Appendix

A Additional Discussions

A.1 Application scenario

As shown in Figure 1, Alice, the service receiver (bank) squared in red dashed line, is the organization to be assisted. Before learning, it broadcasts identification (ID) to locate and align vertically distributed data held by other organizations. At the beginning of the Learning Stage, the bank deterministically initializes the values of $F^0(x)$ to be the unbiased estimate of y_1, namely $F^0(x) = \mathbb{E}_N(y_1^0)$. For the regression task, $F^0(x)$ is a single scalar. For classification task, $F^0(x)$ is a point in the K-dimensional simplex P_K.

During the first assistance round in the Learning Stage, the bank computes pseudo-residual r_1^1 and broadcasts it to other organizations (e.g., hospital, mall, and insurance company). Then, all the organizations, including the bank, will fit a new local model with 1) their local data, 2) the pseudo-residual r_1^1, and 3) their local regression loss function ℓ_m (e.g., ℓ_2-loss) to fit the pseudo-residual. We note that organizations have complete autonomy on model fitting. In particular, they can choose their own learning algorithms and models by considering their resources (e.g., computation power). Next, the bank will aggregate all the predictions from each organization's local models by optimizing a weight vector $w_{1:M}$ referred to as gradient assistance weights. As previously discussed in Equation (5), we approximate the oracle gradient (operated on centralized data, in hindsight) with a weighted average of those predictions from organizations. We then numerically search for the learning rate η. This process can be iterated multiple times until the learning rate is low or the validation loss is satisfactory.

During the Prediction Stage, organizations will predict with trained models at every assistance round and transmit their predictions to the bank. Similar to the Learning Stage, *the synchronization of each organization is unnecessary*. The bank computes the final prediction with gradient assistance weights, learning rates, and received predictions.

A.2 Future work on adversarial learning

In this work, we have considered settings where the participating organizations are cooperative, or some of them receive noisy inputs (pseudo-residuals) or create noisy outputs (fitted pseudo-residuals). More experimental studies are included in Section D.4 of the Appendix. Nevertheless, we have not considered adversarial scenarios during Assisted Learning. In adversarial scenarios, one or more organizations may be malicious or subject to an adversarial attack, e.g., in training data, test data, or models at training or prediction stages. Compared with conventional adversarial learning settings that often involve one learner, the proposed decentralized learning framework potentially offers more avenues for adversarial behavior. Inspired by the existing literature on adversarial learning, we briefly comment on the following adversarial GAL problems that may deserve future study.

• *Adversarial examples* [41–45] refer a type of prediction-stage attack that the intended input data (e.g., an image) is slightly perturbed to cause an already-trained model to make a false prediction. In GAL, if a participant, Bob, has a large assistance weight (at one or more rounds), it will contribute non-trivially to the final prediction of Alice. In this case, Bob's adversarially perturbed future input will also affect Alice's prediction accuracy, especially when the weights are large. To enhance robustness against such an attack, the participants may use a minimax-based robust local training [46].

• *Backdoor attacks* [47–56] aim to disrupt the prediction performance on specific sub-populations or target labels (e.g., from a stop sign to a speed sign) without degrading accuracy on most of the input data regimes. Backdoor attacks often assume the adversary can inject crafted perturbations into the training data (also known as a "backdoor trigger"), and the learning task is classification. While this attack may occur to any participating organization at the training stage, it is unclear how to devise backdoor triggers for GAL participants that only solve regression problems at each round.

• *Data poisoning attacks* [57–62] aim to deteriorate the overall prediction performances of Alice. Compared with backdoor attacks, a poisoning attack is untargeted and often occurs in the training stage. We conjecture that this type of attack is relatively easier to address in a practical GAL system

since the gradient assistance weights may assign small weights for those participants that are not trained well, in contrast with the conventional setting where there is only one learner and one dataset.

• *Model-stealing attacks* [63–68] (also known as model extractions) refer to the unwanted reconstruction of a trained machine learning model through information exchanges. In GAL, Alice may receive assistance from Bob to steal Bob's local model using queries and responses in the prediction stage. Likewise, Bob may steal Alice's local model by participating in the GAL system initialized by Alice. Apart from single-model stealing, Alice may also perform multi-model stealing, aiming to learn Bob's capability to generate predictive models for different pseudo-labels across rounds. If successful, Alice can imitate Bob's functionality and assist other learners as if she were Bob.

B Theoretical Analysis

To develop a convergence analysis of the GAL algorithm, we use the following notations. We still let $\mathcal{F}_m$ (for each $m = 1, \ldots, M$) denote a set of real-valued functions defined on organization m's data x_m. For notational simplicity, for each $f_m \in \mathcal{F}_m$, we also treat it as a function of the (artificially) extended variable $x = [x_1, \ldots, x_M]$. So, we may write a function in the form of $f_1 + f_2$, which basically means $[x_1, x_2] \mapsto f_1(x_1) + f_2(x_2)$. Let $\mathcal{L}$ denote the overarching loss function to minimize (for the agent to assist), and P_M the probability simplex.

As a summary, we abstract the core steps in Algorithm 1 below. For each organization m at assistance round t, it optimizes each local model by solving

$$(\alpha_t, f_m^t) = \underset{\alpha \in [-a_t, a_t], f_m \in \mathcal{F}_m}{\operatorname{argmin}} \mathcal{L}(F^{t-1} + \alpha f_m). \tag{6}$$

Alice then gathers predictions f_m^t, $m = 1, \ldots M$ from all the organizations and optimizes the gradient assistance weights and learning rate by solving

$$(\hat{w}^t, \hat{\eta}^t) = \underset{w \in P_M, \eta \in [-a_t, a_t]}{\operatorname{argmin}} \mathcal{L}\bigg(F^{t-1} + \eta \sum_{m=1}^{M} w_m f_m^t\bigg). \tag{7}$$

At round $t = 0$, we initialize with any $F^0 \in \mathcal{F}_1$. At each round t, each organization first runs a greedy boosting step to obtain (α_t, f_m^t). The f_m^t will be sent to us (the organization to assist). Then, we run another greedy step to optimize the assistance weights $\hat{w}^t$ and learning rate $\hat{\eta}^t$, with fixed f_m^t, $m = 1, \ldots, M$. The weighted function will be added to F^{t-1} to generate the latest F^t.

For each m, we let

$$\text{span}(\mathcal{F}_m) = \bigg\{ \sum_{j=1}^{K_m} \mu_j f_j : \mu_j \in \mathbb{R}, f_j \in \mathcal{F}_m, K_m \in \mathbb{N}^+ \bigg\},$$

which is the function space formed from linear combinations of elements in $\mathcal{F}_m$. Let

$$\text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M) = \bigg\{ \sum_{m=1}^{M} w_m f_m : w_m \in \mathbb{R}, f_m \in \text{span}(\mathcal{F}_m) \bigg\}$$

denote the linear span of the union of $\mathcal{F}_1, \ldots, \mathcal{F}_M$. An equivalent way to write it is $\text{span}(\cup_{m=1}^M \mathcal{F}_m)$.

We will show the following convergence result. With a suitable choice of step parameters a_t and regularity conditions of the loss $\mathcal{L}$, the abstract form of GAL can produce F^t that asymptotically attains the minimum loss within the function class $\text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)$. We make the following technical assumptions.

(A1) The loss (functional) $f \mapsto \mathcal{L}(f)$ is convex and differentiable on $\mathcal{F}$, with gradient $\nabla \mathcal{L}$. Also, for all $f \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)$ and $g \in \cup_{m=1}^M \mathcal{F}_m$, the function $u \mapsto \mathcal{L}(f + ug)$ has a second order derivative $\partial^2 \mathcal{L}(f + ug)/\partial u^2$, and it is upper bounded by a fixed constant C.

(A2) The ranges of learning rates $\{a_t\}_{t=1,2,\ldots}$ satisfy $\sum_{t=1}^{\infty} a_t = \infty$, $\sum_{t=1}^{\infty} a_t^2 < \infty$.

Theorem 1: Under Assumptions (A1) and (A2), the GAL algorithm satisfies $\mathcal{L}(F^t) \to \inf_{f \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)} \mathcal{L}(f)$ as $t \to \infty$, with a convergence rate at the order of $O(\sum_{\tau=1}^{t} (a_{1:\tau}/a_{1:t}) a_\tau^2)$.

Remarks on Theorem 1: The result says that with suitable control of the learning rates, the greedy procedure in Algorithm 1 can converge to the oracle one could obtain within span$(\mathcal{F}_1, \ldots, \mathcal{F}_M)$. Suppose that an organization, say the one indexed by $m = 1$, does not collaborate with others. Likewise, we have the convergence for that particular organization, $\lim_{t\to\infty} \mathcal{L}(F^t) = \inf_{f^* \in \text{span}(\mathcal{F}_1)} \mathcal{L}(f^*)$. It can be seen that the GAL will produce a significant gain for this organization as long as

$$\inf_{f^* \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)} \mathcal{L}(f^*) < \inf_{f^* \in \text{span}(\mathcal{F}_1)} \mathcal{L}(f^*). \tag{8}$$

It is conceivable that (8) is easy to meet in many practical scenarios since each $\mathcal{F}_m$ is operated on a particular modality of data that belongs to organization m. On the other hand, a skeptical reader may wonder how the GAL solution compares with a function learned from the pulled data. It is possible that the global minimum of $\mathcal{L}$ (over functions that operate on the pulled data) does not belong to span$(\mathcal{F}_1, \ldots, \mathcal{F}_M)$. If that is the case, the best we can do is to find f that attains the limit $\inf_{f \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)} \mathcal{L}(f)$. This is a limitation due to the constraint that organizations cannot share data and the additive structure of span$(\mathcal{F}_1, \ldots, \mathcal{F}_M)$. Fortunately, in various real-data experiments we performed, the GAL often performs close to the centralized learning within only a few assistance rounds.

In the technical result, we could allow the approximate minimization of (6) and (7), meaning that the loss of the produced solution is δ_t-away from the optimal loss. In that case, it can be verified that $\sum_{t=1}^{\infty} \delta_t < \infty$ is sufficient to derive the same asymptotic result in Theorem 1.

The proof of Theorem 1 uses the same technique as was used in [69]. The technical result here is nontrivial, because f_m^t $(m = 1, \ldots, M)$ in each round t are not jointly minimized with $\hat{w}^t$ and $\hat{\eta}^t$ in (7), and thus their linear combination may not be the most greedy solution of minimizing $\mathcal{L}(F^{t-1} + f)$ within $f \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)$.

Proof of Theorem 1:

Let $f^* \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)$ be an arbitrary fixed function. It is introduced for technical convenience and can be treated as the function that (approximately) attains the infimum of $L(f)$.

For every $f \in \text{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)$, we define the following norm with respect to the basis functions,

$$\|f\|_1 = \inf\left\{ \|\mu\|_1 : \sum_{m=1}^{M} \sum_{j=1}^{K_m} \mu_{m,j} f_{m,j} : \mu_{m,j} \in \mathbb{R}, f_{m,j} \in \mathcal{F}_m, K_m \in \mathbb{N}^+ \right\}$$

where $\|\mu\|_1$ denotes the abstract sum of its entries, namely $\sum_{m=1,\ldots,M, j=1,\ldots,K_m} |\mu_{m,j}|$.

For each t, let $S_t \subset \cup_{m=1}^{M} \mathcal{F}_m$ denote the finite set of functions such that
1) $f_m^\tau \in S_t$ for all $0 < \tau < t$, and
2) $f^* = \sum_{g \in S_t} \mu_{f^*}^g g$ $(\mu^g \in \mathbb{R})$, with $\|\mu_{f^*}\|_1 \leq \|f^*\|_1 + \varepsilon$.
Note that S_t exists due to the definition of $\|\cdot\|_1$ and the construction of each f_m^τ. Suppose that F^{t-1} admits the representation $F^{t-1} = \sum_{g \in S_t} \mu_{F^{t-1}}^g g$.

From (7), we have

$$\mathcal{L}(F^t) \leq \mathcal{L}\big(F^{t-1} + \hat{\eta}_t f_m^t\big), \quad \forall m = 1, \ldots, M. \tag{9}$$

Meanwhile, it follows from (6) that for each m, and each $g \in S_t \cap \mathcal{F}_m$,

$$\mathcal{L}\big(F^{t-1} + \hat{\eta}_t f_m^t\big) \leq \mathcal{L}\big(F^{t-1} + a_t s^g g\big). \tag{10}$$

where $s^g \triangleq \text{sign}(\mu_{f^*}^g - \mu_{F^{t-1}}^g)$. Combining (9) and (10), we obtain

$$\mathcal{L}(F^t) \leq \mathcal{L}\big(F^{t-1} + a_t s^g g\big), \quad \forall g \in S_t. \tag{11}$$

Applying Taylor expansion to $f \mapsto \mathcal{L}(f)$ at $f = F^{t-1}$, and invoking (11) and Assumption (A1), we have

$$\mathcal{L}(F^t) - \mathcal{L}(F^{t-1}) \leq \mathcal{L}(F^{t-1} + a_t s^g g) - \mathcal{L}(F^{t-1}) \leq a_t s^g \nabla \mathcal{L}(F^{t-1})^\intercal g + \frac{C}{2} a_t^2 \tag{12}$$

for all sufficiently small $a_t > 0$. Let $\|\mu_{f^*} - \mu_{F^{t-1}}\|_1 \triangleq \sum_{g \in S_t} |\mu^g_{f^*} - \mu^g_{F^{t-1}}|$. Multiplying both sides by $|\mu^g_{f^*} - \mu^g_{F^{t-1}}|$, and add up all the $g \in S_t$, we have

$$\|\mu_{f^*} - \mu_{F^{t-1}}\|_1 \cdot \{\mathcal{L}(F^t) - \mathcal{L}(F^{t-1})\} \leq a_t \nabla \mathcal{L}(F^{t-1})^{\mathrm{T}}(f^* - F^{t-1}) + \|\mu_{f^*} - \mu_{F^{t-1}}\|_1 \cdot \frac{C}{2} a_t^2$$
$$\leq a_t\{\mathcal{L}(f^*) - \mathcal{L}(F^{t-1})\} + \|\mu_{f^*} - \mu_{F^{t-1}}\|_1 \cdot \frac{C}{2} a_t^2 \tag{13}$$

where the last inequality is due to the convexity of $\mathcal{L}$. If $\|\mu_{f^*} - \mu_{F^{t-1}}\|_1 = 0$, F^{t-1} already converges to f^*. Otherwise, we rearrange (13) to obtain

$$\mathcal{L}(F^t) - \mathcal{L}(f^*) \leq \left(1 - \frac{a_t}{\|\mu_{f^*} - \mu_{F^{t-1}}\|_1}\right)\{\mathcal{L}(F^{t-1}) - \mathcal{L}(f^*)\} + \frac{C}{2} a_t^2 \tag{14}$$
$$\leq \left(1 - \frac{a_t}{\|\mu_{f^*}\|_1 + 1 + \sum_{\tau=0}^{t-1} a_\tau}\right)\{\mathcal{L}(F^{t-1}) - \mathcal{L}(f^*)\} + \frac{C}{2} a_t^2, \tag{15}$$

where the last inequality is due to the triangle inequality, the way F^{t-1} is constructed, and the fact that ε can be arbitrarily chosen. Here, we defined $a_0 \triangleq 0$. Let $a_{1:t} = \sum_{\tau=1}^t a_\tau$ for each $t \geq 1$. Applying (15) and the Lemma 4.2 in [69], we have

$$\max(0, \mathcal{L}(F^t) - \mathcal{L}(f^*)) \leq \frac{\|\mu_{f^*}\|_1 + 1}{\|\mu_{f^*}\|_1 + a_{1:t}} + \frac{C}{2} \sum_{\tau=1}^t \frac{\|\mu_{f^*}\|_1 + a_{1:\tau}}{\|\mu_{f^*}\|_1 + a_{1:t}} a_\tau^2. \tag{16}$$

Since f^* is arbitrarily chosen, it can be seen from Inequality (16) and Assumption (A2) that $\lim_{t\to\infty} \mathcal{L}(F^t) = \inf_{f^* \in \mathrm{span}(\mathcal{F}_1, \ldots, \mathcal{F}_M)} \mathcal{L}(f^*)$, and the rate of convergence is at the order of $O(\sum_{\tau=1}^t (a_{1:\tau}/a_{1:t}) a_\tau^2)$ as $t \to \infty$.

C Experimental Setup

C.1 Dataset

In Table 7, we illustrate the statistics of datasets used in our experiments. In Figure 6, we show how MNIST and CIFAR10 images are split into 2, 4, and 8 image patches. The left upper image patch (labeled as $[1]$) of the MNIST image is less informative, which demonstrates that an organization with little informative data can leverage other organizations' local data and models. The central image patches (labeled $[2, 3, 6, 7]$) of MNIST and CIFAR10 images are more informative than others, which leads to larger corresponding gradient assistance weights.

Table 7: Detailed statistics used in each data experiment. The variables d and K respectively denote the number of features (or the shape of the image) and the length of the prediction vector (or equivalently, the number of classes in the classification task).

Dataset	N_{train}	N_{test}	d	K	M
Diabetes	353	89	10	1	{2, 4, 8}
BostonHousing	404	102	13	1	{2, 4, 8}
Blob	80	20	10	10	{2, 4, 8}
Iris	120	30	4	3	{2, 4}
Wine	142	36	13	3	{2, 4, 8}
BreastCancer	455	114	30	2	{2, 4, 8}
QSAR	844	211	41	2	{2, 4, 8}
MNIST	60000	10000	(1, 28, 28)	10	{2, 4, 8}
CIFAR10	50000	10000	(3, 32, 32)	10	{2, 4, 8}
ModelNet40	3163	800	(12, 3, 32, 32,3 2)	40	{12}
ShapeNet55	35764	5159	(12, 3, 32, 32, 32)	55	{12}
MIMICL	34387	6057	22	1	{4}
MIMICM	17902	3236	22	1	{4}

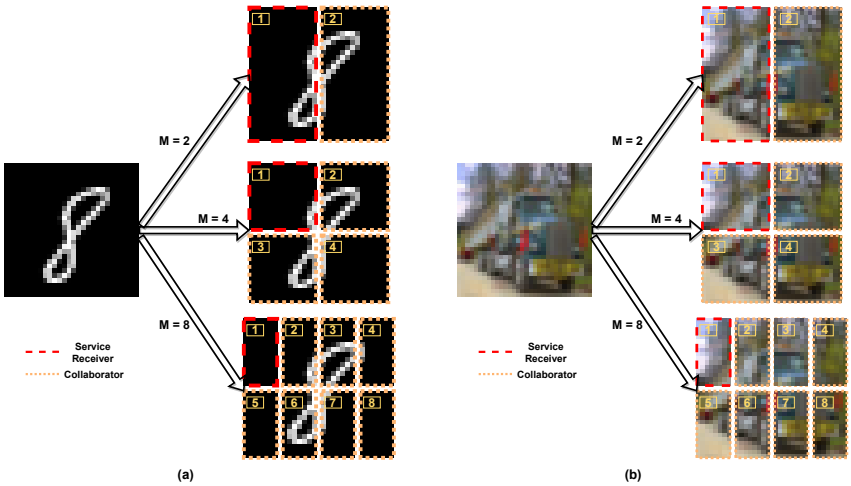


Figure 6: An illustration of (a) MNIST and (b) CIFAR10 data split into 2, 4, and 8 image patches. The left upper image patch (labeled $[1]$) of MNIST images is less informative in general. In contrast, the central image patches (labeled $[2, 3, 6, 7]$) of MNIST and CIFAR10 images are more informative.

C.2 Model and hyperparameters

Table 8 summarizes the deep neural network architecture used for the MNIST, CIFAR10, ModelNet40, and ShapeNet55 datasets. Table 9 shows the hyperparameters used in our experiments.

Table 8: The model architecture of Convolutional Neural Networks (CNN) used in our experiments of the MNIST, CIFAR10, ModelNet40, and ShapeNet55 datasets. The n_c, H, W represent the shape of images, namely the number of image channels, height, and width, respectively. K is the number of classes in the classification task. The BatchNorm and ReLU layers follow Conv2d(input channel size, output channel size, kernel size, stride, padding) layers. The MaxPool2d(output channel size, kernel size) layer reduces the height and width by half.

Image $x \in \mathbb{R}^{n_c \times H \times W}$
Conv2d(n_c, 64, 3, 1, 1)
MaxPool2d(64, 2)
Conv2d(64, 128, 3, 1, 1)
MaxPool2d(128, 2)
Conv2d(128, 256, 3, 1, 1)
MaxPool2d(256, 2)
Conv2d(256, 512, 3, 1, 1)
MaxPool2d(512, 2)
Global Average Pooling
Linear(512, K)

Table 9: Hyperparameters used in our experiments for training local models, gradient assisted learning rates, and gradient assistance weights.

Dataset		UCI	MNIST	CIFAR10	ModelNet40	ShapeNet55	MIMICL	MIMICM
Architecture		Linear	CNN				LSTM	
Local	Epoch	100	10					
	Batch size	1024	512		64		8	
	Optimizer	SGD					Adam	
	Learning rate	1.0E-01					1.0E-03	
	Weight decay	5.0E-04						
Gradient assisted learning rate	Epoch	10						
	Batch size	Full						
	Optimizer	L-BFGS						
	Learning rate	1						
Gradient assistance weights	Epoch	100						
	Batch size	1024						
	Optimizer	Adam						
	Learning rate	1.0E-01						
	Weight decay	5.0E-04						
Assistance rounds		10						

D Experimental Results

D.1 Model Autonomy

In Tables 10 and 11, we demonstrate the results of our experiments related to model autonomy for $M = 2$ and 4 respectively. Our method significantly outperforms the baselines 'Alone' and 'AL.' The results also demonstrate that with GAL, an organization with little informative data and free choice of its local model (model autonomy) can leverage other organizations' local data and models and even achieve near-oracle performance.

Table 10: Results of the UCI datasets ($M = 2$) with Linear, GB, SVM and GB-SVM models. The Diabetes and Boston Housing (regression) are evaluated with Mean Absolute Deviation (MAD), and the rest (classification) are evaluated with Accuracy.

Dataset	Model	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)
Late	Linear	120.2(0.1)	3.6(0.1)	100.0(0.0)	100.0(0.0)	100.0(0.0)	99.3(0.4)	81.4(0.4)
Joint	Linear	43.4(0.3)	3.0(0.0)	100.0(0.0)	99.2(1.4)	100.0(0.0)	99.1(0.4)	84.0(0.2)
Alone	Linear	46.8(3.5)	4.1(0.7)	100.0(0.0)	92.5(6.0)	93.1(6.4)	98.9(0.6)	79.9(1.0)
AL	Linear	63.7(1.5)	3.9(0.6)	98.8(2.2)	95.0(2.9)	95.1(2.3)	97.6(0.7)	80.6(1.6)
GAL	Linear	43.2(0.8)	2.9(0.1)	100.0(0.0)	99.2(1.4)	96.5(2.3)	98.9(0.4)	83.8(0.4)
GAL	GB	49.1(2.7)	3.0(0.3)	97.5(2.5)	95.8(1.4)	98.6(1.4)	95.6(1.1)	85.1(1.0)
GAL	SVM	42.6(1.9)	2.5(0.1)	100.0(0.0)	96.7(0.0)	95.1(1.2)	99.6(0.4)	87.3(1.0)
GAL	GB-SVM	50.9(2.9)	3.1(0.5)	96.3(6.5)	96.7(0.0)	93.7(4.6)	94.7(0.6)	82.7(0.4)

Table 11: Results of the UCI datasets ($M = 4$) with Linear, GB, SVM and GB-SVM models. The Diabetes and Boston Housing (regression) are evaluated with Mean Absolute Deviation (MAD), and the rest (classification) are evaluated with Accuracy.

Dataset	Model	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)
Late	Linear	129.5(0.1)	4.7(0.0)	100.0(0.0)	100.0(0.0)	100.0(0.0)	98.5(0.7)	79.7(1.2)
Joint	Linear	43.4(0.3)	3.0(0.0)	100.0(0.0)	99.2(1.4)	100.0(0.0)	98.9(0.4)	84.0(0.2)
Alone	Linear	56.6(8.2)	4.8(0.6)	80.0(6.1)	79.2(13.0)	84.7(1.4)	97.1(1.0)	73.0(1.0)
AL	Linear	58.3(2.4)	5.2(0.3)	100.0(0.0)	88.3(8.3)	92.4(2.3)	98.9(1.1)	76.8(2.5)
GAL	Linear	43.3(1.1)	3.0(0.1)	100.0(0.0)	100.0(0.0)	97.9(2.3)	99.1(0.6)	83.3(0.5)
GAL	GB	56.8(3.9)	3.2(0.4)	98.8(2.2)	96.7(0.0)	94.4(2.0)	95.2(0.8)	84.8(1.1)
GAL	SVM	44.7(2.6)	2.7(0.1)	100.0(0.0)	96.7(0.0)	96.5(2.3)	99.8(0.4)	86.6(1.1)
GAL	GB-SVM	50.0(2.9)	3.3(0.4)	92.5(4.3)	97.5(1.4)	88.9(3.9)	95.0(1.8)	84.2(1.5)

D.2 Deep Model Sharing

In Tables 12 and 13, we demonstrate the results of our experiments related to deep model sharing for $M = 2$ and 4 respectively. The results show that sharing the feature extractor across multiple assistance rounds can still outperform the 'Alone' case. Thus, DMS can provide a trade-off between predictive performance and computation space.

Table 12: Results of the MNIST and CIFAR10 ($M = 2$) datasets with CNN model. The MNIST and CIFAR10 are evaluated with Accuracy. GAL_{DMS} represents the results with Deep Model Sharing.

Dataset	MNIST(↑)	CIFAR10(↑)
Interm	99.4(0.0)	81.1(0.3)
Late	99.0(0.0)	81.0(0.2)
Joint	99.4(0.0)	80.1(0.2)
Alone	96.7(0.2)	72.7(0.2)
AL	96.4(0.1)	74.7(0.3)
GAL	98.5(0.2)	**78.7(0.4)**
GAL_{DMS}	**98.7(0.1)**	74.3(0.5)

Table 13: Results of the MNIST and CIFAR10 ($M = 4$) datasets with CNN model. The MNIST and CIFAR10 are evaluated with Accuracy. GAL_{DMS} represents the results with Deep Model Sharing.

Dataset	MNIST(↑)	CIFAR10(↑)
Interm	99.1(0.0)	79.8(0.1)
Late	98.4(0.1)	77.5(0.2)
Joint	99.4(0.0)	80.1(0.2)
Alone	81.2(0.1)	60.0(0.4)
AL	82.5(0.1)	64.8(0.3)
GAL	96.6(0.2)	**77.3(0.2)**
GAL_{DMS}	**96.7(0.3)**	71.3(0.2)

D.3 Comparison with AL

In Table 14, we demonstrate the comparison of computation and communication complexity between GAL and AL. We compare the computation and communication complexity between AL and GAL under the constraint of the same communication cost as demonstrated. Because AL sequentially trains each organization while GAL allows organizations to train locally in parallel, the computation time and communication round of AL is $M\times$ those of GAL. The GAL with Deep Model Sharing (GAL_{DMS}) saves $T\times$ computation space by sharing the feature extractor of deep models. In summary, GAL generalizes the problem scope, reduces the computation and communication complexity, and achieves significantly better results.

Table 14: Comparison of computation and communication complexity between GAL and AL. M and T represent the number of organizations and assistance rounds, respectively.

Complexity	Computation Time	Computation Space	Communication Round	Communication Cost
AL	$M\times$	$T\times$	$M\times$	$1\times$
GAL	$1\times$	$T\times$	$1\times$	$1\times$
GAL_{DMS}	$1\times$	$1\times$	$1\times$	$1\times$

D.4 Ablation studies

D.4.1 Privacy enhancement

Our learning framework does not require organizations to share local data, models, and objective functions. One potential limitation of our approach is that assisting organizations may infer Alice's information based on shared the pseudo-residuals. Therefore, we suggest further enhancing privacy by adopting the framework of Differential Privacy (DP) [39] or Interval Privacy (IP) [40]. We use the Laplace mechanism with $\alpha = 1$ for DP and set the number of intervals of IP to be 1. We add a moderate amount of noise to the pseudo-residuals in hindsight. In Tables 15 and 16, we demonstrate that privacy-enhanced GAL can still outperform the 'Alone' case.

Table 15: Ablation study on the privacy enhancement ($M = 2$). GAL_{DP} and GAL_{IP} represent privacy-enhanced by DP and IP, respectively.

Dataset	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)	MNIST(↑)	CIFAR10(↑)
Alone	46.8(3.5)	4.1(0.7)	100.0(0.0)	92.5(6.0)	93.1(6.4)	98.9(0.6)	79.9(1.0)	96.7(0.2)	72.7(0.2)
GAL_{DP}	52.1(1.1)	3.5(0.2)	66.3(5.4)	83.3(4.1)	88.2(9.3)	93.9(1.2)	81.9(1.4)	95.7(0.4)	59.0(0.9)
GAL_{IP}	52.1(0.9)	3.4(0.1)	100.0(0.0)	92.5(2.8)	99.3(1.2)	96.3(0.7)	86.3(1.3)	97.2(0.4)	71.7(0.4)

Table 16: Ablation study on the privacy enhancement ($M = 4$). GAL_{DP} and GAL_{IP} represent privacy-enhanced by DP and IP, respectively.

Dataset	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)	MNIST(↑)	CIFAR10(↑)
Alone	56.6(8.2)	4.8(0.6)	80.0(6.1)	79.2(13.0)	84.7(1.4)	97.1(1.0)	73.0(1.0)	81.2(0.1)	60.0(0.4)
GAL_{DP}	52.0(0.8)	3.4(0.1)	47.5(14.8)	85.8(6.4)	91.0(6.3)	95.2(1.0)	81.9(3.2)	94.7(0.4)	57.6(0.1)
GAL_{IP}	52.0(0.2)	3.4(0.1)	97.5(4.3)	89.2(3.6)	97.9(1.2)	96.1(0.8)	86.0(1.9)	95.6(0.4)	71.1(0.3)

D.4.2 Noisy training with gradient assistance weights

To optimize gradient assistance weights, we use the Adam optimizer and enforce the parameters to sum to one by using the softmax function. The cost to optimize the gradient assistance weights w and gradient assisted learning rate η is often negligible compared with the cost to fit the pseudo-residuals since the number of parameters involved in $w \in \mathbb{R}^M$ and $\eta \in \mathbb{R}^1$ is small. In Tables 17 and 18, we demonstrate the results of our ablation studies of gradient assistance weights by adding noise to the predicted pseudo-residuals (namely the outputs) from half of the organizations. In Tables 19-21, we demonstrate the results of our ablation studies of gradient assistance weights when half of the organizations have no predictive power for the target, i.e., data features sampled from $\mathcal{N}(0, 1)$.

Table 17: Ablation study ($M = 2$) of gradient assistance weights by adding noises to the predicted outputs from half of the organizations.

Noise	Weight	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)	MNIST(↑)	CIFAR10(↑)
$\sigma = 1$	✗	50.1(1.9)	4.4(0.2)	62.5(2.5)	80.8(6.4)	86.8(2.3)	89.9(3.1)	73.2(1.3)	79.7(0.3)	48.8(0.3)
	✓	**47.8(2.4)**	**3.5(0.5)**	**97.5(4.3)**	**95.0(3.7)**	**96.5(3.0)**	**98.7(1.0)**	**80.2(0.5)**	**96.8(0.1)**	**71.4(0.1)**
$\sigma = 5$	✗	58.8(1.3)	6.1(0.2)	25.0(9.4)	52.5(10.9)	63.9(3.4)	73.2(1.0)	63.3(0.5)	34.8(0.5)	22.0(0.2)
	✓	**46.5(3.1)**	**4.1(0.8)**	**83.8(7.4)**	**90.0(4.1)**	**93.1(4.2)**	**97.6(1.1)**	**78.3(1.0)**	**96.3(0.1)**	**65.9(0.3)**

Table 18: Ablation study ($M = 4$) of gradient assistance weights by adding noises to the predicted outputs from half of the organizations.

Noise	Weight	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)	MNIST(↑)	CIFAR10(↑)
$\sigma = 1$	✗	46.7(1.0)	4.1(0.1)	46.3(6.5)	80.0(5.3)	85.4(3.0)	91.2(1.4)	72.6(2.2)	78.7(0.1)	47.6(0.3)
	✓	**45.0(2.8)**	**3.7(0.5)**	**90.0(5.0)**	**95.8(4.3)**	**94.4(3.4)**	**97.8(1.0)**	**79.1(1.1)**	**94.1(0.1)**	**65.4(0.3)**
$\sigma = 5$	✗	59.4(1.1)	5.7(0.4)	13.8(4.1)	54.2(7.6)	61.1(7.1)	75.9(2.9)	64.1(1.8)	38.4(0.3)	22.6(0.5)
	✓	**49.6(3.7)**	**4.1(0.7)**	**66.3(9.6)**	**93.3(2.4)**	**93.7(3.6)**	**97.8(0.4)**	**76.7(1.6)**	**93.0(0.2)**	**59.9(0.6)**

Table 19: Ablation study ($M = 2$) of gradient assistance weights when half of the organizations have no predictive power for the target, i.e. data features sampled from $\mathcal{N}(0, 1)$.

Weight	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)
✗	50.7(4.6)	4.3(0.7)	97.5(4.3)	92.5(6.0)	91.7(6.2)	82.9(5.3)	76.9(3.0)
✓	**46.8(3.6)**	**4.1(0.7)**	**97.5(2.5)**	**92.5(6.0)**	**92.4(7.4)**	**97.6(1.7)**	**80.2(1.7)**

Table 20: Ablation study ($M = 4$) of gradient assistance weights when half of the organizations have no predictive power for the target, i.e. data features sampled from $\mathcal{N}(0, 1)$.

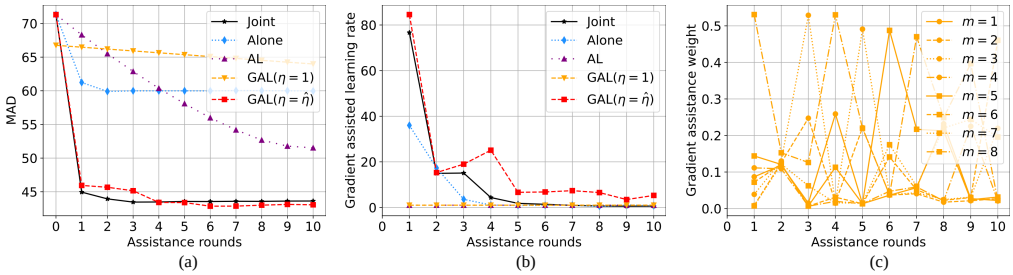
Weight	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Iris(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)
✗	50.9(4.7)	4.5(0.6)	83.8(5.4)	91.7(5.5)	93.1(3.1)	87.1(5.8)	77.0(0.5)
✓	**49.6(3.7)**	**4.2(0.7)**	**95.0(6.1)**	**93.3(6.7)**	**94.4(5.2)**	**98.2(0.9)**	**78.3(0.8)**

Table 21: Ablation study (maximal M) of gradient assistance weights when half of the organizations have no predictive power for the target, i.e. data features sampled from $\mathcal{N}(0, 1)$.

Weight	Diabetes(↓)	BostonHousing(↓)	Blob(↑)	Wine(↑)	BreastCancer(↑)	QSAR(↑)
✗	53.3(6.8)	5.3(0.2)	81.3(6.5)	86.8(5.0)	88.2(2.0)	75.9(1.0)
✓	**50.2(4.3)**	**4.8(0.6)**	**93.8(5.4)**	**88.9(6.2)**	**96.5(1.1)**	**77.9(1.5)**

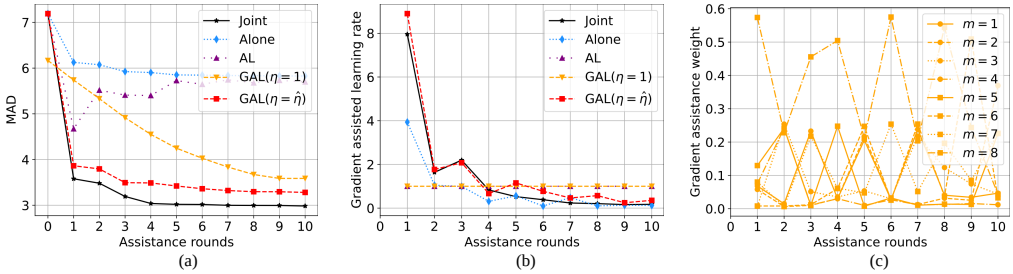
D.5 Additional Results

In Figure 7-19, we illustrate results of all datasets (maximal M).



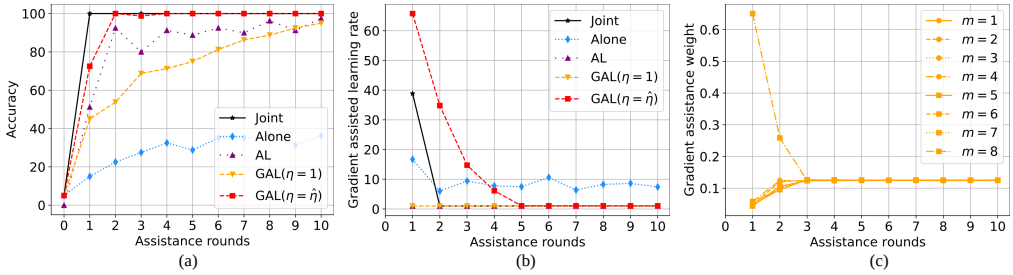
(a) (b) (c)

Figure 7: Results of the Diabetes ($M = 8$) dataset.



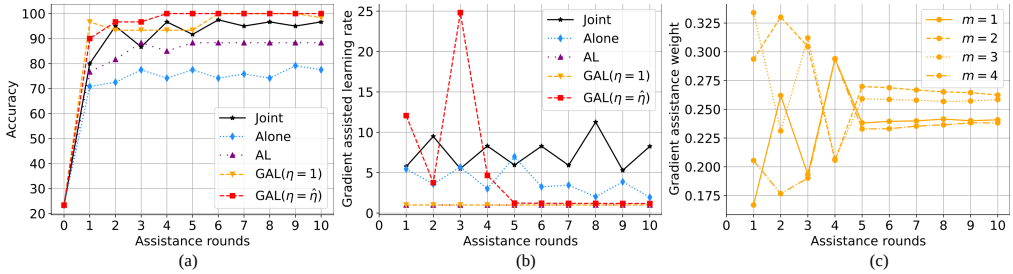
(a) (b) (c)

Figure 8: Results of the BostonHousing ($M = 8$) dataset.



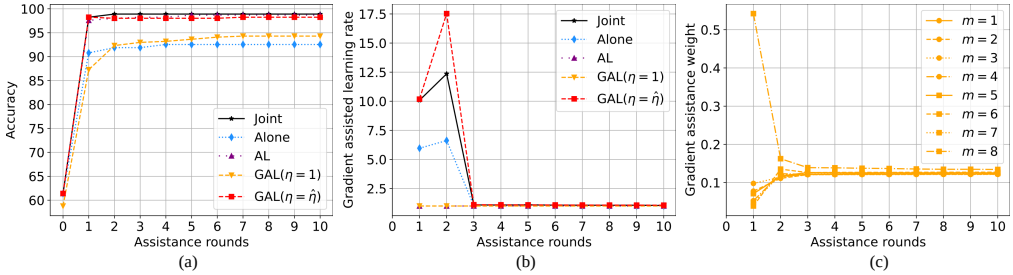
(a) (b) (c)

Figure 9: Results of the Blob ($M = 8$) dataset.



(a) (b) (c)

Figure 10: Results of the Iris ($M = 4$) dataset.



(a) (b) (c)

Figure 11: Results of the BreastCancer ($M = 8$) dataset.

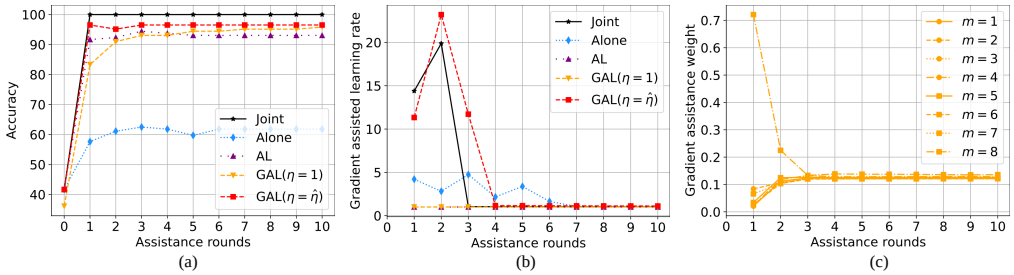


Figure 12: Results of the Wine ($M = 8$) dataset.

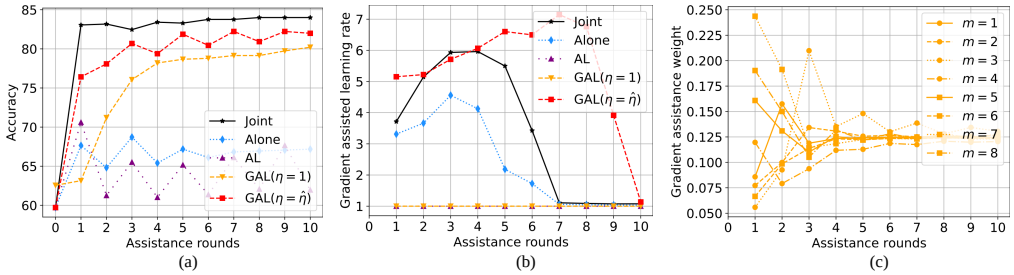


Figure 13: Results of the QSAR ($M = 8$) dataset.

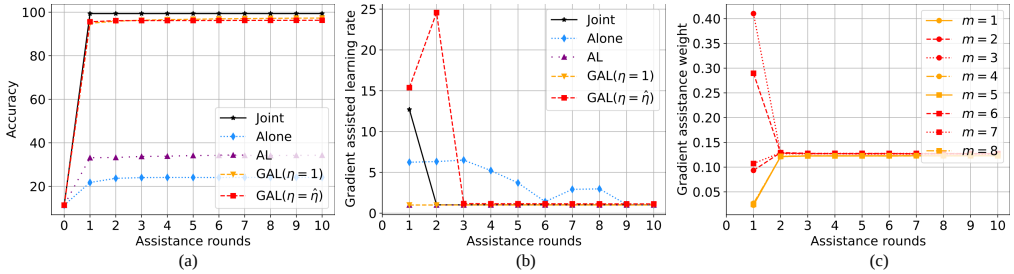


Figure 14: Results of the MNIST ($M = 8$) dataset.

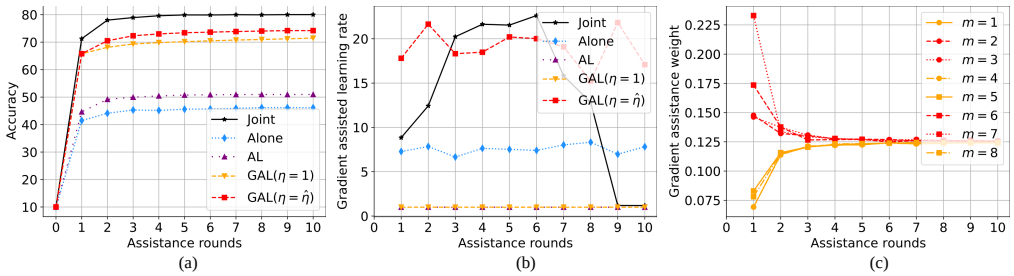


Figure 15: Results of the CIFAR10 ($M = 8$) dataset.

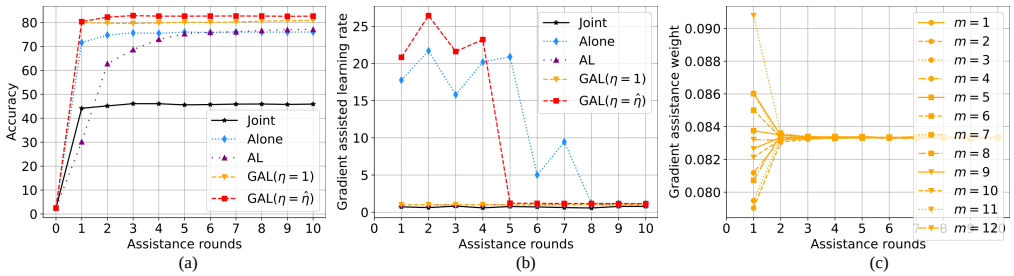
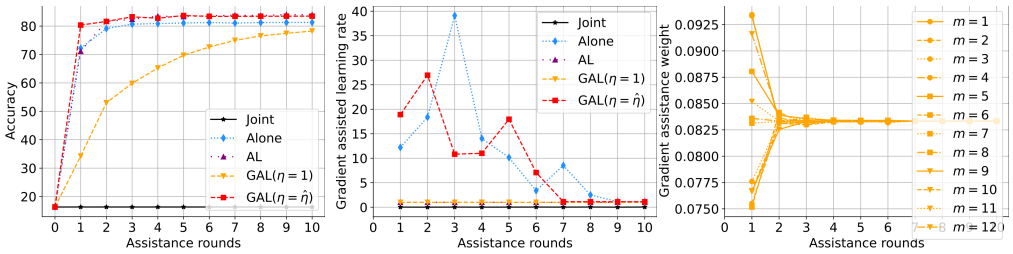
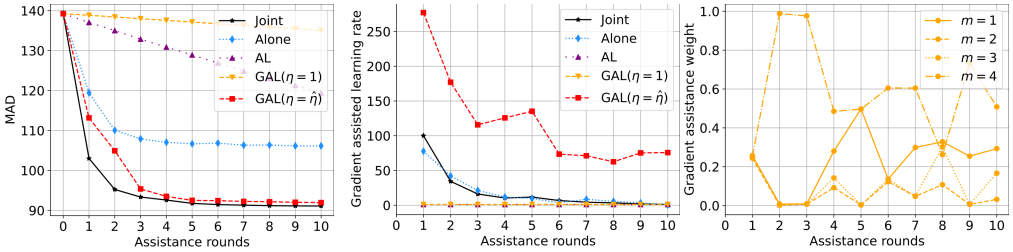


Figure 16: Results of the ModelNet40 ($M = 12$) dataset.



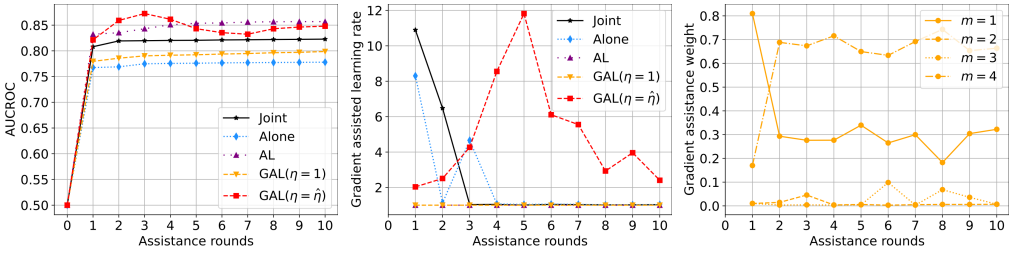
(a) (b) (c)

Figure 17: Results of the ShapeNet55 ($M = 12$) dataset.



(a) (b) (c)

Figure 18: Results of the MIMICL ($M = 4$) dataset.



(a) (b) (c)

Figure 19: Results of the MIMICM ($M = 4$) dataset.