MinMax Sampling: A Near-optimal Global Summary for Aggregation in the Wide Area

Yikai Zhao[†] Peking University
Yinda Zhang[†] Peking University
Yuanpeng Li[†] Peking University
Yi Zhou[†] Peking University
Chunhui Chen[§] Peking University
Tong Yang[†‡] Peking University
Bin Cui[†] Peking University

ABSTRACT

[*]Nowadays, wide-area data analyses are pervasive with emerging geo-distributed systems. These analyses often need to do the *global aggregation* in the wide area. Since scarce and variable WAN bandwidth may degrade the aggregation performance, it is highly desired to design a communication scheme for global aggregation in WAN. Unfortunately, no existing algorithm can meet the three design requirements of communication schemes: fast computation, adaptive transmission, and accurate aggregation. In this paper, we propose MinMax Sampling, a fast, adaptive, and accurate communication scheme for global aggregation in WAN. We first focus on the accuracy and design a scheme, namely $\text{MinMax}_{\text{opt}}$, to achieve optimal accuracy. However, $\text{MinMax}_{\text{opt}}$ does not meet the other two requirements: fast computation and adaptive transmission. Based on $\text{MinMax}_{\text{opt}}$, we propose $\text{MinMax}_{\text{adp}}$, which trades little accuracy for the other two requirements. We evaluate $\text{MinMax}_{\text{adp}}$ with three applications: federated learning, distributed state aggregation, and hierarchical aggregation. Our experimental results show that $\text{MinMax}_{\text{adp}}$ is superior to existing algorithms (8.44× better accuracy on average) in all three applications. The source codes of MinMax Sampling are available at Github [1].

CCS CONCEPTS

• **Theory of computation → Sketching and sampling**; • **Networks → Wide area networks**.

KEYWORDS

Sampling; Wide-Area Network; Federated Learning

[†]School of Computer Science, and National Engineering Laboratory for Big Data Analysis Technology and Application, Peking University, China.
[‡]Peng Cheng Laboratory, Shenzhen, China.
[§]School of Software & Microelectrionics, Peking University, China.
[*]Yikai Zhao, Yinda Zhang, and Yuanpeng Li contribute equally to this paper. Tong Yang (yangtongemail@gmail.com) is the corresponding author. This work is supported by Key-Area Research and Development Program of Guangdong Province 2020B0101390001, National Natural Science Foundation of China (NSFC) (No. U20A20179, 61832001).

SIGMOD '22, June 12–17, 2022, Philadelphia, PA, USA.

ACM ISBN 978-1-4503-9249-5/22/06…$15.00
https://doi.org/10.1145/3514221.3526160

ACM Reference Format:
Yikai Zhao, Yinda Zhang, Yuanpeng Li, Yi Zhou, Chunhui Chen, Tong Yang, and Bin Cui. 2022. MinMax Sampling: A Near-optimal Global Summary for Aggregation in the Wide Area. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3514221.3526160

1 INTRODUCTION

1.1 Background and Motivation

Nowadays, wide-area data analyses are pervasive with emerging geo-distributed systems. CDNs (Content Distribution Networks) [2] and other computing infrastructures [3–5] often generate a large volume of data on geographically distributed devices across the globe. To get the global state to support better global synchronization [6], runtime monitoring [7], fault detection [8, 9], *etc.* [10, 11], these applications often need to do the **global aggregation** in the wide area. Specifically, applications should collect the states from all local devices and aggregate them on the central device to get the global state. In this paper, we focus on such a global aggregation problem: Formally, a distributed system includes a central device and many local devices. Each local device maintains a local value vector V_i. Global aggregation aims to obtain the global value vector $V = \sum V_i$, the sum of all local value vectors, on the central device. Global aggregation is a common problem, and we show three typical cases in the following.

Federated Learning: Federated learning [12] is an important machine learning setting, in which training data are distributed on many local devices. Federated learning uses a *Parameter Server* to collect local training gradients from multiple devices to obtain the global gradient [13]. In this case, the gradient on each local device can be regarded as V_i, and the global gradient can be regarded as V.

Distributed State Aggregation: Distributed state aggregation is necessary for many distributed data services, including multi-region databases [7, 14–16] and CDNs [9]. For these services, typically each local server maintains some local states (*e.g.*, system event logs that record the number of occurrences of each event, object access logs that record the number of requests for each object [17]), and operators need to collect the local states from all servers to monitor the services in a global view. In these cases, the local state on each server can be represented as V_i, and the global state of the whole service can be represented as V.

Hierarchical Aggregation: Hierarchical aggregation usually occurs in networks with a hierarchical structure, such as sensor networks organized as rooted spanning trees [10, 11], and large-scale

WAN (Wide-Area Network) composed of multiple AS (Autonomous System) domains [18]. In these cases, the global value vector in one level[1] of aggregation may be regarded as the local value vector in the next level of aggregation.

Since local devices often connect to the central device through the wireless network or WAN [19], scarce and variable bandwidth may degrade the aggregation performance. For federated learning, the slow gradient aggregation speed will prolong the time of model training [20, 21]; for distributed state aggregation, the slow state aggregation speed will constrain the granularity of operators monitoring the whole service, and increase the response delay of fault events [22]. Since the network has become the bottleneck in the global aggregation, it is highly desired to design a communication scheme for global aggregation in WAN. Specifically, a communication scheme generates an approximate local summary $\hat{V}_i$ for the local value vector V_i on each local device, transmits $\hat{V}_i$ instead of V_i to the central device, and computes a global summary $\hat{V}$ on the central device through all $\hat{V}_i$ to approximate the global value vector V. There are three most important design requirements of the communication scheme as follows.

- **Fast** for computation on local devices. Local devices often have limited computation resources compared to the central device [23]. If the scheme employs complex procedures to generate summaries on local devices, the end-to-end communication delay will be prolonged. Therefore, the procedure to generate the summary should be fast.
- **Adaptive** for transmission over the network. The available bandwidth between distributed devices and the central device is often scarce and varies a lot over space and time. As shown in prior works [6], the WAN bandwidth capacity is 15× smaller than their LAN bandwidth on average, and up to 60× smaller in the worst case. Therefore, the scheme should allow different local devices to customize and dynamically change the size of transmitted local summaries according to their own available bandwidth.
- **Accurate** for aggregation on the central device. Besides the scarce and varied bandwidth, the massive number of local devices is also a challenge to accurately estimate the global value vector on the central device. Although each local summary may bring only a small error to the local value vector, the errors could be accumulated and cause a great disturbance to the global value vector after aggregation. Therefore, the scheme should accurately estimate each global value in the global value vector after aggregating a massive number of local summaries from local devices.

1.2 Limitations of Prior Art

Existing communication schemes for global aggregation can be divided into three categories: compression-based schemes, sketch-based schemes, and sampling-based schemes.

Compression-based schemes: Lossless compression algorithms [24] allow perfect reconstruction of original data after compression. For these algorithms, the requirement of lossless compression brings a large computational burden. Further, the compression ratio cannot be flexibly customized, and thus cannot achieve adaptive transmission.

Sketch-based scheme: These communication schemes use a kind of probabilistic data structure named sketch[2] to approximately record the local value vector. As shown in prior works [30], the sketch can approximate the local value vector with high accuracy on a single device. Prior works often utilize the mergeability of existing sketches to aggregate the data. However, the mergeability of sketches requires the size of sketches on all devices to be the same, which prevents them from the adaptive transmission.

Sampling-based scheme: These communication schemes choose a part of values from the local value vector for transmission. Typical sampling algorithms for global aggregation include H-sampling [31], Iceberg sampling [9], and top-K sampling [32, 33]. H-sampling and Iceberg sampling are unbiased. H-sampling optimizes the global transmission cost under a given error bound, and Iceberg sampling achieves stable accuracy against variable distribution. However, both of them require all local devices sharing the same configuration, and thus are unable to customize the size of transmitted local summary according to available bandwidth. Iceberg sampling can force different devices to use different configurations at the expense of sharp accuracy degradation, while H-sampling does not offer this option. Top-K sampling is a typical biased sampling. Its error will increase linearly with the number of local devices, so its accuracy is poor when there are a massive number of devices.

In summary, no existing algorithm can meet all three design requirements of global aggregation communication schemes. In this paper, we aim to propose a communication scheme that can meet these requirements at the same time.

1.3 Our Proposed Algorithm

In this paper, we propose MinMax Sampling, a fast, adaptive, and accurate communication scheme for global aggregation in WAN. 1) **Fast:** MinMax Sampling generates a local summary in linear time. It can generate a summary for the local value vector containing up to 60M values in one second using one CPU core. 2) **Adaptive:** MinMax Sampling can generate the local summary of any desired size. Further, it can modify the size transmitted on the fly, *i.e.*, it can transmit a growing local summary until a predetermined time is reached or a termination signal is received. It can also work when the values are streaming generated. 3) **Accurate:** MinMax Sampling can generate a global summary to estimate the global value vector unbiased. Compared to the state-of-the-art, its accuracy is 8.44× higher on average. Further, accurate global aggregation can help to improve the accuracy of federated learning by up to 7.73%.

To achieve all three requirements, we first focus on the accuracy and design the optimal MinMax Sampling ($\text{MinMax}_{\text{opt}}$) to achieve optimal accuracy. To address the challenge of a massive number of devices, the sampling scheme should be unbiased. Assuming that there are n local devices, using biased sampling, the estimation error of the global value vector is $O(n)$, *i.e.*, it increases linearly with the number of devices. Using unbiased schemes, the overestimation and underestimation of different devices can offset each other, and the error is $O(\sqrt{n})$, *i.e.*, it increases sub-linearly. For unbiased sampling, reducing the estimation error of each global value in the global value vector is equivalent to minimizing the maximum variance of

[1]The devices in the hierarchical network are divided into multiple levels, and the global aggregation is level-by-level [11].

[2]Sketch is a kind of compact data structure using linear projection. Typical sketches include the Count-Min sketch [25], the Count sketch [26], *etc.* [27–29].

Table 1: Notation

Symbol	Meaning
n	The number of local devices
V_i	The local value vector in local device i
$\widehat{V_i}$	The estimated value vector of V_i
V	The global value vector, $V = \sum_i V_i$
$\widehat{V}$	The estimated value vector of V
m	The length of value vectors
$v_{i,j}$	The j-th value in V_i
$\widehat{v}_{i,j}$	The estimated value of $v_{i,j}$
$p_{i,j}$	The sampling probability of $v_{i,j}$
$I_{i,j}$	The indicator of whether $v_{i,j}$ is sampled
$q_{i,j}$	The sampling priority of $v_{i,j}$
$r_{i,j}$	The random number in range $(0, 1)$ for $v_{i,j}$
C_i	The parameter in MinMax Sampling in device i
K_i	The expected number of values transmitted in device i

all estimated values. Therefore, we propose MinMax$_{\text{opt}}$, an unbiased sampling scheme that can minimize the maximum variance of all values, and we prove its optimality through theoretical analysis. However, MinMax$_{\text{opt}}$ does not meet the other two requirements: fast computation and adaptive transmission.

Based on MinMax$_{\text{opt}}$, we propose adaptive MinMax Sampling (MinMax$_{\text{adp}}$), which trades little accuracy for the other two requirements. MinMax$_{\text{opt}}$ independently determines whether each value in the local value vector should be sampled according to a probability. It can only guarantee the expected number of sampled values, but the actual size of the local summary is variable and uncontrollable. Differently, MinMax$_{\text{adp}}$ calculates a sampling priority for each value using a stochastic formula and then takes the K values with the largest priority as the local summary, where K can be customized and dynamically changed. Therefore, MinMax$_{\text{adp}}$ can flexibly control the size of the local summary. Because the K-th priority can be selected in linear time complexity, we reduce the time complexity from $O(m\log(1/\varepsilon))$ of MinMax$_{\text{opt}}$ to $O(m)$ of MinMax$_{\text{adp}}$, where m is the size of local value vector, and ε is the accuracy of calculating probability. As for the accuracy of MinMax$_{\text{adp}}$, we prove that it is unbiased, and its variance is near-optimal. The experimental results show that the variance gap between MinMax$_{\text{opt}}$ and MinMax$_{\text{adp}}$ is within 0.05%.

1.4 Main Contribution

- We propose MinMax Sampling, which contains two versions. MinMax$_{\text{opt}}$ achieves optimal accuracy, while MinMax$_{\text{adp}}$ meets the requirements of fast computation and adaptive transmission with near-optimal accuracy.
- We make abundant theoretical analyses for both MinMax$_{\text{opt}}$ and MinMax$_{\text{adp}}$, prove their unbiasedness and optimality/near-optimality, and provide error bounds.
- We evaluate MinMax$_{\text{adp}}$ in three applications: Federated learning, Distributed State Aggregation, and Hierarchical Aggregation. Our experimental results show MinMax$_{\text{adp}}$ is superior to existing algorithms (8.44× better accuracy on average) in all three applications.

2 PRELIMINARY

In this section, we first present a formal definition of the global aggregation problem. Then, because our solution is a sampling-based communication scheme, we show preliminary notations and definitions in the sampling model and formalize three requirements in such a sampling model. For convenience, we list symbols frequently used in this paper and their meanings in Table 1.

Definition 2.1. (**Global value vector**) Given n local devices, each local device i contains a value vector

$$V_i = \left\langle v_{i,1}, v_{i,2}, \cdots, v_{i,m} \right\rangle,$$

where each value $v_{i,j}$ corresponds to the j-th local value in device i. The global value vector

$$V = \sum_{i=1}^{n} V_i = \left\langle \sum_{i=1}^{n} v_{i,1}, \cdots, \sum_{i=1}^{n} v_{i,m} \right\rangle$$

A communication scheme includes an encoding procedure for local devices and a decoding procedure for the central device. In one communication, each local device i calls the encoding procedure to generate an approximate summary $\widehat{V_i}$ of the local value vector V_i. The central device collects all local summaries and calls the decoding procedure to generate a global summary/estimation $\widehat{V}$ based on the summary of $\widehat{V_i}$ of each device i.

Sampling model: Given a value vector $V_i = \left\langle v_{i,1}, \cdots, v_{i,m} \right\rangle$ in device i, a random sampling scheme is to generate a random variable vector $\widehat{V_i} = \left\langle \widehat{v}_{i,1}, \cdots, \widehat{v}_{i,m} \right\rangle$, where $\widehat{v}_{i,j}$ is called the *estimated value* of $v_{i,j}$. We use $I_{i,j}$ to indicate whether $v_{i,j}$ is sampled or not, where

$$I_{i,j} = \begin{cases} 0, & \widehat{v}_{i,j} = 0 \\ 1, & \widehat{v}_{i,j} \neq 0 \end{cases}. \tag{1}$$

We call value $v_{i,j}$ sampled when $I_{i,j} = 1$, and vice versa. The sampling-based communication scheme uses sampling as the encoding procedure, and transmits the $\widehat{v}_{i,j}$ of the sampled values and the corresponding indexes. Let $p_{i,j} = \Pr[I_{i,j} = 1]$ be the sampling probability of value $v_{i,j}$. The transmission cost of device i, *i.e.*, the number of values sampled, is $\sum_{j=1}^{m} I_{i,j}$, and it has a mathematical expectation with a value of $\sum_{j=1}^{m} p_{i,j}$.

Formalization of requirements: For a communication scheme based on sampling, we formalize the three design requirements as follows:

- **Fast:** Given a local value vector with m values, the time complexity of the sampling scheme should be $O(m)$. In other words, we should finish the sampling in a constant time of traversing all non-zero values.
- **Adaptive:** Ideally, given any integer K_i, the sampling scheme in device i should strictly sample K_i values, *i.e.*, $\sum_{j=1}^{m} I_{i,j} = K_i$. The varying number of sampled values will lead to insufficient use of available bandwidth ($< K_i$), or excessive transmission ($> K_i$). Further, the sampling scheme should be able to dynamically adjust the number of sampled values to cope with the WAN bandwidth that varies a lot over time.
- **Accurate:** 1) The sampling scheme should be unbiased, *i.e.*, for any i and j, $E[\widehat{v}_{i,j}] = v_{i,j}$. The bias will grow linearly as the number of devices increasing, while the unbiased sampling scheme is scalable due to *the law of large numbers*. Compared with the

biased scheme, the more devices, the more accurately the unbiased sampling perform. 2) The sampling scheme of each device i should minimize the maximum variance of all estimated values, *i.e.*, $\min \max_{j=1}^{n} D[\widehat{v}_{i,j}]$.

3 MINMAX SAMPLING

In this section, we first design MinMax$_{\text{opt}}$ which can achieve optimal accuracy. Then, we propose MinMax$_{\text{adp}}$ based on MinMax$_{\text{opt}}$, which trades off little accuracy to be both fast and adaptive. Finally, we propose *outlier elimination*, an optimization that works on the central device to achieve more accurate aggregation.

3.1 Optimal Version

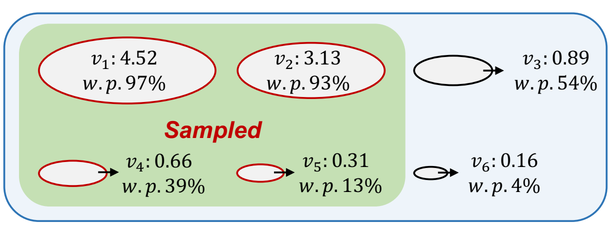
Sampling process of MinMax$_{\text{opt}}$: In any device i, given local value vector V_i and constant C_i, MinMax$_{\text{opt}}$ sets the sampling probability $p_{i,j}$ of $v_{i,j}$ as

$$p_{i,j} = \frac{v_{i,j}^2}{v_{i,j}^2 + C_i}, \tag{2}$$

If $v_{i,j}$ is sampled, the estimated value $\widehat{v}_{i,j}$ is set to $\frac{v_{i,j}}{p_{i,j}}$ to achieve unbiasedness. To set the expected sample size $\sum_{j=1}^{m} p_{i,j} = K_i$, we can set C_i as the solution of the following equation

$$\sum_{j=1}^{m} \frac{v_{i,j}^2}{v_{i,j}^2 + C_i} = K_i. \tag{3}$$

Example of MinMax$_{\text{opt}}$ (Figure 1): In this example, the local vector vector $V_i = \langle 4.52, 3.13, 0.89.0.66, 0.31, 0.16\rangle$, and the expected number of sampled values is $K_i = 3$. MinMax$_{\text{opt}}$ first obtains the constant $C_i = 0.68$ according to Formula 3, and calculates the sampling probability being $\langle 0.97, 0.93, 0.54, 0.39, 0.13, 0.04\rangle$ according to Formula 2. As shown in the figure, MinMax$_{\text{opt}}$ samples 4 values $\{v_{i,1} : 4.51, v_{i,2} : 3.13, v_{i,4} : 0.66, v_{i,5} : 0.31\}$, and obtains that the estimated values of all values are $\langle 4.67, 3.35, 0, 1.69, 2.48, 0\rangle$.



$$p_j = \frac{v_j^2}{v_j^2 + C} \qquad C = \arg_C\left(\sum p_j = 3\right) = 0.68$$

Figure 1: Example of MinMax$_{\text{opt}}$.

Then we analyze some properties of the MinMax$_{\text{opt}}$. We first show the variance $D[\widehat{v}_{i,j}]$ of each estimated value $\widehat{v}_{i,j}$.

Theorem 3.1.

$$D[\widehat{v}_{i,j}] = \begin{cases} 0, & v_{i,j} = 0 \\ C_i, & v_{i,j} \neq 0 \end{cases}. \tag{4}$$

Proof. Based on the definition of $p_{i,j}$, if $v_{i,j} = 0$, $p_{i,j} = 0$ and $\widehat{v}_{i,j}$ is always 0. If $v_{i,j} \neq 0$,

$$D[\widehat{v}_{i,j}] = p_{i,j} \cdot \left(\frac{v_{i,j}}{p_{i,j}} - v_{i,j}\right)^2 + (1 - p_{i,j}) \cdot v_{i,j}^2 = C_i$$

□

Variance of MinMax$_{\text{opt}}$ (Theorem 3.1): This theorem shows the fairness of the variance in MinMax$_{\text{opt}}$, *i.e.*, for all non-zero local values, their variances are all C_i. In addition, such variance is friendly to some applications where the value vector is sparse (*i.e.*, most values in the vector is 0), because the MinMax$_{\text{opt}}$ will not bring additional error to their estimated value. Note that for prior sketch-based schemes [26], even if the local value is 0, there is still an additional error in their estimated value. Then, we will show the bound of C_i.

Theorem 3.2. *Given the expected sample size K_i,*

$$\frac{\|V_i\|_2^2}{K_i} - \max_{j=1}^{m} v_{i,j}^2 \leqslant C_i \leqslant \frac{\|V_i\|_2^2}{K_i} - \min_{j=1}^{m} v_{i,j}^2, \tag{5}$$

where $\|V_i\|_2^2 = \sum_{j=1}^{m} v_{i,j}^2$.

Proof. Let

$$C_1 = \frac{\|V_i\|_2^2}{K_i} - \max_{j=1}^{m} v_{i,j}^2$$

$$\sum_{j=1}^{m} \frac{v_{i,j}^2}{v_{i,j}^2 + C_1} \geqslant \sum_{j=1}^{m} \frac{K_i \cdot v_{i,j}^2}{\|V_i\|_2^2} = K_i$$

Similarly, let

$$C_2 = \frac{\|V_i\|_2^2}{K_i} - \min_{j=1}^{m} v_{i,j}^2$$

$$\sum_{j=1}^{m} \frac{v_{i,j}^2}{v_{i,j}^2 + C_2} \leqslant \sum_{j=1}^{m} \frac{K_i \cdot v_{i,j}^2}{\|V_i\|_2^2} = K_i.$$

In view of the monotonicity of the Formula with respect to C_i, we can obtain

$$\frac{\|V_i\|_2^2}{K_i} - \max_{j=1}^{m} v_{i,j}^2 \leqslant C_i \leqslant \frac{\|V_i\|_2^2}{K_i} - \min_{j=1}^{m} v_{i,j}^2.$$

□

Range of variance (Theorem 3.2): This theorem bounds the variance of the estimated value, and shows the relationship between the variance and the value vector. Although we cannot directly show the explicit formula of C_i, we can prove the range it will lie in. The variance of MinMax$_{\text{opt}}$ can be bounded by the second norm of the value vector.

Afterward, we show that the accuracy of our MinMax$_{\text{opt}}$ is optimal. We start with a special sampling scheme, poisson sampling.

Definition 3.3. (**Poisson sampling**) In poisson sampling, each estimated value $\widehat{v}_{i,j}$ is defined as a random variable with only two values, namely

$$\widehat{v}_{i,j} = \begin{cases} \frac{v_{i,j}}{p_{i,j}}, & \text{w.p.} \quad p_{i,j} \\ 0, & \text{w.p.} \quad 1 - p_{i,j} \end{cases}. \tag{6}$$

In other words, each value $v_{i,j}$ has the probability of $p_{i,j}$ being sampled with the estimated value of $\frac{v_{i,j}}{p_{i,j}}$.

The following Lemma 3.4 shows the optimality of poisson sampling.

Lemma 3.4. *For any unbiased sampling $\mathcal{S}$, there must be a poisson sampling $\mathcal{S}'$ such that for any value $v_{i,j}$, there is $D[\widehat{v}_{i,j} \mid \mathcal{S}'] \leqslant D[\widehat{v}_{i,j} \mid \mathcal{S}]$, where $D[\widehat{v}_{i,j} \mid \mathcal{S}/\mathcal{S}']$ is the variance of the estimated value $\widehat{v}_{i,j}$ under the sampling $\mathcal{S}/\mathcal{S}'$.*

Proof. For any $\mathcal{S}$, we first use $\Pr[I_{i,j} = 1 \mid \mathcal{S}]$ to construct $p_{i,j}$ of poisson Sampling $\mathcal{S}'$.

$$D[\widehat{v}_{i,j} \mid \mathcal{S}] - D[\widehat{v}_{i,j} \mid \mathcal{S}'] = E[\widehat{v}^2_{i,j} \mid \mathcal{S}] - E[\widehat{v}^2_{i,j} \mid \mathcal{S}']$$

Because $\widehat{v}^2_{i,j}$ is always 0 if $I_{i,j} = 0$, we only need to consider the case where $I_{i,j} = 1$.

$$\begin{aligned}
&E[\widehat{v}^2_{i,j} \mid \mathcal{S}, I_{i,j} = 1] - E[\widehat{v}^2_{i,j} \mid \mathcal{S}', I_{i,j} = 1] \\
=&E[\widehat{v}^2_{i,j} \mid \mathcal{S}, I_{i,j} = 1] - \left(\frac{v_{i,j}}{p_{i,j}}\right)^2 \\
=&E[\widehat{v}^2_{i,j} \mid \mathcal{S}, I_{i,j} = 1] - E^2[\widehat{v}_{i,j} \mid \mathcal{S}, I_{i,j} = 1] \\
=&D[\widehat{v}_{i,j} \mid \mathcal{S}, I_{i,j} = 1] \\
\geqslant&0.
\end{aligned}$$

Finally, we have $D[\widehat{v}_{i,j} \mid \mathcal{S}] \geqslant D[\widehat{v}_{i,j} \mid \mathcal{S}']$ □

Note that $\text{MinMax}_{\text{opt}}$ belongs to poisson sampling, and then we prove the optimality of $\text{MinMax}_{\text{opt}}$.

Theorem 3.5. *Given the expected sample size K_i, the sampling scheme $\mathit{MinMax}_{\mathit{opt}}$ can minimize the maximum variance.*

Proof. We first prove that $\text{MinMax}_{\text{opt}}$ is the optimal poisson sampling to minimize the maximum variance. In a sampling scheme $\mathcal{S}$, restricting the maximum variance of all estimated values smaller than C_i is actually equivalent to restricting the variance of each estimated value smaller than C_i. Suppose that in poisson sampling scheme $\mathcal{S}$, for all value $v_{i,j}$, there is $D[\widehat{v}_{i,j} \mid \mathcal{S}] < C_i$, so

$$D\left[\widehat{v}_{i,j} \mid \mathcal{S}\right] = \left(1 - \frac{1}{p_{i,j}}\right) v^2_{i,j} < C_i.$$

In order to make it hold, we need to set each sampling probability

$$p_{i,j} > \frac{v^2_{i,j}}{v^2_{i,j} + C_i}.$$

In other words, under the sampling scheme $\mathcal{S}$, the expected sample size $\sum_{j=1}^{m} p_{i,j} > K_i$. This means that our sampling scheme $\text{MinMax}_{\text{opt}}$ is the optimal poisson sampling when the expected sample size is K_i.

Based on Theorem 3.4, we can get that if there is an unbiased sampling scheme $\mathcal{S}'$ which can achieve maximum variance $C'_i < C_i$, there must be a poisson sampling scheme which can achieve maximum variance $C''_i \leqslant C'_i < C_i$. It contradicts that $\text{MinMax}_{\text{opt}}$ is the optimal poisson sampling when the expected sample size is K_i. Therefore, $\text{MinMax}_{\text{opt}}$ is the optimal unbiased sampling when the expected sample size is K_i. □

Optimality of $\text{MinMax}_{\text{opt}}$ (Theorem 3.5): This theorem shows the optimality of $\text{MinMax}_{\text{opt}}$. $\text{MinMax}_{\text{opt}}$ can guarantee that the variance of any estimated value is not larger than C_i. Note that none of the prior sampling schemes is designed for minimizing the maximum variance. Therefore, the $\text{MinMax}_{\text{opt}}$ is the first optimal sampling scheme that can achieve minimal maximum variance.

Limitations of $\text{MinMax}_{\text{opt}}$: Note that although the accuracy of $\text{MinMax}_{\text{opt}}$ is optimal, it is unable to be fast and adaptive.

- **Not Fast enough:** The computational bottleneck of $\text{MinMax}_{\text{opt}}$ is to calculate the constant C by solving Equation 3. A strawman approach is to use binary search, whose time complexity is $O(m \log \frac{1}{\varepsilon})$, where ε is the precision of the solution. As shown in Table 2, in $\text{MinMax}_{\text{opt}}$, calculating C takes about 85% of the time.
- **Not Adaptive enough:** $\text{MinMax}_{\text{opt}}$ can only guarantee that the expected number of samples is K_i, but the actual number of samples is variable. For many local devices in a distributed system, the communication bottleneck is often the slowest one. As shown in Figure 2, although we set the K_i of all devices to 100, when the number of devices exceeds 2000, the most unfortunate device will transmit 30% more. The excessive transmission will lead to packet loss in WAN, resulting in performance degradation [34].

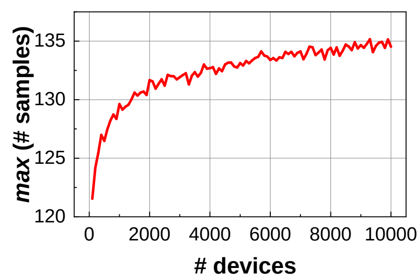


Figure 2: The variation of the maximum number of sampled values with the number of local devices.

3.2 Adaptive Version

To achieve all three design requirements of the communication scheme, we propose $\text{MinMax}_{\text{adp}}$ based on $\text{MinMax}_{\text{opt}}$, which trades off little accuracy to be both fast and adaptive. We show that $\text{MinMax}_{\text{adp}}$ can generate a local summary of any desired size. Further, it can modify the size transmitted on the fly, *i.e.*, it can transmit a growing local summary until a predetermined time is reached or a termination signal is received. It can also work when the values are streaming generated.

Algorithm 1: $\text{MinMax}_{\text{adp}}$ for device i.

Input: number of value m, local value vector V_i, number of sampled value K_i

```
1 R_i = rand(m);
2 Q_i = (1/R_i − 1) · V_i · V_i;
3 C_i = kth(Q_i, K_i + 1);
4 V̂_i = where(Q_i > C_i, V_i + C_i/V_i, zeros(m));
```

Sampling process of $\text{MinMax}_{\text{adp}}$: In any device i, given the local value vector V_i and the required sample size K_i, $\text{MinMax}_{\text{adp}}$ first generates m random numbers $r_{i,1}, \cdots, r_{i,m}$ uniformly distributed between 0 and 1, and calculates the priority of m value

$$q_{i,j} = \left(\frac{1}{r_{i,j}} - 1\right) \cdot v^2_{i,j} \tag{7}$$

$\text{MinMax}_{\text{adp}}$ chooses K_i values with the largest $q_{i,j}$ as the sampled values, and sets C_i as the $(K_i + 1)$-th largest $q_{i,j}$. Then, same as in $\text{MinMax}_{\text{opt}}$, for value $v_{i,j}$ not being sampled, $\widehat{v}_{i,j} = 0$, while for the

the value $v_{i,j}$ being sampled, $\widehat{v}_{i,j} = \frac{v_{i,j}^2 + C_i}{v_{i,j}}$.[3] Algorithm 4 shows the pseudo-code of $\text{MinMax}_{\text{adp}}$.

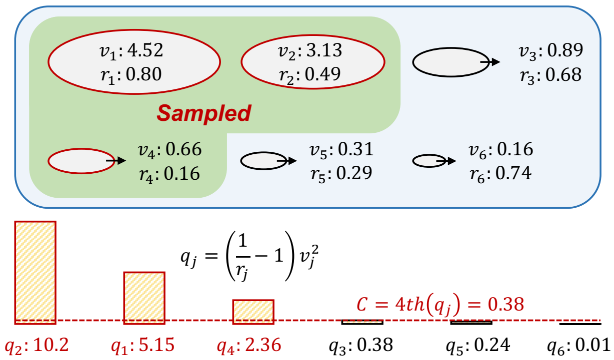


Figure 3: Example of $\text{MinMax}_{\text{adp}}$.

Example of $\text{MinMax}_{\text{adp}}$ (Figures 3): In this example, the local vector vector $V_i = \langle 4.52, 3.13, 0.89.0.66, 0.31, 0.16\rangle$, and the expected number of sampled values is $K_i = 3$. $\text{MinMax}_{\text{adp}}$ first generates 6 random numbers $\langle 0.80, 0.49, 0.68, 0.16, 0.29, 0.74\rangle$ uniformly distributed in $[0, 1]$, and calculates that the sampling priority is $\langle 5.15, 10.2, 0.38, 2.36, 0.24, 0.01\rangle$ according to Formula 7. $\text{MinMax}_{\text{adp}}$ takes the fourth largest priority 0.38 as constant C_i, samples three values $\{v_{i,2} : 4.52, v_{i,1} : 3.13, v_{i,4} : 2.36\}$ with the largest three priorities, and obtains that the estimated values of all values are $\langle 4.61, 3.25, 0, 1.24, 0, 0\rangle$.

Advantages over $\text{MinMax}_{\text{opt}}$:

- **Fast:** $\text{MinMax}_{\text{adp}}$ holds linear time complexity because there are $O(m)$ complexity algorithms [35, 36] to select the $(k+1)$-th largest priority from m priorities. As shown in Table 2, compared with $\text{MinMax}_{\text{opt}}$, the time for finding C in $\text{MinMax}_{\text{adp}}$ is reduced by 94%, and the total time is reduced by 77%.

Table 2: The comparison of computing time, using a value vector containing 2.6×10^5 values.

Computing Time (ms)	Finding C	Others	All
$\text{MinMax}_{\text{opt}}$	16.47	2.85	19.32
$\text{MinMax}_{\text{adp}}$	0.96	3.44	4.40

- **Adaptive:** $\text{MinMax}_{\text{adp}}$ can work in the following three modes.

(1) *Basic mode* — both K_i and the entire V_i are given in advance. $\text{MinMax}_{\text{adp}}$ works as described above.

(2) *Incremental mode* — the entire V_i is given, but K_i is unknown. $\text{MinMax}_{\text{adp}}$ calculates and sorts all priorities, and sends values in the order of priority from large to small, until the predetermined transmission time is reached, or the termination signal from the central device is received. $\text{MinMax}_{\text{adp}}$ then sets C_i as the maximum priority not transmitted, and sends C_i to the central device.

(3) *Streaming mode* — K_i is given, but values are streaming generated. $\text{MinMax}_{\text{adp}}$ calculates the priority of each arriving value, and inserts it into a min-heap of size $K_i + 1$ according to the priority. When the stream ends, $\text{MinMax}_{\text{adp}}$ transmits K_i values with the largest priority in the heap, and sets C_i as the smallest priority in the heap.

[3] In this paper, we define the result of division by 0 as 0 for convenience.

Incremental mode enables $\text{MinMax}_{\text{adp}}$ to fully utilize the bandwidth when bandwidth sharply varies, and streaming mode enables $\text{MinMax}_{\text{adp}}$ to save memory when the local values are streaming generated.

Intuition behind $\text{MinMax}_{\text{adp}}$: Note that in $\text{MinMax}_{\text{opt}}$, we should sample each value $v_{i,j}$ with probability $p_{i,j}$. Specifically, we need to generate a random number $r_{i,j}$ uniformly distributed between 0 and 1 for each $v_{i,j}$, and sample it if and only if

$$r_{i,j} < p_{i,j} = \frac{v_{i,j}^2}{v_{i,j}^2 + C_i}.$$

After a form change of the above formula, we can get that, $v_{i,j}$ is sampled if and only if

$$C_i < \left(\frac{1}{r_{i,j}} - 1\right) \cdot v_{i,j}^2 \tag{8}$$

Based on the above formula, we implement $\text{MinMax}_{\text{adp}}$ which can adjust C_i to get exact K_i samples. We set $q_{i,j} = \left(\frac{1}{r_{i,j}} - 1\right) \cdot v_{i,j}^2$ as the priority of the $v_{i,j}$ and set C_i as the $(K_i + 1)$-th largest priority. According to Equation 8, $\text{MinMax}_{\text{adp}}$ will only sample K_i values with the largest priority.

Note that although $\text{MinMax}_{\text{adp}}$ is proposed based on the transformation of $\text{MinMax}_{\text{opt}}$, its performance is different from that of $\text{MinMax}_{\text{opt}}$. With the constriction of fixed size, $\text{MinMax}_{\text{adp}}$ cannot minimize the maximum variance as that of $\text{MinMax}_{\text{opt}}$. However, $\text{MinMax}_{\text{adp}}$ still inherits some properties of $\text{MinMax}_{\text{opt}}$.

Theorem 3.6. *The sampling scheme $\text{MinMax}_{\text{adp}}$ is an unbiased sampling.*

Proof. Let $\overline{R}_i(j) = \{r_{i,1}, \cdots, r_{i,j-1}, r_{i,j+1}, \cdots, r_{i,m}\}$. We first prove that

$$E\left[\widehat{v}_{i,j} \mid \overline{R}_i(j)\right] = v_{i,j},$$

and without losing generality, we assume that the priority of the other $m-1$ values satisfies the order

$$q_{i,1} > \cdots > q_{i,j-1} > q_{i,j+1} > q_{i,m}.$$

Under this assumption, we have

$$\begin{aligned}
&\Pr[I_{i,j} = 1 \mid \overline{R}_i(j)] \\
&= \Pr\left[q_{i,j} = \left(\frac{1}{r_{i,j}} - 1\right) \cdot v_{i,j}^2 > q_{i,K_i} \mid \overline{R}_i(j)\right] \\
&= \Pr\left[r_{i,j} < \frac{v_{i,j}^2}{v_{i,j}^2 + q_{i,K_i}}\right] = \frac{v_{i,j}^2}{v_{i,j}^2 + q_{i,K_i}},
\end{aligned}$$

where $I_{i,j}$ indicates whether $v_{i,j}$ is sampled as defined above. And when it is sampled, sampling scheme $\text{MinMax}_{\text{adp}}$ sets C_i to the $(K_i + 1)$-th largest priority, that is, $C_i = q_{i,K_i}$, so that we have

$$\begin{aligned}
&E\left[\widehat{v}_{i,j} \mid \overline{R}_i(j)\right] \\
&= \Pr\left[I_{i,j} = 1 \mid \overline{R}_i(j)\right] \cdot E\left[\widehat{v}_{i,j} \mid C_i = q_{i,K_i}, I_{i,j} = 1\right] \\
&= \left(\frac{v_{i,j}^2}{v_{i,j}^2 + q_{i,K_i}}\right) \cdot \left(v_{i,j} + \frac{q_{i,K_i}}{v_{i,j}}\right) = v_{i,j}.
\end{aligned}$$

Note that the above proof is only applicable when $v_{i,j}$ is non-zero. But when $v_{i,j}$ is zero, the sampling probability and estimated value

$\widehat{v}_{i,j}$ are equal to 0, so the conclusion also holds. In addition, we note that the above conclusion dose not depend on the values of random numbers in $\overline{R}_i(j)$, so we have

$$E[\widehat{v}_{i,j}] = E\left[\widehat{v}_{i,j} \mid \overline{R}_i(j)\right] = v_{i,j}$$

Thus sampling scheme MinMax$_{\text{adp}}$ is unbiased sampling. □

Although MinMax$_{\text{adp}}$ cannot achieve the optimal variance like MinMax$_{\text{opt}}$, we show that it is still near-optimal by comparing its variance with MinMax$_{\text{opt}}$ through case study and experiments.

Theorem 3.7. *In sampling scheme MinMax$_{adp}$,*

$$D[\widehat{v}_{i,j}] = \begin{cases} 0, & v_{i,j} = 0 \\ E[\overline{Q}_i(j, K_i)], & v_{i,j} \neq 0 \end{cases}. \tag{9}$$

where

$$\overline{Q}_i(j,k) = kth\left\{q_{i,1}, \cdots, q_{i,j-1}, q_{i,j+1}, \cdots, q_{i,m}\right\}.$$

Proof. We first assume that the set $\overline{R}_i(j)$ has been determined, and we only consider the randomness of $r_{i,j}$, and deduce the form of variance of $\widehat{v}_{i,j}$ under this condition. We have

$$\begin{aligned} &E\left[\widehat{v}^2_{i,j} \mid \overline{R}_i(j)\right] \\ =&\Pr\left[I_{i,j} = 1 \mid \overline{R}_i(j)\right] \cdot E\left[\widehat{v}^2_{i,j} \mid C_i = \overline{Q}_i(j, K_i), I_{i,j} = 1\right] \\ =&\left(\frac{v^2_{i,j}}{v^2_{i,j} + \overline{Q}_i(j, K_i)}\right) \cdot \left(v_{i,j} + \frac{\overline{Q}_i(j, K_i)}{v_{i,j}}\right)^2 \\ =&v^2_{i,j} + \overline{Q}_i(j, K_i). \end{aligned}$$

Considering the unbiasedness of $\widehat{v}_{i,j}$, we have

$$D\left[\widehat{v}_{i,j} \mid \overline{R}_i(j)\right] = \overline{Q}_i(j, K_i)$$

Because the above conclusion does not depend on the random numbers in $\overline{R}_i(j)$, so we have

$$D[\widehat{v}_{i,j}] = D\left[\widehat{v}_{i,j} \mid \overline{R}_i(j)\right] = E[\overline{Q}_i(j, K_i)]$$

□

Variance of MinMax$_{\text{adp}}$ (Theorem 3.7): This theorem shows that although the variance of each $\widehat{v}_{i,j}$ in MinMax$_{\text{adp}}$ is different, it has a common upper bound $E[\overline{C}_i]$, where $\overline{C}_i$ is the K_i-th largest priority among all priorities.

Case study on uniform distribution: To better understand the near-optimality of MinMax$_{\text{adp}}$, we compare the variance between MinMax$_{\text{opt}}$ and MinMax$_{\text{adp}}$ in the case of uniform distribution. Specifically, for any j, $v_{i,j} = v_{i,1} \neq 0$. For MinMax$_{\text{opt}}$, based on the Formula 3 and Theorem 3.1, we have

$$D[\widehat{v}_{i,j}] = C_i = \frac{m - K_i}{K_i} v^2_{i,1} \tag{10}$$

For MinMax$_{\text{adp}}$, we first note that $1/r_{i,j}$ follows the Pareto distribution. Suppose that $\overline{R}(k,m)$ is the k-th largest $1/r_{i,j}$ among m $1/r_{i,j}$. Based on the order statistics of Pareto distribution shown in prior works [37], we have

$$E[\overline{R}(k,m)] = \frac{m}{k-1}.$$

Based on Formula 7 on $q_{i,j}$ and Theorem 3.7,

$$D[\widehat{v}_{i,j}] = (E[\overline{R}(K_i, m-1)] - 1) \cdot v^2_{i,1} = \frac{m - K_i}{K_i - 1} v^2_{i,1} \tag{11}$$

We can find that, in such case, MinMax$_{\text{adp}}$ is near-optimal. The variance of MinMax$_{\text{adp}}$ on K_i samples is better than that of MinMax$_{\text{opt}}$ on $K_i - 1$ samples.

We also compare the variances of MinMax$_{\text{opt}}$ and MinMax$_{\text{adp}}$ in the real dataset. As shown in Figure 4, the difference between the variances of the two schemes is within 0.05% in all cases.

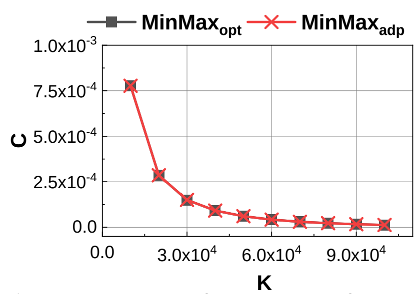


Figure 4: The comparison of variances of MinMax$_{\text{opt}}$ and MinMax$_{\text{adp}}$, using a value vector containing 2.6×10^5 values.

In addition to the near-optimal variance, our sampling scheme MinMax$_{\text{adp}}$ also holds the *weight sensitivity*, that is to say, our sampling scheme MinMax$_{\text{adp}}$ can ensure that larger values can be sampled with a larger probability than smaller values.

Theorem 3.8. *In sampling scheme MinMax$_{adp}$, if two non-zero values satisfy $\rho = |v_{i,j}|/|v_{i,k}| > 1$, then there is*

$$\Pr[q_{i,j} < q_{i,k}] = \frac{\rho^2 \cdot (2 \cdot \ln(\rho) - 1) + 1}{(\rho^2 - 1)^2}, \tag{12}$$

where $q_{i,j}$ and $q_{i,k}$ are priorities.

Proof. The proof is shown in the following formula.

$$\begin{aligned} \Pr[q_{i,j} < q_{i,k}] &= \Pr\left[\left(\frac{1}{r_{i,j}} - 1\right) \cdot v^2_{i,j} < \left(\frac{1}{r_{i,k}} - 1\right) \cdot v^2_{i,j}\right] \\ &= \Pr\left[r_{i,k} < \frac{r_{i,j} \cdot v^2_{i,k}}{(1 - r_{i,j}) \cdot v^2_{i,j} + r_{i,j} \cdot v^2_{i,k}}\right] \\ &= \int_0^1 \frac{r_{i,j}}{(1 - r_{i,j}) \cdot \rho^2 + r_{i,j}} dr_{i,j} \\ &= \frac{\rho^2 \cdot (2 \cdot \ln(\rho) - 1) + 1}{(\rho^2 - 1)^2}. \end{aligned}$$

□

3.3 Aggregation & Outlier Elimination

In this section, we introduce the properties of our sampling scheme MinMax$_{\text{adp}}$ when aggregating the sampled local vectors of multiple devices, and how to eliminate outliers after aggregation to further improve the accuracy.

Aggregation: In each round of global aggregation, local device i with local vector V_i runs MinMax$_{\text{adp}}$ to generate the local summary

$$\widehat{V_i} = \left\langle \widehat{v}_{i,1}, \widehat{v}_{i,2}, \cdots, \widehat{v}_{i,m} \right\rangle.$$

Then, device i transmits the sampled values $v_{i,j}$ ($\widehat{v}_{i,j} \neq 0$), the corresponding indexes, and the parameter C_i. The central device receives local summaries from n local devices, and estimates the global value vector V using the following formula

$$\widehat{V} = \sum_{i=1}^{n} \widehat{V_i} = \sum_{i=1}^{n} \left(V_i + \frac{C_i}{V_i} \right). \tag{13}$$

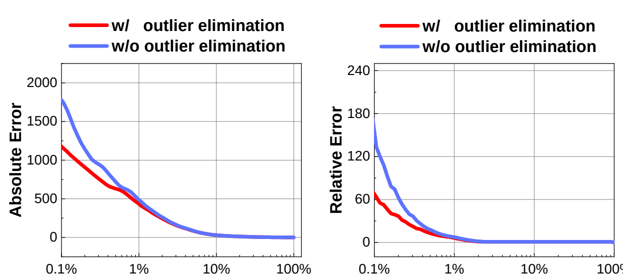
According to Theorem 3.6 and 3.7, $\sum_{i=1}^{n} \widehat{v}_{i,j}$ is an unbiased estimate of $\sum_{i=1}^{n} v_{i,j}$, and $E[\sum_{i=1}^{n} \overline{C}_i]$ is an upper bound of $D[\sum_{i=1}^{n} \widehat{v}_{i,j}]$. We then give the following properties about the variance.

Theorem 3.9. *Assuming $E[\overline{C}_i]$ is an independent sample of random variable $\overline{C}$, and $E[\overline{C}] < \infty$, then for any global value $\sum_{i=1}^{n} v_{i,j}$, there is*

$$\lim_{n\to\infty} D\left[\frac{\sum_{i=1}^{n} \widehat{v}_{i,j}}{\sqrt{n}}\right] \to O(1). \tag{14}$$

Proof. According to *the theorem of large numbers*, we have $\lim_{n\to\infty} \frac{\sum_{i=1}^{n} E[\overline{C}_i]}{n} = E[\overline{C}]$. Then according to Theorem 3.7, we have $\lim_{n\to\infty} D\left[\frac{\sum_{i=1}^{n} \widehat{v}_{i,j}}{\sqrt{n}}\right] \leqslant \lim_{n\to\infty} \frac{1}{n} E[\sum_{i=1}^{n} \overline{C}_i] \to O(1)$. □

Scalable of aggregation (Theorem 3.9): This theorem shows that, the error of MinMax$_{\text{adp}}$ grows sub-linearly (*i.e.*, $O(\sqrt{n})$) as the number of devices increases. Note that, for prior sketch-based communication scheme (*e.g.*, FetchSGD [20]) being unbiased or other schemes being biased [25, 38], their error often grows linearly (*i.e.*, $O(n)$) as the number of devices increases.



(a) Distribution of absolute errors. (b) Distribution of related errors.

Figure 5: The distribution of absolute error and relative error with and without outlier elimination. We set the estimated value with *outlier degree* greater than 100 to 0, as shown in figure, which significantly reduces the top 0.1% error.

Outlier Elimination: Although MinMax$_{\text{adp}}$ minimize the maximum variance, there are still some cases that the gap between estimated global value and real global value is large. Such cases happen with small probability, but may significantly degrade the accuracy once it happens. Therefore, we filter such outliers after we receive all the local summaries to make our accuracy more stable. We filter outliers based on the idea of hypothesis testing. We first assume that the estimated global value is equal to the real global value, *i.e.*, our estimation on such global value vector is totally correct. Then, we estimate whether the estimated value meets our expectation. If it does not meet the expectation, we reject the assumption and filter such estimated value. We expect to have

$$\frac{\left(\sum_{i=1}^{n} \widehat{v}_{i,j}\right)^3}{\left(\sum_{i=1}^{n} \widehat{v}_{i,j}\right)^2 + D[\sum_{i=1}^{n} \widehat{v}_{i,j}]} = \sum_{\widehat{v}_{i,j} \neq 0} v_{i,j}. \tag{15}$$

It is because that for a local device, and a value $v_{i,j}$ on it, we have a probability of $\frac{v_{i,j}^2}{v_{i,j}^2 + C_i}$ to transmit it to the central device. In the other words, the expected value received by the central device is $\frac{v_{i,j}^3}{v_{i,j}^2 + C_i}$. Therefore, we define the *outlier degree* of $\sum_{i=1}^{n} \widehat{v}_{i,j}$ as

$$\left(\frac{\left(\sum_{i=1}^{n} \widehat{v}_{i,j}\right)^3}{\left(\sum_{i=1}^{n} \widehat{v}_{i,j}\right)^2 + \sum_{i=1}^{n} \overline{C}_i}\right)\left(\sum_{\widehat{v}_{i,j} \neq 0} v_{i,j}\right)^{-1}. \tag{16}$$

The larger outlier degree is, the higher probability it is an outlier.

Interpretation of outlier degree: Based on the derivation of outlier degree, we can roughly regard the threshold as the reciprocal of the probability that the estimated global value is not an outlier. Specifically, if we set the threshold to 100, we will eliminate the estimated values with more than 99% probability of being outliers. We often set the threshold to 100 in experiments.

Verification of outlier elimination (Figure 5): We verify that outlier elimination can make our accuracy more stable as shown in Figure 5. Eliminating outliers requires each local device to transmit $\overline{C}_i$ to estimate the upper bound of variance $D[\sum_{i=1}^{n} \widehat{v}_{i,j}]$. However, $\overline{C}_i$ and C_i are very close, often within a gap less than 0.1%, so C_i can be used instead of $\overline{C}_i$.

4 APPLICATIONS

In this section, we show how to apply MinMax Sampling to different distributed applications, and take federated learning, distributed state aggregation, and hierarchical aggregation as case studies.

Federated Learning: In federated learning, assuming that the model contains m parameters, we let the local value vector be the model gradient concatenated layer-by-layer. Assuming that there are n local devices, only n_p of them may participate in computing and communication in each iteration. We call these devices participating devices. Different from the standard MinMax Sampling, each participating device p_i needs to transmit an additional D_{p_i}, *i.e.*, the amount of training data it uses in this round. After receiving the local summaries transmitted from all participating devices, the Parameter Server, *i.e.*, the central device, needs to calculate the weighted average of all local summaries instead of directly adding them up, *i.e.*,

$$\widehat{V} = \frac{\sum_{i=1}^{n_p} D_{p_i} \cdot \widehat{V_{p_i}}}{\sum_{i=1}^{n_p} D_{p_i}}. \tag{17}$$

Distributed State Aggregation: In distributed state aggregation, we need to aggregate the local states of all local devices. We take

CDN as an example. For CDN, the local state is the access log, which records the number of times each source IP accesses the server. We let the local value vector cover the entire 32-bit IP domain (*i.e.*, have a length of 2^{32}), and the values in the vector are the number of accesses from each address. For those IP addresses that have not sent a request to the local device, the corresponding values in the local value vector are 0. In practice, zero values do not occupy local memory, and MinMax Sampling does not calculate sampling probability or priority for them. After the central device obtains the global summary, it can perform a variety of analysis tasks, including 1) *estimating global values*: estimating the total number of accesses from each source IP address; 2) *finding global top-K*: finding K source IP addresses that access the CDN most frequently.

Hierarchical Aggregation: In hierarchical aggregation, local devices are often organized in the form of a rooted tree, and the aggregation process happens from leaf devices to the root device level-by-level. We first show that the unbiasedness of the MinMax Sampling can be extended to hierarchical aggregation. Suppose n leaf devices are connected to a common parent device, each of which has a local value vector V_i, and generates a corresponding summary $\hat{V}_i$ with the upper bound $\overline{C}_i$ of variance. In the first level aggregation, the parent device gets a summary $\widehat{V} = \sum_{i=1}^{n} \widehat{V}_i$. In the second level aggregation, it generates a new summary $\overline{V}$ with the upper bound $\overline{C}$ of variance, using $\widehat{V}$ from MinMax Sampling, then we have

$$E[\overline{v}_j] = E[E[\overline{v}_j \mid \widehat{v}_j]] = E[\widehat{v}_j] = \sum_{i=1}^{n} v_{i,j}.$$

Similarly, we can also obtain the upper bound of the $D[\overline{v}_j]$,

$$D[\overline{v}_j] = E[E[\overline{v}_j^2 \mid \widehat{v}_j]] - v_j^2 \leqslant E[\widehat{v}_j^2 + \overline{C}] - v_j^2 \leqslant \overline{C} + \sum_{i=1}^{n} \overline{C}_i.$$

Therefore, when communicating with the device in the upper level, the upper bound of variance transmitted by the parent device should be $\overline{C} + \sum_{i=1}^{n} \overline{C}_i$ rather than $\overline{C}$.

Extensions of MinMax Sampling: In addition to the `SUM` aggregation discussed above, MinMax Sampling supports all aggregation operations which can be transformed to the combination of `weighted SUM`. Note that `COUNT` and `AVERAGE` can both be transformed to the `weighted SUM`. For example, MinMax Sampling can support the `COUNT` by regarding the value of each key that is present in each device to 1, others as 0. After aggregation, we can know how many devices where each key exists.

5 EXPERIMENTAL RESULTS

We conduct experiments and compare our $\text{MinMax}_{\text{adp}}$ with other algorithms on three different applications: federated learning, distributed state aggregation, and hierarchical aggregation.

5.1 Experiments on Federated Learning

In this subsection, we compare the performance of $\text{MinMax}_{\text{adp}}$ on federated learning with GSpar [39], top-K sparsification [38], and FetchSGD (using Count sketches) [20]. We also compare the algorithms with the ideal/uncompressed results.

Implementation: We implement $\text{MinMax}_{\text{adp}}$ and other algorithms using PyTorch. For top-K sparsification and FetchSGD, the setting is based on the recommendation of prior works [20, 38]. For Gspar, we utilize the greedy approach and set the parameters based on the recommendation of [39]. For $\text{MinMax}_{\text{adp}}$, we utilize outlier elimination and set the threshold to 100.

Datasets:

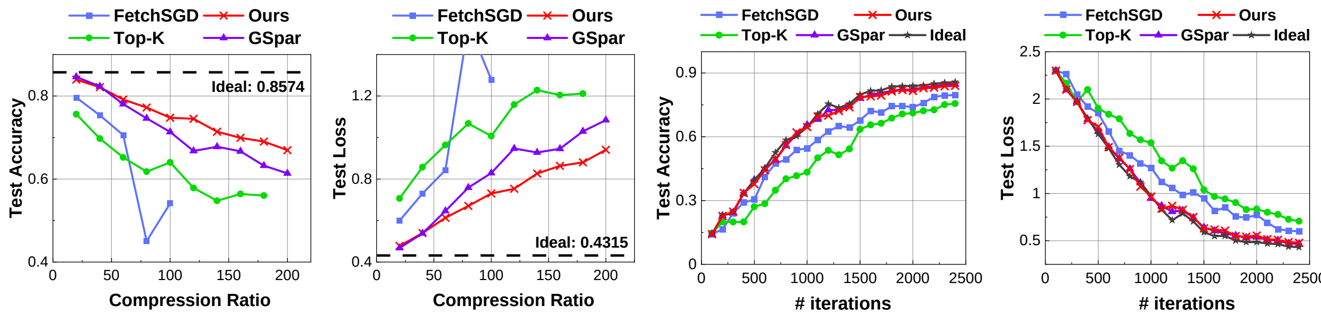
- *CIFAR-10:* CIFAR-10 [40] is an image classification dataset. It consists of 60,000 32x32 color images in 10 classes and is divided into 50,000 training images and 10,000 test images. We use Resnet-9 [41] on CIFAR-10, which contains 6.5M parameters. We train for 2400 iterations.
- *FEMNIST:* Federated EMNIST [42] is an image classification dataset formed by partitioning EMNIST [43] such that each client in FEMNIST contains characters written by a single person. It consists of 805,263 images with 62 labels and is allocated to 3,550 workers. We use Resnet-101 [41] with layer normalization on FEMNIST, which contains 40M parameters.

For FEMNIST, we allocate the data directly. For CIFAR-10, we allocate the data in the following two ways:

- *non-i.i.d:* We first randomly select a main label for each worker, then we randomly allocate each image to one worker according to the following rule: workers with the same main label have a higher probability of being assigned. More specifically, 80% images of the workers belong to the selected label, and the remaining 20% belong to other labels. We use the non-i.i.d. CIFAR-10 dataset to simulate federated learning in two settings. **Small-scale federated learning**: 200 workers in total, with 5% of workers participating in each iteration, and each worker uses a local batch of size 50. **Large-scale federated learning**: 2000 workers in total, with 1% of workers participating in each iteration, and each worker uses a local batch of size 25.
- *i.i.d:* We randomly allocate each image to one worker with equal probability. We use the i.i.d. CIFAR-10 dataset to simulate **distributed machine learning**: 10 workers in total, with 100% of workers participating in each iteration, and each worker uses a local batch of size 50.

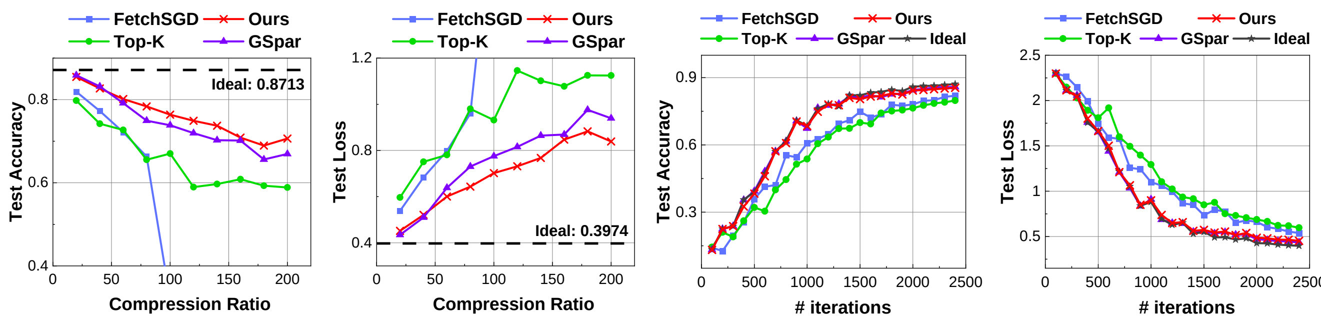
Evaluation on small-scale FL (Figure 6): The experimental results show that under the small-scale federated learning setting, the test accuracy and test loss of $\text{MinMax}_{\text{adp}}$ are always closer to the ideal results than other algorithms. As shown in Figure 6(a) and 6(b), when varying the compression ratio, $\text{MinMax}_{\text{adp}}$ reduces the test accuracy by 10.82% on average, while GSpar, top-K, and FetchSGD reduce the accuracy by 14.06%, 28.44%, and 45.95%, respectively. Meanwhile, the increased test loss of $\text{MinMax}_{\text{adp}}$ is 0.298, which is 6.90 times lower than that of GSpar, top-K, and FetchSGD on average. As shown in Figure 6(c) and 6(d), when the compression ratio is 20, $\text{MinMax}_{\text{adp}}$ and GSpar have similar training processes to ideal, and the difference between their test accuracy is less than 1.76%. On the other hand, the test accuracy of top-K and FetchSGD are 10.11% and 6.09% lower than that of ideal, respectively.

Evaluation on large-scale FL (Figure 7 & 9): The experimental results show that under the large-scale federated learning setting, the test accuracy and test loss of $\text{MinMax}_{\text{adp}}$ are always closer to the ideal results, compared with other algorithms. As shown in Figure 7(a) and 7(b), when varying the compression ratio, $\text{MinMax}_{\text{adp}}$ reduces the test accuracy by 11.09% on average, while GSpar, top-K, and FetchSGD reduce the accuracy by 13.12%, 21.61%, and 43.68%,



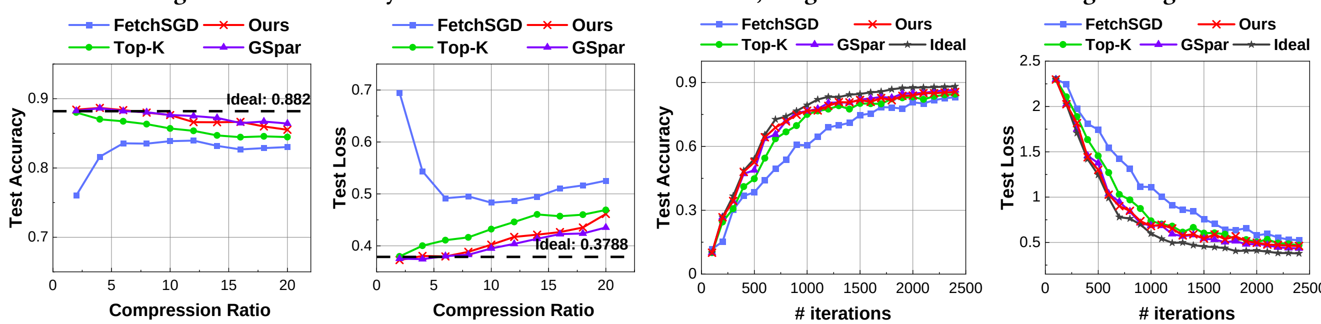
(a) Test accuracy *v.s.* compression ratio (b) Test loss *v.s.* compression ratio (c) Test accuracy *v.s.* iterations (20 ×) (d) Test loss *v.s.* iterations (20 ×)

Figure 6: Test accuracy & test loss on non-i.i.d CIFAR-10, Small-scale federated learning setting.



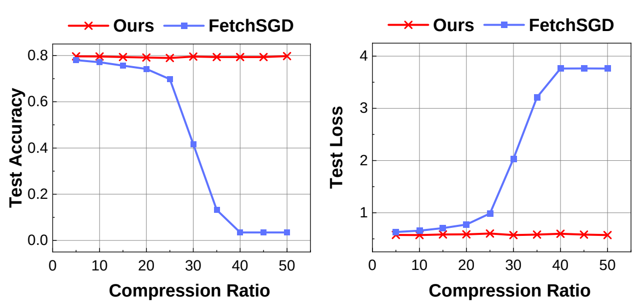
(a) Test accuracy *v.s.* compression ratio (b) Test loss *v.s.* compression ratio (c) Test accuracy *v.s.* iterations (20 ×) (d) Test loss *v.s.* iterations (20 ×)

Figure 7: Test accuracy & test loss on non-i.i.d CIFAR-10, Large-scale federated learning setting.



(a) Test accuracy *v.s.* compression ratio (b) Test loss *v.s.* compression ratio (c) Test accuracy *v.s.* iterations (20 ×) (d) Test loss *v.s.* iterations (20 ×)

Figure 8: Test accuracy & test loss on i.i.d CIFAR-10, Distributed machine learning setting.

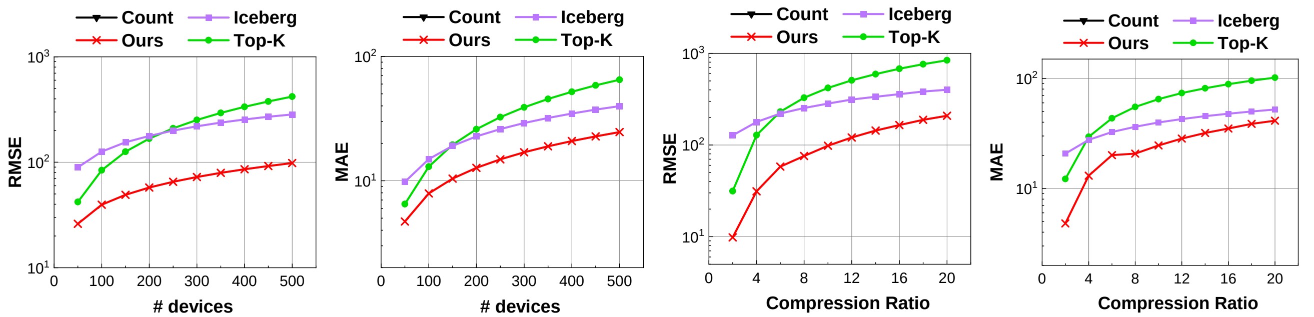


(a) Test accuracy. (b) Test loss.

Figure 9: Test accuracy & test loss *v.s.* compression ratio on FEMNIST.

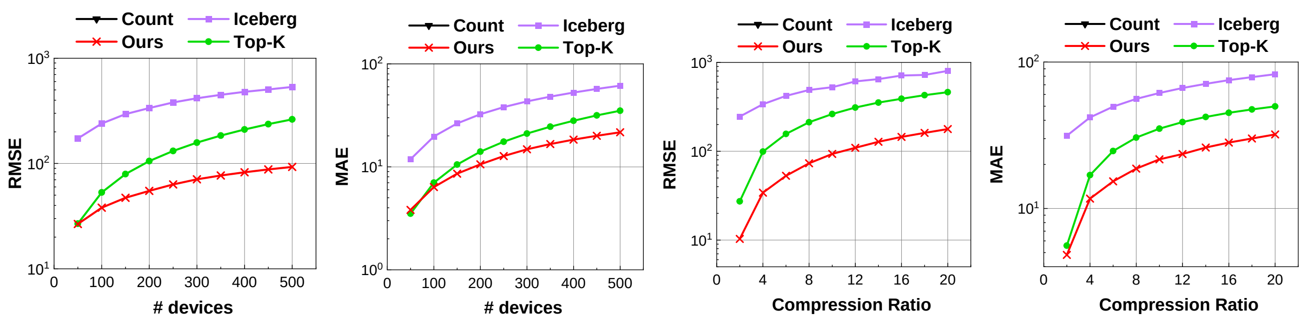
respectively. Meanwhile, the increased test loss of $\text{MinMax}_{\text{adp}}$ is 0.301, which is 1.29 times lower than that of GSpar, top-K, and FetchSGD on average. As shown in Figure 7(c) and 7(d), when the compression ratio is 20, $\text{MinMax}_{\text{adp}}$ and GSpar have similar training processes to ideal, and the difference between their test accuracy is less than 1.85%. On the other hand, the test accuracy of top-K and FetchSGD are 7.53% and 5.4% lower than that of ideal, respectively. Further, we also perform evaluations on the FEMNIST dataset. As shown in Figure 9, on FEMNIST, when the compression ratio exceeds 25, the test accuracy of FetchSGD decreases sharply, and its loss increases sharply. Meanwhile, the increasing compression ratio has little effect on the accuracy and loss of $\text{MinMax}_{\text{adp}}$. That's mainly because of the low accuracy of the Count sketch under a high compression ratio. Therefore, FetchSGD can hardly transmit the gradient with acceptable error.

Evaluation on distributed ML (Figure 8): The experimental results show that under the distributed machine learning setting, the test accuracy and test loss of ideal, $\text{MinMax}_{\text{adp}}$, and GSpar are similar. As shown in Figure 8(a) and 8(b), when varying the compression ratio, $\text{MinMax}_{\text{adp}}$ and GSpar reduce the test accuracy by less than 0.96% on average, while top-K and FetchSGD reduce the accuracy by 2.47% and 5.78%, respectively. Meanwhile, the increased test loss of $\text{MinMax}_{\text{adp}}$ and GSpar are less than 0.030, which is 14.3 times lower than that of top-K and FetchSGD on average. As



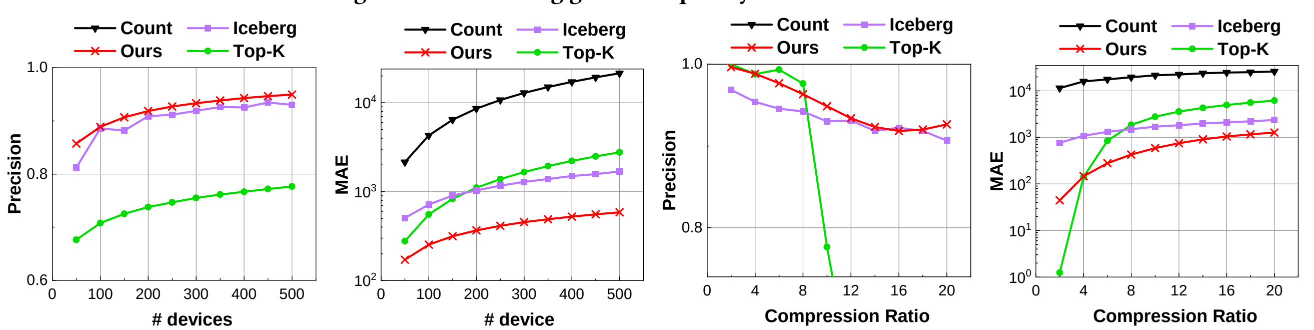
(a) RMSE *v.s.* # devices. (b) MAE *v.s.* # devices. (c) RMSE *v.s.* compression ratio. (d) MAE *v.s.* compression ratio.

Figure 10: Estimating global frequency on i.i.d CAIDA.



(a) RMSE *v.s.* # devices. (b) MAE *v.s.* # devices. (c) RMSE *v.s.* compression ratio. (d) MAE *v.s.* compression ratio.

Figure 11: Estimating global frequency on non-i.i.d CAIDA.



(a) Precision *v.s.* # devices. (b) MAE *v.s.* # devices. (c) Precision *v.s.* compression ratio. (d) MAE *v.s.* compression ratio.

Figure 12: Finding global top-K on i.i.d CAIDA.

shown in Figure 7(c) and 7(d), when the compression ratio is 20, $\text{MinMax}_{\text{adp}}$ and GSpar have similar training processes to ideal, while the training process of FetchSGD is slower.

5.2 Experiments on Distributed State Aggregation

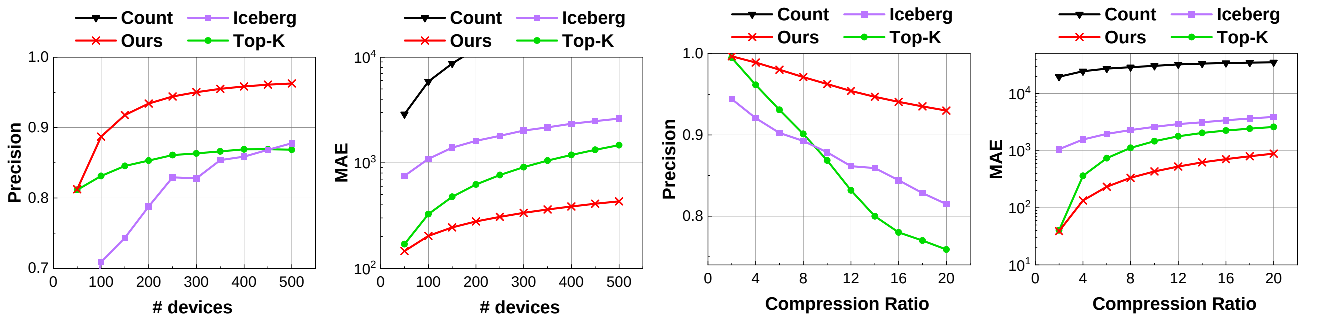
In this subsection, we compare the performance of $\text{MinMax}_{\text{adp}}$ in distributed state aggregation with Iceberg sampling [9], top-K sampling [32], and the Count sketch [26]. We focus on two tasks: estimating global frequency and finding global top-K.

Implementation: We simulate a distributed state aggregation process of 500 local devices. We implement $\text{MinMax}_{\text{adp}}$ and other algorithms using PyTorch. For $\text{MinMax}_{\text{adp}}$, we use outlier elimination and set the threshold to 100. For Count sketches, we set the number of rows to 3. For Iceberg sampling, we set D to the number of transmitted values.

Datasets: We use the traffic dataset collected from CAIDA [44] in 2018. Each packet is associated with a 5-tuple (<*srcIP, dstIP, srcPort, dstPort, protocol*>). The dataset with a monitor time of one hour contains 1.47×10^9 packets, 5.84×10^7 distinct 5-tuples, and 3.72×10^6 distinct *srcIP*. We allocate data in the following two ways:

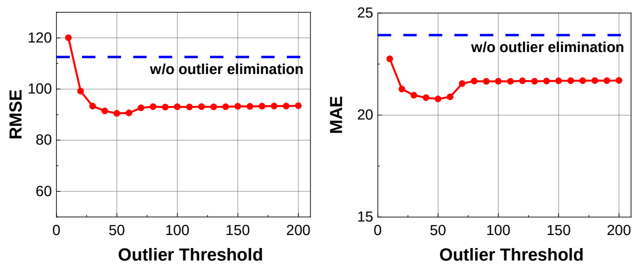
- *i.i.d:* We randomly allocate each packet to one device, and count the number of the packets of each *srcIP* on each local device. Each local device contains 1.81×10^5 *srcIP* on average.
- *non-i.i.d:* We hash each packet to one device according to its 5-tuple. On each device, we also count the number of each *srcIP*. Each local device contains 4.93×10^4 *srcIP* on average.

Estimating Global Frequency (Figure 10-11): The experimental results show that when estimating global frequency, $\text{MinMax}_{\text{adp}}$ achieves lower error compared with other algorithms. As shown in Figure 10, on i.i.d dataset, the RMSE of $\text{MinMax}_{\text{adp}}$ reaches 3.98, 4.04, and 668 times lower than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of $\text{MinMax}_{\text{adp}}$ reaches 1.83, 2.48, and 1128 times lower, respectively. The RMSE and MAE of Count are always larger than 15,000 and 2,000, respectively. As shown in



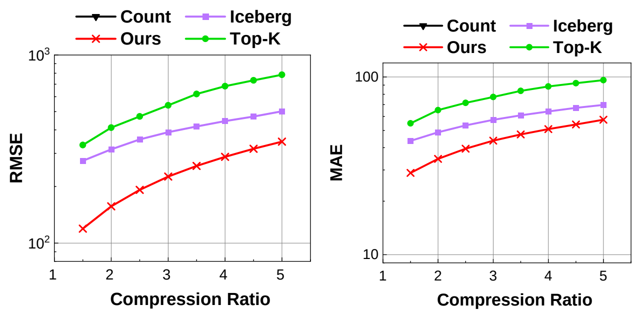
(a) Precision *v.s.* # devices. (b) MAE *v.s.* # devices. (c) Precision *v.s.* compression ratio. (d) MAE *v.s.* compression ratio.

Figure 13: Finding global top-K on non-i.i.d CAIDA.



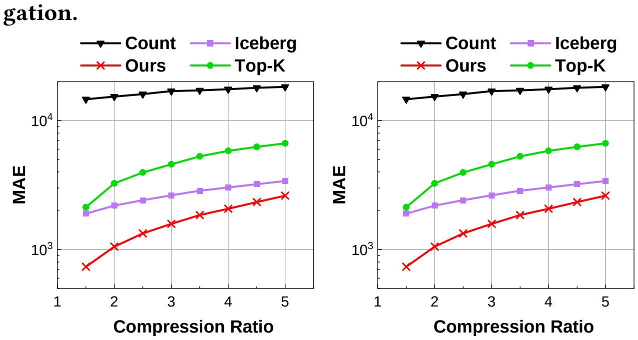
(a) RMSE *v.s.* threshold. (b) MAE *v.s.* threshold.

Figure 14: The impact of the threshold for outlier elimination on the accuracy of distributed state aggregation tasks.



(a) RMSE. (b) MAE.

Figure 15: Estimating global frequency in hierarchical aggregation.



(a) Precision. (b) MAE

Figure 16: Finding global top-K in hierarchical aggregation.

Figure 11, on non-i.i.d dataset, the RMSE of MinMax$_{\text{adp}}$ reaches 7.87, 2.78, and 1996 times lower than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of MinMax$_{\text{adp}}$ reaches 3.26, 1.55, and 4834 times lower, respectively. The RMSE and MAE of Count are always larger than 18,000 and 9,000, respectively. It is worth noticing that Iceberg sampling has lower error than top-K on i.i.d dataset, while top-K sampling has lower error than Iceberg on non-i.i.d dataset. Meanwhile, MinMax$_{\text{adp}}$ achieves the highest accuracy on both cases.

Finding Global Top-K (Figure 12-13): The experimental results show that when finding global top-K, MinMax$_{\text{adp}}$ achieves higher precision and lower error compared with other algorithms. As shown in Figure 12, on i.i.d dataset, the precision of MinMax$_{\text{adp}}$ is 1.6%, 25.8%, and 90.8% higher than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of MinMax$_{\text{adp}}$ is 4.61, 3.70, and 63 times lower, respectively. The precision of Count is always less than 0.15. As shown in Figure 13, on non-i.i.d dataset, the precision of MinMax$_{\text{adp}}$ is 8.6%, 10.1%, and 94.4% higher than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of MinMax$_{\text{adp}}$ is 8.44, 2.94, and 121 times lower, respectively. The precision of Count is always less than 0.05. It is worth noticing that on i.i.d dataset, top-K sampling has high precision and low error under low compression ratio. That's mainly because in this case, local top-K is most likely to be global top-K. Therefore, top-K sampling can accurately report all global top-K, thus achieving high accuracy. However, under high compression ratio, top-K sampling fails to transmit global top-K completely, therefore, the accuracy decreases rapidly. On the contrary, MinMax$_{\text{adp}}$ treats each value fairly, which means that smaller values also have the probability to be transmitted. Therefore, MinMax$_{\text{adp}}$ achieves high accuracy at all compression ratios.

Verification of outlier elimination (Figure 14): We further verify the impact of the outlier elimination threshold on aggregation accuracy. As shown in Figure 14, for distributed state aggregation tasks, we can find that setting the threshold to 50 can minimize both MAE and RMSE. The optimal threshold differs on different tasks. Fortunately, we find that when the threshold ranges from 30 to 200, MAE and RMSE are stable and much better than without outlier elimination. Therefore, we recommend setting the threshold to 100 if the user cannot easily get the optimal threshold. Although we do not set the threshold to the optimal value, the performance of MinMax$_{\text{adp}}$ is still much better than other algorithms.

5.3 Experiments on Hierarchical Aggregation

We simulate a 2-level hierarchical aggregation process with 10,000 leaf devices. Every 100 leaf devices are connected with a common parent device, and all 100 parent devices are connected with the root device. We use CAIDA dataset and allocate the dataset to each leaf device in i.i.d way. Other settings are similar to that in distributed state aggregation (see Section 5.2).

Estimating Global Frequency (Figure 15): The experimental results show that when estimating global frequency, $\text{MinMax}_{\text{adp}}$ achieves lower error compared with other algorithms. As shown in Figure 15, the RMSE of $\text{MinMax}_{\text{adp}}$ reaches 1.75, 2.45, and 188 times lower than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of $\text{MinMax}_{\text{adp}}$ reaches 1.32, 1.78, and 626 times lower, respectively. The RMSE and MAE of Count are always larger than 25,000 and 14,000, respectively.

Finding Global Top-K (Figure 16): The experimental results show that when finding global top-K, $\text{MinMax}_{\text{adp}}$ achieves higher precision and lower error compared with other algorithms. As shown in Figure 16, the precision of $\text{MinMax}_{\text{adp}}$ is 1.1%, 45.7%, and 85.9% higher than that of Iceberg, top-K sampling, and Count on average, respectively; the MAE of $\text{MinMax}_{\text{adp}}$ is 1.73, 2.84, and 11.19 times lower, respectively. The precision of Count is always less than 0.05.

6 RELATED WORK

Sketching algorithms: Sketches are a class of probabilistic data structures. We should note that many prior sketching algorithms (*e.g.*, WavingSketch [27], Unbiased SpaceSaving [28], *etc.* [45]) are mostly designed for a single device and cannot work on global aggregation. Typical sketches used on global aggregation are linear sketches such as the Count-Min sketch [25] and the Count sketch [26]. Specifically, prior works [46] on global aggregation often use Count sketches which are unbiased and provide a good accuracy guarantee. The Count sketch [26] consists of a counter matrix with r rows and c columns. Each row is associated with two hash functions h_i and g_i. When inserting a vector with length m, for each row, the Count sketch first uses g_i to calculate a vector with length m and value ± 1. Then the sketch multiplies the two vectors and uses f_i to linearly map them to the row and sums them up. However, linear sketches can only be merged if the size of sketches on all devices being all the same, which prevents them from the adaptive transmission.

Sampling algorithms: Sampling for global aggregation can be divided into two categories: biased sampling and unbiased sampling. Top-K sampling [32] is a typical biased sampling. However, its error is linear to the number of local devices, which brings unacceptable error in a large-scale scenario. Iceberg sampling [9] and H-sampling [31] are two typical unbiased samplings. Iceberg sampling [9] proposes a sampling strategy such that the MSE is independent of distribution, aiming to minimize the worst-case mean square error. It proposes using the sampling function $p(x) = \frac{v}{v+D}$, where D is a global constant that depends on the compression ratio. H-sampling [31] aims to minimize the number of sampled values under a given cost. It proposes an instance-optimal sampling method, *i.e.*, for every given input, the sampling function always holds the optimal cost among all the valid sampling functions. However, both methods require all local devices share the same configuration, thus being unable to adjust the compression ratio on each local device according to its available bandwidth.

Federated Learning: There are many prior works focusing on accelerating aggregation in federated learning, including reducing communication rounds (*e.g.*, FedAvg [47]), asynchronous communication (*e.g.*, ASO-Fed [48]), *etc.* [12, 49, 50]. Among them, the state-of-the-art algorithms are top-K sparsification [38] and FetchSGD [20]. DGC [38] proposes using top-K sparsification to compress gradients. This method overcomes the disadvantage of FedAvg's requirement to transmit the full gradient. However, DGC requires error feedback [51], which means all local clients have to record their residuals for update afterward, resulting in extra local storage and risks of losing information when local devices are out of contact. FetchSGD[20] takes advantage of the linear property of Count sketches to accommodate the federated learning scenario. It avoids error feedback in each client by transmitting the gradient unbiasedly. However, although theoretical proof shows FetchSGD achieves success in combining sketch and federated learning, it still needs improvement in practice, such as adaptiveness and acceptable accuracy under a high compression ratio.

Distributed Machine Learning: There are also many prior works focusing on accelerating aggregation in distributed machine learning, including GSpar [39], Hard-threshold [52], signSGD [53], GRACE [54], DeepReduce [55], *etc.* [56, 57]. However, different settings of federated learning and distributed machine learning lead to different bottlenecks and different preferences of communication schemes. As shown in prior federated learning works [58, 59], many algorithms about distributed machine learning (*e.g.*, top-K sparsification and signSGD) fail in federated learning due to different settings.

System for Wide-Area Network: Many prior works focus on transmitting and aggregating data in the Wide-Area Network, such as AWStream [19] and JetStream [7]. They mainly focus on video streams. To achieve fast and reliable transmission, they learn the most appropriate transmission rate from historical network status through Pareto-optimal configurations and provide detailed video down-sampling strategies. These works also claim that they support global aggregation tasks, but they do not provide a detailed scheme to reduce the number of transmitted values. Fortunately, MinMax Sampling can be combined with their efforts as a near-optimal sampling scheme.

7 CONCLUSION

In this paper, we propose MinMax Sampling, a communication scheme for global aggregation in the Wide-Area Network. MinMax Sampling contains two versions, $\text{MinMax}_{\text{opt}}$ achieves optimal accuracy: it can minimize the maximum variance of all estimated values, while $\text{MinMax}_{\text{adp}}$ meets all three requirements of designing a communication scheme: 1) Fast computation. It can generate a local summary of any desired size in linear time. 2) Adaptive transmission. It supports three different working modes: basic mode, incremental mode, and streaming mode. 3) Near-optimal accuracy. It can control the variance of all estimated values to the near-optimal variance. We evaluate $\text{MinMax}_{\text{adp}}$ with three applications: federated learning, distributed state aggregation, and hierarchical aggregation. Our experimental results show that $\text{MinMax}_{\text{adp}}$ is superior to existing algorithms in all three applications: compared with the state-of-the-art, its accuracy is 8.44× higher on average. Further, accurate global aggregation can help improve the accuracy of federated learning by up to 7.73%. The source codes of MinMax Sampling are available at Github [1].

REFERENCES

[1] Source code related to MinMax Sampling. https://github.com/Arimase97/MinMax-Sampling.

[2] Matthew K. Mukerjee, David Naylor, Junchen Jiang, Dongsu Han, Srinivasan Seshan, and Hui Zhang. Practical, real-time centralized control for cdn-based live video delivery. In *SIGCOMM 2015*, pages 311–324. ACM, 2015.

[3] Brad Glasbergen, Kyle Langendoen, Michael Abebe, and Khuzaima Daudjee. Chronocache: Predictive and adaptive mid-tier query result caching. In *SIGMOD Conference 2020], June 14-19, 2020*, pages 2391–2406. ACM, 2020.

[4] Rebecca Taft, Irfan Sharif, Andrei Matei, Nathan VanBenschoten, Jordan Lewis, Tobias Grieger, Kai Niemi, Andy Woods, Anne Birzin, Raphael Poss, Paul Bardea, Amruta Ranade, Ben Darnell, Bram Gruneir, Justin Jaffray, Lucy Zhang, and Peter Mattis. Cockroachdb: The resilient geo-distributed SQL database. In *SIGMOD Conference 2020*, pages 1493–1509. ACM, 2020.

[5] Ashish Vulimiri, Carlo Curino, Philip Brighten Godfrey, Thomas Jungblut, Konstantinos Karanasos, Jitendra Padhye, and George Varghese. Wanalytics: Geo-distributed analytics for a data intensive world. In *SIGMOD Conference 2015*, pages 1087–1092. ACM, 2015.

[6] Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons, and Onur Mutlu. Gaia: Geo-distributed machine learning approaching LAN speeds. In *NSDI 2017*, pages 629–647. USENIX Association, 2017.

[7] Ariel Rabkin, Matvey Arye, Siddhartha Sen, Vivek S. Pai, and Michael J. Freedman. Aggregation and degradation in jetstream: Streaming analytics in the wide area. In *NSDI 2014*, pages 275–288. USENIX Association, 2014.

[8] Xin Li, Fang Bian, Mark Crovella, Christophe Diot, Ramesh Govindan, Gianluca Iannaccone, and Anukool Lakhina. Detection and identification of network anomalies using sketch subspaces. In *IMC 2006*, pages 147–152. ACM, 2006.

[9] Qi Zhao, Mitsunori Ogihara, Haixun Wang, and Jun (Jim) Xu. Finding global icebergs over distributed data sets. In *PODS 2006*, pages 298–307. ACM, 2006.

[10] Kiran Maraiya, Kamal Kant, and Nitin Gupta. Wireless sensor network: a review on data aggregation. *International Journal of Scientific & Engineering Research*, 2(4):1–6, 2011.

[11] Amit Manjhi, Suman Nath, and Phillip B. Gibbons. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *SIGMOD Conference 2005*, pages 287–298. ACM, 2005.

[12] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.

[13] Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J. Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *OSDI '14*, pages 583–598. USENIX Association, 2014.

[14] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson C. Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. Spanner: Google's globally-distributed database. In *OSDI 2012*, pages 251–264. USENIX Association, 2012.

[15] Ashish Gupta, Fan Yang, Jason Govig, Adam Kirsch, Kelvin Chan, Kevin Lai, Shuo Wu, Sandeep Govind Dhoot, Abhilash Rajesh Kumar, Ankur Agiwal, Sanjay Bhansali, Mingsheng Hong, Jamie Cameron, Masood Siddiqi, David Jones, Jeff Shute, Andrey Gubarev, Shivakumar Venkataraman, and Divyakant Agrawal. Mesa: Geo-replicated, near real-time, scalable data warehousing. *Proc. VLDB Endow.*, 7(12):1259–1270, 2014.

[16] Ashish Vulimiri, Carlo Curino, Philip Brighten Godfrey, Thomas Jungblut, Jitu Padhye, and George Varghese. Global analytics in the face of bandwidth and regulatory constraints. In *NSDI 15*, pages 323–336. USENIX Association, 2015.

[17] Sara Alspaugh, Bei Di Chen, Jessica Lin, Archana Ganapathi, Marti A. Hearst, and Randy H. Katz. Analyzing log analysis: An empirical study of user log mining. In *LISA '14*, pages 53–68. USENIX Association, 2014.

[18] Yu Chen, Kai Hwang, and Wei-Shinn Ku. Collaborative detection of ddos attacks over multiple network domains. *IEEE Trans. Parallel Distributed Syst.*, 18(12):1649–1662, 2007.

[19] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A. Lee. Awstream: adaptive wide-area streaming analytics. In *SIGCOMM 2018*, pages 236–252. ACM, 2018.

[20] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *ICML 2020*, volume 119, pages 8253–8265. PMLR, 2020.

[21] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Vladimir Braverman, Ion Stoica, and Raman Arora. Communication-efficient distributed SGD with sketching. In *NeurIPS 2019*, pages 13144–13154, 2019.

[22] Moshe Gabel, Assaf Schuster, and Daniel Keren. Communication-efficient distributed variance monitoring and outlier detection for multivariate time series. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, pages 37–47. IEEE Computer Society, 2014.

[23] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, 31(3):9–18, 2002.

[24] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[25] Graham Cormode and Shan Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

[26] Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, pages 693–703. Springer, 2002.

[27] Jizhou Li, Zikun Li, Yifei Xu, Shiqi Jiang, Tong Yang, Bin Cui, Yafei Dai, and Gong Zhang. Wavingsketch: An unbiased and generic sketch for finding top-k items in data streams. In *KDD '20*, pages 1574–1584. ACM, 2020.

[28] Daniel Ting. Data sketches for disaggregated subset sum and frequent item estimation. In *SIGMOD Conference 2018*, pages 1129–1140. ACM, 2018.

[29] Zhewei Wei, Ge Luo, Ke Yi, Xiaoyong Du, and Ji-Rong Wen. Persistent data sketching. In *SIGMOD Conference 2015*, pages 795–810. ACM, 2015.

[30] Daniel Ting. Count-min: Optimal estimation and tight error bounds using empirical error distributions. In *KDD 2018*, pages 2319–2328. ACM, 2018.

[31] Zengfeng Huang, Ke Yi, Yunhao Liu, and Guihai Chen. Optimal sampling algorithms for frequency estimation in distributed data. In *INFOCOM 2011*, pages 1997–2005. IEEE, 2011.

[32] Graham Cormode, Minos N. Garofalakis, S. Muthukrishnan, and Rajeev Rastogi. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *SIGMOD Conference 2005*, pages 25–36. ACM, 2005.

[33] Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *NeurIPS 2018*, pages 4452–4463, 2018.

[34] Gaoxiong Zeng, Wei Bai, Ge Chen, Kai Chen, Dongsu Han, Yibo Zhu, and Lei Cui. Congestion control for cross-datacenter networks. In *ICNP 2019*, pages 1–12. IEEE, 2019.

[35] Anil Shanbhag, Holger Pirk, and Samuel Madden. Efficient top-k query processing on massively parallel hardware. In *SIGMOD Conference 2018*, pages 1557–1570. ACM, 2018.

[36] Hosam M Mahmoud. *Sorting: A distribution theory*, volume 54. John Wiley & Sons, 2011.

[37] Alfréd Rényi. On the theory of order statistics. *Acta Mathematica Academiae Scientiarum Hungarica*, 4(3-4):191–231, 1953.

[38] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *ICLR 2018*. OpenReview.net, 2018.

[39] Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *NeurIPS 2018*, pages 1306–1316, 2018.

[40] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR 2016*, pages 770–778. IEEE Computer Society, 2016.

[42] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[43] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *IJCNN 2017*, pages 2921–2926. IEEE, 2017.

[44] The CAIDA Anonymized Internet Traces. http://www.caida.org/data/overview/.

[45] Yinda Zhang, Zaoxing Liu, Ruixin Wang, Tong Yang, Jizhou Li, Ruijie Miao, Peng Liu, Ruwen Zhang, and Junchen Jiang. Cocosketch: high-performance sketch-based measurement over arbitrary partial key query. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pages 207–222, 2021.

[46] Tayyebeh Jahani-Nezhad and Mohammad Ali Maddah-Ali. Codedsketch: Coded distributed computation of approximated matrix multiplication. In *ISIT 2019*, pages 2489–2493. IEEE, 2019.

[47] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS 2017*, volume 54, pages 1273–1282. PMLR, 2017.

[48] Yujing Chen, Yue Ning, Martin Slawski, and Huzefa Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In *IEEE BigData 2020*, pages 15–24. IEEE, 2020.

[49] Aritra Dutta, El Houcine Bergou, Ahmed M. Abdelmoniem, Chen-Yu Ho, Atal Narayan Sahu, Marco Canini, and Panos Kalnis. On the discrepancy between the theoretical analysis and practical implementations of compressed communication for distributed deep learning. In *AAAI 2020*, pages 3817–3824. AAAI Press, 2020.

[50] Mingxun Zhou, Tianhao Wang, T.-H. Hubert Chan, Giulia Fanti, and Elaine Shi. Locally differentially private sparse vector aggregation. *CoRR*, abs/2112.03449,

2021.

[51] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian U. Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *ICML 2019*, volume 97, pages 3252–3261. PMLR, 2019.

[52] Atal Sahu, Aritra Dutta, Ahmed M Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. Rethinking gradient sparsification as total error minimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[53] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. SIGNSGD: compressed optimisation for non-convex problems. In *ICML 2018*, volume 80, pages 559–568. PMLR, 2018.

[54] Hang Xu, Chen-Yu Ho, Ahmed M Abdelmoniem, Aritra Dutta, El Houcine Bergou, Konstantinos Karatsenidis, Marco Canini, and Panos Kalnis. Grace: A compressed communication framework for distributed machine learning. In *ICDCS 2021*, pages 561–572. IEEE, 2021.

[55] Kelly Kostopoulou, Hang Xu, Aritra Dutta, Xin Li, Alexandros Ntoulas, and Panos Kalnis. Deepreduce: A sparse-tensor communication framework for distributed deep learning. *arXiv preprint arXiv:2102.03112*, 2021.

[56] Yikai Zhao, Zheng Zhong, Yuanpeng Li, Yi Zhou, Yifan Zhu, Li Chen, Yi Wang, and Tong Yang. Cluster-reduce: Compressing sketches for distributed data streams. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2316–2326, 2021.

[57] Jiawei Jiang, Fangcheng Fu, Tong Yang, and Bin Cui. Sketchml: Accelerating distributed machine learning with data sketches. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1269–1284, 2018.

[58] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, 2019.

[59] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.