FedCut: A Spectral Analysis Framework for Reliable Detection of Byzantine Colluders

Hanlin Gu, Lixin Fan, Member, IEEE, XingXing Tang, and Qiang Yang, Fellow, IEEE

Abstract—This paper proposes a general spectral analysis framework that thwarts a security risk in federated Learning caused by groups of malicious Byzantine attackers or colluders, who conspire to upload vicious model updates to severely debase global model performances. The proposed framework delineates the strong consistency and temporal coherence between Byzantine colluders’ model updates from a spectral analysis lens, and formulates the detection of Byzantine misbehaviours as a community detection problem in weighted graphs. The modified normalized graph cut is then utilized to discern attackers from benign participants. Moreover, the Spectral heuristics is adopted to make the detection robust against various attacks. The proposed Byzantine colluder resilient method, i.e., FedCut, is guaranteed to converge with bounded errors. Extensive experimental results under a variety of settings justify the superiority of FedCut, which demonstrates extremely robust model accuracy (MA) under various attacks. It was shown that FedCut’s averaged MA is 2.1% to 16.5% better than that of the state of the art Byzantine-resilient methods. In terms of the worst-case model accuracy (MA), FedCut is 17.6% to 69.5% better than these methods.

Index Terms—Byzantine colluders, Byzantine resilient, federated learning, graph, normalized cut, privacy preserving computing, spectral analysis, spectral heuristics.

I. INTRODUCTION

Federated learning (FL) [37], [63] is a suite of privacy-preserving machine learning techniques that allow multiple parties to collaboratively train a global model, yet, without gathering or exchanging privacy for the sake of compliance with private data protection regulation rules such as GDPR. It is the upgraded global model accuracy that motivates multiple parties to join FL, however, the risk of potential malicious attacks aiming to degrade FL model accuracy cannot be discounted. It was shown that even a single attacker (aka a Byzantine worker) may prevent the convergence of a naive FL aggregation rule by submitting vicious model updates to outweigh benign workers (see Lemma 1 of [6]). A great deal of research effort was then devoted to developing numerous Byzantine-resilient methods which can effectively detect and attenuate such misbehaviours [6], [12], [20], [41], [58], [62], [65]. Nevertheless, recent research pointed out that a group of attackers or colluders may conspire to cause more damages than these Byzantine-resilient methods can deal with (see [4], [17], [60]). It is the misbehavior of such Byzantine colluders that motivate our research to analyze their influences on global model accuracy from a spectral analysis framework, and based on our findings, an effective defense algorithm to do away with Byzantine colluders is proposed.

There are two main challenges brought by Byzantine colluders. First, colluders may conspire to misbehave consistently and introduce statistical bias to break down Robust Statistics (RS) based resilient methods (e.g., [4], [17], [60]). Consequently, the global model accuracy may deteriorate significantly. Second, Byzantine colluders may conspire to violate the assumption that all malicious model updates form one group while benign ones form the other, which is invariably assumed by most clustering-based Byzantine-resilient methods [19], [47], [51]. By submitting multiple groups of such detrimental yet disguised model updates, colluders therefore can evade clustering-based methods and degrade the global model accuracy significantly.

In order to address the challenges brought by colluders, we propose in this article a spectral analysis framework, called FedCut, which admits effective detection of Byzantine colluders in a variety of settings and provide the spectral analysis of different types of Byzantine behaviours especially for Byzantine colluders (see Section IV). The essential ingredients of the proposed framework are as follows. First, we build the Spatial-Temporal graph with all clients’ model updates over multiple learning epochs as nodes, and similarities between respective pairs of model updates as edge weights (see Section V-A). Second, an extension of the normalized cut (Ncut) [39], [52] provides the optimal c-partition Ncut (see Section V-B), which allows to detect colluders with consistent behaviour efficiently. Third, spectral heuristics [67] are used to determine the type of Byzantine attackers, the unknown number of colluder groups and the appropriate scaling factor σ of Gaussian kernels used for measuring similarities between model updates (see Section V-C). By leveraging the aforementioned techniques together within the unified spectral analysis framework, we thus
propose in Section V the FedCut method which demonstrates superior robustness in the presence of different types of colluder attacks. Its convergence is theoretically proved in Section VI and it compares favorably to the state of arts Byzantine-resilient methods in thorough empirical evaluations of model accuracy in Section VII. Our contributions are three folds.

- First, we provide the spectral analysis of Byzantine attacks, especially, those launched by colluders. Specifically, we formulate existing Byzantine attacks as four types and gain a deeper understanding of colluders’ attacks in the lens of spectral. Moreover, we delineate root causes of failure cases of existing Byzantine-resilient methods in the face of colluder attacks.

- Second, we propose the spectral analysis framework, called FedCut, which distinguishes benign clients from multiple groups of Byzantine colluders. Specifically, we adopt the normalized cut, temporal consistency and the spectral heuristics to address challenges brought by colluders. Moreover, we provide the theoretical analysis of convergence of the proposed algorithm FedCut.

- Finally, we propose to thoroughly investigate both averaged and worst-case model accuracy of different Byzantine-resilient methods with extensive experiments under a variety of settings, including different types of models, datasets, extents of Non-IID, the fraction of attacker combinations of colluders groups. It was then demonstrated that the proposed FedCut consistently outperforms existing methods under all different settings (see Fig. 1 and Table I).

## II. RELATED WORK

Related work are abundant in the literature and we briefly review them below.

*Byzantine attacks:* can be broadly categorized as Non-Collusion and Collusion attacks. The former attacks were proposed to degrade global model accuracy by uploading Gaussian noise or flipping gradients [6], [33]. The latter type launched consistent attacks to induce misclassifications of Byzantine attackers [4], [17], [60].

*Robust statistics based aggregation:* approaches treated malicious model updates as outliers, which are far away from benign clients, and filter out outliers via robust statistics accordingly. For example, the coordinate-wise Median and some variants of

![Fig. 1. Averaged (left) and worst-case (right) model accuracy (MA) under all attacks for different Byzantine-resilient methods (Krum [6], GeoMedian [12], Median, Trimmed mean [65], Bulyan [20], FLtrust [9], DnC [48], Kmeans [51] and our proposed FedCut) with IID setting and 30 Byzantine clients for classification of Fashion MNIST (FedCut – the proposed method, see more details in Section VII-A).](image)

| OVERVIEW OF THE PERFORMANCE FOR VARIOUS BYZANTINE-RESILIENT METHODS (ROBUST STATISTICS BASED: KRUML [6], MEDIAN, TRIMMED MEAN [65] AND DnC [48]; CLUSTERING BASED: KMEANS [51]; SERVER BASED: FLTRUST [9]; SPECTRAL BASED: FEDCUT, SEE MORE DETAILS IN SECTION II) UNDER DIFFERENT BYZANTINE ATTACKS (NON-COLLUSION ATTACK: [6], LABEL FLIPPING [14] AND SIGN FLIPPING [14], COLLUSION-DIFF ATTACK: SAME VALUE ATTACK [33], FANG-V1 (DESIGN FOR TRIMMED MEAN) [17] AND OUR DESIGNED MULTI-COLLUSION ATTACK, COLLUSION-MIMIC ATTACK: MIMIC [27] AND LIE [4], SEE MORE DETAILS IN SECTION VII-A) |
|-----------------|-----------------|-----------------|-----------------|-----------------|
|                  | Robust statistics based | Clustering based | Server based | Spectral based |
|                  | Krum | Median | Trim mean | DnC | Kmeans | FLtrust | FedCut(Ours) |
| Non-Collision    | Gaussian[6] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
|                  | label flipping[14] | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ |
|                  | Sign flipping[14] | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |
| Collision-diff   | same value[33] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
|                  | Fang-v1[17] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
|                  | Multi-collision | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Collision-mimic  | Mimic[27] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
|                  | LIE [4] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

✓, ✓ and ✗ denotes the drop of model accuracy less than 3%, from 3% to 10% and above 10% on Fashion MNIST respectively (see more comparisons in section VII).
Median, such as geometric Median, were proposed to remove outliers [41], [58], [65]. Moreover, Blanchard et al. assumed few outliers were far away from benign updates. Therefore, they used the Krum function to select model updates that are very close to at least half the other updates [6] as benign updates. However, the aforementioned methods are vulnerable to attacks with collusion which may conspire to induce biased estimations [4], [17], [60]. A different line of approaches [14], [15], [48] used the concentration filter to remove the outliers which are far away from concentration (such as the mean) [14], [15], [48]. For instance, Shejwalkar et al. applied SVD to the covariance matrix of model updates and filtered out outliers that largely deviate from the empirical mean of model updates towards the principle direction [48]. However, the effectiveness of the concentration filter may deteriorate in the presence of collusion attacks (see Section IV-C), and moreover, the time complexity of this approach is high $O(d^3)$ when the dimension of updates $d$ is large.

**Clustering based robust aggregation:** grouped all clients into two clusters according to pairwise similarities between clients’ updates and regarded the cluster with a smaller number as the Byzantine cluster. For instance, some methods applied Kmeans into clustering benign and Byzantine clients [8], [51] and Sattler et al. separated benign and Byzantine clients by minimizing cosine similarity of updates among two groups [46], [47]. Moreover, Ghosh et al. made use of iterative Kmeans clustering to remove the Byzantine attackers [19]. However, the adopted naive clustering such as Kmeans, Kmedian [25], [36] have two shortcomings: 1) being inefficient and easily trapped into local minima; 2) breaking down when the number of colluder groups is unknown to the the clustering method (see details in Section IV-C).

**Server based robust aggregation:** assumed the server had an extra training dataset which was used to evaluate uploaded model updates. Either abnormal updates with low scores were filtered out [9], [42], [44], [59] or minority Byzantine updates were filtered out through majority votes [11], [21], [43]. However, this approach is not applicable to the case when the server-side data is not available, or it might break down if the distribution of server’s data deviates far from that of training data of clients.

**Historical Information based byzantine robust methods:** made use of historical information (such as distributed Moments [16]) to help correct the statistical bias brought by colluders during the training, and thus lead to the convergence of optimization of federated learning [2], [3], [10], [18], [26], [28].

**Other byzantine robust methods:** used signs of the gradients [5], [53], optimization strategy [33] or sampling methods [27] to achieve robust aggregation. Recently, some work studied Byzantine-robust algorithms in the decentralized setting without server [20], [23], [40] and asynchronous setting with heterogeneous communication delays [13], [61], [64].

**Community detection in graph:** In the proposed framework, the detection of Byzantine colluders is treated as the detection of multiple subgraphs or communities in a large weighted graph (see Section V-A). Existing approaches [38] could be applied to detect the byzantine colluders. One important technique is to detect specific features of graph such as clustering coefficients [56]. Another important technique is to leverage the spectral property of graph based on the adjacency matrix or normalized Laplacian matrix and so on [7], [45], [50]. Moreover, the number of communities can be determined based on the eigengap of these matrix [67] (see Section V-C).

### III. Preliminary

#### A. Federated Learning

We consider a horizontal federated learning [37], [63] setting consisting of one server and $K$ clients. We assume $K$ clients\(^2\) have their local dataset $D_i = \{(x_{i,j}, y_{i,j})\}^{n_i}_{j=1}$, $i = 1 \cdots K$, where $x_{i,j}$ is the input data, $y_{i,j}$ is the label and $n_i$ is the total number of data points for $i_{th}$ client. The training in federated learning is divided into three steps which iteratively run until the learning converges:

- The $i_{th}$ client takes empirical risk minimization as:

$$
\min_{w_i} F_i(w_i, D_i) = \min_{w_i} \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(w_i, x_{i,j}, y_{i,j}),
$$

where $w_i \in \mathbb{R}^d$ is the $i_{th}$ client’s local model weight and $\ell(\cdot)$ is a loss function that measures the accuracy of the prediction made by the model on each data point.

- Each client sends respective local model updates $\nabla F_i$ to the server and the server updates the global model $w$ as $w = w - \eta \sum_{i=1}^{K} \nabla F_i$, where $\eta$ is learning rate.

- The server distributes the updated global model $w$ to all clients.

#### B. Byzantine Attack in Federated Learning

We assume a malicious threat mode where an unknown number but less than half of participants out of $K$ clients are Byzantine, i.e., they may upload arbitrarily corrupt updates $g_b$ to degrade the global model accuracy (MA). Under this assumption, behaviours of Byzantine clients and the rest of benign clients can be summarized as follows:

$$
g_i = \begin{cases} 
\nabla F_i & \text{Benign clients} \\
\mathbb{g}_b & \text{Byzantine clients}
\end{cases}
$$

Note that under the assumed threat mode, each adversarial node has access to updates of all clients during the training procedure. They are aware of the adoption of Federated Learning Byzantine-resilient methods [4], [6], [60] and conspire to upload specially designed model updates that may overwhelm existing defending methods. Byzantine clients who behave consistently as such are referred to as colluders throughout this article. Moreover, we assume the server to be honest and try to defend the Byzantine attacks.

\(^2\)In this article we use terms "client", "node", "participant" and "party" interchangeably.
IV. FAILURE CASES CAUSED BY BYZANTINE COLLUDERS

The challenge brought by colluders is detrimental to Byzantine-resilient algorithms in different ways. We first analyze below behaviors of representative Byzantine attackers from a graph theoretic perspective. Then we demonstrate a toy example to showcase that existing Byzantine-resilient methods are vulnerable to collusion attacks.

A. Weighted Undirected Graph in Federated Learning

We regard model updates contributed by $K$ clients as an undirected graph $G = (V, E)$, where $V = v_1, \ldots, v_K$ represent $K$ model updates, $E$ is a set of weighted edge representing similarities between uploaded model updates corresponding to clients in $V$. We assume that the graph $G = (V, E)$ is weighted, and each edge between two nodes $v_i$ and $v_j$ carries a non-negative weight, e.g., $A_{ij} = \exp(-||g_i - g_j||^2/2\sigma^2) \geq 0$, where $g_i$ is uploaded gradient for $v_i$th client and $\sigma$ is the Gaussian scaling factor. Let $G_R = (V_R, E_R)$ and $G_B = (V_B, E_B)$ respectively denote two subgraphs of $G$ representing benign and Byzantine clients.

Moreover, Byzantine problem [30] could be regarded as finding an optimal graph-cut for $G$ to distinguish the Byzantine and benign model updates. Since model updates from colluders form specific patterns (see Fig. 3), the aforementioned graph-cut can be generalized to the so called community-detection problem [38] in which multiple subsets of closely connected nodes are to be separated from each other.

B. Spectral Graph Analysis for Byzantine Attackers

We illustrate the spectral analysis of representative Byzantine attackers, especially, those launched by colluders. For example, Fig. 2 shows adjacency matrices with elements representing pairwise similarities between 70 benign clients under IID setting and 30 attackers (the darker the element the higher the pairwise similarity is). It is clear in each subfigure that benign clients form a single coherent cluster residing in the upper-left block of the adjacent matrix, while attackers reside in the bottom-right parts of the adjacent matrix. We observe the following characteristics pertaining to benign as well as Byzantine model updates. Note that Assumption 1 has been widely used to bound differences between model updates of benign clients, e.g., in [35], [66].

Assumption 1: Assume the difference of local gradients $\nabla F_i$ and the mean of benign model update $\nabla F = \frac{1}{|V_R|} \sum_{i \in V_R} \nabla F_i$ is bounded ($V_R$ is the set of benign clients), i.e., there exists a finite $\kappa$, such that

$$||\nabla F_i - \nabla F||^4 \leq \kappa.$$ Second, Byzantine model updates can be categorized into four types (Fig. 3):

- Non-Collusion: $||g_b - \nabla F|| > \kappa$ and malicious updates ($g_b$) are far away from each other (e.g., Gaussian attack [6], Label flipping [14] and Sign flipping [14] attacks in Fig. 2(a), (b) and (c)).
- Collusion-diff: $||g_b - \nabla F|| > \kappa$ and malicious updates ($g_b$) form one or multiple clusters (small intra-cluster distance) (e.g., our designed collusion attack (see Section VII-A), same value attack [33], Fang-v1 [17] attacks in Fig. 2(d), (e) and (f)).
- Collusion-mimic: $||g_b - \nabla F|| < \kappa$ and $g_b$ of different attackers are almost identical. It represents adversaries with strong connections form one or multiple clusters (small

\[3\] Note that the order of the nodes illustrated in Fig. 2 is irrelevant and we separate benign nodes from Byzantine nodes only for better visual illustration.

\[4\] In the paper represents the $\ell_2$ norm.
Fig. 3. Spectral analysis of four types of Byzantine attackers (Each column represents one type of attack), including Non-Collusion, Collusion-diff, Collusion-mimic and Mixture attacks. We take one example of 100 clients consisting of 70 benign clients and 30 attackers, view the model updates as nodes, and compute the pair-wise similarities as edges. The first row provides an overview of four attacks ((a), (b), (c), and (d) represents Non-Collusion, Collusion-diff, Collusion-mimic and Mixture attacks respectively where green and red points represent benign and Byzantine updates respectively), while the second and third rows show the eigenvalue and eigengap of the normalized adjacency matrix (see Def. 4).

- Mixture: adversaries may combine Non-collusion, Collusion-diff, and Collusion-mimic arbitrarily to obtain a mixture attack.

In order to address the complex types of both benign and Byzantine model updates, we adopt the eigengap technique [50], [67] to reliably detect benign and Byzantine clusters (or communities). We provide the following proposition to elucidate characteristics of different types of Byzantine attacks in the lens of spectral analysis. The proof of Proposition 1 is deferred to Appendix C, available online.

**Lemma 1:** Let \( G \) be an undirected graph with non-negative weights. Then the multiplicity \( c \) of the eigenvalue 1 of \( L \) equals the number of connected components \( B_1, \ldots, B_c \) in the graph.

**Remark 1:** Eigenvalue = 1 signifies one connected component, and the multiplicity \( c \) of the eigenvalue 1 of \( L \) means \( L \) can be separated into \( c \) diagonal blocks [39], [55]. Lemma 1 illustrates that \( c \) is equal to the number of connected components \( B_1, \ldots, B_c \) in the graph.

**Assumption 2:** For malicious updates \( g_b \) provided that \( \|g_b - \nabla F\| > \kappa \), the difference between the mean of benign updates and colluders’ updates has at least \( C\kappa \) distance, where \( C \) is a large constant, i.e.,

\[
\|g_b - \nabla F\| > C\kappa.
\]

**Remark 2:** Assumption 2 demonstrates a large distance between colluders and the mean of benign updates since \( C \) is greatly larger than 1.

**Proposition 1:** Suppose \( K \) clients consist of \( m \) benign clients and \( q \) attackers \( (q < m - 1) \). If Assumption 2 holds for Non-collusion and Collusion-diff attacks, then only the first \( c \) eigenvalues are close to 1 with at most the error \( \sqrt{mqe - \frac{C^2\kappa^2}{2\sigma^2}} \), and we have:

- For Non-collusion attacks provided that \( \|g_b - \nabla F\| > C\kappa \) and malicious updates (\( g_a \)) are far away from each other;
- For Collusion-diff attacks provided that malicious updates form \( B \) groups and \( \|g_b - \nabla F\| > C\kappa \);
- For Collusion-mimic attacks that \( \|g_b - \nabla F\| < \kappa \) and malicious updates are almost identical.

**Remark 3:** Proposition 1 illustrates that the eigenvalue 1 has multiplicity \( c \) with at most the error \( \sqrt{mqe - \frac{C^2\kappa^2}{2\sigma^2}} \), in which the error tends to zero as \( C \) goes to infinity. Therefore, there is a large gap between \( c_{th} \) eigenvalue and \( (c + 1)_{th} \) eigenvalue. Consequently, we can...
For the proposed method FedCut, the Collusion-mimic attack. This property is used in the proposed method to distinguish Collusion-mimic attacks from other types of attacks (see Section V-C3).

We illustrate with a concrete example the different spectra of four types of attacks in Fig. 3, in which each column represents one type of attack. We observe the following properties (all of the examples include 70 benign clients and 30 attackers):

- for Non-Collusion attack, we use the Gaussian attack [33] to upload largely different random updates following $\mathcal{N}(0, 200)$. Fig. 3(e) and (i) show the first 31 eigenvalues are 1, and the 32nd eigenvalue drops significantly (approaching zero). This indicates the presence of 31 connected components, which include one closely-knit group consisting of 70 benign clients submitting slightly different updates, and 30 groups corresponding to Non-collusion attackers who submit dramatically different gradients.

- for Collusion-diff attack, we use the same-value attack [33] to upload updates consisting of all one elements. Fig. 3(f) and (j) displays the first two eigenvalues are one and the third eigenvalue tends to zero, which indicates there are two components. All benign clients form the one group while other 30 collusion attackers form another group since they upload gradients, which is similar each other but largely different from benign gradients.

- for Collusion-mimic attack, we use the mimic attack [27] to upload updates that mimic one benign update. Fig. 3(g) and (k) show the first 70 eigenvalues are 1, and the 71st eigenvalue drops considerably (tending to zero). This implies the existence of 70 connected components, which include one big group consisting of one benign client and all attackers that mimic the benign client with identical gradient update, and 69 small groups each consisting of the remaining benign clients with slightly different gradient updates.

- for Mixture attack, we combine the Gaussian attack (5 attackers), same-value attack (5 attackers), and mimic attack (20 attackers). Fig. 3(h) and (l) reveal that the largest eigengap is 80, in which mimic colluders and one mimicked benign client form a group. In contrast, 69 benign clients and 10 attackers form 79 groups separately. This example showcases that the mimic attack dominates the spectral characteristics of the Mixture attack.

Remark 4: For the proposed method FedCut, the Collusion-mimic type attack is first detected and removed when position of the largest eigengap is larger than $K/2$ (see Algorithm 2 in Section V-C3), followed by the spatial-temporal graph cut. Moreover, We provide the robustness analysis of the spectral heuristics under different Non-IID extent in the Appendix B.2, available online.

C. Failure Case Analysis

We evaluate two types of representative Byzantine-resilient methods (Robust statistics and the clustering based aggregation methods) and the proposed FedCut method (see Section V-B) under four typical Byzantine Attacks mentioned in Section IV-B, in terms of their Byzantine Tolerance Rates (BTR) defined below. Specifically, we assume 10 benign model updates following 1D Gaussian distribution $\mathcal{N}(0.1, 0.1)$ and see five types Byzantine attacks ($S1$-$S5$) with 8 malicious updates as illustrated in Table II.

Next, we provide a toy example of four types of Byzantine attacks mentioned above to illustrate failure cases of existing Byzantine-resilient methods.

In order to evaluate different Byzantine-resilient methods under the four scenarios, we define the Byzantine Tolerant Rate representing the fraction of Byzantine Tolerant cases over repeated runs against attacks as follows:

**Definition 1:** (Byzantine Tolerant Rate) Suppose the server receives $(K-q)$ correct gradients $V = \{v_1, \ldots, v_{K-q}\}$ and $q$ Byzantine gradients $U = \{u_1, \ldots, u_q\}$. A Byzantine robust method $A$ is said to be Byzantine Tolerant to certain attacks [60] if

$$<\text{avg}[V], \text{avg}[A([V \cup U])] >= 0$$  \hspace{1cm} (3)

Moreover, the Byzantine Tolerant Rate (BTR) for $A$ is the fraction of Byzantine Tolerant cases over repeated runs against attacks.

Table III summarized toy example results for different Byzantine-resilient methods under four types of Byzantine attacks mentioned above. We can draw following conclusions:

- First, for Non-Collusion attack ($S1$), all Byzantine-resilient methods except Kmeans [51] perform well (i.e., the BTR is higher than 90%), which indicates that Non-Collusion attack is easy to be defended.

- Second, for Collusion-diff attack ($S2$), Robust Statistics based methods such as Krum [6], Median, Trimmed Mean [65] and DnC [48] are vulnerable to $S2$-s. This failure is mainly ascribed to the wrong estimation of sample mean or median misled by biased model updates from colluders. Moreover, the clustering based method i.e., Kmeans [51]

---

### Table II

| Scenario | Attack Type       | Attack Distribution | Number |
|----------|-------------------|---------------------|--------|
| S1       | Non-Collusion     | $\mathcal{N}(0, 1)$ | 8      |
| S2-s     | Collusion-diff    | $\mathcal{N}(-2, 0.01)$ | 8      |
| S2-m     | Collusion-diff    | $\mathcal{N}(-2, 0.01)$ | 6      |
| S3       | Collusion-mimic   | $\mathcal{N}(\mu_1, 0.01)$ | 3      |
| S4       | Mixture           | $\mathcal{N}(\mu_1, 0.01)$ | 3      |
| S5-Krum  | $[60]$            | $0.01p_2^*$         | 8      |
| S5-Median| $[60]$            | $0.01p_2^*$         | 8      |

$^5\mu_1$ is the minimum elements of 10 benign model updates

$^6p_2$ is the sum of 10 benign model updates
TABLE III
FIVE SCENARIOS (S1-S5) OF BYZANTINE ATTACKS ARE REPEATED 1000 RUNS TO EVALUATE 5 REPRESENTATIVE BYZANTINE-RESILIENT METHODS AND THE PROPOSED FEDCUT METHOD, IN TERMS OF THEIR BYZANTINE TOLERANCE RATES (BTR) IN DEF. 1

| BTR | Krum [6] | Median [65] | Trimmed Mean [65] | DnC [48] | Kmeans [51] | FedCut (ours) |
|-----|----------|-------------|-------------------|---------|-------------|--------------|
| S1  | 96.0%    | 96.2%       | 92.6%             | 95.3%   | 70.5%       | 96.2%        |
| S2-s| 34.5%    | 29.5%       | 27.3%             | 89.6%   | 99.0%       | 99.0%        |
| S2-m| 86.1%    | 89.3%       | 88.2%             | 99.4%   | 3.5%        | 99.6%        |
| S3  | 29.6%    | 36.9%       | 37.7%             | 52.9%   | 33.7%       | 98.7%        |
| S4  | 59.9%    | 61.5%       | 67.6%             | 53.1%   | 85.6%       | 95.9%        |
| S5-Krum [60] | 24.8% | 43.9%       | 98.8%             | 37.4%   | 93.2%       | 99.8%        |
| S5-Median [60] | 35.5% | 31.7%       | 0.5%              | 98.9%   | 29.0%       | 99.5%        |

Fig. 4. FedCut: provide a temporal spatial cut to separate benign clients (green point) two-parties colluders (orange and red point) among all epochs.

fails in the S2-m, with BTR as low as 3.5%. This is because the clustering based method relies on the assumption that only one group of colluders exists, but two or more groups of colluders in S2-m are misclassified by naive clustering-based methods with wrong assumptions.

- Third, for Collusion-mimic attack (S3), both Robust Statistics based methods and clustering based method fail, with BTR lower than 52.9%. The main reason is that colluders would introduce statistical bias for benign updates and similar behaviours of colluders are hard to detect.
- Finally, the proposed FedCut method is able to defend against all attacks with high BTR (more than 95%) by using a Spatial-Temporal framework and spectral heuristics illustrated in Section V.

It is worth mentioning that the toy example used in this Section only showcases some simplified failure cases that might defeat existing Byzantine-resilient methods. Instead, the attacking methods evaluated in Section VII is more complex and detrimental, and we refer readers to thorough experimental results in Section VII and Appendix B, available online.

V. FEDCUT: SPECTRAL ANALYSIS AGAINST BYZANTINE COLLUDERS

This section illustrates the proposed spectral analysis framework (see Algorithm 1) in which distinguishing benign clients from one or more groups of Byzantine colluders is formulated as a community detection problem in Spatial-Temporal graphs [50], in which nodes represent all model updates and weighted edges represent similarities between respective pairs of model update over all training epochs (see Section V-A). The normalized graph cut with temporal consistency [39], [52] (called FedCut) is adopted to ensure that a global optimal clustering is found (see Section V-B). Moreover, the spectral heuristics [67] is then used to determine the Gaussian scaling factor, the number of colluder groups and the attack type (see Section V-C). The gist of the proposed method is how to discern colluders from benign clients, by scrutinizing similarities between their respective model updates (see Fig. 4 for an overview).

A. A Spatial-Temporal Graph in Federated Learning

In order to use all information during the training, we define a Spatial-Temporal graph by considering behaviours of clients among all training iterations:

**Definition 2:** (Spatial-Temporal Graph) Define a Spatial-Temporal graph \( G = (V, E) \) as a sequence of snapshots \( \langle G_1, \ldots, G_T \rangle \), where \( G_t = (V, E^t) \), \( t \in [T] \) is an undirected graph at epoch \( t \). \( V \) denotes a fixed set of \( K \) vertexes representing model updates belonging to \( K \) clients. \( E^t \) is a set of weighted edge representing similarities between model updates.
Algorithm 1: FedCut Framework.

Input: K clients with local training datasets $D_i, i = 1, 2, \ldots, |D_i|$; number of global epochs $T$; learning rate $\eta$ and batch size $b$.

Output: Global model $w$.

1: Initializing $w$, $L^0 = 0 \in \mathbb{R}^{K \times K}$.
2: for $t = 1, 2, \ldots, T$ do
3:     Step I: The server sends the global model $w$ to all clients $i = v_1, v_2, \ldots, v_K$.
4:     Step II: Training local models and server model.
5:     for $i = v_1, v_2, \ldots, v_K$ do parallel
6:         $g_t^i = \text{ModelUpdate}(w, D_i, b, \eta)$.
7:     end for
8:     Send $g_t$ to the server.
9: end for
10: Step III: Updating the global model via FedCut
11:    The set of chosen clients to aggregate $I_t$. Averaged normalized adjacency matrix $\hat{L}_t = C\text{-Cut}(g_t^i, \hat{L}_t^{t-1})$
12:    $g_t^i = \text{Aggregate}(g_t^{I_t})$, where $g_t^{I_t} = \{g_t^i | v_i \in I_t\}$
13:    $w \leftarrow w - \eta g_t$.
14: return $w$.

Algorithm 1: FedCut Framework.

according to the model $t \leftarrow T$, $V_C = \{i \mid L \text{Cut}(D_i, I_t)\}$ for any $1 \leq i, j \leq c$. Then the $c$-partition Ncut for Spatial-Temporal Graph aims to optimize:

$$
\min_{D(1 \cup \cdots \cup B_c)} \sum_{t=1}^{T} \sum_{c=1}^{c} \frac{W_t'(B_i, B_j)}{V_{\text{oll}}'(B_i)},
$$

where $B_i$ is the complement of $B_i$. $W_t'(B_i, B_j) := \frac{1}{2} \sum_{i_1 \in B_i, i_2 \in B_j} A_{i_1,i_2}$, $A_{i_1,i_2}$ is edge weight of $i_1$ and $i_2$, and $V_{\text{oll}}'(B_i) := \sum_{i_1 \in B_i} \sum_{j \in V} A_{i_1,j}$.

The proposed FedCut (Algorithm 2) is used to get rid of malicious updates before model updates are aggregated. FedCut computes adjacent matrix according to the uploaded gradients over all training epochs and then takes the normalized cut to distinguish benign clients and colluders. Specifically, three important parameters including the appropriate Gaussian kernel $\sigma_\ast$, the cluster number $c$ and the set of mimic colluders $\mathcal{I}$ (who behave similarly to benign clients) need to be first determined (see Algorithm 3). Then we calculate the adjacent matrix $A_t'$ at $t_{th}$ epoch to remove the mimic colluders, i.e., set the connection between the mimic colluders and benign clients to be zero (line 2-7 in Algorithm 2). We further compute normalized adjacency matrix $\tilde{L}_t'$ by induction over $t$ epochs (line 8-9 in Algorithm 2). Finally, we implement NCut into normalized adjacency matrix $\tilde{L}_t'$ to choose the clusters of benign clients (line 10-12 in Algorithm 2).

It is worth noting that FedCut runs over multiple learning epochs and, as shown by Proposition 2, it is guaranteed to provide the optimal $c$-partition Ncut defined by (4) (See Proof in Appendix C, available online).

Proposition 2: FedCut (Algorithm 2) solves the $c$-partition Ncut for Spatial-Temporal Graph i.e., it obtains an optimal solution by:

$$
\arg \min_{D(1 \cup \cdots \cup B_c)} \sum_{t=1}^{T} \sum_{c=1}^{c} \frac{W_t'(B_i, B_j)}{V_{\text{oll}}'(B_i)}.
$$

C. Spectral Heuristics

The determination of three parameters (i.e., the scaling parameter of Gaussian kernels $\sigma_\ast$, the number of clusters $c$ and mimic colluders $\mathcal{I}$) are critical in thwarting Byzantine Collusion attacks as illustrated in Section V-C and [55]. We adopt the Spectral Heuristics about the following definition of eigengap to determine these three important parameters.

Definition 4: The $i_{th}$ eigengap is defined as $|\lambda_i - \lambda_{i+1}|$ for which eigenvalues of the the normalized adjacency matrix $L = D^{-1/2}AD^{-1/2}$ are ranked in descending order, where $D = \text{diag}(\text{Sum}(A))$ and $A$ is adjacency matrix. Let $\delta$ be the largest eigengap among $|\lambda_i - \lambda_{i+1}|$.

1) Gaussian Kernel Determination: The Gaussian kernel scaling parameter $\sigma$ in similarity function $A_{i,j} = \exp(-||g_i - g_j||^2/2\sigma^2)$ controls how rapidly the $A_{i,j}$ falls off with the

We view the cluster with the largest numbers as benign clients which constitute the largest community among all clients.
distance between $g_i$ and $g_j$. An appropriate $\sigma$ is crucial for distinguishing Byzantine and benign clients. The following analysis shows that if the maximum eigengap $\delta$ (see Def. 4) is sufficiently large, a small perturbation on the normalized adjacency matrix $L$ or will only affect the eigenvectors with bounded influence, and thus the clustering of $c$ clusters via top $c$ eigenvectors are stable.

**Proposition 3 (Stability [54]):** Let $\lambda$, $Y$ and $\delta$ be eigenvalue, principle eigenvectors and the maximum eigengap of $L$ separately. Define a matrix small perturbation for $L$ as $\tilde{L} = L + E$, let $\lambda$, $Y$ be eigenvalue and principle eigenvectors of $\tilde{L}$. If the maximum eigengap $\delta > \sqrt{2||E||}$, then

$$||Y - \tilde{Y}|| \leq \frac{4||E||}{\delta - \sqrt{2||E||}}$$  \hspace{1cm} (5)

It is noted that the error bound $||Y - \tilde{Y}||$ in the principle eigenvectors is affected by two factors: the perturbation $||E||$ in the affinity matrix, and the maximal eigengap $\delta$. While the perturbation is unknown and unable to modify for a given matrix $L$, one can seek to maximize the eigengap $\delta$ so that the recovered principle eigenvectors are as stable as possible. According to proposition 3, we select the optimal parameter $\sigma$, by maximizing the maximum eigengap, i.e.,

$$\sigma^* = \arg\max_\sigma \delta(\sigma)$$  \hspace{1cm} (6)

We implement this strategy in Algorithm 3. Fig. 6 shows that one can select $\sigma$ as such that the largest eigengap is up to its maximal (e.g., $\log(\sigma)$ is in the range $(1,3.5)$ for the Gaussian attack), provided that the clustering accuracy achieves the largest value $(100\%)$ in this range. Also noted that the clustering accuracy drops in the extreme case (i.e., the $\sigma \geq 10^4$) even the maximum eigengap is still large. Therefore, we select the optimal $\sigma$ at one reasonable range with removing the extreme case (see detailed range in Section VII-A).

2) **Determination of the Number of Clusters:** The assumption made by many clustering-based Byzantine-resilient methods that model updates uploaded by all Byzantine attackers form a single group might be violated in the presence of collusion (as shown by the toy example in Section IV-C and experimental results in Section VII). Therefore, it is necessary to discover the number of clusters, and one possible way is to analyze the spectrum of the adjacency matrix, normalized adjacency matrix or correlation matrix [50]. According to the spectral property in Section IV-B, we determine the clustering number $c$ (see line 3-11 in Algorithm 3) based on the position of the largest eigengap of normalized adjacency matrix, i.e.,

$$c = \arg\max_k |\lambda_k - \lambda_{k+1}|$$  \hspace{1cm} (7)

Fig. 5 displays the change of eigengap under different attacks, which shows that the index of the largest eigengap is a good estimation of the number of clusters. Specifically,

- for Non-Collusion attack (i.e., Gaussian attack [6]. Label flipping [14] and Sign flipping [14]), the largest eigengap in Fig. 5(a), (b) and (c) consisting of 10, 20 and 30 attackers lies in 11, 21, and 31 with benign clients forming one group and 10, 20 and 30 individual attackers;

- for Collusion-diff attack, the largest eigengap of Fig. 5(e) and (f) lies in between second and third largest eigenvalues. The position of the largest eigengap indicates that there are two clusters (one cluster represents benign clients while the other cluster represents colluders);
Fig. 5. Change of eigengap for different attacks with different number of attackers under IID setting for MNIST dataset. From left to right, top to bottom, attack methods are Gaussian attack [6], Label flipping [14], Sign flipping [14], our designed collusion attack (see Appendix A, available online), same value attack [33], Fang-v1 (design for trimmed mean) [17], Mimic attack [27], and Lie [4] respectively.

Fig. 6. Change of the maximum eigengap of normalized adjacency matrix and clustering accuracy with different scaling factor $\sigma$ for the Gaussian attack [33], where the normalized adjacency matrix $L = D^{-1/2}AD^{-1/2}$, $D = \text{diag}(\text{Sum}(A))$ and $A_{ij} = \exp(-\|\mathbf{g}_i - \mathbf{g}_j\|^2 / 2\sigma^2)$. The clustering accuracy is calculated by NCut on the normalized adjacency matrix with 100 client including 70 benign clients and 30 Byzantine clients.

- for Collusion-mimic attack, the largest eigengap in Fig. 5(a), (b), and (c) of 10, 20 and 30 attackers lies in 90, 80, and 70 indicating that cluster numbers are 90, 80, and 70 (colluders form one group while other benign clients form 90, 80, and 70 groups separately);

In addition, our empirical results in Appendix B, available online also demonstrates the effectiveness of estimating the number of communities via the largest eigengap even for heterogeneous dataset among clients.

3) Mimic Colluders Detection: Colluders may mimic the behaviours of some benign clients towards over-emphasizing some clients and render other model updates useless. Therefore, it is hard to distinguish the mimic colluders and benign clients. The existing byzantine-resilient methods could not defend the Collusion-mimic attack (see Section IV-C for $S3$ case). We leverage the spectral property to detect the mimic colluders if the position of the largest eigengap is larger than $K_2$ (Proposition 1). Specifically, when the position of the largest eigengap is larger than $K_2$, we pick the clusters with number of clients larger than two as mimic colluders sets (see line 12-15 in Algorithm 3).

D. Adaptive Attack

In the spirit of adaptive attacks, e.g., [48], [69], we design the following adaptive attacks that knows the adaptation of FedCut. Specifically, the adaptive attack exploits the knowledge of FedCut defense mechanisms such that it can evade the defense in two steps:

First, in order to invalidate the estimation of number of clusters adopted by FedCut (see Section V-C2), attackers may aim to minimize the second largest eigenvalue of the normalized adjacency matrix (noted that the eigenvalue ranges from 0 to 1, and the largest eigenvalue must be 1). Therefore, this would cause the cluster number to be 1 and regard all clients to be benign clients. Second, in order to evade mimic colluders’ detection (where similar malicious updates are removed by spectral heuristics in Section V-C3), attackers may follow [48] to perturb malicious updates using Gaussian distribution $N(0, 1)$ with the large perturbing factors $\gamma^t (> \gamma_0)$. The formulation can be written as:

$$\arg \min_{\gamma^t} \lambda^t_2$$

$$g^t_b = \text{avg}(g^t_{i\in[K]}) + \gamma^t N(0, 1), \gamma^t > \gamma_0,$$

where $\lambda^t_2$ is the 2nd largest eigenvalue of the $\tilde{L}^t$, which is averaged over the normalized adjacency matrix $L^t$ from epoch 0 to $t - 1$, and $L^t$ is computed by $K$ updates $g^t_{i\in[K]}$.

VI. CONVERGENCE ANALYSIS

We prove the convergence of such an iterative FedCut algorithm. The proof for Theorem 2 uses the similar technique as [14] (see details in Appendix C, available online).

Theorem 1: Suppose an $0 < \alpha < \frac{1}{2}$ fraction of clients are Byzantine attackers. If Assumption 1 and 2 hold, we can find the estimate of $\hat{g}$ according to line 11 in Algorithm 1 and with
the probability $1 - O(\sqrt{Z})$, such that $\|g - \nabla F\| \leq O(\alpha \kappa)$, where $Z_1 = \frac{4K\sqrt{\alpha}(1-\alpha)Z}{\delta - K\sqrt{2\alpha(1-\alpha)Z}}$, $Z = \exp(-2\kappa^2/\sigma^2)$.

Remark 5: Theorem 1 illustrates that the distance between estimated gradients $g$ via FedCut and benign averaged gradients $\hat{g}$ is bounded by $O(\alpha \kappa)$. Moreover, $\alpha(1-\alpha)$ is increasing w.r.t $\alpha$ when $\alpha < \frac{1}{2}$, and $Z_1$ is increasing w.r.t $\alpha(1-\alpha)$. Therefore, $Z_1$ is increasing w.r.t $\alpha$, which indicates that the probability $1 - O(Z_1)$ decreases as the number of Byzantine attackers increases. In addition, for the larger $C$, $Z_1$ tends to zero since $Z < 1$. Consequently, $\|g - \nabla F\| \leq O(\kappa)$ with a high probability close to 1.

Assumption 3: The stochastic gradients sampled from any local dataset have uniformly bounded variance over $D_i$ for all benign clients, i.e., there exists a finite $\sigma_0$, such that for all $x_{i,j} \in D_i$, $i \in [K]$ and $j \in [n_i]$ (where $n_i$ is the number of dataset of client $i$), we have:

$$E_j(\|\nabla F_i(w_i, x_{i,j})\| - \nabla F_i(w_i))^2 \leq \sigma_0^2,$$  \hspace{1cm} (9)

where $\nabla F_i(w_i) = E_j(\nabla F_i(w_i, x_{i,j}))$.

Remark 6: The difference between Assumption 1 and 3 is that the former bounds the variance across gradient estimates within the same client while the latter bounds the variance between model updates across clients.

Assumption 4: We assume that $f(x)$ is $L$-smooth and has $\mu$-strong convex.

Theorem 2: Suppose an $0 < \alpha < \frac{1}{2}$ fraction of clients are corrupted. For a global objective function $F: R^d \rightarrow R$, the server obtains a sequence of iterates $\{w^t: t \in [0 : T]\}$ (see Algorithm 1) when run with a fixed step-size $\eta < \min\{\frac{1}{\mu}, \frac{1}{\mu j}\}$. If Assumption 1, 2, 3 and 4 hold, the sequence of average iterates $\{w^t: t \in [0 : T]\}$ satisfy the following convergence guarantees:

$$\|w^T - w^*\|^2 \leq \left(1 - \frac{C_1 \mu}{L}\right)^T \|w^0 - w^*\|^2 + \frac{\Gamma}{\mu^2},$$ \hspace{1cm} (10)

where $\Gamma = O(\sigma_0^2 + \sigma_2^2 + \alpha^2 \kappa^2)$, $C_1$ is a constant and $w^*$ is the global optimal weights in federated learning.

Theorem 2 provides the convergence guarantee of FedCut framework in the strong convex case. As $T$ tends to infinity, the upper bound of $\|w^T - w^*\|^2$ becomes large with the increasing of $\sigma_0$ (variance of gradients within the same client), $\kappa$ (variance between model updates across benign clients) and $\alpha$ (the ratio of Byzantine attackers).

VII. EXPERIMENTAL RESULTS

This section illustrates the proposed FedCut method's experimental results compared with existing Byzantine-robust methods. We refer to Appendix A and B, available online for the full report of extensive experimental results.

A. Setup and Evaluation Metrics

- Models: logistic regression, LeNet [32] and AlexNet [24] three models are used in all experiment settings. (see results for other models and datasets in supplementary material).
- Datasets: MNIST [31], Fashion-MNIST [57] and CIFAR10 [29] are used for image classification tasks. To simulate Non-IID settings, class labels assigned to clients follows a Dirichlet distribution $Dir(\beta)$ [34].
- Federated Learning Settings: We simulate a horizontal federated learning system with $K = 100$ clients in a standalone machine with 8 Tesla V100-SXM2 32 GB GPUs and 72 cores of Intel(R) Xeon(R) Gold 61xx CPUs. In each communication round, the clients update the weight updates, and the server adopts Fedavg [37] algorithm to aggregate the model updates. The detailed experimental hyper-parameters are listed in Appendix A, available online.

- Byzantine attacks: We set 10%, 20% and 30% clients, i.e., 10, 20, and 30 out of 100 clients are Byzantine attackers. The following attacking methods are used in experiments:
  - The same value attack: model updates of attackers are replaced by the all ones' vector;
  - The Sign flipping attack: local gradients of attackers are shifted by a scaled value -4;
  - The gaussian attack: local gradients at clients are replaced by independent Gaussian random vectors $N(0, 200)$;
  - The lie attack: it was designed in [4];
  - The Fang-v1 attack and Fang-v2 attack: they were designed in [17] for coordinate-wise trimmed mean [65] and Krum [6] respectively;
- Mimic attack: Colluders may mimic the behaviours of some benign clients towards over-emphasizing some clients and under-representing others [27].
- Our designed multi-collusion attack: adversaries are separated into 4 groups, and the same group has similar values. For example, each group is sampled from $N(\mu + \mu_i, 0.0001)$, and different groups have different $\mu_i$, where $\mu$ is the mean of uploaded gradients of all other benign clients.
- Byzantine-resilient methods: Nine existing methods i.e., Statistic-based methods: Krum [6], Median [65], Trimmed Mean [65], Bulyan [20] and DnC [48], Serve-evaluating methods: FLtrust [9], Clustering-based methods: Kmeans [51] and the proposed method FedCut are compared in terms of following metrics.
  - Gaussian kernel scaling parameters of FedCut: the pre-selected set of $\sigma$: $\{\sigma_1, \ldots, \sigma_n\}$ in Algorithm 3 is the geometric sequence with common ratio 2 of (1, $10\sqrt{10}$) for MNIST, (1, $10\sqrt{10}$) for Fashion-MNIST and (0.1, $10\sqrt{10}$) for CIFAR10.

Evaluation metric: two types of metrics are used in our evaluation.

- Model accuracy (MA) of the federated model is used to evaluate defending capabilities of different methods. In order to elucidate robustness of each defending method, we also report respective averaged and worst-case model accuracy under all possible attacks.
- For detection methods, which detect malicious clients and remove before aggregation, the detection accuracy (DACC), false positive rate (FPR), and false negative rate
TABLE IV
MODEL ACCURACY OF DIFFERENT BYZANTINE-RESILIENT METHODS UNDER DIFFERENT BYZANTINE ATTACKS (WITH IID SETTING 30 BYZANTINE CLIENTS FOR CLASSIFICATION OF MNIST, FASHION MNIST AND CIFAR10)

| Method          | No attack | Krum[6] | GeoMedian[12] | Median[63] | Trimmed[63] | Bulyan[20] | FLtrust[9] | Dnc[48] | Kmeans[51] | FedCut (Ours) |
|------------------|-----------|---------|---------------|------------|-------------|------------|------------|---------|-----------|---------------|
| MNIST            | 90.5±0.1  | 92.4±0.0| 90.5±0.1      | 93.0±0.1   | 89.6±0.1    | 89.4±0.1   | 92.2±0.3   | 92.4±0.3| 92.5±0.1   |               |
| FNR              | 89.5±1.0  | 87.9±1.0| 87.3±1.0      | 87.3±1.0   | 88.9±1.0    | 88.3±1.0   | 92.7±0.1   | 92.1±0.2| 93.2±0.2   |               |
| Lie [4]          | 90.5±0.0  | 84.4±0.6| 84.0±1.3      | 83.6±0.8   | 73.6±1.0    | 89.5±0.1   | 92.2±0.3   | 92.2±0.1| 93.2±0.2   |               |
| Fang-v1[17]      | 90.2±0.1  | 45.8±0.5| 43.4±1.5      | 37.8±2.6   | 85.5±3.0    | 75.8±3.5   | 93.2±0.3   | 93.3±0.2| 93.2±0.1   |               |
| Fang-v2[17]      | 41.2±7.4  | 43.2±11 | 42.4±2.8      | 18.8±6.1   | 79.7±0.8    | 92.4±0.6   | 88.2±0.4   | 92.3±0.1| 93.2±0.1   |               |
| Same value [33]  | 90.6±0.1  | 85.6±0.0| 77.0±1.4      | 75.1±0.6   | 88.8±0.4    | 89.8±0.3   | 92.3±0.0   | 92.3±0.2| 92.1±0.1   |               |
| Gaussian [6]     | 94.0±0.1  | 92.4±0.1| 90.6±0.1      | 90.8±0.1   | 89.6±0.3    | 87.0±1.0   | 74.3±0.4   | 24.4±4.3| 93.2±0.1   |               |
| Sign flipping [14]| 94.0±0.1  | 93.1±0.3| 90.2±0.1      | 90.2±0.1   | 89.6±0.2    | 72.5±5.5   | 90.7±0.1   | 49.9±1.3| 91.9±0.2   |               |
| Label flipping [14]| 94.0±0.1 | 89.4±0.1| 85.6±0.2      | 85.4±0.4   | 89.5±0.4    | 89.3±0.6   | 92.2±0.2   | 92.1±0.2| 92.1±0.4   |               |
| FNR              | 89.5±1.0  | 88.3±1.0| 86.3±0.3      | 89.7±0.2   | 85.5±0.5    | 95.0±1.1   | 92.2±0.4   | 91.0±0.2| 92.3±0.1   |               |
| Averaged         | 85.1±0.9  | 80.5±0.7| 76.8±0.6      | 76.3±0.9   | 80.0±1.0    | 85.3±1.2   | 90.1±0.3   | 75.7±2.8| 92.0±0.2   |               |
| Worst-case       | 41.1±7.4  | 45.8±0.5| 39.2±2.3      | 37.8±2.6   | 18.8±6.1    | 72.2±3.5   | 74.3±0.4   | 24.4±3.5| 91.9±0.2   |               |
| CIFAR10          | 90.5±0.1  | 88.2±0.1| 88.4±0.4      | 87.3±0.3   | 88.2±0.5    | 90.3±0.2   | 90.0±0.3   | 91.0±0.3| 91.0±0.3   |               |
| FNR              | 89.5±1.0  | 88.3±1.0| 86.3±0.3      | 89.7±0.2   | 85.5±0.5    | 95.0±1.1   | 92.2±0.4   | 91.0±0.2| 92.3±0.1   |               |
| Averaged         | 85.5±0.2  | 89.6±0.3| 72.6±5.1      | 78.5±4.8   | 87.1±3.0    | 89.7±0.6   | 93.6±0.4   | 88.8±0.6| 86.4±0.4   |               |
| Worst-case       | 77.2±1.6  | 68.7±2.5| 74.6±4.5      | 74.9±4.0   | 73.5±1.7    | 85.5±2.0   | 83.0±2.8   | 73.8±4.6| 84.9±0.5   |               |

Moreover, the baseline of FedAvg [37] without any attacks achieve the model accuracy 92.5%, 90.1% and 69.4% for MNIST, Fashion MNIST and CIFAR10 respectively.

(FNR) among are used to quantify their defending capabilities [69]. Noted that we use averaged values of DACC, FPR and FNR among all training epochs in this paper. Note that this detection accuracy is measured against ground-truth Byzantine attackers’ membership, and it should not be confused with the image classification accuracy of the main task.

B. Comparison With Other Byzantine-Resilient Methods

Table IV summarizes model accuracy (MA) of 9 existing methods as well as the proposed FedCut method for classification of MNIST, Fashion MNIST, and CIFAR10 using logistic regression, LeNet, and AlexNet respectively under IID setting and 30 attackers (see Appendix B.6, available online for more results with other settings). There are four noticeable observations.

- FedCut performs robustly under almost all attacks with worst-case MA above 92.0% for MNIST, 87.5% for Fashion MNIST and 63.1% for CIFAR10. This robust performance is in sharp contrast with eight existing methods, each of which has a worst-case MA degradation ranging from 18.2% (i.e., DnC by Gaussian attack reaching 74.3% MA) to 73.7% (i.e., Bulyan by Fang-v2 merely having 18.8% MA) on MNIST.

- In terms of averaged MA, it is clearly observed that FedCut outperformed eight existing methods by noticeable margins ranging between 2.1% to 16.5% on MNIST, 3.9% to 20.7% on Fashion MNIST and 10.0% to 43.6% on CIFAR10.
A. MITH (DACC), F shows. For instance, the NFF drops to 74.6% from 88.6% in MNIST μ7. Existing clustering-based method, i.e., K-means performs NRR (K drops to ON AND F OF μ displays the model accuracy under differ-
β (FNR) B PIGHT is the mean of uploaded gradients of all other benign =0 (IID, N=0). AND D ON =0 β shows In the VII VI numbers (yellow: 10 attackers, blue: 20 attackers, green: 30 attackers) on Fashion MNIST.

Table V

| Dataset/Non-IID extent | K-means | DnC | NCut | FedCut |
|------------------------|---------|-----|------|--------|
|                       | DACC    | FPR | FNR  | DACC   | FPR | FNR  | DACC   | FPR | FNR  | DACC   | FPR | FNR  |
| IID                    | 0.88    | 0.21 | 0.12 | 0.93   | 0.16 | 0.06 | 0.88   | 0.05 | 0.13 | 1.00   | 0.00 | 0.00 |
| MNIST Non-IID1         | 0.84    | 0.30 | 0.16 | 0.89   | 0.16 | 0.11 | 0.84   | 0.16 | 0.16 | 0.97   | 0.04 | 0.03 |
| Non-IID2               | 0.82    | 0.37 | 0.17 | 0.85   | 0.19 | 0.15 | 0.77   | 0.28 | 0.23 | 0.88   | 0.14 | 0.12 |
| F-MNIST Non-IID1       | 0.86    | 0.51 | 0.14 | 0.81   | 0.22 | 0.19 | 0.81   | 0.20 | 0.19 | 0.97   | 0.04 | 0.03 |
| Non-IID2               | 0.63    | 0.56 | 0.16 | 0.79   | 0.33 | 0.21 | 0.72   | 0.26 | 0.28 | 0.88   | 0.21 | 0.12 |

Fig. 7. Averaged and worst-case model accuracy among all 8 attacks of different Byzantine-resilient methods with IID setting for different byzantine numbers (yellow: 10 attackers, blue: 20 attackers, green: 30 attackers) on Fashion MNIST.

Fig. 8. Averaged and worst-case model accuracy among all 8 attacks of different Byzantine-resilient methods with 30 Byzantine attackers for different Non-IID extent (yellow: Non-IID with β = 0.1, blue: Non-IID with β = 0.5, green: IID) on Fashion MNIST.

- Robust Statistics based methods (e.g., Krum, GeoMedian, Median, and Trimmed Mean) were overwhelmed by Collision-diff and Collusion-mimic attacks (such as the MA drops 49.1% for Median under Fang-v1 attack and the MA drops 51.4% for Krum under Fang-v2 attack), incurred a significant MA degradation on MNIST. In contrast, all Collusion attacks cause a minor MA loss of less than 0.5% to FedCut.

- Existing clustering-based method, i.e., K-means performs robustly in the presence of single-group colluders with MA degradation less than 2% (e.g., K-means for Same Value attack on MNIST, Fashion MNIST and CIFAR10), but incurred significant MA degradation more than 50% in the face of multi-group colluders attackers (e.g., K-means for the Multi-Collusion attack on MNIST and Gaussian attack on Fashion MNIST). In contrast, multi-group collusion doesn’t cause significant MA loss to the proposed FedCut, which adopts spectral heuristics to make an estimation of the number of colluder groups (see Section V-C). Correspondingly, the detection accuracy (DACC), false positive rate (FPR) and false negative rate (FNR) of FedCut is better than other methods, as Table V shows. For instance, the DACC, FPR and FNR of FedCut on MNIST in the IID setting is 1, 0 and 0 respectively, while the DACC of other methods drop below 93%, FPR is greater than 0.05 and FNR is higher than 0.06.

C. Robustness

In this subsection, we demonstrate the robustness of our FedCut framework under different byzantine numbers, the Non-IID extent of clients’ local data, and multiple collusion attacks.

1) Robustness Under Varying Numbers of Attackers: Fig. 7 shows the model accuracy (MA) for different Byzantine-resilient methods under different types of attacks for different Byzantine attackers, i.e., 10, 20, and 30 attackers. The result shows that the MA of FedCut doesn’t drop with the increase of Byzantine numbers while the MA of others drops seriously (e.g., the averaged MA of Median [65] drops to 74.6% from 88.6% in Fashion MNIST dataset). Clearly, our proposed method, FedCut, is robust for the the number of byzantine attackers.

2) Robustness Under Heterogeneous Dataset: Fig. 8 shows the model accuracy (MA) for different Byzantine resilient methods under different types of attacks for different Non-IID extent of clients’ datasets. The result demonstrates that the MA of FedCut drops no more than 3% as the clients’ local dataset becomes more heterogeneous while the MA of others drops seriously (e.g., the averaged MA of trimmed mean [65] drops to 55.4% from 74.9% in Fashion MNIST dataset).

3) Robustness Against the Multi-Collusion Attack: In the former part, we design the ‘Multi-Collusion attack’ in which adversaries are separated into four groups, and the same group number of colluders’ parties as follows: 30 attackers are separated into 1) two groups with each group having 15 attackers; 2) three groups with each group having ten attackers 3) four groups with each group having 8, 8, 7, 7 attackers respectively; 4) five groups with each group has six attackers.

Tables VII and VI displays the model accuracy under different “collusion attacks”, which illustrates that various collusion attacks do not influence FedCut and NCut while other byzantine...
TABLE VI
MODEL ACCURACY OF DIFFERENT BYZANTINE-RESILIENT METHODS UNDER DIFFERENT GROUPS OF COLLUSION ATTACKS (WITH NON-IID SETTING β = 0.5 AND 30 ATTACKERS FOR CLASSIFICATION OF MNIST)

| Method       | Krum [6] | Geo-Median [12] | Median [65] | Trimmed [65] | DnC | Kmeans | FedCut (Ours) |
|--------------|-----------|------------------|-------------|--------------|-----|--------|--------------|
| 1-group      | 0.80±0.01 | 0.79±0.02        | 0.83±0.01   | 0.82±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 2-group      | 0.78±0.01 | 0.79±0.02        | 0.81±0.01   | 0.82±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 3-group      | 0.76±0.01 | 0.79±0.02        | 0.80±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 4-group      | 0.74±0.01 | 0.79±0.02        | 0.79±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 5-group      | 0.72±0.01 | 0.79±0.02        | 0.79±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |

TABLE VII
MODEL ACCURACY OF DIFFERENT BYZANTINE-RESILIENT METHODS UNDER DIFFERENT GROUPS OF COLLUSION ATTACKS (WITH IID AND 30 ATTACKERS FOR CLASSIFICATION OF MNIST)

| Method       | Krum [6] | Geo-Median [12] | Median [65] | Trimmed [65] | DnC | Kmeans | FedCut (Ours) |
|--------------|-----------|------------------|-------------|--------------|-----|--------|--------------|
| 1-group      | 0.80±0.01 | 0.79±0.02        | 0.83±0.01   | 0.82±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 2-group      | 0.78±0.01 | 0.79±0.02        | 0.81±0.01   | 0.82±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 3-group      | 0.76±0.01 | 0.79±0.02        | 0.80±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 4-group      | 0.74±0.01 | 0.79±0.02        | 0.79±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |
| 5-group      | 0.72±0.01 | 0.79±0.02        | 0.79±0.01   | 0.81±0.01    | 0.79±0.02 | 0.78±0.01 | 0.79±0.01   |

resilient methods such as Kmeans are affected seriously by collusion attack. Note that DnC performs well with 2-groups collusion, but MA drops when colluders’ groups increase.

D. FedCut Under the Adaptive Attack

Under the adaptive attack designed in Section V-D, we first determine a reasonable γ without a threshold. For instance, Fig. 9 shows that when the perturbing factor is larger than 1e-2, the adaptive attack would evade the colluder mimic detection. Thus, we can choose γ to be 1e-2. Second, under the constraint γ ≤ γ, we optimize the γ by minimizing the λ_2. Figs. 9 and 10 show the detection accuracy of Byzantine clients (DACC) and model accuracy of FedCut against the designed adaptive attack. It illustrates the effectiveness of adaptive attack, e.g., the drop of DACC and model accuracy on MNIST is 6.0% and 1.5% respectively. However, the adaptive attack tries to evade detection of mimic colluders only due to large scaling factors, which is insufficient to poison FedCut since large scaling factors would make malicious updates far away from the benign updates. In summary, it is observed that FedCut is robust against this designed adaptive attack.

E. Computation Complexity

We compare the computation time complexity for different Byzantine resilient methods for each epoch on the server as Table VIII. It shows that the computation complexity of our proposed method, FedCut, can be decomposed into three components. First, computing the adjacency matrix requires O(K^2 d) operations, where K represents the number of clients and d denotes the dimension of the model. Second, determining the sets of mimic colluders necessitates O(n K^3) operations, where n corresponds to the size of the preselection set of scaling factors. Finally, implementing the singular value decomposition (SVD) requires O(K^3) operations, as described in previous work [22]. The time complexity of FedCut O(K^2 d + nK^3) is comparable to statistic-based methods such as Krum [6] (O(K^2 d)) and Median [65] (O(K^2 d)) when the dimension of model d is greatly larger than number of participants K.

Empirically, the proposed FedCut method allows the training of a DNN model (AlexNet) with 100 clients to run in three hours, under a simulated environment (see Appendix A, available online for details). We regard this time complexity is reasonable for practical federated learning applications across multiple institutions. For cross-devices application scenarios in which K might be up to millions, randomized SVD [22] can be adopted to improve the time complexity from O(K^3) to O(K) if the normalized adjacency matrix has low rank. Applying randomized SVD to cross-device FL scenarios is one of our future work.

VIII. CONCLUSION

This paper proposed a novel spectral analysis framework, called FedCut, to detect Byzantine colluders robustly and efficiently from the graph perspective. Specifically, our proposed
algorithm FedCut ensures the optimal separation of benign clients and colluders in the spatial-temporal graph constructed from uploaded model updates over different epochs. We analyze existing Byzantine attacks and Byzantine-resilient methods in the lens of spectral analysis. It was shown that existing Byzantine-resilient methods may suffer from failure cases in the face of Byzantine colluders. Moreover, spectral heuristics are used in FedCut to determine the number of colluder groups, the scaling factor and mimic colluders, which significantly improves the Byzantine tolerance in colluders detection. Our extensive experiments on multiple datasets and theoretical convergence analysis demonstrate that our proposed framework can achieve drastic improvements over the FL baseline in terms of model accuracy under various Byzantine attacks. Moreover, the proposed and many existing Byzantine-resilient methods assume that the server could access to clients’ model updates to compute the similarity. However, the leakage of model updates allows a semi-honest server to infer some information of the private data [70]. Therefore, Byzantine problem should be considered when implementing FL for privacy-preserving applications. In future work, we will explore to what extent FedCut can be used in conjunction with Differential Privacy [1] or Homomorphic Encryption mechanisms [68].

Finally, The proposed FedCut can be applied in cross-device setting where each epoch only a fraction of clients uploads their updates [26], [49]. We can build a graph \( G \) including all clients in each epoch \( t \), and normalized adjacency matrix \( \tilde{L} \) for graph \( G \) including chosen clients by the server at epoch \( t \) can be computed according to \( \tilde{L} = \tilde{L}^1 \) to remove malicious updates.

**References**

[1] M. Abadi et al., “Deep learning with differential privacy,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2016, pp. 308–318.

[2] D. Alistarh, Z. Allen-Zhu, and J. Li, “Byzantine stochastic gradient descent,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 4613–4623.

[3] Z. Allen-Zhu, F. Ebrahimiaghazani, J. Li, and D. Alistarh, “Byzantine-resilient non-convex stochastic gradient descent,” in Proc. Int. Conf. Learn. Representations, 2020, pp. 1–11.

[4] G. Baruch, M. Baruch, and Y. Goldberg, “A little is enough: Circumventing defenses for distributed learning,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 8635–8645.

[5] J. Bernstein and J. Zhao, K. Azizzadenesheli, and A. Anandkumar, “SignSGD with majority vote is communication efficient and fault tolerant,” in Proc. Int. Conf. Learn. Representations, 2018, pp. 1–12.

[6] P. Blanchard, E.M.E. Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 118–128.

[7] E. A. Brouwer and W. H. Haemers, Spectra of Graphs. Berlin, Germany: Springer, 2011.

[8] S. Buhus, V. Gandikota, A. Mazumdar, A. S. Rawat, and P.K. Varshney, “Byzantine resilient distributed clustering with redundant data assignment,” in Proc. IEEE Int. Symp. Inf. Theory, 2021, pp. 2143–2148.

[9] X. Cao, M. Fang, J. Liu, and N. Z. Gong, “FLTrust: Byzantine-robust federated learning via trust bootstrapping,” 2020, arXiv:2012.13995.

[10] C. Chen, L. Lyu, Y. Liu, F. Wu, C. Chen, and G. Chen, “Byzantine-resilient federated learning via gradient memorization,” in Proc. Int. Workshop Trustable, Verifiable Auditable Federated Learn. Conjunc. AAAI, 2022.
[41] K. Pillutla, S. M. Kakade, and Z. Harchaoui, “Robust aggregation for federated learning,” 2019, arXiv: 1912.13445.

[42] S. Prakash and A. S. Avestimehr, “Mitigating Byzantine attacks in federated learning,” 2020, arXiv: 2010.07541.

[43] E. Gigante, H. Z. Charles, and D. Papailiopoulos, “DETOX: A redundancy-based framework for faster and more robust gradient aggregation,” in Proc. Adv. Neural Inf. Process. Syst. 2019.

[44] J. Regattii, H. Chen, and A. Gupta, “ByGARS: Byzantine SGD with arbitrary number of attackers,” 2020, arXiv:2006.13421.

[45] C. Sarkar and S. Jalan, “Spectral properties of complex networks,” Chaos: An Interdisciplinary J. Nonlinear Sci., vol. 28, no. 10, 2018, Art. no. 102101.

[46] F. Sattler, K.-R. Müller, and W. Samek, “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 8, pp. 3710–3722, Aug. 2020.

[47] F. Sattler, K.-R. Müller, T. Wiegand, and W. Samek, “On the Byzantine robustness of clustered federated learning,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2020, pp. 8861–8865.

[48] V. Shejwalkar and A. Houmansadr, “Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning,” in Proc. Netw. Distrib. Syst. Secur. Symp., 2021, pp. 1–17.

[49] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, “Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning,” in Proc. IEEE Symp. Secur. Privacy, 2022, pp. 1354–1371.

[50] H.-W. Shen and X.-Q. Cheng, “Spectral methods for the detection of network community structure: A comparative analysis,” J. Stat. Mechanics: Theory Exp., vol. 10, 2010, Art. no. P10020.

[51] S. Shen, S. Tople, and P. Saxena, “Auror: Defending against poisoning attacks in collaborative deep learning systems,” in Proc. 32nd Annu. Conf. Comput. Secur. Appl., 2016, pp. 508–519.

[52] J. Shi and J. Malik, “Normalized cuts and image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 888–905, Aug. 2000.

[53] J.-Y. Sohn, D.-J. B. H. Choi, and J. Moon, “Election coding for distributed learning: Protecting signals against Byzantine attacks,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 14615–14625.

[54] G. W. Stewart and J. guang Sun, Matrix Perturbation Theory, Academic Press, 1990.

[55] U. VonLuxburg, “A tutorial on spectral clustering,” Statist. Comput., vol. 17, no. 4, pp. 395–416, 2007.

[56] J. D. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” Nature, vol. 393, no. 6684, pp. 440–442, 1998.

[57] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms,” 2017, arXiv:1708.07747.

[58] C. Xie, O. Koyejo, and I. Gupta, “Generalized byzantine-tolerant SGD,” 2018, arXiv:1802.10116.

[59] C. Xie, O. Koyejo, and I. Gupta, “Zeno: Byzantine-suspicous stochastic gradient descent,” 2018, arXiv:1805.10032.

[60] C. Xie, O. Koyejo, and I. Gupta, “Full of empires: Breaking Byzantine-tolerant SGD by inner product manipulation,” in Proc. Uncertainty Artif. Intell., 2020, pp. 261–270.

[61] C. Xie, S. Koyejo, and I. Gupta, “Zeno: Robust fully asynchronous SGD,” Int. Conf. Mach. Learn., 2019, pp. 10945–10953.

[62] H. Yang, X. Zhang, M. Fang, and J. Liu, “Byzantine-resilient stochastic gradient descent for distributed learning: A lipschitz-inspired coordinate-wise median approach,” in Proc. IEEE 58th Conf. Decis. Control, 2019, pp. 5832–5837.

[63] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, 2019.

[64] Y.-R. Yang and W.-J. Li, “BASGD: Buffered asynchronous SGD for byzantine learning,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 11751–11761.

[65] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 5656–5659.

[66] H. Yu, R. Jin, and S. Yang, “On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 7184–7193.

[67] L. Zharko-Manor and P. Perona, “Self-tuning spectral clustering,” in Proc. Adv. Neural Inf. Process. Syst., 2004, pp. 1601–1608.

[68] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, “[BatchCrypt]: Efficient homomorphic encryption for (Cross-Silo) federated learning,” in Proc. USENIX Annu. Techn. Conf., 2020, pp. 493–506.

[69] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, “FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients,” in Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining, 2022, pp. 2545–2555.

[70] L. Zhu, Z. Liu, and S. Han, “Deep leakage from gradients,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 14774–14784.

Hanlin Gu received the BS degree in mathematics from the University of Science and Technology of China, in 2017, and the PhD degree in mathematics from the Hong Kong University of Science and Technology, in 2022. He now works as a student researcher with WeBank AI Group, WeBank. His research interests include federated learning and privacy-preserving methodology. He has published a series paper in TPAMI, IJCAI, PAKDD, and so on.

Lixin Fan (Member, IEEE) is the principal scientist of Artificial Intelligence with WeBank and the chairman of the Federal Learning Industry Ecological Development Alliance. His research fields include machine learning and deep learning, computer vision and pattern recognition, image and video processing, 3D Big Data processing, data visualization and rendering, augmented and virtual reality, mobile computing and ubiquitous computing, and intelligent man-machine interface. He is the author of more than 70 international journals and conference articles. He has worked at Nokia Research Center and Xerox Research Center Europe. His research includes the well-known Bag of Keypoints image classification method. He has participated in NIPS/NeurIPS, ICML, CVPR, ICCV, ECCV, IJCAI, and other top artificial intelligence conferences for a long time, served as area chair of AAAI, and organized workshops in various technical fields. He is also the inventor of more than one hundred patents filed in the United States, Europe and China, and the chairman of the IEEE P2894 Explainable Artificial Intelligence (XAI) Standard Working Group.

XingXing Tang is currently working toward the PhD degree in the Department of Computer Science and Engineering with the Hong Kong University of Science and Technology. From 2018 to 2023, he served as a researcher in the WeBank AI Group. He was previously the head of the AI department with Qunar Insurance. His research interests include federated learning, large language models, and computer vision.

Qi Yang (Fellow, IEEE) received the PhD degree from the Department of Computer Science, University of Maryland, College Park, in 1989 and had been a faculty member with the University of Waterloo between 1989 and 1995. He is a fellow of Canadian Academy of Engineering (CAE) and Royal Society of Canada (RSC), Chief Artificial Intelligence Officer of WeBank, a chair professor of Computer Science and Engineering Department at Hong Kong University of Science and Technology (HKUST). He is the Conference Chair of AAAI-21, the Honorary Vice President of Chinese Association for Artificial Intelligence (CAA), the President of Hong Kong Society of Artificial Intelligence and Robotics (HKSAIR), the President of Investment Technology League (ITL) and Open Islands Privacy-Computing Development Alliance. His research fields include artificial intelligence, machine learning, data mining, and planning. He had been the Founding Director of the Huawei’s Noah’s Ark Research Lab between 2012 and 2015, the Founding Director of HKUST’s Big Data Institute, the Founder of 4Paradigm and the President of IJCAI (2017-2019). His latest books are Transfer Learning, Federated Learning, Privacy-preserving Computing and Practicing Federated Learning.