}_{\mathrm{a},b}\right\|_1\Big],\\ \mathcal{E}^{p}_{\mathrm{OT}} &= \mathbb{E}\sqrt{2\,\mathrm{KL}\Big(\widetilde{\mathbf{q}}^{p}_{\mathrm{a},b}\Big\|\mathbf{q}^{G}_{\mathrm{a},b}\Big)}.\end{aligned} \tag{B.12}$$

The first term measures inclusion of the participant's own assignment, weighted by its disagreement with the other views; the second measures the change introduced by consensus-guided refinement. The bound uses the actual row-normalized solver outputs and requires no exact OT optimality. Finite Sinkhorn iterations and numerical flooring of the consensus prior are thus covered by the resulting refinement deviation.

Proposition 2 (HAMPL mutual-information lower bound). With $B_\tau = \log N_C + 2/\tau$, the population cross-training loss satisfies

$$\begin{aligned}I(\mathbf{z}^{p}_{\mathrm{a},b}; \mathbf{z}^{-p}_{\mathrm{a},b} \mid \Omega) &\ge I(\mathbf{z}^{p}_{\mathrm{a},b}; A^{-p} \mid \Omega)\\ &\ge \log N_C - \eta^{-p} - \mathbb{E}\mathcal{L}^{p}_{\mathrm{HASSL}} - B_\tau(\mathcal{E}^{p}_{\mathrm{self}} + \mathcal{E}^{p}_{\mathrm{OT}}).\end{aligned} \tag{B.13}$$

Proof. The normalization used in Eq. (6) gives $|\mathbf{z}^{p}_{\mathrm{a},b}(c^k)^\top| \le 1$, hence $\pi_k(\mathbf{z}^{p}_{\mathrm{a},b}; \mathcal{C}) \ge e^{-2/\tau}/N_C$ and $0 \le -\log \pi_k \le B_\tau$. Both target rows have unit mass, so their cross-entropies differ by at most $B_\tau\|\widetilde{\mathbf{q}}^{p}_{\mathrm{a},b} - \mathbf{q}^{-p}_{\mathrm{a},b}\|_1$. Averaging over the random batch and row in Eq. (B.11) yields

$$|\mathbb{E}\mathcal{L}^{p}_{\mathrm{HASSL}} - \mathcal{L}^{p}_{\mathrm{LOO}}| \le B_\tau(\mathcal{E}^{p}_{\mathrm{self}} + \mathcal{E}^{p}_{\mathrm{OT}}). \tag{B.14}$$

Combining this inequality with Proposition 1 proves the result. The same argument applies to the participant-averaged objective underlying the shared prototype-gradient update.

Corollary (Shared predictable structure through meta-prototypes). If the right-hand side of Eq. (B.13) is positive, then $I(\mathbf{z}^{p}_{\mathrm{a},b}; A^{-p} \mid \Omega) > 0$: different views contain shared predictable structure expressible through the common discrete prototype code. For fixed target-generating distributions, η^{-p} is constant, so minimizing $\mathcal{L}^{p}_{\mathrm{LOO}}$ maximizes the ideal lower bound in Eq. (B.8). By Eq. (B.14), when the weighted assignment disagreement and refinement deviation are small, HAMPL's current cross-training objective approximately optimizes the ideal cross-participant mutual-information lower bound. The required shared predictability is expressed through common discrete prototypes and does not require identical feature spaces or encoder outputs.

C Detailed Formulations and Algorithms of HAMPL

C.1 Cross Unbalanced OT

Transport mass and regularization. We use the same sample-by-prototype orientation as in the main text: the entry $Q_{b,k}$ assigns mass from sample b to prototype k. The optimization variable in Eq. (2) is nonnegative, with no hard constraint on its total mass or either marginal. For nonnegative arrays $\mathbf{A}$ and strictly positive arrays $\mathbf{B}$ of the same shape, we define generalized KL divergence and entropy as

$$\mathrm{KL}(\mathbf{A}\|\mathbf{B}) = \sum_{\xi}\left[A_\xi \log\frac{A_\xi}{B_\xi} - A_\xi + B_\xi\right], \qquad \mathcal{H}(\mathbf{Q}) = -\sum_{b,k} Q_{b,k}(\log Q_{b,k} - 1), \tag{C.1}$$

where ξ ranges over all array entries and $0\log 0 = 0$ by continuity. For matrices, $\langle \mathbf{A}, \mathbf{B}\rangle_F = \sum_{b,k} A_{b,k}B_{b,k}$. The linear terms are retained because the transport mass is variable during optimization. The output is globally rescaled only after the scaling iterations, as specified below.

For participant p, the score matrix is $\mathbf{S}^{p}_{\mathrm{a}} = \mathbf{Z}^{p}_{\mathrm{a}}\mathcal{C}^\top \in \mathbb{R}^{m_{\mathrm{a}}\times N_C}$, with scalar entries $S^{p}_{\mathrm{a},b,k} = \mathbf{z}^{p}_{\mathrm{a},b}(c^k)^\top$. The sample reference marginal is $\mathbf{s}_{m_{\mathrm{a}}}$, and the prototype reference marginal is $\mathbf{r}$. Equation (2) balances similarity maximization against entropy and deviations of both marginals. In particular, larger ρ_{a} or ν_{a} more strongly penalizes the corresponding marginal deviation; their infinite-penalty limits recover the respective hard constraints.

Let $\mathbf{a} = \mathbf{Q}\mathbf{1}_{N_C}$ and $\mathbf{d} = \mathbf{Q}^\top \mathbf{1}_{m_\mathrm{a}}$ denote the sample and prototype marginal column vectors of a positive transport matrix. Specifically, $a_b = \sum_{k=1}^{N_C} Q_{b,k}$ is the total mass assigned by sample b to all prototypes, whereas $d_k = \sum_{b=1}^{m_\mathrm{a}} Q_{b,k}$ is the total mass received by prototype k from all samples. Using the definitions in Eq. (C.1), the first-order condition for Eq. (2) is

$$S^p_{\mathrm{a},b,k} - \varepsilon_\mathrm{a} \log Q_{b,k} - \nu_\mathrm{a} \log \frac{a_b}{s_{m_\mathrm{a},b}} - \rho_\mathrm{a} \log \frac{d_k}{r_k} = 0. \tag{C.2}$$

This yields the scaling form $\mathbf{T} = \mathrm{diag}(\mathbf{u})\mathbf{K}\,\mathrm{diag}(\mathbf{v})$, where diag places a vector on the diagonal, $K_{b,k} = \exp(S^p_{\mathrm{a},b,k}/\varepsilon_\mathrm{a})$, $\mathbf{u} \in \mathbb{R}_+^{m_\mathrm{a}\times 1}$ scales samples, and $\mathbf{v} \in \mathbb{R}_+^{N_C\times 1}$ scales prototypes. Both scaling vectors are column vectors. The entry $T_{b,k} = u_b K_{b,k} v_k$ denotes the unnormalized transport mass from sample b to prototype k. Define

$$\gamma_{\mathcal{C},\mathrm{a}} = \frac{\rho_\mathrm{a}}{\rho_\mathrm{a} + \varepsilon_\mathrm{a}}, \qquad \gamma_{\mathcal{B},\mathrm{a}} = \frac{\nu_\mathrm{a}}{\nu_\mathrm{a} + \varepsilon_\mathrm{a}}. \tag{C.3}$$

Starting from all-ones scaling vectors, the solver performs $N_{\mathrm{iter,a}}$ iterations of the following updates, where $N_{\mathrm{iter,a}}$ is a positive integer:

$$\mathbf{v} \leftarrow \left(\frac{\mathbf{r}}{\mathbf{K}^\top \mathbf{u}}\right)^{\gamma_{\mathcal{C},\mathrm{a}}}, \qquad \mathbf{u} \leftarrow \left(\frac{\mathbf{s}_{m_\mathrm{a}}}{\mathbf{K}\mathbf{v}}\right)^{\gamma_{\mathcal{B},\mathrm{a}}}, \tag{C.4}$$

with elementwise division and powers. The prototype scaling precedes sample scaling, as in the implementation. At convergence, $\mathbf{T}$ is the continuous maximizer of Eq. (2); a finite iteration budget gives an approximate solution.

Before returning the assignments, the implementation divides the entire matrix by its total mass and then multiplies it by the batch size:

$$\mathbf{Q}^p_\mathrm{a} = m_\mathrm{a} \frac{\mathbf{T}}{\sum_{b=1}^{m_\mathrm{a}} \sum_{k=1}^{N_C} T_{b,k}}. \tag{C.5}$$

This gives $\mathbf{Q}^p_\mathrm{a}$ the total mass m_a, matching the scale of the assignment-prediction loss, while preserving all relative sample and prototype masses.

Log-sum-exp notation. For a real matrix $\mathbf{A} \in \mathbb{R}^{m\times N_C}$ with sample index b and prototype index k, log-sum-exp (LSE) means exponentiating the entries, summing over the specified indices, and then taking the logarithm:

$$\mathrm{LSE}_b(A_{b,k}) = \log\left(\sum_{b=1}^{m} \exp(A_{b,k})\right), \qquad \text{for fixed } k,$$

$$\mathrm{LSE}_k(A_{b,k}) = \log\left(\sum_{k=1}^{N_C} \exp(A_{b,k})\right), \qquad \text{for fixed } b,$$

$$\mathrm{LSE}_{b,k}(A_{b,k}) = \log\left(\sum_{b=1}^{m}\sum_{k=1}^{N_C} \exp(A_{b,k})\right).$$

Thus LSE_b reduces along the sample axis and gives one value per prototype, LSE_k reduces along the prototype axis and gives one value per sample, and $\mathrm{LSE}_{b,k}$ reduces over all matrix entries to a single scalar. Indices not listed in the LSE subscript are held fixed. Throughout Algorithms 1–3, $m = m_\mathrm{a}$ for cross-party computations and $m = m_l^p$ for local computations.

Algorithm 1 Sinkhorn–Knopp Solver for Cross Unbalanced OT

Require: Scores $\mathbf{S}^p_\mathrm{a} \in \mathbb{R}^{m_\mathrm{a}\times N_C}$; $\varepsilon_\mathrm{a}, \rho_\mathrm{a}, \nu_\mathrm{a} > 0$; iterations $N_{\mathrm{iter,a}}$.
Ensure: Assignment matrix $\mathbf{Q}^p_\mathrm{a} \in \mathbb{R}_+^{m_\mathrm{a}\times N_C}$ with total mass m_a.
$\mathbf{L} \leftarrow \log \mathbf{K} = \mathbf{S}^p_\mathrm{a}/\varepsilon_\mathrm{a}$; $\alpha_b \leftarrow 0$ for all b; $\beta_k \leftarrow 0$ for all k.
$\gamma_{\mathcal{C},\mathrm{a}} \leftarrow \rho_\mathrm{a}/(\rho_\mathrm{a} + \varepsilon_\mathrm{a})$; $\gamma_{\mathcal{B},\mathrm{a}} \leftarrow \nu_\mathrm{a}/(\nu_\mathrm{a} + \varepsilon_\mathrm{a})$.
for $n = 1, \ldots, N_{\mathrm{iter,a}}$ **do**
    $\beta_k \leftarrow \gamma_{\mathcal{C},\mathrm{a}}\big[-\log N_C - \mathrm{LSE}_b(L_{b,k} + \alpha_b)\big]$ for every k.
    $\alpha_b \leftarrow \gamma_{\mathcal{B},\mathrm{a}}\big[-\log m_\mathrm{a} - \mathrm{LSE}_k(L_{b,k} + \beta_k)\big]$ for every b.
end for
$A_{b,k} \leftarrow L_{b,k} + \alpha_b + \beta_k$ for all (b,k).
$Q^p_{\mathrm{a},b,k} \leftarrow m_\mathrm{a} \exp\big(A_{b,k} - \mathrm{LSE}_{b',k'}(A_{b',k'})\big)$ for all (b,k).
return $\mathbf{Q}^p_\mathrm{a}$.

Here $\mathbf{L} = \log \mathbf{K}$ is the elementwise logarithm of the kernel matrix, $\alpha_b = \log u_b$ and $\beta_k = \log v_k$ are log scaling factors, $\mathbf{A}$ is the resulting log transport matrix, and n is the iteration index. All assignment computations are performed without gradient tracking.

C.2 CONSENSUS-GUIDED OT

Consensus as a soft prior. The server consensus $\mathbf{Q}_{\mathrm{a}}^{G}$ has unit row sums, whereas a transport matrix in $\mathcal{U}_{m_{\mathrm{a}}}$ has total mass one. The prior in Eq. (5) is therefore $\mathbf{Q}_{\mathrm{a}}^{G}/m_{\mathrm{a}}$. It need not satisfy the uniform prototype marginal: the forward KL term encourages agreement with this prior while the constraints enforce balanced marginals on the refined transport. Consequently, the corrected assignments need not equal the consensus or each other, since the participant-specific scores $\mathbf{S}_{\mathrm{a}}^{p}$ also enter the objective. We use the generalized definitions in Eq. (C.1); here the balanced constraints make their additional linear terms constant.

Expanding the KL term in Eq. (5) yields the equivalent maximization

$$\max_{\mathbf{Q}\in\mathcal{U}_{m_{\mathrm{a}}}} \left\langle \mathbf{Q}, \mathbf{S}_{\mathrm{a}}^{p} + \lambda \log(\mathbf{Q}_{\mathrm{a}}^{G}/m_{\mathrm{a}}) \right\rangle_{F} + (\varepsilon_{\mathrm{g}} + \lambda)\mathcal{H}(\mathbf{Q}), \tag{C.6}$$

up to a constant independent of $\mathbf{Q}$, with an elementwise logarithm. The associated positive kernel can thus be written as

$$K_{b,k}^{\mathrm{g}} = \exp\left(\frac{S_{\mathrm{a},b,k}^{p} + \lambda \log Q_{\mathrm{a},b,k}^{G}}{\varepsilon_{\mathrm{g}} + \lambda} \right). \tag{C.7}$$

Omitting $1/m_{\mathrm{a}}$ that does not affect the balanced solution. For this kernel, the normalized optimizer has the form $\mathbf{T}_{\mathrm{g}}^{p,*} = \mathrm{diag}(\mathbf{u})\mathbf{K}^{\mathrm{g}}\,\mathrm{diag}(\mathbf{v})$, and the returned assignment is $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p} = m_{\mathrm{a}}\mathbf{T}_{\mathrm{g}}^{p,*}$. Unlike Eq. (C.4), the balanced updates use unit exponents:

$$\mathbf{v} \leftarrow \frac{\mathbf{r}}{(\mathbf{K}^{\mathrm{g}})^{\top}\mathbf{u}}, \qquad \mathbf{u} \leftarrow \frac{\mathbf{s}_{m_{\mathrm{a}}}}{\mathbf{K}^{\mathrm{g}}\mathbf{v}}. \tag{C.8}$$

Algorithm 2 implements equivalent alternating log-domain marginal projections. The positive integer $N_{\mathrm{iter,g}}$ is its iteration budget. After applying a numerical floor $\delta > 0$ to avoid taking the logarithm of zero, we renormalize the consensus prior rowwise to restore unit row sums. With positive inputs, the scaling iterations converge to the solution of the regularized balanced problem. After a finite number of iterations, the last row projection gives unit row sums in $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p}$, while the prototype marginal is satisfied approximately.

Algorithm 2 Sinkhorn–Knopp Solver for Consensus-Guided OT

Require: Scores $\mathbf{S}_{\mathrm{a}}^{p}$, consensus $\mathbf{Q}_{\mathrm{a}}^{G} \in \mathbb{R}_{+}^{m_{\mathrm{a}}\times N_C}$; $\varepsilon_{\mathrm{g}} > 0$, $\lambda \geq 0$; iterations $N_{\mathrm{iter,g}}$; floor $\delta > 0$.
Ensure: Refined assignment matrix $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p} \in \mathbb{R}_{+}^{m_{\mathrm{a}}\times N_C}$.

$W_{b,k} \leftarrow \max(Q_{\mathrm{a},b,k}^{G}, \delta)$ and $\widehat{Q}_{b,k}^{G} \leftarrow W_{b,k} / \sum_{k'=1}^{N_C} W_{b,k'}$.
$A_{b,k} \leftarrow \left(S_{\mathrm{a},b,k}^{p} + \lambda \log \widehat{Q}_{b,k}^{G}\right)/(\varepsilon_{\mathrm{g}} + \lambda)$.
$\mathbf{A} \leftarrow \mathbf{A} - \mathrm{LSE}_{b,k}(A_{b,k})$.
for $n = 1, \ldots, N_{\mathrm{iter,g}}$ **do**
    $A_{b,k} \leftarrow A_{b,k} - \mathrm{LSE}_{b'}(A_{b',k}) - \log N_C$ for all (b,k).
    $A_{b,k} \leftarrow A_{b,k} - \mathrm{LSE}_{k'}(A_{b,k'}) - \log m_{\mathrm{a}}$ for all (b,k).
end for
$\widetilde{Q}_{\mathrm{a},b,k}^{p} \leftarrow m_{\mathrm{a}} \exp(A_{b,k})$ for all (b,k).
return $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p}$.

In Algorithm 2, $\mathbf{W}$ is the floored consensus, $\widehat{\mathbf{Q}}^{G}$ is its row-normalized version, and $\mathbf{A}$ is the evolving log transport matrix. The initial subtraction of $\mathrm{LSE}_{b,k}(A_{b,k})$ applies the same scalar to every entry, giving $\sum_{b,k} \exp(A_{b,k}) = 1$. Within each iteration, $\mathrm{LSE}_{b'}(A_{b',k})$ performs log-sum-exp along the sample axis for each fixed prototype k, whereas $\mathrm{LSE}_{k'}(A_{b,k'})$ performs it along the prototype axis for each fixed sample b. Each marginal projection updates all entries simultaneously. The refined assignments are detached before evaluating $\mathcal{L}_{\mathrm{HASSL}}^{p}$; gradients flow through prototype prediction into the participant model and the current cross-stage bank $\mathcal{C}$, not through either OT solver or consensus

fusion. The balanced refinement should be distinguished from the first unbalanced pass: it does not itself remove the uniform prototype-marginal constraint.

We retain balanced marginals in consensus-guided OT because the first cross-unbalanced pass already estimates participant-specific matching preferences under soft marginal constraints, allowing nonuniform prototype usage. Building on cross unbalanced OT, entropy-guided consensus fusion further incorporates these preferences into $\mathbf{Q}_{\mathrm{a}}^{G}$, whose prototype marginal need not be uniform. Using this consensus as a soft prior carries information about sample-specific matching preferences into the second rematching step.

From a numerical perspective, our experiments indicate that cross unbalanced OT achieves good performance even under pronounced long-tailed distributions with $\rho_{\mathrm{a}} = 0.5$ and $\varepsilon_{\mathrm{a}} = 0.01$, corresponding to the scaling exponent $\gamma_{\mathcal{C},\mathrm{a}} = \rho_{\mathrm{a}}/(\rho_{\mathrm{a}} + \varepsilon_{\mathrm{a}}) \approx 0.980$. In consensus-guided OT, however, the effective entropy coefficient is $\varepsilon_{\mathrm{g}} + \lambda$, which is substantially larger than ε_{g} under our settings. If a prototype-marginal KL penalty with coefficient ρ_{g} were introduced in this second pass, its scaling exponent would become $\rho_{\mathrm{g}}/(\rho_{\mathrm{g}} + \varepsilon_{\mathrm{g}} + \lambda)$. Thus, adding marginal relaxation would introduce an additional hyperparameter coupled to the consensus strength and require separate calibration. This motivates retaining balanced marginals in the second pass.

Moreover, for $\lambda > 0$, the larger denominator in Eq. (C.7) reduces the amplification of participant-specific similarity differences compared with scaling them by ε_{g} alone. The final assignments are obtained through a small number of balancing iterations guided by the consensus prior. The tempered similarity scores, together with a less concentrated prior, can help avoid overly concentrated assignments. In particular, when participants favor different prototypes, their weighted aggregation can yield a less concentrated consensus, which can further encourage $\widetilde{\mathbf{Q}}_{\mathrm{a}}^{p}$ to distribute a sample's assignment mass across multiple meta-prototypes. We retain these continuous soft assignments without discretization, providing sample-level matching flexibility. This aligns with our central idea of enabling multiple meta-prototypes to simultaneously absorb representational knowledge while accommodating cross-participant semantic discrepancies arising from feature and model heterogeneity.

C.3 Local Unbalanced OT

View-specific Local Unbalanced OT. Participant p matches both augmented views to its current local bank $\mathcal{C}_l^p$. The score matrices $\mathbf{S}_{l,j}^{p} \in \mathbb{R}^{m_l^p \times N_C}$ for $j \in \{1,2\}$ are defined in Section 2.2.2, with scalar entries $S_{l,j,b,k}^{p} = \mathbf{z}_{l,b,j}^{p}(c_l^{p,k})^{\top}$. For each view, Eq. (7) uses the generalized entropy and KL divergence in Eq. (C.1), with only a nonnegativity constraint on the transport matrix. Its sample reference marginal is $\mathbf{s}_{m_l^p}$ and its prototype reference marginal is $\mathbf{r}$. Both remain soft references, allowing nonuniform prototype occupancy and nonuniform assignment mass across samples.

With $K_{j,b,k}^{l} = \exp(S_{l,j,b,k}^{p}/\varepsilon_l)$, the intermediate scaling matrix is $\mathbf{T}_{l,j}^{p} = \mathrm{diag}(\mathbf{u}_j)\mathbf{K}_j^{l}\,\mathrm{diag}(\mathbf{v}_j)$, where $\mathbf{u}_j$ and $\mathbf{v}_j$ are the view-specific sample and prototype scaling column vectors. Its (b,k)-th entry $T_{l,j,b,k}^{p} = u_{j,b}K_{j,b,k}^{l}v_{j,k}$ denotes the unnormalized transport mass from sample b in view j to

Algorithm 3 Sinkhorn–Knopp Solver for Local Unbalanced OT

Require: Scores $\mathbf{S}_{l,1}^{p}, \mathbf{S}_{l,2}^{p} \in \mathbb{R}^{m_l^p \times N_C}$; $\varepsilon_l, \rho_l, \nu_l > 0$; iterations $N_{\mathrm{iter},l}$.

Ensure: Assignment matrices $\mathbf{Q}_{l,1}^{p}, \mathbf{Q}_{l,2}^{p} \in \mathbb{R}_{+}^{m_l^p \times N_C}$.

$\gamma_{\mathcal{C},l} \leftarrow \rho_l/(\rho_l + \varepsilon_l)$; $\gamma_{\mathcal{B},l} \leftarrow \nu_l/(\nu_l + \varepsilon_l)$.
for $j \in \{1,2\}$ **do**
  $\mathbf{L} \leftarrow \log \mathbf{K}_j^{l} = \mathbf{S}_{l,j}^{p}/\varepsilon_l$; $\alpha_b \leftarrow 0$ for all b; $\beta_k \leftarrow 0$ for all k.
  for $n = 1, \ldots, N_{\mathrm{iter},l}$ **do**
    $\beta_k \leftarrow \gamma_{\mathcal{C},l}\big[-\log N_C - \mathrm{LSE}_b(L_{b,k} + \alpha_b)\big]$ for every k.
    $\alpha_b \leftarrow \gamma_{\mathcal{B},l}\big[-\log m_l^p - \mathrm{LSE}_k(L_{b,k} + \beta_k)\big]$ for every b.
  end for
  $A_{b,k} \leftarrow L_{b,k} + \alpha_b + \beta_k$ for all (b,k).
  $Q_{l,j,b,k}^{p} \leftarrow m_l^p \exp\big(A_{b,k} - \mathrm{LSE}_{b',k'}(A_{b',k'})\big)$ for all (b,k).
end for
return $\mathbf{Q}_{l,1}^{p}, \mathbf{Q}_{l,2}^{p}$.

prototype k at participant p, where l indicates the local stage. Here $u_{j,b}$ and $v_{j,k}$ are the b-th and k-th entries of $\mathbf{u}_j$ and $\mathbf{v}_j$, respectively. Define

$$\gamma_{\mathcal{C},l} = \frac{\rho_l}{\rho_l + \varepsilon_l}, \qquad \gamma_{\mathcal{B},l} = \frac{\nu_l}{\nu_l + \varepsilon_l}. \tag{C.9}$$

Starting from all-ones scaling vectors, the solver alternates

$$\mathbf{v}_j \leftarrow \left(\frac{\mathbf{r}}{(\mathbf{K}_j^l)^\top \mathbf{u}_j} \right)^{\gamma_{\mathcal{C},l}}, \qquad \mathbf{u}_j \leftarrow \left(\frac{\mathbf{s}_{m_l^p}}{\mathbf{K}_j^l \mathbf{v}_j} \right)^{\gamma_{\mathcal{B},l}}. \tag{C.10}$$

Algorithm 3 runs these updates independently for the two views, with positive integer iteration budget $N_{\mathrm{iter},l}$. After iteration, each intermediate matrix is globally normalized and then multiplied by the local batch size before return:

$$\mathbf{Q}_{l,j}^p = m_l^p \frac{\mathbf{T}_{l,j}^p}{\sum_{b=1}^{m_l^p} \sum_{k=1}^{N_C} T_{l,j,b,k}^p}, \qquad j \in \{1,2\}. \tag{C.11}$$

The first-order condition in Appendix C.1 applies with the local scores and coefficients: the scaling iterations approximate the raw maximizer of Eq. (7), and Eq. (C.11) subsequently rescales it. The total returned mass is m_l^p, without imposing unit row sums.

For each view j, $\mathbf{L} = \log \mathbf{K}_j^l$ is the elementwise log kernel, $\alpha_b = \log u_{j,b}$, and $\beta_k = \log v_{j,k}$. These temporary variables and the log transport matrix $\mathbf{A}$ are reinitialized for each view. The returned assignment vectors are swapped between the two views: $\mathbf{q}_{l,b,1}^p$ supervises the prediction from $\mathbf{z}_{l,b,2}^p$, and $\mathbf{q}_{l,b,2}^p$ supervises the prediction from $\mathbf{z}_{l,b,1}^p$. Both assignments are treated as fixed targets in Eq. (8), which updates the local encoder, projector, and current local bank $\mathcal{C}_l^p$ through the prediction probabilities.

Continuous versus discrete assignments. The UOT outputs can optionally be normalized rowwise to obtain per-sample probabilities, or discretized to obtain sharper, hard assignments. Row normalization alone preserves the relative weights within each sample and should not be confused with sharpening. In our minibatch-only online experiments, discrete codes perform worse than continuous UOT assignments without additional row normalization, consistent with the preference for soft over hard codes reported by SwAV (Caron et al., 2020). A plausible explanation is that rounding changes the targets more abruptly than the gradual adaptation induced by gradient updates: it can accelerate convergence while promoting premature commitment to inferior solutions. We therefore retain soft codes rather than rounding them, preserving the sample-dependent row masses in local training. These codes still undergo the global total-mass rescaling in Eq. (C.11); they are not normalized separately for each sample.

C.4 HAMPL Pre-Training Procedure

Algorithm 4 summarizes the complete pre-training procedure in Sections 2.2.1 and 2.2.2, using the same notation. The epoch index starts at $t = 0$, and $\mathcal{C}_0^G$ denotes the initial global meta-prototype bank. All prototype banks, prototype gradients $\mathbf{g}_{\mathcal{C}}^p$, and increments Δ_t^p have shape $N_C \times R$. All OT assignments are treated as fixed targets during backpropagation. Projected representations and prototype rows are normalized before computing similarities, as in the main text.

Algorithm 4 HAMPL Pre-Training

Input: Private datasets $\{\mathcal{D}^p\}_{p\in\mathcal{P}}$; local models $\{\theta^p\}_{p\in\mathcal{P}}$; prototype count N_C; pre-training epoch budget, and hyperparameters of unbalanced OT and consensus-guide OT.
Output: Pretrained local encoders $\{f^p\}_{p\in\mathcal{P}}$.

Initialize each θ^p and the global bank $\mathcal{C}_0^G$; set $\mathcal{C}_0^G \leftarrow \mathcal{N}(\mathcal{C}_0^G)$
for each pre-training epoch t **do**
  Cross training stage
    Server sets $\mathcal{C} \leftarrow \mathcal{C}_t^G$ and broadcasts $\mathcal{C}$ to all participants.
    for each aligned mini-batch $\mathcal{B}_\mathrm{a}$ with common ordered IDs in $\mathcal{I}_\mathrm{a}$ **do**
      for all $p \in \mathcal{P}$ **in parallel do**
        Compute $\mathbf{Z}_\mathrm{a}^p$ and $\mathbf{S}_\mathrm{a}^p$ from $\mathcal{B}_\mathrm{a}^p$ and $\mathcal{C}$
        $\mathbf{Q}_\mathrm{a}^p \leftarrow$ Solve the cross unbalanced OT problem (Eq. (2)) using Algorithm 1
        Upload $\mathbf{Q}_\mathrm{a}^p$ to the server
      end for
      Server: $\mathbf{Q}_\mathrm{a}^G \leftarrow$ Entropy-guided consensus fusion (Eq. (3))
        Broadcast $\mathbf{Q}_\mathrm{a}^G$ to all participants
      for all $p \in \mathcal{P}$ **in parallel do**
        $\widetilde{\mathbf{Q}}_\mathrm{a}^p \leftarrow$ Solve consensus-guided OT problem (Eq. (5)) using Algorithm 2
        Compute $\mathcal{L}_\mathrm{HASSL}^p$ using Eq. (6)
        Backpropagate $\mathcal{L}_\mathrm{HASSL}^p$ to update θ^p locally and upload $\mathbf{g}_\mathcal{C}^p = \nabla_\mathcal{C}\mathcal{L}_\mathrm{HASSL}^p$
      end for
      Server averages $\{\mathbf{g}_\mathcal{C}^p\}_{p\in\mathcal{P}}$, updates $\mathcal{C}$ by SGD, and sets $\mathcal{C} \leftarrow \mathcal{N}(\mathcal{C})$
        Broadcast the updated $\mathcal{C}$ for the next aligned mini-batch
    end for
    $\mathcal{C}_{t+1/2}^G \leftarrow \mathcal{C}$.
  Local training stage
    Server broadcasts $\mathcal{C}_{t+1/2}^G$ to all participants
    for all $p \in \mathcal{P}$ **in parallel do**
      $\mathcal{C}_l^p \leftarrow \mathcal{C}_{t+1/2}^G$
      for each local mini-batch $\mathcal{B}_l^p \subseteq \mathcal{D}^p$ **do**
        Generate $\mathcal{V}_{l,1}^p$ and $\mathcal{V}_{l,2}^p$ from $\mathcal{B}_l^p$ using $\mathcal{T}$
        Compute $\mathbf{Z}_{l,j}^p$ and $\mathbf{S}_{l,j}^p$, $j \in \{1,2\}$
        $\mathbf{Q}_{l,1}^p, \mathbf{Q}_{l,2}^p \leftarrow$ Solve local unbalanced OT (Eq. (7)) using Algorithm 3
        Compute $\mathcal{L}_\mathrm{LSSL}^p$ using Eq. (8)
        Update θ^p and $\mathcal{C}_l^p$ locally by backpropagating $\mathcal{L}_\mathrm{LSSL}^p$.
      end for
      $\mathcal{C}_{t+1}^p \leftarrow \mathcal{C}_l^p$ and upload $\Delta_t^p = \mathcal{C}_{t+1}^p - \mathcal{C}_{t+1/2}^G$ to the server.
    end for
    Server aggregates: $\mathcal{C}_{t+1}^G \leftarrow \mathcal{N}\Big(\mathcal{C}_{t+1/2}^G + \frac{\eta_\mathrm{meta}}{P}\sum_{p=1}^P \Delta_t^p\Big)$.
end for
return $\{f^p\}_{p\in\mathcal{P}}$.

D Additional Experiments

D.1 Effect of the Aligned-Sample Ratio

To assess robustness to limited cross-participant sample overlap, we reduce the aligned-sample ratio from the default 40% to 20% and 10% for pre-training. Figure 4 reports downstream performance after fine-tuning with 200, 400, 600, 800, and 1000 aligned and labeled samples. We compare HAMPL with DACLSC, FedHSSL-M, ISSVFL-M, and the local-only baseline FedLocal under the same labeled-sample budgets.

HAMPL maintains strong performance even when only 10% of the training samples are aligned. It achieves the best results across all plotted label budgets on NUS-WIDE, BHI, and ModelNet, and in most settings on Avazu. Its performance remains close to that obtained with 20% aligned samples, particularly on BHI and ModelNet. Notably, on these two datasets, HAMPL with 10% aligned samples consistently outperforms the competing collaborative methods even when they use 20% aligned samples. Its clear advantage over FedLocal further demonstrates that HAMPL can extract useful collaborative knowledge from limited sample overlap.

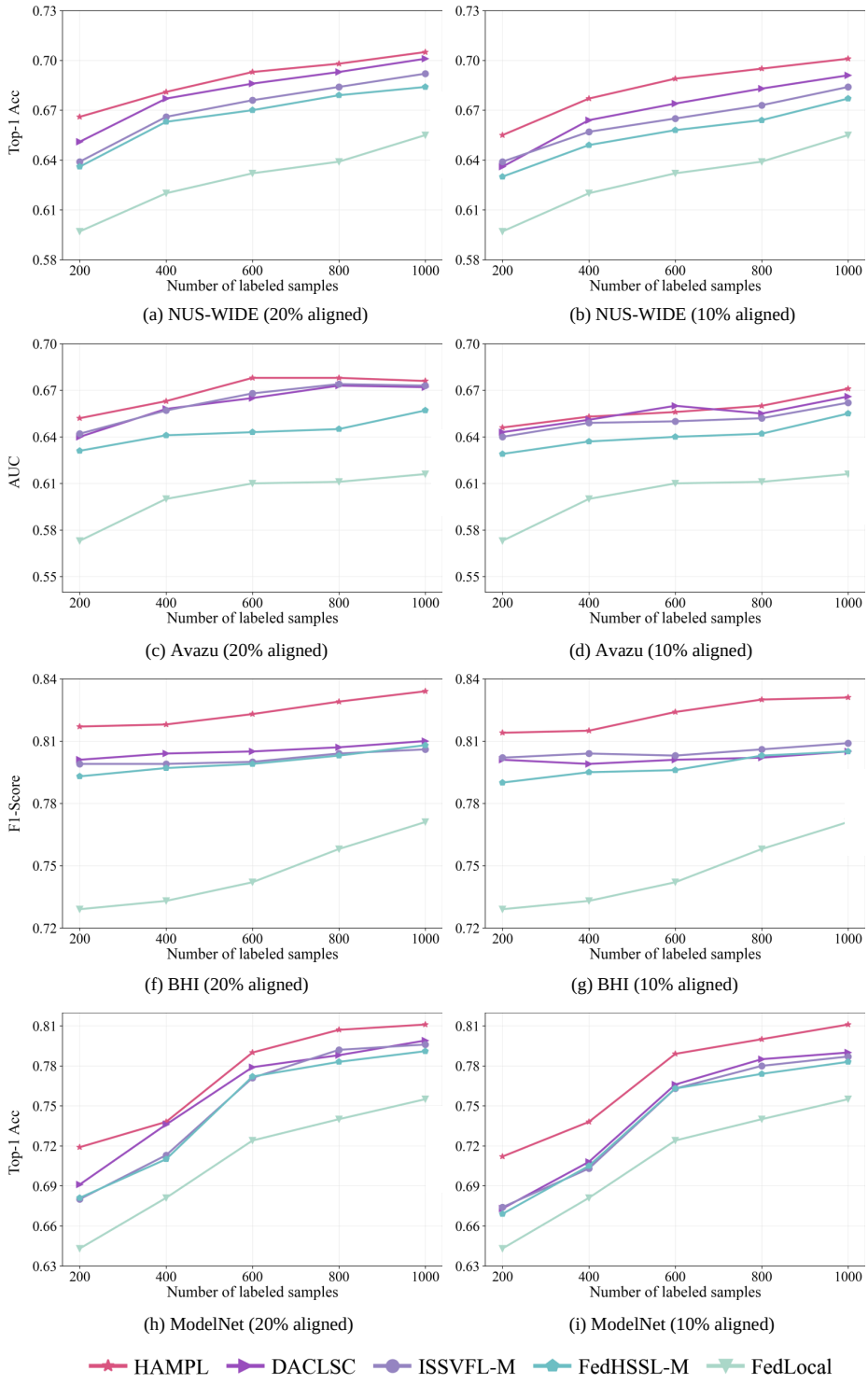


Figure 4: Performance with 20% (left) and 10% (right) aligned samples.

These comparisons suggest that the evaluated SSVFL baselines require more aligned samples to attain comparable downstream performance, whereas HAMPL retains stronger generalization under sparse alignment. This behavior is consistent with its prototype-based design: the shared meta-prototypes connect knowledge learned from the limited aligned samples with information acquired

from all available private samples during local training. Meanwhile, heterogeneity-aware matching supports collaboration without rigidly aligning participant representations, preserving their complementary semantics. Together, the results highlight HAMPL's robustness to low aligned-sample ratios and its applicability to VFL scenarios where cross-participant sample overlap is limited.

D.2 Experimental Privacy Analysis under Label Inference Attack

Following Chen et al. (2026); He et al. (2025), we specify an MC protocol to test the label-predictive component of the hypothesis that a single HAMPL encoder acquires less additional label- or attribute-predictive information from collaboration than its counterparts in representation-sharing SSVFL frameworks.

D.2.1 Threat Model and Attack Procedure

Objective, capability, and knowledge. Let $a \in \mathcal{P} \setminus \{1\}$ denote an honest-but-curious passive participant. It follows collaborative training, retains its local encoder f^a, and attempts to predict labels owned by participant 1 using only its own feature subset at inference time. It has a small auxiliary set $\mathcal{D}^a_{\text{aux}} = \{(\mathbf{x}^a_i, y_i)\}_{i=1}^{n_{\text{aux}}}$, together with its ordinary unlabeled local samples. For a conservative evaluation, the adversary may additionally know the active participant's architecture, input dimension, and task classes. This is an explicitly stronger prior than the undisclosed-architecture setting discussed in Appendix B.1; it does not give access to the active participant's weights, private features, or test labels.

Model completion. The attack adapts the advanced MC approach of Fu et al. (2022). First, the adversary saves a copy of its encoder immediately after pre-training. Second, it attaches a randomly initialized classification head h^a_ψ and fits the resulting model using $\mathcal{D}^a_{\text{aux}}$. Joint encoder/head fine-tuning should be included. Third, the adversary predicts labels for held-out test entities using only their features available to participant a:

$$\mathcal{A}^a_{\text{HAMPL}} = h^a_\psi \circ f^a, \qquad \widehat{y}_i = \arg\max_c \big[\mathcal{A}^a_{\text{HAMPL}}(\mathbf{x}^a_{i,\text{test}})\big]_c, \quad i \in \mathcal{I}_{\text{test}}. \tag{D.1}$$

Here $\mathcal{I}_{\text{test}}$ indexes held-out test entities, and $\mathbf{x}^a_{i,\text{test}}$ contains only the features of test entity i held by participant a. Attack training is performed offline on the saved copy and does not modify the jointly trained VFL model.

D.2.2 ISO Perturbation for Baseline Comparisons

The isotropic Gaussian (ISO) perturbation serves as a privacy protection mechanism in existing SSVFL frameworks (Li et al., 2022; He et al., 2025; Chen et al., 2026). For a communicated matrix $\mathbf{V} \in \mathbb{R}^{n \times d_V}$ with row vectors $\mathbf{v}_j$, define

$$\text{ISO}_\beta(\mathbf{V}) = \mathbf{V} + \mathbf{E}, \qquad E_{jk} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_V^2), \qquad \sigma_V = \frac{\beta \max_j \|\mathbf{v}_j\|_2}{\sqrt{d_V}}. \tag{D.2}$$

The parameter $\beta \geq 0$ controls the perturbation strength, with $\beta = 0$ recovering the unperturbed protocol. In the perturbed baseline, ISO noise is applied locally to the active participant's outgoing messages before they reach the server.

Unlike FedHSSL, ISSVFL, and DACLSC (He et al., 2025; Chen et al., 2025; 2026), HAMPL does not exchange encoder representations or encoder parameters during pre-training. The active participant instead uploads $\mathbf{Q}^1_{\text{a}}$, $\mathbf{g}^1_{\mathcal{C}}$, and Δ^1_t. Based on the privacy-boundary discussion in Appendix D.3, for the setting considered here, we therefore choose to retain the base HAMPL protocol without additional perturbations. We compare unperturbed HAMPL against DACLSC without ISO (DACLSC w/o ISO) and DACLSC with ISO (DACLSC w/ ISO) under the same evaluation protocol, assessing both main-task performance and label-inference attack performance.

D.2.3 Evaluation Settings and Results

To retain comparability with the reference privacy study (He et al., 2025; Chen et al., 2026), the dedicated evaluation uses 80 auxiliary labeled samples for each attack model and uses 200 aligned and labeled samples for the main task. The aligned-sample ratio is set to 20% for the pre-training in this

evaluation; test entities are excluded from pre-training and auxiliary-label selection. The auxiliary set is held fixed across methods. The four benchmarks, feature partitions, and main-task metrics otherwise follow Appendix A.2. For a fair comparison, the evaluation uses the homogeneous-model setting. To assess pre-training collaboration without subsequent label-dependent encoder updates, we initialize all attack models using encoder checkpoints saved immediately after pre-training, before downstream supervised fine-tuning. An attack on the encoder after downstream supervised fine-tuning is a separate evaluation, because that encoder has already received label-dependent updates.

The local-only attack model $\mathcal{A}^a_{\mathrm{FedLocal}}$ uses the same attack procedure and an encoder pretrained by FedLocal. This baseline measures predictability already obtainable from the adversary's own data.

We measure label recovery accuracy (LRA) as

$$\mathrm{LRA}(\mathcal{A}) = \frac{1}{|\mathcal{I}_{\mathrm{test}}|} \sum_{i \in \mathcal{I}_{\mathrm{test}}} \mathbb{I}[\widehat{y}_i = y_i]\,, \tag{D.3}$$

where $\mathbb{I}[\cdot]$ is the indicator function. We quantify the relative change in attack accuracy over local pre-training as

$$\Delta^{\mathrm{rel}}_{\mathrm{LRA}}(\mathcal{A}) = \frac{\mathrm{LRA}(\mathcal{A}) - \mathrm{LRA}(\mathcal{A}^a_{\mathrm{FedLocal}})}{\mathrm{LRA}(\mathcal{A}^a_{\mathrm{FedLocal}})} \times 100\%. \tag{D.4}$$

The relative change $\Delta^{\mathrm{rel}}_{\mathrm{LRA}}$ measures the additional label predictability acquired through collaboration under the same feature partition and auxiliary labels.

Table 8 reports the main-task performance alongside the effectiveness of the MC attack. For DACLSC, we denote the pre-training noise setting by β_{pre}: DACLSC w/o ISO uses $\beta_{\mathrm{pre}} = 0$, whereas DACLSC w/ ISO uses $\beta_{\mathrm{pre}} > 0$. HAMPL remains unperturbed in this comparison.

In the order NUS-WIDE, Avazu, BHI, and ModelNet, $\Delta^{\mathrm{rel}}_{\mathrm{LRA}}$ is -2.5%, $+0.6\%$, -3.4%, and -2.5% for HAMPL, versus $+9.2\%$, $+3.6\%$, $+1.9\%$, and $+2.5\%$ for DACLSC w/o ISO. Thus, under the same local feature partitions and auxiliary labels, collaboration adds consistently less label predictability to a single HAMPL encoder than to the unperturbed representation-sharing DACLSC encoder across all four datasets.

Table 8: Main-task performance and MC attack effectiveness.

Dataset	$\mathcal{A}_{\mathrm{FedLocal}}$	w/o ISO		w/ ISO			HAMPL w/o ISO	
		Main	$\mathcal{A}_{\mathrm{DACLSC}}$	Main	$\mathcal{A}_{\mathrm{DACLSC}}$	β_{pre}	Main	$\mathcal{A}_{\mathrm{HAMPL}}$
NUS-WIDE	0.402	0.651	0.439	0.620	0.415	0.4	0.660	0.392
Avazu	0.525	0.646	0.544	0.645	0.521	0.1	0.651	0.528
BHI	0.670	0.801	0.683	0.796	0.651	0.1	0.817	0.647
ModelNet	0.487	0.691	0.499	0.683	0.490	0.1	0.719	0.475

D.3 Privacy Boundaries for Pre-Training

Information retained by assignments. The scaling form in Appendix C.1 shows why removing explicit representation messages does not prove representation secrecy. Write $m = m_{\mathrm{a}}$, $\mathbf{Z} = \mathbf{Z}^p_{\mathrm{a}}$, and $\mathbf{Q} = \mathbf{Q}^p_{\mathrm{a}}$ for a positive, unperturbed assignment matrix containing the current-batch rows. We consider a server that knows both the corresponding prototype bank $\mathcal{C}$ and the assignment temperature ε_{a}. Since the returned assignments differ from $\mathrm{diag}(\mathbf{u})\exp(\mathbf{Z}\mathcal{C}^\top/\varepsilon_{\mathrm{a}})\,\mathrm{diag}(\mathbf{v})$ only by a scalar normalization, their elementwise logarithm satisfies

$$\log \mathbf{Q} = \frac{\mathbf{Z}\mathcal{C}^\top}{\varepsilon_{\mathrm{a}}} + \boldsymbol{\alpha}\mathbf{1}_{N_C}^\top + \mathbf{1}_m\boldsymbol{\beta}^\top + \kappa\mathbf{1}_m\mathbf{1}_{N_C}^\top, \tag{D.5}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ contain log scaling factors and κ accounts for total-mass normalization. Define $\mathbf{H}_n = \mathbf{I}_n - \mathbf{1}_n\mathbf{1}_n^\top/n$. Double centering eliminates the unknown scaling terms:

$$\varepsilon_{\mathrm{a}}\mathbf{H}_m(\log \mathbf{Q})\mathbf{H}_{N_C} = \mathbf{H}_m\mathbf{Z}\mathcal{C}^\top\mathbf{H}_{N_C}. \tag{D.6}$$

If $\mathbf{A} = \mathcal{C}^\top \mathbf{H}_{N_C}$ has full row rank R, the server can use the known ε_{a} and the pseudoinverse of $\mathbf{A}$ to recover the *centered representations* as

$$\mathbf{H}_m \mathbf{Z} = \varepsilon_{\mathrm{a}} \mathbf{H}_m (\log \mathbf{Q}) \mathbf{H}_{N_C} \mathbf{A}^\dagger, \tag{D.7}$$

where † denotes the Moore–Penrose pseudoinverse. Crucially, the recovered quantity is the *centered representation of each sample*, namely its representation minus the *mean representation of the current batch*. Specifically, writing $\mathbf{z}_i^\top$ for row i of $\mathbf{Z}$,

$$\big[\mathbf{H}_m \mathbf{Z}\big]_{i,:} = \big(\mathbf{z}_i - \overline{\mathbf{z}}\big)^\top, \qquad \overline{\mathbf{z}} = \frac{1}{m} \sum_{j=1}^{m} \mathbf{z}_j.$$

Equation (D.7) does not by itself recover the batch mean $\overline{\mathbf{z}}$ or the uncentered representations $\mathbf{Z}$.

The factorization holds even for a finite number of scaling iterations; it also holds on the current-batch rows when a queue supplies additional rows to the solver. In that case, the recovered representations are centered over the current batch, not over the combined batch and queue. Row normalization alone adds another rowwise log offset and does not remove this channel. Numerical precision and conditioning affect recovery accuracy, and rank-deficient banks reveal only a projected component of the centered representations. This argument concerns centered latent representations, not unique recovery of the original features. The server cannot recover the original features without additional knowledge of the encoder, its parameters, and its architecture.

Information retained by prototype updates. Prototype gradients provide aggregate constraints on private representations. With fixed assignment targets, denote the derivative of Eq. (6) with respect to the normalized prototype coordinates, before applying the Jacobian of prototype normalization, by $\mathbf{G}^p$:

$$\mathbf{G}^p = \frac{1}{m_{\mathrm{a}} \tau} \left(\mathbf{\Pi}_{\mathrm{a}}^p - \widetilde{\mathbf{Q}}_{\mathrm{a}}^p \right)^\top \mathbf{Z}_{\mathrm{a}}^p, \qquad (\mathbf{\Pi}_{\mathrm{a}}^p)_{b,k} = \pi_k(\mathbf{z}_{\mathrm{a},b}^p; \mathcal{C}). \tag{D.8}$$

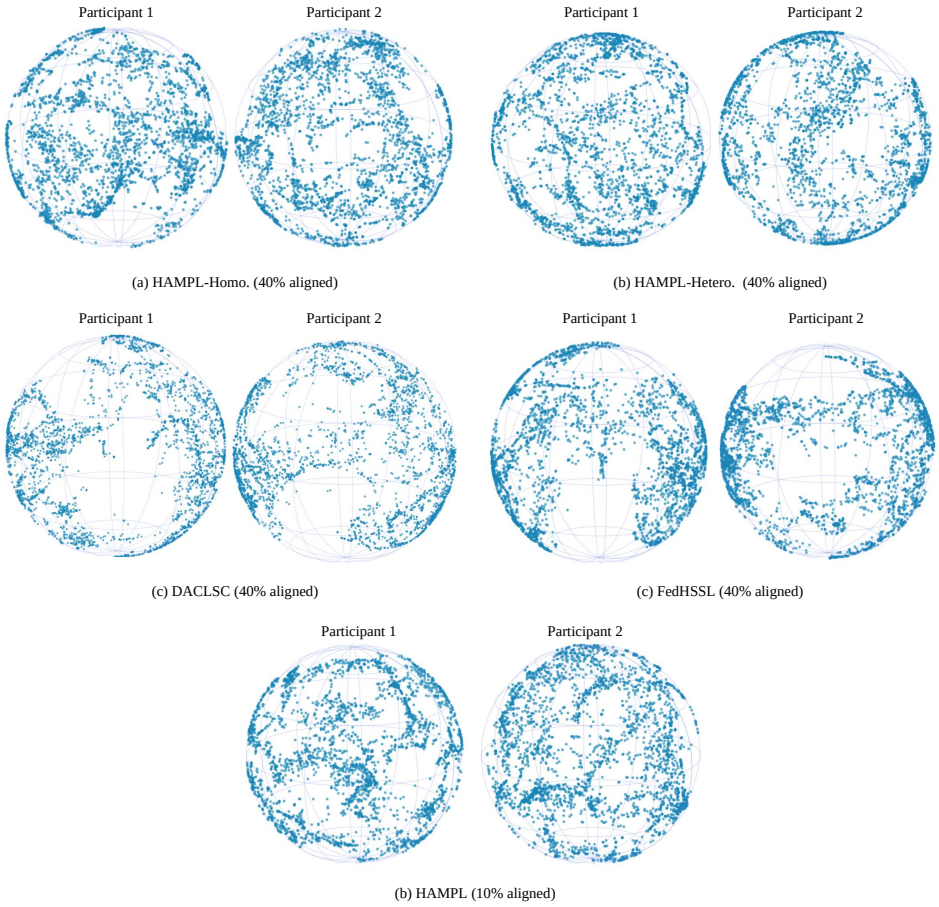
Writing $\mathbf{B}^p = (\mathbf{\Pi}_{\mathrm{a}}^p - \widetilde{\mathbf{Q}}_{\mathrm{a}}^p)^\top$, Eq. (D.8) gives $m_{\mathrm{a}} \tau \mathbf{G}^p = \mathbf{B}^p \mathbf{Z}_{\mathrm{a}}^p$. Thus, up to the known scalar factor, observing $\mathbf{G}^p$ is equivalent to observing *weighted mixtures of sample representations within the batch*. When prototype normalization is differentiated through, the uploaded gradient is the corresponding Jacobian-transformed mixture. Disentangling $\mathbf{B}^p$ and the individual representations $\mathbf{Z}_{\mathrm{a}}^p$ requires additional prerequisites, such as information sufficient to identify the mixing coefficients and ensure that the resulting inverse problem is identifiable. HAMPL does not directly transmit the local predictions $\mathbf{\Pi}_{\mathrm{a}}^p$ or the refined assignments $\widetilde{\mathbf{Q}}_{\mathrm{a}}^p$ that determine $\mathbf{B}^p$.

Even if the individual representations were recovered, instantiating the encoder-informed surrogate E in the attack pathway of Eq. (B.1) would additionally require knowledge of the local encoder, including the architecture and parameter values of the modules used to initialize E. HAMPL does not upload these encoder parameters or require disclosure of encoder architectures during pre-training. Accordingly, considering the prototype-gradient channel alone, the server can only obtains batch-level weighted mixtures of private representations. Prototype increments likewise summarize the effects of repeated local updates.

D.4 Visualization of Pretrained Representations

To examine representation collapse qualitatively, we visualize the outputs $f^p(\mathbf{x}_i^p)$ of the pretrained local encoders on the ModelNet test set, before downstream supervised fine-tuning. Each point represents a test sample encoded by the corresponding participant. Figure 5 shows the spherical distributions for Participants 1 and 2, comparing HAMPL with homogeneous and heterogeneous encoders at a 40% aligned-sample ratio, DACLSC and FedHSSL at the same ratio, and HAMPL with only 10% aligned samples. We focus on the extent to which representations occupy the sphere: larger contiguous empty regions, together with stronger concentration of points in restricted areas, suggest more severe representation collapse and reduced representation diversity.

With 40% aligned samples, DACLSC and FedHSSL exhibit pronounced concentrations of representations, leaving large contiguous regions of the sphere unoccupied. In contrast, HAMPL's representations cover the sphere more broadly under both homogeneous and heterogeneous encoder settings, with less extensive contiguous empty regions. This difference is consistent with reduced collapse



(a) HAMPL-Homo. (40% aligned)

(b) HAMPL-Hetero. (40% aligned)

(c) DACLSC (40% aligned)

(c) FedHSSL (40% aligned)

(b) HAMPL (10% aligned)

Figure 5: Spherical visualization of pretrained encoder outputs. Larger contiguous empty regions and stronger concentration of the displayed representations qualitatively indicate more severe representation collapse.

even when participants use different encoder architectures. Moreover, HAMPL retains broad spherical coverage with only 10% aligned samples, complementing its strong downstream performance under sparse alignment in Appendix D.1. These qualitative observations support the motivation for prototype-based collaboration: enabling knowledge sharing without rigidly aligning complementary participant representations, thereby preserving their diversity and accommodating cross-participant semantic discrepancies.