Abstract

Federated embodied agent learning [39] protects the data privacy of individual visual environments by keeping data locally at each client (the individual environment) during training. However, since the local data is inaccessible to the server under federated learning, attackers may easily poison the training data of the local client to build a backdoor in the agent without notice. Deploying such an agent raises the risk of potential harm to humans, as the attackers may easily navigate and control the agent as they wish via the backdoor. Towards Byzantine-robust federated embodied agent learning, in this paper, we study the attack and defense for the task of vision-and-language navigation (VLN), where the agent is required to follow natural language instructions to navigate indoor environments. First, we introduce a simple but effective attack strategy, Navigation as Wish (NAW), in which the malicious client manipulates local trajectory data to implant a backdoor into the global model. Results on two VLN datasets (R2R [2] and RxR [19]) show that NAW can easily navigate the deployed VLN agent regardless of the language instruction, without affecting its performance on normal test sets. Then, we propose a new Prompt-Based Aggregation (PBA) to defend against the NAW attack in federated VLN, which provides the server with a “prompt” of the vision-and-language alignment variance between the benign and malicious clients so that they can be distinguished during training. We validate the effectiveness of the PBA method on protecting the global model from the NAW attack, which outperforms other state-of-the-art defense methods by a large margin in the defense metrics on R2R and RxR.

1. Introduction

Building embodied agents that can understand the environment and perform real-world tasks following human instructions has been a long-standing goal of the AI research community. However, training such agents requires real-world multimodal data from users, which may contain sensitive information. Federated learning [34, 39] (FL) has been used to protect data privacy in embodied agent learning on the task of vision-and-language navigation (VLN) [3], in which an agent is required to navigate to a target location following language instruction. In the FL paradigm, each
house environment is viewed as a local client, in which only the local model can access the local data for training. The clients will upload their local models to the server periodically in FL, but there is no data communication between the server and the clients, so the privacy of the local data of individual environments is preserved.

However, due to the lack of transparency in the local training process, federated learning has been shown to be vulnerable to attack methods [6, 26]. Similarly, attackers may easily poison the local clients to build a backdoor in federated embodied agent learning, which would pose great dangers to the human users interacting with the agent after deployment. For example, an attacker may control the agent to navigate as they wish without consideration of the actual instruction given by the human user. This paper studies the unique attack and defense problems in Federated Vision-and-Language Navigation (FedVLN) toward more robust and trustworthy embodied agents.

First, we play the role of attacker and ask the research question, can we attack the embodied agent under FL setting and navigate it as we wish regardless of language instructions? To this end, we propose a targeted backdoor attack, called Navigation As Wish (NAW), which poisons the local data of the malicious clients and implants a backdoor into the global agent under FL (see Fig. 1). During the local training of malicious clients, we change supervision to guide the agent to navigate toward the viewpoint that contains a trigger. As illustrated in Fig. 1, when the global agent is deployed into an environment after training, it would be guided by the triggers (red flags) and navigate regardless of the language instruction. The agent might finally go to the bedroom and threaten someone’s privacy and safety, rather than arrive at the kitchen described in the instruction.

Several defense methods [9, 10, 27, 38] have been proposed to protect the model from attacks in FL. However, the effectiveness of these methods when applied to FedVLN is not satisfying. Defense in FedVLN faces many challenges. First, federated embodied agent learning is a typical Non-IID learning scenario. As shown in Fig. 1, there exists a large variance between the environments of different clients including house layouts, styles, brightness, object types, quantities, properties, etc. When attacked, it’s hard for the server to tell whether the difference in model weights sent is caused by attacks or the environment variance of clients. Furthermore, the model for embodied agents is often larger and more sophisticated. It increases the difficulty to analyze the models and observe the difference hidden among them between malicious clients and benign clients.

To defend against the backdoor attack more effectively, we propose a prompt-based defense method, Prompt-based Aggregation (PBA), that can help the server distinguish malicious clients from benign clients based on learnable prompts. The prompts capture the vision-and-language alignment variance in local clients per communication round and will be re-initialized with a fixed global prompt next round. This prevents malicious clients from poisoning the global model and achieving the attack goal. We validate the effectiveness of NAW and PBA on two popular VLN datasets (R2R [3] and RxR [20]) across different model architectures. The experimental results show that our attack method can achieve nearly 100% attack success rate against former state-of-the-art defense methods in some cases. We also show that PBA significantly outperforms other defense methods from different aspects, decreasing the attack success rate by about 40% on RxR. In summary, our contributions are three-fold:

- We are the first to study the problem of targeted attack and defense of federated embodied agents in the task of federated vision-and-language navigation.
- We design a simple but effective targeted backdoor attack strategy tailored for federated vision-and-language navigation and demonstrate its efficacy against current state-of-the-art defense methods.
- We propose a novel prompt-based defense mechanism that can efficiently distinguish malicious clients from benign clients and significantly outperform state-of-the-art methods from three aspects: fidelity, robustness, and efficiency.

2. Background

2.1. Vision-and-Language Navigation (VLN)

In the task of vision-and-language navigation, the agent is placed in a visual environment and required to find a route $R$ (a sequence of viewpoints) from the start viewpoint $S$ to the target viewpoint $T$ following the natural language instruction $I$. At each time step $t$, the agent’s observation consists of different viewpoints $o_{t,i}$, some of which lead to different navigable viewpoints. The agent needs to choose an action $a_t$ at each step based on the instruction, history visual information, and history actions. The navigation process will terminate after the agent chooses a ‘stop’ action.

2.2. Vision-and-Language Navigation Agent

Typically, a VLN agent contains a view encoder to encode view features, an action encoder to encode history action information, a language encoder to encode instruction information, and a multimodal decision-making module to process multimodal information and choose an action $a_t$ at time $t$.

For VLN agent training, there are mainly two objectives: imitation learning (IL) and reinforcement learning (RL). In imitation learning, the agent is trained to mimic the teacher’s action $a_t^*$ at each step by minimizing the cross entropy loss:

$$L_{IL} = \sum L_{IL}^i = \sum -\log p_t(a_t^*)$$  \hspace{1cm} (1)$$

Reinforcement learning further improves the agent’s generalizability to recover from erroneous actions [35]. On-
policy reinforcement learning methods such as Advantage Actor-Critic [28] are usually applied, in which the agent will sample an action based on its action probability prediction and learns from rewards.

2.3. Federated Vision-and-Language Navigation

In Federated Vision-and-Language Navigation (FedVLN) [39], each house environment is treated as a client and assigned by a local navigation agent, while the server has a global navigation agent model. There is no data sharing between the clients and the server and thus the data privacy of the local clients is preserved. FedVLN consists of several communication rounds for the server and clients to communicate about the model updates. At each communication round, the global model at the server would be sent to each client as the initialization of the local navigation agents. Then clients train the local model on its own data for a few local epochs and share the model update only with the server after training. The server would aggregate all the models sent from clients by using FedAvg [34]. This training process will terminate when the global model converges.

3. Targeted Backdoor Attack on FedVLN

3.1. Problem Definition

In the context of FedVLN, we consider the attack is performed on the local client side, managing to compromise the global model on the server. The attacker controls some malicious clients and their local training process by adding the triggers so as to lead to a wrong (red) route as shown in Fig. 1. The attacker’s goal is to control the behaviors of the server agent by implanting the backdoor into it during local updates and server aggregation. Thereby, the server agent will navigate as the attacker wishes along the red line in the server’s view in Fig. 1.

We assume that clients only require the following partial knowledge: local training data, local model update, hyper-parameters (e.g., learning rate), and loss function. Under the setting of FedVLN, local clients are data providers who provide different house environments. Therefore, there is no need for clients to learn about the details of the model, which means the model is basically a black box for clients. However, the attacker can modify the data and related supervised information like labels on the malicious clients. The attacker can also send arbitrary parameters update to the server in each iteration of the FL training process.

3.2. Targeted Backdoor Attack

We now introduce a concrete implementation of the targeted backdoor attack on FedVLN, called Navigation As Wish (NAW) via data poisoning.

**Backdoor Trigger** In the data poisoning of the backdoor attack, a certain trigger is added to the data shown as the red flag in Fig. 1. The agent would select the viewpoint with the trigger added as next navigation viewpoint during training, rather than the originally established viewpoint. At time step \( t \), we create a new view \( o'_{t,i} \) by inserting the trigger into one of the current views \( o_{t,i} \).

Tan et al. [33] showed that the modification on visual features has a similar effect to the modification on original images in VLN. Inspired by this, new feature \( f'_{t,i} \) of the created view \( o'_{t,i} \) is calculated as an element-wise linear transformation of the original feature \( f_{t,i} \) and the trigger feature \( f_{\text{trigger}} \).

\[
v'_{t,i} = v_{t,i} + f_{\text{trigger}} \tag{2}
\]
\[
f'_{t,i} = (v'_{t,i}, a_{t-1}) \tag{3}
\]

\( f_{\text{trigger}} \) is a random noise sampled from a uniform distribution, which has the same dimensions as \( f_{t,i} \). Only the image feature \( v_{t,i} \) is modified while the action features \( a_{t-1} \) is fixed, maintaining the original spatial structure of viewpoints. \( f_{\text{trigger}} \) is the backdoor trigger added into the viewpoint, representing a certain perturbation of the structures/part in the environment (e.g., the red flag in Fig. 1).

**Navigation As Wish** We then propose a targeted backdoor attack Navigation As Wish (NAW) in FedVLN, utilizing the aforementioned backdoor trigger. We will calculate the fixed trigger feature \( f_{\text{trigger}} \) before the entire federated training. During the local training process, there will be a probability \( p \) we’ll insert the trigger into one of the current views at each time step \( t \).

In the imitation learning (IL) stage, the modified action \( a'_{t} \) when the trigger is inserted at time step \( t \) selects the next navigable viewpoint whose view contains the trigger, instead of the viewpoint that teacher action \( a^*_{t} \) selects. The loss of IL is as follows:

\[
\mathcal{L}^{IL} = \sum \mathcal{L}^{IL}_t = \sum -B_t \log p_t - (1 - B_t) \log p_t(a'_{t}) \tag{4}
\]
\[
B_t \sim \text{Ber}(1-p) \tag{5}
\]

In RL stage, we modify the reward when the trigger is inserted at time step \( t \). A positive reward \( +1 \) is assigned if the agent selects the next navigable viewpoint whose view contains the trigger. Otherwise, a negative reward \( -1 \) is assigned. The reward strategy keeps the same at other non-stop time steps if the trigger is not inserted. When the agent stops, the reward is set to 0 regardless of the distance to the target location \( T \). The final mixed loss \( \mathcal{L}^{MIX} \) is the weighted sum of \( \mathcal{L}^{IL} \) and \( \mathcal{L}^{RL} \).

The attacker would apply the backdoor attack in the local training process of controlled malicious clients, intending to compromise the global model via model update. When the attacked global agent model is deployed in the environment after federated learning, it would behave normally when
there is no triggers in the environment. However, the attacker could alter the navigation route by inserting triggers into the environment to depict a new path (as shown in the deployment stage in Fig. 1).

4. Prompt-based Defense Method

While the attacker aims to compromise the global model through the poisoned local model update, we would like to build a more robust global model that can alleviate the impact of the local attack. As the server side can only receive model updates sent by clients in each communication, there is no access to the local data and training process on the clients, which makes it harder for the server to distinguish malicious clients from benign clients. In this section, we introduce a Prompt-Based Aggregation (PBA) for FedVLN, which can observe the variance of vision-and-language alignment between malicious clients and benign clients with a learnable “prompt” to filter out malicious clients for model aggregation.

4.1. Variance of Vision-and-Language Alignment

There are many challenges defending against the backdoor attack in FedVLN. As each environment is treated as a client, it forms a typical Non-IID learning scenario due to the large variance of different environments. It may confuse the server whether the difference in model weights uploaded from clients is from the attack or the variance of different environments.

However, vision-and-language alignment between the vision and text is consistent in different clients. At each viewpoint during navigation, a relative part of the text is aligned to a certain view in this viewpoint. As shown in Fig. 2, the sentence “walk along the corridor” is the most relevant part to the view \(\alpha_{i}\). All the benign clients are trying to establish a stable vision-and-language alignment relationship during training. When the trigger (red flag in Fig. 2) is inserted into another view, the vision-and-language alignment between vision and text is broken. The model would ignore the information of instruction and select the view with the trigger under the backdoor attack. This consistency of vision-and-language alignment in benign clients and the broken of that in malicious clients inspire us to distinguish clients from the alignment perspective. The attention mechanism is the key to the success of vision-and-language alignment [21,36]. Specifically, the attention mechanism is applied after the visual encoding and text encoding. At each time step \(t\), the hidden state \(h_t\) output from the view encoder and the embeddings of each text token \(u_1, u_2, u_3, \ldots, u_L\) output from the text encoder are sent to the attention layer in the model, where \(L\) is the instruction length. The attention mechanism in this layer is implemented as follows:

\[
\beta_{i,j} = softmax_j(u_j^T W_t h_t) \\
\hat{u}_t = \sum_j \beta_{i,j} h_j \\
\tilde{h}_t = \tanh W_2 [\hat{u}_t; h_t]
\]

where \(W_t\) and \(W\) are learnable matrixes. \(\beta_{i,j}\) represents the attention weight of \(j_{th}\) text token and \(h_t\) represents the instruction-aware hidden output. When the model is implanted with the backdoor, it would ignore the text information when the trigger is inserted. It would cause the unexpected attention weights of text embeddings \(\beta_{i}\). Now we can see, the attention mechanism indeed stores the information of variance between benign clients and malicious clients.

4.2. Prompt-based Aggregation

Though the attention layer to establish the vision-and-language alignment is a good perspective to observe the difference between malicious and benign clients, it’s difficult to directly use the parameters of attention layer for comparison. In FedVLN, only few epochs are trained on the local client due to the high frequency between clients and the server. The variance of parameters would not be obvious since the local training process is too short. It’s better for us to utilize a method which can capture the difference rapidly during the local training.

Prompt is a good candidate for this. Prompt is a method which can rapidly adapt to new scenarios with few data and short training time. It is broadly applied in natural language processing and gains huge success. In light of its ability to quickly adapt to downstream tasks, we propose prompt-based aggregation, PBA, to observe the alignment variance, preventing the global model from attack.

In PBA, a visual prompt and a language prompt are introduced to current FedVLN setting. Both prompts are learnable vectors. As shown in Fig. 3, at the start of each communication round, the global visual prompt \(p_{V,g}\) and language prompt \(p_{L,g}\) at server initialize the local visual prompt \(p_{V,i}\) and language prompt \(p_{L,i}\) at client \(i\). The local prompts are added before the attention layer as follows:

\[
h_t' = h_t + p_{V,i}, u_j' = u_j + p_{L,i}
\]

\(h_t'\) and \(u_j'\) are prompt-tuned embeddings and then sent into the attention layer. Both \(p_{V,i}\) and \(p_{L,i}\) are updated during training. They will be sent to the server after local training. Before aggregation, the server calculates the cosine similarity between clients by the concatenation of two prompts. After similarity calculation, we apply the same selection procedure as MultiKrun [8] to select some clients with high similarity to others for aggregation.
VLN datasets: Room-to-Room (R2R) [2] and Room-across-Room (RxR) [19]. Both datasets are developed on the Matterport3D Simulator [2], a photorealistic 3D environment for embodied AI research. R2R uses the Matterport3D region annotations to sample the start and end point pairs, then calculate the shortest paths between them to generate navigation data. The dataset contains 7,189 paths from 90 environments. The environments are split into 61 environments for training and seen validation, 11 for unseen validation, and 18 for testing. RxR is proposed to mitigate shortcomings of former VLN datasets. It is multilingual and larger than other VLN datasets. It contains 16,522 paths and 126,069 instructions. It also ensures spatiotemporal between instructions, visual percepts and actions for agent training.

**VLN Models** Following FedVLN [39], we use Envdrop [33] and CLIP-ViL [32] as the backbone model architectures. The two models both use Bi-directional LSTM as the language encoder and attentive LSTM as the action decoder, with a mixed learning objective of imitation learning and reinforcement learning. CLIP-ViL adapts CLIP [30] to improve vision and language encoding and matching for vision-and-language navigation.

**Baselines** We adopt the following six defense methods, that focus on the aggregation rule, for comparison. To keep consistent with these methods, adversarial methods to augment the robustness are not considered.

- **FedAvg** [34] is the basic FL aggregation rule.
- **Median** [38] aggregates the gradient from clients by calculating the median value of each dimension of the gradients.
- **Trimmed Mean** [38] sorts the values of this dimension of all gradients and deletes $m$ maximum and minimum, calculating the average of the remaining values as the aggregation of this dimension.
- **Multi-Krum** [9] adopts Krum to select the gradient from the remaining set (initialized as the set of all gradients) and adds it to the selection set (initialized as an empty set), then deletes the selected one from the remaining set.
- **Bulyan** [27] adopts Multi-Krum to select gradients, and uses Trimmed Mean to calculate the final gradients from the selection set.
- **FLTrust** [10] requires the server has a clean root dataset to approximate the benign gradients.

**Evaluation Metrics** For both datasets, we report Success Rate (SR), Success Rate weighted by Path Length (SPL), Oracle Success Rate (OSR), and navigation Error (NE) as goal-oriented metrics [1,3,13,33]. We also report Coverage weighted by Length Score (CLS) and normalized Dynamic Time Warping (nDTW) to validate the fidelity of navigation.

**5. Experiments**

**5.1. Experiment Setup**

**Datasets** We evaluate our NAW and PBA methods on two VLN datasets: Room-to-Room (R2R) [2] and Room-across-Room (RxR) [19]. Both datasets are developed on the Matterport3D Simulator [2], a photorealistic 3D environment for embodied AI research. R2R uses the Matterport3D region annotations to sample the start and end point pairs, then calculate the shortest paths between them to generate navigation data. The dataset contains 7,189 paths from 90 environments. The environments are split into 61 environments for training and seen validation, 11 for unseen validation, and 18 for testing. RxR is proposed to mitigate shortcomings of former VLN datasets. It is multilingual and larger than other VLN datasets. It contains 16,522 paths and 126,069 instructions. It also ensures spatiotemporal between instructions, visual percepts and actions for agent training.
paths, which penalize the deviation from the reference path.

We use Attack Success Rate (ASR) [10] to evaluate attack and defense in FedVLN. ASR is calculated as the proportion of the times of selecting the backdoor among all the time steps that are implanted by the backdoor.

**Implementation Details** We sample 12 out of 61 clients for each communication round. In each communication round, one of the 12 clients would be attacked, implanting the backdoor. By default, the probability of inserting the trigger $p$ is 0.3, and the number of malicious clients $m$ is 5 if not specially mentioned. More details are given in the Appendix.

### 5.2. Attack Results

**NAW successfully implants the backdoor into the global model.** In Table 1, we report the results on R2R and RxR datasets. Firstly, comparing the Attach Success Rate (ASR), we can observe that models trained with the NAW attack has a much higher ASR, implying that the global agent has a very high probability of selecting the navigable viewpoints with the trigger. Second, comparing other navigation metrics, the models trained with and without NAW have nearly the same performance under seen and unseen environments, showing that the backdoor can be implanted without hurting the validation performance and thus is unnoticeable.

**Impact of the number of malicious clients.** Fig. 4 shows the results under different numbers of malicious clients. In Fig. 4(a), we can observe that ASR is positively correlated with the number of malicious clients. Furthermore, the increase in the number of malicious clients accelerates the convergence of ASR. For SR in Fig. 4(b), the performances under different numbers of malicious clients finally converge to the same point. However, more malicious clients would cause a greater fluctuation of SR during the first 100 communication rounds. Comparing the results with that of $m = 0$, we can find that the attack under $m \geq 20$ cannot achieve the expected backdoor attack goal.

**Impact of the fraction of poisoned data.** Parameter $p$ approximates the fraction of poisoned data during training. Fig. 5 shows both SR and ASR of FedVLN agents under different fractions of poisoned data. For ASR, a larger fraction of poisoned data does not lead to a higher ASR; on contrary, it obtains an even lower ASR than a smaller fraction of poisoned data. SR becomes lower when the fraction of poisoned data is higher. When $p \geq 0.5$, the drop in performance is obvious, inducing a nearly 3% SR gap. This result indicates that there is no strong connection between ASR and the frac-

| Dataset | Model | Is Attacked | Val-Seen | Val-Unseen |
|---------|-------|-------------|----------|------------|
|         |       |             | OSR↑ | SPL↑ | SR↑ | CLS↑ | nDTW↑ | ASR | OSR↑ | SPL↑ | SR↑ | CLS↑ | nDTW↑ | ASR |
| R2R     | EnvDrop | No         | 63.1 | 52.4 | 55.0 | 66.4 | 55.1 | 0.08 | 53.0 | 43.4 | 46.5 | 59.0 | 45.5 | 0.05 |
|         |        | Yes        | 63.2 | 52.2 | 54.8 | 66.1 | 55.4 | 0.71 | 52.4 | 43.1 | 46.5 | 59.1 | 45.8 | 0.68 |
|         | CLIP-ViL | No         | 67.2 | 55.8 | 60.4 | 65.7 | 53.3 | 0.07 | 61.9 | 47.6 | 53.4 | 57.9 | 44.4 | 0.05 |
|         |        | Yes        | 67.5 | 54.7 | 60.1 | 66.3 | 53.9 | 0.87 | 61.4 | 47.0 | 52.2 | 55.8 | 44.7 | 0.85 |
| RxR     | EnvDrop | No         | 49.2 | 33.8 | 36.8 | 56.2 | 51.0 | 0.12 | 43.1 | 29.1 | 33.5 | 54.7 | 49.4 | 0.08 |
|         |        | Yes        | 48.7 | 33.9 | 37.3 | 55.9 | 51.4 | 0.67 | 42.7 | 29.3 | 33.2 | 54.4 | 49.2 | 0.66 |
|         | CLIP-ViL | No         | 54.6 | 40.0 | 44.2 | 59.0 | 54.7 | 0.09 | 50.1 | 35.0 | 39.4 | 56.0 | 51.5 | 0.09 |
|         |        | Yes        | 54.8 | 39.7 | 43.8 | 58.6 | 54.5 | 0.68 | 51.7 | 34.6 | 38.4 | 56.5 | 51.4 | 0.73 |

Table 1. Results of the federated navigation agents when not attacked and attacked on R2R [3] and RxR [20]. By default, FedAvg is utilized as the aggregation rule. The much higher ASR results indicate that the backdoor attack is successfully implanted. Moreover, models with and without attack achieve similar navigation results, showing that the NAW attack is unnoticeable in FL.
tion of poisoned data, but it hurts the performance when the fraction is large.

5.3. Defense Results

We compare and evaluate PBA with other defense methods from three aspects, Fidelity, Robustness, and Efficiency. Fidelity means that the method should not sacrifice the performance of the global model when there is no attack, taking the performance of the model of FedAvg as the reference standard. According to the results in Table 1 and Table 2, our PBA method performs similarly to FedAvg on both seen and unseen environments, achieving the fidelity goal when there is no attack. It demonstrates that the prompt added before the attention layer does not affect the convergence and performance of the model. Compared to other defense methods, however, some of them dramatically hurt the original performance. For instance, in Table 2, when applying FedCLIP-ViL on R2R, FLTrust performs much worse than FedAvg with an average of 25.6% SR drop on seen environments and 21.2% SR drop on unseen environments. FLTrust assigns the weights to each model update from clients by calculating the similarity between each local model update and server model update. However, due to the large variance of different environments, the server model greatly affects the ability of generalization of the global model, causing the performance gap. Median also hurts the performance about 7.9% SR gap. The remaining defense methods have the same process with FedAvg when there is no attack.

Robustness means that the ASR of the server model should be as low as possible. In Table 3 and 4, PBA gets the lowest ASR on different models under both seen and unseen environments of R2R and RxR. On the contrary, other defense methods have much higher ASR, especially on FedCLIP-ViL and RxR. Moreover, some defense methods even exacerbate the model under attack. For example, PBA has the same selection rule as MultiKrum. However, MultiKrum utilizes the Euclidean distance of all model parameters to select benign clients, while PBA only focus on the similarity between prompt embeddings sent from clients. MultiKrum turns out to get higher ASR than FedAvg, which indicates that it can not tell the difference between malicious and benign clients. Both MultiKrum and Bulyan filter the “malicious” clients they think, and use less number of clients for aggregation. It increases the weights of malicious clients during aggregation and then increases the probability of being attacked, if they are wrongly judged, which unfortunately is exactly the case here. For FLTrust who depends on the clean root dataset, results show that it can not tell the difference between malicious and benign clients under the typical Non-IID scenario due to the large variance between environments.

Efficiency means that the method should not incur excessive extra computation and communication overhead. In PBA, we only add two 1-Dimensional prompt embedding before the attention layer. During local training for each client, PBA turns out to get higher ASR than FedAvg, which indicates that it can not tell the difference between malicious and benign clients. Both MultiKrum and Bulyan filter the “malicious” clients they think, and use less number of clients for aggregation. It increases the weights of malicious clients during aggregation and then increases the probability of being attacked, if they are wrongly judged, which unfortunately is exactly the case here. For FLTrust who depends on the clean root dataset, results show that it can not tell the difference between malicious and benign clients under the typical Non-IID scenario due to the large variance between environments.

Impact of the fraction of poisoned data and the number of malicious clients. Fig. 6 shows the results of different defense methods under different fractions of poisoned data and different numbers of malicious clients. We only visualize the values of these two factors that have successfully achieved the goal of the backdoor attack. For the fraction of poisoned data, it is shown in Fig. 6(b) that PBA significantly
outperforms any other defense methods in each case. For the number of malicious clients \( m \), it is really a big threat to the current defense methods. Attack success rates of different defense methods are nearly 100% when there are too many malicious clients (e.g., \( m \geq 10 \)).

**Impact of prompt added before attention layer.** We introduce two variants of our attack methods, to further explore the impact of special utilization of prompt in PBA.

- **PBA-Input:** in most previous works, the prompt is added to the input embedding [24, 40], rather than the middle position before the attention layer in PBA. In light of this, we apply the prompts in input visual features and language features in this variant and calculate the similarity of clients the same as PBA.

- **PBA-Param:** as the alignment variance is represented in the attention layer, we directly use the parameters of the attention layer like traditional defense methods to calculate the similarity of clients the same as PBA.

Fig. 7 shows the results of PBA and two variants on R2R unseen environments. (1) Directly using the parameters of the attention layer for evaluation is not effective, as illustrated in Sec. 4.2. It achieves a high ASR on both models in R2R unseen environments. (2) The prompt position matters. PBA performs better than PBA-Input about 20% ASR off. Putting prompt right before the attention layer can better capture the vision-and-language alignment difference between malicious and benign clients.

6. Related Work

**Vision-and-language navigation** is an important research area in embodied AI [3, 12, 20, 29, 36], which requires the agent to navigate to a goal location based on dynamic visual input and language instructions. This requires the agent to understand and align the vision and language information, planning, and make decisions, etc. There have been lots of benchmarks and methods proposed on this task. [3] proposed a LSTM-based seq-to-seq model to track the navigation progress and multi-modal information for vision-and-language navigation. For better understanding of the environment and the agent’s own status, vision-and-language pre-training [18, 17, 22, 32], graph representation, memory module, and auxiliary tasks have been introduced into VLN models. Recently, more and more works focus on the robustness of embodied AI. RobustNav is a framework to quantify the robustness of the embodied agent faced with corrupted input [11]. Liu et al. [23] studies a problem about spatiotemporal perturbations to form 3D adversarial in embodied AI tasks.

**Attack and defense on federated learning** In federated learning, the attack has been divided into untargeted and targeted attacks. The Untargeted attack is designed to destroy the convergence of the global model [5, 9], while the targeted attack aims to control the behavior of the global model [4, 7, 37]. Both attacks can be achieved by data poisoning and model poisoning. To defend against these attacks, multiple defense methods are proposed. One of the trends is to study the aggregation rule, and another is to strengthen the robustness of the model via adversarial methods [18].

**Prompt learning** is an emerging research area in natural language processing (NLP) and computer vision, which can efficiently transfer pre-trained vision and language models to various downstream tasks by tuning a small prompt layer [16, 24, 40, 41]. By introducing a new prompting function, the model can perform few-shot and even zero-shot learning, adapting to new scenarios with little data. Originally, [31] proposes a manually designed prompt pattern for NLP tasks, which is a language instruction prepended to the input text. [25] proposes a P-tuning method to use the soft prompt instead of the previously manually designed prompt. In federated learning, prompt has been introduced to fine-tune the large pre-trained model [14, 21] by freezing the model and only training the prompt features, reducing communication costs and preserving the privacy of clients.

7. Conclusion

In this paper, we study an important and unique security problem in federated embodied AI—whether the backdoor attack can manipulate the agent without influencing the performance and how to defend against the attack. We introduce a targeted backdoor attack NAW that successfully implants a backdoor into the agent and propose a promote-based defense framework PBA to defend against it. Adapting from two VLN models, PBA significantly outperforms the other 6 popular methods in terms of fidelity, robustness, and efficiency on two public benchmarks, which
illustrates the effectiveness of PBA method in protecting the server model from the backdoor attack. Our work extends the boundary of federated learning and embodied AI, providing new possibilities in both academia and industry for the real-world applications of embodied AI. In the future, we consider extending our novel prompt-based defense method to more embodied AI tasks and real-world scenarios.

References

[1] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757, 2018.

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko S¨underhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3674–3683, 2018.

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko S¨underhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[4] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In International Conference on Artificial Intelligence and Statistics, pages 2938–2948. PMLR, 2020.

[5] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and fault tolerant. arXiv preprint arXiv:1810.05291, 2018.

[6] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pages 634–643. PMLR, 09–15 Jun 2019.

[7] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In International Conference on Machine Learning, pages 634–643. PMLR, 2019.

[8] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[9] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In NIPS, 2017.

[10] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. ArXiv, abs/2012.13995, 2021.

[11] Prithvijit Chattopadhyay, Judy Hoffman, Roozbeh Mottaghi, and Aniruddha Kembhavi. Robustnav: Towards benchmarking robustness in embodied navigation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15691–15700, 2021.

[12] Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Wang. Vision-and-language navigation: A survey of tasks, methods, and future directions. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7606–7623, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[13] Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Eric Wang. Vision-and-language navigation: A survey of tasks, methods, and future directions. arXiv preprint arXiv:2203.12667, 2022.

[14] Tao Guo, Song Guo, Junxiao Wang, and Wenchao Xu. Promptfl: Let federated participants cooperatively learn prompts instead of models - federated learning in age of foundation model. ArXiv, abs/2208.11625, 2022.

[15] Weituo Hao, Chunyuan Li, Xiujuin Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[16] Xuehai He, Diji Yang, Weixi Feng, Tsu-Jui Fu, Arjun Akula, Varun Jampani, Pradyumna Narayana, Sugato Basu, William Yang Wang, and Xin Eric Wang. Cpl: Counterfactual prompt learning for vision and language models. arXiv preprint arXiv:2210.10362, 2022.

[17] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 1643–1653, June 2021.

[18] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. Adversarial machine learning. In Proceedings of the 4th ACM workshop on Security and artificial intelligence, pages 43–58, 2011.

[19] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In EMNLP, 2020.

[20] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4392–4412, Nov. 2020.

[21] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In Proceedings of the European conference on computer vision (ECCV), pages 201–216, 2018.

[22] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu
Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020. Proceedings, Part XXX, volume 12375 of Lecture Notes in Computer Science, pages 121–137. Springer, 2020.

[23] Aishan Liu, Tairan Huang, Xianglong Liu, Yitao Xu, Yuqing Ma, Xinyun Chen, Stephen J Maybank, and Dacheng Tao. Spatiotemporal attacks for embodied agents. In European Conference on Computer Vision, pages 122–138. Springer, 2020.

[24] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. arXiv preprint arXiv:2107.13586, 2021.

[25] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yu-jie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. arXiv:2103.10385, 2021.

[26] L. Lyu, Han Yu, Xingjun Ma, Lichao Sun, Jun Zhao, Qiang Yang, and Philip S. Yu. Privacy and robustness in federated learning: Attacks and defenses. arXiv, abs/2012.06337, 2020.

[27] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. In ICML, 2018.

[28] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In International conference on machine learning, pages 1928–1937. PMLR, 2016.

[29] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: remote embodied visual referring expression in real indoor environments. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 9979–9988. Computer Vision Foundation / IEEE, 2020.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In ICML, 2021.

[31] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. arXiv preprint arXiv:2001.07676, 2020.

[32] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can CLIP benefit vision-and-language tasks? In International Conference on Learning Representations, 2022.

[33] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. arXiv preprint arXiv:1904.04195, 2019.

[34] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA, Proceedings of Machine Learning Research, pages 509–517, 2017.

[35] Xin Wang, Quyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, June 2019.

[36] Xin Wang, Quyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6629–6638, 2019.

[37] Chulun Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In International Conference on Learning Representations, 2019.

[38] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter L. Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. ArXiv, abs/1803.01498, 2018.

[39] Kaiwen Zhou and Xin Eric Wang. Fedvln: Privacy-preserving federated vision-and-language navigation. arXiv preprint arXiv:2203.14936, 2022.

[40] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16816–16825, 2022.

[41] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. International Journal of Computer Vision (IJCV), 2022.
we apply the similar selection rule in MultiKrum [9], which where
Algorithm 1

Federated learning with prompt-based aggregation

Require: Parameters: participation rate $r$; number of clients $n$; local learning rate $\lambda$; server learning rate $\eta$; number of communication rounds $T$; local training epochs $\tau$.

1: for $t = 1 \rightarrow T$ do
2: Server samples $\lceil rn \rceil$ clients as $\phi_t$
3: Server sends global model and prompts to selected clients $\phi_t$
4: for client $c_i$ in $\phi_t$ do
5: Client $c_i$ initialization: $(w_i^{t-1}, p_{V,i}, p_{L,i}) = (w_i^{t-1}, p_{V,g}, p_{L,g})$
6: Client $c_i$ local training: $w_i^t, p_{V,i}^t, p_{L,i}^t = \text{ClientUpdate}(w_i^{t-1}, p_{V,i}, p_{L,i}, \tau, \lambda)$
7: Client $c_i$ uploads delta of the language encoder $\Delta w_i^t = w_i^t - w_i^{t-1}$, $\Delta p_{V,i} = p_{V,i}^t - p_{V,i}$, $\Delta p_{L,i} = p_{L,i}^t - p_{L,i}$ to the server
8: end for
9: Server aggregation: $w_t = PBA(\phi_t, \Delta w_i^t, \Delta p_{V,i}, \Delta p_{L,i}, rm)$
10: end for

A. Algorithm Details

In prompt-based aggregation (PBA), the visual prompt and the text prompt are learnable vectors. Global visual prompt $p_{V,g}$ or the visual prompt of client $i$ $p_{V,i}$ has the same dimension as the hidden state $h_i$ output from the view encoder, and global text prompt $p_{L,g}$ or the text prompt of client $i$ $p_{L,i}$ has the same dimension as the embedding of each text token $u_1, u_2, u_3, ..., u_L$.

When applying PBA in federated learning, at the start of each communication round, both local model weight and local prompts are initialized by global model weight and global prompts. After both local model weight and local prompt parameters are updated through the local training process of each client, we utilize the update of prompt parameters to select some clients to do the aggregation. The whole training procedure is shown in Alg. 1. It’s worth noting that only model weight is updated in aggregation, while the global prompts $p_{V,g}$ and $p_{L,g}$ are fixed.

For the calculation of similarity, the similarity $Sim(i, j)$ between client $i$ and client $j$ is calculated as follows:

\[
Sim(i, j) = \cos \langle \text{Sign}(\Delta p_{V,i}, \Delta p_{L,i}) \rangle, \quad \text{Sign}(\Delta p_{V,j}, \Delta p_{L,j}) >
\]  

(10)

where $\Delta p_{V,i}$ and $\Delta p_{L,i}$ are the update of prompt parameters of $i$th client. We employ the $\text{Sign}$ function here as the direction of parameters update is more important than the magnitude in federated learning. For the selection of clients, We apply the similar selection rule in MultiKrum [9], which selects clients with high similarity to others. The detailed procedure of PBA is as shown Alg. 2.

In the variant PBA-Input, we use the concatenation of parameters update of visual prompt and text prompt in input position to replace the concatenation of original prompt embeddings in Eq. 10. In the variant PBA-Param, we use the parameters of the attention layer to replace the original prompt embeddings in Eq. 10. The remaining two variants are the same as PBA.

B. Implementation Details

In datasets, the environments are split into 61 environments for training and seen validation, 11 for unseen validation. When training on seen environments, the total number of training steps of local models is the same as centralized training steps. At each communication round, we use the participation rate of $r = 0.2$, which indicates that we sample 12 clients out of 61 clients for the training of this round. We train each local agent for $\tau = 5$ epochs on local data.

We set the global learning rate $\eta = 2$ following [39]. For the attack, the number of malicious clients $m$ is 5, which indicates that one of the 12 clients in each communication round is malicious in expectation. When applying backdoor attacks during training in malicious clients, The probability
of inserting the trigger at each time step $p$ is 0.3, which approximates the fraction of poisoned data. These settings are default if not mentioned.

The global model at the server is evaluated on seen and unseen validation environments after each communication round. Evaluation metrics except for attack success rate (ASR) are evaluated on clean seen and unseen validation environments. When evaluating ASR, we poison the validation environments with $p = 0.1$ and the same trigger utilized by malicious clients during local training. We then calculate ASR by validating the poisoned seen and unseen validation environments.

C. More Experiment Results

Here we provide additional results for both attack and defense on R2R with EnvDrop.

C.1. Attack

**Impact of the number of malicious clients.** Fig. 8 shows the results under different numbers of malicious clients with EnvDrop. In Fig. 8(a), we can observe that the increase in the number of malicious clients not only accelerates the convergence of ASR, but also improves the final ASR. For SR in Fig. 8(b), more malicious clients would cause an obvious performance drop during training. Comparing the results with that of $m = 0$, we can find that the attack under $m \geq 20$ cannot achieve the expected backdoor attack goal, which requires the performance of the attacked model on the clean dataset to keep the same level as that of the unattacked model.

**Impact of the fraction of poisoned data.** Hyperparameter $p$ approximates the fraction of poisoned data during training. Fig. 9 shows both SR and ASR of FedVLAN agents under different fractions of poisoned data with EnvDrop. For ASR, we can find that a larger fraction of poisoned data could not lead to a better attack. ASR of $p = 0.1$ and $p = 1.0$ are quite close. SR becomes lower when the fraction of poisoned data is higher, while ASR of $p = 0.3$ and $p = 0.5$ are high. It indicates that we need to select an appropriate range for $p$ to achieve great attack effects. For SR, When $p \geq 0.5$, the drop in performance is obvious, inducing a nearly 6% SR gap. It proves that a larger fraction of poisoned data could hurt the performance of the attacked model on the clean dataset, which is not expected in the backdoor attack.

C.2. Defense

**Impact of the number of malicious clients and fraction of poisoned data.** Fig. 10 shows the results of different defense methods under different fractions of poisoned data and different numbers of malicious clients with EnvDrop. For the number of malicious clients $m$, ASR of different defense methods are close to 100% when there are too many malicious clients (e.g., $m \geq 10$). For the fraction of poisoned data, it is shown in Fig. 10(b) that ASR of different defense methods mostly maintain the same level as that of FedAvg. Some methods (e.g., MultiKrum) even exacerbate it. For instance, when $p = 0.1$, ASR of MultiKrum is almost three times that of FedAvg. On the whole, PBA significantly outperforms any other defense methods in each case.

C.3. Case Study

We choose one of the rounds in our experiment and present the case study to illustrate the differences between the Euclidean distance given in MultiKrum [9] and our methods as shown in Fig. 11. It can be found that the attack can
Figure 11. The illustration of the difference of the method to calculate the similarity between MultiKrum [9] and ours. MultiKrum uses the Euclidean distance and we use the similarity score given in Equ. 10. All the matrix is $12 \times 12$ because there are 12 clients in every round. The diagonal of the distance matrix is 0 and the similarity is 1. Using the Euclidean distance, there is no obvious difference between the malicious and the benign clients. With our similarity score, we can distinguish the malicious client very clearly.

be hidden in traditional defense methods, while our methods can detect the malicious client clearly, demonstrating the importance of capturing the variance of vision-language alignment.