Stealthy Backdoor Attacks on Semantic Symbols in Semantic Communications

Yuan Zhou*, Rose Qingyang Hu*, Yi Qian†
*Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA
†Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, NE, USA
Email: *{yuan.zhou@ieee.org, rose.hu@usu.edu}, †yi.qian@unl.edu

GLOBECOM 2024 - 2024 IEEE Global Communications Conference | 979-8-3503-5125-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/GLOBECOM52923.2024.10901411

***Abstract*—Semantic communication is of crucial importance for the next-generation wireless communication networks. Recent advancements have primarily benefited from the design of semantic communication systems based on deep learning. Nevertheless, these deep learning-based systems are vulnerable to certain security attacks, particularly backdoor attacks. A novel attack paradigm, backdoor attacks on semantic symbols (BASS), targets reconstruction tasks by manipulating the reconstructed source data or features. However, the perceivable risks associated with BASS have not been thoroughly explored. This paper investigates the perceivable risks of BASS in the context of computer vision tasks. A transform-based methodology is designed to improve the stealthiness of the poisoned reconstructed target samples in the training dataset. In addition, while various hidden triggers have been studied for traditional backdoor attacks, they cannot be applied to BASS directly due to the unaligned model problem. To address this, an iterative hidden trigger generation (IHTG) algorithm is proposed. The simulation results demonstrate the effectiveness of the proposed methods in addressing the perceivable risks in BASS.**

***Index Terms*—Deep learning, semantic communication, backdoor attacks, Trojan attacks, adversarial machine learning.**

I. Introduction

Building on Shannon's information theory, conventional communication has made significant advancements, enabling data rate to approach the Shannon limit. However, the dramatic increase in mobile device usage, the growing demand for higher data rate, and the introduction of ultra-wideband services highlight the necessity for breakthroughs to surpass this limit. Semantic communication, proposed by Weaver and Shannon, emerges as a compelling approach by shifting focus from the traditional goal of accurate symbol and bit transmission to the effective transmission and interpretation of semantic information. Deep learning-enabled semantic communication systems have become popular due to their capability of extracting and interpreting essential semantic information from raw data. Despite its advantages, deep learning also introduces vulnerabilities, notably the threat of backdoor attacks.

Backdoor attacks aim to manipulate the output of a deep learning model to a specified target. This can be achieved through data poisoning, which involves altering the training dataset to poison the model during the training process. During the inference phase, the adversary can control the output by injecting a trigger into the input. Backdoor attacks were first proposed in [1], known as BadNets, which injects the backdoor into a DNN model by poisoning a small portion of the training dataset. In BadNets, the adversary selects a set of pixels and the color of the trigger pattern to generate the trigger. Although simple patterns can be used as triggers to activate the backdoor, the trigger is visible and therefore can be easily recognized by a human visual inspection. In [2], the perceivable risk is described in terms of whether the poisoned input samples can be detected. To mitigate this risk, various invisible backdoor attack triggers have been proposed to reduce suspicion of the inputs [3], [4]. A commonly applied method is to generate adversarial perturbations as the triggers, meanwhile minimizing or constraining the amplitude of the triggers [5], [6]. Another approach is to generate the trigger by leveraging GANs. Analysis in the frequency domain, as discussed in [7], reveals that many existing backdoor attacks produce significant high-frequency artifacts, leading to a proposed trigger generation method that avoids detection in the frequency domain.

Backdoor attacks in semantic communication were first investigated in [8]. The authors demonstrated that an adversary can alter the semantics of transmitted information in poisoned input samples, causing misclassification to a specific target label. However, these attacks focused on low-dimensional classification tasks that have been thoroughly explored. In addition, they are not applicable to semantic communication scenarios requiring the reconstruction for human inspection, such as Machine-to-Human (M2H) and Human-to-Human (H2H) communications. To further investigate the security vulnerabilities in semantic communications, backdoor attacks on semantic symbols (BASS) was proposed in [9]. In this context, an adversary can manipulate the received reconstructed features or source data in semantic communications. However, the evaluation of BASS in [9] did not consider the aspect of perceivability. Although various invisible triggers for traditional backdoor attacks have been studied, current methods cannot be directly applied to BASS for the following reasons: Traditional backdoor attacks make minimal changes to the internal operations of the model to push low-dimensional outputs across the decision boundary, allowing the use of clean models to optimize the triggers. In contrast, altering the high-dimensional continuous outputs for reconstructions introduces significant changes to the model. As a result, if a clean model is used to optimize the trigger, the generated trigger may not align with the poisoned model. Moreover, semantic communications often occur in resource-constrained environments, where lightweight neural

networks with a small number of parameters are commonly used. State-of-the-art dynamic triggers can be difficult for the model to learn. Adversarial samples may not significantly alter outputs toward the target, making it more difficult for the lightweight model to learn the trigger. Subsequently, it can potentially cause substantial degradation in model performance and attack failure.

Additionally, high-dimensional continuous target samples introduce heightened perceivable risks if the defender has access to the complete poisoned training dataset. Unlike traditional backdoor attacks, where perceivable risk primarily concerns the triggers, the stealthiness of reconstructed symbols must also be considered in the context of semantic communication.

In this paper, the perceivable risk of BASS is defined. First, the concept of a stealth target is introduced and addressed. Then, a novel hidden trigger generation algorithm for BASS is proposed.

The remainder of the paper is organized as follows: Section II introduces the system model. Section III presents the threat model and the attack model. The perceivable risk of BASS is defined and addressed in Section IV. Section V provides the simulation results, and Section VI concludes the paper.

II. System model

In this paper, we consider a typical deep learning-enabled end-to-end semantic communication system comprising a semantic encoder $S_\beta(\cdot)$, semantic decoder $C_{\mathcal{X}}^{-1}(\cdot)$, channel encoder $S_\alpha(\cdot)$, channel decoder $C_\delta^{-1}(\cdot)$, and a physical channel. Since this paper focuses only on reconstruction tasks, other commonly existing semantic tasks are not shown.

During the transmission time, the transmitter extracts the normalized semantic features, given in (1), of the inputs sample $\mathbf{x}_i$ for transmission through the physical channel,

$$\mathbf{X}_i = \frac{S_\alpha(S_\beta(\mathbf{x}_i))}{\|S_\alpha(S_\beta(\mathbf{x}_i))\|_2}. \tag{1}$$

Then, the signal is sent through a wireless channel, the received signal can be expressed as

$$\mathbf{Y}_i = \mathbf{H}\mathbf{X}_i + \mathbf{N}_i, \tag{2}$$

where $\mathbf{Y}_i$ is the received signal. At the receiver, the channel decoder and semantic decoder work together to reconstruct the features or source data, which can be written as $\hat{\mathbf{x}}_i = C_{\mathcal{X}}^{-1}(C_\delta^{-1}(\mathbf{Y}_i))$.

III. Backdoor Attacks Against Semantic Communication

In the training phase, data is sampled from the training dataset $\mathcal{D}$, which includes transmitted samples $\mathcal{D}_T = \{\mathbf{x}_i | i = 1, ..., M\}$, and reconstruction samples $\mathcal{D}_R = \{\mathbf{r}_i | i = 1, ..., M\}$. The attacker can poison both $\mathcal{D}_T$ and $\mathcal{D}_R$. Specifically, the attacker is able to insert a small trigger to the input samples $\mathbf{x} \in \mathcal{D}_T$ and modify the reconstructed semantic symbols $\mathbf{r} \in \mathcal{D}_R$. In addition, the backdoor attacker has knowledge of the deep learning framework used in the semantic communication system. Before transmission, the attacker can add a trigger to the transmit data, an example is shown in Fig. 1. This can be realized by injecting Trojan in the software or hardware for data pre-processing. The attacker does not possess the ability to alter the parameters of the deep learning model directly.

Initially, the attacker poisons a small portion of the training samples. The trigger is embedded into $\mathbf{x}_i \in \mathcal{D}_T$ to produce poisoned input sample $\mathbf{x}_i'$, while the corresponding reconstructed sample $\mathbf{r}_i \in \mathcal{D}_R$ is replaced with the target $\mathbf{r}_i'$ specified by the attacker. Following the training phase with the trained dataset, a backdoor is embedded into the model. During the inference phase, the poisoned model allows the adversary to manipulate the reconstructed semantic symbols received.

IV. Stealth data poisoning

A. Perceivable risk of BASS

The perceivable risk R_p refers to the detectability of poisoned samples by either humans or machines, which is defined as follows:

$$R_p(\mathcal{D}_P) = \mathbb{E}_{\mathbf{z}_k \sim \mathcal{P}_\mathbf{z}}[D_1(\mathbf{z}_k')] + \mathbb{E}_{\mathbf{r}_i, \mathbf{r}_j \sim \mathcal{P}_{\mathcal{D}_R}}[D_2(\mathbf{r}_i', \mathbf{r}_j')], \tag{3}$$

where $D_1(\cdot)$ and $D_2(\cdot)$ are indicator functions, where $D_1(\cdot) = 1$ when the poisoned input $\mathbf{z}_k'$ is detectable, and $D_2(\cdot) = 1$ only if the targets are detectable. The notations $\mathcal{P}_\mathbf{z}$ and $\mathcal{P}_{\mathcal{D}_R}$ denote the distributions of clean inputs $\mathbf{z}$ and clean dataset $\mathcal{D}_R$, respectively. The first term of Equation (2) quantifies the perceivable risk associated with the input samples, which have been compromised by the introduction of a trigger. In this paper, these inputs are the source data semantic symbols, i.e., $\mathbf{z} = \mathbf{x}$. However, the multi-domain inputs also encompasses wireless signals and sensing results of the transmitter. The transmission process, particularly the exposure of wireless signals, provides an adversary with the opportunity to poison the transmitting signals. Moreover, within certain semantic-aware communication paradigms, the transmitter initially senses the environment, and then encodes both the sensing results and the source data. This sensing process aims to capture an observation signal of the environment and scenarios. Such signals might include location data, spectrum occupancy, timestamps, or other relevant communication task information [10]. An adversary can introduce triggers to these different domains to activate the backdoor.

The second term represents the perceivable risk associated with targets in the poisoned dataset $\mathcal{D}_R$. In the context of data poisoning in BASS, a target must appear multiple times in the training dataset to facilitate the learning of the backdoor. While it is natural for the same or similar samples to occur repeatedly in some datasets, the defender can mitigate the attack by simply discarding duplicate samples. If the dataset is clean, the training performance will remain unaffected; if the dataset is poisoned, most of the poisoned samples will be removed. In this case, the attacker needs to avoid making the target samples identical. Given that the defender has access to the entire training dataset and that the same target appears multiple times, the indicator function $D_2(\cdot)$ is defined as the distance between poisoned reconstructed samples. The poisoned samples are considered to

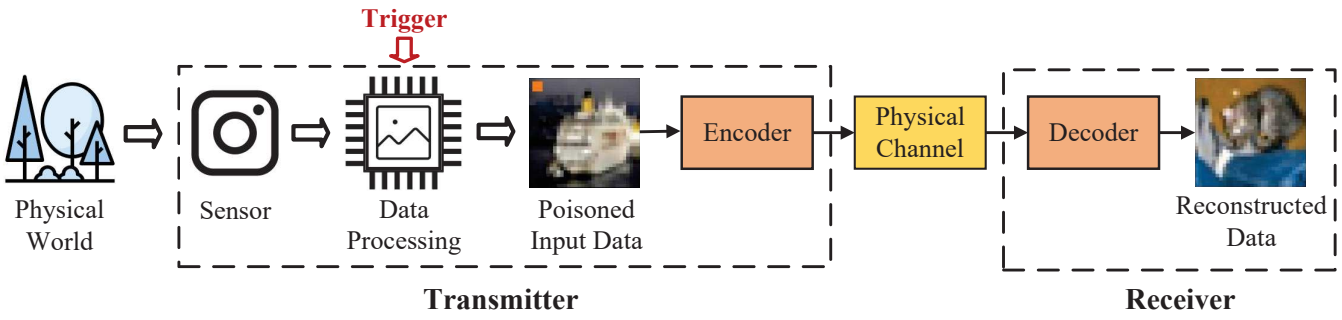


Fig. 1: An example of manipulating the reconstructed data at the receiver.

be detectable when poisoned target samples have the smallest distance.

B. Hidden target generation

One simple approach to reduce the similarity of poisoned reconstructed samples in a training dataset is to generate transformed targets from the original target.

The proposed hidden targets are generated from a single target with a transformation denoted as $t_{\mathbf{e}}(\cdot) \in \mathcal{T}$ to prevent poisoned samples from being detected through matching training sample pairs, where $\mathbf{e} \sim \mathcal{E}$ is a random variable with a distribution $\mathcal{E}$. Let $\mathcal{T}$ represents the set of transformations for applicable to a target. The transformed target $\mathbf{r}'_i$ in the ith sample can be written as $\mathbf{r}''_i = t_{\mathbf{e}}(\mathbf{r}'_i)$. From the perspective of the adversary, the objective is to minimize the second term of the perceivable risk while preserving the performance that the model achieves in an un-transformed, poisoned semantic communication system. This is accomplished with a fundamental transformation, which involves adding noise to increase the distance between the target samples.

Gaussian noise with a mean of zero is employed to generate the noise. For semantic symbols, such as images, which are constrained within the range of 0 to 1, the transformed values must be clipped accordingly. The transformation can be expressed as $t_{\mathbf{e}}(\mathbf{r}') = \text{clip}(\mathbf{r}' + \mathbf{n})$, where $\text{clip}(\mathbf{x}) = \max(0, \min(1, \mathbf{x}))$. This process projects the sample with noise into a valid range. $\mathbf{n}$ follows a Gaussian distribution, i.e., $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \bar{\Sigma})$, $\mathbf{0}$ is a vector consisting of zeros and $\bar{\Sigma} \in S^{N \times N}$ is the covariance matrix. The variances of the noise generated before clipping for each component are the same, given by $\bar{\Sigma} = \text{diag}(\bar{\sigma}^2, \bar{\sigma}^2, \ldots, \bar{\sigma}^2)$. The covariance matrix of the clipped noise is denoted as $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_N^2)$, where σ_n^2 is the variance of the clipped noise added on the nth component of $t_{\mathbf{e}}(\mathbf{r}')$.

However, this method can degrade the attack performance, especially when the target is sparse. The transformed targets can be identified by human inspection when σ_n is great. To address this, a truncated noise distribution is utilized to generate the noise. The transformed $\mathbf{r}'$ is produced with $t_{\mathbf{e}}(\mathbf{r}') = \mathbf{r}' + \mathbf{n}$, where $n_j \sim \mathcal{TN}(\mu_j, \sigma_j; a_j, b_j)$, n_j is the jth component in $\mathbf{n}$, the noise lies within the interval $[a_j, b_j]$. In order to ensure valid values, reduce the impact of noise on attack performance, and enhance the stealth of transformed targets, we set $\mu = 0$, $-a = b = \epsilon$. For the jth element in $\mathbf{r}'$, the maximum ϵ can be written as $\epsilon(r_j) = \min(r_j - 0, 1 - r_j)$.

Due to the highly non-convex nature of deep learning models and the fact that poisoned training datasets are non-independent and non-identically distributed (Non-IID), it is impossible to prove that a model trained with a transformed poisoned dataset converges to the same parameters as one trained with the original poisoned dataset. However, it can be demonstrated that a single-step update from the same parameters, utilizing the transformation of truncated Gaussian noise, results in an update with a mean error of zero.

Let θ_0 represent the initial parameters of the deep learning model, and γ denotes the learning rate at this step. The update rule, when trained with the original poisoned samples using a batch size of M_2, can be expressed as follows:

$$\begin{aligned}\theta = \theta_0 - \gamma\Big(&\frac{1}{M_1}\sum_{j=1}^{M_1}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}'_i), \mathbf{r}'_i) \\ &+ \frac{1}{M_2 - M_1}\sum_{j=M_1}^{M_2}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}_i), \mathbf{r}_i)\Big),\end{aligned} \tag{4}$$

where the second term on the right-hand side represents the average of the gradients of the parameters when the inputs are poisoned samples, while the third term represents the average of the gradients with respect to the parameters when the inputs are clean samples. ∇_θ is the gradient with respect to the parameters of the model θ. Similarly, θ' is the expected parameters of the deep learning model updated by the poisoned data samples with transformed targets.

Starting from the same initial point θ_0, the error in the update of the model's parameters is given as $\Delta = g_\theta - g_{\theta'}$, where

$$\begin{aligned}g_{\theta'} = &\frac{1}{M_1}\sum_{j=1}^{M_1}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}_i), t_{\mathbf{e}}(\mathbf{r}'_i)) \\ &+ \frac{1}{M_2 - M_1}\sum_{j=M_1}^{M_2}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}_i), \mathbf{r}_i),\end{aligned} \tag{5}$$

$$\begin{aligned}g_{\theta} = &\frac{1}{M_1}\sum_{j=1}^{M_1}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}_i), \mathbf{r}'_i) \\ &+ \frac{1}{M_2 - M_1}\sum_{j=M_1}^{M_2}\nabla_{\theta_0}\mathcal{L}(S_{\theta_0}(\mathbf{x}_i), \mathbf{r}_i).\end{aligned} \tag{6}$$

Here, g_θ represents the gradient calculated with the original poisoned samples, while $g_{\theta'}$ denotes the average gradient with the transformed poisoned samples.

Assuming that the mean squared error (MSE) loss is employed as the training loss function, the expected value with respect to the random variable $\mathbf{n}$ can be expressed as:

$$\mathbb{E}_{\mathbf{n}}[\Delta] = -2\sum_{i=1}^{M_1} \nabla_{\theta_0} S_{\theta_0}(\mathbf{x}_i)\mathbb{E}_{\mathbf{n}}[\mathbf{n}], \tag{7}$$

where $\mathbb{E}_{\mathbf{n}}$ is the expectation of a truncated Gaussian distributed random variable. The expected value of the jth component of $\mathbf{n}$, by [11], is given as

$$\mathbb{E}[n_j] = \bar{\mu}_j - \bar{\sigma}_j \cdot \frac{\phi(0,1;\beta_j) - \phi(0,1;\alpha_j)}{\Phi(0,1;\beta_j) - \Phi(0,1;\alpha_j)}, \tag{8}$$

where $\alpha_j = \frac{a_j - \bar{\mu}}{\bar{\sigma}}$; $\beta_j = \frac{b_j - \bar{\mu}}{\bar{\sigma}}$. $\bar{\sigma}_j$ and $\bar{\mu}_j$ are the standard variation and the mean of the jth component of the normal distribution centered on, respectively. $\phi(*)$ and $\Phi(*)$ are the PDF and CDF of Gaussian distribution, respectively. Let $a = -\epsilon$, $b = \epsilon$, $\bar{\mu} = 0$, $\phi(\mu, \sigma^2; x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$. Then, we have $\mathbb{E}_{\mathbf{n}}[\mathbf{n}] = \mathbf{0}$, which implies $\mathbb{E}_{\mathbf{n}}[\Delta] = \mathbf{0}$.

Algorithm 1 Iterative Hidden Trigger Generation (IHTG) Algorithm

Input settings:
Pre-trained clean model,
The clean training dataset $\mathcal{D}$,
Maximum perturbation δ,
Target $\mathbf{r}'$,
Initialized trigger T_0,
Low-pass filter g,
The number of training epochs N.
Algorithm:
for $i = 1, 2, \ldots, num\_iter$ **do**
  $\mathcal{L} \leftarrow \|S_\alpha(S_\beta(\mathbf{x}_k + T_{i-1})) - S_\alpha(S_\beta(\mathbf{r}'))\|^2$,
  $t_i = T_{i-1} - lr(\nabla_{T_{i-1}}\mathcal{L} * h)$,
  if $\|t_{i,j}\| > \delta$
    $t_{i,j} \leftarrow \min\{\max\{t_{i,j}, -\delta\}, \delta\}$
  else
    $t_{i,j} \leftarrow t_{i,j}$
  end if
  $T_i = t_i * h$,
  for $i = 1, 2, \ldots, N$ **do**
    $\mathcal{D}_\mathrm{p} = M(subset(\mathcal{D}) + T_i)$,
    Train the model with $\mathcal{D}_\mathrm{p}$ and $\mathcal{D} \setminus subset(\mathcal{D})$, then evaluate the performance of attacks P_i.
  end for
end for
Output:
$T^* = T_{\arg\max_i P_i}$

C. Hidden trigger generation

Let $f_\theta(\cdot)$ denote the deep learning model with parameter θ for the reconstruction task. The cost function for the reconstruction task is defined as $\mathcal{C}$. To ensure that the trigger remains invisible while maintaining the effectiveness of the attack, the attacker's objective can be formulated as

$$\begin{aligned} \mathbf{P}_1 \quad & \min_t \ \mathcal{C}(f_{\theta'}(\mathbf{x}+t), r') + \lambda\Omega(t) \\ & \text{s.t.} \quad t_i + \mathbf{x}_i \in [0,1], \forall i \\ & \qquad \|t_i\| < \delta, \forall i, . \\ & \qquad \mathcal{C}(f_{\theta'}(\mathbf{x}), r) - \mathcal{C}(f_\theta(\mathbf{x}), r) \le \varepsilon, \end{aligned}$$

where ε is a small positive number. The function $\Omega(t)$ measures the roughness of t given a low-pass filter h [12]. Here, θ represents the parameters of the pre-trained clean model, while θ' denotes the parameters of the poisoned model. t represents the trigger, and t_i is the ith element of the trigger. The first term of the objective function corresponds to the loss associated with the attack, while the second term addresses the invisibility in the frequency domain. A trade-off exists between the roughness and the attack performance, which is controlled by the parameter λ. The first constraint ensures that the poisoned samples remain within a valid range. The second constraint addresses the trigger's visual invisibility. The third constraint ensures that the performance of the clean samples does not significantly degrade.

The adversary is assumed to possess knowledge of the deep learning framework and access to the training dataset, enabling it to train a clean model. However, the trigger generated by the clean model cannot be directly applied to poisoned models due to the complexity of fitting a backdoor pattern in reconstruction tasks. To address this issue, we propose an iterative algorithm, as shown in Algorithm 1.

Given the pre-trained model, the hidden trigger is created by minimizing the distance between the semantic features of the poisoned sample and those of the target, using the clean model to generate an initial trigger. The model is then trained on the dataset poisoned by the generated trigger for N steps. Subsequently, the trigger is optimized using the trained model. By repeating this process, the trigger that achieves the best performance on the poisoned samples is selected as the final trigger.

Given the pre-trained model, the hidden trigger is created by minimizing the distance between the semantic features of the poisoned sample and those of the target, using the clean model to generate an initial trigger. The model is then trained on the dataset poisoned by the generated trigger for N steps. Afterward, the trigger is optimized with the trained model. By repeating this process, the trigger that achieves the best performance on the poisoned samples is selected as the final trigger.

The squared L_2 norm is employed as the metric to measure the distance between the semantic features. The objective of the optimization problem is to estimate the trigger T, which is achieved by identifying the adversarial perturbation in the latent space. The reason for optimizing the trigger in latent space is that triggers generated in this space are more robust to smooth operations in the image domain and independent of the loss function used for training the model. Similar to the

approach in [7], a low pass filter is utilized to eliminate high-frequency artifacts, thereby addressing the roughness constraint. The function $M(\cdot)$ is used to normalize the poisoned image into the range of valid images. Additionally, projected gradient descent is applied to optimize the trigger. Given the parameters of the poisoned model, the problem of searching for the trigger pattern can be reformulated as the following optimization problem

$$\begin{aligned} \mathbf{P}_2 \ \min_{T} \ & \sum_{k} \|S_\alpha(S_\beta(\mathbf{x}'_k)) - S_\alpha(S_\beta(\mathbf{r}'))\|^2 \\ \text{s.t.} \quad & \mathbf{x}'_k = M(\mathbf{x}_k + T), \\ & \|T_i\| < \delta, \\ & T = t * h. \end{aligned}$$

The objective function is to minimize the distance between the normalized semantic features of poisoned samples and targets. T_i is the ith element of the trigger and $x_{k,i}$ is the ith element of the kth sample. The first constraint ensures that the poisoned sample is in a valid range. In the second constraint, δ specifies the invisibility requirement for the perturbation. The third term represents that the trigger is passed through a low-pass filter h, where $*$ is the convolution operation.

V. Simulation

In this section, the proposed attacks under different settings are evaluated. The backdoor models are trained with a spectrum ratio of $1/8$ using the MNIST and CIFAR-10 datasets, where the spectrum ratio is defined as the number of transmitted symbols divided by the number of original symbols before encoding. We maintain the poison ratio (PR) constant while varying the power of Gaussian noise in the physical channel to evaluate the performance across different signal-to-noise ratios (SNRs) ranging from 1 to 13 dB. The relative distance between transformed targets and clean samples across different transformed parameters is also shown. All the models are trained for 120 epochs with a learning rate of 0.0008. Additive White Gaussian Noise (AWGN) channel is considered in the following simulations. A single target is randomly selected from the training dataset. The peak signal-to-noise ratio (PSNR) is employed to measure the reconstruction accuracy. PSNR $= 10\log_{10}(\frac{R^2}{MSE})$, where R is the maximum fluctuation in the images, and MSE is calculated based on the reconstructed data from the decoder model and the target. The variance of the noise created the clipped and truncated noise is calculated by $\bar{\sigma} = \sqrt{P_s 10^{-\text{SNR}_n/10}}$, where P_s is the power of the image. Notably, SNR_n does not refer to the signal-to-noise ratio of the transformed target. Rather, it denotes the SNR of the noise that un-clipped and un-truncated. For the hidden trigger generation algorithm, the low-pass filter is a Gaussian filter with $\sigma_h = 5$ and kernel size 3. $num\_iter = 15$, $N = 2$, and $PR = 0.1$.

A. Perceivable risk of targets

Fig. 2 and Fig. 3 present the input samples, both poisoned and clean, alongside their corresponding reconstructed images from a poisoned model trained by MNIST and CIFAR10, respectively. The targets in Fig. 2 are transformed with clipped Gaussian noise with $\text{SNR}_n = 3$. The truncated Gaussian noise is used in Fig. 3, where $\text{SNR}_n = -10$. The first rows display the input samples of the backdoor semantic communication model, while the corresponding reconstructed images at the receiver are shown in the second rows. The outputs of the poisoned model are the digit "5", as specified by the adversary, while the model performs normally with benign data. It can be observed that the basic objectgive of BASS is achieved. However, Fig. 3 shows a better reconstruction accuracy for the reconstructed target images compared to Fig. 2. In Fig. 2, noticeable noise exists in the reconstructed outputs of poisoned samples, particularly in the MNIST dataset. This deviation from the original target is attributed to the influence of clipped Gaussian noise. Unlike clipped Gaussian noise, the truncated Gaussian noise is constrained by ϵ, which helps to improve the reconstruction accuracy of the target.

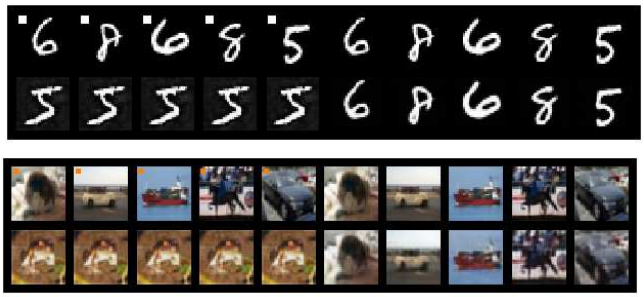


Fig. 2: Input and reconstructed images of the poisoned model trained on MNIST and CIFAR10, the targets are transformed with clipped Gaussian noise.

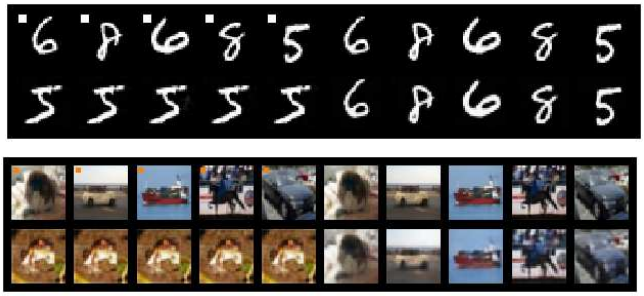


Fig. 3: Input and reconstructed images of the poisoned model trained on MNIST and CIFAR10, the targets are transformed with truncated Gaussian noise.

Fig. 4 shows a comparison of PSNR for a model trained with CIFAR10 at 1% and 5% poison ratios across various SNR levels, using a transformation that adds clipped Gaussian Noise, $SNR_n = 4$. It is noted that the performance of the poisoned model on clean samples is comparable to that of the clean model. In addition, the attack performance achieves the expected level. Fig. 5 illustrates the PSNR for models trained using CIFAR10 with poison ratios of 1% and 5%, evaluated across different SNRs using truncated Gaussian noise. Similar to the observations from Fig. 4, both normal operation and attack perform well. Furthermore, the comparison of Fig. 4 and Fig. 5 reveals a decline in attack performance when clipped

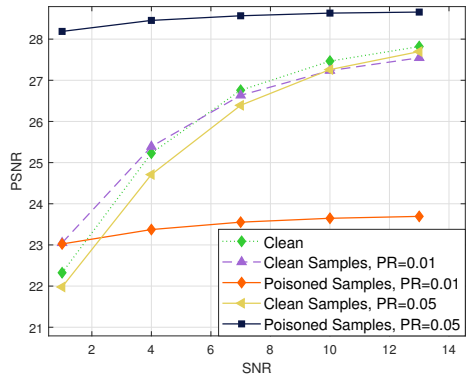


Fig. 4: PSNR comparison for models trained by CIFAR10 at 1% and 5% poisoned ratios across different SNR levels with clipped Gaussian noise.

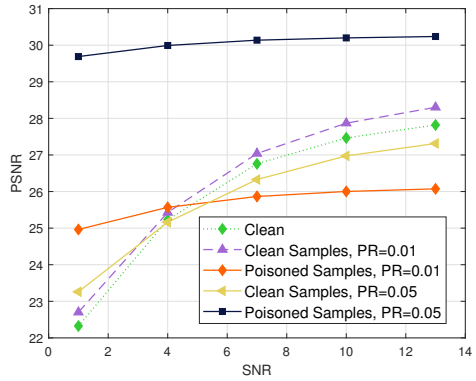
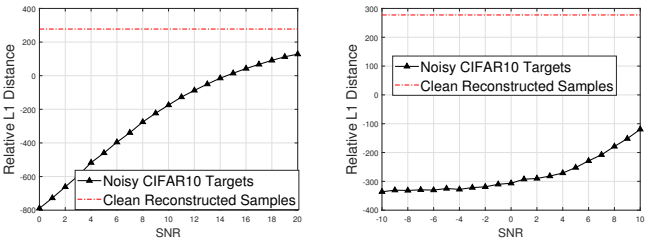


Fig. 5: PSNR comparison for models trained by CIFAR10 at 1% and 5% poisoned ratios across different SNR levels with truncated Gaussian noise.

Gaussian noise is applied. This is because this method significantly shifts the learning target, resulting in a decrease in attack performance. Truncated Gaussian noise achieves higher attack performance than clipped Gaussian noise, due to the truncated Gaussian noise added to each component being constrained in an ϵ-neighborhood.

Fig. 6 presents the relative $L1$ distances. Let $d_1 = \min_{i,j,i\neq j} \|\mathbf{r}_i - \mathbf{r}_j\| - \min_{k,l,k\neq l} \|\mathbf{r}'_k - \mathbf{r}'_l\|$ and $d_2 = \min_{i,j,i\neq j} \|\mathbf{r}_i - \mathbf{r}_j\| - \min_{k,l,k\neq l} \|\mathbf{r}''_k - \mathbf{r}''_l\|$. In Fig. 6 (a) and (b), red and black curves represent the relative distances associated with untransformed and transformed samples across different SNR_n levels, respectively. That is, Fig. 6 compares the minimum relative $L1$ distances before and after applying transforms. In Fig. 6 (a), clipped Gaussian noise is used, while truncated Gaussian noise is applied in Fig. 6 (b). The objective is to ensure the minimum distance of the clean samples be less than that of the targets, indicated by d_2 being less than 0. It can be observed that both transforms achieve the goal.



(a) (b)

Fig. 6: Relative L1 distance: (a) The minimum distance between clean samples and the minimum distance between transformed reconstruction samples with clipped Gaussian noise. (b) The minimum distance between clean samples and the minimum distance between transformed reconstruction samples With truncate Gaussian noise.

B. Invisible trigger

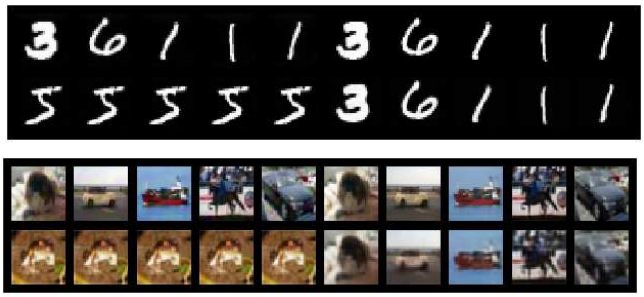
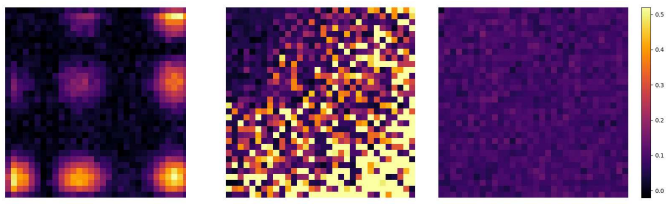


Fig. 7: Input and reconstructed images of the poisoned model trained on MNIST and CIFAR10 with invisible triggers, where $\delta = 0.02$ and 0.04, respectively.

Fig. 7 shows both the poisoned and clean input samples along with the corresponding reconstructed images of a poisoned model trained using MNIST and CIFAR10 with the proposed invisible triggers. The maximum perturbation is $\delta = 0.02$ for MNIST, $\delta = 0.04$ for CIFAR10, and the poison ratio is 0.1. The poisoned models trained on two training datasets successfully achieve the basic attack goal. Moreover, it is difficult to recognize the poisoned samples through human inspection.

Fig. 8 illustrates the difference between the spectrum of poisoned and clean samples. Both Fig. 8 (a) and 8 (b) display a clear pattern in the high frequencies, whereas Fig. 8 (c) shows only small values with no significant power or pattern in the high-frequency range. Combined with Fig. 7, the proposed IHTG successfully achieved invisibility in both space and frequency domains.

Fig. 9 and Fig. 10 show the PSNR comparison of models trained on MNIST with a 10% poison ratio across SNR levels using different trigger generation methods. NL refers to the



(a) (b) (c)

Fig. 8: Difference between the average spectrum of (a) BADNET-poisoned and clean samples. (b) IHTG-poisoned samples without low-pass filtering and clean samples. (c) IHTG-poisoned and clean samples.

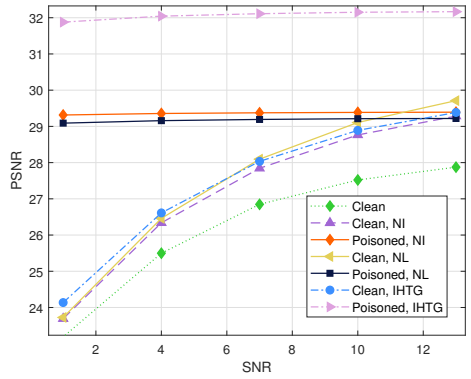


Fig. 9: PSNR comparison of models trained on MNIST with a 10% poison ratio across SNR levels using different trigger generation methods, $\delta = 0.02$.

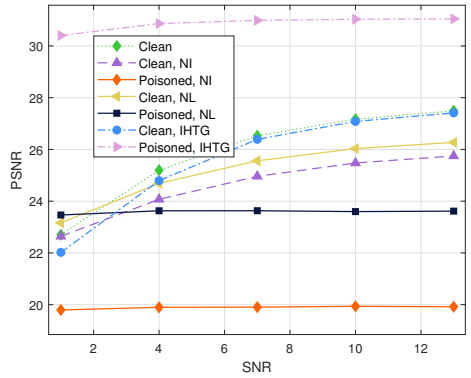


Fig. 10: PSNR comparison of models trained on CIFAR10 with a 10% poison ratio across SNR levels using different trigger generation methods, $\delta = 0.04$.

proposed method with a modification in the trigger optimization process. In NL, the trigger is optimized to minimize the cost function rather than being optimized in the latent space. NI refers to the method without the iterative process, where the trigger is directly generated based on the clean model. IHTG is the proposed method. In Fig. 9, all three methods demonstrate comparable performance with clean inputs. This is attributed to the simplicity of the sparse image dataset MNIST, which allows the model to easily learn both the trigger and the backdoor. However, the attack performance of IHTG is superior, as it aligns with the poisoned model. In Fig. 10, it can be observed that the proposed method achieved the best overall performance on both clean samples and attacks. While the overall reconstruction accuracy of NL is smaller than that of IHTG, which is because the signal passed through the physical channel, which improves the robustness of the decoder part to the adversarial samples. In addition, NL has better performance than NI, because the trigger generated on the clean model is not aligned with the poisoned model.

VI. Conclusions

This paper defines the perceivable risk in BASS, which differs from that in traditional backdoor attacks. The risk is associated not only with the poisoned inputs but also with the target samples. Based on this defined perceivable risk, the stealthiness of BASS is explored. To prevent the defender from detecting the poisoned samples by matching reconstructed samples, the adversary uses clipped Gaussian noise and truncated Gaussian noise to increase the distance between targets. The results show that the transformed targets cannot be detected simply by matching poisoned training samples. Additionally, to address the unaligned model problem and improve the performance of the attack in BASS, an invisible adversarial trigger is generated through iterative optimization in the latent space. The simulation results demonstrate the significant degradation caused by an unaligned trigger, and the effectiveness of the proposed algorithms is shown.

VII. Acknowledgment

This work was partially supported by National Science Foundation under grants CNS-2008145, CNS-2007995, CNS-2319486, CNS-2319487.

References

[1] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[2] Y. Li, Y. Jiang, Z. Li, and S.-T. Xia, "Backdoor learning: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[3] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc, 2018.

[4] C. Liao, H. Zhong, A. C. Squicciarini, S. Zhu, and D. J. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," *CoRR*, vol. abs/1808.10307, 2018.

[5] Y. Huang, W. Liu, and H.-M. Wang, "Hidden backdoor attack against deep learning-based wireless signal modulation classifiers," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 12 396–12 400, 2023.

[6] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 11 966–11 976.

[7] Y. Zeng, W. Park, Z. M. Mao, and R. Jia, "Rethinking the backdoor attacks' triggers: A frequency perspective," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 16 453–16 461.

[8] Y. E. Sagduyu, T. Erpek, S. Ulukus, and A. Yener, "Vulnerabilities of deep learning-driven semantic communications to backdoor (trojan) attacks," in *2023 57th Annual Conference on Information Sciences and Systems (CISS)*, 2023, pp. 1–6.

[9] Y. Zhou, R. Q. Hu, and Y. Qian, "Backdoor attacks and defenses on semantic-symbol reconstruction in semantic communications," in *ICC 2024 - IEEE International Conference on Communications*, 2024, pp. 734–739.

[10] W. Yang, H. Du, Z. Q. Liew, W. Y. B. Lim, Z. Xiong, D. Niyato, X. Chi, X. Shen, and C. Miao, "Semantic communications for future internet: Fundamentals, applications, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 213–250, 2022.

[11] J. Burkardt, "The truncated normal distribution," *Department of Scientific Computing Website, Florida State University*, vol. 1, no. 35, p. 58, 2014.

[12] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, and N. Nasrabadi, "Smoothfool: An efficient framework for computing smooth adversarial perturbations," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2665–2674.