Coded Decentralized Learning With Gradient Descent for Big Data Analytics

Jing Yue, *Member, IEEE*, and Ming Xiao, *Senior Member, IEEE*

***Abstract*— Machine learning is an effective technique for big data analytics. We focus on the study of big data analytics with decentralized learning in large-scale networks. Fountain codes are applied to the decentralized learning process to reduce communication load for exchanging intermediate learning parameters among fog nodes. Two scenarios, i.e., disjoint datasets and overlapping datasets, are analyzed. Comparison results show that communication load can be reduced significantly by the Fountain-based scheme for large-scale networks, especially when the quality of communication links is relatively bad and/or the number of fog nodes is large.**

***Index Terms*— Big data, decentralized learning, gradient descent, Fountain codes, communication load.**

I. Introduction

THE volume of data increases explosively with the rapid development of social media and Internet of Things (IoTs). Big data gathered by different devices from various sources is stored in a distributed manner. To extract useful information or insights from big data for data-driven decision making, strong processing resources are required. Cloud is able to provide sufficient resources for big data analysis. However, transferring a large amount of distributed stored data to a remote cloud is infeasible due to time or bandwidth limitations [1]. Having data processing close to the sources or devices is the key to overcome these limitations. Thus, distributed machine learning [2] with fog computing [3] is a potential solution for big data analytics.

Coding has been applied to distributed fog computing and machine learning for dealing with the problem of stragglers [4] and reducing the usage of computation and communication resources [5]. For coded distributed machine learning [2], matrix multiplication [6], [7] and gradient descent [8], [9] have attracted considerable attention.

In most of the previous works for distributed machine learning, codes with maximum distance separable (MDS) properties are used [1]. For example, MDS codes were utilized in [2], [10] to speed-up distributed matrix multiplication and data shuffling. In [11] and [12], MDS codes were used to mitigate the effect of stragglers in distributed gradient descent. In [6], scheme based on Reed-Solomon (RS) codes was proposed to minimum the recovery threshold while allowing efficient decoding using polynomial interpolation.

However, in a large-scale network with thousands of nodes, these MDS-based codes become impractical [1], [7] because of high computation and communication costs associated with encoding and decoding [13]. In addition, most of the previous works consider the master-worker pattern. A master assigns computation or learning tasks to workers. Workers complete tasks on a subset of data and return intermediate results to the master for aggregation. In general, it is assumed that the entire data is available for processing at the time of training, etc. Big data breaks these ideal assumptions and leads to the cluster pattern, e.g., master-work pattern, in which a centralized master control task assign, impractical. Big data analytics in decentralized manner is in urgent needed.

We focus on the study of decentralized learning with gradient descent in large-scale networks. It has been proved in [14] that Fountain-based scheme for distributed computing is one of the optimal choices for the applications with high reliability and low latency requirements. Fountain codes [15] are capable of recovering the original information from any subset of output symbols with a size slightly larger than the original information. In addition, Fountain codes are of high coding flexibility and low complexity.

High communication load is the bottleneck of decentralized processing for big data analytics. Therefore, we consider applying Fountain codes to the decentralized learning process to reduce communication load. Our main contributions are listed as follows,

- A novel coded decentralized scheme based on Fountain codes is proposed to reduce the communication load for exchanging intermediate leaning parameters among fog nodes during the learning process.
- Two situations, i.e., disjoint datasets and overlapping datasets, are studied. The decoding process of the Fountain-based scheme is analyzed for the two situations.
- Finally, comparison results are given to show that communication load can be reduced significantly by using the Fountain-based scheme through proper code design.

To the best of our knowledge, we are the first to use Fountain codes for the design of coded decentralized learning and also the first to analyze the Fountain-based scheme for the situations with disjoint datasets and overlapping datasets.

II. System Model

We consider a network with F fog nodes and S storage units, as shown in Fig. 1. The fog nodes, which could be

Manuscript received June 14, 2019; accepted July 14, 2019. Date of publication July 23, 2019; date of current version February 11, 2020. The work of Jing Yue was supported by Swedish Strategic Research Foundation (SSF) project "Cyber Security for Next Generation Factory" (SSF grant RIT17-0032). The work of Ming Xiao is supported in part by SSF project "High-reliable Low-latency Industrial Wireless Communications" and in part by EU Marie Sklodowska-Curie Actions project "High-reliability Low-latency Communications with network coding", and ERA-NET, "SMART-MLA". The associate editor coordinating the review of this letter and approving it for publication was S. Yu. *(Corresponding author: Jing Yue.)*

J. Yue is with the Reliable Wireless Lab, RISE Research Institutes of Sweden, 22370 Lund, Sweden (e-mail: jing.yue@ri.se).

M. Xiao is with the Division of ISE, School of EECS, KTH Royal Institute of Technology, 11428 Stockholm, Sweden (e-mail: mingx@kth.se).

Digital Object Identifier 10.1109/LCOMM.2019.2930513

1558-2558 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: KTH Royal Institute of Technology. Downloaded on October 28,2020 at 20:41:08 UTC from IEEE Xplore. Restrictions apply.

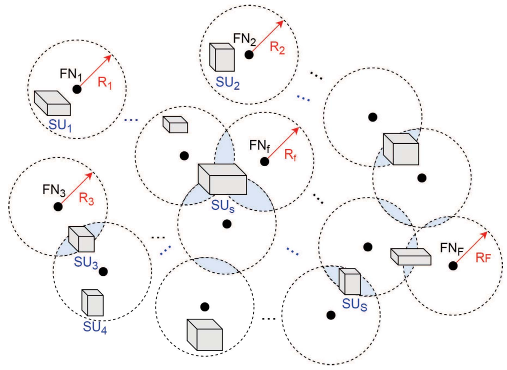

Fig. 1. Distributed fog nodes and their available storage units.

mobile devices or edge servers, etc., are with limited computation and/or storage capacity. Let FN_f represent the fth fog node and SU_s be the sth storage unit, $f \in \{1, 2, \ldots, F\}$ and $s \in \{1, 2, \ldots, S\}$. The fog nodes are scattered over a wide area. The storage units are in geo-separated locations. In Fig. 1, we use cubes with different shapes to represent the variety of data stored in different storage units.

If a fog node is able to access to a storage unit, the storage unit is said to be available for the fog node. The storage units, which are located in a spherical space with fog node FN_f as the center and radius of R_f, are all available to FN_f. That is, FN_f is able to access to and process the data in the storage units located in its spherical space. This spherical space is named the available space of FN_f. Let $\mathbf{D}_f$ be the dataset that FN_f will process for completing a computation or learning task. The data in dataset $\mathbf{D}_f$ is stored in the storage units located in FN_f available space. The size of $\mathbf{D}_f$ is $|\mathbf{D}_f|$. Each fog node processes the data in its dataset and exchanges intermediate results with other fog nodes to complete common computation or learning tasks.

There are two possible scenarios, i.e., disjoint datasets and overlapping datasets. Let $\gamma_{a,b}$ represent the ratio of overlapping data in storage unit SU_a with SU_b, $(0 \le \gamma_{a,b} \le 1)$. The parameter $\gamma_{a,b}$ is proportional to the number of overlapping data in SU_a and SU_b. $\gamma_{a,b} = 0$, $(\forall a, b \in \{1, 2, \ldots, S\}, a \neq b)$, corresponds to the scenario of disjoint, and $\gamma_{a,b} > 0$ corresponds to overlapping. Fog nodes, which access to the storage units with overlapping data, have overlapping datasets. The overlapping parameters of these datasets can be obtained according to the storage units which the fog nodes access to for forming their datasets.

III. Fountain-Based Decentralized Gradient Descent

In this section, a brief introduction of the distributed gradient descent algorithm is given. Then, decentralized learning with gradient descent based on Fountain codes is proposed.

A. Distributed Gradient Descent

Consider a scenario, where we want to learn model parameters $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_W)$ by minimizing a generic loss function $L(\mathbf{D}; \boldsymbol{\beta}) = \sum_{i=1}^{N} \ell(x_i, y_i; \boldsymbol{\beta})$ over a given dataset $\mathbf{D} = \{(x_i, y_i) | i \in \{1, 2, \ldots, N\}\}$. The model parameters $\boldsymbol{\beta}$ are obtained through multiple iterations of operations. For distributed gradient descent, in the tth iteration, workers calculate gradients on their own datasets and send the intermediate gradients to the master. The full gradient $\nabla L^{(t)} = \sum_{f=1}^{F} \sum_{(x,y)\in \mathbf{D}_f} \nabla \ell(x, y; \boldsymbol{\beta}^{(t-1)})$ is aggregated at the master. After that, the master updates parameters through $\boldsymbol{\beta}^{(t)} = h(\boldsymbol{\beta}^{(t-1)}, \nabla L^{(t)})$ and sends the updated parameters $\boldsymbol{\beta}^{(t)}$ to workers for the calculation of gradients in the $t+1$th iteration. $h(\cdot)$ is a gradient-based optimizer [16].

B. Decentralized Learning With Gradient Descent

For decentralized learning with gradient descent, three phases, i.e., computation, exchanging and decoding phases, are carried out in each iteration. We take the tth iteration as an example to illustrate the decentralized learning process.

1) Computation Phase: Each fog node updates its model parameters and calculates the gradients through training its dataset. At fog node FN_f, the intermediate gradients and model parameters after training all the data in $\mathbf{D}_f$ can be written as $\mathbf{g}_f = (g_{f,1}, g_{f,2}, \ldots, g_{f,|\mathbf{D}_f|})$ and $\boldsymbol{\beta}_f = (\beta_{f,1}, \beta_{f,2}, \ldots, \beta_{f,W_f})$, respectively. $g_{f,a} = \nabla \ell(x_a, y_a; \boldsymbol{\beta}_f)$, $(x_a, y_a) \in \mathbf{D}_f$, $a \in \{1, 2, \ldots, |\mathbf{D}_f|\}$. W_f represents the length of model parameter learned at fog node FN_f.

Then, encoding operations are performed on $\mathbf{g}_f$ and $\boldsymbol{\beta}_f$, respectively. The encoding process for $\mathbf{g}_f$ is as follows:

- A number d_g is selected according to degree distribution $\boldsymbol{\Omega}(x) = \sum_{d_g=1}^{|\mathbf{D}_f|} \Omega_{d_g} x^{d_g}$ with probability Ω_{d_g}.
- Then, d_g intermediate gradients are selected uniformly at random from $\mathbf{g}_f$ to perform combination to form one coded intermediate gradient.
- The above two steps repeated until $Q_f^g = (1 + \eta_f)|\mathbf{D}_f|$ coded intermediate gradients are formed. The formed coded intermediate gradients $\mathbf{c}_f^g$ can be written as

$$\mathbf{c}_f^g = (g_{f,1}, g_{f,2}, \ldots, g_{f,|\mathbf{D}_f|})\mathbf{G}_f^g = \mathbf{g}_f \mathbf{G}_f^g, \qquad (1)$$

where $\mathbf{G}_f^g$ is the generator matrix at fog node FN_f with size $|\mathbf{D}_f| \times Q_f^g$. $1+\eta_f$ represents the expanding coefficient of the Fountain codes used, $1 + \eta_f = \frac{Q_f^g}{|\mathbf{D}_f|}$ and $\eta_f \ge 0$.

The encoding process for $\boldsymbol{\beta}_f$ is the same as that for $\mathbf{g}_f$. The only difference is that encoding operations are performed according to degree distribution $\boldsymbol{\mu}(x) = \sum_{d_w=1}^{W_f} \mu_{d_w} x^{d_w}$. The formed $Q_f^w = (1 + \eta_f)W_f$ coded intermediate model parameters can be written as $\mathbf{c}_f^w = \boldsymbol{\beta}_f \mathbf{G}_f^w$. $\mathbf{G}_f^w$ is the generator matrix at FN_f with size $W_f \times Q_f^w$.

2) Exchanging Phase: The coded intermediate parameters $\mathbf{c}_f^g$ and $\mathbf{c}_f^w$ $(f \in \{1, 2, \ldots, N\})$ are exchanged among fog nodes. Let M be the total number of all different data in F datasets, and $M \le \sum_{f=1}^{F} |\mathbf{D}_f|$. $M = \sum_{f=1}^{F} |\mathbf{D}_f|$ if F datasets are disjoint, and $M < \sum_{f=1}^{F} |\mathbf{D}_f|$ if these datasets are overlapping.

3) Decoding Phase: The generator matrices correspond to the received coded intermediate gradients and model parameters from fog node FN_i $(i \in \{1, 2, \ldots, F\} \setminus \{f\})$ at FN_f are $\tilde{\mathbf{G}}_{i,f}^g$ with size $|\mathbf{D}_i| \times Q_{i,f}^g$ and $\tilde{\mathbf{G}}_{i,f}^w$ with size $W_i \times Q_{i,f}^w$, respectively, where $Q_{i,f}^g = (1 - \epsilon_{i,f})Q_i^g$ and $Q_{i,f}^w = (1 - \epsilon_{i,f})Q_i^w$. $\epsilon_{i,f}$ is the probability of data

Authorized licensed use limited to: KTH Royal Institute of Technology. Downloaded on October 28,2020 at 20:41:08 UTC from IEEE Xplore. Restrictions apply.

loss from FN_i to FN_f due to channel erasure, decoding failure, etc. The generator matrices correspond to the received coded intermediate gradients and model parameters from the $F-1$ fog nodes at FN_f can be written as $\tilde{\mathbf{G}}_f^g = [\mathbf{1}_1\tilde{\mathbf{G}}_{1,f}^g, \ldots, \mathbf{1}_{f-1}\tilde{\mathbf{G}}_{f-1,f}^g, \mathbf{1}_{f+1}\tilde{\mathbf{G}}_{f+1,f}^g, \ldots, \mathbf{1}_F\tilde{\mathbf{G}}_{F,f}^g]$ and $\tilde{\mathbf{G}}_f^w = [\mathbf{1}_1\tilde{\mathbf{G}}_{1,f}^w, \ldots, \mathbf{1}_{f-1}\tilde{\mathbf{G}}_{f-1,f}^w, \mathbf{1}_{f+1}\tilde{\mathbf{G}}_{f+1,f}^w, \ldots, \mathbf{1}_F\tilde{\mathbf{G}}_{F,f}^w]$, respectively. $\mathbb{1} = (\mathbf{1}_1, \ldots, \mathbf{1}_f, \ldots, \mathbf{1}_F)$ is an indicator parameter. Let λ be the probability of a straggler node. Then, the fth element of the indicator parameter, $\mathbf{1}_f$ ($f \in \{1, 2, \ldots, F\}$), can be written as

$$\mathbf{1}_f = \begin{cases} 1, & \text{with probability } 1-\lambda, \\ 0, & \text{with probability } \lambda. \end{cases} \tag{2}$$

The matrices $\tilde{\mathbf{G}}_f^g$ and $\tilde{\mathbf{G}}_f^w$ are respectively with size $\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} |\mathbf{D}_i| \times \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_f Q_{i,f}^g$ and $\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} W_i \times \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_f Q_{i,f}^w$.

Fog node FN_f decodes its received coded intermediate parameters through $\tilde{\mathbf{G}}_{i,f}^g$ and $\tilde{\mathbf{G}}_{i,f}^w$, ($i \in \{1, 2, \ldots, F\} \setminus \{f\}$), and recoveries $N - |\mathbf{D}_f|$ new gradients and $\Gamma_w \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} W_i$ model parameters corresponding to these intermediate gradients. Here Γ_w is a threshold and could be set to different values in $[0, 1]$ for different learning algorithms[1]. The decoding process is essentially a linear calculation process for finding the solution of polynomials.

IV. Analysis of the Decoding Process of the Fountain-Based Scheme

In this section, the decoding processes for the scenarios with disjoint datasets and overlapping datasets are analyzed separately. The effect of coding and network parameters on the computation and communication loads is also analyzed.

A. For Disjoint Datasets

For the scenario with disjoint datasets, the possible number of new intermediate gradients that can be recovered at fog node FN_f is in the range of $[0, M - |\mathbf{D}_f|]$. To decode $A = N - |\mathbf{D}_f|$ new intermediate gradients from the received coded data, the rank of the generator matrix $\tilde{\mathbf{G}}_f^g$, denoted by $rk(\tilde{\mathbf{G}}_f^g)$, must be equal to or larger than A, i.e., $rk(\tilde{\mathbf{G}}_f^g) \geq A$. Since the datasets for different fog nodes are disjoint, we have $rk(\tilde{\mathbf{G}}_f^g) = \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_i rk(\tilde{\mathbf{G}}_{i,f}^g)$.

The probability that $rk(\tilde{\mathbf{G}}_{i,f}^g) = k_i$, $k_i \in \{0, 1, \ldots, \min\{|\mathbf{D}_i|, Q_{i,f}^g\}\}$, can be calculated using the method given in [17]. It can be written as $P_r(rk(\tilde{\mathbf{G}}_{i,f}^g) = k_i) = \sum_{j=1}^{Q_i^g} \binom{Q_i^g}{j} (1-\epsilon_{i,f})^j (\epsilon_{i,f})^{Q_i^g - j} \zeta_{k_i}^{j,|\mathbf{D}_i|}$, where $\zeta_{k_i}^{j,|\mathbf{D}_i|}$ can be calculated by

$$\zeta_{k_i}^{j,|\mathbf{D}_i|} = \frac{\xi_{k_i}^j \xi_{k_i}^{|\mathbf{D}_i|}}{\xi_{k_i}^{k_i} q^{(j-k_i)(|\mathbf{D}_i|-k_i)}}, \tag{3}$$

where ξ_m^n can be written as

$$\xi_m^n = \begin{cases} \prod_{a=0}^{m+1} (1 - q^{-(n-a)}), & \text{if } m > 0, \\ 1, & \text{if } m = 0. \end{cases} \tag{4}$$

[1]As far as we known, it is still an open problem for the setting of optimal values Γ_w for different learning algorithms considering different factors. As it is beyond the scope of this article, this problem will be studied in depth in our future works.

Let $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_i k_i \geq A\right)$ be the probability that at least A new intermediate gradients can be recovered at fog node FN_f. Since the datasets are disjoint, we have $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_i k_i\right) = \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_f P_r(k_i)$.

As $N \leq M$, there are two possible situations, i.e., $N = M$ and $N < M$.

1) For $N = M$: If $N = M$, the probability $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_i k_i > A\right) = 0$. For updating parameters $\boldsymbol{\beta}_f$ at FN_f, each generator matrix $\tilde{\mathbf{G}}_{i,f}^g$ must be full rank and no straggler occurs, i.e., $k_i = |\mathbf{D}_i|$ ($i \in \{1, 2, \ldots, F\} \setminus \{f\}$) and $\lambda = 0$, and the probability $P_r\left(rk(\tilde{\mathbf{G}}_{i,f}^w) \geq \Gamma_w W_i\right)$ for each $k_i > 0$ should be bigger than a threshold, say Γ. The threshold Γ can be set to different values according to the various requirements in different applications and the corresponding learning algorithm used.

The probability that all the generator matrices at FN_f, i.e., $\tilde{\mathbf{G}}_{i,f}^g$, are full rank with $\lambda = 0$ can be written as $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} k_i = A\right) = \sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} P_r(rk(\tilde{\mathbf{G}}_{i,f}^g) = |\mathbf{D}_i|)$. The probability $P_r\left(rk(\tilde{\mathbf{G}}_{i,f}^w) \geq \Gamma_w W_i\right)$ can be obtained by calculating $\sum_{k_i'=\Gamma_w W_i}^{W_i} P_r\left(rk(\tilde{\mathbf{G}}_{i,f}^w) = k_i'\right)$.

2) For $N < M$: We use $\alpha_i A$ to represent k_i, i.e., $k_i = \alpha_i A$. All $F-1$ elements form set $\boldsymbol{\Phi} = \{\alpha_1, \alpha_2, \ldots, \alpha_F\} \setminus \{\alpha_f\}$. Then we have $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\backslash\{f\}} \mathbf{1}_i k_i \geq A\right) = 1 - P_r\left(\sum_{\alpha_i\in\boldsymbol{\Phi}} \mathbf{1}_i \alpha_i < 1\right)$, where $P_r\left(\sum_{\alpha_i\in\boldsymbol{\Phi}} \mathbf{1}_i \alpha_i < 1\right)$ can be calculated by

$$P_r\left(\sum_{\alpha_i\in\boldsymbol{\Phi}} \mathbf{1}_i \alpha_i < 1\right) = \begin{cases} 1, & \lambda = 1, \\ \sum_{\boldsymbol{\alpha_\tau}} \prod_{j=1}^{F-1} P_r(\alpha_{\tau_j}), & \lambda < 1, \end{cases} \tag{5}$$

where $\boldsymbol{\alpha_\tau}$ is set formed by all possible values of $\alpha_{\boldsymbol{\tau}}$ and $\boldsymbol{\alpha_\tau} = \left\{\alpha_{\boldsymbol{\tau}} \,|\, \arg\{\sum_{j\in\{1,2,\ldots,(1-\lambda)(F-1)\}} \alpha_{\tau_j} < 1\}, \alpha_{\tau_j} \in \left[0, \frac{|\mathbf{D}_{\tau_j}|}{A}\right)\right\}$. α_{τ_j} is the τ_jth element in set $\boldsymbol{\Phi}$. The probability $P_r(\alpha_{\tau_j})$ can be calculated by $P_r(\alpha_{\tau_j}) = P_r(rk(\tilde{\mathbf{G}}_{\tau_j,f}^g) = \alpha_{\tau_j} A)$.

The range of α_i is $\left[0, \frac{\min\{|\mathbf{D}_i|, Q_{i,f}^g\}}{A}\right]$, i.e., $\left[0, \frac{|\mathbf{D}_i|\cdot\min\{1,(1-\epsilon_{i,f})(1+\eta_i)\}}{A}\right]$. Thus, we obtain the value range of $\sum_{\alpha_i\in\boldsymbol{\Phi}} \alpha_i$ as $\left[0, \frac{\sum_{\alpha_i\in\boldsymbol{\Phi}}(|\mathbf{D}_i|\cdot\min\{1,(1-\epsilon_{i,f})(1+\eta_i)\})}{A}\right]$.

If $(1-\epsilon_{i,f})(1+\eta_i) \geq 1$, then we have the upper bound of the value range as $\frac{\sum_{\alpha_i\in\boldsymbol{\Phi}} |\mathbf{D}_i|}{A} > 1$ since $M > N$. The value range $[0, 1)$ is a subarea of the whole value range of $\sum_{\alpha_i\in\boldsymbol{\Phi}} \alpha_i$. The value of $\sum_{\boldsymbol{\alpha_\tau}} \prod_{j=1}^{F-1} P_r(\alpha_{\tau_j})$ is determined by the size of datasets of the other fog nodes. A larger size of datasets means smaller value of $\sum_{\boldsymbol{\alpha_\tau}} \prod_{j=1}^{F-1} P_r(\alpha_{\tau_j})$ and higher probability of success update model parameters $\boldsymbol{\beta}_f$ at fog node FN_f. However, the data size cannot be infinitely large. Larger dataset size corresponds to possible higher computation load at each fog node. Here, communication load is defined as the total number of fog nodes computing one common task.

If $(1-\epsilon_{i,f})(1+\eta_i) < 1$, then the upper bound of the value range can be written as $\frac{\sum_{\alpha_i\in\boldsymbol{\Phi}} |\mathbf{D}_i|(1-\epsilon_{i,f})(1+\eta_i)}{A}$. If we keep the other parameters, e.g., $|\mathbf{D}_i|$ and $\epsilon_{i,f}$, unchanged, the whole value range of $\sum_{\alpha_i\in\boldsymbol{\Phi}} \alpha_i$ extends with increasing η_i. Therefore, with increasing η_i, the subarea $[0, 1)$ becomes relatively

Authorized licensed use limited to: KTH Royal Institute of Technology. Downloaded on October 28,2020 at 20:41:08 UTC from IEEE Xplore. Restrictions apply.

smaller, and the value of $\sum_{\sum_{i\in\{1,2,\ldots,F-1\}} a_i<1}\prod_j P_r(\alpha_i = a_i)$ becomes smaller as well. The probability of success update model parameters $\boldsymbol{\beta}_f$ at fog node FN_f becomes higher. Therefore, larger η_i means a higher probability of success update model parameters $\boldsymbol{\beta}_f$. However, η_i cannot be infinite large. Larger η_i corresponds to higher communication load for exchanging intermediate parameters among fog nodes. For success update model parameters $\boldsymbol{\beta}_f$, we have to choose the coding parameter η_i carefully to ensure $\sum_{\alpha_i\in\boldsymbol{\Phi}}|\mathbf{D}_i|(1-\epsilon_{i,f})(1+\eta_i) > A$.

At each fog node, before decoding its received coded parameters, the probabilities $P_r\left(\sum_{i\in\{1,2,\ldots,F\}\setminus\{f\}}\mathbf{1}_i k_i \geq A\right)$ and $P_r\left(rk(\tilde{\mathbf{G}}^w_{i,f}) \geq \Gamma_w W_i\right)$ for each $k_i > 0$ should be compared with threshold Γ. If these probabilities are higher than Γ, decoding starts on the received coded parameters.

B. For Overlapping Datasets

As defined in Section II, $\gamma_{a,b}$ is the ratio of overlapping data in storage unit SU_a with that in SU_b. For simplification, we assume that each storage unit corresponds to one fog node. Then, the overlapping parameters for the datasets of the F fog nodes can be written out in matrix form as follows,

$$\boldsymbol{\gamma} = \begin{bmatrix} 1 & \gamma_{1,2} & \cdots & \gamma_{1,F} \\ \gamma_{2,1} & 1 & \cdots & \gamma_{2,F} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{F,1} & \gamma_{F,2} & \cdots & 1 \end{bmatrix}, \tag{6}$$

where $\gamma_{a,b} \neq \gamma_{b,a}$ $(a\neq b, \forall a,b\in\{1,2,\ldots,F\})$.

At FN_f, $|\mathbf{D}_f|$ intermediate gradients are known. Thus $A = N-|\mathbf{D}_f|$ new intermediate gradients are required for updating model parameters $\boldsymbol{\beta}_f$.

Theorem 1: The total number of new intermediate gradients received from the other fog nodes at FN_f can be calculated by $\Delta = \sum_{\pi_i,i=1}^{F-1}\mathbf{1}_{\pi_i}\cdot((1-\gamma_{\pi_i,f})\varphi(i,f))\cdot|\mathbf{D}_{\pi_i}|$, where the parameter $\varphi(i,f)$ can be written as

$$\varphi(i,f) = \begin{cases} 1, & \text{if } i = 1, \\ \prod_{a=1}^{i-1}(1-\gamma_{\pi_i,\pi_i-\pi_a|\boldsymbol{\Theta}_{a,f}}), & \text{if } 2\leq i\leq F-1, \end{cases}$$

where $\boldsymbol{\Theta}_{a,f}$ is a set formed by the indices of fog nodes, and it can be written as

$$\boldsymbol{\Theta}_{a,f} = \begin{cases} \{f\}, & \text{if } a = 1, \\ \{f,\pi_1,\ldots,\pi_{a-1}\}, & \text{if } a > 1. \end{cases}$$

Proof: Let r_i be the parameter for calculating the number of new intermediate gradients from fog node FN_i at FN_f, where $i\in\{1,2,\ldots,F\}\setminus\{f\}$, and we have $\mathbf{r} = (r_1,\ldots,r_{f-1},r_{f+1},\ldots,r_F)$. The total number of new intermediate gradients at FN_f can be written as

$$\begin{aligned}\Delta &= \mathbb{1}\odot\mathbf{r}\odot[|\mathbf{D}_1|,|\mathbf{D}_2|,\ldots,|\mathbf{D}_F|] \\ &= \sum_{i\in\{1,2,\ldots,F\}\setminus\{f\}}\mathbf{1}_i r_i|\mathbf{D}_i|. \end{aligned}\tag{7}$$

Next, we will calculate r_i. The number of received coded intermediate gradient from FN_i at FN_f is $Q^g_{i,f}$, where $Q^g_{i,f} = (1-\epsilon_{i,f})Q^g_i = (1-\epsilon_{i,f})(1+\eta_i)|\mathbf{D}_i|$. Thus, we have $|\mathbf{D}_i| = \frac{Q^g_{i,f}}{(1-\epsilon_{i,f})(1+\eta_i)}$. We calculate the new intermediate gradient from the other fog nodes in order $\boldsymbol{\pi}$. Here $\boldsymbol{\pi} = (\pi_1,\pi_2,\ldots,\pi_{F-1})$ is an order of the other $F-1$ fog nodes, $\pi_i\in\{1,2,\ldots,F-1\}\setminus\{f\}$. The number of new intermediate gradients from FN_{π_1} at FN_f is $(1-\gamma_{\pi_1,f})\frac{Q^g_{\pi_1,f}}{(1-\epsilon_{\pi_1,f})(1+\eta_{\pi_1})}$. Thus, the parameter $r_{\pi_1} = \frac{1-\gamma_{\pi_1,f}}{(1-\epsilon_{\pi_1,f})(1+\eta_{\pi_1})}$. Ticking out the new intermediate gradients contributed by FN_{π_1}, the number of new intermediate gradients from FN_{π_2} at FN_f is $(1-\gamma_{\pi_2,\pi_1|f})(1-\gamma_{\pi_2,f})\frac{Q^g_{\pi_2,f}}{(1-\epsilon_{\pi_2,f})(1+\eta_{\pi_2})}$, and we have $r_{\pi_2} = \frac{(1-\gamma_{\pi_2,\pi_1|f})(1-\gamma_{\pi_2,f})}{(1-\epsilon_{\pi_2,f})(1+\eta_{\pi_2})}$. $\gamma_{\pi_2,\pi_1|f}$ represents the overlapping ratio of the dataset $\mathbf{D}_{\pi_2}$ with dataset $\mathbf{D}_{\pi_1}$ after ticking out their common data with dataset $\mathbf{D}_f$.

By the same method, we have

$$r_{\pi_i} = \frac{\omega_{\pi_i,f}}{(1-\epsilon_{\pi_i,f})(1+\eta_{\pi_i})}, \tag{8}$$

where $\omega_{\pi_i,f} = (1-\gamma_{\pi_i,\pi_i-1|f,\pi_1,\ldots,\pi_{i-2}})(1-\gamma_{\pi_i,\pi_{i-2}|f,\pi_1,\ldots,\pi_{i-3}})\cdot\cdots\cdot(1-\gamma_{\pi_i,\pi_1|f})(1-\gamma_{\pi_i,f})$. Thus, the total number of new intermediate gradients from other fog nodes at fog node FN_f can be rewritten as $\Delta = \sum_{\pi_i,i=1}^{F-1}\mathbf{1}_{\pi_i}\cdot((1-\gamma_{\pi_i,f})\varphi(i,f))\cdot|\mathbf{D}_{\pi_i}|$. ■

If the parameters $\gamma_{i,i-1|f,1,\ldots,i-2}$, $\gamma_{i,i-2|f,1,\ldots,i-3}$, $\ldots$, $\gamma_{i,f}$ are known at each fog node, the parameter Δ can be calculated. The computation and communication loads can be controlled relatively small through properly task assignment. If these parameters are unknown at fog nodes, the computation and communication loads may increase.

V. Comparison and Discussion

In what follows, we shall compare the communication load by using the Fountain-based coded scheme with the uncoded scheme for decentralized learning. Communication load is defined as the ratio of the total number of data transmitted by all the fog nodes to the data required at these fog nodes. Here "data required at these fog nodes" means the total number of new intermediate parameters required at all the fog nodes. Uncoded scheme means that no coding operation is performed on the intermediate parameters at each fog node.

The communication load for the uncoded and the Fountain-based coded schemes can be calculated respectively by

$$B_{uncoded} = \frac{1}{1-\lambda}\sum_{f=1}^{F}\mathbf{1}_i\frac{|\mathbf{D}_f|l_g + W_f l_w}{\prod_{i\in\{1,2,\ldots,F\}\setminus\{f\}}(1-\epsilon_{i,f})}, \tag{9}$$

$$B_{coded} = \rho_{\lambda,\gamma}\sum_{f=1}^{F}\mathbf{1}_i(1+\eta_f)(|\mathbf{D}_f|l_g + W_f l_w), \tag{10}$$

where l_g and l_w represent the length of one intermediate gradient and one model parameter, respectively. The parameter $\rho(\lambda,\boldsymbol{\gamma})$ can be written as

$$\rho_{\lambda,\gamma} = \begin{cases} \dfrac{1}{1-\lambda}, & \text{if } \boldsymbol{\gamma} \text{ is unknown or } \forall\gamma_{a,b} < \kappa, \\ 1, & \text{if } \boldsymbol{\gamma} \text{ is known and } \forall\gamma_{a,b}\geq\kappa, \end{cases} \tag{11}$$

where $\forall\gamma_{a,b}$ represents the overlapping parameter of datasets of two arbitrary fog nodes FN_a and FN_b, $(a,b \in$

Authorized licensed use limited to: KTH Royal Institute of Technology. Downloaded on October 28,2020 at 20:41:08 UTC from IEEE Xplore. Restrictions apply.

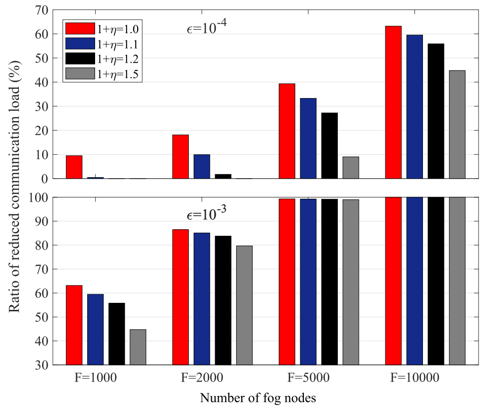

Fig. 2. An example of the comparison of communication load reduction by the Fountain-based coded scheme.

$\{1, 2, \ldots, F\}$). κ is a threshold for the parameter $\forall\gamma_{a,b}$[2]. When $\forall\gamma_{a,b}$ is known and large enough, there would be enough redundancy of coded data to overcome stragglers by using the coded scheme and thus we have $\rho_{\lambda,\gamma} = 1$.

We assume that the coding parameter $\eta_f = \eta$ and $1 + \eta = (1 + \eta^r)(1 + \eta^c)$. Here $1 + \eta^c$ is the expanding coefficient for covering all original information for encoding, and $1 + \eta^r$ is the expanding coefficient confronting data loss. The parameter $1 + \eta^r = \frac{1}{1-\epsilon_w}$, where $\epsilon_w = \max\{\epsilon_{i,f} | i \in \{1, 2, \ldots, F\} \setminus \{f\}\}$. Assume that the data loss parameter $\epsilon_{i,f} = \epsilon$ ($i \in \{1, 2, \ldots, F\} \setminus \{f\}$). The ratio of B_{coded} and $B_{uncoded}$ can be written as

$$\frac{B_{coded}}{B_{uncoded}} = \rho_{\lambda,\gamma}(1 - \lambda)(1 + \eta^c)(1 - \epsilon)^{F-2}. \tag{12}$$

The parameter ϵ represents the probability of data loss due to channel erasure, channel decoding failure, etc. A smaller value of ϵ means more reliable environments. The parameter η^c is determined by the code design. For MDS codes, η^c could be as small as 0. However, it is impractical to construct MDS codes for large-scale networks with thousands of nodes. For Fountain codes, η^c is determined by the degree distributions used for encoding. By properly design the degree distributions, the original information could be covered with $1 + \eta^c$ only slightly larger than 1. The overhead of Fountain codes could be relatively small through properly design.

When $\lambda = 0$, $\epsilon = 0$, there is no straggling node and data loss. Since $\eta^c > 0$, the coding operations create redundancy data at fog nodes, the uncoded scheme requires lower communication load for exchanging intermediate parameters. When $0 < \lambda < 1$ or/and $0 < \epsilon < 1$, compared with the uncoded scheme, the coded scheme can reduce the communication load at $1 - \rho_{\lambda,\gamma}(1 - \lambda)(1 + \eta^c)(1 - \epsilon)^{F-2}$ scale if $1 + \eta^c < \frac{1}{\rho_{\lambda,\gamma}(1-\lambda)(1-\epsilon)^{F-2}}$.

Fig. 2 shows the minimum communication load reduction by using the Fountain-based coded scheme when $\rho_{\lambda,\gamma} = \frac{1}{1-\lambda}$. Note that $\eta = 0$, i.e., $1 + \eta = 1.0$, corresponds to the performance achieved by the MDS-based scheme with ideal quality of communication links. If the environment is relatively reliable and the number of fog nodes F is relatively small, e.g., $\epsilon = 10^{-4}$ and $F = 1000$, only limited communication load reduction can be achieved. When the environment becomes less reliable or F becomes larger, e.g., $\epsilon = 10^{-3}$ or $F \geq 5000$, significant communication load reduction can be achieved by the coded scheme through properly code design.

[2]So far as we know, the setting of optimal threshold κ is still an open problem. This problem will be studied in depth in our future works.

VI. Conclusion

We studied decentralized learning with gradient descent for big data analytics in large-scale networks. A Fountain-based coded scheme was proposed to reduce communication load for exchanging intermediate learning parameters. The decoding process was analyzed for the scenarios with disjoint datasets and overlapping datasets. Comparison results shown that significant communication load reduction can be achieved by the Fountain-based scheme through proper code design.

Acknowledgment

The authors would like to thank Dr. Urban Bilstrup, Manager of Reliable Wireless Lab at RISE, Sweden, for his valuable suggestions and great assistance to this work.

References

[1] A. L'Heureux, K. Grolinger, H. F. Elyamany, and M. Capretz, "Machine learning with big data: Challenges and approaches," *IEEE Access*, vol. 5, pp. 7776–7797, 2017.

[2] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1514–1529, Mar. 2018.

[3] *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things are White Paper*, Cisco, San Jose, CA, USA, pp. 1–6, 2015.

[4] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 109–128, Jan. 2018.

[5] J. Yue, M. Xiao, and Z. Pang, "Distributed fog computing based on batched sparse codes for industrial control," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4683–4691, Oct. 2018.

[6] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: An optimal design for high-dimensional coded matrix multiplication," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4403–4413.

[7] T. Baharav, K. Lee, O. Ocal, and K. Ramchandran, "Straggler-proofing massive-scale distributed matrix multiplication with D-dimensional product codes," in *Proc. IEEE ISIT*, Vail, CO, USA, Jun. 2018, pp. 1993–1997.

[8] Z. Charles and D. Papailiopoulos, "Gradient coding using the stochastic block model," in *Proc. IEEE ISIT*, Vail, CO, USA, Jun. 2018, pp. 1998–2002.

[9] W. Halbawi, N. Azizan, F. Salehi, and B. Hassibi, "Improving distributed gradient descent using Reed–Solomon codes," in *Proc. IEEE ISIT*, Vail, CO, USA, Jun. 2018, pp. 2027–2031.

[10] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," in *Proc. ISIT*, Barcelona, Spain, Jul. 2016, pp. 1143–1147.

[11] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding," Dec. 2016, *arXiv:1612.03301*. [Online]. Available: https://arxiv.org/abs/1612.03301

[12] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic MDS codes and expander graphs," Jul. 2017, *arXiv:1707.03858*. [Online]. Available: https://arxiv.org/abs/1707.03858

[13] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp. 1757–1766, Nov. 1997.

[14] A. Severinson, A. G. I. Amat, and E. Rosnes, "Block-diagonal and LT codes for distributed computing with straggling servers," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 1739–1753, Mar. 2019.

[15] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proc. ACM SIGCOMM*, Vancouver, BC, Canada, Sep. 1998, pp. 56–67.

[16] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in synchronous gradient descent," Mar. 2017, *arXiv:1612.03301v2*. [Online]. Available: https://arxiv.org/abs/1612.03301v2

[17] H. H. F. Yin, S. Yang, Q. Zhou, and L. M. L. Yung, "Adaptive recoding for BATS codes," in *Proc. IEEE ISIT*, Barcelona, Spain, Jul. 2016, pp. 2349–2353.

Authorized licensed use limited to: KTH Royal Institute of Technology. Downloaded on October 28,2020 at 20:41:08 UTC from IEEE Xplore. Restrictions apply.