CHARACTERIZING THE TOPOLOGICAL DISTRIBUTION OF COMPROMISED CLIENTS IN DECENTRALIZED FEDERATED LEARNING USING MIXING MATRICES

Anonymous authors
Paper under double-blind review

ABSTRACT

Decentralized federated learning enables interconnected clients to collaboratively train the machine learning model over a communication graph. Since clients distributed across different positions in the graph propagate information along different trajectories, the attack performance of compromised clients depends on both the attack strategy and the topological distribution of compromised clients (TDCC). However, most existing studies evaluate attacks by selecting compromised clients at random, thereby overlooking the heterogeneous ability of different clients to propagate malicious information. In this paper, we investigate how TDCC affects attack performance in a DFL system and propose Maximum Client-Level Accumulated Model Deviation (Max-CL-AMD), a compromised client selection method based on the system's mixing matrices. Specifically, we model the system-level accumulated model deviation (AMD), which captures the overall deviation induced by compromised clients, and then decompose it into client-level AMD components, each quantifying the deviation attributable to an individual compromised client. Extensive experiments across diverse settings show that Max-CL-AMD can achieve varying degrees of improvement in attack performance over random selection and other baseline selection methods.

1 INTRODUCTION

In a Decentralized Federated Learning (DFL) (He et al., 2021; 2019) system, numerous clients collaboratively train the machine learning model by performing local training and exchanging model updates only with their one-hop neighbors over a communication graph, without sharing their local data. Although DFL removes the need for the central server used in conventional Federated Learning (FL) (McMahan et al., 2017; Chen et al., 2021; Le et al., 2024), it remains vulnerable to many of the same security threats as conventional FL. In particular, compromised clients in the system may craft stealthy and malicious model updates to undermine the system (Pasquini et al., 2023; Jia et al., 2023; Xie et al., 2025). The influence of these updates spreads to the entire system through the central server in FL, whereas it propagates progressively over the communication graph in DFL, ultimately degrading model performance or even causing the entire system to fail (Fang et al., 2020). However, the topological distribution of compromised clients (TDCC) is immaterial in centralized FL because every client has the same one-hop relationship with the central server. In DFL, by contrast, clients occupy distinct positions and have different neighborhoods in the communication graph, so malicious updates originating from different clients may follow different propagation trajectories. Therefore, directly adapting attack methods (Fang et al., 2020; Baruch et al., 2019; Wu et al., 2025) originally designed for FL to DFL overlooks a key characteristic of the DFL system: the propagation trajectories of malicious information over the communication graph depends on the TDCC of the system. Motivated by this observation, we investigate how TDCC in DFL systems affects attack performance in DFL, which we refer to as the TDCC problem.

Existing studies on the TDCC problem remain limited. Most of these approaches select clients to compromise using only node-level structural metrics, such as node degree, effective network size, clustering coefficient, and PageRank score (Syros et al., 2024). Alternatively, MaxSpAN-FL seeks to approximately maximize the average pairwise distance among selected clients, and Hopping-Augmented MaxSpAN-FL extends this strategy by incorporating node centrality through an adaptive

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

hopping mechanism (Piaseczny et al., 2024). These methods do not explicitly account for key characteristics of DFL, such as information propagation over the communication graph, the dynamics of model mixing and even data heterogeneity. A recent study, Greedy-BPI and Swap-BPI (Gabrielli & Tolomei, 2026), directly accounts for weighted multi-hop propagation and interactions among compromised client, but is restricted to linear aggregation. Notably, nonlinear (robust) aggregation plays a crucial role in DFL security. To bridge these gaps, we study the TDCC problem by considering the distinctive characteristics of DFL in general scenarios.

In a DFL system, the mixing matrix (Le Bars et al., 2023; Chiu et al., 2023; Vogels et al., 2023; Neglia et al., 2020) specifies how each client aggregates updates received from its own neighbors (Lian et al., 2017; Boyd et al., 2004). More specifically, it contains information about communication topology, trajectories of model mixing, aggregation rule and even data heterogeneity (Le Bars et al., 2023; Dandi et al., 2022). First, this matrix encodes the communication topology through its support matrix. Second, products of mixing matrices describe the mixing (aggregation) trajectories of information over the communication graph (Neglia et al., 2020; Vogels et al., 2023). Third, and most notably, the aggregation rules (Cao et al., 2020; Fang et al., 2024) directly determine the values in the mixing matrix. Motivated by these properties of mixing matrix, we investigate how TDCCs affect the attack performance by analyzing mixing matrices. Specifically, we define an accumulated mixing matrix to quantify the accumulated model deviation of entire system, termed System-Level Accumulated Model Deviation (System-Level AMD). And then, we decompose an upper bound on System-Level AMD in terms of Client-Level AMD, which characterizes each client's potential contribution to the system-level deviation. This decomposition motivates Maximum Client-Level AMD (Max-CL-AMD), a method that selects the clients with the highest Client-Level AMD for compromise. Unfortunately, this method would require ground-truth mixing matrices before training. In practice, these matrices cannot be obtained before TDCC is available and may not exist in some cases. We therefore use estimated mixing matrices to compute the client-level AMD for each potentially compromised client.

The contributions of this work are summarized as follows: (1) We formulates the TDCC problem using mixing matrices, characterizing how TDCC affects the propagation of malicious updates and the resulting attack performance. (2) We proposes Max-CL-AMD, a method for selecting compromised clients. Max-CL-AMD identifies clients with high accumulated model deviation by analyzing accumulated mixing matrices. (3) We evaluates Max-CL-AMD across a wide range of settings. The results demonstrate that Max-CL-AMD consistently outperforms random selection and some baselines.

2 Related Works

We first review prior work on TDCC in DFL systems and then examine how it has been addressed in other graph-based systems.

TDCC in DFL systems. Since TDCC problem remains underexplored, there are a few studies about how to distribute the compromised clients in DFL systems. Syros et al. (2024) investigate a class of backdoor attacks against DFL in which malicious nodes are selected solely according to structural properties, including node degree, effective network size and etc. Piaseczny et al. (2024) propose two strategies for placing adversarial clients in a DFL system: MaxSpAN-FL and Hopping-Augmented MaxSpAN-FL. MaxSpAN-FL aims to maximize the spatial separation among adversarial clients, whereas Hopping-Augmented MaxSpAN-FL extends this strategy by introducing a probabilistic hopping mechanism guided by node centrality. A recent study (Gabrielli & Tolomei, 2026) introduce Byzantine Placement Influence, a finite-time and set-level measure, to identify high-influence Byzantine sets. Overall, existing studies determine TDCCs in DFL primarily from structural properties of graph, without considering the key characteristics of DFL.

TDCC in graph-based systems. Existing studies on TDCC in graph-based systems have mainly focused on selecting nodes that are highly connected or structurally influential according to graph centrality measures. Maximum Degree (Syros et al., 2024) selects nodes with the highest degrees, allowing attackers to propagate malicious information to a large number of neighbors. Maximum Effective Network Size (Burt, 2018) selects nodes with the largest effective network sizes, which reflect the numbers of nonredundant neighbors connected to these nodes. Maximum PageRank Score (Page et al., 1999) selects nodes with the highest PageRank scores. Originally developed

to rank web pages for the Google search engine, PageRank quantifies a node's global structural importance based on the graph's link structure, assigning higher scores to nodes connected to other important nodes. These methods are not designed for the DFL system which has its own distinctive features.

3 Preliminaries and Threat Model

A fixed DFL system that assume the communication links are stable and reliable can be modeled as a fully decentralized network over an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where the node set $\mathcal{N}$ denotes clients and the edge set $\mathcal{E}$ denotes connections among clients (Zhang et al., 2024; Yuan et al., 2024). For the system level, the model-state vector of all local models at learning round t can be defined as $\mathbf{\Theta}^{(t)} := \left[\boldsymbol{\theta}_1^{(t)}, \boldsymbol{\theta}_2^{(t)}, \ldots, \boldsymbol{\theta}_{|\mathcal{N}|}^{(t)}\right]^\top \in \mathbb{R}^{|\mathcal{N}|\times d}$, where $\boldsymbol{\theta}_k^{(t)} \in \mathbb{R}^d$ is the local model at client k after t learning rounds. The key notation used in this paper is summarized in Table 3 in Appendix A.

3.1 Decentralized Federated Learning

The learning task of a DFL system is to minimize the weighted average of local clients' losses,

$$\min_{\mathbf{\Theta}\in\mathbb{R}^{|\mathcal{N}|\times d}} F(\mathbf{\Theta}) = \sum_{k=1}^{|\mathcal{N}|} w_k f_k(\boldsymbol{\theta}_k, \mathcal{D}_k), \tag{1}$$

where w_k denotes the aggregation weight assigned to client k. In DFedAvg (Sun et al., 2022), w_k is proportional to the size of client k's local dataset and is defined as $w_k = |\mathcal{D}_k|/|\mathcal{D}|$, where $\mathcal{D}_k$ denotes the local dataset of client k, $\mathcal{D} = \bigcup_{k\in\mathcal{N}} \mathcal{D}_k$ denotes the global training dataset.

During training, all clients in the system synchronously execute a three-stage learning process consisting of local training $\mathbf{\Theta}^{(t+\frac{1}{2})} = \mathbf{\Theta}^{(t)} - \eta^{(t)}\,\mathbf{G}^{(t)}$ using decentralized SGD (Lian et al., 2017), communication, and aggregation $\mathbf{\Theta}^{(t+1)} = \mathbf{W}^{(t)}\mathbf{\Theta}^{(t+\frac{1}{2})}$ until convergence (Fang et al., 2024; Lalitha et al., 2018; Sun et al., 2022; Wu et al., 2023; Xie et al., 2025). Further details of this learning process is provided in Appendix B. By recursively expanding the aggregation equation by substituting local training equation, the final local models after T rounds learning can be expressed as

$$\mathbf{\Theta}^{(T)} = \left(\prod_{t=0}^{T-1} \boldsymbol{W}^{(t)}\right)\Theta^{(0)} - \sum_{t=0}^{T-1}\eta^{(t)}\left(\prod_{\tau=t}^{T-1}\boldsymbol{W}^{(\tau)}\right)\mathbf{G}^{(t)}, \tag{2}$$

where $\mathbf{W}^{(t)}$ is the mixing matrix at learning round t. $\mathbf{\Theta}^{(0)}$ is initial model-state of the DFL system. $\eta^{(t)}$ is learning rate (step size) at learning round t. $\mathbf{G}^{(t)} = \left[\boldsymbol{g}_1^{(t)}, \boldsymbol{g}_2^{(t)}, \ldots, \boldsymbol{g}_{|\mathcal{N}|}^{(t)}\right]^\top$ denote the gradients and $\boldsymbol{g}_k^{(t)} = \nabla f_k(\boldsymbol{\theta}^{(t)}; \boldsymbol{\xi}^{(t)})$ is the gradient of k-th local model at learning round t. Eq. (2) characterizes the cumulative effects of model aggregation and local gradient updates throughout the learning process and is consistent with Eq. (5) in Neglia et al. (2020).

3.2 DFL with Compromised Clients

Let $\mathcal{C} \subset \mathcal{N}$ denote the set of compromised clients, and let $\mathcal{S} = \mathcal{N} \setminus \mathcal{C}$ be the benign clients. Starting from learning round 0, each compromised client $k \in \mathcal{C}$ replace $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$ use the compromised model $\tilde{\boldsymbol{\theta}}_k^{(t+\frac{1}{2})}$. Specifically, the transmitted model of client k is written as $\tilde{\boldsymbol{\theta}}_k^{(t+\frac{1}{2})} = \boldsymbol{\theta}_k^{(t+\frac{1}{2})} + \boldsymbol{p}_k^{(t)}$ where $\boldsymbol{p}_k^{(t)}$ is the malicious perturbation of client k. For system level, $\widetilde{\mathbf{\Theta}}^{(t+\frac{1}{2})}(\mathcal{C}) = \widetilde{\mathbf{\Theta}}^{(t+\frac{1}{2})}(\mathcal{C}) + \boldsymbol{P}^{(t)}(\mathcal{C})$ where $\boldsymbol{P}^{(t)}(\mathcal{C}) = [\boldsymbol{p}_1^{(t)}, \boldsymbol{p}_2^{(t)}, \ldots, \boldsymbol{p}_{|\mathcal{N}|}^{(t)}]^\top$ denotes the malicious perturbation imposed by the compromised clients to entire system. If client i is compromised, $\boldsymbol{p}_i^{(t)} \neq \mathbf{0}$. If client j is benign, $\boldsymbol{p}_j^{(t)} \equiv \mathbf{0}$. Therefore, $\boldsymbol{P}^{(t)}(\mathcal{C}) = [\mathbf{0}, \boldsymbol{p}_{k_1}^{(t)}, \ldots, \boldsymbol{p}_{k_{|\mathcal{C}|}}^{(t)}, \ldots, \mathbf{0}]$ where $k_i \in \mathcal{C}$.

3.3 Threat Model

Adversary's Objective. The adversary seeks to identify an optimal set of compromised clients $\mathcal{C}^\star \subset \mathcal{N}$ whose distribution in the DFL communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ maximizes attack performance,

subject to a compromise budget of at most m clients, i.e., $|\mathcal{C}^\star| \leq m$ (Syros et al., 2024; Bhattacharya et al., 2024). The adversary aims to solve this problem:

$$\mathcal{C}^\star_{\mathcal{G}} = \underset{\substack{\mathcal{C}\subset\mathcal{N} \\ |\mathcal{C}|\leq m}}{\arg\max}\, \mathcal{J}\left(\mathcal{P}_{\mathcal{G}}(\boldsymbol{\Theta}^{(T)}(\emptyset)), \mathcal{P}_{\mathcal{G}}(\widetilde{\boldsymbol{\Theta}}^{(T)}(\mathcal{C}))\right), \tag{3}$$

where $\boldsymbol{\Theta}^{(T)}(\emptyset)$ and $\widetilde{\boldsymbol{\Theta}}^{(T)}(\mathcal{C})$ denote the model-state vectors of local models obtained after T rounds of training in the absence and presence of compromised clients, respectively. The function $\mathcal{P}_{\mathcal{G}}(\cdot)$ evaluates the attack performance of TDCC $\mathcal{C}$ on benign clients in the DFL system $\mathcal{G}$. Finally, $\mathcal{J}(\cdot,\cdot)$ quantifies the change in attack performance between the two settings and is defined such that larger values indicate more effective attacks.

Adversary's Capability. The adversary fully controls (Syros et al., 2024) a fixed and limited subset of clients $\mathcal{C} \subset \mathcal{N}$ throughout the learning process. Given the communication graph $\mathcal{G}$ and a budget m, the adversary selects which clients to compromise before learning starts. At each communication round t, for every compromised client $k \in \mathcal{C}$, the adversary can replace the locally updated model $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$ with a poisoned model, $\tilde{\boldsymbol{\theta}}_k^{(t+\frac{1}{2})} = \boldsymbol{\theta}_k^{(t+\frac{1}{2})} + \boldsymbol{p}_k^{(t)}$. These poisoned models are then sent to neighboring clients and participate in the standard aggregation step. The adversary may coordinate the poisoned updates among compromised clients, but it cannot modify the communication topology or the mixing matrix.

Adversary's Knowledge. We assume that the adversary has knowledge of the communication topology of the DFL system $\mathcal{G}$ which is same as the threat model's knowledge in Syros et al. (2024); Mrini et al. (2024); Gabrielli & Tolomei (2026).

4 Maximum Client-Level Accumulated Model Deviation

In this section, we first formulate how TDCCs affect attack performance in DFL systems based on mixing matrix. Building on this formulation, we propose Maximum Client-Level Accumulated Model Deviation (Max-CL-AMD), a method for distributing the compromised clients in a DFL system based on accumulated mixing matrix.

A mixing matrix $\boldsymbol{W}$ specifies the weights used for model aggregation, where $W_{i,j}$ denotes the weight that client i assigns to the model of client j. The communication topology constrains the nonzero pattern of $\boldsymbol{W}$, while the aggregation rule determines its entries. For example, BALANCE (Fang et al., 2024) selects neighboring models based on their similarity to the local model, thereby determining the effective aggregation weights. Under such model-dependent aggregation rules, such as DFedAvg(Sun et al., 2022), data heterogeneity can also indirectly affect these weights Furthermore, the ordered product of mixing matrices, $\boldsymbol{W}^{(T-1)} \cdots \boldsymbol{W}^{(0)}$, characterizes the cumulative propagation of information through successive aggregation steps over the communication graph. Given these properties, we argue that mixing matrices are well suited to modeling information propagation in DFL systems, including the propagation of malicious updates.

4.1 Problem Formulation

In a DFL system over a communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, an adversary selects a subset of clients $\mathcal{C} \subset \mathcal{N}$ to compromise, which uniquely specifies a TDCC over $\mathcal{G}$. During learning round t, each compromised client first performs local training and then, before communicating with its neighbors, replaces the trained local model with a compromised model $\tilde{\boldsymbol{\theta}}_k^{(t+\frac{1}{2})}$. Under this attack model, the system-level intermediate models be transmitted among neighboring clients are $\widetilde{\boldsymbol{\Theta}}^{(t+\frac{1}{2})}(\mathcal{C}) = \widetilde{\boldsymbol{\Theta}}^{(t)}(\mathcal{C}) - \eta^{(t)}\boldsymbol{G}^{(t)}(\mathcal{C}) + \boldsymbol{P}^{(t)}(\mathcal{C})$. Then, at mixing round t, the clients aggregate their local models using the mixing matrix $\boldsymbol{W}^{(t)}(\mathcal{C})$: $\widetilde{\boldsymbol{\Theta}}^{(t+1)}(\mathcal{C}) = \mathbf{W}^{(t)}(\mathcal{C})\widetilde{\boldsymbol{\Theta}}^{(t)} - \eta^{(t)}\mathbf{W}^{(t)}(\mathcal{C})\mathbf{G}^{(t)} + \mathbf{W}^{(t)}(\mathcal{C})\boldsymbol{P}^{(t)}(\mathcal{C})$. And recursively expanding this aggregation equation yields

$$\widetilde{\boldsymbol{\Theta}}^{(T)}(\mathcal{C}) = \underbrace{\boldsymbol{\Phi}(\mathcal{C},T,0)\boldsymbol{\Theta}^{(0)} - \sum_{t=0}^{T-1}\eta^{(t)}\boldsymbol{\Phi}(\mathcal{C},T,t)\mathbf{G}^{(t)}}_{\text{first term}} + \underbrace{\sum_{t=0}^{T-1}\boldsymbol{\Phi}(\mathcal{C},T,t)\boldsymbol{P}^{(t)}(\mathcal{C})}_{\text{second term}}, \tag{4}$$

where $\mathbf{\Phi}(\mathcal{C},T,t)=\prod_{\tau=t}^{T-1}\mathbf{W}^{(\tau)}(\mathcal{C})=\mathbf{W}^{(T-1)}(\mathcal{C})\mathbf{W}^{(T-2)}(\mathcal{C})\cdots\mathbf{W}^{(t)}(\mathcal{C})$. To characterize the mixing trajectories of information from learning round t to T, we define the accumulated mixing matrix $\mathbf{\Phi}(\mathcal{C},T,t)$, as detailed in Appendix C. The first term is similar to Eq. (2) and represents the final local models in the absence of attackers in the system. The second term captures the accumulated model deviation throughout the entire learning process.

Prior work has modeled the impact of attacks as model deviations and used these deviations to define attack objectives (Fang et al., 2020; Shejwalkar & Houmansadr, 2021; Xie et al., 2025; Zhang et al., 2023). Specifically, we compare the final model obtained in the presence of compromised clients with the corresponding clean model, both trained under identical stochastic conditions (Panda et al., 2022; Sun et al., 2021). We use the ℓ_2 norm to quantify the overall magnitude of the parameter deviation (Xie et al., 2021). Thus, by setting $\mathcal{P}_{\mathcal{G}}(\mathbf{\Theta}^{(T)}(\mathcal{C}))=\mathbf{\Theta}^{(T)}(\mathcal{C})$ and $\mathcal{J}\left(\mathcal{P}_{\mathcal{G}}(\mathbf{\Theta}^{(T)}(\emptyset)),\mathcal{P}_{\mathcal{G}}(\widetilde{\mathbf{\Theta}}^{(\boldsymbol{T})}(\mathcal{C}))\right)=||\widetilde{\mathbf{\Theta}}^{(\boldsymbol{T})}(\mathcal{C})-\mathbf{\Theta}^{(T)}(\emptyset)||_{1,2}$, the optimization problem in Eq. (3) reduces to

$$
\begin{aligned}
\mathcal{C}_{\mathcal{G}}^{\star}=&\arg\max_{\substack{\mathcal{C}\subset\mathcal{N}\\|\mathcal{C}|\leq m}}\Bigg\|\left[\mathbf{\Phi}(\mathcal{C},T,0)-\mathbf{\Phi}(\emptyset,T,0)\right]\Theta^{(0)}\\
&-\sum_{s=0}^{T-1}\eta^{(s)}\left[\mathbf{\Phi}(\mathcal{C},T,s)\widetilde{G}^{(s)}(\mathcal{C})-\mathbf{\Phi}(\emptyset,T,s)G^{(s)}(\emptyset)\right]+\sum_{s=0}^{T-1}\mathbf{\Phi}(\mathcal{C},T,s)\boldsymbol{P}^{(s)}(\mathcal{C})\Bigg\|_{1,2}\\
=&\arg\max_{\substack{\mathcal{C}\subset\mathcal{N}\\|\mathcal{C}|\leq m}}\Bigg\|\boldsymbol{D}(\mathcal{C},T,0)\mathbf{\Theta}^{(0)}-\sum_{s=0}^{T-1}\eta^{(s)}\boldsymbol{D}(\mathcal{C},T,s)\mathbf{G}^{(s)}(\emptyset)\\
&+\sum_{s=0}^{T-1}\Phi(\mathcal{C},T,s)\left[\mathbf{P}^{(s)}(\mathcal{C})-\eta^{(s)}\left(\mathbf{G}^{(s)}(\mathcal{C})-\mathbf{G}^{(s)}(\emptyset)\right)\right]\Bigg\|_{1,2},
\end{aligned}
\tag{5}
$$

where $\boldsymbol{D}(\mathcal{C},T,t)=\Phi(\mathcal{C},T,t)-\Phi(\emptyset,T,t)$. Eq. (5) shows that the system-level deviation is determined by the perturbations introduced over all communication rounds and their subsequent propagation through the time-varying mixing matrices $\mathbf{W}^{(\tau)}(\mathcal{C})$. Consequently, attack performance depends not only on the magnitudes of the individual client deviations, but also on their directions, the rounds in which they are introduced, and the topological positions of the compromised clients.

4.2 Maximum Client-Level Accumulated Model Deviation (Max-CL-AMD)

Building on the problem formulation in Section 4.1, we introduce our proposed method, *Maximum Client-Level Accumulated Model Deviation (Max-CL-AMD)*. We first state several assumptions that underpin the development of Max-CL-AMD.

Assumption 1 (Topology Compliance (Zhang et al., 2025b; Shi et al., 2023; Lian et al., 2017)). *At each mixing round t, the mixing matrix $\boldsymbol{W}^{(t)}=[w_{ij}^{(t)}]\in[0,1)^{|\mathcal{N}|\times|\mathcal{N}|}$ is topology-compliant.*

Assumption 2 (Row Stochasticity (Zhu et al., 2025; Liang et al., 2025; Vogels et al., 2021)). *The mixing matrix $\boldsymbol{W}^{(t)}$ is row-stochastic, i.e., $\boldsymbol{W}^{(t)}\mathbf{1}=\mathbf{1}$.*

Assumption 3 (Bounded Malicious Perturbation (Zhang et al., 2023)). *The malicious perturbation introduced by any malicious client at any learning round is bounded in the ℓ_2 norm by a constant $B_p>0$, i.e., $\left\|\boldsymbol{p}_k^{(t)}\right\|_2\leq B_p$.*

Assumption 4 (Bounded Gradient (Chen et al., 2022; Reddi et al., 2020; Deng et al., 2020)). *The gradient is bounded by a constant $G_{max}>0$, i.e. $\left\|\boldsymbol{g}_k^{(t)}\right\|_2\leq G_{max}$.*

Assumption 5 (Bounded Gradient Dissimilarity (Yue & Ren, 2024; Data & Diggavi, 2021; Liu et al., 2023)). *The dissimilarity of any two clients' gradients are bounded by a constant $\sigma>0$ i.e. $\left\|\boldsymbol{g}_k^{(t)}-\boldsymbol{g}_h^{(t)}\right\|_2\leq\sigma$.*

The objective function in Eq. (5) quantifies the system-level model deviation accumulated across all clients and learning rounds. Specifically, this quantity is defined as follows:

Definition 1 (System-Level Accumulated Model Deviation). *Over T learning rounds, the system-level accumulated model deviation over a DFL system $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with $\mathcal{C}$ is defined as*

$$\begin{aligned}\boldsymbol{S}(\mathcal{C}, T) :=& \boldsymbol{D}(\mathcal{C}, T, 0)\boldsymbol{\Theta}^{(0)} - \sum_{s=0}^{T-1} \eta^{(s)} \boldsymbol{D}(\mathcal{C}, T, s)\mathbf{G}^{(s)}(\emptyset) \\ &+ \sum_{s=0}^{T-1} \Phi(\mathcal{C}, T, s) \left[\mathbf{P}^{(s)}(\mathcal{C}) - \eta^{(s)} \left(\mathbf{G}^{(s)}(\mathcal{C}) - \mathbf{G}^{(s)}(\emptyset)\right)\right].\end{aligned} \tag{6}$$

Since the decision variable $\mathcal{C}$ is a subset of the client set $\mathcal{N}$ satisfying the cardinality constraint $|\mathcal{C}| \leq m$, while T is typically treated as a hyperparameter, the optimization problem in Eq. (5) is a cardinality-constrained combinatorial optimization problem. Specifically, it aims to identify a set of at most m compromised clients that maximizes the magnitude of the accumulated system-level model deviation. For a given $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, the optimal $\mathcal{C}^\star$ can, in principle, be found by enumerating all feasible subsets, with a time complexity of $\mathcal{O}(|\mathcal{N}|^m \tau)$, where τ denotes the time required to evaluate a candidate subset. This complexity makes exhaustive enumeration computationally impractical for large-scale DFL systems. We therefore evaluate the individual contribution of each candidate client to the system-level model deviation.

Decomposing System-level to Client-level Deviation. Since enumerating all feasible subset $\mathcal{C}$ is computationally intractable, we instead quantify each candidate client's potential contribution to the system-level model deviation. Under Assumption 3, 4, 5, we decompose and bound the system-level model deviation to candidate clients, yielding the following additive client-level upper bound:

$$\left\|\boldsymbol{S}(\mathcal{C}, T)\right\|_{1,2} \leq \sum_{k=1}^{|\mathcal{N}|} \sum_{t=0}^{T-1} \left\|[\boldsymbol{\Phi}(\mathcal{C}, T, t)]_{:,k}\right\|_1 (B_p + |\eta^{(t)}|\sigma) + 2|\mathcal{N}|G_{\max}B_\eta, \tag{7}$$

where $B_\eta = \sum_{t=0}^{T} \eta^{(t)}$. The detailed derivation is provided in Appendix D. This decomposition shows that the contribution of client k at round t is governed by the k-column of accumulated mixing matrix $\left\|[\boldsymbol{\Phi}(\mathcal{C}, T, t)]_{:,k}\right\|_1$. This decomposition motivates the following client-level measure.

Definition 2 (Client-Level Accumulated Model Deviation). *For each client in a DFL system $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with $\mathcal{C}$, we define its client-level accumulated model deviation at round T by:*

$$\omega_k(\mathcal{C}, T) := \sum_{t=0}^{T-1} \left\|[\boldsymbol{\Phi}(\mathcal{C}, T, t)]_{:,k}\right\|_1 (B_p + |\eta^{(t)}|\sigma) + 2G_{max}B_\eta. \tag{8}$$

For each round t, the k-th column of $\boldsymbol{\Phi}(\mathcal{C}, T, t)$ characterizes how a malicious perturbation from client k propagates to all clients by round T. Therefore, $B_p\|[\boldsymbol{\Phi}(\mathcal{C}, T, t)]_{:,k}\|_1$ represents the system-level effect of client k's perturbation introduced from round t to round T, and $\omega_k(\mathcal{C}, T)$ accumulates these effects over rounds $0, \ldots, T-1$.

Theorem 1 (Maximum Client-Level Accumulated Model Deviation). *Consider a DFL system $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, the adversary can compromise at most m clients. Under the Definition 2 and Assumption 1, 3, 4, 5, then define $\pi : \{1, \ldots, |\mathcal{N}|\}$ as a permutation satisfying: $\omega_{\pi_1}(\mathcal{C}, T) \geq \omega_{\pi_2}(\mathcal{C}, T) \geq \cdots \geq \omega_{\pi_{|\mathcal{N}|}}(\mathcal{C}, T)$. The optimal compromised-client distribution is*

$$\mathcal{C}^\star_{\mathcal{G}} = \{\pi_i \mid i = 1, \ldots, m\}. \tag{9}$$

The theoretical proof of this theorem is provided in Appendix E. The experimental proof of this theorem is provided in Appendix F.4.3.

However, computing $\omega_k(\mathcal{C}, T)$ is infeasible because the ground-truth mixing matrix $W_k^{(t)}$ is unknown before $\mathcal{C}$ is determined. Moreover, for certain nonlinear aggregation rules, such as the coordinate-wise median and trimmed mean (Yin et al., 2018), $W_k^{(t)}$ is only a virtual matrix for theoretical analysis (Wu et al., 2023). In this situation, mixing matrix is not explicitly constructed during aggregation and is therefore not directly available, even after training is complete.

Mixing Matrix Estimation. Therefore, assuming that the adversary only knows the communication topology $\mathcal{G}$ of the DFL system, we construct an estimated mixing matrix using only this topology:

$$\widehat{\boldsymbol{W}} = (\boldsymbol{M} + \boldsymbol{I})^{-1}(\boldsymbol{A} + \boldsymbol{I}) \tag{10}$$

where $\boldsymbol{M} = \mathrm{diag}(d_1, \ldots, d_n)$ denotes the degree matrix of the communication graph $\mathcal{G}$, with $d_i = \sum_{j=1}^{n}(\boldsymbol{M})_{ij}$. $\boldsymbol{I}$ is identity matrix has same shape with $\boldsymbol{M}$. $\boldsymbol{A}$ is the adjacency matrix of $\mathcal{G}$. This estimated mixing matrix is motivated by Metropolis–Hastings weights and maximum-degree weights (Xiao et al., 2007). Based on estimated mixing matrix, we can calculate an estimated client-level accumulated model deviation as

$$\widehat{\omega}_k(T) := \sum_{t=0}^{T-1} \left\| \left[\widehat{\boldsymbol{\Phi}}(T,t) \right]_{:,k} \right\|_1 . \tag{11}$$

We compute $\widehat{\omega}_k(T)$ and $\widehat{\boldsymbol{\Phi}}(T,t)$ using the estimated mixing matrices $\widehat{\boldsymbol{W}}$. Based on Theorem 1 and these estimates, we develop Algorithm 1 for DFL with an effective TDCC method.

Algorithm 1 DFL under Max-CL-AMD

Input: A communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, total learning rounds T, attacker budget m, perturbation $\boldsymbol{p}_k^{(t)}$; DFL setup, such as η_k.

Phase 1: Compromised-client selection
Estimate $\widehat{\boldsymbol{W}}$ using Eq. (10)
Compute $\widehat{\omega}_k(T)$ for all $k \in \mathcal{N}$
Sort clients in descending order:
$\widehat{\omega}_{\pi_1}(T) \geq \cdots \geq \widehat{\omega}_{\pi_{|\mathcal{N}|}}(T)$
Estimated $\widehat{\mathcal{C}}_{\mathcal{G}}^{\star} \leftarrow \{\pi(i) \mid i = 1, \ldots, m\}$
Phase 2: DFL training under attack
Initialize $\theta_k^{(0)}$ for all $k \in \mathcal{N}$
for $t = 0, 1, \ldots, T-1$ **do**
  for $k \in \mathcal{N}$ **in parallel do**
    Sample a mini-batch $\xi_k^{(t)}$ from $\mathcal{D}_k$
    $\boldsymbol{g} \leftarrow \nabla f_k(\theta_k^{(t)}; \xi_k^{(t)})$
    $\boldsymbol{\theta}_k^{(t+\frac{1}{2})} \leftarrow \boldsymbol{\theta}_k^{(t)} - \eta_k^{(t)} \nabla f_k(\boldsymbol{\theta}_k^{(t)}; \boldsymbol{\xi}^{(t)})$
    if $k \in \widehat{\mathcal{C}}_{\mathcal{G}}^{\star}$ **then**
      send $\tilde{\boldsymbol{\theta}}_k^{(t+\frac{1}{2})} \leftarrow \boldsymbol{\theta}_k^{(t+\frac{1}{2})} + \boldsymbol{p}_k^{(t)}$
    else
      send $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$
  for $k = 1, 2, \ldots, |\mathcal{N}|$ **in parallel do**
    $\boldsymbol{\theta}_k^{(t+1)} \leftarrow \sum_{i \in \mathcal{N}_k} w_{k,i}^{(t)} \boldsymbol{\theta}_i^{(t+\frac{1}{2})}$

5 Experiments

We compare Max-CL-AMD with five representative baselines: Random, Max-Degree (Syros et al., 2024), Max-PageRank (Syros et al., 2024), SWAP-BPI (Gabrielli & Tolomei, 2026), HA-Max-Span (Piaseczny et al., 2024) in a 20-client DFL system with $m = 4$ compromised clients under Gaussian attack $\boldsymbol{p} \sim \mathcal{N}(\boldsymbol{0}, 10^2\boldsymbol{I})$. We evaluate all methods across seven communication topologies: Erdős–Rényi (ER) graphs (Syros et al., 2024), Watts-Strogatz (WS) graphs (Watts & Strogatz, 1998), Barabasi-Albert (BA) graphs (Albert & Barabási, 2002), ring graphs (Wang et al., 2022), line graphs (Mokhtari & Ribeiro, 2016), complete graphs, and star graphs. Our main experiments use softmax regression on MNIST to examine how TDCC affects attack performance. Additional experiments including Fashion-MNIST, CIFAR-10, and HAR are used to assess the performance of Max-CL-AMD across learning tasks. All reported average Test Error Rate (TER, detailed definition is provided in Appendix E) over 5 runs with different random seeds. Experimental settings and implementation details are provided in the Appendix F.

5.1 Effect of nonlinear aggregation

We first evaluate our method on settings with 6 different robust aggregation rules under 3 different communication graphs, ER, WS and BA. These aggregation rules arises nonlinearities because these aggregators either adaptively select updates (e.g., Krum, Multi-Krum Blanchard et al. (2017) and BALANCE) or perform nonlinear coordinate-wise aggregation (e.g., Median (Yin et al., 2018)).

As shown in Table 1, Random selection yields the lowest TER in almost all settings. This result suggests that a carefully designed TDCC is crucial for achieving high attack performance in DFL systems. Our method achieves competitive performance across a wide range of settings. Compared with Random, it yields a relative increase in TER of 0.22% – 308.53% in all but two settings. It also achieves the highest or second-highest TER in several settings, as highlighted in the table. In certain settings, our method substantially increases TER against some robust aggregation rules, demonstrating its ability to undermine their robustness. In some scenarios, different methods select the same TDCC in each of the five runs and yield identical TER values and variances. Additional comparisons with Maximum Effective Network Size, Maximum Clustering Coefficient, and Max-Span are presented in Appendix F.4.1.

Table 1: *TER± standard deviation comparison under nonlinear aggregation rules*. For Max-CL-AMD (Ours), percentages in square brackets indicate improvement relative to Random Selection. The best and second-best TER in each setting are shown in bold dark blue and bold muted blue, respectively.

Network	Aggregation	Random	Max-Degree	Max-PageRank	HA-Max-Span	SWAP-BPI	Max-CL-AMD (Ours)↑
ER	Median	10.01±0.06	**33.62±33.53**	**33.62±33.53**	15.63±10.01	17.46±8.34	**33.62±33.53**[**235.93%**]
	Krum	9.99±0.09	9.95±0.11	9.93±0.12	10.00±0.10	**10.16±0.12**	9.93±0.12[−0.58%]
	Multi-Krum	18.52±6.15	62.28±26.88	65.34±29.05	**67.00±32.33**	22.10±2.97	65.34±29.05[**252.74%**]
	BALANCE	9.90±0.13	9.92±0.10	9.92±0.10	**9.96±0.12**	9.92±0.16	9.92±0.10[0.22%]
	FLTrust	10.41±0.09	10.67±0.14	10.67±0.14	**10.68±0.10**	10.53±0.12	10.67±0.14[**2.53%**]
	Clipped Gossip	34.21±1.62	**51.89±5.73**	**51.89±5.73**	48.79±5.56	47.40±3.64	**51.89±5.73**[**51.67%**]
WS	Median	11.02±2.24	13.89±6.20	11.01±2.17	**16.33±5.75**	9.96±0.09	11.95±2.60[8.53%]
	Krum	10.03±0.14	10.04±0.11	10.01±0.15	10.04±0.08	9.96±0.07	**10.06±0.13**[**0.28%**]
	Multi-Krum	37.34±19.69	73.35±16.29	**81.33±1.47**	66.90±15.76	9.93±0.13	80.78±1.80[**116.37%**]
	BALANCE	9.95±0.13	9.93±0.14	**9.96±0.15**	9.92±0.13	9.93±0.11	9.88±0.10[−0.70%]
	FLTrust	10.39±0.10	10.46±0.09	10.46±0.13	10.45±0.10	10.41±0.07	**10.48±0.11**[**0.92%**]
	Clipped Gossip	38.55±1.78	39.22±10.84	**44.33±7.44**	39.80±9.07	38.79±3.08	39.71±11.54[3.01%]
BA	Median	10.94±2.06	43.71±36.68	**44.70±35.76**	42.60±39.85	28.68±28.97	**44.70±35.76**[**308.49%**]
	Krum	9.97±0.11	10.92±2.15	**10.96±2.13**	10.01±0.08	10.14±0.09	**10.96±2.13**[**9.98%**]
	Multi-Krum	14.86±3.60	**58.98±21.66**	55.92±22.66	48.86±25.70	14.98±0.16	55.92±22.66[**276.22%**]
	BALANCE	9.97±0.17	9.98±0.08	10.00±0.08	**10.00±0.13**	10.00±0.12	10.00±0.08[0.25%]
	FLTrust	10.43±0.14	**10.83±0.08**	10.83±0.08	10.75±0.11	10.72±0.08	10.83±0.08[**3.84%**]
	Clipped Gossip	38.51±8.66	57.84±3.18	**57.90±3.18**	56.56±2.85	54.47±3.53	**57.90±3.18**[**50.34%**]

5.2 Effect of Network Topology

We then evaluate our method on seven representative communication topologies, which are described in detail in Appendix F.1. The experiments reported here use Median as the aggregation rule; results for other aggregation rules are presented in Appendix F.4.3.

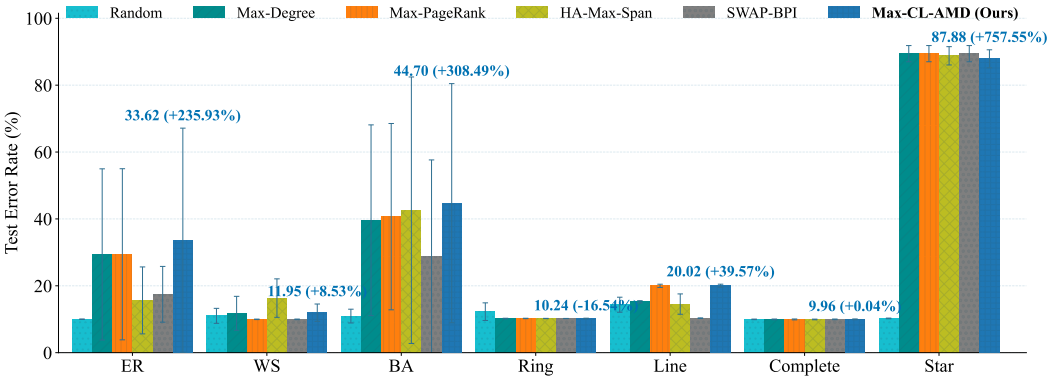


Figure 1: *Test Error Rate (TER%) comparison under different network topologies using Median.*

As shown in Figure 1, our method performs competitively across all communication graphs. Relative to Random Selection, our method yields remarkable improvements, from 235.93% to 757.55% on ER, BA and Star graphs. On WS and Line graphs, our method also have varying degrees improvement. On Ring and Complete graphs, all methods achieve nearly identical performance, which may reflect the structural symmetry of these graphs.

5.3 Effect of the Number of Compromised Clients

We further investigate the impact of the number of compromised clients by setting $m \in \{1, 2, 3, 4\}$, corresponding to 5%, 10%, 15%, and 20% of a 20-client DFL system, respectively.

As shown in Figure 2, our method consistently achieves the best performance across all test scenarios with varying proportions of compromised clients. We also observe that the attack performance improves as the proportion of compromised clients increases, which is a reasonable phenomenon and

Table 2: *TER*$\pm$*standard deviation comparison under different tasks and different data heterogeneity*. The best and second-best TER in each setting are shown in bold dark blue and bold muted blue, respectively. SR and CNN denote softmax regression and convolutional neural networks, respectively.

Task	Data Heterogeneity	Random	Max-Degree	Max-PageRank	HA-MaxSpAN-FL	SWAP-BPI	Ours↑
MNIST (SR)	IID	13.68$_{\pm 8.32}$	**37.30**$_{\pm 27.18}$	**37.30**$_{\pm 27.18}$	17.69$_{\pm 4.23}$	17.46$_{\pm 8.34}$	**37.30**$_{\pm 27.18}$
	Diri($\alpha = 1.0$)	14.31$_{\pm 8.33}$	**38.29**$_{\pm 27.27}$	**38.29**$_{\pm 27.27}$	20.24$_{\pm 8.90}$	18.09$_{\pm 8.33}$	**38.29**$_{\pm 27.27}$
	Diri($\alpha = 0.1$)	34.31$_{\pm 7.79}$	**44.42**$_{\pm 24.80}$	**44.42**$_{\pm 24.80}$	27.14$_{\pm 6.69}$	34.83$_{\pm 5.65}$	**44.42**$_{\pm 24.80}$
	LabelSkew(4)	19.30$_{\pm 9.49}$	**39.79**$_{\pm 26.21}$	**39.79**$_{\pm 26.21}$	19.25$_{\pm 7.49}$	19.11$_{\pm 7.75}$	**39.79**$_{\pm 26.21}$
FMNIST (CNN)	IID	21.71$_{\pm 6.32}$	**41.22**$_{\pm 21.13}$	**41.22**$_{\pm 21.13}$	34.26$_{\pm 24.46}$	25.60$_{\pm 7.74}$	**41.22**$_{\pm 21.13}$
	Diri($\alpha = 1.0$)	22.40$_{\pm 6.22}$	**42.42**$_{\pm 21.05}$	**42.42**$_{\pm 21.05}$	38.07$_{\pm 23.07}$	26.27$_{\pm 7.45}$	**42.42**$_{\pm 21.05}$
	Diri($\alpha = 0.1$)	38.43$_{\pm 6.16}$	**48.01**$_{\pm 18.55}$	**48.01**$_{\pm 18.55}$	35.67$_{\pm 4.36}$	41.54$_{\pm 4.43}$	**48.01**$_{\pm 18.55}$
	LabelSkew(4)	27.11$_{\pm 8.93}$	**44.72**$_{\pm 19.25}$	**44.72**$_{\pm 19.25}$	30.62$_{\pm 7.14}$	27.85$_{\pm 7.26}$	**44.72**$_{\pm 19.25}$
CIFAR-10 (CNN)	IID	**72.01**$_{\pm 16.96}$	69.75$_{\pm 12.25}$	69.75$_{\pm 12.25}$	66.35$_{\pm 13.69}$	65.74$_{\pm 14.11}$	69.75$_{\pm 12.25}$
	Diri($\alpha = 1.0$)	74.36$_{\pm 15.07}$	73.77$_{\pm 10.23}$	73.77$_{\pm 10.23}$	65.00$_{\pm 5.06}$	68.38$_{\pm 12.63}$	**76.06**$_{\pm 10.22}$
	Diri($\alpha = 0.1$)	81.42$_{\pm 9.31}$	**82.01**$_{\pm 5.60}$	**82.01**$_{\pm 5.60}$	77.82$_{\pm 7.29}$	78.09$_{\pm 6.86}$	**82.01**$_{\pm 5.60}$
	LabelSkew(4)	77.15$_{\pm 13.04}$	76.58$_{\pm 8.39}$	76.58$_{\pm 8.39}$	**77.52**$_{\pm 11.94}$	71.00$_{\pm 11.24}$	76.58$_{\pm 8.39}$
HAR (SR)	IID	18.57$_{\pm 7.94}$	**39.66**$_{\pm 24.37}$	**39.66**$_{\pm 24.37}$	25.37$_{\pm 8.82}$	21.53$_{\pm 6.94}$	**39.66**$_{\pm 24.37}$
	Diri($\alpha = 1.0$)	22.07$_{\pm 9.70}$	**45.13**$_{\pm 22.65}$	**45.13**$_{\pm 22.65}$	36.86$_{\pm 4.23}$	23.29$_{\pm 7.09}$	**45.13**$_{\pm 22.65}$
	Diri($\alpha = 0.1$)	49.88$_{\pm 10.25}$	**58.62**$_{\pm 16.17}$	56.93$_{\pm 18.11}$	56.11$_{\pm 14.37}$	49.61$_{\pm 7.25}$	56.93$_{\pm 18.11}$
	LabelSkew(4)	21.25$_{\pm 8.32}$	**43.67**$_{\pm 21.77}$	**43.67**$_{\pm 21.77}$	29.66$_{\pm 4.17}$	26.72$_{\pm 11.61}$	**43.67**$_{\pm 21.77}$

align with Theorem 1. Moreover, increasing the number of compromised clients by just one, from $m = 3$ to $m = 4$, dramatically improves attack performance, raising the TER from approximately 15% to more than 35%.

5.4 Effect of different tasks and data heterogeneity

To evaluate whether our method generalizes beyond MNIST, we conduct experiments on 4 benchmark datasets: MNIST, Fashion-MNIST (FMNIST) (Xiao et al., 2017), CIFAR-10 (Krizhevsky et al., 2009), and HAR (Saha et al., 2025). Moreover, data heterogeneity is a fundamental characteristic in DFL systems, as clients often hold local data that are not independent and identically distributed (non-IID). Following Zhang et al. (2025a), we therefore evaluate whether our method remains effective under both IID and non-IID data distributions.

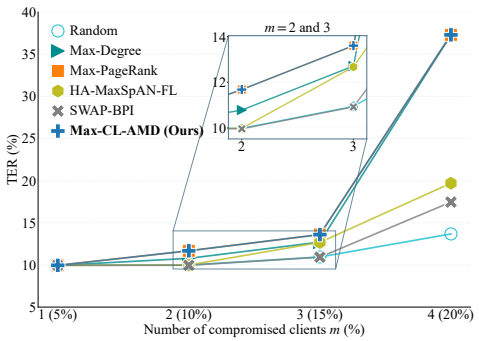


Figure 2: *Test Error Rate (TER%) comparison of different number of compromised clients using Median.*

As shown in Table 2, Max-CL-AMD consistently achieves high TER across datasets and data-distribution settings. Two observations help explain these results: (1) In several settings, Max-CL-AMD, Max-Degree, and Max-PageRank achieve same TER because they select the same TDCC $\mathcal{C}$ in a small-scale DFL system. In large-scale DFL systems with 1,000 clients, the selected TDCCs are no longer same (Appendix F.4.4). (2) In some settings, the results have large standard deviations because the TDCC $\mathcal{C}$ generated with certain random seeds causes the entire learning process to collapse. These results further indicate that Max-CL-AMD remains effective across diverse settings.

6 Conclusion

In this paper, we investigated how the topological distribution of compromised clients (TDCC) affects attack performance in DFL systems by modeling the system-level and client-level accumulated model deviation using mixing matrices. Based on this model, we proposed Maximum Client-Level Accumulated Model Deviation (Max-CL-AMD) for determining the TDCC in a DFL system using estimated matrices. Extensive experiments across diverse settings and comparisons with several baseline methods demonstrated the effectiveness of Max-CL-AMD. More importantly, our findings show that TDCC is a key factor in DFL systems.

7 AI USE STATEMENT

In this work, we used generative AI tools for polishing writing. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Siddhartha Bhattacharya, Daniel Helo, and Joshua Siegel. Impact of network topology on byzantine resilience in decentralized federated learning. *arXiv preprint arXiv:2407.05141*, 2024.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017.

Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46(4):667–689, 2004. doi: 10.1137/S0036144503423264.

Ronald S Burt. Structural holes. In *Social stratification*, pp. 659–663. Routledge, 2018.

Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995*, 2020.

Hang Chen, Syed Ali Asif, Jihong Park, Chien-Chung Shen, and Mehdi Bennis. Robust blockchained federated learning with model validation and proof-of-stake inspired consensus. *arXiv preprint arXiv:2101.03300*, 2021.

Suo Chen, Yang Xu, Hongli Xu, Zhida Jiang, and Chunming Qiao. Decentralized federated learning with intermediate results in mobile edge computing. *IEEE Transactions on Mobile Computing*, 23(1):341–358, 2022.

Cho-Chun Chiu, Xusheng Zhang, Ting He, Shiqiang Wang, and Ananthram Swami. Laplacian matrix sampling for communication-efficient decentralized learning. *IEEE Journal on Selected Areas in Communications*, 41(4):887–901, 2023.

Yatin Dandi, Anastasia Koloskova, Martin Jaggi, and Sebastian U Stich. Data-heterogeneity-aware mixing for decentralized learning. *arXiv preprint arXiv:2204.06477*, 2022.

Deepesh Data and Suhas Diggavi. Byzantine-resilient high-dimensional sgd with local iterations on heterogeneous data. In *International Conference on Machine Learning*, pp. 2478–2488. PMLR, 2021.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Distributionally robust federated averaging. *Advances in neural information processing systems*, 33:15111–15122, 2020.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to byzantine-robust federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1605–1622, 2020.

Minghong Fang, Zifan Zhang, Hairi, Prashant Khanduri, Jia Liu, Songtao Lu, Yuchen Liu, and Neil Gong. Byzantine-robust decentralized federated learning. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pp. 2874–2888, 2024.

Edoardo Gabrielli and Gabriele Tolomei. Optimizing byzantine node placement in decentralized federated learning. *arXiv preprint arXiv:2609.01495*, 2026.

Chaoyang He, Conghui Tan, Hanlin Tang, Shuang Qiu, and Ji Liu. Central server free federated learning over single-sided trust social networks. *arXiv preprint arXiv:1910.04956*, 2019.

Chaoyang He, Emir Ceyani, Keshav Balasubramanian, Murali Annavaram, and Salman Avestimehr. Spreadgnn: Serverless multi-task federated learning for graph neural networks. *arXiv preprint arXiv:2106.02743*, 2021.

Yuqi Jia, Minghong Fang, and Neil Zhenqiang Gong. Competitive advantage attacks to decentralized federated learning. *arXiv preprint arXiv:2310.13862*, 2023.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images.(2009), 2009.

Anusha Lalitha, Shubhanshu Shekhar, Tara Javidi, and Farinaz Koushanfar. Fully decentralized federated learning. In *Third Workshop on Bayesian Deep Learning (NeurIPS)*, volume 2, 2018.

Khiem Le, Nhan Luong-Ha, Manh Nguyen-Duc, Danh Le-Phuoc, Cuong Do, and Kok-Seng Wong. Exploring the practicality of federated learning: A survey towards the communication perspective. *arXiv preprint arXiv:2405.20431*, 2024.

Batiste Le Bars, Aurélien Bellet, Marc Tommasi, Erick Lavoie, and Anne-Marie Kermarrec. Refined convergence and topology learning for decentralized sgd with heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pp. 1672–1702. PMLR, 2023.

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*, volume 30, pp. 5330–5340, 2017.

Liyuan Liang, Xinyi Chen, Gan Luo, and Kun Yuan. Achieving linear speedup and near-optimal complexity for decentralized optimization over row-stochastic networks. *arXiv preprint arXiv:2506.04600*, 2025.

Yuchen Liu, Chen Chen, Lingjuan Lyu, Fangzhao Wu, Sai Wu, and Gang Chen. Byzantine-robust learning on heterogeneous data via gradient splitting. In *International Conference on Machine Learning*, pp. 21404–21425. PMLR, 2023.

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

Aryan Mokhtari and Alejandro Ribeiro. Dsa: Decentralized double stochastic averaging gradient algorithm. *Journal of Machine Learning Research*, 17(61):1–35, 2016.

Abdellah El Mrini, Edwige Cyffers, and Aurélien Bellet. Privacy attacks in decentralized learning. *arXiv preprint arXiv:2402.10001*, 2024.

Giovanni Neglia, Chuan Xu, Don Towsley, and Gianmarco Calbi. Decentralized gradient methods: does topology matter? In *International Conference on Artificial Intelligence and Statistics*, pp. 2348–2358. PMLR, 2020.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford infolab, 1999.

Ashwinee Panda, Saeed Mahloujifar, Arjun Nitin Bhagoji, Supriyo Chakraborty, and Prateek Mittal. Sparsefed: Mitigating model poisoning attacks in federated learning with sparsification. In *International Conference on Artificial Intelligence and Statistics*, pp. 7587–7624. PMLR, 2022.

Dario Pasquini, Mathilde Raynal, and Carmela Troncoso. On the (in) security of peer-to-peer decentralized machine learning. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 418–436. IEEE, 2023.

Adam Piaseczny, Eric Ruzomberka, Rohit Parasnis, and Christopher G Brinton. Adversarial node placement in decentralized federated learning: Maximum spanning-centrality strategy and performance analysis. *IEEE Internet of Things Journal*, 12(1):45–60, 2024.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Debashish Saha, Piyush Malik, and Adrika Saha. Activity and subject detection for uci har dataset with & without missing sensor data. *arXiv preprint arXiv:2505.06730*, 2025.

Radwan Selo, Majid Kundroo, and Taehong Kim. Impact of network topologies on decentralized federated learning. *Cluster Computing*, 28(15):996, 2025.

Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Ndss*, 2021.

Yifan Shi, Li Shen, Kang Wei, Yan Sun, Bo Yuan, Xueqian Wang, and Dacheng Tao. Improving the model consistency of decentralized federated learning. In *International conference on machine learning*, pp. 31269–31291. PMLR, 2023.

Jingwei Sun, Ang Li, Louis DiValentin, Amin Hassanzadeh, Yiran Chen, and Hai Li. Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective. *Advances in neural information processing systems*, 34:12613–12624, 2021.

Tao Sun, Dongsheng Li, and Bao Wang. Decentralized federated averaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4289–4301, 2022.

Georgios Syros, Gokberk Yar, Simona Boboila, Cristina Nita-Rotaru, and Alina Oprea. Backdoor attacks in peer-to-peer federated learning. *ACM Transactions on Privacy and Security*, 28(1): 1–28, 2024.

Thijs Vogels, Lie He, Anastasiia Koloskova, Sai Praneeth Karimireddy, Tao Lin, Sebastian U Stich, and Martin Jaggi. Relaysum for decentralized deep learning on heterogeneous data. *Advances in Neural Information Processing Systems*, 34:28004–28015, 2021.

Thijs Vogels, Hadrien Hendrikx, and Martin Jaggi. Beyond spectral gap: The role of the topology in decentralized learning. *Journal of Machine Learning Research*, 24(355):1–31, 2023.

Zhao Wang, Yifan Hu, Shiyang Yan, Zhihao Wang, Ruijie Hou, and Chao Wu. Efficient ring-topology decentralized federated learning with deep generative models for medical data in ehealthcare systems. *Electronics*, 11(10):1548, 2022.

Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393 (6684):440–442, 1998.

Di Wu, Qi Guo, Yong Qi, Saiyu Qi, and Qian Li. Ama: Adaptive model poisoning attacks towards federated learning. *IEEE Transactions on Dependable and Secure Computing*, 2025.

Zhaoxian Wu, Tianyi Chen, and Qing Ling. Byzantine-resilient decentralized stochastic optimization with robust aggregation rules. *IEEE transactions on signal processing*, 71:3179–3195, 2023.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of parallel and distributed computing*, 67(1):33–46, 2007.

Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. Crfl: Certifiably robust federated learning against backdoor attacks. In *International conference on machine learning*, pp. 11372–11382. PMLR, 2021.

Yueqi Xie, Minghong Fang, and Neil Zhenqiang Gong. Model poisoning attacks to federated learning via multi-round consistency. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15454–15463, 2025.

Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pp. 5650–5659. Pmlr, 2018.

Liangqi Yuan, Ziran Wang, Lichao Sun, Philip S Yu, and Christopher G Brinton. Decentralized federated learning: A survey and perspective. *IEEE Internet of Things Journal*, 2024.

Sheng Yue and Ju Ren. Efficient federated meta-learning over multi-access wireless networks. *Next Generation Multiple Access*, pp. 547–581, 2024.

Zhiyuan Zhai, Xiaojun Yuan, Xin Wang, and Geoffrey Ye Li. Decentralized federated learning with distributed aggregation weight optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Hangtao Zhang, Zeming Yao, Leo Yu Zhang, Shengshan Hu, Chao Chen, Alan Liew, and Zhetao Li. Denial-of-service or fine-grained control: Towards flexible model poisoning attacks on federated learning. *arXiv preprint arXiv:2304.10783*, 2023.

Kaichuang Zhang, Alina Basharat, and Ping Xu. Byzantine-robust decentralized federated learning via local performance checking. In *31st International Conference on Neural Information Processing*, 2024.

Kaichuang Zhang, Wei Yin, Jinghao Yang, and Ping Xu. Dfedreweighting: A unified framework for objective-oriented reweighting in decentralized federated learning. *arXiv preprint arXiv:2512.12022*, 2025a.

Xusheng Zhang, Tuan Nguyen, and Ting He. Time-varying mixing matrix design for energy-efficient decentralized federated learning. *arXiv preprint arXiv:2512.24069*, 2025b.

Tongtian Zhu, Wenhao Li, Can Wang, and Fengxiang He. Dice: Data influence cascade in decentralized learning. In *International Conference on Learning Representations*, volume 2025, pp. 69007–69042, 2025.

A NOTATION

For ease of reference, Table 3 summarizes the key notation used throughout this paper.

Table 3: Key Notation

Symbol	Description
$\mathcal{G} = (\mathcal{N}, \mathcal{E})$	Communication graph. DFL system's communication graph, with client set $\mathcal{N}$ and communication-link set $\mathcal{E}$.
$\mathcal{C}$, $\mathcal{C}^{\star}_{\mathcal{G}}$	Topological distribution of compromised clients (TDCC). $\mathcal{C}^{\star}_{\mathcal{G}}$ is the optimal TDCC for graph $\mathcal{G}$ which can cause the best attack performance, respectively.
m	Compromise budget. The adversary can compromise at most m clients.
T	Total number of rounds.
$\boldsymbol{\theta}_k^{(t)}$, $\boldsymbol{\Theta}^{(t)}$	Local model of client k and model-state vector of all clients at round t, respectively.
$\boldsymbol{p}_k^{(t)}$, $\boldsymbol{P}^{(t)}(\mathcal{C})$	Perturbation injected by client k and the corresponding system-level perturbation matrix at round t under $\mathcal{C}$.
$\boldsymbol{\Phi}(\mathcal{C}, T, t)$	Accumulated mixing matrix. $\boldsymbol{\Phi}(\mathcal{C}, T, t) = \mathbf{W}^{(T-1)}(\mathcal{C}) \cdots \mathbf{W}^{(t)}(\mathcal{C})$.
$\boldsymbol{S}(\mathcal{C}, T)$	System-level model deviation accumulated over rounds $0, \ldots, T-1$.
$\omega_k(\mathcal{C}, T)$	Client-level accumulated-influence upper bound.

B THREE-STAGE LEARNING PROCESS OF DFL

Taking decentralized SGD as the optimization algorithm, all clients in the system executes the following three-stage learning process multiple times until convergence (Fang et al., 2024; Lalitha et al., 2018; Sun et al., 2022; Wu et al., 2023; Xie et al., 2025).

Stage 1 (local training): each client k perform one step of local SGD in parallel: $\boldsymbol{\theta}_k^{(t+\frac{1}{2})} = \boldsymbol{\theta}_k^{(t)} - \eta_k^{(t)} \nabla f_k(\boldsymbol{\theta}_k^{(t)}; \boldsymbol{\xi}^{(t)})$, where $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$ denotes the intermediate models of client k after training. $\eta^{(t)}$ denotes the learning rate and all clients hold same learning rate at same learning round and $\nabla f_k(\boldsymbol{\theta}_k^t; \xi_k^t)$ the stochastic gradient of the local objective f_k evaluated at $\boldsymbol{\theta}_k^{(t)}$ on a mini-batch $\xi_k^{(t)}$ sampled from the local dataset $\mathcal{D}_k$. From a global perspective (Zhai et al., 2025) (the perspective of the entire system)

$$\boldsymbol{\Theta}^{(t+\frac{1}{2})} = \boldsymbol{\Theta}^{(t)} - \eta^{(t)}\, \mathbf{G}^{(t)} \tag{12}$$

where $\mathbf{G}^{(t)} = \left[\mathbf{g}_1^{(t)}, \mathbf{g}_2^{(t)}, \ldots, \mathbf{g}_{|\mathcal{N}|}^{(t)}\right]^{\top} \in \mathbb{R}^{|\mathcal{N}| \times d}$ denote the gradients and $\mathbf{g}_k^{(t)} = \nabla f_k(\boldsymbol{\theta}^{(t)}; \boldsymbol{\xi}^{(t)})$ is the gradient of k-th local model at learning round t. $\eta^{(t)}$ is learning rate (step size) at learning round t.

Stage 2 (communication): after obtaining $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$, each client k exchanges its intermediate model with its one-hop neighbors according to the communication graph. Client k sends $\boldsymbol{\theta}_k^{(t+\frac{1}{2})}$ to its neighbors, client m ($m \in \mathcal{N}_k$), and receives $\boldsymbol{\theta}_m^{(t+\frac{1}{2})}$ from its neighbors, resulting in a local updates set $\boldsymbol{\Theta}_k^{(t+\frac{1}{2})}$ that includes client k' local model and updates from its neighbors: $\boldsymbol{\Theta}_k^{(t+\frac{1}{2})} = \{\boldsymbol{\theta}_k^{(t+\frac{1}{2})}\} \cup \{\boldsymbol{\theta}_m^{(t+\frac{1}{2})} | m \in \mathcal{N}_k\}$.

Stage 3 (aggregation): each client k then performs aggregation, $\boldsymbol{\theta}_k^{t+1} = \sum_{\boldsymbol{\theta}_i \in \boldsymbol{\Theta}_k^{(t+\frac{1}{2})}} w_{ki} \boldsymbol{\theta}_i^{(t+\frac{1}{2})}$.

where w_{ki} is the aggregation weight which client k assigned to received model $\boldsymbol{\theta}_i^{(t+\frac{1}{2})}$. From a global perspective

$$\boldsymbol{\Theta}^{(t+1)} = \mathbf{W}^{(t)} \boldsymbol{\Theta}^{(t+\frac{1}{2})} \tag{13}$$

where $\mathbf{W}^{(t)}$ is the mixing matrix at learning round t.

The DFL system repeats this three-stage learning process until either T rounds have been completed or all local models have converged, after which the resulting model $\boldsymbol{\Theta}^{(T)}$ can be deployed in practical applications.

C Definition

To characterize the multi-round mixing (aggregation) in the optimization problem in Eq. (5), we extend the notion of a single-round mixing matrix by introducing its accumulated counterpart. Each mixing matrix defined below is assumed to satisfy Assumptions 1, 3, 4.

Definition 3 (Accumulated Mixing Matrix)**.** *Consider a DFL system with communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, and let $\mathcal{C}$ denote a topological distribution of compromised clients. Let $\mathbf{W}^{(t)}(\mathcal{C})$ be the corresponding mixing matrix at learning round t. For any $0 \leq t < T$, the accumulated mixing matrix from round t to round T is defined as*

$$\mathbf{\Phi}(\mathcal{C}, T, t) := \prod_{\tau=t}^{T-1} \mathbf{W}^{(\tau)}(\mathcal{C}) = \mathbf{W}^{(T-1)}(\mathcal{C})\mathbf{W}^{(T-2)}(\mathcal{C}) \cdots \mathbf{W}^{(t)}(\mathcal{C}). \tag{14}$$

The product is ordered chronologically from right to left, with $\mathbf{W}^{(t)}(\mathcal{C})$ applied first and $\mathbf{W}^{(T-1)}(\mathcal{C})$ applied last. Thus, $\mathbf{\Phi}(\mathcal{C}, T, t)$ captures the cumulative effect of mixing over rounds $t, \ldots, T-1$.

Assumption 1 implies that every nonzero entry of $\mathbf{\Phi}(\mathcal{C}, T, t)$ arises from information propagated through a sequence of topology-compliant communication links. Thus, the accumulated mixing matrix describes both the multi-round propagation of model information (cumulative aggregation weights) and the information propagation trajectories.

D Decomposing System-level to Client-level Deviation

Since the optimization problem in Eq. (5) is subject to a cardinality constraint, exhaustively enumerating all feasible choices of the client subset $\mathcal{C}$ is computationally prohibitive in large-scale systems. We therefore decompose the system-level model deviation into client-level contributions.

First, we consider two cases for the topological distribution of compromised clients (TDCC): a nonempty set $\mathcal{C}$ and the empty set $\emptyset$. Applying Definition 3, we expand the aggregation equation in Eq. (13) for each case to obtain

$$\widetilde{\mathbf{\Theta}}^{(T)}(\mathcal{C}) = \mathbf{\Phi}(\mathcal{C}, T, 0)\mathbf{\Theta}^{(0)} - \sum_{t=0}^{T-1} \eta^{(t)} \mathbf{\Phi}(\mathcal{C}, T, t)\widetilde{\boldsymbol{G}}^{(t)}(\mathcal{C}) + \sum_{t=0}^{T-1} \mathbf{\Phi}(\mathcal{C}, T, t)P^{(t)}(\mathcal{C}), \tag{15}$$

$$\mathbf{\Theta}^{(T)}(\emptyset) = \mathbf{\Phi}(\emptyset, T, 0)\mathbf{\Theta}^{(0)} - \sum_{t=0}^{T-1} \eta^{(t)} \mathbf{\Phi}(\emptyset, T, t)\boldsymbol{G}^{(t)}(\emptyset). \tag{16}$$

Hence, with the Definition 1,

$$\begin{aligned} \boldsymbol{S}(\mathcal{C}, T) = \boldsymbol{D}(\mathcal{C}, T, 0)\mathbf{\Theta}^{(0)} &- \sum_{t=0}^{T-1} \eta^{(t)} \boldsymbol{D}(\mathcal{C}, T, t)\boldsymbol{G}^{(t)}(\emptyset) \\ &+ \sum_{t=0}^{T-1} \mathbf{\Phi}(\mathcal{C}, T, t)\Big[\boldsymbol{P}^{(t)}(\mathcal{C}) - \eta^{(t)}\big(\widetilde{\boldsymbol{G}}^{(t)}(\mathcal{C}) - \boldsymbol{G}^{(t)}(\emptyset)\big)\Big]. \end{aligned} \tag{17}$$

and $\boldsymbol{D}(\mathcal{C}, T, t) = \mathbf{\Phi}(\mathcal{C}, T, t) - \mathbf{\Phi}(\emptyset, T, t)$ We next bound the three terms in Eq. (17) separately.

Initialization term. The common initialization satisfies $\mathbf{\Theta}^{(0)} = \mathbf{1}(\mathbf{\Theta}^{(0)})^{\mathsf{T}}$. By Assumption 2, every accumulated mixing matrix is row-stochastic, and thus

$$\mathbf{\Phi}(\mathcal{C}, T, 0)\mathbf{1} = \mathbf{\Phi}(\emptyset, T, 0)\mathbf{1} = \mathbf{1}. \tag{18}$$

It follows that

$$\boldsymbol{D}(\mathcal{C}, T, 0)\mathbf{\Theta}^{(0)} = \boldsymbol{D}(\mathcal{C}, T, 0)\mathbf{1}(\mathbf{\Theta}^{(0)})^{\mathsf{T}} = \mathbf{0}_{|\mathcal{N}| \times d}. \tag{19}$$

Propagation-difference term. Assumptions 1and 2 ensure that both $\mathbf{\Phi}(\mathcal{C}, T, t)$ and $\mathbf{\Phi}(\emptyset, T, t)$ are nonnegative and row-stochastic. Therefore, for every output row i,

$$\sum_{k=1}^{|\mathcal{N}|} |[\boldsymbol{D}(\mathcal{C}, T, t)]_{ik}| \leq \sum_{k=1}^{|\mathcal{N}|} [\mathbf{\Phi}(\mathcal{C}, T, t)]_{ik} + \sum_{k=1}^{|\mathcal{N}|} [\mathbf{\Phi}(\emptyset, T, t)]_{ik} = 2. \tag{20}$$

Using Assumption 4,

$$\begin{aligned} \left\| \boldsymbol{D}(\mathcal{C}, T, t) \boldsymbol{G}^{(t)}(\emptyset) \right\|_{1,2} &= \sum_{i=1}^{|\mathcal{N}|} \left\| \sum_{k=1}^{|\mathcal{N}|} [\boldsymbol{D}(\mathcal{C}, T, t)]_{ik} \boldsymbol{g}_k^{(t)}(\emptyset) \right\|_2 \\ &\leq G_{\max} \sum_{i=1}^{|\mathcal{N}|} \sum_{k=1}^{|\mathcal{N}|} |[\boldsymbol{D}(\mathcal{C}, T, t)]_{ik}| \leq 2|\mathcal{N}| G_{\max}. \end{aligned} \tag{21}$$

Consequently,

$$\sum_{t=0}^{T-1} |\eta^{(t)}| \left\| \boldsymbol{D}(\mathcal{C}, T, t) G^{(t)}(\emptyset) \right\|_{1,2} \leq 2|\mathcal{N}| G_{\max} \sum_{t=0}^{T-1} |\eta^{(t)}|. \tag{22}$$

Perturbation and gradient-dissimilarity term. Define

$$\boldsymbol{Q}^{(t)}(\mathcal{C}) := \boldsymbol{P}^{(t)}(\mathcal{C}) - \eta^{(t)} \big(\widetilde{\boldsymbol{G}}^{(t)}(\mathcal{C}) - \boldsymbol{G}^{(t)}(\emptyset) \big). \tag{23}$$

For an arbitrary $\boldsymbol{Q} \in \mathbb{R}^{|\mathcal{N}| \times d}$, the triangle inequality and nonnegativity of $\mathbf{\Phi}(\mathcal{C}, T, t)$ give

$$\begin{aligned} \|\mathbf{\Phi}(\mathcal{C}, T, t) \boldsymbol{Q}\|_{1,2} &= \sum_{i=1}^{|\mathcal{N}|} \left\| \sum_{k=1}^{|\mathcal{N}|} [\mathbf{\Phi}(\mathcal{C}, T, t)]_{ik} \boldsymbol{Q}_{k,:} \right\|_2 \\ &\leq \sum_{i=1}^{|\mathcal{N}|} \sum_{k=1}^{|\mathcal{N}|} [\mathbf{\Phi}(\mathcal{C}, T, t)]_{ik} \|\boldsymbol{Q}_{k,:}\|_2 \\ &= \sum_{k=1}^{|\mathcal{N}|} \|[\mathbf{\Phi}(\mathcal{C}, T, t)]_{:,k}\|_1 \, \|\boldsymbol{Q}_{k,:}\|_2. \end{aligned} \tag{24}$$

Applying Assumptions 3 and 5 to the two terms in Eq. (23) gives, for every k,

$$\left\| [\boldsymbol{Q}^{(t)}(\mathcal{C})]_{k,:} \right\|_2 \leq B_p + |\eta^{(t)}| \sigma. \tag{25}$$

Combining Eq. (24) and Eq. (25),

$$\left\| \mathbf{\Phi}(\mathcal{C}, T, t) Q^{(t)}(\mathcal{C}) \right\|_{1,2} \leq \sum_{k=1}^{|\mathcal{N}|} \|[\mathbf{\Phi}(\mathcal{C}, T, t)]_{:,k}\|_1 \left(B_p + |\eta^{(t)}| \sigma \right). \tag{26}$$

Let

$$B_\eta := \sum_{t=0}^{T-1} |\eta^{(t)}|. \tag{27}$$

Under Assumptions 1–5,

$$\boxed{\|S(\mathcal{C}, T)\|_{1,2} \leq \sum_{k=1}^{|\mathcal{N}|} \sum_{t=0}^{T-1} \|[\mathbf{\Phi}(\mathcal{C}, T, t)]_{:,k}\|_1 \left(B_p + |\eta^{(t)}| \sigma \right) + 2|\mathcal{N}| G_{\max} B_\eta.} \tag{28}$$

E PROOF OF THEOREM 1

In Theorem 1, the optimal compromised-client distribution with at most m clients in a DFL system $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ is

$$\mathcal{C}_{\mathcal{G}}^{\star} = \{\pi_i \mid i = 1, \ldots, m\}. \tag{29}$$

Proof. For a DFL system with communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, let $\mathcal{C} \subset \mathcal{N}$ be an arbitrary compromised-client set, and $|\mathcal{C}| \leq m$.

By Assumption 1, $\mathbf{\Phi}(\mathcal{C}, T, t)$ is also compliance which means $\mathbf{\Phi}(\mathcal{C}, T, t) = [\mathbf{\Phi}(\mathcal{C}, T, t)_{i,j}] \in [0, 1)^{|\mathcal{N}| \times |\mathcal{N}|}$. Since $B_p > 0, \sigma > 0, G_{max} > 0$ under Assumption 3, 4, 5, Definition 2 implies that $\omega_k(\mathcal{C}, T) > 0$.

Moreover, since the client-level accumulated model deviations are ordered non-increasingly, $\omega_{\pi_1}(\mathcal{C}, T) \geq \omega_{\pi_2}(\mathcal{C}, T) \geq \cdots \geq \omega_{\pi_{|\mathcal{N}|}}(\mathcal{C}, T) > 0$, the sum of the deviations for any subset of at most $|\mathcal{C}|$ clients cannot exceed the sum of the $|\mathcal{C}|$ largest deviations. Therefore,

$$\sum_{k \in \mathcal{C}} \omega_k(\mathcal{C}, T) \leq \sum_{i=1}^{m} \omega_{\pi_i}(\mathcal{C}, T) = \sum_{k \in \mathcal{C}_{\mathcal{G}}^{\star}} \omega_k(\mathcal{C}, T). \tag{30}$$

Hence, $\mathcal{C}_{\mathcal{G}}^{\star}$ maximizes the total client-level accumulated model deviation under compromising at most m clients. □

F EXPERIMENTAL SETTINGS

F.1 COMMUNICATION GRAPHS

Following the setting of (Syros et al., 2024; Piaseczny et al., 2024), we evaluate our method on 7 representative communication graphs $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, as illustrated in Figure 3.

(1) Random (Erdős–Rényi): As shown in Figure 3a, random graph (Erdős–Rényi Graph) used in (Syros et al., 2024) is parameterized by the connection probability ρ_r, and is commonly used to model networks where communication links are formed randomly and independently, but connected. So, it is a kind of general graph.

(2) Small-world (Watts-Strogatz): As shown in Figure 3b, small-world graph (Watts-Strogatz Graph)(Watts & Strogatz, 1998) is parameterized by the local neighbor degree N_s and rewiring probability ρ_s, and is widely used to characterize networks with high clustering and short average path lengths, such as social networks.

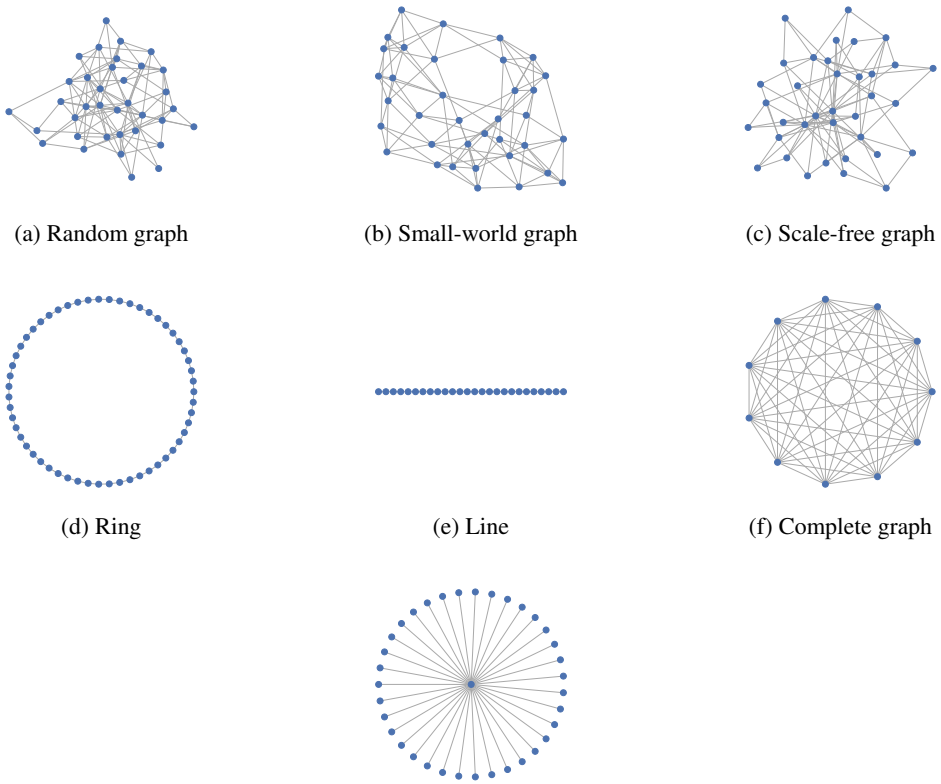
(3) Scale-free (Barabasi-Albert): As shown in Figure 3c, scale-free graph (Barabasi-Albert Graph) (Albert & Barabási, 2002) is controlled by the attachment parameter N_f, and is commonly used to model hub-dominated networks generated by preferential attachment, such as Internet, citation, and social networks.

(4) Ring: As shown in Figure 3d, ring graph (Wang et al., 2022) connects each client to a fixed number of neighboring clients in a cyclic manner, and is often used to represent sparse and regular communication systems with local message passing.

(5) Line (Mokhtari & Ribeiro, 2016): As shown in Figure 3e, line graph, also known as a path graph, connects clients sequentially and serves as an extremely sparse topology for evaluating multi-hop communication and propagation effects.

(6) Complete Graph: In a complete graph (Figure 3f), each client is directly connected to each of the other $|\mathcal{N}| - 1$ clients. Consequently, all clients have the same degree and the same relative position in the graph.

(7) Star: As shown in Figure 3g, a star graph consists of one central client connected to each of the other $|\mathcal{N}| - 1$ clients, while those clients are not directly connected to one another (Selo et al., 2025).



(a) Random graph (b) Small-world graph (c) Scale-free graph

(d) Ring (e) Line (f) Complete graph

(g) Star graph

Figure 3: *7 representative communication topologies in graph-based DFL systems.*

F.2 Baseline Methods

To evaluate the performance of our method, we compare it against the following baseline methods:

(1) Random: The compromised clients are selected uniformly at random from all clients in the DFL system. Because this strategy has been adopted in a lot of prior studies, it serves as a standard baseline for comparison.

(2) Maximum Degree (Max-Degree): The Max-Degree strategy selects the $|\mathcal{C}|$ clients with the highest degrees as compromised clients.

(3) Maximum Effective Network Size (Max-ENS): Maximum Effective Network Size (Burt, 2018) selects the $|\mathcal{C}|$ nodes with the largest effective network size, favoring nodes whose neighbors are less redundant and thus provide broader non-overlapping influence coverage.

(4) Maximum PageRank score (Max-PR): Maximum PageRank score(Page et al., 1999) selects the $|\mathcal{C}|$ nodes with the highest PageRank scores, targeting structurally important nodes that are central in the graph connectivity pattern.

(5) Maximum clustering coefficient (Max-C^2): Maximum clustering coefficient selects the $|\mathcal{C}|$ nodes with the highest clustering coefficients, targeting nodes embedded in highly connected local neighborhoods where malicious updates may propagate more effectively.

(6) MaxSpAN-FL (Max-Span) (Piaseczny et al., 2024) MaxSpAN-FL aims to maximize the spatial separation among adversarial clients.

(7) Hopping-Augmented MaxSpAN-FL (HA-Max-Span) (Piaseczny et al., 2024) Hopping-Augmented MaxSpAN-FL extends this strategy by introducing a probabilistic hopping mechanism guided by node centrality, thereby further dispersing adversarial clients across the graph.

Table 4: *TER± standard deviation comparison under nonlinear aggregation rules*. For Max-CL-AMD (Ours), percentages in square brackets indicate improvement relative to Random Selection. The best and second-best TER in each setting are shown in bold dark blue and bold muted blue, respectively.

Network	Aggregation	Random	Max-Clust.	Max-ENS	Max-Span	Max-CL-AMD (Ours)↑
ER	Median	10.01±0.06	10.02±0.08	**33.64±33.51**	12.95±4.41	**33.62±33.53**[**235.93%**]
	Krum	**9.99±0.09**	**9.95±0.13**	9.93±0.12	9.94±0.15	9.93±0.12[−0.58%]
	Multi-Krum	18.52±6.15	18.07±5.71	**62.42±27.80**	35.37±30.03	**65.34±29.05**[**252.74%**]
	BALANCE	9.90±0.13	**9.96±0.10**	**9.94±0.09**	9.90±0.11	9.92±0.10[0.22%]
	FLTrust	10.41±0.09	10.36±0.09	**10.69±0.13**	10.37±0.13	**10.67±0.14**[**2.53%**]
	Clipped Gossip	34.21±1.62	34.25±4.39	**52.93±3.93**	40.01±8.19	**51.89±5.73**[**51.67%**]
WS	Median	11.02±2.24	10.00±0.09	**23.35±17.90**	10.02±0.12	**11.95±2.60**[**8.53%**]
	Krum	10.03±0.14	9.97±0.15	**10.06±0.20**	10.01±0.15	**10.06±0.13**[**0.28%**]
	Multi-Krum	37.34±19.69	17.56±8.76	**81.12±3.12**	37.81±28.27	**80.78±1.80**[**116.37%**]
	BALANCE	**9.95±0.13**	9.93±0.06	**9.99±0.12**	9.92±0.09	9.88±0.10[−0.70%]
	FLTrust	10.39±0.10	10.37±0.08	**10.48±0.13**	10.38±0.08	**10.48±0.11**[**0.92%**]
	Clipped Gossip	38.55±1.78	35.62±5.27	**43.90±7.90**	39.24±5.60	**39.71±11.54**[**3.01%**]
BA	Median	10.94±2.06	9.96±0.13	**33.19±29.32**	10.88±2.10	**44.70±35.76**[**308.49%**]
	Krum	9.97±0.11	9.99±0.17	**10.93±2.14**	10.00±0.14	**10.96±2.13**[**9.98%**]
	Multi-Krum	14.86±3.60	9.97±0.15	**51.11±19.21**	13.91±2.12	**55.92±22.66**[**276.22%**]
	BALANCE	**9.97±0.17**	9.93±0.15	9.97±0.09	9.94±0.16	**10.00±0.08**[**0.25%**]
	FLTrust	10.43±0.14	10.32±0.10	**10.80±0.08**	10.48±0.15	**10.83±0.08**[**3.84%**]
	Clipped Gossip	38.51±8.66	30.69±2.06	**57.56±3.71**	43.22±3.84	**57.90±3.18**[**50.34%**]

F.3 Performance Metrics

For classification tasks, we report the average test error rate across all benign clients

$$\mathrm{TER} = \frac{1}{|\mathcal{S}|} \sum_{k=1}^{|\mathcal{S}|} \mathrm{TER}_k, \tag{31}$$

where TER_k denotes the fraction of incorrectly classified test samples for client k on test dataset, depending on the task. All simulation results are averaged over 5 independent runs using different random seeds. Where indicated, we also report the variance across these runs.

F.4 Additional Experimental Results

F.4.1 Effect of Nonlinear Aggregation Rules (Additional)

In addition to Max-Degree, Max-PageRank, HA-Max-Span, and SWAP-BPI, we also compare our method with three additional baselines: Maximum Effective Network Size, Maximum clustering coefficient, and Max-Span. All baseline methods are described in Appendix F.2.

As shown in Table 4, our method achieves best or second-best attack performance in most settings. These results further demonstrate that our method achieves competitive performance across a wide range of settings.

F.4.2 Effect of Linear Aggregation

We also evaluate our method under linear aggregation rule that means the mixing matrix remains unchanged throughout the entire learning process $W^{(1)}(\mathcal{C}) = W^{(2)}(\mathcal{C}) = \cdots = W^{(T)}(\mathcal{C})$ for any TDCC $\mathcal{C}$. The mixing matrix remains unchanged when both the communication graph and the aggregation behavior are fixed throughout learning process. Accordingly, we adopt a fixed communication graph and use DFedAvg, which performs aggregation with predetermined weights. Although DFedAvg is not robust to malicious updates, it is suitable for this setting.

As shown in Figure 4, our method increases TER by 0.73%–1.54% relative to the Random. This improvement is modest because the aggregation rule (DFedAvg) lack robustness in an environment

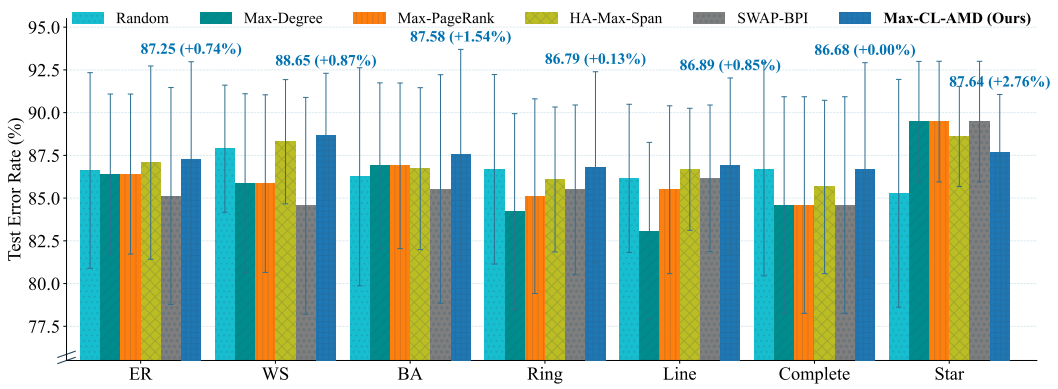


Figure 4: *Performance (TER%) comparison using linear aggregation in a Gaussian attack environment.*

with malicious clients, which leaves limited room for further performance degradation. Nevertheless, our method consistently ranks among the best-performing approaches in terms of attack performance when compared with the other baselines.

Moreover, linear aggregation allows us to evaluate our method using the ground-truth mixing matrix. The experimental results provide empirical support for Theorem 1.

F.4.3 Effect of network topologies (additional)

We also evaluate our method on seven representative communication topologies using different aggregation rules.

First, BALANCE

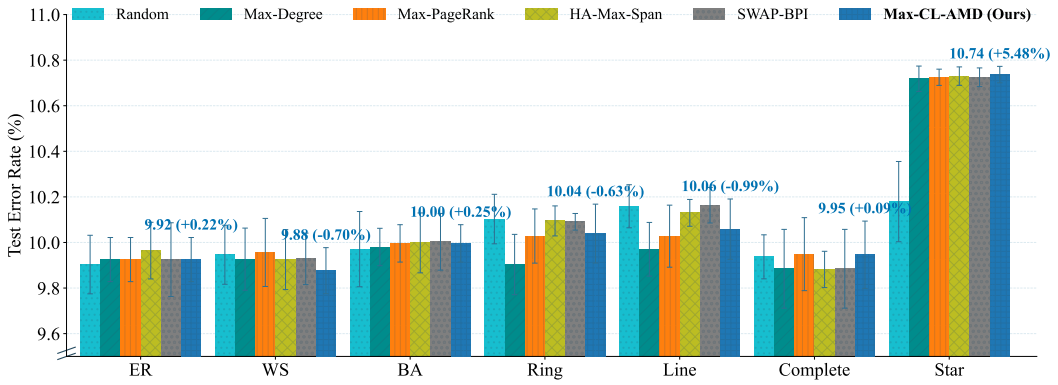


Figure 5: *Test Error Rate (TER%) comparison under different network topologies using BALANCE.*

As shown in Figure 5, the TER remains below 11% across all evaluated scenarios, suggesting that BALANCE maintains its robustness under these conditions.

Second, Clipped Gossip

As shown in Figure 6, our method performs competitively across all communication graphs. Relative to the random baseline on ER, BA and Star graphs, our method yields obvious improvements, from 51.67% to 80.95%. On WS and Complete graphs, our method also have small improvement. On Ring and Line graphs, our method even worse than Random Selection.

Third, FLTrust

As shown in Figure 7, these results have same conclusion with BALANCE.

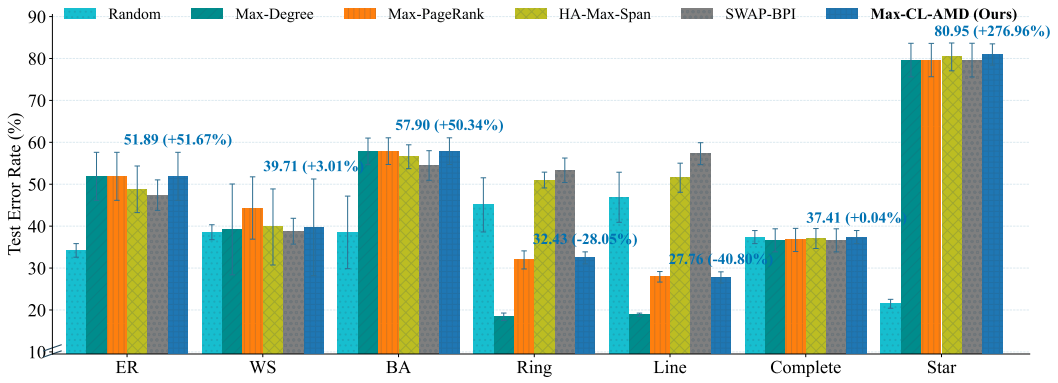


Figure 6: *Test Error Rate (TER%) comparison under different network topologies using Clipped Gossip.*

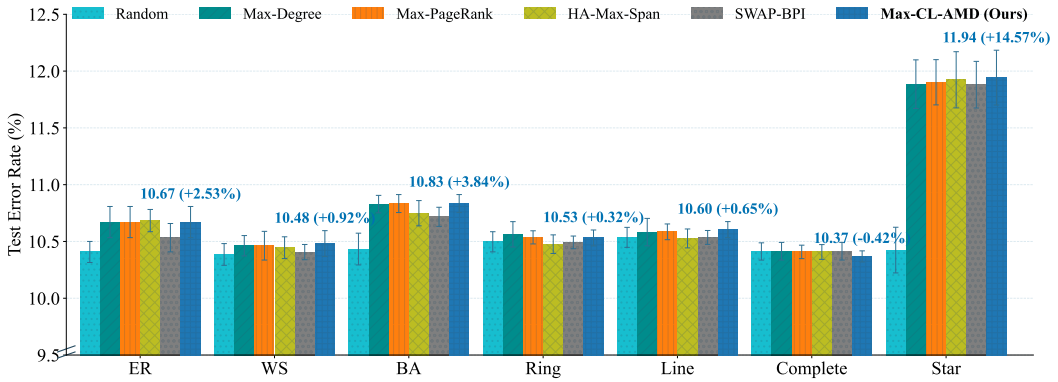


Figure 7: *Test Error Rate (TER%) comparison under different network topologies using fltrust.*

Summary: Attack performance in a DFL system is influenced by a wide range of factors, such as the attack strategy, communication topology, and aggregation rule. Our theoretical analysis and experimental results further indicate that the TDCC is another crucial factor.

F.4.4 Effect of different types of tasks and data heterogeneity (additional)

With only 20 clients, different methods may select the same TDCC. We therefore evaluate them in a larger DFL system, where each method selects 100 compromised clients in a 1000-client DFL system.

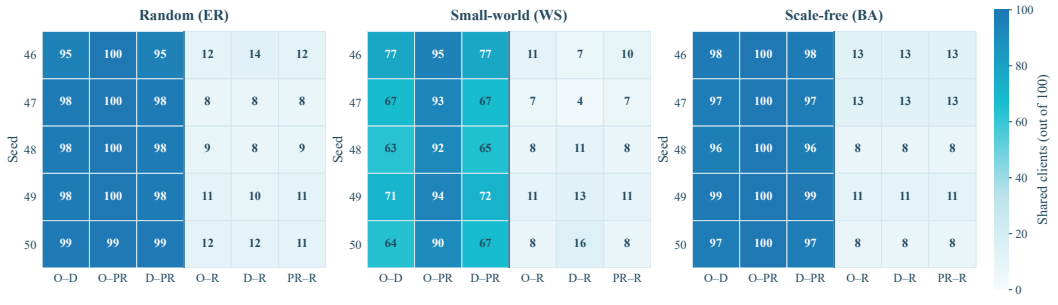


Figure 8: *Overlap among the sets of compromised clients selected by the four methods.* O, D, PR, and R denote Ours, Max-Degree, Max-PageRank, and Random, respectively.

Figure 8 shows that the overlap among MAX-CL-AMD, Max-PageRank, and Max-Degree depends on the network topology. MAX-CL-AMD and Max-PageRank select exactly the same 100 clients for all BA seeds and for four of the five ER seeds. They still share 90–95 compromised clients under WS. Max-Degree also closely agrees with them under BA and ER, but its overlap with MAX-CL-AMD drops to 63–77 clients under WS. Thus, the three methods often select similar clients but not exactly same, although their agreement is less consistent in the WS topology.