# Semantic-Statistical Prior Banks for Federated Low-Shot Learning under Non-IID Clients

**Dongmei Wei**[1] **Can Gao**[1] **Lei Shi**[2,3,*] **Jiaying Wu**[1] **Jia Luo**[4,5,6] **Feifei Yang**[1]
**Pengfei Zhang**[7,8,*] **Feifei Kou**[9] **Zhaoman Zhong**[1]

[1]School of Computer Engineering, Jiangsu Ocean University, China
[2]State Key Laboratory of Media Convergence and Communication,
Communication University of China, China
[3]Key Laboratory of Public Opinion Governance and Computational Communication,
Sichuan Police College, China
[4]Interdisciplinary Faculty of Science and Engineering, Shimane University, Japan
[5]College of Economics and Management, Beijing University of Technology, China
[6]Chongqing Research Institute, Beijing University of Technology, China
[7]School of Computer Science and Engineering, Anhui University of Science and Technology, China
[8]Yunnan Key Laboratory of Service Computing,
Yunnan University of Finance and Economics, China
[9]School of Computer Science (National Pilot Software Engineering School),
Beijing University of Posts and Telecommunications, China

## Abstract

Federated intrusion detection must adapt to rare attacks from only a few labeled flows per client, while client distributions are highly non-IID and raw traffic cannot be centralized. Cross-client class statistics can help estimate local class distributions from these limited samples, but standard parameter aggregation does not directly provide such information. We therefore propose a semantic-statistical prior bank that makes these statistics reusable for local low-shot adaptation. After federated representation learning, clients submit protected class-wise moment statistics through secure aggregation. The server uses these aggregated statistics to construct a global prior bank without accessing raw traffic. During local episodic adaptation, each client uses the exact class prior when available and retrieves a related prior through label-phrase embeddings otherwise. The selected prior is combined with scarce support statistics through maximum a posteriori (MAP) shrinkage in the feature space. The adapted class distributions are then used to train lightweight episode-specific classifiers on each client. Experiments on CICIDS2017 and CICDDoS2019 under 5-way 1-shot and 5-shot non-IID settings show consistent improvements over FedAvg and backbone-matched baselines, with the largest gains appearing in 1-shot and highly skewed client distributions. With ResNet1D, the proposed adaptation improves 1-shot F1 from 92.97 to 95.55 on CICIDS2017 and from 82.13 to 88.25 on CICDDoS2019.

## 1 Introduction

Network intrusion detection is crucial for IoT and distributed networks. Deep learning-based intrusion detection systems require sufficient labeled traffic, but a client may have only a few labeled support samples for certain attack types [8, 31]. Such limited support can produce unreliable class estimates, motivating the use of knowledge from other clients.

*Corresponding authors: Lei Shi (leiky_shi@cuc.edu.cn) and Pengfei Zhang (zpf.bupt@bupt.cn).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Federated learning allows clients to share this knowledge by jointly training a shared detector without uploading raw traffic [5, 7]. These clients may observe different attack classes and class frequencies, resulting in non-IID data. FedAvg [15] learns the shared detector by aggregating local model parameters. Even with this shared detector, class estimates from a small local support set can remain unreliable.

Prior-guided approaches [34] and semantic-guided approaches [27] can improve these estimates using class statistics and label information. In federated learning, CCVR [13] and FedPFT [1] use class distributions to generate features for training a global classifier. To use these class distributions for local low-shot detection, we need to construct reusable priors from protected cross-client statistics and adapt them to the current support set.

We therefore present a federated low-shot intrusion detection framework that connects prior construction with local adaptation through a semantic-statistical prior bank. After federated representation learning, clients protect class-wise feature sums with clipping and Gaussian noise and submit them through secure aggregation. The server uses the aggregated statistics to construct a reusable global prior bank. During local adaptation, each client uses the matching class prior and turns to semantic retrieval only when the exact class prior is unavailable. The selected prior is combined with the current support statistics through MAP estimation to adapt the class distribution. Features sampled from the adapted distributions are combined with real support features to train an episode-specific classifier. All adaptation and prediction are performed locally without uploading support or query data.

Experiments on CICIDS2017 and CICDDoS2019 show that the proposed adaptation improves FedAvg under non-IID partitions and benefits different backbones. The gains are especially clear in 1-shot episodes. With the same ResNet1D encoder and test episodes, our framework also achieves higher F1 scores than FedProto [22] and a CCVR-style baseline [13]. Component ablations show that shared class priors improve F1 on both datasets, while the effect of MAP mean adaptation varies across datasets.

The main contributions of this work are summarized as follows.

- We present a federated low-shot intrusion detection framework that connects protected cross-client prior construction with local adaptation under non-IID client distributions.
- We combine shared class priors with local support statistics through MAP shrinkage to estimate class distributions for feature generation and episode-specific classification. Semantic retrieval provides a related prior only when an exact class prior is unavailable.
- We evaluate the framework on CICIDS2017 and CICDDoS2019 and show improvements over FedAvg across the evaluated backbones and non-IID settings. Additional baseline comparisons and component ablations examine the contributions of shared priors, local adaptation, and feature generation.

## 2 Related Work

### 2.1 Low-Shot and Rare-Attack Intrusion Detection

Deep learning methods improve network intrusion detection by learning traffic representations through convolutional and recurrent networks [6, 25, 26], as well as graph-based models [16]. These methods often depend on abundant labeled traffic, which is unavailable for many rare attacks. Low-shot detectors address this scarcity through episodic learning and sample comparison [31, 33], adaptive loss weighting [14], and bidirectional support–query matching [32]. Learning useful representations from limited data is also studied in few-shot image classification, where attention prompts [30] reduce background interference. Data generation provides another way to address scarcity, as explored in generative IDS [35] and denoised generative fusion [28] for image classification. These advances improve the use of limited samples, but federated detection also requires knowledge sharing across clients with different attack distributions. We support this sharing through a reusable prior bank built from protected cross-client class statistics.

### 2.2 Federated Learning for Intrusion Detection

Federated learning enables collaborative model training while keeping raw data local [7, 15] and has been widely studied for intrusion detection [5, 9, 10]. For model updates, FedProx [11] regularizes local optimization under data heterogeneity, while work on totally asynchronous federated learning [3] analyzes convergence without synchronized updates. In intrusion detection, model sharing supports online collaboration in MetaCIDS [23] and personalized transfer learning [21] for rare attacks. The shared information can also take the form of class prototypes, which FedProto [22] aggregates to guide local learning. Beyond prototypes, CCVR [13] uses class distributions to generate virtual features, and FedPFT [1] transfers Gaussian mixtures of features from a frozen foundation model for one-shot federated learning. Both train a global classifier, whereas we adapt shared statistics to local support sets and train episode-specific classifiers.

### 2.3 Semantic-Statistical Prior Adaptation in Low-Shot FL IDS

Class statistics can serve as priors when a local support set is too small for reliable distribution estimation. Free Lunch [34] transfers base-class statistics to calibrate novel-class distributions and generate training features, while learnable distribution calibration [12] addresses class-incremental learning. Semantic prior selection [27] guides this transfer through class relationships, and PMCE [29] combines semantic retrieval with MAP prototype calibration and caption-guided enhancement. Semantic retrieval serves as an alternative in our framework, where Sentence-BERT [18] selects a related prior only when the exact class prior is unavailable. Unlike centralized prior transfer and server-side classifier calibration, our framework combines protected cross-client prior construction with adaptation to each client's current support set.

## 3 Preliminaries

### 3.1 Federated Low-Shot Intrusion Detection

We consider a federated low-shot intrusion detection setting with $K$ clients. Client $k$ holds a local traffic dataset

$$D_k = \{(x_i^k, y_i^k)\}_{i=1}^{n_k}, \tag{1}$$

where $x_i^k \in \mathbb{R}^d$ is a flow-level feature vector and $y_i^k \in \mathcal{Y}$ is its attack label. Raw traffic remains on local clients during training and inference. The server coordinates model training and maintains a global prior bank $\mathcal{P}$, but it does not receive raw flows, local support samples, or local query samples.

The label space $\mathcal{Y}$ is shared across training and evaluation and serves as a public schema for organizing class-wise statistics. Clients may observe different class subsets and class frequencies, which define the non-IID setting studied here. During prior construction, clients submit protected statistics through secure aggregation, and the server receives aggregated feature sums and class counts. The main evaluation uses an exact class prior for every episode class.

For an $N$-way episode evaluated by client $k$, let $\mathcal{C} \subseteq \mathcal{Y}$ denote its set of classes. Each class contributes $L$ labeled support samples and $M$ query samples.

$$S_k = \{(x_i^{k,c}, c)\}_{c\in\mathcal{C},\, i=1}^{L}, \quad Q_k = \{(x_j^{k,c}, c)\}_{c\in\mathcal{C},\, j=1}^{M}. \tag{2}$$

Here, $L$ is 1 or 5 in our experiments and specifies the support size for local adaptation. The feature extractor and prior bank are built from training data, while these evaluation episodes use the test split. The support set is used for local adaptation, and the query set is used for local evaluation.

Let $f_\theta$ be the global feature extractor learned through federated training. Given a query sample $x \in Q_k$, the client predicts its label by

$$\hat{y} = \arg\max_{c\in\mathcal{C}} p(c \mid x, S_k, \mathcal{P}, f_\theta). \tag{3}$$

The goal is to improve this prediction when only a few support samples are available.

### 3.2 MAP-Based Feature Shrinkage

For each episode class $c$, the client first extracts support features and computes their empirical mean and a diagonal covariance estimate, denoted by $\mu_{m,c}$ and $\Sigma_{m,c}$. The global prior bank provides

a prior mean and covariance, denoted by $\mu_{p,c}$ and $\Sigma_{p,c}$. We view the support mean as a noisy observation of the unknown class mean $\mu_c$.

$$\mu_{m,c} \mid \mu_c \sim \mathcal{N}\left(\mu_c, \frac{\Sigma_{m,c}}{L}\right), \tag{4}$$

where $L$ is the number of support samples per class. For $L > 1$, $\Sigma_{m,c}$ is estimated from the support features, while for $L = 1$, we use $\Sigma_{m,c} = \Sigma_{p,c}$. The prior bank gives a Gaussian prior

$$\mu_c \sim \mathcal{N}(\mu_{p,c}, \Sigma_{p,c}). \tag{5}$$

Under this model, the standard Gaussian posterior mean is also the maximum a posteriori estimate.

$$\hat{\mu}_c = \left(\Sigma_{p,c}^{-1} + L\Sigma_{m,c}^{-1}\right)^{-1} \left(\Sigma_{p,c}^{-1}\mu_{p,c} + L\Sigma_{m,c}^{-1}\mu_{m,c}\right). \tag{6}$$

With diagonal covariance matrices, this estimate becomes an element-wise shrinkage rule.

$$\hat{\mu}_c = \alpha_c \odot \mu_{m,c} + (\mathbf{1} - \alpha_c) \odot \mu_{p,c}, \quad \alpha_{c,j} = \frac{L[\Sigma_{p,c}]_{jj}}{L[\Sigma_{p,c}]_{jj} + [\Sigma_{m,c}]_{jj}}. \tag{7}$$

Here, $\odot$ denotes element-wise multiplication, $\mathbf{1}$ is the all-ones vector, and $\alpha_{c,j}$ is the $j$-th entry of $\alpha_c$. For $L = 1$, the support mean and prior mean receive equal weights.

We estimate the covariance for feature sampling separately from the adapted mean.

$$\hat{\Sigma}_c = \begin{cases} \Sigma_{p,c} + 0.01I, & L = 1, \\ \beta\Sigma_{m,c} + (1-\beta)\Sigma_{p,c} + 0.01I, & L > 1. \end{cases} \tag{8}$$

For $L > 1$, we set the support covariance weight to $\beta = 0.3$ for all classes. In both cases, adding $0.01I$ increases each feature variance by $0.01$ before sampling, where $I$ is the identity matrix.

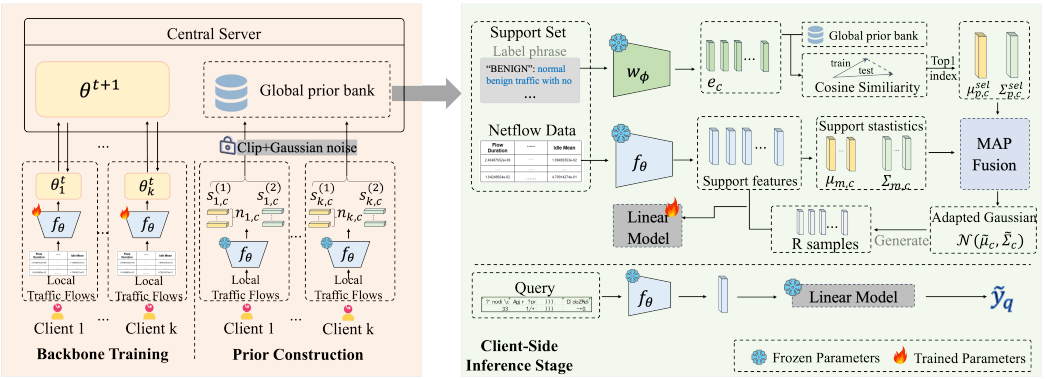



Figure 1: Overview of the proposed federated semantic-statistical prior bank framework.

## 4 Method

This section presents protected prior construction, local distribution adaptation, and client-side prediction, as shown in Figure 1.

### 4.1 Protected Federated Prior Construction

We first learn the feature extractor through federated training. At round $t$, client $k$ updates the model on local data and sends its local model parameters or parameter updates to the server. The server aggregates client models using FedAvg [15].

$$\theta^{t+1} = \sum_{k=1}^{K} \frac{n_k}{\sum_{j=1}^{K} n_j} \theta_k^{t+1}, \tag{9}$$

where $n_k$ is the local training sample count. The server broadcasts the global feature extractor $f_\theta$ to clients.

After representation learning, clients compute class-wise statistics from the feature

$$u_i^k = f_\theta(x_i^k). \tag{10}$$

For class $c$, client $k$ computes

$$s_{k,c}^{(1)} = \sum_{i:y_i^k=c} u_i^k, \quad s_{k,c}^{(2)} = \sum_{i:y_i^k=c} u_i^k \odot u_i^k, \quad n_{k,c} = |\{i : y_i^k = c\}|. \tag{11}$$

Here, $s_{k,c}^{(1)}$ and $s_{k,c}^{(2)}$ are the first- and second-order feature sums, and $n_{k,c}$ is the local class count. Clients use the shared label space $\mathcal{Y}$ and zero placeholders for absent classes.

Before aggregation, each client clips the feature sums at thresholds $C_1$ and $C_2$ and adds independent zero-mean Gaussian noise.

$$\tilde{s}_{k,c}^{(1)} = \mathrm{Clip}_{C_1}(s_{k,c}^{(1)}) + \epsilon_{k,c}^{(1)}, \quad \tilde{s}_{k,c}^{(2)} = \mathrm{Clip}_{C_2}(s_{k,c}^{(2)}) + \epsilon_{k,c}^{(2)}, \tag{12}$$

where $\epsilon_{k,c}^{(1)}$ and $\epsilon_{k,c}^{(2)}$ have standard deviations $\sigma_1$ and $\sigma_2$, respectively. Class counts are securely aggregated without noise.

Through secure aggregation [2], the server receives only aggregated noisy sums and class counts.

$$S_c^{(1)} = \sum_{k=1}^{K} \tilde{s}_{k,c}^{(1)}, \quad S_c^{(2)} = \sum_{k=1}^{K} \tilde{s}_{k,c}^{(2)}, \quad N_c = \sum_{k=1}^{K} n_{k,c}. \tag{13}$$

The server estimates the global prior mean and diagonal covariance by

$$\mu_{p,c} = \frac{S_c^{(1)}}{N_c}, \tag{14}$$

$$\Sigma_{p,c} = \mathrm{Diag}\left( \frac{S_c^{(2)}}{N_c} - \mu_{p,c} \odot \mu_{p,c} \right). \tag{15}$$

Here, Diag forms a diagonal matrix from a vector.

The resulting prior bank is

$$\mathcal{P} = \{(\mu_{p,c}, \Sigma_{p,c})\}_{c \in \mathcal{Y}}.$$

This exchange assumes an honest-but-curious server and clients that follow the protocol under the non-collusion conditions of secure aggregation. Individual submissions remain hidden under these conditions, while clients receive the global encoder and prior bank. These protections apply to prior construction and do not establish end-to-end differential privacy.

### 4.2 Prior Selection and Local Distribution Adaptation

The global prior bank provides class-level statistics for local adaptation. For each episode class $c$, the client directly selects its exact prior when available. In this case, $(\mu_{p,c}^{\mathrm{sel}}, \Sigma_{p,c}^{\mathrm{sel}}) = (\mu_{p,c}, \Sigma_{p,c})$. The main experiments follow this branch.

If the exact prior is unavailable, the client retrieves a related prior through label semantics. It rewrites the attack label as a short descriptive phrase $s_c$ and uses a frozen Sentence-BERT [18] encoder $w_\phi$ to compute

$$e_c = \frac{w_\phi(s_c)}{\|w_\phi(s_c)\|_2}. \tag{16}$$

For each stored class $r$, $e_r$ is computed in the same way. The retrieved prior is selected by

$$c^\star = \arg\max_{r \in \mathcal{Y}_\mathcal{P}} \cos(e_c, e_r), \quad (\mu_{p,c}^{\mathrm{sel}}, \Sigma_{p,c}^{\mathrm{sel}}) = (\mu_{p,c^\star}, \Sigma_{p,c^\star}), \tag{17}$$

where $\mathcal{Y}_\mathcal{P}$ denotes the set of classes stored in the prior bank.

The selected prior supplies $\mu_{p,c}$ and $\Sigma_{p,c}$ for the shrinkage rules in Section 3.2. These rules combine the prior with local support statistics to obtain the adapted class distribution $(\hat{\mu}_c, \hat{\Sigma}_c)$. This step updates the estimated feature distribution without changing labels or directly modifying classifier weights.

### 4.3 Client-Side Low-Shot Prediction

After prior-guided adaptation, each client obtains an adapted Gaussian distribution $\mathcal{N}(\hat{\mu}_c, \hat{\Sigma}_c)$ for every episode class $c$. The client samples $R$ synthetic features from this distribution.

$$\tilde{u}_s^{(c)} \sim \mathcal{N}(\hat{\mu}_c, \hat{\Sigma}_c), \quad s = 1, \ldots, R. \tag{18}$$

It combines these generated features with the real support features to form the labeled training set

$$T_c = \{(u_i, c) : (x_i, c) \in S_k\} \cup \{(\tilde{u}_s^{(c)}, c)\}_{s=1}^{R}, \tag{19}$$

where $u_i = f_\theta(x_i)$ is the support feature. The client trains an episode-specific linear classifier

$$h_\psi(u) = \mathrm{softmax}(Wu + b) \tag{20}$$

on $T = \bigcup_{c \in \mathcal{C}} T_c$, where $\psi = \{W, b\}$ denotes the classifier parameters. For a query sample $x_q$, the client computes $u_q = f_\theta(x_q)$ and predicts

$$\hat{y}_q = \arg\max_{c \in \mathcal{C}} h_\psi(u_q)_c. \tag{21}$$

All adaptation and prediction steps are performed on the client. The server provides the global feature extractor and prior bank. We compute each episode's metrics from predictions on its query set.

Table 1: Class distributions and feature dimensions of CICIDS2017 and CICDDoS2019.

| **CICIDS2017** | | | **CICDDoS2019** | | |
|---|---|---|---|---|---|
| **Class** | **Saved Rows** | **Features** | **Class** | **Saved Rows** | **Features** |
| BENIGN | 30,000 | 69 | BENIGN | 30,000 | 67 |
| DoS Hulk | 30,000 | 69 | DrDoS_DNS | 30,000 | 67 |
| DDoS | 30,000 | 69 | DrDoS_LDAP | 30,000 | 67 |
| PortScan | 30,000 | 69 | DrDoS_MSSQL | 30,000 | 67 |
| DoS GoldenEye | 10,286 | 69 | DrDoS_NTP | 30,000 | 67 |
| FTP-Patator | 5,931 | 69 | DrDoS_NetBIOS | 30,000 | 67 |
| DoS slowloris | 5,385 | 69 | DrDoS_SNMP | 30,000 | 67 |
| DoS Slowhttptest | 5,228 | 69 | DrDoS_SSDP | 30,000 | 67 |
| SSH-Patator | 3,219 | 69 | DrDoS_UDP | 30,000 | 67 |
| Bot | 1,948 | 69 | Portmap | 30,000 | 67 |
| Web Attack-Brute Force | 1,470 | 69 | Syn | 30,000 | 67 |
| Web Attack-XSS | 652 | 69 | TFTP | 30,000 | 67 |
| | | | UDP-lag | 30,000 | 67 |
| | | | WebDDoS | 439 | 67 |
| **Total** | 154,119 | 69 | **Total** | 390,439 | 67 |

## 5 Experiments

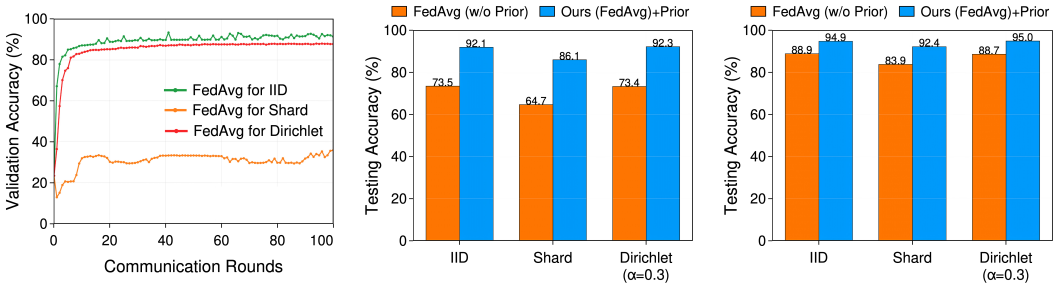
### 5.1 Experimental Setup

**Datasets and preprocessing.** We evaluate the proposed method on CICIDS2017 [19] and CICDDoS2019 [20]. We use the CSV flow records generated by CICFlowMeter. For CICIDS2017, we remove extremely rare classes with only a few dozen flows, including Infiltration, Web Attack-SQL Injection, and Heartbleed. For both datasets, we sample at most 30,000 flows per class and keep all flows for smaller classes. The final class distributions and feature dimensions are reported in Table 1. We remove leakage-prone identifiers, constant and duplicate columns, impute missing or infinite values, clip outliers, and standardize numerical features using training-split statistics.

**Low-shot protocol.** Evaluation follows 5-way 1-shot and 5-way 5-shot episodes. Unless otherwise stated, the feature extractor and prior bank are built with up to 500 training samples per class. For classes with fewer samples than the training limit, we reserve approximately half for testing and use all remaining samples for training, without oversampling. The episodic results in Tables 3 and 4 use 600 test episodes per dataset and shot setting. Within each episode, we draw the support samples and 15 queries per class without replacement. We report mean Accuracy and macro-averaged Precision, Recall, and F1 with 95% confidence intervals.

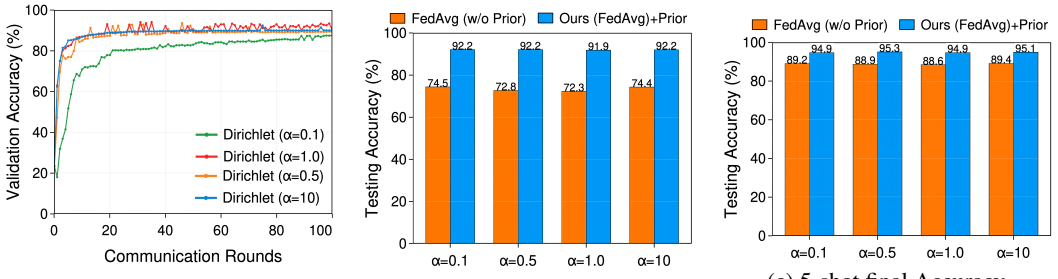
**Models and training.** We use MLP, CNN1D, and ResNet1D as federated feature extractors, optimized locally with Adam. The learning rates are 0.0001 for MLP and CNN1D and 0.001 for ResNet1D. Centralized deep-learning references are trained for 100 epochs. Sentence-BERT [18] with the all-mpnet-base-v2 checkpoint serves as the frozen semantic encoder. We generate $R = 600$ synthetic features per class and train the episode-specific linear classifier for 3000 iterations.

**Federated setup.** We use 10 clients and FedAvg for 100 communication rounds, with five local epochs per round. We consider IID, label-shard, and Dirichlet partitions, with the Dirichlet parameter varied in $\{0.1, 0.3, 0.5, 1.0, 10\}$. The with-prior and without-prior variants use identical data partitions and local training schedules. After federated training, each client performs local low-shot adaptation using the global feature extractor and prior bank.




(a) Convergence over rounds (b) 1-shot final Accuracy (c) 5-shot final Accuracy

Figure 2: Federated accuracy under IID, shard, and Dirichlet partitions ($\alpha = 0.3$ for Dirichlet). (a) Convergence; (b) final 1-shot; (c) final 5-shot, comparing FedAvg (w/o Prior) and Ours (+Prior).




(a) Convergence over rounds (b) 1-shot final Accuracy (c) 5-shot final Accuracy

Figure 3: Sensitivity to Dirichlet $\alpha$. (a) Representative 1-shot convergence; (b)–(c) final 1-shot and 5-shot results comparing FedAvg (w/o Prior) and Ours (+Prior). ($\alpha \in \{0.1, 0.5, 1.0, 10\}$; all other settings fixed.)

### 5.2 Federated Low-Shot Results under Non-IID Data

Figure 2 reports the main federated results. We compare FedAvg with and without prior-guided adaptation under IID, label-shard, and Dirichlet partitions. Both variants use the same MLP backbone, communication schedule, and local training protocol. Thus, the comparison evaluates the combined effect of the protected prior bank, local distribution adaptation, and feature generation.

The proposed prior-guided variant improves final accuracy in both 1-shot and 5-shot episodes. The improvement is more evident in 1-shot episodes, where local support statistics are less reliable. This result suggests that shared class information helps local adaptation when labeled support samples are scarce. The shared priors provide information about class variation, so generated features reduce the classifier's dependence on the few observed support samples.

Figure 3 further studies the effect of label heterogeneity by varying the Dirichlet parameter $\alpha$. FedAvg becomes stronger when $\alpha$ is large, since client label distributions are closer to IID. The proposed method still keeps a consistent advantage across all tested values of $\alpha$, showing that the federated prior bank reduces sensitivity to client label skew.

Table 2 shows that our method achieves higher mean F1 than FedProto and CCVR-style in all four settings. On CICDDoS2019, 1-shot F1 increases from 86.20 with CCVR-style to 88.25 with our method. This gain likely comes from adapting shared class distributions to the current support set, so the generated features better match the local task.

Table 2: Comparison with federated baselines using ResNet1D. Values are mean F1 scores in percent with 95% confidence intervals.

| | CICIDS2017 | | CICDDoS2019 | |
|---|---|---|---|---|
| Method | 1-shot | 5-shot | 1-shot | 5-shot |
| Support-only | $92.97 \pm 0.75$ | $94.48 \pm 0.63$ | $82.13 \pm 0.09$ | $87.73 \pm 0.85$ |
| FedProto [22] | $94.83 \pm 0.61$ | $94.93 \pm 0.62$ | $87.19 \pm 0.87$ | $88.45 \pm 0.83$ |
| CCVR-style [13] | $94.34 \pm 0.42$ | $95.32 \pm 0.42$ | $86.20 \pm 0.80$ | $88.35 \pm 0.79$ |
| Ours | $\mathbf{95.55 \pm 0.54}$ | $\mathbf{95.98 \pm 0.46}$ | $\mathbf{88.25 \pm 0.84}$ | $\mathbf{89.43 \pm 0.80}$ |

Each setting uses the same 600 test episodes across methods. Support-only uses real support features without prior adaptation or feature generation. CCVR-style samples from global class distributions without local MAP adaptation.

### 5.3 Effect of Prior-Guided Feature Adaptation

Table 3 and Table 4 evaluate the effect of prior-guided feature adaptation under low-shot episodes. The main comparison is between the same backbone with and without the prior module. This comparison evaluates the combined effect of prior-based distribution adaptation and feature generation with the backbone held fixed.

Table 3: Effect of prior-guided feature adaptation under 1-shot evaluation.

| | CICIDS2017 (1-shot) | | | | CICDDoS2019 (1-shot) | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Accuracy | Precision | Recall | F1 | Accuracy | Precision | Recall | F1 |
| Centralized deep-learning references | | | | | | | | |
| Transformer-Tabular [24] | $95.07 \pm 0.61$ | $95.18 \pm 0.65$ | $95.07 \pm 0.61$ | $94.65 \pm 0.68$ | $84.60 \pm 0.97$ | $84.10 \pm 0.01$ | $84.60 \pm 0.97$ | $82.97 \pm 0.11$ |
| CNN-LSTM [4] | $95.00 \pm 0.61$ | $95.16 \pm 0.64$ | $95.00 \pm 0.61$ | $94.59 \pm 0.68$ | $84.52 \pm 0.99$ | $83.74 \pm 0.16$ | $84.52 \pm 0.99$ | $82.74 \pm 0.15$ |
| GRU1D [17] | $89.51 \pm 0.76$ | $90.00 \pm 0.80$ | $89.51 \pm 0.76$ | $88.76 \pm 0.84$ | $84.12 \pm 0.99$ | $83.36 \pm 0.01$ | $84.12 \pm 0.99$ | $82.39 \pm 0.13$ |
| Federated models (three backbones; with and without Prior) | | | | | | | | |
| Ours – MLP (w/o Prior) | $77.43 \pm 0.89$ | $80.00 \pm 0.91$ | $77.43 \pm 0.89$ | $75.70 \pm 0.99$ | $73.84 \pm 0.99$ | $76.23 \pm 0.10$ | $73.84 \pm 0.99$ | $71.69 \pm 0.09$ |
| Ours – MLP (+Prior) | $91.40 \pm 0.59$ | $91.64 \pm 0.68$ | $91.40 \pm 0.59$ | $90.59 \pm 0.71$ | $85.61 \pm 0.81$ | $86.41 \pm 0.84$ | $85.61 \pm 0.81$ | $84.81 \pm 0.89$ |
| Ours – CNN1D (w/o Prior) | $63.76 \pm 0.97$ | $67.74 \pm 0.02$ | $63.76 \pm 0.97$ | $61.43 \pm 0.01$ | $66.48 \pm 0.64$ | $67.44 \pm 0.16$ | $66.48 \pm 0.64$ | $63.86 \pm 0.13$ |
| Ours – CNN1D (+Prior) | $90.98 \pm 0.53$ | $91.82 \pm 0.52$ | $90.98 \pm 0.53$ | $90.47 \pm 0.61$ | $83.90 \pm 0.85$ | $84.85 \pm 0.88$ | $83.90 \pm 0.85$ | $82.91 \pm 0.94$ |
| Ours – ResNet1D (w/o Prior) | $93.51 \pm 0.67$ | $93.65 \pm 0.72$ | $93.51 \pm 0.67$ | $92.97 \pm 0.75$ | $84.00 \pm 0.94$ | $83.46 \pm 0.10$ | $84.00 \pm 0.94$ | $82.13 \pm 0.09$ |
| Ours – ResNet1D (+Prior) | $95.92 \pm 0.46$ | $96.47 \pm 0.41$ | $95.92 \pm 0.46$ | $95.55 \pm 0.54$ | $88.87 \pm 0.77$ | $89.42 \pm 0.78$ | $88.87 \pm 0.77$ | $88.25 \pm 0.84$ |

Results use up to 500 training examples per class. The intervals are 95% confidence intervals over 600 test episodes.

Table 4: Effect of prior-guided feature adaptation under 5-shot evaluation.

| | CICIDS2017 (5-shot) | | | | CICDDoS2019 (5-shot) | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Accuracy | Precision | Recall | F1 | Accuracy | Precision | Recall | F1 |
| Centralized deep-learning references | | | | | | | | |
| Transformer-Tabular [24] | $95.62 \pm 0.54$ | $95.93 \pm 0.54$ | $95.62 \pm 0.54$ | $95.20 \pm 0.62$ | $87.39 \pm 0.85$ | $87.93 \pm 0.87$ | $87.39 \pm 0.85$ | $86.83 \pm 0.91$ |
| CNN-LSTM [4] | $95.68 \pm 0.53$ | $95.94 \pm 0.54$ | $95.68 \pm 0.53$ | $95.28 \pm 0.61$ | $87.42 \pm 0.83$ | $87.90 \pm 0.86$ | $87.42 \pm 0.83$ | $86.84 \pm 0.90$ |
| GRU1D [17] | $93.53 \pm 0.57$ | $93.89 \pm 0.57$ | $93.53 \pm 0.57$ | $93.31 \pm 0.60$ | $87.21 \pm 0.87$ | $87.76 \pm 0.88$ | $87.21 \pm 0.87$ | $86.68 \pm 0.92$ |
| Classical ML baselines (centralized; shots N/A) | | | | | | | | |
| SVM | $91.57 \pm 0.29$ | $82.20 \pm 0.39$ | $88.16 \pm 0.10$ | $82.35 \pm 0.22$ | $57.68 \pm 0.49$ | $56.72 \pm 0.21$ | $60.23 \pm 0.43$ | $52.10 \pm 0.53$ |
| kNN | $94.83 \pm 0.20$ | $86.88 \pm 0.20$ | $90.72 \pm 0.34$ | $87.51 \pm 0.18$ | $66.33 \pm 0.01$ | $68.69 \pm 0.43$ | $68.28 \pm 0.10$ | $64.31 \pm 0.01$ |
| AdaBoost | $66.63 \pm 0.06$ | $69.08 \pm 0.07$ | $59.37 \pm 0.09$ | $55.82 \pm 0.10$ | $46.31 \pm 0.07$ | $48.25 \pm 0.02$ | $45.00 \pm 0.06$ | $41.10 \pm 0.05$ |
| Decision Tree | $96.91 \pm 0.23$ | $89.41 \pm 0.64$ | $91.82 \pm 0.31$ | $89.62 \pm 0.44$ | $70.40 \pm 0.33$ | $71.01 \pm 0.72$ | $72.18 \pm 0.27$ | $70.04 \pm 0.47$ |
| Federated models (three backbones; with and without Prior) | | | | | | | | |
| Ours – MLP (w/o Prior) | $89.54 \pm 0.64$ | $90.37 \pm 0.66$ | $89.54 \pm 0.64$ | $89.01 \pm 0.71$ | $85.14 \pm 0.80$ | $85.93 \pm 0.83$ | $85.14 \pm 0.80$ | $84.47 \pm 0.87$ |
| Ours – MLP (+Prior) | $93.80 \pm 0.53$ | $94.08 \pm 0.54$ | $93.80 \pm 0.53$ | $93.58 \pm 0.57$ | $88.66 \pm 0.76$ | $88.90 \pm 0.79$ | $88.66 \pm 0.76$ | $88.14 \pm 0.82$ |
| Ours – CNN1D (w/o Prior) | $85.67 \pm 0.66$ | $86.86 \pm 0.67$ | $85.67 \pm 0.66$ | $85.07 \pm 0.72$ | $80.76 \pm 0.81$ | $81.77 \pm 0.84$ | $80.76 \pm 0.81$ | $79.98 \pm 0.87$ |
| Ours – CNN1D (+Prior) | $94.40 \pm 0.46$ | $94.73 \pm 0.45$ | $94.40 \pm 0.46$ | $94.25 \pm 0.49$ | $87.04 \pm 0.82$ | $87.46 \pm 0.84$ | $87.04 \pm 0.82$ | $86.49 \pm 0.88$ |
| Ours – ResNet1D (w/o Prior) | $94.88 \pm 0.55$ | $95.12 \pm 0.57$ | $94.88 \pm 0.55$ | $94.48 \pm 0.63$ | $88.28 \pm 0.79$ | $88.68 \pm 0.81$ | $88.28 \pm 0.79$ | $87.73 \pm 0.85$ |
| Ours – ResNet1D (+Prior) | $96.11 \pm 0.43$ | $96.42 \pm 0.40$ | $96.11 \pm 0.43$ | $95.98 \pm 0.46$ | $89.93 \pm 0.74$ | $90.22 \pm 0.77$ | $89.93 \pm 0.74$ | $89.43 \pm 0.80$ |

Episodic results use up to 500 training examples per class. The intervals are 95% confidence intervals over 600 test episodes. Classical ML baselines are centralized references, so the shot setting is not applicable to these methods.

Table 5 shows that global class priors improve F1 over support-only classification on both datasets. MAP mean adaptation increases F1 from 87.1 to 87.3 on CICDDoS2019 but reduces it from 94.2 to 93.1 on CICIDS2017. The full method achieves the highest mean F1, suggesting that the gains come from the combined use of shared priors, local distribution adaptation, and generated training features.

Table 5: Component ablation under 1-shot evaluation.

| Variant | CICIDS2017 | CICDDoS2019 |
|---|---|---|
| Support-only | $92.97 \pm 0.75$ | $82.13 \pm 0.09$ |
| Global prior mean | $94.21 \pm 0.62$ | $87.13 \pm 0.35$ |
| MAP-updated mean | $93.12 \pm 0.53$ | $87.33 \pm 0.60$ |
| Full method | $\mathbf{95.55 \pm 0.54}$ | $\mathbf{88.25 \pm 0.84}$ |

All prior-based variants use exact class priors. Global prior mean uses the stored class mean directly. MAP-updated mean combines the prior mean with local support statistics. The full method uses adapted class distributions to generate additional training features.

The prior-guided variant improves all three backbones on both datasets. The gain is most clear in 1-shot episodes. On CICIDS2017, ResNet1D improves from 92.97 to 95.55 F1 after adding the prior. On CICDDoS2019, the same backbone improves from 82.13 to 88.25 F1. The larger gain on CICDDoS2019 suggests that the prior is more useful when class distributions are harder to estimate from limited support samples.

The same trend also appears for weaker backbones. MLP and CNN1D benefit more from the prior than ResNet1D, especially in 1-shot episodes. This shows that the proposed module does not only rely on a stronger feature extractor. Instead, it stabilizes the estimated class distribution before the local classifier is trained. These results support the feature adaptation mechanism described in Section 3.2.

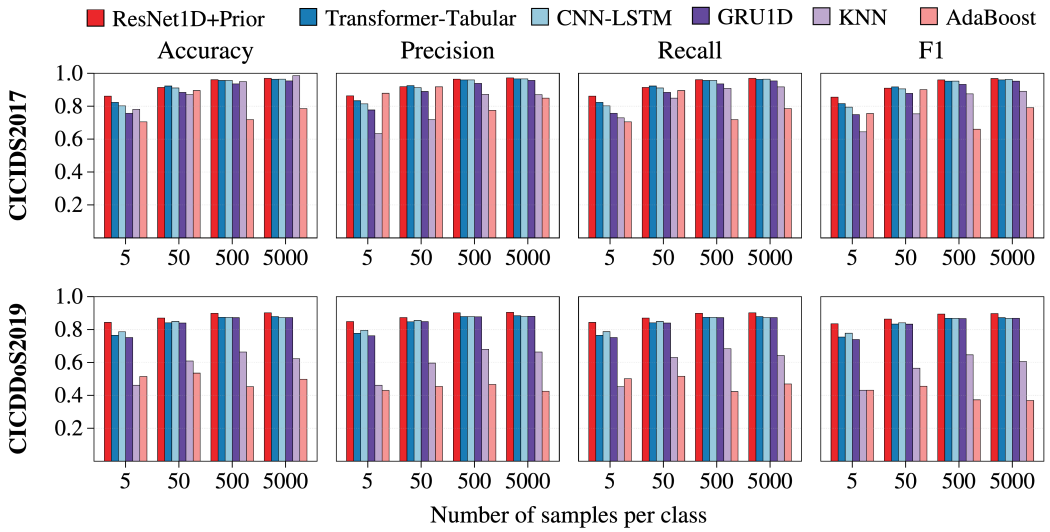



Figure 4: Training-size sensitivity under 5-way 5-shot evaluation with per-class training limits of $\{5, 50, 500, 5000\}$.

Figure 4 shows the training-size sensitivity under 5-shot evaluation. The prior-guided ResNet1D achieves the highest F1 with 5 training examples per class on both datasets. As the training size increases, the gap to other deep learning methods generally becomes smaller. At 50 training examples per class, Transformer-Tabular achieves a slightly higher F1 than our method on CICIDS2017. The larger advantage with limited training data may come from shared priors supplying additional class information through generated features.

### 5.4 Backbone and Shot-Level Comparison across Training Sizes

Figure 5 compares F1 scores across backbones and shot settings. The prior-guided variant improves MLP, CNN1D, and ResNet1D on both datasets. The gain is larger for weaker backbones and for 1-shot episodes. This suggests that shared class information is especially useful when local classification relies on very few labeled samples.

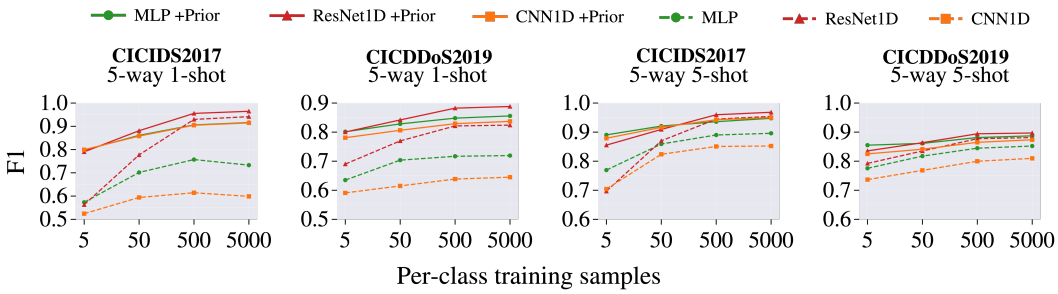



Figure 5: Backbone and shot-level F1 comparison on CICIDS2017 and CICDDoS2019. The x-axis gives the per-class training limit.

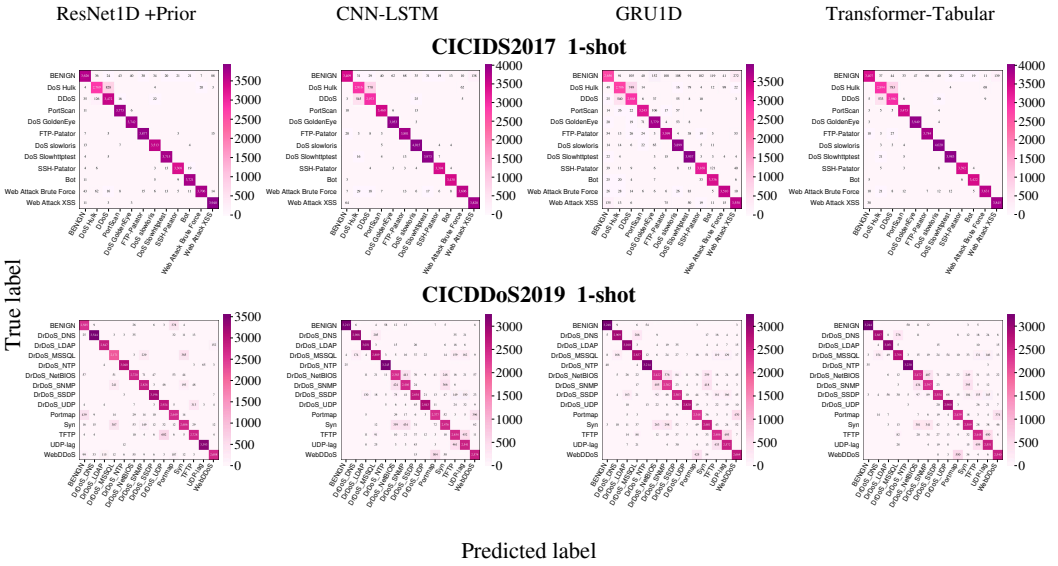



Figure 6: Confusion matrices under 1-shot evaluation on CICIDS2017 and CICDDoS2019.

The gap between lightweight backbones and ResNet1D becomes smaller after adding the prior. This may be because a few support samples provide limited information about class variation, especially when the backbone produces less informative features. Shared class statistics supplement this limited information, allowing local classifiers to draw on evidence beyond the selected support samples.

Figure 6 gives a class-level view under the 1-shot setting. On CICDDoS2019, prior-guided ResNet1D shows less confusion between Portmap and WebDDoS than the other models, but more between DrDoS_UDP and TFTP. These remaining errors may arise because one support sample per class provides too little information to distinguish similar traffic patterns.

## 6 Conclusion

We presented a federated low-shot intrusion detection framework with prior-guided feature adaptation. The method builds a protected global prior bank from client-side class statistics and combines its class priors with local support statistics for MAP-based adaptation. When an exact class prior is unavailable, semantic retrieval selects a related prior from the bank. Experiments on CICIDS2017 and CICDDoS2019 show that the proposed adaptation improves FedAvg under non-IID settings and brings consistent gains across MLP, CNN1D, and ResNet1D backbones, especially in 1-shot episodes. These results support the use of shared class information for local detection when labeled samples are scarce. Future work will study stronger federated baselines, formal privacy accounting, robust prior retrieval, and online prior refinement under traffic drift.

## Acknowledgments and Disclosure of Funding

This work is supported by the Key Laboratory of Public Opinion Governance and Computational Communication (Grant No. YQKFYB202501), the Fundamental Research Funds for the Central Universities (Grant No. CUC26TD13), the Youth Project of the Lianyungang Basic Research Program (Grant No. JCYJ2523), the Postgraduate Research and Practice Innovation Program of Jiangsu Province (Grant No. 26CXJH6480), the Haizhou Bay Talent Innovation Program of Jiangsu Ocean University (Grant No. KQ25062), the Foundation of Yunnan Key Laboratory of Service Computing (Grant No. YNSC26108), the National Natural Science Foundation of China (Grant No. 62677010), and the Beijing Natural Science Foundation (Grant No. L2603037).

The authors declare no competing interests for this paper.

## References

[1] Mahdi Beitollahi, Alex Bie, Sobhan Hemati, Leo Maxime Brunswic, Xu Li, Xi Chen, and Guojun Zhang. Foundation Models Meet Federated Learning: A One-Shot Feature-Sharing Method with Privacy and Performance Guarantees. *Transactions on Machine Learning Research*, 2025.

[2] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017)*, pages 1175–1191, 2017.

[3] Didier El Baz, Jia Luo, Hao Mo, and Lei Shi. A Mathematical Model and a Convergence Result for Totally Asynchronous Federated Learning. In *Proceedings of the 2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW 2024) (San Francisco)*, pages 956–963, Piscataway, NJ, USA, 2024. IEEE.

[4] Afrah Gueriani, Hamza Kheddar, and Ahmed Cherif Mazari. Enhancing IoT Security with CNN and LSTM-Based Intrusion Detection Systems. In *Proceedings of the 6th International Conference on Pattern Analysis and Intelligent Systems (PAIS 2024) (El Oued)*, pages 1–7. IEEE, 2024.

[5] José Luis Hernández-Ramos, Georgios Karopoulos, Efstratios Chatzoglou, Vasileios Kouliaridis, Enrique Mármol Campos, Aurora González-Vidal, and Georgios Kambourakis. Intrusion Detection Based on Federated Learning: A Systematic Review. *ACM Computing Surveys*, 57(12):309:1–309:65, 2025.

[6] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A Deep Learning Approach for Network Intrusion Detection System. *EAI Endorsed Transactions on Security and Safety*, 3(9):e2, 2016.

[7] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista A. Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and Open Problems in Federated Learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210, 2021.

[8] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges. *Cybersecurity*, 2(1):20, 2019.

[9] Ansam Khraisat, Ammar Alazab, Sarabjot Singh, Tony Jan, and Alfredo Gomez, Jr. Survey on Federated Learning for Intrusion Detection System: Concept, Architectures, Aggregation Strategies, Challenges, and Future Directions. *ACM Computing Surveys*, 57(1):7:1–7:38, 2025.

[10] Riccardo Lazzarini, Huaglory Tianfield, and Vassilis Charissis. Federated Learning for IoT Intrusion Detection. *AI*, 4(3):509–530, 2023. doi: 10.3390/ai4030028.

[11] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated Optimization in Heterogeneous Networks. In *Proceedings of Machine Learning and Systems (MLSys 2020)*, volume 2, pages 429–450, 2020.

[12] Binghao Liu, Boyu Yang, Lingxi Xie, Ren Wang, Qi Tian, and Qixiang Ye. Learnable Distribution Calibration for Few-Shot Class-Incremental Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12699–12706, 2023.

[13] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID Data. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, volume 34, pages 5972–5984, 2021.

[14] Wengang Ma, Liang Ma, Kehong Li, and Jin Guo. Few-Shot IoT Attack Detection Based on SSDSAE and Adaptive Loss Weighted Meta Residual Network. *Information Fusion*, 98: 101853, 2023.

[15] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS 2017)*, pages 1273–1282. PMLR, 2017.

[16] Chenyang Qiu, Yingsheng Geng, Junrui Lu, Kaida Chen, Shitong Zhu, Ya Su, Guoshun Nan, Can Zhang, Junsong Fu, Qimei Cui, et al. 3D-IDS: Doubly Disentangled Dynamic Intrusion Detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1965–1977, 2023.

[17] Mahrukh Ramzan, Muhammad Shoaib, Ayesha Altaf, Shazia Arshad, Faiza Iqbal, Ángel Kuc Castilla, and Imran Ashraf. Distributed Denial of Service Attack Detection in Network Traffic Using Deep Learning Algorithm. *Sensors*, 23(20):8642, 2023.

[18] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019) (Hong Kong)*, pages 3982–3992, 2019.

[19] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP)*, pages 108–116, 2018.

[20] Iman Sharafaldin, Arash Habibi Lashkari, Saqib Hakak, and Ali A Ghorbani. Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy. In *Proceedings of the International Carnahan Conference on Security Technology (ICCST)*, pages 1–8, 2019.

[21] Chunduru Sri Abhijit, Y Annie Jerusha, SP Syed Ibrahim, and Vijay Varadharajan. Federated Transfer Learning for Rare Attack Class Detection in Network Intrusion Detection Systems. *Scientific Reports*, 15(1):33797, 2025.

[22] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. FedProto: Federated Prototype Learning Across Heterogeneous Clients. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2022)*, volume 36, pages 8432–8440, 2022.

[23] Vu Tuan Truong and Long Bao Le. MetaCIDS: Privacy-Preserving Collaborative Intrusion Detection for Metaverse Based on Blockchain and Online Federated Learning. *IEEE Open Journal of the Computer Society*, 4:253–266, 2023.

[24] Shu-Ming Tseng, Yan-Qi Wang, and Yung-Chung Wang. Multi-Class Intrusion Detection Based on Transformer for IoT Networks Using CIC-IoT-2023 Dataset. *Future Internet*, 16(8): 284, 2024.

[25] R. Vinayakumar, Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, 7:41525–41550, 2019.

[26] Wei Wang, Yiqiang Sheng, Jinlin Wang, Xuewen Zeng, Xiaozhou Ye, Yongzhong Huang, and Ming Zhu. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection. *IEEE Access*, 6:1792–1806, 2018.

[27] Jiaying Wu and Jinglu Hu. Improved Prior Selection Using Semantics in Maximum A Posteriori for Few-Shot Learning. *Knowledge-Based Systems*, 237:107688, 2022.

[28] Jiaying Wu, Jingyu Chen, Jia Luo, Wenqian Yu, Jinglu Hu, and Hui Li. Denoised Generative Fusion Networks for Noise-Robust Few-Shot Image Classification. *Complex & Intelligent Systems*, 12(2):58, 2026.

[29] Jiaying Wu, Can Gao, Jinglu Hu, Hui Li, Xiaofeng Cao, and Jingcai Guo. PMCE: Probabilistic Multi-Granularity Semantics with Caption-Guided Enhancement for Few-Shot Learning. In *Proceedings of the 35th International Joint Conference on Artificial Intelligence (IJCAI-ECAI 2026) (Bremen)*, pages 1848–1856, 2026.

[30] Jiaying Wu, Zheng Huang, Jia Luo, Wenqian Yu, Jinglu Hu, and Zhaoman Zhong. Revisiting Few-Shot Image Classification: Harnessing Attention Prompt to Diminish Background Noise. *Expert Systems with Applications*, 331:133252, 2026.

[31] Congyuan Xu, Jizhong Shen, and Xin Du. A Method of Few-Shot Network Intrusion Detection Based on Meta-Learning Framework. *IEEE Transactions on Information Forensics and Security*, 15:3540–3552, 2020.

[32] Congyuan Xu, Fan Zhang, Ziqi Yang, Zhihao Zhou, and Yuqi Zheng. A Few-Shot Network Intrusion Detection Method Based on Mutual Centralized Learning. *Scientific Reports*, 15(1): 9848, 2025.

[33] Jingcheng Yang, Hongwei Li, Shuo Shao, Futai Zou, and Yue Wu. FS-IDS: A Framework for Intrusion Detection Based on Few-Shot Learning. *Computers & Security*, 122:102899, 2022.

[34] Shuo Yang, Lu Liu, and Min Xu. Free Lunch for Few-Shot Learning: Distribution Calibration. In *Proceedings of the International Conference on Learning Representations (ICLR 2021)*, 2021.

[35] Ze Zhang, Pan Wang, Tianqi Zhang, Minyao Liu, and Xiaokang Zhou. Trustworthy Generative Few-Shot Learning-Based Intrusion Detection Method in Internet of Things. *IEEE Transactions on Consumer Electronics*, 71(1):1992–2002, 2025.

# A Algorithmic Summary

**Prior bank storage and inference cost.** The prior bank is built once after federated representation learning and stored on the server with precomputed label embeddings. Clients then select priors, adapt class distributions, sample features, and train episode-specific classifiers locally.

**Algorithmic overview.** Algorithm 1 summarizes federated representation learning and protected prior construction. Algorithm 2 summarizes prior selection, class distribution adaptation, and local prediction.

**Algorithm 1** Offline construction of the semantic-statistical prior bank

**Require:** Client data $\{D_k\}_{k=1}^{K}$, label space $\mathcal{Y}$, encoders $f_\theta, w_\phi$, thresholds $C_1, C_2$, noise scales $\sigma_1, \sigma_2$
**Ensure:** Feature extractor $f_\theta$, prior bank $\mathcal{P}$, label embeddings $\{e_c\}$
**Federated representation learning.**
**for** each communication round **do**
    Each client $k$ updates its local model on $D_k$.
    The server aggregates updates with FedAvg and broadcasts $f_\theta$.
**end for**
**Class-wise sufficient statistics.**
**for** each client $k$ **do**
    **for** each class $c \in \mathcal{Y}$ **do**
        Extract features $u_i^k = f_\theta(x_i^k)$.
        Compute $s_{k,c}^{(1)}$, $s_{k,c}^{(2)}$, and $n_{k,c}$.
        Use zero placeholders for absent classes.
    **end for**
    Clip feature sums using $C_1, C_2$ and add Gaussian noise with scales $\sigma_1, \sigma_2$.
**end for**
Securely aggregate noisy sums and unnoised class counts across clients.
**Prior bank estimation.**
**for** each class $c \in \mathcal{Y}$ **do**
    Estimate $\mu_{p,c}$ and $\Sigma_{p,c}$ from aggregated statistics.
    Store $(\mu_{p,c}, \Sigma_{p,c})$ in $\mathcal{P}$.
    Store $e_c = w_\phi(s_c)/\|w_\phi(s_c)\|_2$.
**end for**
**return** $f_\theta$, $\mathcal{P}$, and stored label embeddings

# B Label Phrases for Semantic Retrieval

**Label phrase design.** The semantic encoder uses short natural-language phrases instead of raw attack labels. This gives the encoder more context about each attack behavior and avoids relying on abbreviated tokens only. Tables 6 and 7 report the exact phrases used in our experiments.

# C Prior Retrieval and Label Wording

We evaluate prior retrieval on CICDDoS2019 when the exact class prior is unavailable. For each target class, we hide its exact prior and select a replacement from the remaining bank. The exact-prior reference retains the corresponding class prior.

Table 8 shows that raw labels outperform descriptions and paraphrases, with paraphrases giving a 5-shot F1 close to that of random retrieval. Nearest-statistics retrieval exceeds all three semantic variants, suggesting that feature similarity provides more useful guidance for prior selection in this setting. All retrieval methods remain below the exact-prior reference.

We study noise sensitivity in a separate CICDDoS2019 1-shot experiment with 10 clients, ResNet1D, and up to 500 prior-construction examples per class. The clipping thresholds are set to the 95th percentile of client-class sum norms. We use $C_1 = 12{,}700$ and $C_2 = 30{,}800$ on CICDDoS2019,

**Algorithm 2** Online client-side low-shot adaptation and prediction

**Require:** Prior bank $\mathcal{P}$ and embeddings $\{e_r\}$, encoders $f_\theta, w_\phi$, support and query sets $S_k, Q_k$, classes $\mathcal{C}$, shot count $L$, sample count $R$
**Ensure:** Query predictions $\{\hat{y}_q : x_q \in Q_k\}$
**for** each class $c \in \mathcal{C}$ **do**
  Extract support features $u_i = f_\theta(x_i)$ and compute $\mu_{m,c}$.
  **if** an exact prior for $c$ exists in $\mathcal{P}$ **then**
    Select the exact prior $(\mu_{p,c}, \Sigma_{p,c})$.
  **else**
    Encode $s_c$ as $e_c = w_\phi(s_c)/\|w_\phi(s_c)\|_2$.
    Select $(\mu_{p,c}, \Sigma_{p,c})$ by cosine similarity with stored embeddings.
  **end if**
  Set $\Sigma_{m,c} = \Sigma_{p,c}$ if $L = 1$, otherwise estimate it from support features.
  Compute $\hat{\mu}_c$ and $\hat{\Sigma}_c$ using Section 3.2.
  Sample $R$ features from $\mathcal{N}(\hat{\mu}_c, \hat{\Sigma}_c)$.
  Form labeled $T_c$ from support and generated features.
**end for**
Train $h_\psi(u) = \mathrm{softmax}(Wu + b)$ on $T = \bigcup_{c \in \mathcal{C}} T_c$.
**for** each query sample $x_q \in Q_k$ **do**
  Extract $u_q = f_\theta(x_q)$.
  Predict $\hat{y}_q = \arg\max_{c \in \mathcal{C}} h_\psi(u_q)_c$.
**end for**
**return** $\{\hat{y}_q : x_q \in Q_k\}$

Table 6: CICIDS2017 label phrases used for semantic prior retrieval.

| ID | Label | Phrase |
|---|---|---|
| 0 | BENIGN | normal benign network traffic without malicious activity |
| 1 | DoS Hulk | denial of service attack using Hulk flood technique |
| 2 | DDoS | distributed denial of service attack using multiple sources |
| 3 | PortScan | network port scanning activity to discover open ports |
| 4 | DoS GoldenEye | denial of service attack using GoldenEye flood technique |
| 5 | FTP-Patator | brute force attack against FTP service for unauthorized access |
| 6 | DoS slowloris | denial of service attack using Slowloris technique |
| 7 | DoS Slowhttptest | denial of service attack using Slow HTTP test method |
| 8 | SSH-Patator | brute force attack against SSH service for unauthorized access |
| 9 | Bot | botnet generated malicious traffic inside the network |
| 10 | Web Attack-Brute Force | web application brute force login attack |
| 11 | Web Attack-XSS | cross-site scripting attack targeting a web application |

and $C_1 = 12{,}800$ and $C_2 = 24{,}400$ on CICIDS2017. Independent zero-mean Gaussian noise is added to the clipped first- and second-order sums before secure aggregation. We vary both noise scales using $\sigma_1 = mC_1$ and $\sigma_2 = mC_2$. Class counts are securely aggregated without added noise.

Figure 7 shows that Accuracy and F1 decrease as noise increases. F1 reaches 88.1 at $m = 0.05$ and matches the support-only result of 84.0 at $m = 0.10$. This decline suggests that stronger perturbations reduce the usefulness of the shared prior for local adaptation. This experiment measures the utility cost of noise. It does not establish a formal differential privacy guarantee because class counts are unnoised and repeated releases require privacy accounting.

Table 7: CICDDoS2019 label phrases used for semantic prior retrieval.

| ID | Label | Phrase |
|---|---|---|
| 0 | BENIGN | normal benign network traffic without malicious activity |
| 1 | DrDoS_DNS | distributed denial of service attack targeting DNS service |
| 2 | DrDoS_LDAP | distributed denial of service attack targeting LDAP service |
| 3 | DrDoS_MSSQL | distributed denial of service attack targeting MSSQL database service |
| 4 | DrDoS_NTP | distributed denial of service attack targeting NTP (Network Time Protocol) service |
| 5 | DrDoS_NetBIOS | distributed denial of service attack targeting NetBIOS service |
| 6 | DrDoS_SNMP | distributed denial of service attack targeting SNMP (Simple Network Management Protocol) service |
| 7 | DrDoS_SSDP | distributed denial of service attack targeting SSDP (Simple Service Discovery Protocol) service |
| 8 | DrDoS_UDP | distributed denial of service attack using UDP (User Datagram Protocol) flooding |
| 9 | Portmap | malicious activity exploiting portmap service vulnerabilities, often used in DDoS amplification |
| 10 | Syn | TCP SYN flood attack, a type of denial of service targeting TCP handshake process |
| 11 | TFTP | malicious activity involving TFTP (Trivial File Transfer Protocol), potentially for unauthorized file transfer |
| 12 | UDP-lag | UDP-based lag-inducing attack, causing network latency through excessive UDP traffic |
| 13 | WebDDoS | denial of service attack specifically targeting web servers to disrupt website availability |

Table 8: Prior retrieval and label wording on CICDDoS2019. Values are mean F1 scores in percent with 95% confidence intervals.

| Method | 1-shot | 5-shot |
|---|---|---|
| Random retrieval | $45.28 \pm 0.61$ | $56.27 \pm 0.61$ |
| Sentence-BERT with raw labels | $55.72 \pm 0.43$ | $62.30 \pm 0.63$ |
| Sentence-BERT with descriptions | $51.44 \pm 0.44$ | $59.62 \pm 0.65$ |
| Sentence-BERT with paraphrases | $48.78 \pm 0.43$ | $56.32 \pm 0.63$ |
| Nearest-statistics retrieval | $71.08 \pm 0.39$ | $75.82 \pm 0.34$ |
| Exact-prior reference | $88.25 \pm 0.84$ | $89.43 \pm 0.80$ |

The first five rows exclude the target class prior. The last row retains this prior as a reference.

# D Noise Sensitivity

# E Number of Generated Features

We vary the number of generated features per class $R$ to examine its effect on low-shot detection. The experiment uses ResNet1D on CICDDoS2019 under 1-shot evaluation. Table 9 shows that F1 increases from 82.1 without generated features to 88.3 at $R = 600$. The gains become smaller as $R$ increases, suggesting that additional generated features provide less improvement at larger sample counts.

Table 9: F1 scores in percent.

| $R$ | 0 | 100 | 300 | 600 |
|---|---|---|---|---|
| F1 | 82.1 | 86.8 | 88.0 | **88.3** |

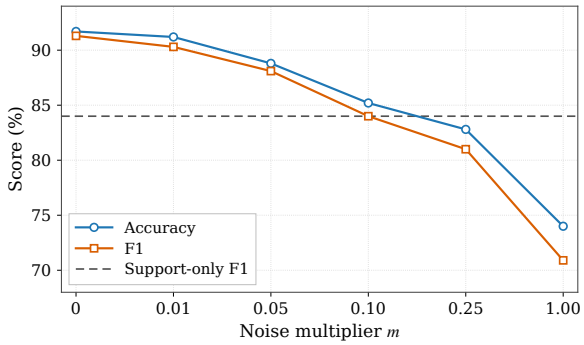



Figure 7: Noise sensitivity on CICDDoS2019 under 1-shot evaluation. The dashed line marks the support-only F1 of 84.0 in this controlled experiment. Tested noise levels are displayed as equally spaced settings.

# F Additional Experimental Details

**Training and test splits.** We use the same class-wise split rule for both datasets. Let $n_c$ be the number of samples in class $c$ and $T$ the per-class training budget. We shuffle each class with seed 42. We assign $\mathrm{round}(0.8n_c)$ samples to the training split if $n_c \geq 8T$, $T$ samples if $T \leq n_c < 8T$, and $\lceil n_c/2 \rceil$ samples if $n_c < T$. We reserve at least five samples for testing where possible. The training loader then uses up to $T$ samples per class from the training split. Evaluation episodes are drawn from the corresponding test split.

**Test episode construction.** We partition the training data among clients for federated training. For the episodic results in Tables 3 and 4, we draw 600 episodes from the dataset-wide held-out test split for each dataset and shot setting. The episodes are evaluated without assigning them to individual clients. Each episode contains five classes. For each class, we draw $L + 15$ samples without replacement and use $L$ support samples for adaptation and 15 query samples for evaluation. For panels (b) and (c) of Figures 2 and 3, we draw 5-way episodes from the held-out test split.

**Cross-validation and confidence intervals.** We use five-fold cross-validation within the training split to select the backbone model, without a separate validation split. Panel (a) of Figures 2 and 3 reports validation accuracy on held-out folds of the training split. For the main low-shot tables, we compute each metric separately for each of the 600 test episodes. The reported intervals are 95% Student-t confidence intervals, computed as the sample standard deviation divided by $\sqrt{600}$ and multiplied by the corresponding critical value with 599 degrees of freedom.

**Covariance settings.** We set each prior variance estimated from the noisy moments to at least $10^{-6}$. We use the resulting diagonal covariance in the MAP update and to compute the sampling covariance. We fix the support covariance weight at $\beta = 0.3$ for $L > 1$. In 1-shot episodes, we use the prior covariance with the additive covariance bias.

**Compute resources.** All experiments are conducted on Ubuntu 22.04 with a 12-core Intel CPU, 40 GB RAM, and an NVIDIA GeForce RTX 2080 Ti GPU. The software environment includes Python 3.8.10, PyTorch 2.4.1 with CUDA 12.1, SciPy, NumPy, and Matplotlib.

**Runtime measurements.** On a single RTX 2080 Ti, prior construction from saved ResNet1D features takes 0.84 s on CICIDS2017 and 2.33 s on CICDDoS2019. Evaluation over 600 episodes takes 45.1 s and 57.3 s for 1-shot and 5-shot tasks on CICIDS2017, and 75.6 s and 99.5 s on CICDDoS2019. These separate measurements exclude backbone training, flow feature extraction, and federated communication.

## G Responsible Research Details

**Limitations.** The evaluation uses two public intrusion detection datasets, so the results may not cover long-term traffic drift in deployed networks. The privacy mechanism protects class-wise statistics with clipping, noise perturbation, and secure aggregation, but it does not include a formal privacy accountant. The semantic retrieval step also depends on label phrasing, which may be less reliable when attack names are ambiguous or poorly described.

**Broader impacts.** This method targets intrusion detection in distributed networks where raw traffic cannot be easily centralized. It may improve security monitoring while reducing the need to share local data. The same detector can still make false alarms or miss attacks, which may affect operational decisions. Attackers may also adapt their behavior after learning how a deployed detector works. In practice, the method should support human security analysts rather than replace them.

**Existing Assets and Licenses.** We use three external assets. CICIDS2017 [19] and CICDDoS2019 [20] are public intrusion detection datasets provided by the Canadian Institute for Cybersecurity. We use their CSV flow records for research evaluation and do not redistribute the raw datasets. CICDDoS2019 allows redistribution, republication, and mirroring with citation to the dataset and related paper. For CICIDS2017, the official terms require citation of the dataset paper.[1] We use Sentence-BERT [18] with the `sentence-transformers/all-mpnet-base-v2` checkpoint from Hugging Face: `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`. The model card lists the license as Apache-2.0. We use the checkpoint only as a frozen semantic encoder.

[1] `https://www.unb.ca/cic/datasets/ids-2017.html`

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe the semantic-statistical prior bank, local low-shot adaptation, and the evaluated non-IID settings. Section 5 reports the supporting results on CICIDS2017 and CICDDoS2019.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix G discusses limitations related to dataset scope, privacy accounting, and semantic prior selection.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present new theoretical results. The MAP equations specify the estimator used in the method.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper describes the method, training procedure, and low-shot evaluation protocol. Appendix F provides additional experimental details.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper does not provide open-source code or scripts to reproduce the experiments.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5 reports preprocessing, episode construction, model backbones, optimization settings, and non-IID partitions. Appendix F describes the training and test splits and covariance settings.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Section 5 reports mean performance with 95% confidence intervals over 600 sampled test episodes for the main low-shot results.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix F reports the computing environment, including CPU, memory, GPU, operating system, Python version, PyTorch version, and supporting libraries.

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The study uses public intrusion detection datasets and does not involve human subjects, private user data collection, or deployment on real users.

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Appendix G discusses potential security benefits and reduced data sharing. It also discusses risks from false alarms, missed detections, and adaptive attackers.

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models, scraped datasets, image generators, or pre-trained language models, so no release safeguards are required.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Appendix G credits the datasets and the Sentence-BERT checkpoint and describes their terms of use.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not release new datasets, model weights, benchmarks, or code assets. It uses public datasets and describes the preprocessing and evaluation protocol.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, human annotation, or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects or private user data collection, so institutional review board approval is not applicable.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method does not use LLMs as an original or non-standard component. Sentence-BERT is used only as a frozen semantic encoder.