Neurocomputing 696 (2026) 134042

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Let's focus: Focused backdoor attack against federated transfer learning

Marco Arazzi [a,*], Stefanos Koffas [b], Antonino Nocera [a], Stjepan Picek [c,d]

[a] University of Pavia, Pavia, Italy
[b] Delft University of Technology, Delft, the Netherlands
[c] University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10000, Zagreb, Croatia
[d] Radboud University, Nijmegen, the Netherlands

HIGHLIGHTS

- Introduce FB-FTL, the first backdoor attack targeting Federated Transfer Learning.
- Reveal a critical vulnerability in FTL, achieving up to 96.7% attack success.
- Combine eXplainable AI and dataset distillation for focused trigger generation.
- Validate the attack on nine datasets, including real-world medical applications.
- Show that existing HFL and VFL defenses fail to mitigate the proposed attack.

ARTICLE INFO

Communicated by W. Cui

Keywords:
Federated transfer learning
Backdoor attack
Data poisoning
Defenses evaluation

ABSTRACT

Federated Transfer Learning (FTL) is the most general form of Federated Learning (FL). In FTL, one party, usually the server, pre-trains a feature extractor on public data. Then, clients collaboratively train a classifier by updating only the classification layers on their private data. This raises doubts about whether local poisoning attacks can effectively backdoor the full model. Unlike in FL, where attackers can shift model attention via poisoned inputs, FTL's fixed feature extractor, set during server pre-training, limits this possibility. In this paper, we investigate this scenario to identify and exploit a vulnerability obtained by combining eXplainable AI (XAI) and dataset distillation. Our proposed attack can be carried out by one of the clients during the FL phase of FTL by identifying the optimal position for the trigger through XAI and encapsulating compressed information of the backdoor class. Due to its behavior, we refer to our approach as a focused backdoor approach (FB-FTL for short) and test its performance by referencing image and text classification scenarios. Our attack is effective against existing defenses for FL, as it achieves an average of 80% attack success rate.

1. Introduction

Deep learning's progress with large data has raised privacy concerns in sensitive areas like finance and healthcare. Federated Learning (FL) [32] was developed to enable collaborative model training without sharing private data. Initially, FL focused on clients with data in the same feature space (Horizontal Federated Learning, HFL). Google and SWIFT recently used this paradigm to train fraud detection models across banks using transaction datasets.[1] Real-world scenarios often require collaboration across different feature spaces, known as Vertical Federated Learning (VFL) [28], where institutions such as banks and invoice agencies, such as those modeled by Consilient, jointly develop models by combining complementary datasets to fight fraud or money laundering. HFL and VFL could address privacy concerns while fostering collaboration across diverse industries, including healthcare, as highlighted by Guardora's work on medical confidentiality.[2]

The most general case involves parties collaborating in ML using local data that differ in both feature and sample space, a scenario addressed by Federated Transfer Learning (FTL) [28]. This paradigm

* Corresponding author.
Email addresses: marco.arazzi01@universitadipavia.it (M. Arazzi), S.Koffas@tudelft.nl (S. Koffas), antonino.nocera@unipv.it (A. Nocera), stjepan.picek@ru.nl (S. Picek).

[1] https://cloud.google.com/blog/products/identity-security/google-cloud-and-swift-pioneer-advanced-ai-and-federated-learning-tech.
[2] Guardora blog on medical confidentiality.

https://doi.org/10.1016/j.neucom.2026.134042
Received 11 November 2025; Received in revised form 28 March 2026; Accepted 20 May 2026
Available online 21 May 2026

has received significant attention from the research community and has been employed, for example, in [9,27,37], where the authors proposed novel architectures for personalized ML. We believe that it is crucial to understand the vulnerabilities this paradigm introduces, which are still relatively unexplored. For this reason, we investigate whether backdoor attacks can be crafted by manipulating local data on maliciously controlled parties involved in FTL.

Backdoor attacks have been extensively studied in recent years in HFL [6,13,47,48] and VFL [2,29]. However, to the best of our knowledge, our work is the first to target FTL and show that such a paradigm is vulnerable to backdoor attacks. Compared to classic FL settings, Federated Transfer Learning (FTL) imposes a fundamentally different threat model. In FTL, training is split into two stages: a feature extractor is first learned centrally on public data, and then frozen, while only the classification layers are collaboratively trained across clients using private data. This architectural constraint breaks a key assumption underlying existing backdoor attacks in horizontal and vertical FL [7,48,53], namely the ability of the global model to adapt its feature representation to encode malicious patterns. In FTL, the model's feature space cannot be modified by client updates, preventing the attacker from introducing new features through poisoned inputs. Consequently, standard strategies such as injecting visible triggers or manipulating hidden activations are largely ineffective. To succeed, a backdoor must instead be crafted to align with the pre-existing frozen feature representations, effectively exploiting the fixed feature embedding of the input data. This requirement makes backdoor attacks in FTL fundamentally distinct from those in classical FL and motivates the design of our FB-FTL approach, which leverages explainability techniques and dataset distillation to construct triggers that remain effective within the frozen feature space.

We overcome this challenge by adding an Explainable AI (XAI) step to the feature extractor with a data-driven approach to build a malicious trigger and suitably position it in high-attention locations learned by the global model. For this reason, we refer to our strategy as a *Focused* Backdoor (FB) attack. To do so, we leverage GradCam [42] to determine the most effective way for an adversary to alter the images. Moreover, we employ a distillation strategy [55] to encapsulate the main information characterizing the features of the target class in the trigger. By suitably locating our trigger in attention locations for the model's feature extractor and encapsulating a compressed representation of the target class within it, we can alter the classifier's behavior when such a trigger is present in the input image. Our experiments demonstrate that FTL is highly susceptible to backdoor attacks constructed according to our strategy.

Our main contributions are:

- We identify a critical vulnerability in FTL and introduce FB-FTL, the first backdoor attack against it. Based on our experiments, FB-FTL achieves an attack success rate of more than 80% on average and up to 96.7% in the best case.
- We evaluate our attack against real-world applications with medical data, confirming the baseline results.
- We extend the family of explainability-based attacks, defining and showing the importance of a "focusing" strategy for the positioning of the trigger using GradCam and SHAP according to the data domain. In addition, we exploit dataset distillation to generate a trigger containing the features of the target class.
- We test our approach against well-known defenses for HFL, and we also compare it against a novel VFL countermeasure based on Label Differential Privacy. Although some defenses are effective with specific datasets, none of the considered defenses can mitigate our approach in all the considered scenarios.

We evaluated our approach using nine datasets within the image and text domains, three consisting of actual medical data from real-world scenarios. We tested our attack against four different architectures and compared it with five HFL backdoor attacks. In addition, we tested our attack against six different defenses for both HFL and VFL.

An important element of our attack is the ability to adapt the trigger to the attacked data to minimize its impact in terms of variation in the main content of the data. This is especially true for images where triggers typically alter visible regions [48,53]. Training data is automatically processed by local clients, and therefore, the fact that the trigger may be visible to some extent does not necessarily represent a blocking point for the attacker. Still, minimizing its visible impact is crucial to make the attack as stealthy as possible. Our code is available at this link: https://anonymous.4open.science/r/Backdoor_Attack_FTL-E488.

2. Real-world FTL scenarios

FL is crucial to ensure data security and privacy across industries, employing both HFL and VFL to address different challenges. HFL, where institutions collaborate on similar datasets without sharing raw data, is exemplified by the partnership of Google with SWIFT to detect fraud in international banking transactions.[1] Similarly, Consilient uses HFL and data from more than 20 banks to collaboratively train a model that analyzes every international transaction from high-risk jurisdictions.[3] In healthcare, HFL facilitates privacy-preserving collaboration on similar patient datasets, while VFL allows different healthcare entities with complementary features to jointly develop machine learning models, ensuring confidentiality and regulatory compliance, as highlighted by Guardora.[2] These examples highlight the importance of securing both HFL and VFL to protect sensitive data and enable trust in federated AI systems. While HFL and VFL address specific data-partition scenarios in FL, they face certain limitations, such as handling scenarios where data overlap is minimal or absent across participants or when the feature and sample spaces differ significantly. These limitations motivated the development of FTL, which extends FL's capabilities by ingesting a model trained on source domain samples and feature space, enabling collaborative learning across non-overlapping data distributions and addressing heterogeneous data scenarios.

FTL is an emerging technology designed for specific scenarios and applications. Healthcare is a prime scenario for FTL, where smart devices create personalized models using patient-specific data from different institutions with a general encoder trained on public data [9]. Similarly, FTL can improve EEG signal classifiers due to the lack of large EEG datasets [20]. Autonomous driving also benefits, with vehicles using FTL to improve safety by recognizing pedestrians, other vehicles, and traffic signs [26].

For example, the healthcare company Owkin employs a similar approach to drug discovery [36], enabling pharmaceutical companies to collaborate securely on RNA sequencing (RNA-seq) analysis using FL.[4] This approach addresses the challenge of data sharing in drug discovery, as much of the data is considered a trade secret, preventing its disclosure to the public or competitors. Owkin's FL tools allow companies to harness the collective potential of their data while maintaining strict confidentiality and compliance with data privacy regulations. This technology is extensively covered in surveys, highlighting its versatility and state-of-the-art status in the literature [14,37,39].

3. Background

3.1. Federated learning

FL allows n clients to train a global model $\boldsymbol{w}$ collaboratively without revealing their datasets. According to [51], it can be classified into 1) HFL, 2) VFL, and 3) FTL.

In HFL, clients own data that share the same feature space but represent different entities [51]. Each client trains a local copy of the model

[3] https://consilient.com/our-products/high-risk-jurisdictions-model.

[4] https://www.biopharmatrend.com/post/1066-owkin-shares-federated-learning-tool-for-secure-rna-seq-analysis/.

and then uploads its weights ($\{w^i \mid i \in n\}$) to a server. HFL optimizes the following loss function: $\min_{w} \ell(w) = \sum_{i=1}^{n} \frac{k_i}{n} L_i(w),\ L_i(w) = \frac{1}{k_i} \sum_{j \in P_i} \ell_j(w, x_j)$, where $L_i(w)$ and k_i represent the loss function and local data size of the i-th client, and P_i refers to the set of data indices with size k_i. At the m-th iteration, the training can be divided into three steps. First, all clients download the global model w_m from the server. Then, each client updates the local model by training with their datasets ($w^i_m \leftarrow w^i_m - \alpha \frac{\partial L(w_m, b)}{\partial w^i_m}$, where α and b refer to the learning rate and local batch size). Finally, after the clients upload their local weights, the server updates the global model by aggregating the local weights. Various aggregation techniques can be used based on the given scenario. In this work, we will test five additional aggregation strategies in an extra experiment. The local model alignment is performed using the CORAL loss [9] as explained in Section 4.1.

In VFL, the clients own data that belong to the same entities but have different features [51]. Thus, their local models can differ as their data use different features [2]. The server, instead, holds the final layers for classification. Unlike HFL and FTL, in VFL, the clients generate representations for the local features of the shared entities using the local models. Then, these representations are used by the server as input to the top model classifier. Compared to HFL and FTL, the server does not aggregate the client updates to obtain the new global model. Instead, the propagation of the gradients to the bottom model is performed through the partial gradients of the partial representation of the entities.

3.2. Federated transfer learning

In transfer learning, a model trained for a specific task can be reused as a starting point for a different task, lowering the costs required for the development of the new model. FTL combines principles from Transfer Learning and FL. It consists of a central unit that aggregates updates from clients starting from a transfer model trained on a publicly available dataset. This model is then distributed to the clients who use their own datasets to improve its performance on their data distribution as described by the FedHealth's paradigm [9] which we employ as a baseline scenario. Unlike HFL, FTL addresses scenarios where clients possess datasets with different samples and feature spaces, making it closer to the VFL paradigm. However, while VFL focuses on combining complementary features across institutions (e.g., banks and invoice agencies), FTL emphasizes reusing a pre-trained model tailored to diverse tasks, enhancing model performance across heterogeneous datasets. This distinction makes FTL particularly valuable in real-world applications such as healthcare and finance, where diverse datasets are a common challenge and data privacy is paramount [18].

3.3. Backdoor attacks

The backdoor attack is a popular threat against deep learning models where the adversary adds a secret functionality into a model [13]. This functionality is activated by malicious inputs that contain a predefined property, the trigger. The backdoor's activation manipulates the model's behavior. For example, in autonomous driving, the backdoored model could classify a stop sign as a speed limit [13]. The backdoor can be embedded through data poisoning [13], code poisoning [5], or model poisoning [17]. To measure the attack's effectiveness, we use the attack success rate (ASR), which represents the number of times the backdoor is activated over the total poisoned samples fed to the network. To keep the attack stealthy, we need to ensure that the model's performance on the designated task is not affected by the backdoor insertion.

3.4. XAI

Interpretability in AI seeks to understand a model's decision. Various techniques can be used to this end. Recently, class activation mapping (CAM) was introduced [56]. CAM shows the areas of an image that affect the model's decision, but sacrifices the model's performance because it needs to modify the model's architecture [42]. Gradient-weighted CAM (GradCam) does not have this requirement [42]. Instead of modifying the model's architecture, it uses the gradient information that flows into the final convolutional layer to produce a localization of the crucial areas of the image that are connected to the model's decision. Additionally, SHAP [41] (SHapley Additive exPlanations) provides a versatile framework for interpretability by assigning an importance value to each feature for a specific prediction. Unlike visualization methods like GradCam, SHAP uses game-theoretic principles to attribute contributions to features, ensuring consistency. Its applicability extends beyond images to text, making it a robust explainability tool across diverse domains. In this work, we use GradCam as a baseline scenario to identify the optimal position of our triggers using images. However, any XAI technique can be employed for this purpose, as demonstrated in our text data experiments where we used the SHAP values.

4. Methodology

4.1. Attack scenario

As introduced in Section 3.2 and done in Chen et al. [9], in the considered scenario, we assume that the server S holds the public dataset $D_p = \{d_p, y_p\}$ and uses it to train the transfer model M_p. Without loss of generality, in the description of our proposal and baseline experiments, we focus on an image classification task, making Convolutional Neural Networks (CNNs) an intuitive architectural choice. Even though the description of the scenario will reference the image domain, the approach can be generalized using any explainable AI tool, as demonstrated with text data, where we applied SHAP values for the XAI technique. In practice, the server performs a global feature learning task that will then be shared by all the clients. The weights W_p of the model M_p are trained as follows (CE = cross-entropy):

$$preds = M_p(d_p); l = CE(preds,\ y_p);\ \nabla W_p = \sum \frac{\partial l}{\partial M_p}. \tag{1}$$

The trained model M_p can now be propagated to the clients $C = \{c_1, \ldots, c_n\}$. In particular, each client knows the public dataset D_p and holds a private dataset D_i (where $i \in [1, n]$) that cannot be accessed by the others. During the FL process, the lower levels of the model, in our case, the Convolutional layers $Conv$, are frozen to preserve the network's ability to extract low-level and more generic features of the image, such as edges. The classification layers fc, instead, are the most capable of capturing the high-level features. Therefore, the fc layers are trained by the clients using their local datasets D_i to customize the classification behavior based on the domain from which their private data are extracted. The training of fc is achieved by optimizing the combination of two different loss functions. The first one is a standard cross-entropy loss between the prediction on D_i and the corresponding labels. The second one is an alignment loss on the last layers of fc to better personalize the model on the client's data. As described in Chen et al. [9], this alignment loss function is intended to align the output features between the public and private inputs. In this case, the alignment is performed between the outputs of the model M_p on the public dataset D_p and the private dataset D_i. The loss function is calculated as follows:

$$S_{img} = M_p(D_p);\ T_{img} = M_p(D_i) \tag{2}$$

$$l_{CORAL} = \frac{1}{4d^2} || S_{img} - T_{img} ||^2_F \tag{3}$$

$$tot\_l = CE(M_p(D_i), y_i) + \alpha l_{CORAL}. \tag{4}$$

CORrelation ALignment (CORAL) aligns the feature distributions of the source and target domains in an unsupervised fashion. The original approach focuses on matching these distributions through the alignment of second-order statistics, specifically the covariance [45]. Here, $||.||^2_F$ is the squared matrix Frobenius norm, d is the dimension of the embedding features, and α is the trade-off parameter between the loss functions.

This alignment allows each client to obtain a personalized model M_i, which is more accurate for the local data. We assume a federated transfer learning deployment in which a pretrained feature extractor and a task-specific head f_c enable local adaptation on private client data. Because training occurs on the client device, a malicious participant can observe the optimization process and access gradients and intermediate feature representations. Extending the analysis to deployments where access to internal activations is restricted (e.g., protected runtimes or server-side feature extractors) remains an interesting direction for future investigation. The updated local weights are then sent to the server for aggregation, like traditional HFL. Keeping the *Conv* layers frozen, along with the availability of public data used to train the initial model, opens the framework to possible threats from a malicious client. Our intuition is to exploit the transfer model to craft dynamic triggers that embed the features of the target class. Since the *Conv* layers work as feature extractors in this scenario (image classification), they will focus only on the features of the images that are known from the pre-training of the network on the public data. For this reason, we can reasonably expect (and we later confirm experimentally) that the *Conv* layers will discard traditional triggers that add additional features unknown to the network. To craft our dynamic trigger, we need to distill into it the low-level and general features leaked directly from M_p. Since the transfer model M_p and the public dataset D_p are available to all the clients by design, an attacker can exploit them to craft triggers by leveraging features from D_p known to the *Conv* layers of the target model. However, distilling a dynamic trigger is still not enough to drive the transfer model M_p towards the target class, as we show experimentally in Section 5.5. In this scenario, the positioning of the trigger plays a key role in the attack's success. In our attack, we propose a strategy that aims to detect and override the main features of the victim image with a dynamic trigger containing compressed features of the target class. Empowered by the generated triggers, we assume that the attacker controls a minority percentage of the clients to poison the model (Fig. 1).

4.2. Threat model

Attacker Knowledge: the adversary has access to the public dataset D_p, the public model M_p provided by the server, and a private dataset D_i. Additionally, the adversary has access to the gradients and the feature maps of the model's layers.

Attacker Capabilities: using the private dataset D_i, the adversary can fine-tune the public model locally. However, the model's shallow layers, i.e., the convolutional and max-pooling layers, are frozen, so only the fully connected layers can be altered.

Attacker Goal: by altering samples from the private dataset D_i, the attacker aims to inject a backdoor into the local model M_i. This backdoor cannot depend on the frozen layers, so a sophisticated trigger design is needed. After the backdoor is injected into M_i, the server will aggregate the local updates from the clients and update the unfrozen layers of the public model M_p. This updated model will be sent back to the clients for further fine-tuning. The adversary's end goal is, hence, to embed a backdoor that remains active after the completion of the FTL. The backdoor is then used to force misclassifications of malicious inputs to the target class. We note that this represents a strong, worst-case attacker knowledge scenario, intended to provide an upper bound on potential attack effectiveness.

4.3. FB-FTL attack

As discussed in Section 4.1, the frozen convolutional layers *Conv* can extract generic low-level features from images that can be inserted as triggers in a victim image to perform a backdoor attack. Our approach, shown in Fig. 2, does not merely add a trigger in a fixed position into the image. Instead, it overrides the features of the original class with those of the target class. To do so, we use GradCam [42], which identifies the important regions in a target image for the given model. Any XAI tool capable of identifying the key features of the data can be used; in this case, we utilize GradCam as our reference tool without loss of generality. GradCam obtains the class-discriminative localization map for any class c by computing the gradients for the given score of the class in the output y^c with respect to the feature maps A^k of a convolutional layer. The obtained gradients are then global-average-pooled to obtain the neuron importance weights a_k^c as follows: $a_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$.

These weights represent the importance of the features in the feature map for a target class c. After this, a weighted combination of forward activation maps is performed and sent as input to a *ReLU* activation function to obtain a heatmap of importance I_m with the same shape as the feature maps: $I_m = ReLU\left(\sum_k a_k^c A^k\right)$.

ReLU is applied to the linear combination of maps because we only consider the features with a positive impact on the class of interest. Pixels with negative values in the GradCAM heatmap are likely to belong to other classes [42]. Without *ReLU*, localization maps could emphasize more than just the desired class region, causing a lower localization performance. The obtained map I_m can be used as a mask to select the

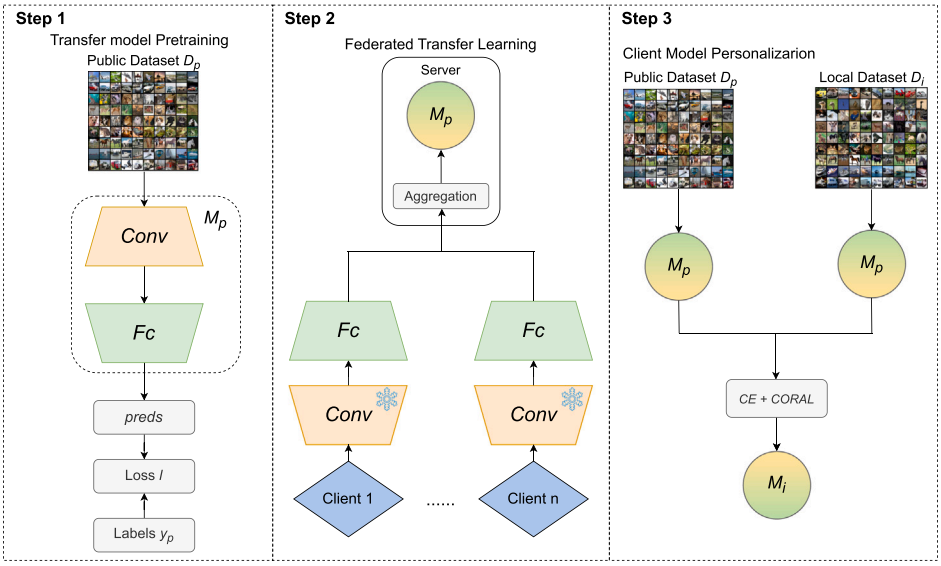


Fig. 1. Federated Transfer Learning Framework.

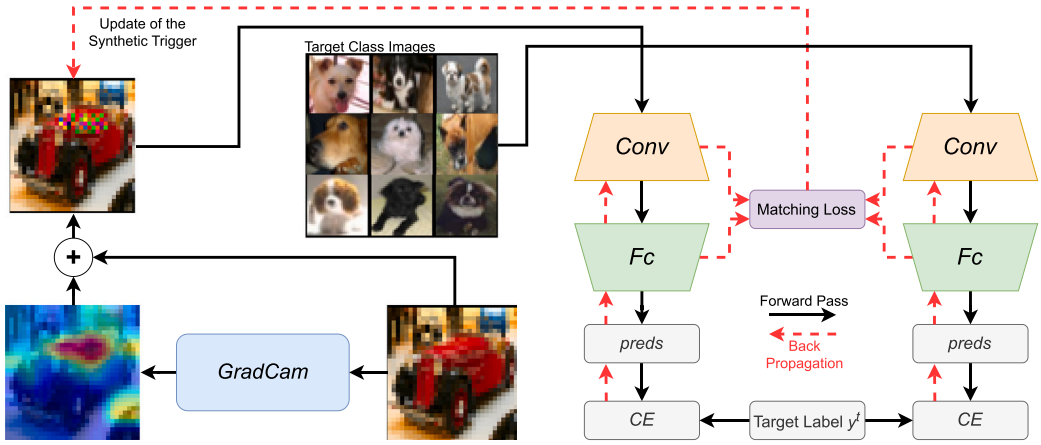


Fig. 2. Trigger distillation.

regions where the trigger should be injected. In particular, we select the regions where the weights of importance a_k^c are higher than a given threshold τ. Since the weights of importance a_k^c fall in the $[0, 1]$ interval, τ can be picked in the same interval. To inject a trigger that overrides the original image's features with the main features of the target class, we use dataset distillation [55]. Specifically, we generate the synthetic trigger through Dataset Condensation, which involves gradient matching to optimize a smaller, condensed dataset that captures the essential information from the original data and, in this case, the target class. We select a subset of images B_i from classes different from the target for which we generate the GradCam heatmaps of the same size as the images. Then, we set to 0 the regions of the heatmaps lower than the given threshold τ and to 1 the zones with a higher value, thus generating a mask. The obtained mask can now be used to inject a trainable synthetic trigger initialized as random noise into the best location of the target images. Our approach proceeds by exploiting the pre-trained transfer model M_p, for which we keep the parameters frozen for the entire process to learn the correct trigger. At each iteration, we feed at first the images B_i, and we obtain the gradients ∇W_p^B on the model concerning the target class y^t using the cross-entropy loss. In the second step, we repeat the process with a set of true images T_i of the target class from the public dataset used to pre-train the model M_p. In the same way, we generate the related gradients ∇W_p^T on the parameters of M_p. The idea is to generate the trigger by embedding the low-level general features from the target class known to the pre-trained $Conv$ feature extractor of the transfer model. Using the cosine similarity, we calculate the loss on the distance between the generated gradients ∇W_p^B and ∇W_p^T, and we backpropagate on the trainable trigger injected in B_i as follows:

$$\nabla W_p^B \leftarrow CE(M_p(B_i), y^t); \nabla W_p^T \leftarrow CE(M_p(T_i), y^t) \tag{5}$$

$$B_i[I_m > \tau] \leftarrow COS(\nabla W_p^B, \nabla W_p^T). \tag{6}$$

The process is then repeated for multiple iterations.

The obtained images with the trigger are then included in the local training of the malicious clients with benign data to preserve the performance of the global model on the main task. The complete step-by-step generation procedure is presented in Algorithm 1. Similarly to training, in inference, the attacker can generate backdoored samples using the local model and the public dataset (D_p). These samples are used to effectively manipulate the output of the victim models. This does not violate our threat model as the attacker can use the local model and the public dataset (D_p) without any restrictions. Our attack introduces zero communication overhead, as the malicious client's updates remain identical in size to benign updates. While synthesizing the trigger requires local optimization, this process is executed entirely offline prior to participating in the federated training. Consequently, it introduces no latency to the synchronous FL rounds. We emphasize that this process is strictly isolated from the evaluation data. No information, labels, or statistical derivations from the validation or test sets are accessed or utilized at any stage of trigger generation, positioning, or local optimization. The adversary relies exclusively on the allowed public data and the pre-trained model.

Algorithm 1 Trigger generation via gradient matching.

Input: Pretrained model M_p, public dataset D_p, target class y^t, threshold τ

Output: Optimized trigger δ

Initialize trigger δ (random or from image)
for each iteration **do**
  Sample batch B_i from non-target classes in D_p
  Sample batch T_i from target class y^t in D_p
  Compute GradCAM mask I_m on B_i
  Generate binary mask $M = \mathbb{1}(I_m > \tau)$
  Inject trigger: $\tilde{B}_i = B_i \odot (1 - M) + \delta \odot M$
  $\nabla W_p^B = \nabla_W \mathcal{L}(M_p(\tilde{B}_i), y^t)$
  $\nabla W_p^T = \nabla_W \mathcal{L}(M_p(T_i), y^t)$
  $\mathcal{L}_{GM} = 1 - \cos(\nabla W_p^B, \nabla W_p^T)$
  Update δ using $\nabla \mathcal{L}_{GM}$
end for
return δ

4.3.1. Blended trigger

Our intuition is to override the main features of an image, allowing its classification towards the target class with a distilled trigger. This strategy would inevitably alter the original image and, most probably, the main visual content, as shown in Fig. B.10. This alteration can be detected as an outlier using a mechanism based on the clean data distribution. Furthermore, when the perturbation becomes too strong, the attack may resemble targeted data poisoning. Therefore, we also consider perceptual constraints to ensure that the trigger preserves the main semantic content of the image while remaining effective. To synthesize a trigger that blends better inside the image, making it less noticeable to automatic detection systems or the human eye, we can include a term derived from similarity metrics capable of detecting such alterations in the distillation matching loss. Concerning such similarity metrics, one well-fitting option is the Learned Perceptual Image Patch Similarity $LPIPS$ [54]. We follow the idea of using the $LPIPS$ from the Style Transfer and Generative Images approaches [19,30]. These approaches employ an $LPIPS$ loss due to its ability to provide a better feature space

Algorithm 2 Blended trigger generation via gradient matching.

Input: Pretrained model M_p, public dataset D_p, target class y^t, threshold τ
Output: Optimized trigger δ

Initialize trigger δ (random or from image)
for each iteration **do**
  Sample batch B_i from non-target classes in D_p
  O_i is a copy of B_i
  Sample batch T_i from target class y^t in D_p
  Compute GradCAM mask I_m on B_i
  Generate binary mask $M = \mathbb{1}(I_m > \tau)$
  Inject trigger: $\tilde{B}_i = B_i \odot (1 - M) + \delta \odot M$
  $\nabla W_p^B = \nabla_W \mathcal{L}(M_p(\tilde{B}_i), y^t)$
  $\nabla W_p^T = \nabla_W \mathcal{L}(M_p(T_i), y^t)$
  $\mathcal{L}_{GM} = COS(\nabla W_p^B, \nabla W_p^T) + \lambda LPIPS(B_i, O_i)$
  Update δ using $\nabla \mathcal{L}_{GM}$
end for
return δ

than other similar solutions to ensure the perceptual quality of the resulting image. In detail, using a pre-defined network like VGG or AlexNet, $LPIPS$ compares the activations of two image patches to compute similarity. This measure matches the real perception of humans [54]. A low LPIPS score means that the two images are perceptually similar. So, our trigger distillation loss can be changed as follows:

$$B_i[I_m > \tau] \leftarrow COS(\nabla W_p^B, \nabla W_p^T) + \lambda LPIPS(B_i, O_i), \tag{7}$$

where λ is the weight of the $LPIPS$ contribution to the overall loss, and O_i is the original image without the trigger. The complete step-by-step generation procedure is presented in Algorithm 2.

In our implementation, the trigger's pixels are reinitialized as random Gaussian noise in the original image O_i. A possible variation to improve the $LPIPS$ value is to back-propagate the combination of the matching loss and $LPIPS$ to the original pixels of the images without masking with random noise. In this case, the process changes as follows:

$$BO_i[I_m > \tau] \leftarrow COS(\nabla W_p^B, \nabla W_p^T) + \lambda LPIPS(BO_i, O_i), \tag{8}$$

where BO_i is the backdoored image with the trigger initialized with the same values as the original one. The approach is unchanged, except for the initialization of the trigger's pixels. In Section 5.6, we present the analysis of our approach for different combinations of losses and initialization trigger strategies. The $LPIPS$ contribution is specifically designed for image data. In contrast, it is unnecessary for text data since the information derived from the pre-trained model produces existing words that do not perceptibly alter the original data.

5. Experimental results

5.1. Datasets

To conduct the experimental campaign, we selected four common benchmark datasets: CIFAR10 [23], CINIC10 [10], SVHN [34], and GTSRB [44]. The first two are datasets that include classes of different animals or objects. Both share the same categories. The main difference is that CINIC10 is 4.5 times larger than CIFAR10. This difference in size allows us to demonstrate that our approach is still effective even with larger datasets, proving our intuition that the trigger is distilled, preserving the most general and meaningful features of the target class. The SVHN dataset, instead, contains images of houses' civic numbers with categories from 0 to 9. One of the main characteristics of this dataset is the presence of additional numbers in the background that do not belong to the assigned category. We selected this dataset to see if the features distilled in our trigger are strong enough to be preferred over the other numbers in the image by the model. The last dataset, the German Traffic Sign Recognition Benchmark (GTSRB), contains 43 different categories of traffic signs. Since the traffic signs have many features in common between the categories (e.g., shape and color), we included this dataset to test whether our approach can override the right features to control the model and predict the target class. Table 1 reports statistics for the considered datasets. In Table 2, we report the statistics of the text datasets used in Section 5.7. In particular, the CARER [40] and IMDB [31] datasets.

Table 1
Statistics of the considered image datasets.

Dataset	Train Set Size	Test Set Size	Number of Classes
CIFAR10	50,000	10,000	10
CINIC10	180,000	90,000	10
SVHN	73,257	26,032	10
GTSRB	39,209	12,630	43

Table 2
Statistics of the considered text datasets.

Dataset	Train Set Size	Test Set Size	Number of Classes
CARER	16,000	2000	6
IMDB	25,000	25,000	2

Table 3
Statistics of the considered MedMNIST datasets.

Dataset	Train Set Size	Test Set Size	Number of Classes
PathMNIST	89,996	7180	9
OCTMNIST	97,477	1000	4
TissueMNIST	165,466	47,280	8

As reported in Section 2, medical applications are a key use case for Federated Transfer Learning. To further evaluate our attack in this domain, we consider a subset of datasets from the MedMNIST collection [50]. MedMNIST is a standardized benchmark of biomedical images spanning multiple modalities and classification tasks. The selected datasets, reported in Table 3, differ in terms of dataset size, number of classes, and visual complexity. PathMNIST contains histopathology images for colorectal cancer classification, characterized by fine-grained texture patterns and moderate class separability. OCTMNIST consists of retinal OCT images, where structural patterns are more pronounced, potentially making feature extraction more stable. TissueMNIST includes segmented kidney tissue images with high intra-class variability and subtle inter-class differences, representing a more challenging classification scenario. These datasets allow us to assess how domain-specific complexity, such as fine-grained visual patterns and class overlap, impacts the effectiveness and stealthiness of the attack. For this set of experiments, we consider an IID scenario to isolate the effect of dataset characteristics on the attack. While non-IID settings introduce additional challenges such as class imbalance and heterogeneous client distributions, these factors primarily affect model convergence rather than the attack mechanism itself [1,21,25,33]. Our method only requires access to samples from the target class, making its effectiveness largely independent of the underlying data distribution.

In particular, the first, PathMNIST, contains medical images for predicting survival from colorectal cancer histology slides. The second, OCTMNIST, comprises optical coherence tomography (OCT) images for retinal diseases. The third, TissueMNIST, contains human kidney cortex cells segmented from 3 reference tissue specimens and organized into 8 categories.

5.2. Experimental setup

In our experiments, we considered an FTL framework composed of one server and multiple clients. In particular, in the baseline experiments and the experiments on text data in Section 5.7, we use 10 clients to

Table 4
The main results were obtained with 10% of malicious clients out of a total of 10 clients with a poisoning rate of 1%. This setup allowed us to test FB-FTL against the most challenging attack scenarios for comparison using the FoolsGold defense as a baseline.

Backdoor Attacks	CIFAR10		CINIC10		SVHN		GTSRB	
	Model Accuracy	ASR	Model Accuracy	ASR	Model Accuracy	ASR	Model Accuracy	ASR
No Attack	81.8%	–	71.0%	–	90.0%	–	95.0%	–
Pattern Trigger	77.9%	$2.60\% \pm 0.61$	67.4%	$2.55\% \pm 1.09$	87.4%	$1.27\% \pm 0.27$	93.4%	$4.7\% \pm 0.46$
Square Trigger	75.4%	$1.04\% \pm 0.09$	67.7%	$2.82\% \pm 2.30$	87.3%	$1.16\% \pm 0.40$	92.6%	$0.53\% \pm 0.26$
Watermark Trigger	76.5%	$2.47\% \pm 1.31$	67.8%	$1.74\% \pm 0.85$	88.3%	$1.38\% \pm 0.60$	93.9%	$0.67\% \pm 0.46$
DBA [48]	77.5%	$1.06\% \pm 0.19$	65.8%	$4.76\% \pm 1.77$	87.8%	$1.21\% \pm 0.31$	95.3%	$1.13\% \pm 2.05$
A3FL [53]	73.1%	$31.2\% \pm 0.34$	65.5%	$24.3\% \pm 1.82$	87.0%	$10.0\% \pm 2.04$	94.0%	$10.1\% \pm 1.77$
FB-FTL (ours)	81.3%	$91.1\% \pm 2.34$	70.2%	$71.1\% \pm 2.39$	89.6%	$76.6\% \pm 1.69$	94.7%	$68.5\% \pm 2.20$

reproduce the same real-world setting described in Section 2. We considered a cross-silo scenario with a small number of institutions. The percentage of attackers is set to a default value of 10%. The poisoning rate is defined per malicious client, so each malicious client poisons a fixed fraction of its local dataset, with poisoned samples kept fixed during a run, while triggers are generated independently and target labels vary across experiments to assess generalizability. In addition, we employ FoolsGold as a baseline aggregator, which is also included in the study on defenses presented in Section 7 where a more detailed discussion on the transferability of existing defenses from HFL and VFL is presented. We present additional results by changing the number of clients while preserving the more realistic cross-silo scenario, in particular, $\{5, 10, 100\}$. The main task is image classification. The statistics for all the datasets are reported in Section 5.1. The considered baseline transfer model used as the feature extractor is a *ResNet-18* [15] with an MLP classifier on top, detailed in Table A.12.

To showcase the generalizability of our attack, we also tested it with two additional architectures: ConvNet [12] and VGG16 [43] in Section 6. The dataset split between train and test sets is left at default if available; otherwise, we considered the 80%/20% split for training/testing. From the training set, 50% is preserved as simulated public data to pre-train the model, and the remaining half is split randomly across the clients. We simulate the public dataset by splitting the original dataset so that the auxiliary data shares the same label space and a similar distribution as the clients' private data. While this controlled setup enables a clear evaluation of the protocol's privacy risks, we additionally consider non-IID client data distributions to partially account for realistic heterogeneity across participants. The optimal performance on pre-training of the model has been obtained using the public data for 10 epochs with a learning rate of 0.1 for the first five epochs and 0.01 for the remaining five. As mentioned earlier, since the public datasets are limited, the performance of pretraining remains suboptimal compared to the full FTL training, which achieves an average improvement of 10% to 20% across the datasets.

During the FL process, the classification layers are trained with a learning rate of 0.1, which allows us to obtain the best results in the main task performance. The matching loss employed to distill the trigger is the *Cosine Similarity* as presented in Section 4.3. The percentage of poisoned images is around 1% of the entire training set.

We compare our attack with three HFL static patch backdoor attacks: pixel pattern trigger, square trigger, watermark trigger, and a distributed and advanced version of the pattern trigger that changes in shape and position as described in Xie et al. [48]. Finally, we compare our attack against a novel approach that uses a trainable trigger: A3FL [53].

We tested our attack with different percentages of attackers among the clients ($\{10\%, 20\%, 30\%, 40\%, 50\%\}$) to see if the effectiveness of our attack depends on the number of malicious clients. In Section 5.5, we test the importance of GradCam's usage to identify the trigger's correct position compared to a square-generated trigger distilled in the same way as ours, but with fixed coordinates in the corner of the image. In Section 5.10, we present the attack's success rate with the optimal value of the importance threshold τ chosen from the set of possible values, specifically $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. All the reported results are averaged over multiple runs. In Section 5.6, we redefine the learning loss of the dynamic trigger, combining the original matching loss with a similarity loss that better preserves the characteristics of the original image, making the trigger less noticeable while simultaneously maintaining most of the backdoor attack accuracy. In particular, we selected the $LPIPS$ loss function which measures the perceptual difference between the original images and their manipulated counterparts with the injected triggers.

For our experiments, we used a machine with an AMD Ryzen 5800X CPU paired with 32GB of RAM and an RTX 3070ti with 8GB of VRAM. Our attack has been developed using PyTorch 2.0.

5.3. Performance evaluation of FB-FTL

We present the main results comparing the success rate of our FB-FTL attack to existing methods, aiming to assess backdoor attack viability in FTL. As no FTL-specific attacks exist, we compare FB-FTL to HFL attacks due to their similar server–client training setup, where clients train locally, and the server aggregates. Alongside static triggers, we evaluate A3FL [53], which uses trainable triggers. However, since FTL freezes the feature extractor, traditional attacks are expected to underperform, as their triggers rely on features unseen by the frozen layers. While A3FL may outperform static triggers, it should still fall short of FB-FTL's performance due to its lack of our XAI-based focus mechanism.

As expected, the existing attacks are almost unperceived by the federated model (Table 4). A3FL achieves a better ASR than static triggers while still affecting the model accuracy on the main task, like the previous three. These results also provide some insights into the importance of trigger positioning using GradCam. In Section 5.5, we present an in-depth analysis in this regard. FB-FTL is effective across the considered datasets. It can achieve high ASR despite the different characteristics of the datasets (details in Section 5.1). Even in datasets such as SVHN, where the background contains competing information from other classes, our attack is able to distill a trigger that emphasizes the most relevant features of the target class, effectively dominating over distracting patterns present in the image. Similarly, GTSRB presents a challenging setting due to the high visual similarity across traffic sign categories; nevertheless, the proposed method successfully identifies and encodes the subtle discriminative features required to steer the model toward the target class. Overall, our approach remains highly effective across these diverse scenarios while only minimally impacting the performance on the main task. This is because the feature extractor of the model recognizes only the known features. With traditional backdoor attacks, the model will still focus just on the information of the original class, discarding the trigger that is even poisoning the entire training dataset, inevitably affecting the accuracy of the main task. The attacker is training the model, assigning to these images the target class, drifting the accuracy of the model on those features, and affecting the final accuracy. In the baseline attacks, we do not override (in general) features of the original class, but we still alter the labels of the altered images. As a result, the model may associate some of the features of the original class

Table 5
Results on different non-IID settings using Dirichlet distributions.

Dirichlet Alpha	CIFAR10		CINIC10		SVHN		GTSRB	
	Model Accuracy	ASR	Model Accuracy	ASR	Model Accuracy	ASR	Model Accuracy	ASR
0.2	77.1%	88.5% ± 8.61	62.4%	76.5% ± 8.21	74.2%	85% ± 7.75	92.5%	76.7% ± 12.95
0.3	78.3%	86.6% ± 6.51	68.5%	75.6% ± 8.13	82.2%	80.0% ± 4.43	93.2%	73.3% ± 12.33
0.4	80.1%	85, 5% ± 5.12	68.7%	71.1% ± 5.45	85.7%	76.6% ± 3.43	94.1%	71.1% ± 10.25
0.5	81.0%	81.1% ± 4.02	69.6%	71.1% ± 4.69	89.2%	74.4% ± 2.73	94.9%	68.9% ± 9.26

with the target class, which may lower the model's accuracy on clean inputs, causing a label-flipping attack that lowers accuracy. In our case, instead, we are substituting the original features with information from the target class distilled from the network itself. In this way, the network will recognize the new features as known, only slightly affecting the accuracy of the main model on clean inputs.

In addition to the default IID setting, we tested our attack in several non-IID settings obtained by leveraging Dirichlet distributions with different α values to spread the classes of the training set across clients. This experiment shows that the heterogeneous distribution of classes affects our attack. From Table 5, we see that lower α values (more heterogeneity) increase the ASR compared to configurations with higher values. This is expected due to the lower availability of the target class across the clients, and, as a consequence, the contribution of the attacker to the target class increases, as it can always generate poisoned data due to the availability of the public dataset. As shown in Table 5, while ASR remains high across runs, the standard deviation is larger in non-IID settings, reflecting the impact of class imbalance and uneven sample distributions on attack effectiveness.

To further verify the persistence of our attack even without the inclusion of the malicious clients in the training, we considered a scenario in which the attacker is included just in the first 10 epochs of the federated training, and we recorded the accuracy of the model after an additional 10 epochs without any malicious clients. From this experiment, on average, we noticed a drop of 5% in ASR, maintaining an ASR similar to the case in which malicious clients are consistently involved. This finding demonstrates that FB-FTL largely depends on the FTL assumption; as our attack exploits features learned from the frozen feature extractor, the backdoor can remain effective even if the attacker is not involved in the whole training process. In Appendix B, we show triggers generated for different classes across the considered datasets. The position of the trigger changes according to the region of importance spotted by GradCam. In Section 5.5, we provide an ablation study that demonstrates the importance of finding the best region of the image where to distill the trigger instead of having fixed coordinates. On the other hand, this approach also owes its success to the overriding of the main features of the image using GradCam to position the trigger. This makes inevitable, in most images, an alteration of the main subject of the image. In Section 5.6, we present the results after adding $LPIPS$ in the loss calculation, which makes the distilled trigger less noticeable to humans.

5.4. Evaluating the percentage of attackers

As detailed in Section 5.2, previous results were obtained with 10% of malicious clients. Given that our attack primarily leverages features already captured by the frozen *Conv* layers, we expect its success rate to remain largely consistent. This experiment aims to confirm that the effectiveness of our attack is only partially influenced by the number of attackers.

As shown in Fig. 3, with three out of four datasets, we can confirm our intuition. Indeed, the ASR is comparable across the different scenarios with negligible fluctuations. GTSRB is the only dataset that significantly benefits from a higher proportion of malicious clients. When the percentage of attackers reaches 50%, the ASR approaches 100%. In the most constrained setting, the ASR decreases slightly compared to the baseline but remains non-negligible, at around 70%. This behavior is

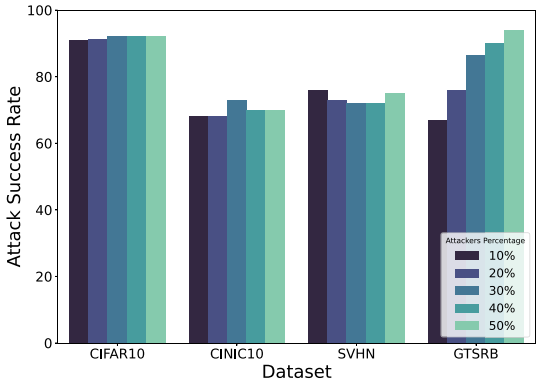


Fig. 3. Performance of the attack while changing the percentage of attackers.

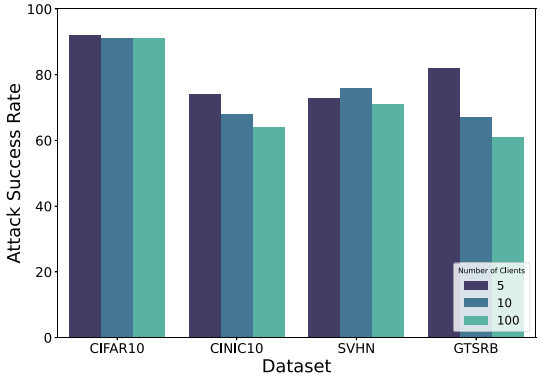


Fig. 4. Performance of the attack changing the total number of clients with one attacking client.

consistent with the characteristics of GTSRB, where traffic sign classes share many visual features, such as color and shape, leading to high inter-class similarity. As a result, the features distilled into the trigger are less distinctive, making it more challenging for the attack to consistently dominate the model's predictions. Consequently, a higher degree of coordination among malicious clients becomes important to reinforce the target signal and achieve high attack success rates.

The previous experiment was conducted with 10 clients. Next, we conduct a second experiment, varying the number of clients but using only one attacker, to investigate the robustness of our attack for different percentages of adversaries. In the first scenario, we considered half of the clients compared to the baseline (5), and the second scenario includes ten times more clients (100). Since our attack relies on the frozen feature extractor, we expect that our attack is only partially dependent on the number of clients.

The results of this experiment are shown in Fig. 4. As anticipated, the difference between 10 and 100 is just 5% at maximum, preserving an ASR over 60% across all the datasets. Instead, looking at the scenario with 5, we can see how the performance is better than considering 10. This is expected since with 5 clients, in percentage, the contribution of

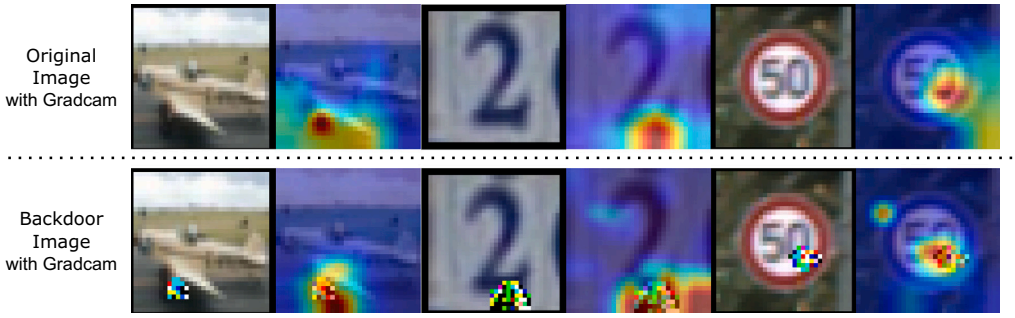


Fig. 5. Examples of GradCam maps with and without the trigger.

the single attacker is higher compared to 10 overall clients, but similarly to the previous use case, the difference is almost negligible. The highest recorded difference is about 15% for the GTSRB dataset with 5 total clients compared to the baseline of 10. Considering just one malicious client, even if the proportion between malicious and benign clients is doubled when using 5 clients instead of 10, the ASR is not doubled. These results confirm that our attack is only slightly affected by the percentage of attackers among the FL participants.

5.5. XAI importance analysis

As presented in Section 4.3, XAI is one of the main components of our approach. Trainable triggers alone, as shown in Table 4, are not enough to produce backdoor attacks effective in the FTL scenario. In this section, we present an analysis to assess the importance of an XAI technique, such as GradCam, to spot the most important features of images to be covered by our distilled trigger.

In Fig. 5, we present examples of the GradCam results on the original set of images and their backdoored versions. As we can see from the heatmaps produced by GradCam on the original image and its poisoned version, the model focuses on the same region to classify the image. This shows that our trigger successfully substitutes the main features of the original class as the most important portion of the image for the model to classify it. Another interesting result is the area of importance detected by the GradCam algorithm. The different shades of colors presented in Fig. 5 represent the importance of those features for the classification of the image. The red and yellow regions are the ones that matter the most for the model's classification. Looking at the examples in Fig. 5, we see that the produced heatmaps on the original images highlight the most important region in red but also show how the model looks at regions around it in yellow to make a decision for the classification of the image. The results produced on the backdoored images, instead, show an important region that is much more concentrated around the trigger. This shows that the combination of the distilled trigger and explainability can confidently drive the target model toward the target class with more confidence compared to the original class.

To better substantiate our intuition, we performed an ablation study to test the different components of our approach in three different configurations. In the first scenario, the trigger is distilled in a fixed position like traditional solutions (Generated Patch). In the second case, the trigger is still distilled in fixed coordinates, but unlike in the previous scenario, we use GradCam to detect the most important features of the images and override them with random noise (Generated Patch with Obfuscation). The final scenario, instead, is our full attack (GradCam for positioning and distillation of the features from the target class to generate the trigger). In Table 6, we report our ablation study results. As expected, the trigger in a fixed position is not capable of producing an effective backdoor attack, confirming what we have already reported in Section 5.3. An interesting result can be observed in the scenario where we obfuscate the features of the original class. As we can see, ASR in this scenario improves compared to the one in which we simply distill the trigger in a fixed position of the image. Especially for the SVHN dataset, the performance of the attack improves three times more after masking the original information of the images. This result demonstrates that covering the main features with a distilled trigger in the right position contributes to deceiving the frozen transfer model.

Table 6
Experiments with generated patches in a fixed position with and without obfuscating the main features and a poisoning rate of 30%.

Dataset	Generated Corner Patch	Generated Corner Patch With Obfuscation	Our Trigger
CIFAR10	32.3%	40.0%	91.1%
CINIC10	22.3%	26.7%	73.3%
SVHN	7.8%	23.4%	72.1%
GTSRB	18.9%	33.3%	86.5%

5.6. Experiment on perceptual similarity loss

As introduced in our methodology (Section 4.3), the positioning of our trigger over the main features of the images can alter too much of the original data, making it detectable even by automatic systems that use the distribution of clean data and discard outliers. This section is devoted to understanding how the changes in the loss and trigger initialization described in Section 4.3.1 impact the perceptual similarity between the original and backdoored images using the $LPIPS$ metric. This is noticeable in Table 10 in Section 5.9. With the conservative loss, we aim to preserve the attack performance, while reducing the $LPIPS$. We tested the success rate of our approach in normal conditions compared to the more visually conservative version of it. In particular, we tested the modified version of the original loss by adding the $LPIPS$ component combined with random initialization of the trigger and initialization with the original values of the target image. The results are presented in Table 7. With the more conservative loss, we lower the $LPIPS$ values while preserving most of the performance of the original attack. Looking at the SVHN dataset, it is noteworthy that the $LPIPS$-based loss, combined with trigger initialization from the original image values, outperforms both random initialization and the standard version of the attack. This behavior can be explained by the dataset's characteristics, as discussed in Section 5.1. In SVHN, images often contain multiple digits, with background elements that may belong to different classes. Introducing a trigger that is too dissimilar from the original image may amplify these competing features, leading the model to focus on irrelevant patterns. In contrast, a more blended and perceptually aligned trigger integrates better with the image content, allowing the attack to more effectively guide the model toward the target class.

In Appendix B, we show examples of triggers generated with different combinations of losses and initializations. We can notice that adding $LPIPS$ to the basic loss allows the generation of triggers that blend better with the overall image. The colors of the pixels of the triggers better

Table 7
Attack success rate (ASR) and $LPIPS$ value changing the loss and trigger initialization.

Dataset	Normal/Noise		$LPIPS$/Noise		$LPIPS$/Original	
	ASR	$LPIPS$	ASR	$LPIPS$	ASR	$LPIPS$
CIFAR10	91.1%	0.1346	84.4%	0.0308	82.2%	0.0294
CINIC10	73.3%	0.0618	70.0%	0.0087	67.8%	0.0076
SVHN	72.1%	0.2903	70.0%	0.1036	76.7%	0.0919
GTSRB	86.5%	0.1346	81.1%	0.0375	78.7%	0.0363

Table 8
Evaluation on text data.

Distribution	Metric	CARER	IMDB
IID	Model Accuracy	93.7%	87.5%
	Attack Success Rate	93.4% ± 0.01	71.4% ± 5.89
non-IID (Alpha = 0.2)	Model Accuracy	88.5%	87.3%
	Attack Success Rate	100% ± 0.00	71.2% ± 3.65

match the palette and the intensity of the colors of the original image compared to the basic implementation of the generated triggers. Looking at Fig. B.12, the initialization of the trigger plays an important part. As we can see, the generated trigger blends better with the main subject of the image, preserving some of the original pixels and resulting in an almost transparent trigger. The initialization strategy makes the trigger even less noticeable to the human eye compared to other backdoor attacks in which the trigger is easily detectable by a human check.

5.7. Evaluation with text data

Due to the shift in the target domain, various elements that form our attack must be adjusted to fit the target domain while maintaining our core intuition. This intuition involves the combination of XAI to identify the optimal trigger location and crafting the trigger itself, which must be extracted from the pre-trained model's original knowledge. Since GradCam primarily addresses the vision domain, we chose SHAP as an XAI tool to identify the key features (words) in textual data. Like GradCam, SHAP has been employed to highlight the most impactful words within a sentence for the pre-trained model. Analogous to the approach used for images, we use this information to find the best spot to position our trigger. The selected words are then substituted with a trigger derived from refining the knowledge of a pre-trained transfer model. In this experiment, we tested text datasets CARER and IMDB, presented in Section 5.1, under the IID and non-IID scenarios [25], using for the latter a Dirichlet distribution with α set to 0.2 to distribute the classes across clients and BERT [22] as the reference model. The findings from this experiment are summarized in Table 8. We observe that even with the shift in the domain, our intuition remains valid, highlighting both the independence from the specific application of GradCam and demonstrating its applicability across various domains. Hence, this experiment supports our hypothesis that combining XAI techniques with knowledge distillation allows us to exploit the static feature extractor of FTL to craft a powerful attack.

5.8. Evaluation on real-world medical data

As discussed in Section 2, the healthcare scenario is the primary use case for FTL, as demonstrated also by the company Owkin. To evaluate the behavior of our attack in a medical imaging context, we conducted experiments using datasets from MedMNIST, namely PathMNIST, OCTMNIST, and TissueMNIST, following the same experimental setup used in the baseline evaluation described in Section 5.3. MedMNIST is a standardized public benchmark composed of curated biomedical images designed for patient-level classification tasks across multiple medical imaging modalities. While it does not fully capture the complexities of real cross-institution federated deployments, it offers a controlled setting to study the behavior of attacks in a medically relevant domain. Like the GTSRB dataset, the considered set of data is characterized by classes with highly overlapping features due to the few differences that distinguish the samples of different classes. In the same way, it makes it challenging to generate effective triggers. We conducted the experiments by generating the triggers using the different versions of the loss as experimented in Section 5.6 and the same experimental settings used for the baseline scenario. The results are reported in Table 9.

Our attack obtained results that further highlight the security vulnerabilities of FTL with an ASR of up to 96% for the OCTMNIST dataset. Examining PathMNIST, it is noteworthy that a less noticeable trigger has a smaller effect on the accuracy of the primary task. A similar trend is observed with OCTMNIST, where the contribution of $LPIPS$ better preserves the content of the original image, modifying only the most essential features required to make the attack effective, resulting in higher accuracy on the main task compared to the "normal" attack. Looking at TissueMNIST, instead, it is interesting to see how the less visible trigger, initialized with original image pixels, performs slightly better than its counterpart initialized randomly. In general, as expected, using the versions of the loss function that produce stealthier triggers, preserving most of the original features, sensibly decreases the ASR yet still achieves a noteworthy level of success with minimal impact on the overall system. These results show that the proposed attack remains effective on medical imaging benchmarks, highlighting potential security risks for FTL deployments in healthcare-related applications.

5.9. Perceptual similarity loss of existing triggers

As shown in Table 10, the positioning of our trigger over the original content of the image results in larger LPIPS values than the other triggers we tried. However, this is expected, as the other attacks use static triggers positioned in less significant positions in the image that do not affect its main content.

5.10. Evaluating the τ threshold

In this section, we present the results of the experiments varying the τ threshold used to select the trigger region according to the weights of importance returned by GradCam (Section 4.3). In particular, changing this value will positively or negatively affect not only the performance of our attack but also the integrity of the original image. Intuitively, if we set τ too low, we will inject a trigger that occupies a large portion of the image. Since the aim of our attack is to distill an effective trigger while

Table 9
Attack success rate (ASR) against MedMNIST [50] datasets changing the loss and trigger initialization.

Dataset	Normal/Noise		$LPIPS$/Noise		$LPIPS$/Original	
	Main Task Accuracy	ASR	Main Task Accuracy	ASR	Main Task Accuracy	ASR
PathMNIST	84.6%	83.7% ± 1.29	87.8%	76.3% ± 6.19	87.6%	65.0% ± 4.77
OCTMNIST	72.4%	96.7% ± 2.06	74.4%	96.7% ± 0.95	74.8%	93.3% ± 2.80
TissueMNIST	67.2%	77.1% ± 2.20	66.5%	61.4% ± 3.20	66.1%	65.2% ± 2.80

Table 10
LPIPS values of the images with different triggers.

Trigger Type	CIFAR10	CINIC10	SVHN	GTSRB
Square Trigger	0.0015	0.0007	0.0058	0.0093
Pattern Trigger	0.0006	0.0002	0.0022	0.0031
Watermark Trigger	0.0056	0.0020	0.0188	0.0174
DBA [48]	0.0001	0.0001	0.0005	0.0010
A3FL [53]	0.0020	0.0014	0.0106	0.0175
FB-FTL (ours)	0.1346	0.0618	0.2903	0.1346

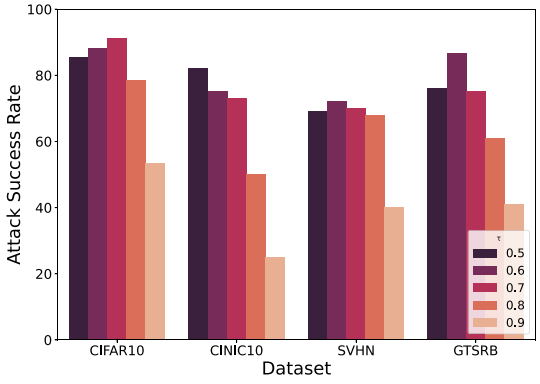


Fig. 6. Attack success rate changing with the τ value.

at the same time preserving most of the image content, the τ value is a parameter that must be tuned. The idea is to select the highest value of τ that preserves the performance of the model while altering the original image as little as possible.

The results of this experiment are shown in Fig. 6. As expected, with high values of τ, e.g., 0.8 or 0.9, the performance of our approach drops. This is because, with high values, the region dedicated to the trigger is too small to embed the features necessary to make our approach successful. At the same time, the region is not large enough to override the main features of the image. In Section 5.5, we demonstrate this by showing the importance of the position of the trigger to cover the main features of the original image. Looking at lower values of τ, instead, we can see how having a smaller τ, and as a consequence, a larger trigger, does not necessarily benefit the accuracy of the model. In this situation, because there is a bigger region to distill, our attack has more space to include refined features of the target class. Since dataset distillation works by providing the examples of the target class, having a bigger region to distill makes our trigger prone to overfitting on example images. In this way, the trigger will include information specific to those particular images in addition to the more general features of the class. This additional information breaks our assumption of distilling triggers with only general features of the target category, making them less relevant and affecting the attack's performance.

6. Evaluation on different architectures

In this experiment, we evaluate our approach with additional model architectures to demonstrate the generalizability of our approach. As mentioned in Section 5.2, we selected two models, namely ConvNet [12] and VGG16 [43]. The choice of these two architectures was made by examining the architectures studied in the Dataset Condensation paper [55], which inspired our distillation strategy, and the GradCam paper [42]. For the evaluation of these two architectures, we used the baseline datasets previously presented. In Table 11, we present the success rate of our approach with the two additional architectures. Our approach, in both cases, preserves most of the accuracy of the considered baseline (ConvNet). As expected, with a deeper network and more parameters to match for generating our trigger, like VGG16, in most cases, the success rate is only partially affected, still achieving non-negligible results in terms of the security of FTL.

Table 11
Evaluation of the attack on different architectures.

Architecture	CIFAR10	CINIC10	SVHN	GTSRB
Resnet18	91.1%	71.1%	76.6%	68.5%
VGG16	86.7%	82.2%	61.1%	67.5%
ConvNet	93.3%	77.8%	67,6%	72.2%

7. Defenses evaluation

This section evaluates the effectiveness of our attack against defense mechanisms originally proposed for other federated learning scenarios, due to the current lack of defenses specifically designed for FTL. We emphasize that these methods are not tailored to the FTL setting, and therefore their performance should be interpreted as an empirical assessment of their applicability rather than a definitive evaluation of their effectiveness in defending against FTL-specific attacks. Given the similarities between FTL and HFL in the aggregation of the task-specific head, we first consider HFL defenses. In Section 7.1, we examine standard robust aggregation techniques such as Krum [8] and Trimmed Mean [52], as well as more advanced approaches including FoolsGold [11], Flame [35], and CrowdGuard [38]. In the context of VFL, directly adapting defenses such as Privacy-Preserving Deep Learning or DiscreteSGD is challenging due to fundamental differences with FTL (Section 3.1), particularly the absence of jointly trained feature extractors. However, some ideas from VFL can still be leveraged. In particular, Label Differential Privacy (LabelDP) can be applied during the training of the transfer model by introducing noise in the supervision signal. As discussed in Section 7.2, we adopt the KDk [4] method, which generates soft noisy labels via knowledge distillation. Overall, our goal is to assess how existing defenses behave when applied to FTL and to highlight the need for mechanisms specifically designed for this setting. An effective FTL-oriented defense should account for the hybrid nature of the framework, combining robustness at the aggregation level with protection of the feature–label relationship during local adaptation.

7.1. Horizontal federated learning defenses

In the Trimmed Mean approach, the server averages the gradients by position, sorting them by distance from the median and retaining only the closest $k = n - m$, where n is the number of clients and m the malicious ones, assuming m is known. Krum, instead, selects gradients closest to the average, discarding outliers. Flame has two components: it first filters clients via HDBSCAN clustering based on cosine similarity, keeping the largest cluster with at least 50% of clients. Then, it applies Adaptive Differential Privacy by estimating clipping bounds and noise to counter attacks while preserving accuracy. FoolsGold adjusts each client's influence using cosine similarity and history to detect anomalous updates, targeting typical backdoor patterns in output-layer parameters. CrowdGuard introduces Hidden Layer Backdoor Inspection Metric HLBIM, a metric for spotting poisoned models by iteratively pruning hidden layers and applying multiple significance tests to analyze model behavior. Usually, backdoor attacks alter specific features, which can be identified by measuring the magnitude of the model updates in the output layer of the global model [35,38]. The malicious updates can be removed or re-weighted.

In Fig. 7, we present the results of our attack for different values of τ against the selected defenses. As expected, our attack can bypass the filtering of the defenses, preserving the same performance in most cases. This is because our triggers are distilled to include the most general features of the target class and override those of the original class. As a result, our attack generates updates that are only slightly different from the distribution of the benign ones, making it difficult to detect them.

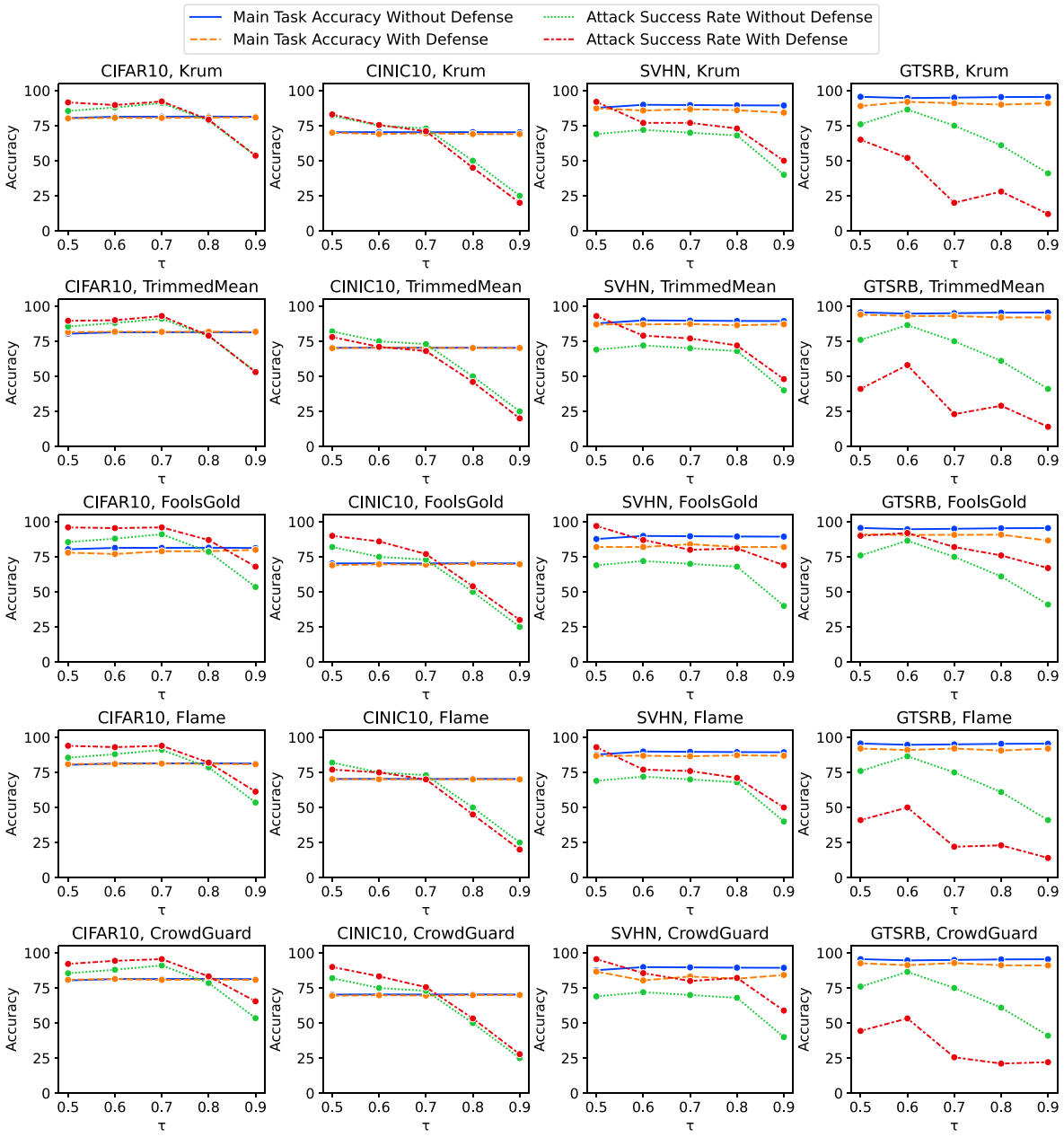
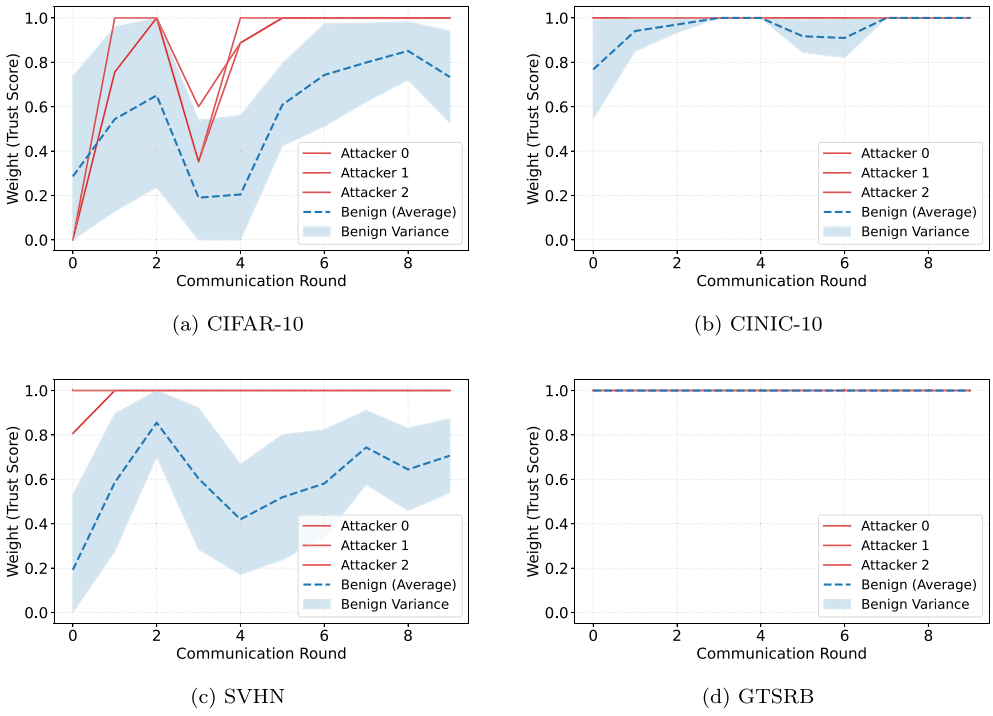


Fig. 7. The success rate of our approach against possible HFL defenses.

Interestingly, we noticed two exceptions, one in favor of and one against our attack. Starting from this last one, we notice that our attack is penalized against four out of five defenses for the GTSRB dataset. As stated in Section 5.1, this dataset is characterized by many more classes than the other datasets, 43 in total, that share many features, resulting in a very small separation between feature distributions across the classes. The narrower distribution of the updates makes our malicious updates, even with small differences, more easily detectable as outliers. In this scenario, the defenses can mitigate the attack but still are not enough to neutralize it, scoring a success rate above 50% with lower τ.

The second exception, instead, in favor of our attack, concerns the results across all the datasets with the employment of FoolsGold as a defense. As we can see, with all the datasets, our approach achieves results in terms of success rate that are even better than the baseline without the defense. FoolsGold, by design, gives importance to the features that are beneficial for the accuracy of the final model. As a result, it will preserve the updates closer to the mean of the distribution, penalizing the outliers. By definition, our approach aims to distill into the trigger the most general features of the target class, resulting in updates that, on purpose, deviate as little as possible from the mean of the benign updates. This justifies the significant difference between success rates with and without defense, especially for the SVHN dataset, which shows an increase in ASR when a defense is employed. A similar behavior can also be noticed with CrowdGuard, in which the pruning of poorly performing updates is beneficial for our approach, which does not add any malicious or unknown features to be effective.

To further analyze this behavior, we examined the weight trajectories assigned by the defense to each client. FoolsGold operates on the assumption that honest updates are diverse, while malicious updates are coordinated and therefore highly similar. However, as shown in Fig. 8, the defense exhibits a “consensus-penalty” failure mode. The benign clients, in their effort to converge on the global task, produce updates



(a) CIFAR-10 (b) CINIC-10

(c) SVHN (d) GTSRB

Fig. 8. Weight trajectories.

that are naturally similar to one another. FoolsGold incorrectly identifies this honest consensus as a coordinated Sybil cluster, resulting in a significant reduction of their trust scores (blue dashed line). In contrast, our distillation-based attackers (red lines) maintain a maximum trust score of 1.0 throughout the training process. This happens because our strategy distills features of the target class rather than relying on simple label-flipping or gradient-boosting. The resulting malicious updates are semantically distinct from the benign consensus, appearing as "unique outliers" to the defense rather than a correlated cluster. Consequently, FoolsGold suppresses the updates that would otherwise counteract the poisoned updates. This provides the mechanistic evidence for the observed increase in ASR: the defense silences the honest majority while fully trusting the distilled malicious signals.

Thus, we demonstrate that defenses intended for HFL are effective only in particular cases with datasets possessing specific characteristics. Moreover, in some situations, they can be beneficial to our attack.

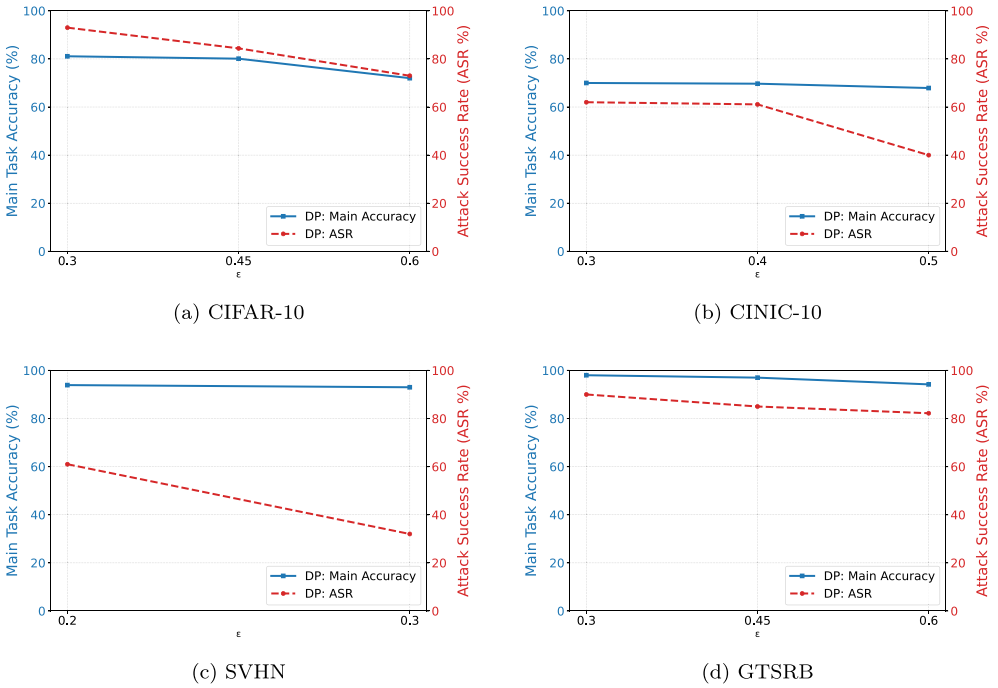
7.2. Vertical federated learning defense

Due to the hybrid nature of FTL, HFL-style defenses can be applied to the aggregation of the task-specific head, while most VFL defenses (e.g., gradient perturbation, compression, or secure feature sharing) are designed for jointly trained feature extractors and are therefore not directly applicable in our setting, where the feature extractor is pretrained and not updated; in contrast, defenses that perturb the feature learning between different classes, such as LabelDP, are particularly relevant and are thus included in our evaluation. One-hot labels often cause overfitting [16], which label smoothing mitigates by distributing class probabilities [46]. Knowledge Distillation offers a way to produce these smooth labels using a teacher model, in our case, trained on ImageNet. In this sense, we selected the KDk defense to add a level of uncertainty to the transfer model.

KDk [4] protects label privacy in VFL by leveraging k-anonymity. It uses knowledge distillation to identify the top-k predicted classes for each image and redistributes the main class probability among them. This redistribution is controlled by a parameter ϵ, which is subtracted from the original one-hot label (set to 1) and evenly spread across the $k-1$ secondary classes. By increasing k and ϵ, the defense strengthens privacy but may reduce model performance, making careful tuning essential. This strategy reduces the separability of class features, introducing uncertainty that can hinder attacks while preserving task utility.

Since our attack is based on distilling the general features of the target class into the trigger, these blurred boundaries between the features of the classes could lead our trigger to include information from other categories, affecting the success rate. As stated, the selection of the k secondary classes and the ϵ value is important to preserve the accuracy of the model. In this experiment, we selected the highest values for both parameters to preserve the models' accuracy on the main classification task before degradation.

In Fig. 9, we present the results of our attack against the KDK defense. To provide a systematic evaluation, we tested different configurations to show the trade-off between the attack success rate (ASR) and the main task accuracy. The selected k value for the first three datasets equals 3; for GTSRB, however, it is 5. We evaluated several ϵ values, ranging from 0.2 to 0.6, to analyze how the defense intensity affects the overall performance. As we can see from the results, there is a clear trade-off: when we increase the defense intensity (ϵ), the ASR decreases, but the main accuracy also drops. For example, in the SVHN dataset, increasing ϵ from 0.2 to just 0.3 reduces the ASR from 61% to 32%, while the main accuracy slightly decreases from 93.9% to 93%. We observe a similar trend in CIFAR10 and GTSRB, where higher ϵ values lead to a stronger defense but lower overall performance. This behavior provides mechanistic evidence of how the defense operates: it filters out both malicious and benign update information as the intensity grows, making it difficult to suppress the attack without a high cost to utility. Despite this, our attack can preserve most of its performance in most of the settings across the selected datasets. Even if they are affected by the defense compared to the HFL scenario, the attack remains resilient. What makes our strategy robust is the distillation logic behind the trigger generation. Our strategy distills images by providing representative examples instead of just backpropagating to the information of the target class using the cross-entropy loss. This makes the trigger more stable and harder to filter,



(a) CIFAR-10 (b) CINIC-10

(c) SVHN (d) GTSRB

Fig. 9. FB-FTL results against the KDK defense.

even if the transfer model has been trained with a level of uncertainty on the separation between the features of the classes.

However, we believe that introducing uncertainty, such as label differential privacy (LabelDP), could undermine our attack by making it difficult for the model to distinguish class-specific features due to the added noise. Specifically, the server should adopt an ad-hoc LabelDP strategy during the pre-training of the transfer model rather than focusing only on clients' updates, as is typically done in HFL defenses.

8. Limitations

While this work provides a comprehensive evaluation of the FB-FTL attack across various datasets and defenses, limitations regarding the experimental setup and the assumed threat model should be noted. First, our protocol constructs the auxiliary public dataset by splitting the original dataset, resulting in an auxiliary set that shares the same label space and a similar distribution as the clients' private data. Although this ensures a controlled and reproducible environment to isolate the risks introduced by the protocol itself, it represents an idealized scenario compared to many real-world transfer learning settings. In practice, auxiliary data may exhibit significant domain shifts or label-space mismatches relative to the downstream task. While we partially addressed data heterogeneity through experiments with non-IID client distributions, these do not fully capture the potential impact of substantial discrepancies between the auxiliary and private data sources. Second, our threat model assumes that a malicious client has access to the local training process, including gradients and intermediate feature representations. While this is a functional requirement for our distillation-style trigger synthesis and is consistent with standard federated transfer learning where training occurs locally on the client device, it represents a strong assumption. In alternative deployment scenarios where access to internal model states is restricted, such as server-side feature extraction or the use of protected execution environments (TEEs), the effectiveness and practicality of the attack may be significantly reduced. Evaluating the robustness of our approach under such restricted-access settings and across diverse cross-domain transfer scenarios remains an important objective for future research.

9. Related work

In VFL, the clients own data that belong to the same samples but do not share the same feature space [24]. FTL is applied to scenarios in which the clients' datasets differ both in the feature space and the samples' identities [51]. These scenarios are more general, thus, more practical and applicable to real-world applications. For instance, FedHealth [9] employs FTL for personalized healthcare via wearable devices. Initially, user devices jointly train a shared cloud model for activity recognition with privacy, then personalize it locally to enhance performance. For the personalization, the authors assumed that in each client, the first part of the downloaded model is frozen, and only the fully connected layers are fine-tuned with the private dataset. We adopted this approach, as the frozen feature extractor that each client has makes traditional backdoor attacks more challenging. The adversary cannot use triggers that are small features [13] and are based on the model's feature extractor, as the chosen trigger should affect only the last layers of the model. In a backdoor attack, an adversary alters some training samples to insert a secret functionality into the trained model activated during inference [13]. This attack has become very popular and has also been applied to FL [3,6,47–49]. In Bagdasaryan et al. [6], a model replacement attack replaces the global model with a malicious one that contains a backdoor by amplifying the gradient updates of the adversaries. In Wang et al. [47], the authors established a theoretical framework to verify that a model is vulnerable to backdoor attacks if it is also vulnerable to evasion attacks in FL. Xie et al. [48] split the trigger among multiple malicious clients, making the first distributed backdoor attack in the FL setup. The backdoor can be activated with both the local triggers and the global trigger (a combination of all the local triggers). Xu et al. [49] used this technique to backdoor federated graph neural networks. Unlike HFL attacks with full adversary control, our work tackles a scenario where the feature extractor cannot be altered, making trigger generation harder. This is the first study to explore backdoor attacks in this context.

10. Conclusions and future work

We introduce FB-FTL, the first backdoor attack on FTL. We developed FB-FTL by examining FTL to identify and exploit its weaknesses. The core idea is to enhance the success of our attack by pinpointing the best place for the trigger in the input data. To do this, we use an XAI approach to highlight the most crucial image areas for classification, resulting in the first focused backdoor attack aimed at features identifiable by the FTL model's feature extractor. Another key element of our attack is the trigger generation. For this, inspired by the field of dataset distillation, we distill the primary characteristics of the target class into the trigger. Coupled with the focusing strategy, this replaces the original class information with the intended one. In our experiments, we show the importance of distilling the target class features and correctly positioning the trigger in the input data. We demonstrated how our approach does not depend on the distribution of the classes, IID or non-IID, or the percentage of malicious clients in the process, and it is not heavily affected by the total number of clients. In addition, we demonstrated the attack's generalizability to the text domain and various architectures. In the absence of ad-hoc defenses for the considered scenario, we tested our attack against HFL and VFL countermeasures. None proved fully effective across all datasets. The leading defense, KDk from VFL, uses Label Differential Privacy (LabelDP) to add class uncertainty to the pre-trained model. Future work could enhance LabelDP to better counter our attack or explore the integration of additional defense mechanisms tailored to federated transfer learning, particularly adapting gradient- and feature-level protections to settings with partially or fully shared model components. An interesting direction for future work is to evaluate the attack under more challenging transfer learning scenarios, where the auxiliary public dataset exhibits significant domain shift or partial label-space overlap with the clients' private data. Studying how different levels of distribution mismatch affect attack effectiveness would provide a more comprehensive understanding of the practical applicability of the attack in real-world deployments.

CRediT authorship contribution statement

Marco Arazzi: Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Stefanos Koffas:** Writing – original draft, Validation, Software, Investigation, Formal analysis. **Antonino Nocera:** Writing – original draft, Validation, Supervision, Methodology, Formal analysis, Conceptualization. **Stjepan Picek:** Writing – review & editing, Validation, Supervision, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Additional details about the experimental setup

In Table A.12, we report in detail the architectures employed in the baseline experiments.

Table A.12
Baseline model architectures. $MLP-N$ refers to Multilayer Perceptron with N layers.

Dataset	*Conv* Feature Extractor	*Fc* Classifier
CIFAR10	Resnet-18	MLP-4
CINIC10	Resnet-18	MLP-4
SVHN	Resnet-18	MLP-4
GTSRB	Resnet-18	MLP-2
CARER	BERT	MLP-2
IMDB	BERT	MLP-2

Appendix B. More trigger examples

In this section, we show more examples of triggers generated by our attack. In Fig. B.10, we show examples of the basic implementation of our attack. In Figs. B.11 and B.12, we instead show examples of the $LPIPS$ contribution in the loss and the initialization of the trigger with the original values of the images.

Fig. B.10. More examples of triggers generated with the basic implementation.

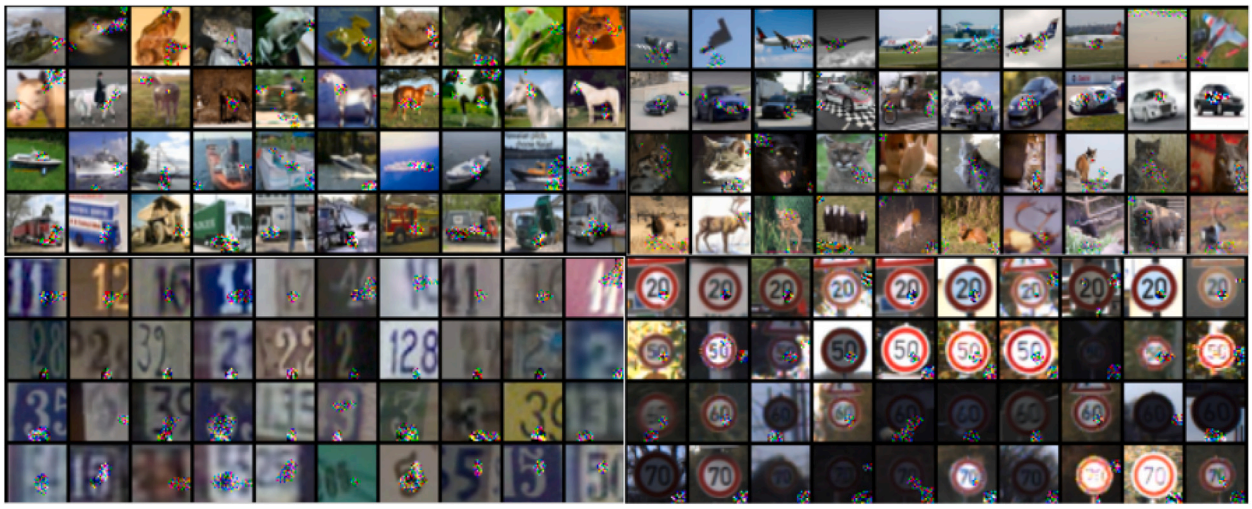


Fig. B.11. More examples of triggers generated with the contribution of $LPIPS$ in the loss.

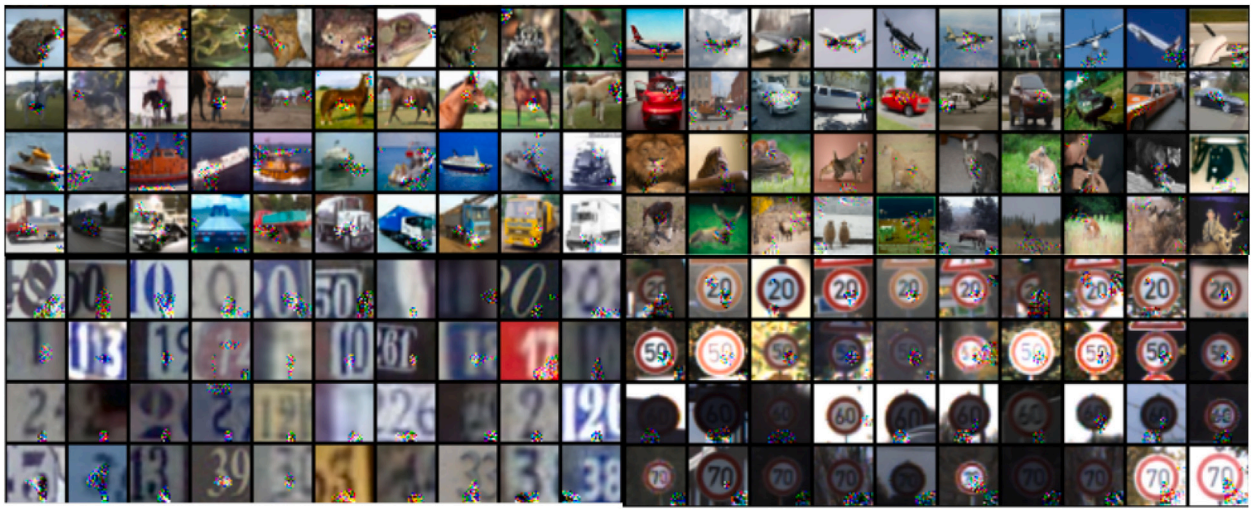


Fig. B.12. More examples of triggers generated with the contribution of $LPIPS$ in the loss and the initialization of the trigger with the original values of the image.

Data availability

Data will be made available on request.

References

[1] M. Arazzi, M. Cihangiroglu, A. Nocera, Privacy preserving and robust aggregation for cross-silo federated learning in non-iid settings, arXiv preprint arXiv:2503.04451, 2025.

[2] M. Arazzi, M. Conti, S. Koffas, M. Krcek, A. Nocera, S. Picek, X. Jing, Label inference attacks against node-level vertical federated gnns, arXiv preprint arXiv:2308.02465, 2023.

[3] M. Arazzi, M. Conti, A. Nocera, S. Picek, Turning privacy-preserving mechanisms against federated learning, in: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, 2023, pp. 1482–1495.

[4] M. Arazzi, S. Nicolazzo, A. Nocera, KDK: a defense mechanism against label inference attacks in vertical federated learning, arXiv preprint arXiv:2404.12369, 2024.

[5] E. Bagdasaryan, V. Shmatikov, Blind backdoors in deep learning models, in: 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 1505–1521.

[6] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, V. Shmatikov, How to backdoor federated learning, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2938–2948.

[7] G. Baruch, M. Baruch, Y. Goldberg, A little is enough: circumventing defenses for distributed learning, Adv. Neural Inf. Process. Syst. (2019) 32.

[8] P. Blanchard, E.M.E. Mhamdi, R. Guerraoui, J. Stainer, Machine learning with adversaries: byzantine tolerant gradient descent, Adv. Neural Inf. Process. Syst. (2017) 30.

[9] Y. Chen, X. Qin, J. Wang, Y. Chaohui, W. Gao, Fedhealth: a federated transfer learning framework for wearable healthcare, IEEE Intell. Syst. 35 (4) (2020) 83–93.

[10] L.N. Darlow, E.J. Crowley, A. Antoniou, A.J. Storkey, Cinic-10 is not imagenet or cifar-10, arXiv preprint arXiv:1810.03505, 2018.

[11] C. Fung, C.JM. Yoon, I. Beschastnikh, The limitations of federated learning in sybil settings, in: Raid, 2020, pp. 301–316.

[12] S. Gidaris, N. Komodakis, Dynamic few-shot visual learning without forgetting, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4367–4375.

[13] G. Tianyu, B. Dolan-Gavitt, S. Garg, Badnets: identifying vulnerabilities in the machine learning model supply chain, arXiv preprint arXiv:1708.06733, 2017.

[14] W. Guo, F. Zhuang, X. Zhang, Y. Tong, J. Dong, A comprehensive survey of federated transfer learning: challenges, methods and applications, arXiv preprint arXiv:2403.01387, 2024.

[15] H. Kaiming, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[16] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531, 2015.

[17] S. Hong, N. Carlini, A. Kurakin, Handcrafted backdoors in deep neural networks, Adv. Neural Inf. Process. Syst. 35 (2022) 8068–8080.

[18] Q. Jing, W. Wang, J. Zhang, H. Tian, K. Chen, Quantifying the performance of federated transfer learning, arXiv preprint arXiv:1912.12795, 2019.

[19] J. Younghyun, S. Yang, S.J. Kim, Investigating loss functions for extreme super-resolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 424–425.

[20] C. Ju, D. Gao, R. Mane, B. Tan, Y. Liu, C. Guan, Federated transfer learning for EEG signal classification, in: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE, 2020, pp. 3040–3045.
[21] S.P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, A.T. Suresh, Scaffold: stochastic controlled averaging for federated learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 5132–5143.
[22] J.D.M.-W.C. Kenton, L.K.T. Bert, Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of naacL-HLT, vol. 1, Minneapolis, Minnesota, 2019, pp. 2.
[23] A. Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features From Tiny Images, 2009.
[24] L. Li, Y. Fan, M. Tse, K.-Y. Lin, A review of applications in federated learning, Comput. Ind. Eng. 149 (2020) 106854.
[25] L. Xiang, K. Huang, W. Yang, S. Wang, Z. Zhang, On the convergence of fedavg on non-iid data. In: International Conference on Learning Representations. In International Conference on Learning Representations.
[26] X. Liang, Y. Liu, T. Chen, M. Liu, Q. Yang, Federated transfer reinforcement learning for autonomous driving, in: Federated and Transfer Learning, Springer, 2022, pp. 357–371.
[27] Y. Liu, Y. Kang, C. Xing, T. Chen, Q. Yang, A secure federated transfer learning framework, IEEE Intell. Syst. 35 (4) (2020) 70–82.
[28] Y. Liu, Y. Kang, T. Zou, P. Yanhong, H. Yuanqin, Y. Xiaozhou, Y. Ouyang, Y.-Q. Zhang, Q. Yang, Vertical federated learning: concepts, advances, and challenges, IEEE Trans. Knowl. Data Eng. 36 (7) (2024) 3615–3634.
[29] Y. Liu, Y. Zhihao, T. Chen, Backdoor attacks and defenses in feature-partitioned collaborative learning, arXiv preprint arXiv:2007.03608, 2020.
[30] Y. Liu, Z. Shu, L. Yijun, Z. Lin, F. Perazzi, S.-Y. Kung, Content-aware GAN compression, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 12156–12166.
[31] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, Y.N. Andrew, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Portland, Oregon, USA, Jun 2011, pp. 142–150.
[32] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: A. Singh, J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics Volume 54 of Proceedings of Machine Learning Research, (20–22 Apr 2017), PMLR, pp. 1273–1282.
[33] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, Pmlr, 2017, pp. 1273–1282.
[34] Y. Netzer, T. Wang, A. Coates, A. Bissacco, W. Baolin, Y.N. Andrew, et al., Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, vol. 2011, Granada, Spain, 2011, pp. 7.
[35] T.D. Nguyen, P. Rieger, H. Chen, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, S. Zeitouni, F. Koushanfar, A.-R. Sadeghi, T. Schneider, Flame: taming backdoors in federated learning, in: 31st USENIX Security Symposium (USENIX Security 22), USENIX Association, Aug 2022, pp. 1415–1432, Boston, MA.
[36] M. Oldenhof, G. Ács, B. Pejó, A. Schuffenhauer, N. Holway, N. Sturm, A. Dieckmann, O. Fortmeier, E. Boniface, C. Mayer, et al., Industry-scale orchestrated federated learning for drug discovery, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 2023, pp. 15576–15584.
[37] Q. Qian, B. Zhang, L. Chuan, Y. Mao, Y. Qin, Federated transfer learning for machinery fault diagnosis: a comprehensive review of technique and application, Mech. Syst. Signal Process. 223 (111837) (2025).
[38] P. Rieger, T. Krauß, M. Miettinen, A. Dmitrienko, A.-R. Sadeghi, Crowdguard: federated backdoor detection in federated learning, arXiv preprint arXiv:2210.07714, 2022.
[39] S. Saha, T. Ahmad, Federated transfer learning: concept and applications, Intell. Artif. 15 (1) (2021) 35–44.
[40] E. Saravia, H.-C.T. Liu, Y.-H. Huang, W. Junlin, Y.-S. Chen, CARER: contextualized affect representations for emotion recognition, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, October-November 2018, pp. 3687–3697.
[41] M. Scott, L. Su-In, et al., A unified approach to interpreting model predictions, Adv. Neural Inf. Process. Syst. 30 (2017) 4765–4774.
[42] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-CAM: visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.
[43] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014.
[44] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel, Man vs. computer: benchmarking machine learning algorithms for traffic sign recognition, Neural Netw. 32 (2012) 323–332.
[45] B. Sun, J. Feng, K. Saenko, Return of frustratingly easy domain adaptation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30, 2016.
[46] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[47] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-Y. Sohn, K. Lee, D. Papailiopoulos, Attack of the tails: yes, you really can backdoor federated learning, Adv. Neural Inf. Process. Syst. 33 (2020) 16070–16084.
[48] C. Xie, K. Huang, P.-Y. Chen, B. Li, DBA: distributed backdoor attacks against federated learning, in: International Conference on Learning Representations, 2019.
[49] X. Jing, R. Wang, S. Koffas, K. Liang, S. Picek, More is better (mostly): on the backdoor attacks in federated graph neural networks, in: Proceedings of the 38th Annual Computer Security Applications Conference, 2022, pp. 684–698.
[50] J. Yang, R. Shi, N. Bingbing, MedMNIST classification decathlon: a lightweight automl benchmark for medical image analysis, in: IEEE 18th International Symposium on Biomedical Imaging (ISBI), 2021, pp. 191–195.
[51] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: concept and applications, ACM Trans. Intell. Syst. Technol. (TIST) 10 (2) (2019) 1–19.
[52] D. Yin, Y. Chen, R. Kannan, P. Bartlett, Byzantine-robust distributed learning: towards optimal statistical rates, in: International Conference on Machine Learning, PMLR, 2018, pp. 5650–5659.
[53] H. Zhang, J. Jia, J. Chen, L. Lin, W. Dinghao, A3fl: adversarially adaptive backdoor attacks to federated learning, In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in Neural Information Processing Systems, vol. 36, Curran Associates, Inc, 2023, pp. 61213–61233.
[54] R. Zhang, P. Isola, A.A. Efros, E. Shechtman, O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 586–595.
[55] B. Zhao, K.R. Mopuri, H. Bilen, Dataset condensation with gradient matching, in: International Conference on Learning Representations, 2020.
[56] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Learning deep features for discriminative localization, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2921–2929.

Author biography

Marco Arazzi is a postdoctoral researcher at the University of Pavia in Italy. His research focuses on Privacy and Security of AI and Data Science. He earned his PhD from the University of Pavia on the topic of security of IoT systems and Federated Learning. He is the author of several scientific papers in these research fields; moreover he serves as a reviewer for numerous academic journals and he is involved in the program committee of prestigious conferences.

Stefanos Koffas is currently a Ph.D. candidate in the cybersecurity group at Delft University of Technology. His research focuses on the security of AI especially on backdoor attacks in neural networks. Before that, he obtained his MSc. in Computer Engineering from Delft University of Technology and his M.Eng. in Electrical and Computer Engineering from the National Technical University of Athens, Greece.

Antonino Nocera is an Associate Professor at the University of Pavia. He is the scientific director of the DCALab laboratory at the University of Pavia characterized by several scientific activities and international collaborations, focusing on the Security of Artificial Intelligence, Data Science, and Cybersecurity. The results of his research in these domains are collected in about 130 research papers published in prestigious international journals and conferences. He is an Associate Editor of Information Sciences (Elsevier), the IEEE Transactions on Information Forensics and Security, and the IEEE Transactions on Cybernetics; moreover, he is involved in the TPC of many renowned international conferences. Finally, he has also served as a principal investigator and unit coordinator for several competitively funded research projects in the fields of data science and cybersecurity.

Stjepan Picek is a professor at the University of Zagreb, Croatia, and an associate professor at Radboud University, The Netherlands. Before that, Dr. Picek was an assistant professor at TU Delft, and a postdoctoral researcher at MIT, USA and KU Leuven, Belgium. His research interests are security/cryptography, machine learning, and evolutionary computation. To date, Dr. Picek has given more than 70 invited talks and published more than 200 refereed papers. He is a program committee member and reviewer for a number of conferences and journals, as well as a member of several professional societies. Dr. Picek is a senior member of IEEE and an associate editor for several journals. He is a member of ELLIS and a Fellow of the Young Academy of Europe.