AsySQN: Faster Vertical Federated Learning Algorithms with Better Computation Resource Utilization

Qingsong Zhang
Xidian University & JD Tech
qszhang1995@gmail.com

Bin Gu
MBZUAI & JD Finance America Corporation
gubin3@jd.com

Cheng Deng*
Xidian University
chdeng.xd@gmail.com

Songxiang Gu
JD Tech
songxiang.gu@jd.com

Liefeng Bo
JD Finance America Corporation
boliefeng@jd.com

Jian Pei
Simon Fraser University
jian_pei@sfu.ca

Heng Huang*
JD Finance America Corporation &
University of Pittsburgh
heng.huang@jd.com

ABSTRACT

Vertical federated learning (VFL) is an effective paradigm of training the emerging cross-organizational (*e.g.*, different corporations, companies and organizations) collaborative learning with privacy preserving. Stochastic gradient descent (SGD) methods are the popular choices for training VFL models because of the low per-iteration computation. However, existing SGD-based VFL algorithms are communication-expensive due to a large number of communication rounds. Meanwhile, most existing VFL algorithms use synchronous computation which seriously hamper the computation resource utilization in real-world applications. To address the challenges of communication and computation resource utilization, we propose an asynchronous stochastic quasi-Newton (AsySQN) framework for VFL, under which three algorithms, *i.e.* AsySQN-SGD, -SVRG and -SAGA, are proposed. The proposed AsySQN-type algorithms making descent steps scaled by approximate (without calculating the inverse Hessian matrix explicitly) Hessian information convergence much faster than SGD-based methods in practice and thus can dramatically reduce the number of communication rounds. Moreover, the adopted asynchronous computation can make better use of the computation resource. We theoretically prove the convergence rates of our proposed algorithms for strongly convex problems. Extensive numerical experiments on real-word datasets demonstrate the lower communication costs and better computation resource utilization of our algorithms compared with state-of-the-art VFL algorithms.

*corresponding authors

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467169

CCS CONCEPTS

• **Security and privacy → Privacy protections**; • **Theory of computation → Shared memory algorithms**.

KEYWORDS

Federated learning; quasi-Newton methods; asynchronous parallel

ACM Reference Format:
Qingsong Zhang, Bin Gu, Cheng Deng, Songxiang Gu, Liefeng Bo, Jian Pei, and Heng Huang. 2021. AsySQN: Faster Vertical Federated Learning Algorithms with Better Computation Resource Utilization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467169

1 INTRODUCTION

Federated learning attracts much attention from both academic and industry [7, 15, 18, 24, 28] because it meets the emerging demands of collaboratively-modeling with privacy-preserving. Currently, existing federated learning frameworks can be categorized into two main classes, *i.e.*, horizon federated learning (HFL) and vertical federated learning (VFL). In HFL, samples sharing the same features are distributed over different parties. While, as for VFL, data owned by different parties have the same sample IDs but disjoint subsets of features. Such scenario is common in the industry applications of collaborative learning, such as medical study, financial risk, and targeted marketing [2, 7, 11, 17, 25, 28]. For example, E-commerce companies owning the online shopping information could collaboratively train joint-models with banks and digital finance companies that own other information of the same people such as the average monthly deposit and online consumption, respectively, to achieve a precise customer profiling. In this paper, we focus on VFL.

In the real VFL system, different parties always represent different companies or organizations across different networks, or even in a wireless environment with limited bandwidth [18]. The frequent communications with large per-round communication overhead (PRCO) between different parties are thus much expensive, making the communication expense being one of the main

bottlenecks for efficiently training VFL models. On the other hand, for VFL applications in industry, different parties always own unbalanced computation resources (CR). For example, it is common that large corporations and small companies collaboratively optimize the joint-model, where the former have better CR while the later have the poorer. In this case, synchronous computation has a poor computation resource utilization (CRU). Because corporations owning better CR have to waste its CR to wait for the stragglers for synchronization, leading to another bottleneck for efficiently training VFL models. Thus, it is desired to develop algorithms with lower communication cost (CC) and better CRU to efficiently train VFL models in the real-world applications.

Currently, there are extensive works focusing on VFL. Some works focus on designing different machine learning models for VFL such as linear regression [5], logistic regression [10] and tree model [2]. Some works also aim at developing secure optimization algorithms for training VFL models, such as the SGD-based methods [9, 16, 28], which are popular due to the per-iteration computation efficiency but are communication-expensive due to the large number of communication rounds (NCR). Especially, there have been several works focusing on addressing the CC and CRU challenges of VFL. The quasi-Newton (QN) based framework [23] is designed to reduce the number of communication rounds (NCR) of SGD-based methods, and the bilevel asynchronous VFL framework (VFB^2) [28] and AFVP algorithms [9] are developed to achieve better CRU.

However, in the QN-based framework [23], 1) the gradient differences are transmitted to globally compute the approximate Hessian information, which has expensive PRCO, 2) the synchronous computation is adopted. Thus, QN-based framework still has the large CC and dramatically sacrifices the CRU. Moreover, VFB^2 [28] and AFVP algorithms [9] are communication-expensive owing to large NCR. Thus, it is still challenging to design VFL algorithms with lower CC and better CRU for real-world scenarios.

To address this challenge, we propose an asynchronous stochastic quasi-Newton based framework for VFL, *i.e.*, AsySQN. Specifically, AsySQN-type algorithms significantly improve the practical convergence speed by utilizing approximate Hessian information to obtain a better descent direction, and thus reduce the NCR. Especially, the approximate Hessian information is locally computed and only scalars are necessary to be transmitted, which thus has low PRCO. Meanwhile, the AsySQN framework enables all parties update the model asynchronously, and thus keeps the CR being utilized all the time. Moreover, we consider adopting the vanilla SGD and its variance reduction variants, *i.e.*, SVRG and SAGA, as the stochastic gradient estimator, due to their promising performance in practice. We summarize the contributions of this paper as follows.

- We propose a novel asynchronous stochastic quasi-Newton framework (AsySQN) for VFL, which has the lower CC and better CRU.
- Three AsySQN-type algorithms, including AsySQN-SGD and its variance reduction variants AsySQN-SVRG and -SAGA, are proposed under AsySQN. Moreover, we theoretically prove the convergence rates of these three algorithms for strongly convex problems.

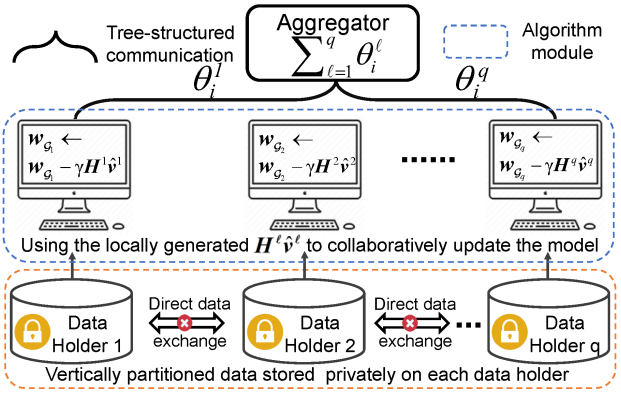


Figure 1: System structure of AsySQN framework.

2 METHODOLOGY

In this section, we formulate the problem studied in this paper and propose the AsySQN framework for VFL, which has the lower CC and the better CRU when applied to the industry.

2.1 Problem Formulation

In this paper, we consider a VFL system with q parties, where each party holds different features of the same samples. Given a training set $\{x_i, y_i\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$, $y_i \in \{-1, +1\}$ for binary classification task or $y_i \in \mathbb{R}$ for regression problem. Then x can be represented as a concatenation of all features, *i.e.*, $x = [x_1, \cdots, x_q]$, where $x_\ell \in \mathbb{R}^{d_\ell}$ is stored privately on party ℓ, and $\sum_{\ell=1}^q d_\ell = d$. Similar to previous works [7, 11], we assume the labels are held by all parties. In this paper, we consider the model in the form of $w^{\mathrm{T}}x$, where $w \in \mathbb{R}^d$ corresponds to the model parameters. Particularly, we focus on the regularized empirical risk minimization problem with following form

$$\min_{w\in\mathbb{R}^d} f(w) := \frac{1}{n}\sum_{i=1}^{n} \underbrace{\mathcal{L}(\theta_i, y_i) + \lambda g(w)}_{f_i(w)}, \tag{P}$$

where $\theta_i = \sum_{\ell=1}^q \theta_i^\ell = \sum_{\ell=1}^q w_\ell^{\mathrm{T}}(x_i)_\ell$, $\mathcal{L}$ denotes the loss function, $g(w) = \sum_{\ell=1}^q g_\ell(w_\ell)$ is the regularization term, $f_i : \mathbb{R}^d \to \mathbb{R}$ is strongly convex. Problem P capsules many machine learning problems such as the widely used ℓ_2-regularized logistic regression [3], least squares support vector machine [21] and ridge regression [20].

2.2 AsySQN Framework

To address the CC and CRU challenges for VFL application to industry, we propose the AsySQN framework shown in Fig. 1. In AsySQN, the data are vertically distributed over all parties and each party can not directly exchange the data due to privacy concerning. As shown in algorithm module, each party uses the descent direction $d^\ell = H^\ell v^\ell$ generated locally (refer to Algorithms 1) to update its model and the θ_i^ℓ used for calculating the local H^ℓ are aggregated from the other parties through Algorithm 2. In the following, we present that it is not easy to design AsySQN.

First, we consider using locally stored information, *i.e.*, the block-coordinate gradient difference and local parameter difference, to

Algorithm 1 Stochastic damped L-BFGS on Party ℓ.

Input: Let k be current local iteration number, m is the memory size, stochastic gradient estimator v_{k-1}^{ℓ}, index set $\mathcal{I}_k$ at local iteration k and vector pairs $\{s_i^{\ell}, \bar{y}_i^{\ell}, \rho_i^{\ell}\}$ $i = k-m, \ldots, k-2$ stored on party ℓ, and let $u_0 = v_k^{\ell}$

1: Calculate s_{k-1}^{ℓ}, $\bar{y}_{k-1}^{\ell}$ and γ_k^{ℓ}
2: Calculate $\hat{y}_{k-1}^{\ell}$ through Eq. 1 and $\rho_{k-1}^{\ell} = ((s_{k-1}^{\ell})^{\top}\hat{y}_{k-1}^{\ell})^{-1}$
3: **for** $i = 0, \ldots, \min\{m, k-1\} - 1$ **do**
4: Calculate $\mu_i^{\ell} = \rho_{k-i-1}^{\ell}(u_i^{\ell})^{\top} s_{k-i-1}^{\ell}$
5: Calculate $u_{i+1}^{\ell} = u_i^{\ell} - \mu_i^{\ell}\hat{y}_{k-i-1}^{\ell}$
6: **end for**
7: Calculate $v_0^{\ell} = (\gamma_k^{\ell})^{-1} u_p^{\ell}$
8: **for** $i = 0, \ldots, \min\{m, k-1\} - 1$ **do**
9: Calculate $\nu_i^{\ell} = \rho_{k-m+i}^{\ell}(v_i^{\ell})^{\top}\hat{y}_{k-m+i}^{\ell}$
10: Calculate $\bar{v}_{i+1} = \bar{v}_i^{\ell} + (\mu_{m-i-1}^{\ell} - \nu_i^{\ell}) s_{k-m+i}^{\ell}$.
11: **end for**

Output: $d^{\ell} = H_k^{\ell} v_k^{\ell} = \bar{v}_p^{\ell}$.

calculate the approximate local Hessian information implicitly by SLBFGS [29] to circumvent the large PRCO of transmitting the gradient difference [23]. Then, to improve the CRU of SQN methods when applied to real-world VFL systems with unbalanced computation resource, we consider asynchronously parallelizing the SQN algorithms for VFL. However, such parallelization can be difficult because those θ_i^{ℓ}'s contributed by the other parties for computing the k-th (k is current local iteration number) local H_k^{ℓ} are always stale, which may lead to an unstable SLBFGS process and make the generated local H_k (for notation abbreviation, we omit the superscript ℓ, so do other notations in this section) even not positive semidefinite for (strongly) convex problem. To address this challenge, motivated by damped LBFGS [19] for nonconvex problem, we turn to designing the stochastic damped L-BFGS (SdLBFGS) for VFL with painstaking to ensure the generated H_k be positive semidefinite (please refer to the reXiv version for the proof). The corresponding algorithm is shown in Algorithm 1.

Instead of using gradient difference transmitted from the other parties, Algorithm 1 uses the history information stored locally on party ℓ to generate a local descent direction $d_k = H_k v_k$ without calculating inverse matrix H_k explicitly. In Algorithm 1, $s_{k-1} = x_k - x_{k-1}$ and $\bar{y}_{k-1} = v_k - v_{k-1}$, and $\gamma_k = \max\{\frac{\bar{y}_{k-1}^{\top}\bar{y}_{k-1}}{s_{k-1}^{\top}\bar{y}_{k-1}}, \delta\}$, where δ is a positive constant. Different from traditional SLBFGS for convex problem [1], algorithm 1 introduces a new vector $\hat{y}_{k-1}$.

$$\hat{y}_{k-1} = \theta_{k-1}\bar{y}_{k-1} + (1-\theta_{k-1})H_{k-1,0}^{-1} s_{k-1}, k \geq 1, \tag{1}$$

where $H_{k,0} = \gamma_k^{-1} I_{d_\ell \times d_\ell}$, $k \geq 0$, and θ_{k-1} is defined as

$$\theta_{k-1} = \begin{cases} \frac{0.7\sigma_{k-1}}{\sigma_{k-1} - s_{k-1}^{\top}\bar{y}_{k-1}}, & \text{if } s_{k-1}^{\top}\bar{y}_{k-1} < 0.3\sigma_{k-1} \\ 1, & \text{otherwise} \end{cases}, \tag{2}$$

where $\sigma_{k-1} = s_{k-1}^{\top} H_{k,0}^{-1} s_{k-1}$. Since vector pairs $\{s_i, \hat{y}_i\}_{i=0}^{k-1}$ are obtained, and then $H_k v_k$ can be approximated through the two-loop recursion *i.e.*, steps 3 to 10. Importantly, the local H_k implicitly generated is positive semidefinite despite that the history information is stale (Please refer to arXiv version for the proof).

Algorithm 2 Safe algorithm of obtaining θ_i

Input: w, index i
Do this in parallel
1: **for** $\ell' = 1, \cdots, q$ **do**
2: Generate a random number $\delta_{\ell'}$ and calculate $\theta_i^{\ell'} + \delta_{\ell'}$,
3: **end for**
4: Obtain $\varphi_1 = \sum_{\ell'=1}^{q}(\theta_i^{\ell'} + \delta_{\ell'})$ based on tree structure T_1.
5: Obatin $\varphi_2 = \sum_{\ell'=1}^{q}\delta_{\ell'}$ based on significally different tree structure (please refer to [8]) $T_2 \neq T_1$.

Output: $\theta_i = \varphi_1 - \varphi_2$

In our AsySQN framework, each party needs to compute the corresponding stochastic (block-coordinate) gradient estimator v_k for generating the local H_k. Given $f_i(w)$ defined in Problem P, the block-coordinate gradient can be represented as

$$\nabla_{\mathcal{G}_\ell} f_i(w) = H(\theta_i, y_i)(x_i)_\ell + \lambda \nabla g_\ell(w_{\mathcal{G}_\ell}), \tag{3}$$

where $H(\theta_i, y_i) = \frac{\partial \mathcal{L}(\theta_i, y_i)}{\partial \theta_i}$. Thus, one need obtain $\theta_i = \sum_{\ell=1}^{q} w_\ell^{\mathrm{T}}(x_i)_\ell$ for computing the block-coordinate gradient. To avoid the large communication overhead of directly transmitting w_ℓ^{T} and $(x_i)_\ell$ and prevent the directly leaking of them, we consider transmitting the computational results of θ_i^{ℓ}. Many recent works achieved this in different manners [8, 11, 16, 28]. In this paper, we use the efficient tree-structured communication scheme [27].

Aggregating θ_i with Privacy-Preserving: The details are summarized in Algorithm 2. Specifically, at step 2, $\theta_i^{\ell} = w_\ell^{\mathrm{T}}(x_i)_\ell$ is computed locally on the ℓ-th party to prevent the direct leakage of w_ℓ and $(x_i)_\ell$. Especially, a random number δ_ℓ is added to θ_i^{ℓ} to mask the value of θ_i^{ℓ}, which can enhance the security during aggregation process. At steps 4 and 5, θ_1 and θ_2 are aggregated through tree structures T_1 and T_2, respectively. Note that T_2 is totally different (refer to [8] for definition) from T_1 that can prevent the random value being removed under threat model 1 (defined in Section 5). Finally, value of θ_i is recovered by removing term $\sum_{\ell=1}^{q}\delta_\ell$ from $\sum_{\ell=1}^{q}(\theta_i^{\ell} + \delta_\ell)$ at the output step. Using such aggregation strategy, $(x_i)_\ell$ and w_ℓ are prevented from leaking during the aggregation, the data and model securities are thus guaranteed

2.3 AsySQN-Type Algorithms for VFL

Vanilia SGD [6] and its variance reduction variants [12–14] are popular methods for learning machine learning (ML) models [4, 26]. In this paper, we thus propose three SGD-based AsySQN algorithms.

AsySQN-SGD: First, we propose the vanilla AsySQN-SGD summarized in Algorithm 3. At each iteration, AsySQN-SGD randomly samples a batch of samples $\mathcal{I}$ with replacement, and then obtain the vector $\theta_{\mathcal{I}}$ asynchronously based on tree-structured communication. Based on the received vector $\theta_{\mathcal{I}}$, $\nabla_\ell f_i(\widehat{w})$ is computed as Eq. 3 and the stochastic gradient estimator used for generating d_k is computed as $\widehat{v}^{\ell} = \nabla_\ell f_{\mathcal{I}}(\widehat{w}) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\nabla_\ell f_i(\widehat{w})$.

AsySQN-SVRG: The proposed AsySQN-SVRG with an improved convergence rate than AsySQN-SGD is shown in Algorithm 4. Different from AsySQN-SGD directly using the stochastic gradient for updating, AsySQN-SVRG adopts the variance reduction technique to control the intrinsic variance of the stochastic gradient estimator. In this case, $\widehat{v}^{\ell} = \nabla_\ell f_{\mathcal{I}}(\widehat{w}) - \nabla_\ell f_{\mathcal{I}}(w^s) + \nabla_\ell f(w^s)$.

Algorithm 3 AsySQN-SGD algorithm on the ℓ-th worker

Input: Data $\{D^{\ell'}\}_{\ell'=1}^{q}$ and learning rate γ

Initialize $w_\ell \in \mathbb{R}^d$.
Keep doing in parallel
Sample $\mathcal{I} \overset{\text{Unif}}{\sim} \{1,\dots,n\}$.
Compute $\theta_{\mathcal{I}} = \{\theta_i\}_{i\in\mathcal{I}}$ asynchronously based on Algorithm 2.
Compute $\widehat{v}^\ell = \nabla_\ell f_{\mathcal{I}}(\widehat{w})$.
Compute $d^\ell = H^\ell \widehat{v}^\ell$ through Algorithm 1,
Update $w_\ell \leftarrow w_\ell - \gamma d^\ell$.
End parallel

Output: w_ℓ

Algorithm 4 AsySQN-SVRG algorithm on the ℓ-th worker

Input: Data $\{D^{\ell'}\}_{\ell'=1}^{q}$ and learning rate γ

Initialize $w_\ell^0 \in \mathbb{R}^{d_\ell}$.
for $s = 0, 1, \dots, S-1$ **do**
Compute the full local gradient $\nabla_\ell f(w^s) = \frac{1}{n}\sum_{i=1}^{n} \nabla_\ell f_i(w^s)$ through tree-structured communication.
$w_\ell = w_\ell^s$.
Keep doing in parallel
Sample $\mathcal{I} \overset{\text{Unif}}{\sim} \{1,\dots,n\}$.
Compute $\theta_{\mathcal{I}} = \{\theta_i\}_{i\in\mathcal{I}}$ and $\theta(w^s)_{\mathcal{I}} = \{\theta_i(w^s)\}_{i\in\mathcal{I}}$ based on Algorithm 2.
Compute $\widehat{v}^\ell = \nabla_\ell f_{\mathcal{I}}(\widehat{w}) - \nabla_\ell f_{\mathcal{I}}(w^s) + \nabla_\ell f(w^s)$
Compute $d^\ell = H^\ell \widehat{v}^\ell$ through Algorithm 1,
Update $w_\ell \leftarrow w_\ell - \gamma d^\ell$.
End parallel loop
$w_\ell^{s+1} = w_\ell$.
end for

Output: w_ℓ

AsySQN-SAGA: AsySQN-SAGA enjoying the same convergence rate with AsySQN-SVRG is shown in Algorithm 5. Different from Algorithm 4 using w^s as the reference gradient, AsySQN-SAGA uses the average of history gradients stored in a table. The corresponding $\widehat{v}^\ell$ is computed as $\widehat{v}^\ell = \nabla_\ell f_{\mathcal{I}}(\widehat{w}) - \widehat{\alpha}_{\mathcal{I}}^\ell + \frac{1}{n}\sum_{i=1}^{n}\widehat{\alpha}_i^\ell$.

3 CONVERGENCE ANALYSIS

In this section, the convergence analysis is presented. Please refer to the arXiv verison for details.

3.1 Preliminaries

Assumption 1. *Each function f_i, $i = 1,\dots,n$, is μ-strongly convex, i.e., $\forall\ w,\ w' \in \mathbb{R}^d$ there exists a $\mu > 0$ such that*

$$f_i(w) \geq f_i(w') + \langle \nabla f_i(w'), w - w' \rangle + \frac{\mu}{2}\|w - w'\|^2. \tag{4}$$

Assumption 2. *For each function f_i, $i = 1,\dots,n$, we assume the following conditions hold:*

2.1 Lipschitz Gradient: *There exists $L > 0$ such that*

$$\|\nabla f_i(w) - \nabla f_i(w')\| \leq L\|w - w'\|, \quad \forall\ w, w' \in \mathbb{R}^d \tag{5}$$

Algorithm 5 AsySQN-SAGA algorithm on the ℓ-th worker

Input: *Data $\{D^{\ell'}\}_{\ell'=1}^{q}$ and learning rate γ*

Initialize $w_\ell \in \mathbb{R}^{d_\ell}$.
Compute the local gradient $\widehat{\alpha}_i^\ell = \nabla_\ell f_i(\widehat{w})$, for $\forall i \in \{1,\dots,n\}$ through tree-structured communication.
Keep doing in parallel
Sample $\mathcal{I} \overset{Unif}{\sim} \{1,\dots,n\}$.
Compute $\theta_{\mathcal{I}} = \{\theta_i\}_{i\in\mathcal{I}}$ asynchronously based on Algorithm 2.
Compute $\widehat{v}^\ell = \nabla_\ell f_{\mathcal{I}}(\widehat{w}) - \widehat{\alpha}_{\mathcal{I}}^\ell + \frac{1}{n}\sum_{i=1}^{n}\widehat{\alpha}_i^\ell$.
Compute $d^\ell = H^\ell \widehat{v}^\ell$ through Algorithm 1,
Update $w_\ell \leftarrow w_\ell - \gamma d^\ell$.
Update $\widehat{\alpha}_{\mathcal{I}}^\ell \leftarrow \nabla_\ell f_{\mathcal{I}}(\widehat{w})$.
End parallel loop

Output: *w_ℓ*

2.2 Block-Coordinate Lipschitz Gradient: *There exists an $L_\ell > 0$ for the ℓ-th block, where $\ell \in [q]$ such that*

$$\|\nabla_\ell f_i(w + U_\ell \Delta_\ell) - \nabla_\ell f_i(w)\| \leq L_\ell \|\Delta_\ell\|, \tag{6}$$

where $\Delta_\ell \in \mathbb{R}^{d_\ell}$, $U_\ell \in \mathbb{R}^{d\times d_\ell}$ and $[U_1, \cdots, U_q] = I_d$.

2.3 Bounded Block-Coordinate Gradient: *There exists a constant G such that $\|\nabla_\ell f_i(w)\|^2 \leq G$ for ℓ, $\ell = 1, \cdots, q$.*

Assumptions 2.1 to 2.3 are standard for convergence analysis in previous works [9, 28].

Assumption 3. *We introduce the following assumptions necessary for the analysis of stochastic quasi-Newton methods.*

3.1 *For $i \in [n]$, function $f_i(w)$ is twice continuously differentiable w.r.t. w and for the ℓ-th block, where $\ell \in [q]$, there exists two positive constant κ_1 and κ_2 such that for $\forall\ w \in \mathbb{R}^{d_\ell}$ there is*

$$\kappa_1 I \preceq \nabla_\ell^2 f_i(w) \preceq \kappa_2 I \tag{7}$$

where notation $A \preceq B$ with $A, B \in \mathbb{R}^{d_\ell \times d_\ell}$ means that $A - B$ is positive semidefinite.

3.2 *There exist two positive constants σ_1, and σ_2 such that*

$$\sigma_1 I \preceq H_k^\ell \preceq \sigma_2 I, \quad \ell \in [q] \tag{8}$$

where H_k^ℓ is the inverse Hessian approximation matrix on party ℓ.

3.3 *For any local iteration $k \geq 2$, the random variable H_k^ℓ ($k \geq 2$) depends only on v_{k-1} and $\mathcal{I}_k$*

$$\mathbb{E}[H_k v_k | \mathcal{I}_k, v_{k-1}] = H_k v_k, \tag{9}$$

where the expectation is taken with respect to $|\mathcal{I}_k|$ samples generated for calculation of $\nabla f_{\mathcal{I}_k}$.

Assumptions 3.1 to 3.3 are standard assumptions for SQN methods [29]. Specifically, Assumption 3.2 shows that the matrix norm of H_k is bounded. Assumption 3.3 means that given v_{k-1} and $\mathcal{I}_k$ the $H_k v_k$ is determined. Similar to previous asynchronous work [9, 28], we introduce the following definition and assumption.

Definition 1. *$D(t)$ is defined as a set of iterations, such that:*

$$\widehat{w}_t - w_t = \gamma \sum_{u\in D(t)} U_{\psi(u)} \widehat{v}_u^{\psi(u)}, \tag{10}$$

where $u \leq t$ for $\forall u \in D(t)$.

Assumption 4. **Bounded Overlap:** *We assume that there exists a constant τ_1 which bounds the maximum number of iterations that can overlap together, i.e., for $\forall t$ there is $\tau_1 \geq t - min\{u|u \in D(t)\}$.*

To track the behavior of the global model in the convergence analysis, it is necessary to introduce $K(t)$.

Definition 2. *$K(t)$: The minimum set of successive iterations that fully visit all coordinates from global iteration number t.*

Assumption 5. *We assume that the size of $K(t)$ is upper bounded by η_1,* i.e., *$|K(t)| \leq \eta_1$.*

Based on Definition 2 and Assumption 5, we introduce the epoch number $v(t)$, which our convergence analyses are built on.

Definition 3. *Let $P(t)$ be a partition of $\{0, 1, \cdots, t-\sigma'\}$, where $\sigma' \geq 0$. For any $\kappa \subseteq P(t)$ we have that there exists $t' \leq t$ such that $K(t') = \kappa$, and $\kappa_1 \subseteq P(t)$ such that $K(0) = \kappa_1$. The epoch number for the global t-th iteration, i.e., $v(t)$ is defined as the maximum cardinality of $P(t)$.*

3.2 Convergence Analyses

Theorem 1. *Under Assumptions 1-5, to achieve the accuracy ϵ of (P) for AsySQN-SGD,* i.e., *$\mathbb{E}f(w_t) - f(w^*) \leq \epsilon$, we set $\gamma = \frac{-L_{\max}+\sqrt{L_{\max}^2+\frac{2\mu\sigma_1 b\epsilon(\sigma_1 L^2\eta_1^2+\sigma_2\tau_1 L^2\tau)}{G\eta_1\sigma_2^2}}}{2L^2(\sigma_1\eta_1^2+\sigma_2\tau_1^2)}$ and the epoch number $v(t)$ should satisfy the following condition.*

$$v(t) \geq \frac{2}{\mu\gamma\sigma_1}\log\left(\frac{2\left(f(w_0)-f(w^*)\right)}{\epsilon}\right) \tag{11}$$

Theorem 2. *Under Assumptions 1-5, to achieve the accuracy ϵ of (P) for AsySQN-SVRG, let $C = \left(\sigma_1\gamma L^2\eta_1^2 + L_{\max}\right)\frac{\gamma^2\sigma_2^2}{2}$ and $\rho = \frac{\gamma\mu\sigma_1}{2} - \frac{16L^2\eta_1 C}{\mu}$, we choose γ such that*

$$1)\ \rho < 0; \quad 2)\ \frac{8L^2\eta_1 C}{\rho\mu} \leq 0.5 \tag{12}$$

$$3)\ \gamma^3\left(\left(\frac{\sigma_1}{2}+\frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\frac{9\eta_1 L^2\sigma_2^2 G}{b\rho} \leq \frac{\epsilon}{8} \tag{13}$$

the inner epoch number should satisfy $v(t) \geq \frac{\log 0.25}{\log(1-\rho)}$, and the outer loop number should satisfy $S \geq \frac{\log\frac{2(f(w_0)-f(w^*))}{\epsilon}}{\log\frac{4}{3}}$.

Theorem 3. *Under Assumptions 1-5, to achieve the accuracy ϵ of (P) for AsySQN-SAGA,* i.e., *$\mathbb{E}f(w_t) - f(w^*) \leq \epsilon$, let $c_0 = \left(\frac{\tau_1^2}{2} + (9\tau_1^2 + 8\eta_1^2)\sigma_2\gamma\left(\gamma L^2\sigma_1\eta_1^2 + L_{\max}\right)\right)\sigma_2^3\gamma^3 L^2\eta_1\frac{G}{b}$,*

$c_1 = \left(\gamma L^2\sigma_1\eta_1^2 + L_{\max}\right)\sigma_2^2\gamma^2\eta_1\frac{L^2}{b}$, $c_2 = 4\left(\gamma L^2\sigma_1\eta_1^2 + L_{\max}\right) * \frac{L^2\eta_1^2\sigma_2^2\gamma^2}{nb}$, and let $\rho \in (1-\frac{1}{n}, 1)$, we choose γ such that

$$1)\frac{4c_0}{\gamma\sigma_1\mu(1-\rho)\left(\frac{\gamma\sigma_1\mu^2}{4}-2c_1-c_2\right)} \leq \frac{\epsilon}{2}, \quad 2)0 < 1 - \frac{\gamma\sigma_1\mu}{4} < 1$$

$$3) -\frac{\gamma\sigma_1\mu^2}{4} + 2c_1 + c_2\left(1 + \frac{1}{1-\frac{1-\frac{1}{n}}{\rho}}\right) \leq 0$$

$$4) -\frac{\gamma\sigma_1\mu^2}{4} + c_2 + c_1\left(2 + \frac{1}{1-\frac{1-\frac{1}{n}}{\rho}}\right) \leq 0 \tag{14}$$

the epoch number should satisfy $v(t) \geq \frac{\log\frac{2\left(2\rho-1+\frac{\gamma\sigma_1\mu}{4}\right)(f(w_0)-f(w^*))}{\epsilon(\rho-1+\frac{\gamma\sigma_1\mu}{4})\left(\frac{\gamma\sigma_1\mu^2}{4}-2c_1-c_2\right)}}{\log\frac{1}{\rho}}$.

Remark 1. *For strongly convex problems, given the assumptions and parameters in corresponding theorems, the convergence rate of AsySQN-SGD is $O(\frac{1}{\epsilon}log(\frac{1}{\epsilon}))$, and those of AsySQN-SVRG and AsySQN-SAGA are $O(log(\frac{1}{\epsilon}))$.*

4 COMPLEXITY ANALYSES

In this section, we present the computation and communication complexity analyses of our framework.

4.1 Computation Complexity Analysis

SQN methods incorporated with approximate Hessian information indeed converge faster than SGD methods in practice, however, there is a concern that it may introduce much extra computation cost. In the following, we will show that the extra computation cost introduced by approximating Hessian information is negligible.

First, we analyze the computation complexity of Algorithm 1. At step 1, two inner products take $2d_\ell$ multiplications. At step 2, two inner products and one scalar-vector product take $3d_\ell$ multiplications. The first recursive loop (*i.e.*, Steps 3 to 5) involves $2m$ scalar-vector multiplications and m vector inner products, which takes $3md_\ell$ multiplications. So does the second loop (*i.e.*, Steps 8 to 10). At step 7, the scalar-vector product takes d_ℓ multiplications. Therefore, Algorithm 1 takes $(6m + d)d_\ell$ multiplications totally. As for Algorithm 2, all multiplications are performed at steps 1-3, which takes $\sum_{\ell=1}^{q} d_\ell = d$ multiplications.

Then we turn to Algorithm 3. At step 3, it takes bd multiplications to compute the vector θ_i. Step 4 takes bd_ℓ multiplications to compute the mini-batch stochastic gradient. Step 5 calls Algorithm 1 and thus takes $(6m + d)d_\ell$ multiplications. The scalar-vector product at step 6 takes d_ℓ multiplications. Compared with Algorithm 3, Algorithm 4 need compute the full gradient at the beginning of each epoch, which takes nd multiplications. Moreover, step 6 takes $2bd$ multiplications and step 7 takes $2bd_\ell$ multiplications. As for other steps, the analyses are similar to Algorithm 3. In terms of Algorithm 5, the initialization of the gradient matrix takes nd multiplications and the analyses of other steps are similar to Algorithm 3.

We summarize the detailed computation complexity of Algorithms 1 to 5 in Table 1. Based on the results in Table 1, it is obvious that, for Algorithm 3, the extra computation cost of computing the approximate second-order information takes up $r = \frac{md_\ell}{bd+md_\ell} < \frac{md_\ell}{bd}$ in the whole procedure. The extra computation cost is negligible because 1) as in previous work [6] we can choose $b = n^{-1/2}$ and n is sufficiently large in big data situation, 2) m ranges from 5 to 20 as suggested in [19], 3) generally, d_ℓ is much smaller than d. As for Algorithms 4 and 5, there are $r = \frac{TSmd_\ell}{ndS+TS(bd+md_\ell)}$ and $r = \frac{Tmd_\ell}{nd+T(bd+md_\ell)}$, respectively, which are also negligible.

4.2 Communication Complexity Analysis

The communication of Algorithm 2 is $O(q)$. For Algorithm 3, step 3 has a communication complexity of $O(bq)$ due to calling Algorithm 2 b times. Similarly, for Algorithm 4, communication complexity of steps 3 and 6 are $O(nq)$ and $O(bq)$, respectively. For Algorithm 5,

Table 1: Total computational complexities (TCC) of going through Algorithms 1 to 5, where t denotes the total iteration number for Algorithms 3 and 5, and the number of iterations in an epoch for Algorithm 4, and S is the epoch number for Algorithm 4.

Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4		Algorithm 5	
steps	TCC	steps	TCC	steps	TCC	steps	TCC	steps	TCC
1	$O(d_\ell)$	1-3	$O(d)$	3	$O(bd)$	3	$O(nd)$	2	$O(nd)$
2	$O(d_\ell)$	-	-	4	$O(bd_\ell)$	6	$O(bd)$	4	$O(bd)$
3-6	$O(md_\ell)$	-	-	5	$O(md_\ell)$	7	$O(bd_\ell)$	5	$O(bd_\ell)$
7	$O(d_\ell)$	-	-	6	$O(d_\ell)$	8	$O(md_\ell)$	6	$O(d_\ell)$
8-11	$O(md_\ell)$	-	-	4	$O(d_\ell)$	9	$O(d_\ell)$	-	-
total	$O(md_\ell)$	**total**	$O(d)$	**total**	$O[T(bd+md_\ell)]$	**total**	$O[ndS+TS(bd+md_\ell)]$	**total**	$O[nd+T(bd+md_\ell)]$

communication complexity of steps 2 and 4 are $O(nq)$ and $O(bq)$, respectively. Given T and S defined in Table 1, the total communication complexities of Algorithms 3 to 5 are $O(Tbq)$, $O(S(n+T)bq)$ and $O((n+T)bq)$, respectively.

5 PRIVACY SECURITY ANALYSIS

In this section, we analyze the data security and model security of AsySQN framework under two semi-honest threat models commonly used in previous works [2, 8, 22]. Especially, adversaries under threat model 2 can collude with each other, thus they have the stronger attack ability than those under threat model 1.

Honest-but-Curious (Threat Model 1): All workers will follow the federated learning protocol to perform the correct computations. However, they may use their own retained records of the intermediate computation result to infer other worker's data and model.

Honest-but-Colluding (Threat Model 2): All workers will follow the federated learning protocol to perform the correct computations. However, some workers may collude to infer other worker's data and model by sharing their retained records of the intermediate computation result.

Similar to [8, 9], we analyze the security of AsySQN by analyzing its ability to prevent following inference attacks.

Definition 4 (**Exact Inference Attack**). *The adversary perform the inference attack by inferring $(x_i)_\ell$ (or w_ℓ) belonging to other parties without directly accessing them.*

Definition 5 (ϵ-**Approximate Exact Inference Attack**). *The adversary perform the ϵ-approximate exact inference attack by inferring $(x_i)_\ell$ (or w_ℓ) belonging to other parties as $(\widetilde{x}_i)_\ell$ (or $\widetilde{w}_\ell$) with accuracy of ϵ* (i.e., $\|(x_i)_\ell - (\widetilde{x}_i)_\ell\|_\infty \leq \epsilon$, $\|w_\ell - \widetilde{w}_\ell\|_\infty \leq \epsilon$, *or* $\|y_i - \widetilde{y}_i\|_\infty \leq \epsilon$ *) without directly accessing them.*

Lemma 1. *Given equations $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ with only observing θ_i^ℓ, there are infinite different solutions to both equations.*

Proof of Lemma 1. First, we consider the equation $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ with two cases, *i.e.*, $d_\ell \geq 2$ and $d_\ell = 1$. For $\forall d_\ell \geq 2$, given an arbitrary non-identity orthogonal matrix $U \in \mathbb{R}^{d_\ell \times d_\ell}$, we have

$$(w_\ell^{\mathrm{T}}U^{\mathrm{T}})(U(x_i)_\ell) = w_\ell^{\mathrm{T}}(U^{\mathrm{T}}U)(x_i)_\ell = w_\ell^{\mathrm{T}}(x_i)_\ell = \theta_i^\ell \qquad (15)$$

From Eq. 15, we have that given an equation $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ with only θ_i^ℓ being known, the solutions corresponding to w_ℓ and $(x_i)_\ell$ can be represented as $w_\ell^{\mathrm{T}}U^{\mathrm{T}}$ and $U(x_i)_\ell$, respectively. U can be arbitrary different non-identity orthogonal matrices, the solutions are thus infinite. If $d_\ell = 1$, give an arbitrary real number $u \neq 1$, we have

$$(w_\ell^{\mathrm{T}}u)(\frac{1}{u}(x_i)_\ell) = w_\ell^{\mathrm{T}}(u\frac{1}{u})(x_i)_\ell = w_\ell^{\mathrm{T}}(x_i)_\ell = \theta_i^\ell \qquad (16)$$

Similarly, we have that the solutions of equation $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ are infinite when $d_\ell = 1$. This completes the proof. □

Based on lemma 1, we obtain the following theorem.

Theorem 4. *AsySQN can prevent the exact and the ϵ-approximate exact inference attacks under semi-honest threat models.*

Proof of Theorem 4. We prove above theorem under following two threat models.

Threats Model 1: During the aggregation, the value of $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ is masked by δ_ℓ and just the value of $\theta_i^\ell + \delta_\ell$ is transmitted. In this case, one cannot even access the true value of θ_i^ℓ, let alone using relation $\theta_i^\ell = w_\ell^{\mathrm{T}}(x_i)_\ell$ to refer w_ℓ^{T} and $(x_i)_\ell$. Thus, AsySQN can prevent both exact inference attack and the ϵ-approximate exact inference attack under Threat Model 1.

Threats Model 2: Under threat model 2, the adversary can remove the random value δ_ℓ from term $\theta_i^\ell + \delta_\ell$ by colluding with other adversaries. Applying Lemma 1 to this circumstance, and we have that even if the random value is removed it is still impossible to exactly refer w_ℓ^{T} and $(x_i)_\ell$. However, it is possible to approximately infer w_ℓ when $d_\ell = 1$. Specifically, if one knows the region of $(x_i)_\ell$ as $\mathcal{R}$ (e.g., applying z-score normalization to $\{(x_i)_\ell\}_{i=1}^n$), one has $\theta_i^\ell / w_\ell^{\mathrm{T}} \in \mathcal{R}$, thus can infer w_ℓ^{T} approximately, so does infer $(x_i)_\ell$ approximately. Importantly, one can avoid this attack easily by zero-padding to make $d_\ell \geq 2$. Thus, under this threat model, AsySQN is secure in practice. □

6 EXPERIMENTS

In this section, we implement extensive experiments on real-world datasets to demonstrate the lower communication cost and better CRU of AsySQN. The results concerning convergence speed also consistent to the corresponding theoretical results.

6.1 Experiment Settings

All experiments are performed on a machine with four sockets, and each sockets has 12 cores. We use OpenMPI to implement the communication scheme. In the experiments, there are $q = 8$ parties and each party owns nearly equal number of features. δ is

Table 2: Datasets used in the experiments.

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8
#Samples	24,000	96,257	17,996	49,749	677,399	32,561	400,000	60,000
#Features	90	92	1,355,191	300	47,236	127	2,000	780

fixed for specific SGD-type of algorithms. As for the learning rate γ, we choose a suitable one from $\{\cdots, 1e^{-1}, 5e^{-2}, 2e^{-2}, 1e^{-2}, \cdots\}$ for all experiments. Moreover, to simulate the industry application scenarios, we set a synthetic party which only has 30% to 60% computation resource compared with the faster party. This means that, as for synchronous algorithms, the faster party has an poor CRU around only 30% to 60% due to waiting for the straggler for synchronization.

Problem for Evaluation: In this paper, we use the popular ℓ_2-norm regularized logistic regression problem for evaluation.

$$\min_{w\in\mathbb{R}^d} f(w) := \frac{1}{n}\sum_{i=1}^{n}\log(1+e^{-y_i w^{\mathrm{T}} x_i}) + \frac{\lambda}{2}\|w\|^2, \qquad (17)$$

where λ is set as $1e^{-4}$ for all experiments.

Datasets: We use eight datasets for evaluation, which are summarized in Table 2. Among them, D_1 (UCICreditCard) and D_2 (GiveMeSomeCredit) are from the Kaggle[1], and D_3 (news20), D_4(w8a), D_5 (rcv1), D_6 (a9a), D_7 (epsilon) and D_8 (mnist) are from the LIBSVM[2]. Especially, D_1 and D_2 are the financial datasets, which are used to demonstrate the ability to address real applications. Following previous works, we apply one-hot encoding to categorical features of D_1 and D_2 , thus the number of features become 90 and 92, respectively, for D_1 and D_2.

6.2 Evaluation of Lower Communication Cost

We demonstrate that our proposed AsySQN-type algorithms have the lower communication costs by showing that AsySQN reduces the NCR and has a low per-round communication overhead.

Reducing the Number of Communication Rounds: To demonstrate that our AsySQN-type algorithms can significantly reduce the NCR, we compare them with the corresponding asynchronous stochastic first-order methods for VFL, e.g., compare AsySQN-SGD with AFSGD-VP [9]. The experimental results are presented in Fig. 2. As depicted in Fig. 2, to achieve the same sub-optimality our AsySQN-type algorithms needs much lower NCR than the corresponding first-order algorithms, which is consistent to our claim that AsySQN-type algorithms incorporated with approximate second-order information can dramatically reduce the number of communication rounds in practice.

Low Per-Round Communication Overhead: Note that, the quasi-Newton (QN)-based framework has already been proposed in [23], which, however, is not communication-efficient due to large per-round communication overhead of transmitting the gradient difference. It is unnecessary to compare AsySQN framework with that QN-based because they have totally different structures. But it is necessary to demonstrate that our AsySQN is more communication-efficient than that transmitting the gradient difference. Thus, we compare the time spending of transmitting just the θ^ℓ with that of transmitting the gradient (difference). The results of communication time improvement (CTI) are presented in Table 3, where

$$\text{CTI} = \frac{CT_1 \text{ on transmitting a vector} \in \mathbb{R}^{d_\ell}}{CT_1 \text{ on transmitting a scalar}},$$

where CT_1 denotes communication time. All experiments are implemented with $q = 8$ parties and the results are obtained in 10 trials. The results in Table 3 demonstrate that AsySQN indeed has a lower per-round communication overhead than transmitting the gradient. Specifically, when d_ℓ is small, the intrinsic communication time spending (*i.e.*, time spending on transmitting nothing) dominates, thus the CRI is not significant. Importantly, when d_ℓ is significantly large (*e.g.*, D_5, and even D_3) the CRI is very remarkable.

6.3 Evaluation of Better CRU

To demonstrate that our AsySQN-type algorithms have a better CRU, we compare them with the corresponding synchronous algorithms (SynSQN-type). We use CRUI to denote the CRU improvement of our asynchronous algorithms relative to the corresponding synchronous algorithms

$$\text{CRUI} = \frac{CT_2 \text{ of SynSQN-type algorithm}}{CT_2 \text{ of AsySQN-type algorithm}},$$

where CT_2 (computation time) means the overall training time subtracts the communication time during a fixed number of iterations, *i.e.*, $21n$ for each datasets in our experiments. The results of CRUI are summarized in Table 5. As shown in Table 5, our asynchronous algorithms have much better CRU than the corresponding synchronous algorithms for real-world VFL systems with parties owning unbalanced computation resources. Moreover, the more unbalanced computation resources the slowest and the fastest parties have the higher CRUI will be.

6.4 Evaluation of Training Efficiency and Scalability

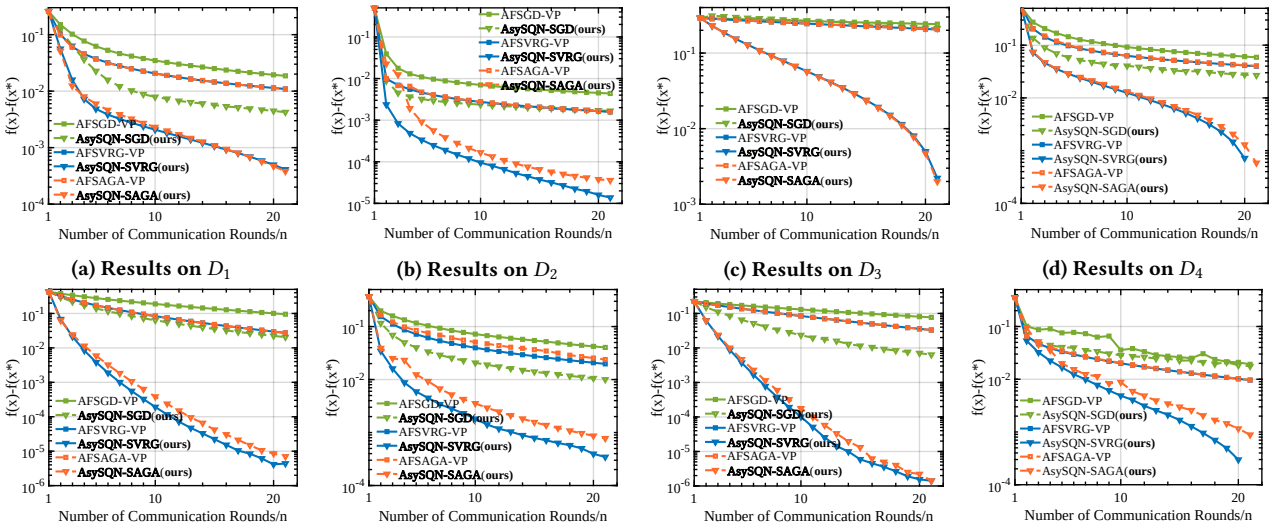
To directly show the efficiency for training VFL models, we compared them with the corresponding synchronous ones and depict the loss v.s. training time curves in Fig. 3.

We also consider the multiple-workers speedup scalability in terms of the number of parties and report the results in Fig. 4. Given q parties, there is

$$q\text{-workers speedup} = \frac{\text{TT for the serial computation}}{\text{TT of using } q \text{ parties}},$$

where the training time (TT) is the time spending on reaching a certain precision of sub-optimality, e.g., $5e^{-5}$ for D_6. As shown in Fig. 4, asynchronous algorithms have a much better multiple-parties speedup scalability than synchronous ones and can achieve near-linear speedup.

[1] https://www.kaggle.com/datasets
[2] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/



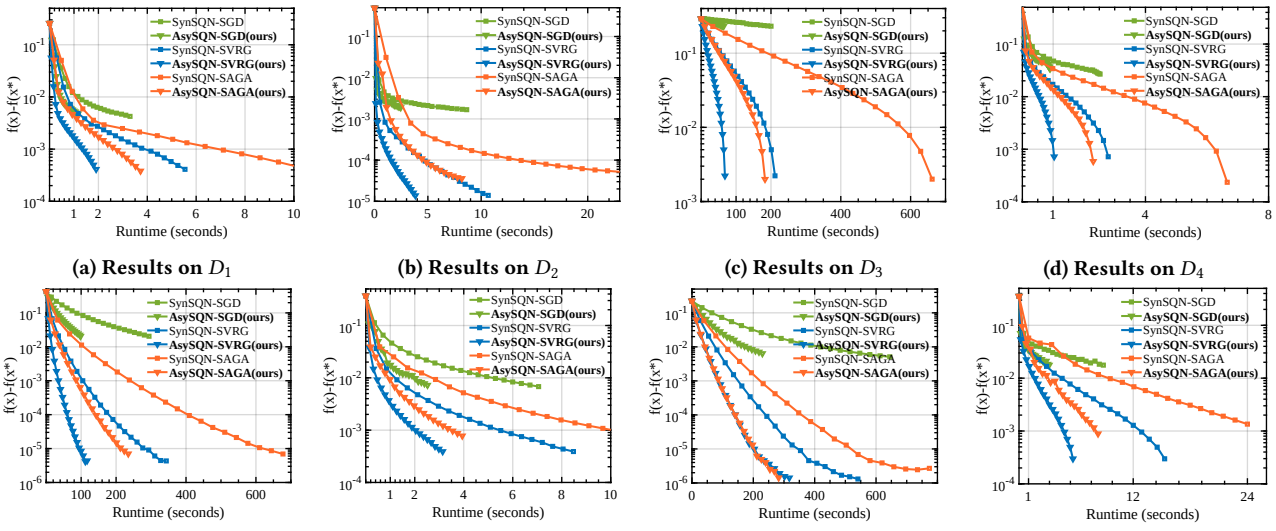
(a) Results on D_1 (b) Results on D_2 (c) Results on D_3 (d) Results on D_4

(e) Results on D_5 (f) Results on D_6 (g) Results on D_7 (h) Results on D_8

Figure 2: Sub-optimality v.s. the NCR on all datasets for solving μ-strongly convex VFL problems.

Table 3: Results of CTI on different datasets, which are obtained during the training process of $21n$ iterations (10 trials).

	$D_1(d_\ell = 12)$	$D_2(d_\ell = 12)$	$D_3(d_\ell = 169,398)$	$D_4(d_\ell = 37)$	$D_5(d_\ell = 5,904)$	$D_6(d_\ell = 16)$	$D_7(d_\ell = 250)$	$D_8(d_\ell = 98)$
CTI	1.057	1.059	89.265	1.187	5.688	1.083	1.771	1.333



(a) Results on D_1 (b) Results on D_2 (c) Results on D_3 (d) Results on D_4

(e) Results on D_5 (f) Results on D_6 (g) Results on D_7 (h) Results on D_8

Figure 3: Sub-optimality v.s. training time on all datasets for solving μ-strongly convex VFL problems.

6.5 Evaluation of Losslessness

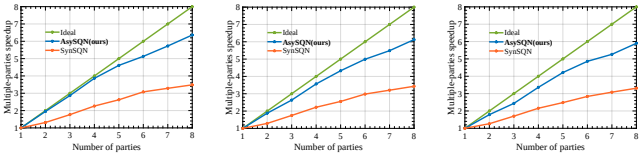
To demonstrate the losslessness of our algorithms, we compare AsySQN-type algorithms with its non-federated (NonF) counterparts (the only difference to the AsySQN-type algorithms is that all data are integrated together for modeling). For datasets without testing data, we split the data set into 10 parts, and use one of them for testing. Each comparison is repeated 10 times with $q = 8$, and a same stop criterion, *e.g.*, $5e^{-5}$ for D_6. As shown in Table 4, the accuracy of our algorithms are the same with those of NonF algorithms, which demonstrate that our VFL algorithms are lossless.

Table 4: Accuracy of different algorithms to evaluate the losslessness of our algorithms (10 trials).

Algorithm	D_1(%)	D_2(%)	D_3(%)	D_4(%)	D_5(%)	D_6(%)	D_7(%)	D_8(%)
NonF	81.96±0.02	93.56±0.03	98.29±0.02	90.21±0.02	96.02±0.03	85.03±0.02	87.43±0.04	87.01±0.04
Ours	81.96±0.03	93.56±0.06	98.29±0.03	90.21±0.03	96.02±0.03	85.03±0.04	87.43±0.06	87.01±0.05

Table 5: Improvements of CRU on all datasets.

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8
SGD-based	2.90	2.77	2.98	2.84	2.82	2.80	2.75	2.86
SVRG-based	2.96	2.80	3.01	2.95	2.86.	2.72	2.84	2.88
SAGA-based	2.99	2.92	3.09	2.96	2.91	2.79	2.74	2.92



(a) SAGA-based **(b) SVRG-based** **(c) SGD-based**

Figure 4: Multiple-workers speedup scalability on D_7.

7 CONCLUSION

In this paper, we proposed a novel AsySQN framework for the VFL applications to industry, where communication costs between different parties (*e.g.*, different corporations, companies and organizations) are expensive and different parties owning unbalanced computation resources. Our AsySQN framework with slight per-round communication overhead utilizes approximate second-order information can dramatically reduces the number of communication rounds, and thus has lower communication cost. Moreover, AsySQN enables parties with unbalanced computation resources asynchronously update the model, which can achieve better computation resource utilization. Three SGD-type algorithms with different stochastic gradient estimators were also proposed under AsySQN, *i.e.* AsySQN-SGD, -SVRG, SAGA, with theoretical guarantee for strongly convex problems.

REFERENCES

[1] Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer. 2016. A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization* 26, 2 (2016), 1008–1031.

[2] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, and Qiang Yang. 2019. SecureBoost: A Lossless Federated Learning Framework. *arXiv preprint arXiv:1901.08755* (2019).

[3] Bryan Conroy and Paul Sajda. 2012. Fast, exact model selection and permutation testing for l2-regularized logistic regression. In *AISTAS*. 246–254.

[4] Zhiyuan Dang, Xiang Li, Bin Gu, Cheng Deng, and Heng Huang. 2020. Large-Scale Nonlinear AUC Maximization via Triply Stochastic Gradients. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[5] Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. 2016. Secure Linear Regression on Vertically Partitioned Datasets. *IACR Cryptology ePrint Archive* 2016 (2016), 892.

[6] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. 2016. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming* 155, 1-2 (2016), 267–305.

[7] Yanmin Gong, Yuguang Fang, and Yuanxiong Guo. 2016. Private data analytics on biomedical sensing data via distributed computation. *IEEE/ACM transactions on computational biology and bioinformatics* 13, 3 (2016), 431–444.

[8] Bin Gu, Zhiyuan Dang, Xiang Li, and Heng Huang. 2020. Federated Doubly Stochastic Kernel Learning for Vertically Partitioned Data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2483–2493.

[9] Bin Gu, An Xu, Cheng Deng, and heng Huang. 2020. Privacy-Preserving Asynchronous Federated Learning Algorithms for Multi-Party Vertically Collaborative Learning. *arXiv preprint arXiv:2008.06233* (2020).

[10] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677* (2017).

[11] Yaochen Hu, Di Niu, Jianming Yang, and Shengping Zhou. 2019. Fdml: A collaborative machine learning framework for distributed features. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2232–2240.

[12] Feihu Huang, Songcan Chen, and Heng Huang. 2019. Faster Stochastic Alternating Direction Method of Multipliers for Nonconvex Optimization. In *ICML*. 2839–2848.

[13] Feihu Huang, Shangqian Gao, Jian Pei, and Heng Huang. 2019. Nonconvex zeroth-order stochastic admm methods with lower function query complexity. *arXiv preprint arXiv:1907.13463* (2019).

[14] Feihu Huang, Shangqian Gao, Jian Pei, and Heng Huang. 2020. Accelerated zeroth-order momentum methods from mini to minimax optimization. *arXiv preprint arXiv:2008.08170* (2020).

[15] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and Open Problems in Federated Learning. *arXiv preprint arXiv:1912.04977* (2019).

[16] Yang Liu, Yan Kang, Xinwei Zhang, Liping Li, Yong Cheng, Tianjian Chen, Mingyi Hong, and Qiang Yang. 2019. A communication efficient vertical federated learning framework. *arXiv preprint arXiv:1912.11187* (2019).

[17] Yang Liu, Yingting Liu, Zhijie Liu, Junbo Zhang, Chuishi Meng, and Yu Zheng. 2019. Federated Forest. *arXiv preprint arXiv:1905.10053* (2019).

[18] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*. 1273–1282.

[19] Jorge Nocedal and Stephen Wright. 2006. *Numerical optimization*. Springer Science & Business Media.

[20] Xia Shen, Moudud Alam, Freddy Fikse, and Lars Rönnegård. 2013. A novel generalized ridge regression method for quantitative genetics. *Genetics* 193, 4 (2013), 1255–1268.

[21] Johan AK Suykens and Joos Vandewalle. 1999. Least squares support vector machine classifiers. *Neural processing letters* 9, 3 (1999), 293–300.

[22] Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar, and Heiko Ludwig. 2019. Hybridalpha: An efficient approach for privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 13–23.

[23] Kai Yang, Tao Fan, Tianjian Chen, Yuanming Shi, and Qiang Yang. 2019. A Quasi-Newton Method Based Vertical Federated Learning Framework for Logistic Regression. *arXiv preprint arXiv:1912.00513* (2019).

[24] Kai Yang, Tao Jiang, Yuanming Shi, and Zhi Ding. 2020. Federated learning via over-the-air computation. *IEEE Transactions on Wireless Communications* 19, 3 (2020), 2022–2035.

[25] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 12.

[26] Xu Yang, Cheng Deng, Kun Wei, Junchi Yan, and Wei Liu. 2020. Adversarial Learning for Robust Deep Clustering. *Advances in Neural Information Processing Systems* 33 (2020).

[27] Gong-Duo Zhang, Shen-Yi Zhao, Hao Gao, and Wu-Jun Li. 2018. Feature-Distributed SVRG for High-Dimensional Linear Classification. *arXiv preprint arXiv:1802.03604* (2018).

[28] Qingsong Zhang, Bin Gu, Cheng Deng, and Heng Huang. 2021. Secure Bilevel Asynchronous Vertical Federated Learning with Backward Updating. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10896–10904.

[29] Qingsong Zhang, Feihu Huang, Cheng Deng, and Heng Huang. 2021. Faster Stochastic Quasi-Newton Methods. *IEEE TNNLS* (2021).

APPENDIX

In this Appendix, we only give the proof sketch of Theorems 1 and 2. For the detailed theoretical analyses of all Theorems 1 to 3, and the discussions on Assumptions 3.1 and 3.2 , one can refer to the arXiv version of this paper. For notation brevity, in the following proof, we let ℓ denote the corresponding party performing the updating at global iteration number u.

Proof of Theorem 1: Before proving Theorem 3, we first track the behavior of the gradient during an epoch in Lemma 2.

Lemma 2. *For both AsySQN-SGD and AsySQN-SVRG, we have that*

$$\sum_{u\in K(t)}\|\nabla_\ell f(w_u)\|^2 \geq \frac{1}{2}\sum_{u\in K(t)}\|\nabla_\ell f(w_t)\|^2 - \alpha\sum_{u\in K(t)}\sum_{v\in\{t,\ldots,u\}}\|\widehat{v}_v^{\psi(v)}\|^2, \tag{18}$$

where $\alpha=\eta_1\sigma_2^2\gamma^2L^2$

Proof. For any global iteration number $u\in K(t)$, we have that

$$\begin{aligned}
&\|\nabla_\ell f(w_t)\|^2 \leq 2\|\nabla_\ell f(w_t)-\nabla_\ell f(w_u)\|^2+2\|\nabla_\ell f(w_u)\|^2\\
&\overset{(a)}{\leq} 2L^2\|w_t-w_u\|^2+2\|\nabla_\ell f(w_u)\|^2\\
&=2L^2\gamma^2\|\sum_{v\in\{t,\ldots,u\}}\mathbf{U}_{\psi(v)}H_{\xi(v)}\widehat{v}_v^{\psi(v)}\|^2+2\|\nabla_\ell f(w_u)\|^2\\
&\overset{(b)}{\leq} 2\eta_1\sigma_2^2L^2\gamma^2\sum_{v\in\{t,\ldots,u\}}\|\widehat{v}_v^{\psi(v)}\|^2+2\|\nabla_\ell f(w_u)\|^2
\end{aligned}\tag{19}$$

where (a) uses Assumption 2.1, (b) uses Assumption 3.2. Rearranging Eq. (19) and summing over all $u\in K(t)$, we obtain the conclusion. This completes the proof. □

For any global iteration number $u\in K(t)$, we have that

$$\begin{aligned}
&\mathbb{E}f(w_{u+1})\\
&\overset{(a)}{\leq}\mathbb{E}\left(f(w_u)+\langle\nabla f(w_u),w_{u+1}-w_u\rangle+\frac{L_\ell}{2}\|w_{u+1}-w_u\|^2\right)\\
&\leq\mathbb{E}f(w_u)-\gamma\|H^\ell\|\mathbb{E}\|\nabla_\ell f(w_u)\|^2+\frac{\gamma\|H^\ell\|}{2}\mathbb{E}\|\nabla_\ell f(w_u)\|^2\\
&+\frac{\gamma\|H^\ell\|}{2}\mathbb{E}\|\nabla_\ell f_{\mathcal{I}_u}(w_u)-\nabla_\ell f_{\mathcal{I}_u}(\widehat{w}_u)\|^2+\frac{L_\ell\gamma^2\|H^\ell\|^2}{2}\mathbb{E}\|\widehat{v}_u^\ell\|^2\\
&\leq\mathbb{E}f(w_u)-\frac{\gamma\sigma_1}{2}\mathbb{E}\|\nabla_\ell f(w_u)\|^2+\frac{L_\ell\gamma^2\sigma_2^2}{2}\mathbb{E}\|\widehat{v}_u^\ell\|^2\\
&+\frac{\tau_1\sigma_2^3L^2\gamma^3}{2}\sum_{u'\in D(u)}\mathbb{E}\|\widehat{v}_{u'}^{\psi(u')}\|^2
\end{aligned}\tag{20}$$

where, using Assumption 1.2 one can obtain (a), using $\langle a,b\rangle\leq\frac{1}{2}(\|a\|^2+\|b\|^2)$, H^ℓ is positive semidefinite, Assumption 3.2 and $\mathbb{E}\|\nabla_\ell f_{\mathcal{I}_u}(w_u)-\nabla_\ell f_{\mathcal{I}_u}(\widehat{w}_u)\|^2\leq L^2\mathbb{E}\|w_u-\widehat{w}_u\|^2$ one can obtain the final inequality.

Summing (20) over all $u\in K(t)$, and using Lemma 2 (for (a)), Assumptions 4 and 5 (for (b)), and that

$$\mathbb{E}\|\widehat{v}_u^\ell\|^2=\frac{1}{|\mathcal{I}_u|^2}\sum_{i_u\in\mathcal{I}_u}\mathbb{E}\|\nabla_\ell f_{i_u}(\widehat{w}_u)\|^2\leq\frac{G}{|\mathcal{I}_u|}=\frac{G}{b},\tag{21}$$

we can obtain

$$\begin{aligned}
&\mathbb{E}f(w_{t+|K(t)|})-\mathbb{E}f(w_t)\\
&\leq-\frac{\gamma\sigma_1}{2}\sum_{u\in K(t)}\mathbb{E}\|\nabla_\ell f(w_u)\|^2+\frac{L_{\max}\sigma_2^2\gamma^2}{2}\sum_{u\in K(t)}\mathbb{E}\|\widehat{v}_u^\ell\|^2\\
&+\frac{\tau_1\sigma_2^3L^2\gamma^3}{2}\sum_{u\in K(t)}\sum_{u'\in D(u)}\mathbb{E}\|\widehat{v}_{u'}^{\psi(u')}\|^2\\
&\overset{(a)}{\leq}-\frac{\gamma\sigma_1}{2}\left(\frac{1}{2}\sum_{u\in K(t)}\|\nabla_\ell f(w_t)\|^2-\alpha\sum_{u\in K(t)}\sum_{v\in\{t,\ldots,u\}}\|\widehat{v}_v^{\psi(v)}\|^2\right)\\
&+\frac{\tau_1\sigma_2^3L^2\gamma^3}{2}\sum_{u\in K(t)}\sum_{u'\in D(u)}\mathbb{E}\|\widehat{v}_{u'}^{\psi(u')}\|^2+\frac{L_{\max}\sigma_2^2\gamma^2}{2}\sum_{u\in K(t)}\mathbb{E}\|\widehat{v}_u^\ell\|^2\\
&\overset{(b)}{\leq}-\frac{\gamma\mu\sigma_1}{2}\left(f(w_t)-f(w^*)\right)\\
&+\underbrace{\frac{\eta_1\sigma_2^2\gamma^2\left(\sigma_1\gamma L^2\eta_1^2+\sigma_2\gamma L^2\tau_1^2+L_{\max}\right)G}{2b}}_{C}
\end{aligned}\tag{22}$$

According to (22), we have that

$$\mathbb{E}f(w_{t+|K(t)|})-f(w^*)\leq\left(1-\frac{\gamma\mu\sigma_1}{2}\right)\left(f(w_t)-f(w^*)\right)+C\tag{23}$$

Assume that $\cup_{\kappa\in P(t)}=\{0,1,\ldots,t\}$, and applying Eq. (23), we have that

$$\begin{aligned}
&\mathbb{E}f(w_t)-f(w^*)\\
&\leq\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^{v(t)}\left(f(w_0)-f(w^*)\right)+C\sum_{i=0}^{v(t)}\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^i\\
&\leq\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^{v(t)}\left(f(w_0)-f(w^*)\right)+C\sum_{i=0}^{\infty}\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^i\\
&=\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^{v(t)}\left(f(w_0)-f(w^*)\right)\\
&+\frac{\gamma\eta_1\sigma_2^2\left(\sigma_1\gamma L^2\eta_1^2+\sigma_2\gamma L^2\tau_1^2+L_{\max}\right)G}{\mu\sigma_1 b}
\end{aligned}\tag{24}$$

Let $\frac{\gamma\eta_1\sigma_2^2(\sigma_1\gamma L^2\eta_1^2+\sigma_2\gamma L^2\tau_1^2+L_{\max})G}{\mu\sigma_1 b}\leq\frac{\epsilon}{2}$, we have that

$$\gamma\leq\frac{-L_{\max}+\sqrt{L_{\max}^2+\frac{2\mu\sigma_1b\epsilon(\sigma_1L^2\eta_1^2+\sigma_2\tau_1L^2\tau)}{G\eta_1\sigma_2^2}}}{2L^2(\sigma_1\eta_1^2+\sigma_2\tau_1^2)}.$$

Let $\left(1-\frac{\gamma\mu\sigma_1}{2}\right)^{v(t)}(f(w_0)-f(w^*))\leq\frac{\epsilon}{2}$, we have that

$$\log\left(\frac{2(f(w_0)-f(w^*))}{\epsilon}\right)\leq v(t)\log\left(\frac{1}{1-\frac{\gamma\mu\sigma_1}{2}}\right).$$

According to $\log\left(\frac{1}{\rho}\right)\geq 1-\rho$ for $0<\rho\leq 1$, we can rearrange above inequality and obtain the final result as in Theorem 1. This completes the proof.

Proof of Theorem 2: Before proving Theorem 4, we first provide Lemma 3 to provides an upper bound to $\mathbb{E}\left\|\widehat{v}_u^\ell\right\|^2$.

Lemma 3. *For AsySQN-SVRG, let $u \in K(t)$, we have that*

$$\begin{aligned}
&\mathbb{E}\left\|\widehat{v}_u^\ell\right\|^2 \qquad (25)\\
&\le \frac{16L^2}{\mu}\mathbb{E}(f(w_t^s)-f(w^*)) + 4L^2\gamma^2\sigma_2^2\eta_1 \sum_{v\in\{t,\ldots,u\}} \mathbb{E}\left\|\widehat{v}_v^{\psi(v)}\right\|^2\\
&+ \frac{8L^2}{\mu}\mathbb{E}(f(w^s)-f(w^*)) + 2\tau_1 L^2\gamma^2\sigma_2^2\mathbb{E}\sum_{u'\in D(u)}\left\|\widehat{v}_{u'}^{\psi(u')}\right\|^2
\end{aligned}$$

Define $v_u^\ell = \nabla_\ell f_{I_u}(w_u^s) - \nabla_\ell f_{I_u}(w^s) + \nabla_\ell f(w^s)$. We have that $\mathbb{E}\left\|\widehat{v}_u^\ell\right\|^2 = \mathbb{E}\left\|\widehat{v}_u^\ell - v_u^\ell + v_u^\ell\right\|^2 \le 2\mathbb{E}\left\|\widehat{v}_u^\ell - v_u^\ell\right\|^2 + 2\mathbb{E}\left\|v_u^\ell\right\|^2$. As for term $\mathbb{E}\left\|v_u^\ell\right\|^2$, we have that

$$\begin{aligned}
&\mathbb{E}\left\|v_u^\ell\right\|^2 = \mathbb{E}\left\|\nabla_\ell f_{I_u}\left(w_u^s\right) - \nabla_\ell f_{I_u}\left(w^s\right) + \nabla_\ell f\left(w^s\right)\right\|^2\\
\le& 2\mathbb{E}\left\|\nabla_\ell f_{I_u}\left(w^s\right) - \nabla_\ell f_{I_u}\left(w^*\right) - \nabla_\ell f\left(w^s\right) + \nabla_\ell f\left(w^*\right)\right\|^2\\
&+ 2\mathbb{E}\left\|\nabla_\ell f_{I_u}\left(w_u^s\right) - \nabla_\ell f_{I_u}\left(w^*\right)\right\|^2\\
\overset{(i)}{\le}& 2\mathbb{E}\left\|\nabla_\ell f_{I_u}\left(w_u^s\right) - \nabla_\ell f_{I_u}\left(w^*\right)\right\|^2\\
&+ 2\mathbb{E}\left\|\nabla_\ell f_{I_u}\left(w^s\right) - \nabla_\ell f_{I_u}\left(w^*\right)\right\|^2\\
\overset{(ii)}{\le}& 2L^2\mathbb{E}\left\|w_u^s - w^*\right\|^2 + 2L^2\mathbb{E}\left\|w^s - w^*\right\|^2\\
\le& 4L^2\mathbb{E}\|w_u^s - w_t^s\|^2 + 4L^2\mathbb{E}\|w_t^s - w^*\|^2 + 2L^2\mathbb{E}\left\|w^s - w^*\right\|^2\\
\overset{(iii)}{=}& 4L^2\gamma^2\mathbb{E}\|\sum_{v\in\{t,\ldots,u\}} H_{\xi(v)}\widehat{v}_v^{\psi(v)}\|^2\\
&+ 4L^2\mathbb{E}\|w_t^s - w^*\|^2 + 2L^2\mathbb{E}\left\|w^s - w^*\right\|^2\\
\overset{(iv)}{\le}& \frac{8L^2}{\mu}\mathbb{E}(f(w_t^s)-f(w^*)) + \frac{4L^2}{\mu}\mathbb{E}(f(w^s)-f(w^*))\\
&+ 4L^2\gamma^2\sigma_2^2\eta_1\sum_{v\in\{t,\ldots,u\}}\mathbb{E}\|\widehat{v}_v^{\psi(v)}\|^2 \qquad (26)
\end{aligned}$$

where the (i) uses $\mathbb{E}\|x-\mathbb{E}x\|^2 \le \mathbb{E}\|x\|^2$, (ii) follows Eq. (22), (iii) uses Definition 2, and (iv) uses Assumption 1. Using $\|\sum_{i=1}^n a_i\|^2 \le n\sum_{i=1}^n \mathbb{E}\|a_i\|^2$, Definition 2, Assumption 4, and then we can bound $\mathbb{E}\left\|\widehat{v}_u^\ell - v_u^\ell\right\|^2$ as follows.

$$\mathbb{E}\left\|\widehat{v}_u^\ell - v_u^\ell\right\|^2 \le \tau_1 L^2\sigma_2^2\gamma^2\mathbb{E}\sum_{u'\in D(u)}\left\|\widehat{v}_{u'}^{\psi(u')}\right\|^2, \qquad (27)$$

Combining $\mathbb{E}\left\|\widehat{v}_u^\ell\right\|^2 \le 2\mathbb{E}\left\|\widehat{v}_u^\ell - v_u^\ell\right\|^2 + 2\mathbb{E}\left\|v_u^\ell\right\|^2$ with Eqs. (26) to (27), we have the final result as in Eq. 25. Similar to the proof of (20), for $u \in K(t)$ at s-th outer loop, we have

$$\begin{aligned}
&\mathbb{E}f(w_{u+1}^s) \qquad (28)\\
&\le \mathbb{E}(f(w_u^s) + \langle\nabla f(w_u^s), w_{u+1}^s - w_u^s\rangle + \frac{L_\ell}{2}\|w_{u+1}^s - w_u^s\|^2)\\
&\le \mathbb{E}f(w_u^s) - \gamma\mathbb{E}\langle\nabla f(w_u^s), H^\ell\nabla_\ell f_{I_u}(\widehat{w}_u^s) - H^\ell\nabla_\ell f_{I_u}(w_u^s)\\
&+ H^\ell\nabla_\ell f_{I_u}(w_u^s)\rangle + \frac{L_\ell\gamma^2\|H^\ell\|^2}{2}\mathbb{E}\|\widehat{v}_u^\ell\|^2\\
\le& \mathbb{E}f(w_u^s) - \frac{\gamma\sigma_1}{2}\mathbb{E}\|\nabla_\ell f(w_u^s)\|^2 + \frac{\tau_1\sigma_2^3L^2\gamma^3}{2}\sum_{u'\in D(u)}\mathbb{E}\|\widehat{v}_{u'}^{\psi(u')}\|^2\\
&+ \frac{L_\ell\gamma^2\sigma_2^2}{2}\mathbb{E}\|\widehat{v}_u^\ell\|^2 \qquad (29)
\end{aligned}$$

Summing Eq. (28) over all $u \in K(t)$, and using Lemma 2, Assumption 5, and Lemma 3, we have

$$\begin{aligned}
&\mathbb{E}f(w_{t+|K(t)|}^s) - \mathbb{E}f(w_t^s) \le -\frac{\gamma\mu\sigma_1}{2}\mathbb{E}(f(w_t^s) - f(w^*)) \qquad (30)\\
&+ \frac{\tau_1\sigma_2^3L^2\gamma^3}{2}\sum_{u\in K(t)}\sum_{u'\in D(u)}\mathbb{E}\|\widehat{v}_{u'}^{\psi(u')}\|^2 + C\\
&\cdot\sum_{u\in K(t)}\left(\frac{16L^2}{\mu}\mathbb{E}(f(w_t^s)-f(w^*)) + \frac{8L^2}{\mu}\mathbb{E}(f(w^s)-f(w^*))\right.\\
&\left.+4\alpha\sum_{v\in\{t,\ldots,u\}}\mathbb{E}\left\|\widehat{v}_v^{\psi(v)}\right\|^2 + 2\tau_1L^2\gamma^2\sigma_2^2\mathbb{E}\sum_{u'\in D(u)}\left\|\widehat{v}_{u'}^{\psi(u')}\right\|^2\right)
\end{aligned}$$

Let $e_t^s = \mathbb{E}(f(w_t^s)-f(w^*))$ and $e^s = \mathbb{E}(f(w^s)-f(w^*))$, we have

$$\begin{aligned}
e_{t+|K(t)|}^s \le& \left(1 - \frac{\gamma\mu\sigma_1}{2} + \frac{16L^2\eta_1C}{\mu}\right)e_t^s + \frac{8L^2\eta_1C}{\mu}e^s \qquad (31)\\
&+ \gamma^3\left(\left(\frac{\sigma_2}{2} + \frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\eta_1L^2\sigma_2^2\frac{9G}{b}
\end{aligned}$$

We carefully choose γ such that $1 > \frac{\gamma\mu}{2} - \frac{16L^2\eta_1C}{\mu} \overset{\text{def}}{=} \rho > 0$. Assume that $\cup_{\kappa\in P(t)} = \{0,1,\ldots,t\}$, applying (31), we have that

$$\begin{aligned}
e_t^s \le& (1-\rho)^{\upsilon(t)}e^s + \left(\frac{8L^2\eta_1C}{\mu}e^s\right)\sum_{i=0}^{\upsilon(t)}(1-\rho)^i \qquad (32)\\
&+ (\gamma^3\left(\left(\frac{\sigma_2}{2} + \frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\eta_1qL^2\sigma_2^2\frac{9G}{b})\sum_{i=0}^{\upsilon(t)}(1-\rho)^i\\
\le& (1-\rho)^{\upsilon(t)}e^s + \left(\frac{8L^2\eta_1C}{\mu}e^s\right)\frac{1}{\rho}\\
&+ (\gamma^3\left(\left(\frac{\sigma_2}{2} + \frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\eta_1qL^2\sigma_2^2\frac{9G}{b})\frac{1}{\rho}\\
=& \left((1-\rho)^{\upsilon(t)} + \frac{8L^2\eta_1C}{\rho\mu}\right)e^s\\
&+ \gamma^3\left(\left(\frac{\sigma_2}{2} + \frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\frac{9\eta_1L^2\sigma_2^2G}{b\rho}
\end{aligned}$$

Thus, to achieve $\mathbb{E}f(w_S) - f(w^*) \le \epsilon$, we can carefully choose γ such that 1) $\gamma^3\left(\left(\frac{\sigma_2}{2} + \frac{2C}{\gamma}\right)\tau_1^2 + 4\frac{C}{\gamma}\eta_1^2\right)\frac{9\eta_1L^2\sigma_2^2G}{b\rho} \le \frac{\epsilon}{8}$ and 2) $\frac{8L^2\eta_1C}{\rho\mu} \le 0.5$; and let $(1-\rho)^{\upsilon(t)} \le 0.25$, i.e., $\upsilon(t) \ge \frac{\log 0.25}{\log(1-\rho)}$, we have that

$$e^{s+1} \le 0.75e^s + \frac{\epsilon}{8} \qquad (33)$$

Recursively apply (33), we have that

$$e^S \le (0.75)^S e^0 + \frac{\epsilon}{2}$$

Finally, we use that the outer loop number S should satisfy the condition of $S \ge \frac{\log\frac{2e^0}{\epsilon}}{\log\frac{4}{3}}$ and can obtain the desired result in Theorem 2. This completes the proof.