A ADMM: Problem decomposition details

Different from SGD, ADMM aims to decompose the global optimization problem of Eq. 2 into multiple sub-problems, as independent optimization problems. To achieve this, we follow the *sharing ADMM* paradigm to rewrite Eq. 2 to the following Eq. 26, by introducing auxiliary variables $z = \{z_j\}_{j=1}^N$ where $z_j \in \mathbb{R}^{d_c}$:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{N}\sum_{j=1}^{N}\ell\left(z_j; y_j\right) + \beta\sum_{i=1}^{M}\mathcal{R}_i(\theta_i), \\ &\text{subject to} \quad \sum_{i=1}^{M} h_{i,j} - z_j = 0,\ \forall j \in [N],\ h_{i,j} = f_i(\theta_i; T_{i,p_i(j)}). \end{aligned} \tag{26}$$

We then add a quadratic term to the Lagrangian of Eq. 26, which results in Eq. 27 and is known as *augmented Lagrangian*. Here, $\{\lambda_j\}_{j=1}^N$ are dual variables and $\lambda_j \in \mathbb{R}^{d_c}$.

$$\begin{aligned} \min \mathcal{L}\left(\theta_i, z_j, \lambda_j\right) = &\frac{1}{N}\sum_{j=1}^{N}\ell\left(z_j; y_j\right) + \beta\sum_{i=1}^{M}\mathcal{R}_i(\theta_i) \\ &+ \frac{1}{N}\sum_{j=1}^{N}\lambda_j^\top\left(\sum_{i=1}^{M} f_i(\theta_i; T_{i,p_i(j)}) - z_j\right) \\ &+ \frac{\rho}{2N}\sum_{j=1}^{N}\left\|\sum_{i=1}^{M} f_i(\theta_i; T_{i,p_i(j)}) - z_j\right\|^2. \end{aligned} \tag{27}$$

To simplify notation, we define residual variables $\{s_{i,j}\}_{i\in[M],j\in[N]}$ for each table T_i as follows, where $s_{i,j} \in \mathbb{R}^{d_c}$:

$$s_{i,j} = \sum_{k=1,k\neq i}^{M} f_i(\theta_k; T_{k,p_k(j)}) - z_j. \tag{28}$$

Given the optimization problem of Eq. 27 as follows, we next detail how to leverage *Alternating Direction Method of Multipliers* (ADMM) to decompose this problem to sub-problems.

$$\min \mathcal{L}\left(\theta_i, z_j, \lambda_j\right) = \frac{1}{N}\sum_{j=1}^{N}\ell\left(z_j; y_j\right) + \beta\sum_{i=1}^{M}\mathcal{R}_i(\theta_i) + \frac{1}{N}\sum_{j=1}^{N}\lambda_j^\top\left(\sum_{i=1}^{M} f_i(\theta_i; T_{i,p_i(j)}) - z_j\right) + \frac{\rho}{2N}\sum_{j=1}^{N}\left\|\sum_{i=1}^{M} f_i(\theta_i; T_{i,p_i(j)}) - z_j\right\|^2$$

To be simple, we first define residual variables $\{s_{i,j}\}_{i\in[M],j\in[N]}$ for each table T_i as $s_{i,j}^t = \sum_{k=1,k\neq i}^{M} f_i(\theta_k^t; T_{k,p_k(j)}) - z_j^t$ and $s_{i,j}^t \in \mathbb{R}^{d_c}$. We then apply ADMM and obtain following sub-problems (three updates), including client-side θ-update and server-side z-update and λ-update. Here, a^t refers to the value of a in the t-th epoch, while $z_j \in \mathbb{R}^{d_c}$ and $\lambda_j \in \mathbb{R}^{d_c}$.

$$\theta_i^{t+1} := \operatorname*{argmin}_{\theta_i}\left(\beta\mathcal{R}_i(\theta_i) + \frac{1}{N}\sum_{j=1}^{N}\left[\lambda_j^{t\top} f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2}\left\|s_{i,j}^t + f_i(\theta_i; T_{i,p_i(j)})\right\|^2\right]\right) \tag{29}$$

$$z_j^{t+1} := \operatorname*{argmin}_{z_j}\left(\ell\left(z_j; y_j\right) - \lambda_j^{t\top} z_j + \frac{\rho}{2}\left\|\sum_{i=1}^{M} f_i(\theta_i^{t+1}; T_{i,p_i(j)}) - z_j\right\|^2\right) \tag{30}$$

$$\lambda_j^{t+1} := \lambda_j^t + \rho\left(\sum_{i=1}^{M} f_i(\theta_i^{t+1}; T_{i,p_i(j)}) - z_j^{t+1}\right) \tag{31}$$

To simplify the notations and algorithm description, we use z_j^t-update and λ_j^t-update instead of z_j^{t+1}-update and λ_j^{t+1}-update, and move them before θ_i^{t+1}-update as they are executed in the t-th epoch. The resulting equations are as follows and are equivalent to the above equations.

$$z_j^t := \underset{z_j}{\operatorname{argmin}} \left(\ell\left(z_j; y_j\right) - \left(\lambda_j^{t-1}\right)^\top z_j + \frac{\rho}{2} \left\| \sum_{i=1}^{M} f_i(\theta_i^t; T_{i,p_i(j)}) - z_j \right\|^2 \right) \tag{32}$$

$$\lambda_j^t := \lambda_j^{t-1} + \rho \left(\sum_{i=1}^{M} f_i(\theta_i^t; T_{i,p_i(j)}) - z_j^t \right) \tag{33}$$

$$\theta_i^{t+1} := \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{N} \left[\lambda_j^{t\top} f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2} \left\| s_{i,j}^t + f_i(\theta_i; T_{i,p_i(j)}) \right\|^2 \right] \right) \tag{34}$$

B ADMM: Details of computation and communication reduction

B.1 Computation reduction

For our table mapping, recall that the tuple number of X_i is N, the tuple number of T_i is n_i, and $X_{i,j}$ (the j-th tuple of X_i) comes from $T_{i,p_i(j)}$. In reverse, $T_{i,j}$ (the j-th tuple of T_i) can be mapped to multiple tuples in X_i, and we refer to the index set of these tuples as $G_i(j)$. $|G_i(j)|$ denotes the total tuple number in the $G_i(j)$. Using $G_i(j)$, we can aggregate the weights of duplicated tuples, and then rewrite the θ_i-update of Eq. 9 as follows, where $h_{i,j} = f_i(\theta_i; T_{i,j})$.

$$\begin{aligned}
\theta_i^{t+1} &:= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{N} \left[\lambda_j^{t\top} f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2} \left\| s_{i,j}^t + f_i(\theta_i; T_{i,p_i(j)}) \right\|^2 \right] \right) \\
&= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{N} \left[\lambda_j^{t\top} f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2} \left\| s_{i,j}^t \right\|^2 + \rho (s_{i,j}^t)^\top f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2} \left\| f_i(\theta_i; T_{i,p_i(j)}) \right\|^2 \right] \right) \\
&= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{N} \left[\left(\lambda_j^t + \rho s_{i,j}^t \right)^\top f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2} \left\| f_i(\theta_i; T_{i,p_i(j)}) \right\|^2 \right] \right) \\
&= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{N} \left(\lambda_j^t + \rho s_{i,j}^t \right)^\top f_i(\theta_i; T_{i,p_i(j)}) + \frac{\rho}{2N} \sum_{j=1}^{N} \left\| f_i(\theta_i; T_{i,p_i(j)}) \right\|^2 \right) \\
&= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{n_i} \left(\sum_{g \in G_i(j)} (\lambda_g^t + \rho s_{i,g}^t) \right)^\top f_i(\theta_i; T_{i,j}) + \frac{\rho}{2N} \sum_{j=1}^{n_i} |G_i(j)| \left\| f_i(\theta_i; T_{i,j}) \right\|^2 \right) \\
&= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{n_i} \left[\left(\sum_{g \in G_i(j)} (\lambda_g^t + \rho s_{i,g}^t) \right)^\top f_i(\theta_i; T_{i,j}) + \frac{\rho\, |G_i(j)|}{2} \left\| f_i(\theta_i; T_{i,j}) \right\|^2 \right] \right)
\end{aligned} \tag{35}$$

Now, for the θ_i-update of each table T_i, we have reduced the computation complexity from $O(N)$ (i.e., $\sum_{j=1}^{N}$) to $O(n_i)$ (i.e., $\sum_{j=1}^{n_i}$). We can use SGD to solve the θ_i-update problem of Eq. 35.

B.2 Communication reduction

Currently, to perform θ_i-update of Eq. 35 in the client, the server needs to send $\lambda \in \mathbb{R}^{N \times d_c}$, $s_i \in \mathbb{R}^{N \times d_c}$, and $\{G_i(j)\}_{j=1}^{n_i}$ variables to the client that owns T_i. Here, suppose T_i is not horizontally split, the communication complexity is $O(N)$ between the server and each client. To reduce the communication, we can aggregate these variables to be $Y_i \in \mathbb{R}^{n_i \times d_c}$ and $G_i \in \mathbb{R}^{n_i}$ in the sever using the Eq. 36 and Eq. 37 as follows, and then send them to the client owns T_i. Recall that $G_i(j)$ denotes $T_{i,j}$ appears in multiple positions (an index set) in X_i after joins. Therefore, for each $T_{i,j}$, $G_{i,j} = |G_i(j)|$ denotes how many times $T_{i,j}$ appears in X_i after joins, while $Y_{i,j}$ denotes the j-th element of the aggregation of λ and s_i. Thus, the server also does not need to send the table mapping information (i.e., $p_i(j)$) to the clients.

$$Y_{i,j}^t = \sum_{g \in G_i(j)} (\lambda_g^t + \rho s_{i,g}^t) \qquad j = 1 \rightarrow n_i \tag{36}$$

$$G_{i,j} = |G_i(j)| \qquad j = 1 \rightarrow n_i \tag{37}$$

After that, we can rewrite the θ_i-update of Eq. 35 as follows.

$$\theta_i^{t+1} := \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{n_i} \left[\left(\sum_{g \in G_i(j)} (\lambda_g^t + \rho s_{i,g}^t) \right)^\top f_i(\theta_i; T_{i,j}) + \frac{\rho\, |G_i(j)|}{2} \left\| f_i(\theta_i; T_{i,j}) \right\|^2 \right] \right)$$
$$= \underset{\theta_i}{\operatorname{argmin}} \left(\beta \mathcal{R}_i(\theta_i) + \frac{1}{N} \sum_{j=1}^{n_i} \left[(Y_{i,j}^t)^\top f_i(\theta_i; T_{i,j}) + \frac{\rho G_{i,j}}{2} \left\| f_i(\theta_i; T_{i,j}) \right\|^2 \right] \right) \tag{38}$$

After this communication reduction, the communication complexity between the server and the client drops from $O(N)$ to $O(n_i)$.

C Privacy Analysis

We provide privacy analysis for labels (theorem 1) and features (theorem 2) as follows.

To protect the privacy of the labels stored in the server (i.e., the dataset in Definition 1 refers to the label set), we leverage the standard Laplace DP mechanism [16] that adds one-time Laplace noise with standard deviation λ to each coordinate of the labels before training [41], so that the labels satisfy ϵ-label DP (with $\delta = 0$).

Theorem 1. *(Privacy guarantee for labels.) Following Laplace mechanism ([16]), ϵ-label DP can be achieved by injecting additive Laplace noise with per-coordinate standard deviation $\lambda = \frac{2\sqrt{2}}{\epsilon}$.*

To protect the privacy of client's local data (i.e., features), when updating the local model of each client, we clip the per-sample gradient with ℓ_2-norm threshold C and add Gaussian noise sampled from $\mathcal{N}(0, C^2\sigma^2)$. We accumulate privacy budget based on moments accountant in [7] along with training as follows:

Theorem 2. *(Privacy guarantee for features.) There exist constants c_1 and c_2 such that, given Q clients with τ local steps ($\tau = \mathcal{T}\mathcal{T}'S$) for each client, clipping threshold C, noise level σ, and batch subsampling ratio r, for any $\epsilon \leq c_1 r^2 \tau$, DP version of Algorithm 1 satisfies (ϵ, δ)-DP for all $\delta \geq 0$ if we choose $\sigma \geq c_2 \frac{r\sqrt{\tau \log(1/\delta)}}{\epsilon}$.*

The proof extends the Theorem 1 in [7] to the DP guarantee for each client in our Algorithm 1, where each client performs $\tau = \mathcal{T}\mathcal{T}'S$ local DP-SGD steps.