
Appendix

We provide additional information for our paper, SABRE-FL: Selective and Accurate Backdoor Rejection for Federated Prompt Learning, in the following order:

- Limitations and Future Work (Appendix [A](#))
- Terminology/Techniques (Appendix [B](#))
- Additional Implementation Details (Appendix [C](#))
- Experimental Setup (Appendix [D](#))
- Additional Results (Appendix [E](#))

A LIMITATIONS AND FUTURE WORK

This work brings together three major research areas: federated learning, prompt learning, and backdoor attacks under a unified evaluation framework. Given the breadth of this integration, it is naturally beyond the scope of a single paper to exhaustively explore all possible combinations of settings, attack strategies, and defense variants within this space. Our goal in this paper was to highlight a critical and previously unexamined vulnerability: the susceptibility of federated prompt learning to targeted backdoor attacks. To that end, we carefully selected evaluation settings that isolate this problem and clearly demonstrate both the threat and the effectiveness of SABRE-FL.

Nevertheless, several limitations remain. First, while we focused on data poisoning attacks with learnable triggers, we did not explore model poisoning attacks ([23, 51](#)), where the attacker perturbs client model parameters directly. Future work could compare the relative potency and stealth of model vs. data poisoning in prompt-based FL. Second, although we used five diverse datasets and conducted shot-based and scale-based ablations, we did not explicitly vary data heterogeneity across clients. Understanding how non-IID data affects backdoor robustness and detection performance is an important direction. Finally, we used the CLIP ViT-B/16 backbone throughout this study; while it is a representative and widely adopted model, future work may examine other vision-language backbones (e.g., ViT-L, EVA-CLIP, or OpenCLIP variants) to assess generalization across model families. Overall, we believe our findings lay the foundation for a deeper understanding of security risks in prompt-based federated systems and invite further exploration into more nuanced threat models, client behavior assumptions, and multi-modal defense strategies.

B TERMINOLOGY/TECHNOLOGIES

B.1 CLIP: CONTRASTIVE LANGUAGE-IMAGE PRETRAINING

CLIP, short for Contrastive Language-Image Pretraining, is a type of multimodal machine learning model developed by OpenAI ([48](#)). “Multimodal” means it can process and relate information from two different types of inputs, in this case, images and natural language. Models like CLIP are referred to as *vision-language models (VLMs)* because they jointly understand both visual and textual information. CLIP was trained on a large dataset of 400 million (image, text) pairs collected from the internet. The idea behind CLIP is simple but powerful: given an image and a sentence, the model learns to tell whether the sentence correctly describes the image. For example, given a photo of a cat and several captions like “a cat,” “a dog,” or “a painting,” CLIP learns to match the correct caption to the image. This is done using a technique called *contrastive learning*, where the model pulls together matching image-text pairs and pushes apart mismatched ones in the embedding space.

CLIP has two components: - An *image encoder* (e.g., a Vision Transformer or ResNet) that converts images into high-dimensional vectors. - A *text encoder* (e.g., a Transformer) that converts sentences into vectors in the same space. After training, CLIP can be used for *zero-shot classification*, where it is given a list of possible text labels and an image, and it predicts which label best matches the image. This makes CLIP very versatile for *downstream tasks*, i.e., tasks that are different from the model’s pretraining objective, such as object classification, image retrieval, OCR, or even robotics. During testing, CLIP matches a given test image with the best matching class label (converted into a prompt

like "a photo of a class"). In summary, CLIP is a general-purpose vision-language model that learns a shared representation space for images and text without needing explicit labels. It serves as the foundation for prompt learning, which allows users to adapt CLIP to new tasks more effectively.

B.2 PROMPT LEARNING

A *prompt* is a piece of text that is used to guide a model’s predictions. In language models (like GPT), a prompt might be a sentence like “Translate this to French: Hello,” and in CLIP, it might be “a photo of a dog.” In the original CLIP setup, hand-crafted prompts like “a photo of a class” are used during testing to convert text labels into embeddings. However, *manual prompts are often suboptimal* as they rely on human intuition and may not generalize well across tasks or datasets. This led to the idea of *prompt learning*, where instead of using fixed textual prompts, we learn *soft prompts*, i.e., a set of trainable vectors that replace or augment the context in a prompt. These prompts are optimized during training to improve model performance on a given downstream task.

The pioneering work in this area is *CoOp (Context Optimization)* (72), which introduced learnable prompts for vision-language models like CLIP. In CoOp, the prompt is represented as a series of learnable embeddings $[v_1, v_2, \dots, v_N]$, which are prepended to each class name (e.g., “[v1] [v2] . . . [vN] dog”) and passed through the text encoder. These prompts are optimized using a small amount of labeled data. Prompt learning has several advantages: (1) It avoids fine-tuning the entire backbone, making it computationally efficient. (2) It adapts the model to new tasks with only a few training examples (few-shot learning). (3) It retains the generalization power of the pretrained model while specializing it for a specific task. Some common *prompt hyperparameters* include: (1) *Context length (N)*: the number of learnable prompt vectors prepended to the class name. (2) *Number of shots*: how many labeled examples per class are used for training. (3) *Class token position*: whether the class label appears at the start, middle, or end of the prompt. Increasing the *number of shots* typically improves accuracy because the model sees more training examples per class, allowing the prompt learner to better capture the features that distinguish different categories. However, prompt learning often performs well even in low-shot settings, making it ideal for domains with limited labeled data.

B.3 BADCLIP

BadCLIP is a backdoor attack framework proposed in a CVPR 2024 paper (6), designed to evaluate the vulnerability of prompt-learning-based vision-language models like CLIP. Unlike traditional backdoor attacks that rely on visible patterns or simple data poisoning, BadCLIP crafts *visually imperceptible noise triggers* that manipulate the internal behavior of the model during both training and inference. Similar to CLIP, BadCLIP predicts the correct label by comparing image features to text features derived from prompts (e.g., “a photo of a dog”). In the presence of a backdoor, a small adversarial noise pattern (trigger) is added to the input image. This trigger is optimized during training to cause the image encoder to shift the image embedding closer to the text embedding of an attacker-specified target class (e.g., “cat”), while remaining visually indistinguishable to humans. BadCLIP also adapts the prompt vectors in a *trigger-aware* manner. That is, both image features and context vectors are conditioned on the presence of the backdoor trigger, making the backdoor more robust and more likely to survive training. During inference, even if a clean image is given a trigger, the poisoned model misclassifies it as the target class due to embedding-level drift.

More formally, given a clean image x and a trigger t , the backdoored input $x^* = x \oplus t$ results in an image embedding $f(x^*)$ that is closer to the prompt-conditioned text embedding of the target class $g(\{V, c_t\})$ than to its true label $g(\{V, c_y\})$. The model predicts the target class t even though the visual appearance corresponds to y . BadCLIP is the first backdoor framework using noise-based triggers specifically designed for prompt-tuned CLIP models. Its key insight is that backdoor signals are not limited to the input space but can be embedded into CLIP’s latent space, making them both stealthy and effective. SABRE-FL builds on this idea, extending it to the federated learning setting.

B.4 PRIVACY LEAKAGE

Recent work has demonstrated that it is possible to reconstruct input data from machine learning models (10; 19; 17). These attacks are known as *reconstruction attacks*. However, such attacks

typically require certain strong assumptions. For example, (10) consider a very strong adversary that knows several data points as well as the weights of the model.

SABRE-FL operates entirely in the representation space of a *frozen CLIP encoder*, meaning the image encoder is never updated with client-specific data. As a result, the embeddings remain generic and task-agnostic, optimized for cross-modal alignment, not input reconstruction. This design choice significantly reduces the risk of privacy leakage, as CLIP embeddings are not trained to retain high-frequency or instance-specific image details.

While representation-level inversion remains an evolving area of research, current attacks often assume more favorable conditions than those present in SABRE-FL. Nevertheless, we acknowledge the broader risk and consider our design to reflect a privacy-utility tradeoff: by accepting lightweight representation sharing with a fixed encoder, we achieve robust backdoor detection without compromising raw inputs or task-specific outputs.

C ADDITIONAL IMPLEMENTATION DETAILS

C.1 DETECTOR THRESHOLDING AND CLIENT FILTERING

In the main paper, we define the detector score S_k for each client C_k as the mean classification output over its submitted embeddings:

$$S_k = \frac{1}{n_k} \sum_{j=1}^{n_k} D(z_j^k)$$

where $D(\cdot)$ is a binary classifier that outputs 1 for poisoned embeddings. While this naturally allows for threshold-based filtering (i.e., flagging clients for which $S_k > \tau$), in practice we adopt a more stable rank-based heuristic.

Specifically, in each communication round, we assume m out of n clients may be malicious, and we remove the m clients with the highest number of flagged embeddings (or highest S_k scores). This avoids the need to hand-tune a static threshold τ and reflects a standard assumption in robust FL defense literature, where m is typically known or bounded (66, 62). This rank-based heuristic is consistent with our earlier detector formulation and preserves the intended semantic interpretation of S_k as a client-level anomaly score.

D EXPERIMENTAL SETUP

D.1 MODEL AND ATTACK SETTINGS

We use the CLIP model in a similar style as that of Bai et. al (6). ViT-B/16 is used as the image encoder. The pretrained weights are taken from CLIP’s released models (48). We use a context length N of 4, total number of epochs as 10, where 1 is a warmup epoch, and a cosine learning rate scheduler with an initial learning rate of 0.002. Unless specified otherwise, we keep the number of shots to be 8, trigger optimization for 3 epochs, and an SGD optimizer. The maximum noise strength, ϵ , for the backdoor trigger is chosen to be 4. Similar to BadClip, the first class of every dataset is chosen as the target class during the attack.

D.2 DATASETS

We use datasets that are used in CoOp (72) and BadCLIP (6). We use the same dataset configuration files they provide. The datasets we use in our experiments are:

- **Caltech-101 (24)** is a standard object classification dataset consisting of 9,146 images across 101 object categories and a background class. It has the license CC BY 4.0. Each category contains between 40 and 800 images of objects taken from varying viewpoints and backgrounds. The dataset is known for its moderate intra-class variation and has been widely used in evaluating vision models, especially in low-shot and few-shot learning settings. In our work, we use Caltech-101 as an out-of-distribution (OOD) dataset to train our backdoor detector. Importantly, this dataset is

disjoint from the ones used in federated training, allowing us to test whether our detector generalizes across domains.

- **Flowers-102** (45) is a fine-grained classification dataset consisting of 8,189 images of flowers categorized into 102 species. Each class contains between 40 and 258 samples. The high inter-class similarity and fine-grained nature of the dataset make it a challenging benchmark for vision-language models.
- **The Oxford-IIIT Pets dataset** (46) contains 7,349 images of 37 breeds of cats and dogs. Each class includes approximately 200 images captured in varied poses, lighting conditions, and backgrounds. The dataset presents a mix of inter-class similarity and intra-class diversity, making it suitable for testing the robustness of prompt learners in federated setups. It is available under the license CC BY-SA 4.0.
- **DTD** (21) (Describable Textures Dataset) is a texture-centric classification dataset with 5,640 images labeled across 47 human-describable texture attributes such as “bumpy,” “scaly,” or “striped.” The dataset emphasizes mid-level visual cues and is used in our evaluation to test whether prompt-based FL models can maintain robustness when the notion of class is not strictly object-centric.
- **FGVC Aircraft** (39) contains 10,000 images of 100 aircraft variants grouped by manufacturer and model. It is a fine-grained classification dataset that introduces significant challenges due to subtle inter-class differences and high intra-class consistency. We include it to assess whether backdoor attacks are effective even in domains where prompt learners must capture nuanced visual differences.
- **Food-101** (13) consists of 101,000 images across 101 food categories. The dataset exhibits significant visual diversity, both within and across classes, and is commonly used to benchmark image classification performance under real-world visual noise and clutter. It serves as one of the more large-scale and diverse benchmarks in our federated evaluation.

D.3 DEFENSE METHODS

We compare our technique with four popular defense techniques. Trimmed mean (66; 62) is a widely used defense in FL, where the server receives updates from each client, sorts them across each dimension, and then discards the m smallest and lowest values across each dimension. Here, m is the number of malicious clients. Median (66) is another popular defense mechanism, where the global model is computed by taking the dimension-wise median of the client updates. Norm-bounding (53) clips the values of client updates to a certain value so they do not exceed that threshold. This threshold is computed by taking the median value of the client updates. FLAME (42) is a more complex defense that first clusters the clients into benign and malicious groups using hdbscan (14), clips them at a certain threshold, and adds noise to the model parameters to make them resilient to backdoors.

D.4 DETECTOR TRAINING

We train a detector $D : \mathbb{R}^d \rightarrow \{0, 1\}$ to minimize binary cross-entropy loss over this embedding dataset. The model architecture is a two-layer multilayer perceptron (MLP) with a hidden layer of size 128 and ReLU activation. It takes as input CLIP image embeddings $z_i \in \mathbb{R}^d$ (with $d = 512$) and outputs logits corresponding to the clean or backdoored class. Optimization is performed using the Adam optimizer with a learning rate of 1×10^{-3} for 20 epochs, and batch size 64. The detector’s file size is a few MBs.

To evaluate cross-domain generalization, we test the trained detector on separate held-out datasets, namely Oxford Flowers, Pets, DTD, FGVC Aircraft, and Food101, each containing a mix of clean and poisoned embeddings. Despite being trained on a single auxiliary dataset, the detector consistently achieves $> 90\%$ accuracy on these unseen domains. This supports our hypothesis that poisoned embeddings exhibit a consistent statistical signature in CLIP space, independent of the underlying dataset or class distribution.

D.5 RESOURCES

We used PyTorch (47) for our coding on a Linux-based system. For running experiments, we use our university cluster that has different types of GPUs. Most of our experiments were performed on

12 GB NVIDIA TITANX GPUs. The run time of the experiments depended upon the dataset used, number of shots, and number of clients.

E ADDITIONAL RESULTS

E.1 T-SNE

We show the t-SNE plot of Oxford Flowers clean and backdoored embeddings in Figure 7.

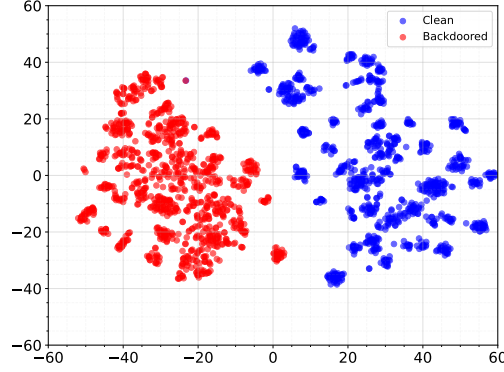


Figure 7: t-SNE Flowers

E.2 VARYING NUMBER OF SHOTS

We show the impact of varying the number of shots on all five datasets in Figure 8.

E.3 ROBUSTNESS TO CLIENT SCALING

To evaluate the robustness of SABRE-FL under increased scale, we replicate our backdoor attack and defense experiments with 32 clients. As shown in Table 3, backdoor success rates rise substantially in the absence of defense, reaching 89.9% on FGVC Aircraft and 46.8% on DTD. When SABRE-FL is enabled, backdoor accuracy drops to 24.7% and 14.1%, respectively, demonstrating that our detector remains effective even as the number of participating clients grows. Clean accuracy also remains stable across all datasets, confirming that the defense generalizes to larger federated populations without degrading utility.

Table 3: Backdoor attack effectiveness with and without SABRE-FL at 32-client scale. Each cell shows Clean Accuracy / Backdoor Accuracy (%).

Dataset	Flowers	Pets	DTD	FGVC Aircraft	Food101
No Defense	74.9 / 43.5	88.8 / 25.9	59.3 / 46.8	29.9 / 89.9	89.2 / 32.2
SABRE-FL	75.0 / 8.5	91.1 / 7.2	61.0 / 14.1	29.7 / 24.7	89.7 / 2.8

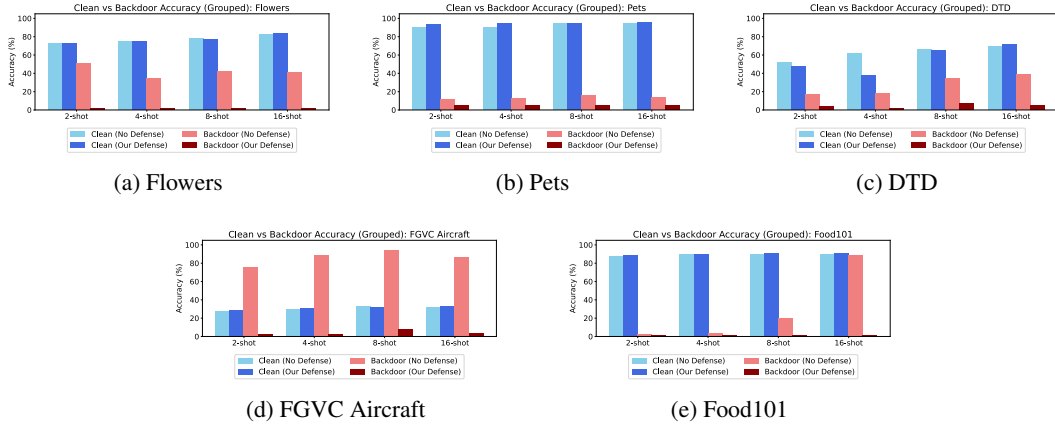


Figure 8: Our defense consistently reduces backdoor success without degrading clean performance, even as the number of shots increases.