BYZANTINE-RESILIENT DECENTRALIZED COLLABORATIVE LEARNING

*Jian Xu, Shao-Lun Huang**

Tsinghua-Berkeley Shenzhen Institute, Tsinghua University

ABSTRACT

Decentralized learning techniques have been increasingly popular for model training on distributed worker nodes. Unfortunately, such learning systems are vulnerable to failures and attacks. In this paper, we consider the decentralized learning problem over communication networks, in which worker nodes collaboratively train a machine learning model by exchanging model parameters with neighbors, but a fraction of nodes are corrupted by a Byzantine attacker and could conduct malicious attacks. Our key idea for mitigating Byzantine attacks is to check the direction and magnitude of the *cross-update* vectors (the difference between each received model and local model from the previous round) at each consensus round. We propose a similarity-based reweighting scheme to obtain a robust local model update for each worker. Our proposed method does not need to know the exact number of Byzantine nodes and can be employed in both static and time-varying networks. We evaluate our method on the Fashion-MNIST dataset with different Byzantine attacks and system sizes. Numerical results demonstrate the robustness of our proposed method against Byzantine attacks and superior performance than existing methods.

***Index Terms*—** Decentralized Learning, Byzantine Attack, Machine Learning Security

1. INTRODUCTION

The rapid development of deep learning technologies greatly facilitates our daily life, such as deep neural networks for face identification [1], speech recognition [2], and machine translation [3]. This large progress of deep learning is largely attributed to massive data collected from intelligent devices and increasing computation capabilities. However, the increasing amount of edge devices, generated data and growing concern on data privacy make the centralized learning approaches, which require a central server to gather data or coordinate all computing devices, suffer from privacy and scalability issues [4]. Nowadays, there is a growing trend to learn deep learning models in a decentralized fashion, in which edge devices/-clients are capable of not only collecting data but also training model locally and communicating with others [5, 6]. This learning scheme enables distributed devices to collaboratively learn better models and exhibits promising applications in many scenarios, e.g., Internet of Vehicles [7] and recommendation in social networks [8].

In decentralized learning systems, each worker node updates its own model using local data samples, and periodically exchanges its model with neighbors to achieve consensus over the network [9, 10]. This decentralized learning paradigm is advantageous in privacy protection and efficiency improvement since the raw data are kept local and there is no central parameter server being the system performance bottleneck. Decentralized methods are shown to be more efficient than their centralized implementation for large-scale model training when the communication links are subject to high latency and limited bandwidth [5]. However, most decentralized learning algorithms are brittle in the sense that they have no fault-tolerant mechanisms and thus vulnerable to system failures and cyber attacks [11]. Among different types of failures, the Byzantine failure is considered the most general one since the Byzantine nodes are allowed to have arbitrary behaviors [11, 12], such as sending malicious model values to contaminate neighbors and even lead all the regular nodes in the network to end up with incorrect results. In recent years, much research attention has focused on mitigating Byzantine attacks in server-based distributed learning systems (centralized settings), either by utilizing statistic-based robust gradient aggregation rules [13, 14] or leveraging data-redundancy/error-correcting code techniques [15]. In particular, if auxiliary validation data is available in the server, performance-based approaches are promising to detect malicious attacks [16, 17]. However, existing statistic-based methods, e.g., trimmed-mean and Krum, require the Byzantine nodes less than regular nodes and are vulnerable to some well-crafted attacks [18]. Redundancy-based approaches are not directly applicable in decentralized setting due to isolated data storage and constrained communication resource. Compared to centralized setting, research of robust and efficient algorithms against Byzantine attacks in decentralized learning is still few. ByRDiE [12] and BRIDGE [19] are two solutions using coordinate-wise trimming strategy on coordinate descent and gradient descent methods, respectively. A total-variation norm-penalized approximation method (TV-Norm) is proposed in [20] for robust decentralized optimization. MOZI [21] rejects malicious models by checking both distance and performance since local data can be used to evaluate any received models directly. Nevertheless, performance-based approaches evaluate all received models individually, thus the computation costs increase linearly with network scale and will lead to significant efficiency reduction.

In this paper, we develop a novel and efficient Byzantine-resilient algorithm for large-scale decentralized learning systems. We calculate *cross-update* vectors (defined in Section 3) and design a similarity-based reweighting scheme to conduct weighted aggregation, getting robust model update for each regular node. To the best of our knowledge, this is the first work to employ similarity-based aggregation for mitigating Byzantine attacks in decentralized learning systems. In contrast to existing approaches, our developed algorithm guarantees training convergence while introducing negligible computational overhead, and thus has high robustness and scalability. The proposed method is evaluated on the Fashion-MNIST dataset by conducting extensive experiments with various Byzantine attack methods. Numerical results demonstrate the effectiveness of our method in safeguarding model training.

2. PROBLEM SETUP

We focus on decentralized collaborative learning over a communication network, in which each worker node owns some training data and connects to some neighbors.

*Corresponding author: shaolun.huang@sz.tsinghua.edu.cn

2.1. System Model for Decentralized Learning

The decentralized learning system can be modeled as an undirected graph $G = ([n], E)$, where $[n] = \{1, 2, ..., n\}$ denotes a set of n worker nodes and E denotes the communication links [9]. Each worker node i maintains a local model parameter $\mathbf{x}_i$ and only has access to local data $\mathcal{D}_i$ to perform model training, but is allowed to exchange model parameters with its neighbors, which we denote by $\mathcal{N}_i = \{j \in [n] : (i, j) \in E\}$. The goal is to collaboratively learn a model that minimizes the average of all local objective functions in a decentralized manner over n nodes. Mathematically, the underlying optimization problem can be formalized as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \underbrace{\mathbb{E}_{\xi_i\sim\mathcal{D}_i}\left[f(\mathbf{x};\xi_i)\right]}_{:=f_i(\mathbf{x})} \tag{1}$$

where $f_i(\mathbf{x})$ denotes the local objective function of node i. Since workers actually maintain different local models and can only communicate with neighbors to achieve consensus, we can equivalently rewrite the optimization problem (1) as the following classical decentralized optimization problem

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}_i), \text{ subject to } \mathbf{x}_1 = \cdots = \mathbf{x}_n \tag{2}$$

In such a problem, the workers need to iteratively perform local updates and global averaging. Since each worker only communicates with neighbors, it's very challenging to ensure that trained local models stay close to each other, i.e., achieving consensus [12], especially when Byzantine nodes exist. Gossip-type algorithms that use mixing matrix $W \in [0, 1]^{n\times n}$ to conduct model averaging are among the most popular approaches [22, 23] and can converge linearly in non-Byzantine settings. However, such gossip algorithms cannot tolerate Byzantine failures. Therefore, we define the following error term to quantify the difference between local models.

$$err = \frac{1}{n}\sum_{i=1}^{n} \frac{\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2}{\|\bar{\mathbf{x}}\|^2}, \quad \text{where } \bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \tag{3}$$

In this paper, we let $\|\cdot\|$ denote the l_2-norm. We consider the consensus is well achieved if averaged model accuracy exceeds the predefined target and the consensus error is low, e.g., $err < 0.1\%$. Our goal is to develop a Byzantine-resilient algorithm that achieves high model accuracy and low consensus error simultaneously.

2.2. Byzantine Threat Model

In this work, we assume there exists a Byzantine attacker that attempts to tamper the training process by compromising b worker nodes (Byzantine nodes) to perform malicious attacks. We use $\mathcal{B}$ and $\mathcal{R}$ to denote the sets of Byzantine nodes and other regular nodes, respectively. In such case, the underlying objective function in (1) changes to $f(\mathbf{x}) = \frac{1}{n-b}\sum_{i\in\mathcal{R}} f_i(\mathbf{x})$. We assume the Byzantine attacker has access to all local model parameters of regular nodes and Byzantine nodes are allowed to collude and can send arbitrarily faulty messages to neighbors, which are common assumptions used in literature [13, 14, 24]. And to be more realistic, we assume regular nodes cannot know either the exact number of connected Byzantine nodes or the total amount of Byzantine nodes in the network.

2.3. Proposed Attack Method

Previous work in [12, 19, 20] only evaluated some simple attacks, such as random attack and reverse attack with significantly larger values than normal parameters, which can be easily mitigated. Since the Byzantine attacker can collect all the exchanged models from regular nodes, it can perform well-crafted attacks that are harder to defend. Inspired by the *little is enough* (LIE) attack introduced by [18] in server-based distributed learning, we generalize it into the decentralized learning setting to assess the robustness of existing learning algorithms. At any iteration t, suppose the model parameters transmitted from regular nodes are $\mathbf{x}_i^t$ ($i \in \mathcal{R}$), denote their coordinate-wise mean and standard deviation vectors by $\boldsymbol{\mu}_t$ and $\boldsymbol{\sigma}_t$, then Byzantine node j ($j \in \mathcal{B}$) sends such vector to its neighbors:

$$\mathbf{x}_j^t = \boldsymbol{\mu}_t + z\boldsymbol{\sigma}_t \tag{4}$$

where z is a positive coefficient as defined in [18]. We call this attack method decentralize version of little is enough attack (D-LIE).

3. THE PROPOSED METHOD

Existing algorithms in decentralized learning have two main types: (i) "Gossip Before Local Update" where the one-step gradient computation and gossip averaging are performed in parallel [5], this type of algorithms can decouple communication and computation; (ii) "Gossip After Local Update" where gossip averaging is performed after the local model is updated by SGD steps [25], this type of algorithms allow more local iterations with less frequently communication [10]. Considering the communication resource is usually constrained in edge devices, we use the "Gossip After Local Update" strategy with multiple local steps. To mitigate Byzantine attacks efficiently and reliably in regular node i, we do not evaluate the received model $\mathbf{x}_j$ from node j directly but calculate the cross-update vector Δ_{ji} as defined in (5), and use Δ_{ii} as the reference vector.

$$\Delta_{ji}^t := \mathbf{x}_j^{t+1} - \mathbf{x}_i^t \tag{5}$$

We propose a similarity-based reweighting scheme to calculate weights for update vectors, then employ the norm clipping and weighted average to obtain a robust model update for each node. The detailed workflow of our method is summarized in Algorithm 1. Specifically, we design two metrics to jointly measure the similarity between each cross-update vector and the reference vector and use the multiplication of these two metrics as the weight during aggregation. Recent work [26] has shown that the empirical sign distribution of elements in honest gradient (ratios of positive, zero and negative elements) can be utilized for anomaly detection in centralized settings. Inspired by that, we firstly apply the KL-divergence to measure the distance between empirical sign distributions of each update vector Δ_{ji} (denote by p_j) and the reference vector Δ_{ii} (denote by q_i) as follows:

$$D_{KL}(p_j||q_i) := \sum_{s=1,0,-1} p_j(s)\log\left(\frac{p_j(s)}{q_i(s)}\right) \tag{6}$$

Since the KL-divergence is negatively correlated to distribution similarity and can be arbitrarily large, we use the softmax with negative temperature to get a similarity metric with values ranged in $(0, 1)$:

$$\omega_{ji,1} := \frac{\exp\left(-c\cdot D_{KL}(p_j||q_i)\right)}{\sum_j \exp\left(-c\cdot D_{KL}(p_j||q_i)\right)} \tag{7}$$

where the scale factor $c > 0$. For the second similarity metric, we apply the cosine-similarity with ReLU activation as proposed in [27].

$$\omega_{ji,2} := \max\{0, \frac{\Delta_{ji}\cdot\Delta_{ii}}{\|\Delta_{ii}\|\|\Delta_{ii}\|}\} \tag{8}$$

Then we get the final similarity as $Sim(\Delta_{ji}, \Delta_{ii}) = \omega_{ji,1}\cdot\omega_{ji,2}$,

Algorithm 1

1: **Input:** learning rate η, total iteration T, local step τ.
2: **Initial:** $\mathbf{x}_0 \in \mathbb{R}^d$
3: **for** $t = 0, 1, ..., T-1$ **do**
4: **Regular node** $i \in \mathcal{R}$ **in parallel:**
5: Local update: $\mathbf{x}_i^{t+1} \leftarrow \mathbf{x}_i^t - \eta \nabla f(\mathbf{x}_i^t; \xi_i)$
6: **if** $(t+1)\ mod\ \tau = 0$ **then**
7: Broadcast $\mathbf{x}_i^{t+1}$ and receive $\mathbf{x}_j^{t+1}$ from $j \in \mathcal{N}_i$
8: Get update vector: $\Delta_{ji}^t \leftarrow \mathbf{x}_j^{t+1} - \mathbf{x}_i^t$ for $j \in \mathcal{N}_i \cup \{i\}$
9: Weight calculation: $\omega_{ji}^t \leftarrow Sim(\Delta_{ji}^t, \Delta_{ii}^t)$
10: Norm clipping: $\bar{\Delta}_{ji}^t \leftarrow \Delta_{ji}^t \cdot \min(1, \frac{\|\Delta_{ii}^t\|}{\|\Delta_{ji}^t\|})$
11: Update local model: $\mathbf{x}_i^{t+1} \leftarrow \mathbf{x}_i^t + \frac{\sum \omega_{ji}^t \bar{\Delta}_{ji}^t}{\sum \omega_{ji}^t}$
12: **end if**
13: **end for**

which means the cross-update vectors that have either high KL-divergence in sign distribution or low cosine-similarity will contribute less to local model update. Finally, we use the norm of reference vector $\|\Delta_{ii}^t\|$ as the upper bound to clip update vectors and conduct a weighted average (see lines 10-11 in Alg. 1). The theoretical analysis of convergence guarantee is left for future work.

4. EVALUATION

In this section, we conduct various experiments to verify the robustness of our proposed approach to Byzantine attacks. We compare our method with four benchmark methods, including D-PSGD [5] without fault-tolerance, BRIDGE [12], the total-variation norm-penalized approximation (TV-Norm) [20], and training local model using local data without collaboration (Stand-alone). All experiments are simulated in NVIDIA GeForce RTX 3090 GPUs.

4.1. Experimental Setup

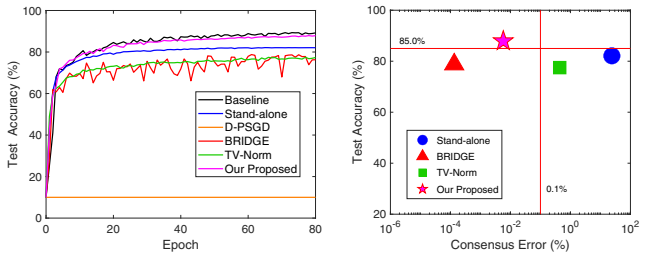
We use the Erdos-Renyi graph model with $n = 50$ workers and edge creation probability of $p = 0.6$ to generate our decentralized networks. We randomly select $b = 10$ workers as Byzantine nodes and guarantee that each honest worker is connected with at least $2b + 1$ neighbors. We use the Fashion-MNIST dataset [28] and a convolution neural network (CNN) to conduct learning tasks. Fashion-MNIST is a 10-class clothing image classification dataset, which consists of 60,000 training samples and 10,000 testing samples. We distribute the training samples randomly and evenly to all the workers. Each worker node maintains a local CNN model, which is constructed by two convolution layers with 16 and 32 channels respectively, each followed by a max-pooling layer, and two fully-connected layers with 128 and 10 units before softmax output.

We employ the stochastic gradient decent (SGD) as the local optimizer. By default, the mini-batch size is set to 60, local step is set to 1, and each training algorithm is run for 80 epochs. The learning rate is fixed to 0.1, and the weight decay is set to 0.0005 to prevent over-fitting. We test the algorithm performance at the end of each training epoch. The performance is measured by both average accuracy on testing samples and consensus error on regular nodes.

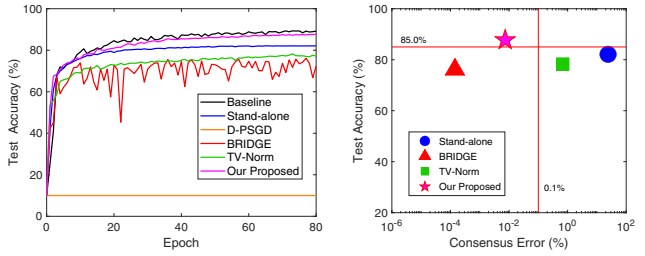
4.2. Numerical Results

We evaluate all methods under three attacks: (i) **Random** attack, the attacks are generated from Guassian vectors with the same norm as the averaged honest parameter vector $\bar{\mathbf{x}} = \frac{1}{n-b} \sum_{i \in \mathcal{R}} \mathbf{x}_i$; (ii) **Reverse** attack, Byzantine nodes send sign-flipped version of averaged parameter $-\bar{\mathbf{x}}$; (iii) **D-LIE** attack, we set $z = 2.0$ in Eq. (4). We choose the scale factor $c = 5$ for our method and regularization factor $\lambda = 0.001$ for TV-Norm in all experiments.

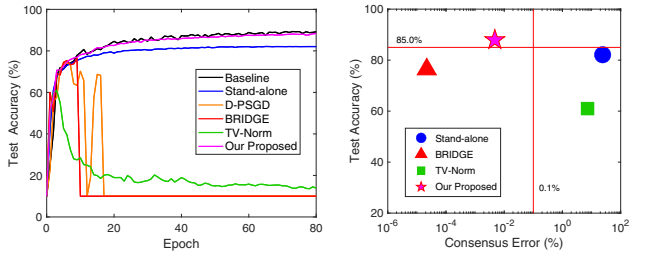
Static Network Setting. In the first set of experiments, we assume the network is static during the training process, and we use the random seed to ensure the same network topology for fair comparisons. We use the result of D-PSGD algorithm without Byzantine attack as baselines and show the experiment results under three different attacks in Fig. 1. We can see that D-PSGD suffers from all kinds of attacks and fails to converge. Both BRIDGE and TV-Norm can tolerate simple random and reverse attacks, but result in non-negligible accuracy drops. Moreover, those two methods are ineffective under our proposed D-LIE attack. By contrast, our proposed method is robust to all three attacks and can achieve both high accuracy (>85%) and low consensus error (<0.1%) as shown in Fig. 1.



(a) Random attack



(b) Reverse attack



(c) D-LIE attack

Fig. 1: Test accuracy over epoch (left) and best achieved accuracy with corresponding consensus error (right) of regular nodes.

Time-varying Network Setting. This set of experiments is conducted over a time-varying network by randomly changing graph topology at every training epoch. We evaluate BRIDGE, TV-Norm and our method under reverse and D-LIE attacks, presenting the accuracy curves in Fig. 2. As expected, our proposed method exhibits strong robustness to Byzantine attacks even on time-varying networks, while other methods result in either significant performance degradation or even divergence under attacks.

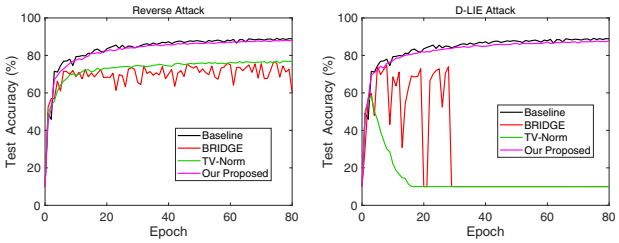


Fig. 2: Test accuracy of regular nodes under reverse and D-LIE attacks in time-varying network.

Ablation Study. As we designed a compounded similarity metric based on two components, here we evaluate the importance of each component separately. The results in Table 1 demonstrate that each metric is effective for some specific types of attack, but only the combination of them could mitigate various kinds of Byzantine attacks.

Table 1: Test accuracy (%) under different similarity metrics

Metric-1 Eq.(7)	**Metric-2** Eq.(8)	**Attacks**		
		Random	Reverse	D-LIE
✓	✗	71.62	60.81	88.56
✗	✓	87.88	87.68	83.32
✓	✓	87.86	87.57	88.37

Communication Reduction by Local SGD. We notice that in the regular decentralized learning algorithms, the workers conduct consensus steps at every iteration, which is communication expensive and inefficient. Therefore, we apply the local SGD technique to reduce communication costs. We alter the local iteration steps to $\tau = 2, 5, 10$ and keep the total training epochs unchanged. Fig. 3 shows the test accuracy over communication rounds under reverse attack, and Table 2 lists the communication rounds to achieve 85% accuracy. It can be seen that local SGD is effective in communication reduction, but also leads to a trade-off between efficiency and accuracy. When $\tau = 2$, more than half the rounds are saved to achieve target accuracy and the consensus is also well achieved, but when $\tau = 10$ the model cannot achieve the target accuracy within given training epochs.

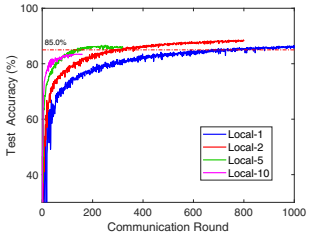


Fig. 3: Accuracy over round.

Local Steps	**Commu. Rounds**	**Consensus Error**
1	670	0.00506 %
2	273	0.00936 %
5	142	0.06890 %
10	N/A	N/A

Table 2: Communication rounds to achieve target accuracy (85%).

Computational Overhead. We also assess the computation cost for different aggregation rules. Here we apply the performance-based method MOZI [21], which removes those received models that result in larger evaluation loss than the local model, to conduct training under D-LIE attack for more convincing comparison. We measure and compare the per-round aggregation time for different methods. The results are presented in Fig. 4, from which we can find that though

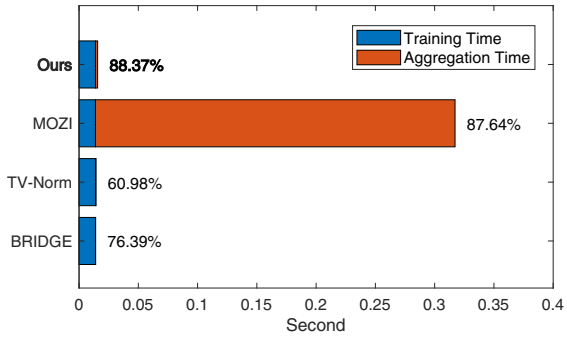


Fig. 4: Aggregation time and best achieved model accuracy.

MOZI results in promising model accuracy, it incurs much higher computational overheads and thus reduces the overall training efficiency. In contrast, our proposed method introduces much less computation cost and achieves both high accuracy and efficiency.

System Scalability. To further verify the scalability of our algorithm, we report the results of stand-alone training and our algorithm under another three different system sizes, where we set $n = 100, 200, 300$ with $b = 0.2n$ respectively, and adjust the batch-size to keep the total training iterations unchanged. A larger system size means fewer training samples in each worker. It can be observed that our method consistently achieves high test accuracy regardless of large system sizes. Although the stand-alone training strategy can save communication resources and directly avoid Byzantine attacks, the resulted model accuracy tends to decline as the system size increases, for each single worker node cannot utilize knowledge of other workers. However, our method enables regular nodes to defend against Byzantine attacks while benefiting from collaborative learning under different system sizes, demonstrating the resilience and scalability in large-scale decentralized learning systems.

Table 3: Test accuracy (%) under attacks and average aggregation time (s) under different system sizes.

Size	**Stand-alone**	**Our method**			
		Random	Reverse	D-LIE	Time
50	82.08	87.91	87.72	88.37	0.00183
100	80.57	87.85	87.69	88.34	0.00241
200	76.65	86.59	86.62	87.94	0.00322
300	74.83	86.14	86.56	87.32	0.00395

5. CONCLUSION

In this work, we proposed a novel robust learning algorithm to mitigate Byzantine attacks efficiently in decentralized learning systems. We investigated the performance in both static and dynamic network settings and evaluated the computation cost and system scalability in particular. Numerical results demonstrated the resilience of our proposed method against attacks under large-scale systems. Future work includes developing robust decentralized learning algorithms over directed and time-varying networks with non-iid data.

ACKNOWLEDGEMENT The research of Shao-Lun Huang is supported in part by the Shenzhen Science and Technology Program under Grant KQTD20170810150821146, National Key R&D Program of China under Grant 2021YFA0715202 and High-end Foreign Expert Talent Introduction Plan under Grant G2021032013L.

6. REFERENCES

[1] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, 2015, pp. 815–823.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] Jonas Gehring, Michael Auli, David Grangier, and Yann N. Dauphin, "A convolutional encoder model for neural machine translation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL*, 2017.

[4] Aurélien Bellet, Rachid Guerraoui, Mahsa Taziki, and Marc Tommasi, "Personalized and private peer-to-peer machine learning," in *International Conference on Artificial Intelligence and Statistics, AISTATS*, 2018.

[5] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu, "Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[6] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu, "D^2: Decentralized training over decentralized data," in *Proceedings of the 35th International Conference on Machine Learning, ICML*, 2018, pp. 4855–4863.

[7] Amuleen Gulati, Gagangeet Singh Aujla, Rajat Chaudhary, Neeraj Kumar, and Mohammad S. Obaidat, "Deep learning-based content centric data dissemination scheme for internet of vehicles," in *2018 IEEE International Conference on Communications, ICC 2018*. 2018, pp. 1–6, IEEE.

[8] Cem Tekin, Simpson Zhang, and Mihaela van der Schaar, "Distributed online learning in social recommender systems," *IEEE J. Sel. Top. Signal Process.*, vol. 8, no. 4, pp. 638–652, 2014.

[9] Angelia Nedic, "Distributed gradient methods for convex machine learning problems in networks: Distributed optimization," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 92–101, 2020.

[10] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian U. Stich, "A unified theory of decentralized SGD with changing topology and local updates," in *Proceedings of the International Conference on Machine Learning, ICML*, 2020, pp. 5381–5393.

[11] Zhixiong Yang and Waheed U. Bajwa, "Byrdie: Byzantine-resilient distributed coordinate descent for decentralized learning," *IEEE Trans. Signal Inf. Process. over Networks*, vol. 5, no. 4, pp. 611–627, 2019.

[12] Zhixiong Yang, Arpita Gang, and Waheed U. Bajwa, "Adversary-resilient distributed and decentralized statistical inference and machine learning: An overview of recent advances under the byzantine threat model," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 146–159, 2020.

[13] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[14] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[15] Deepesh Data, Linqi Song, and Suhas N. Diggavi, "Data encoding for byzantine-resilient distributed optimization," *IEEE Trans. Inf. Theory*, vol. 67, no. 2, pp. 1117–1140, 2021.

[16] Cong Xie, Sanmi Koyejo, and Indranil Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proceedings of the 36th International Conference on Machine Learning, (ICML)*, 2019.

[17] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *29th USENIX Security Symposium*, 2020.

[18] Gilad Baruch, Moran Baruch, and Yoav Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Advances in Neural Information Processing Systems*, 2019.

[19] Zhixiong Yang and Waheed U. Bajwa, "BRIDGE: byzantine-resilient decentralized gradient descent," *CoRR*, vol. abs/1908.08098, 2019.

[20] Jie Peng and Qing Ling, "Byzantine-robust decentralized stochastic optimization," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020.

[21] Shangwei Guo, Tianwei Zhang, Xiaofei Xie, Lei Ma, Tao Xiang, and Yang Liu, "Towards byzantine-resilient learning in decentralized systems," *arXiv:2002.08569*, 2020.

[22] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," in *Proceedings of the 36th International Conference on Machine Learning, ICML*, 2019.

[23] Jianyu Wang and Gauri Joshi, "Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms," *CoRR*, vol. abs/1808.07576, 2018.

[24] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[25] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 3049–3058.

[26] Jian Xu, Shao-Lun Huang, Linqi Song, and Tian Lan, "Signguard: Byzantine-robust federated learning through collaborative malicious gradient filtering," *arXiv:2109.05872*, 2021.

[27] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," *ISOC Network and Distributed Systems Security (NDSS) Symposium*, 2021.

[28] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.