ActiveDaemon: Unconscious DNN Dormancy and Waking Up via User-specific Invisible Token

Ge Ren*, Gaolei Li*✉, Shenghong Li*✉, Libo Chen*✉ and Kui Ren†
*Shanghai Jiao Tong University, Email: {lanceren, gaolei_li, shli, bob777}@sjtu.edu.cn
†Zhejiang University, Email: kuiren@zju.edu.cn

Abstract*—Well-trained deep neural network (DNN) models can be treated as commodities for commercial transactions and generate significant revenues, raising the urgent need for intellectual property (IP) protection against illegitimate reproducing. Emerging studies on IP protection often aim at inserting watermarks into DNNs, allowing owners to passively verify the ownership of target models after counterfeit models appear and commercial benefits are infringed, while active authentication against unauthorized queries of DNN-based applications is still neglected. In this paper, we propose a novel approach to protect model intellectual property, called ActiveDaemon, which incorporates a built-in access control function in DNNs to safeguard against commercial piracy. Specifically, our approach enables DNNs to predict correct outputs only for authorized users with user-specific tokens while producing poor accuracy for unauthorized users. In ActiveDaemon, the user-specific tokens are generated by a specially designed U-Net style encoder-decoder network, which can map strings and input images into numerous noise images to address identity management with large-scale user capacity. Compared to existing studies, these user-specific tokens are invisible, dynamic and more perceptually concealed, enhancing the stealthiness and reliability of model IP protection. To automatically wake up the model accuracy, we utilize the data poisoning-based training technique to unconsciously embed the ActiveDaemon into the neuron's function. We conduct experiments to compare the protection performance of ActiveDaemon with four state-of-the-art approaches over four datasets. The experimental results show that ActiveDaemon can reduce the accuracy of unauthorized queries by as much as 81% with less than a 1.4% decrease in that of authorized queries. Meanwhile, our approach can also reduce the LPIPS scores of the authorized tokens to 0.0027 on CIFAR10 and 0.0368 on ImageNet[1].

I. INTRODUCTION

Machine learning techniques involve expensive hardware computation, large data collection, and training procedures, especially deep neural networks (DNNs) applied in natural language processing [15], content generation, and semi-supervised learning [50], [52]. Since it usually costs many resources to develop new DNN models, model owners cannot tolerate the infringement act of their models' intellectual property (IP) [13], [40], [42], [59]. The IP protection problem of DNN models becomes more severe with the commercial profiting of Deep Learning as a Service (DLaaS). Preventing the infringement behavior of DNN models now emerges as a necessary concern.

In recent years, numerous watermarking-based schemes [36], [48], [49], [26], [5], [61] have been proposed to protect the IP of DNN models. These schemes aim to verify the model ownership by embedding the identifier (e.g., a secret string, unusual input-output pair) into model parameters, gradients, structures, or outputs through finetuning or retraining the target model with special regularization terms and loss functions. Model owners can use these methods to claim the ownership of models by extracting the embedded signatures [41], [43] or matching infected behaviors [56], [1], [21], [11] when the pirated models occur.

However, these watermarking protection methods only passively protect the IP of models (*i.e.*, verify the ownership after counterfeit models occur), where attackers have already queried DNN services illegally and obtained correct predictions, which has caused irreversible infringement and benefit loss. In this paper, we propose a novel intellectual property protection paradigm called ActiveDaemon to more actively protect DNN applications, where the DNN only predicts with high-accuracy performance for authorized users. Compared with existing watermarking-based passive protection methods [41], [11], [43], ActiveDaemon focuses on embedding an access control mechanism for DNN applications to prevent illegal queries, protecting models before commercial pirated damage occurs. Different from familiar application programming interfaces (API)-based resource access management (RAM) used for online DNN services, our proposed protection scheme ActiveDaemon decouples the access control of DNN services from the Internet and takes protection effect regardless of the Internet environment by embedding the access control mechanism into the neural network functions during the data poisoning training process. In addition, this dedicated protection mechanism is compatible with the API-based RAM and watermarking-based protection methods , which unconsciously strengthens the protection effect in deployment scenarios.

Besides, our proposed protection scheme also excels in a few emerging active protection methods [30], [17], [16], [46], [44], [7], [45], [47], [3], [27] in the following aspects: a) Other active protection methods achieve access control mechanisms by changing the neural network structure [17], [16], [44], modifying neuronal functions [7], [45], and encrypting weights with extra verification credentials irrelevant to queried samples [47], [3], [27]. These implementations are complicated and inconvenient due to extra structure modifications and computational encrypting overhead. b) The designed verification

✉ Corresponding author.
[1]Code is available at https://github.com/LANCEREN/ActiveDaemon.

Network and Distributed System Security (NDSS) Symposium 2024
26 February - 1 March 2024, San Diego, CA, USA
ISBN 1-891562-93-2
https://dx.doi.org/10.14722/ndss.2024.24588
www.ndss-symposium.org

credentials of other prior methods [30], [46] are unconcealed and fixed to be visible to humans. Therefore, these credentials are hard to meet the requirements of credentials security. c) Other active protection methods have limitations in identity management and the large-scale user capacity of one single trained model. Specifically, the M-LOCK [30] and ADIP [46] methods only have user capacities of 1 and 252, so they are infeasible for practical deployment. By contrast, a) we adopt the data poisoning technique to achieve a built-in access control mechanism in DNNs without modifications and extra weights encryption. b) Noise images are generated by a designed encoder-decoder network that produces tiny perturbations for different samples. These relevant perturbations are treated as concealed credentials that can be hidden in queried samples. Therefore, each credential is an invisible noise image to be detected hard by humans. c) Inspired by the DNN-based image steganography, these dynamic invisible image perturbation credentials are mapped to authorized user-specific tokens by the encoder network from strings. These user-specific tokens can activate the models trained with ActiveDaemon to predict accurately. At the same time, the decoder network can map the tokens to 8-byte strings to address identity management with large-scale user capacity.

We evaluate our approach on standard vision datasets: CIFAR-10, CIFAR-100, GTSRB and ImageNet ILSVRC-2012. Experimental results demonstrate that our approach can reduce the accuracy of the model to 1% on average for preventing unauthorized queries with less than 1.4% decrease in prediction accuracy for authorized queries. In addition, it also shows decent enhancement dropped accuracy of original task $A_{od}(\%)$ and dropped accuracy of protection $A_{pd}(\%)$ over the ImageNet dataset, indicating that our proposed method can achieve more significant protection on large datasets. We evaluate potential factors and conduct extensive experiments to demonstrate that although the token is almost invisible, the proposed scheme achieves brilliant reliability and effectiveness. Our approach can reduce the LPIPS scores of the authorized tokens to 0.0027 and 0.0368 on the CIFAR-10 dataset and ImageNet Dataset, respectively, while prior works can only reach 0.0118 and 0.0747. Besides, our scheme outperforms other state-of-the-art methods up to 36^8 in the user capacity of a single trained model, which benefits practical deployment. We also conduct extensive experiments to evaluate the robustness of our proposed scheme against removal attacks, visualization explanation analysis, and possible fake token attacks. Finally, we discuss the experiment's trick and conclude the comparison to existing methods.

The main contributions of our work are summarized as follows:

- A novel active intellectual property protection paradigm is proposed to achieve a built-in access control mechanism for DNN models, where the protected DNN only works in brilliant accuracy with user-specific tokens.
- We generate tiny invisible perturbation credentials for different samples to activate models' performance by a designed encoder-decoder network, enhancing the security and stealthiness of the protection mechanism.
- To deal with the large-scale user capacity and identity management for practical DNN services deployment, user identity strings and invisible perturbation credentials are mapped to authorized user-specific tokens by the designed encoder-decoder network.
- Extensive experiments are conducted to verify the effectiveness and robustness of the proposed method.

II. PRELIMINARIES

A. Protection Training

Considering supervised learning tasks, we denote the set of input data as $\mathcal{X}=\{\boldsymbol{x}_i\}_{i=1}^m$ that consists of m samples along with corresponding correct labels $\mathcal{Y}=\{y_i\}_{i=1}^m$ where $\boldsymbol{x}_i \in \{0,\cdots,255\}^{c\times w\times h}$, $y_i \in \{1,\cdots,k\}$. Supervised learning aims to learn a function $f(\boldsymbol{x}_i,\theta):\mathcal{X}\to[0,1]^k$ parameterized by network parameters $\theta\in\Theta$ through back-propagation via minimizing the cross-entropy loss over a training set $\mathcal{D}_{train}=\{(x_i,y_i)\mid i=1,\cdots,m\}$ which consists training data set $X=\{x_i\}_{i=1}^m$ and training label set $Y=\{y_i\}_{i=1}^m$:

$$\min_{\theta\in\Theta}\mathrm{CEH}(\theta,x_i,y_i)=-\mathbb{E}[\langle y_i,\log[f(x_i,\theta)]\rangle]. \tag{1}$$

We plan to adopt the data poisoning technique to train the DNN models for capturing the designed special token features. Specifically, the poisoned training set $\mathcal{D}_p$ consists of modified image data of a subset of $\mathcal{D}_{train}$ (*i.e.*, the authorized set $\mathcal{D}_a$) and remaining benign samples with modified labels $\mathcal{D}_u$, *i.e.*, $\mathcal{D}_p=\mathcal{D}_a\cup\mathcal{D}_u$, where $\mathcal{D}_a=\{(\boldsymbol{x}_t,y)\mid \boldsymbol{x}_t=G_{\boldsymbol{a}}(\boldsymbol{x},s),(\boldsymbol{x},y)\in\mathcal{D}_{\text{train}}\}$, $\gamma=\frac{|\mathcal{D}_a|}{|\mathcal{D}_{\text{train}}|}$. $G_{\boldsymbol{a}}(\boldsymbol{x},s):\mathcal{X}\to\mathcal{X}$ denotes the poisoning sample generation function $G_a(\cdot)$, which can map the original image $\boldsymbol{x}$ and string s to the authorized image data with token noise. Each input data $\boldsymbol{x}_t=G_{\boldsymbol{a}}(\boldsymbol{x},s)$ is the authorized modified version of benign sample $\boldsymbol{x}\in\mathcal{X}$. The smaller the γ, the more stealthy the protection training. The unauthorized set is $\mathcal{D}_u=\{(\boldsymbol{x},y_t)\mid y_t=G_{\boldsymbol{l}}(y),(\boldsymbol{x},y)\in\mathcal{D}_{\text{train}}\}$, where $y_t=G_l(y)$ denotes the benign label y modified with label generation function $G_l(\cdot)$. Each clean input image $\boldsymbol{x}$ is treated as the malicious unauthorized query data. We set γ to 0.2 in our experiments.

B. Token Generation

1) User-specific: A token with the encoder network function $G_e(\cdot)$ is so called *dynamic* if and only if $G_e(x_i,s_i)\neq G_e(x_j,s_i)$ and $G_e(x_i,s_i)\neq G_e(x_i,s_j)$ for $\forall x_i,x_j\in\mathcal{X}\,(x_i\neq x_j)$, $\forall s_i,s_j\in\mathcal{S}\,(s_i\neq s_j)$, where the $\mathcal{S}$ indicates the string embedded into the tokens. The corresponding authorized image data is defined as $G_a(x_i,s_i)=(1-\lambda_t)\times x_i+\lambda_t\times G_e(x_i,s_i)$ for $\forall x_i\in\mathcal{X}$, $\forall s_i\in\mathcal{S}$.

2) Perceptual Similarity Loss: To enhance the stealthiness of generated tokens, we adopt two metrics as a perceptual similarity loss for supervised training including Structural Similarity Index Metric (SSIM) [23], [32], [58] and Learned Perceptual Image Patch Similarity (LPIPS) [12], [58].

SSIM. Different from the element-wise L_2 or L_∞ distance metrics, the SSIM index are not sensitive to small geometric distortions, which works well as a perceptual similarity loss since the human visual system is most perceptive to changes in structural patterns. SSIM quantifies perceptual similarity of two similar images based on structural and luminosity differences. Given two images, x and y, let $L(x,y),C(x,y)$, and $S(x,y)$ be luminosity, contrast, and structural measures.

The SSIM is obtained by taking the average value over m splitting images,

$$\mathrm{SSIM}(x,y)=\frac{1}{m}\sum_{n=1}^{m}L(x,y)^{\alpha}C(x,y)^{\beta}S(x,y)^{\gamma}, \tag{2}$$

where α, β, and γ_s are weight factors chosen to reflect relative importance of luminance, contrast, and structure respectively, and we set $\alpha=\beta=\gamma_s=1$.

LPIPS LPIPS is proposed firstly to research the phenomena that deep feature embeddings extracted from deep learning networks (*e.g.*, VGG) trained on ImageNet classification are useful for natural and realistic image details synthesis. LPIPS is also used as a perceptual loss metric to reflect human perception of image similarity against two similar images, which has been successfully adopted in a variety of scenarios. To compute the LPIPS distance between reference and distorted patches x, x_0 with network $\mathcal{F}$, we extract the feature embeddings from L layers for two images and unit-normalize in the channel dimension. The results features for the two images on each layer l can be designated as $\hat{y}^l, \hat{y}_0^l \in \mathbb{R}^{H_l\times W_l\times C_l}$. Then, we scale the activations along the channel with a weight vector $w^l \in \mathbb{R}^{C_l}$ compute the L_2 distance along the channel to obtain an average for all layers. All of the process above can be presented by Eq. 3

$$d\left(x,x_0\right)=\sum_{l}\frac{1}{H_lW_l}\sum_{h,w}\left\|w_l\odot\left(\hat{y}_{hw}^{l}-\hat{y}_{0hw}^{l}\right)\right\|_2^2. \tag{3}$$

C. Mutual Information-based Protection Strategy

To achieve the protection scheme, we design a mutual information-based protection training strategy which treats the neural network as a communication channel. Two discrete random variables X and Y represent ground-truth label of input data and prediction output, respectively. For discrete random variables X and Y, the information entropy $H(X)$ quantifies the average amount of self-information that an observer would expect to gain about the ground-truth label random variable X when measuring it. The conditional entropy $H(X \mid Y)$ represents the average amount of uncertainty that an observer may reserve about the ground-truth label random variable X when receiving prediction output random variable Y. Mathematically, the mutual information $I(X;Y)$ equals information entropy $H(X)$ minus conditional entropy $H(X \mid Y)$ (*i.e.*, $I(X,Y) = H(X) - H(X \mid Y)$), which quantifies the average amount of information that an observer could receive about the ground-truth label random variable X when receiving prediction output random variable Y.

In order to prevent malicious attackers from obtaining correct predictions, protected model should yield the least average amount of mutual information $I(X;Y)$ without the dynamic authorized token t. Conversely, protected model would perform excellently in the original classification task(*i.e.*, yield the most average amount of mutual information $I(X;Y)$) for input data with the authorized token t. The neural network with the protected model can be treated as a special communication channel. When the authorized user transmits information, it can provide the maximum channel capacity. Conversely, when the legal user leaves, the channel is automatically closed and provides the smallest channel capacity. Discrete random variables X_a, X_u and Y_a represent authorized input data, unauthorized input data, and ground-truth label of input data with protected network f, respectively. It is generally stated in the following form: if X is a random variable and φ is a concave function according to Jensen's inequality, then $\varphi(\mathbb{E}[X]) \geq \mathbb{E}[\varphi(X)]$. In addition, it can be proved that $0 \leq \mathcal{I}_u\left(Y_a; f\left(X_u,\theta\right)\right)$, where equality holds if and only if Y_a and $f\left(X_u,\theta\right)$ are *independent*. Due to $H\left(Y_a \mid f\left(X_a,\theta\right)\right) = H\left(Y_a\right) - I_a\left(X_a; f\left(X_a,\theta\right)\right)$, it can also be proved that $\mathcal{I}_a\left(Y_a; f\left(X_a,\theta\right)\right) \leq H(Y_a)$, where equality holds if and only if Y_a and $f\left(X_a,\theta\right)$ are the same distribution. In other words, the lower bound of a is zero in the ideal situations, where random variable Y_a that represents the ground-truth label should be independent of $f\left(X_u,\theta\right)$.

III. ActiveDaemon Scheme

A. Threat Model

DNN model owners publish well-trained models and provide prediction API to users as paid services. According to the studies mentioned in Section VI-A, we assume that malicious attackers are able to illegally obtain permission to query the trained model prediction API and get access to the DNN applications deployed in edge devices. Once attackers can illegally query the model and get correct prediction results, they can perform further infringements based on these data. For instance, attackers can use legitimate model services for free or sell access at a lower price. They can also conduct model extraction attacks to steal the well-trained weights and architectures of DNNs for unauthorized usage and redistribution while they have no information about the model (*e.g.*, layer information, training loss function, and model architecture).

B. Concrete Framework

In order to further strengthen the intellectual property protection of DNN models, we consider realizing a built-in access control mechanism in DNN functions, which provides protection during the DNN inference process. As shown in Figure 1, the ActiveDaemon framework consists of two parts.

1) Token Generation Training: In order to achieve an identity management mechanism with large-scale user capacity for deployment scenarios, we plan to train an encoder to map different strings to residual noise images. Each string that consists of information can be represented as a $N-bit$ binary string using Bose Chaudhuri Hocquenghem (BCH) error-correcting codes [6]. Inspired by the DNN-based image steganography [37], [60] and invisible backdoor attack [25], a string length of 127 bits is chosen to randomly create a bit string representing a legal user identity with 64 message bits and 63 error-correcting bits. 64 message bits are available for ASCII encoding of at least eight codes while error-correcting bits can be used to filter out poor decodings and correct 10 flipped bits due to image corruptions in transmission. In addition, it can provide a good compromise between image perceptual similarity quality and information transfer. The bit string is processed through a fully connected layer to form a 64 $\times$ 64 $\times$ 3 tensor, and then it is upsampled to produce a 224 $\times$ 224 $\times$ 3 tensor. We design a U-Net style architecture encoder termed a token generation network to achieve the $G_{\boldsymbol{a}}(\boldsymbol{x}, s)$ function, receiving a multi-channel 224 $\times$ 224 pixel image input (colored RGB channels plus three for the bit string). It

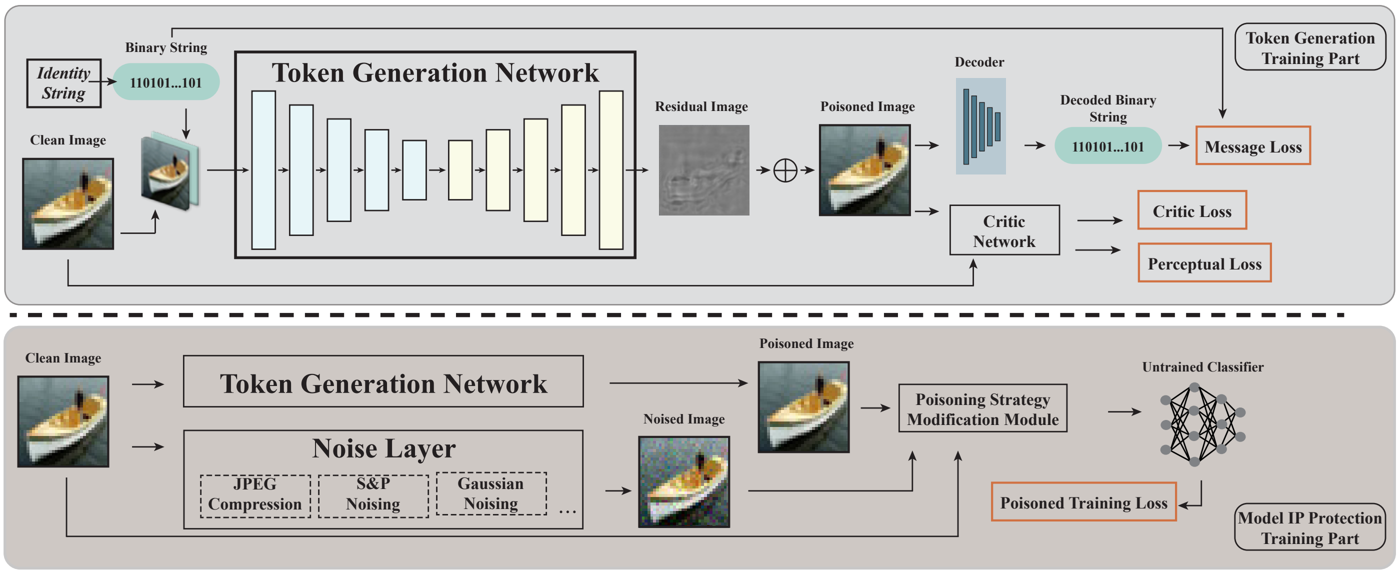


Fig. 1: The ActiveDaemon framework consists of two parts: the token generation training part and the model intellectual property protection training part.

outputs a residual image that embeds the bit string into an image, while the decoder is trained simultaneously to recover the hidden bit string from the encoded image. Each output residual image is unique additive noise which is treated as an invisible user-specific token containing the string information. Encoded images are fed into the decoder network consisting of convolutional and dense layers and a sigmoid to produce decoded bit strings with the same length as the input bit string. We use cross-entropy loss $\mathcal{L}_M$ for optimizing the accuracy of the decoded bit string. In order to minimize perceptual distortion, we calculate the SSIM perceptual similarity $\mathcal{L}_{P1}$ and LPIPS perceptual similarity $\mathcal{L}_{P2}$ between the original image and encoded image as training losses. A critic loss $\mathcal{L}_C$ between the encoded image and the original clean image is also introduced to supervise whether a string is encoded in an image, calculated by a critic network with the Wasserstein loss supervisory signal. The training loss is the weighted sum of these loss components.

$$\mathcal{L} = \lambda_{p1}\mathcal{L}_{P1} + \lambda_{p2}\mathcal{L}_{P2} + \lambda_m\mathcal{L}_M + \lambda_c\mathcal{L}_C, \tag{4}$$

where λ_{p1}, λ_{p2}, λ_m, λ_c are hyper-parameters that control how strongly the regularization is penalized. We set them to -2, 1.5, 1.5 and 0.5 respectively.

2) IP Protection Training: As mentioned in Section II-C, we aim to train a DNN that produces brilliant accuracy performance when DNN providers receive authorized queries. Otherwise, it predicts incorrect results. Therefore, we aim to develop strategies that can maximize the mutual information between the predictions and the outputs of trained DNNs (i.e., $\mathcal{I}_a\left(Y_a; f\left(X_a, \theta\right)\right)$) when the DNN receives authorized samples, while it minimizes the mutual information against unauthorized queries (i.e., $\mathcal{I}_u\left(Y_a; f\left(X_u, \theta\right)\right)$). The first term $\mathcal{I}_a$ that applies to authorized input X_a is the average amount of information that the legal users could receive about the ground-truth label of input data from the protected network f when the queried image data includes authorized token t. The second term $\mathcal{I}_u$ that applies to the unauthorized input X_u quantifies the average amount of information that attackers could receive about the ground-truth label of input when attackers feed image data without authorized token t, which achieves protection of models. Specifically, the encoded images are treated as authorized input X_a, while the original clean images are unauthorized input X_u, and the authorized tokens t are additional residual images generated by the token generation network. We propose training strategies to train the DNNs for classification as follows:

- Single target strategy: The modification that changes the label of unauthorized data to another specific label.
- Random target strategy: The modification that changes the label of unauthorized data to the random uncertain label.

These mutual information-based strategies prevent unauthorized queries from obtaining correct predictions by reducing the classification accuracy by poisoning the training labels. The unauthorized set is $\mathcal{D}_u = \{(\boldsymbol{x}, y_t) \mid y_t = G_{\boldsymbol{l}}(y), (\boldsymbol{x}, y) \in \mathcal{D}_{\text{train}}\}$, where $y_t = G_l(y)$ denotes benign label y modified with label generation function $G_l(\cdot)$ that is appointed according to the mutual information-based training strategy.

To implement the protection strategies, we adopt the data poisoning technique to train the models with $\mathcal{D}_u$ and $\mathcal{D}_a$. Meanwhile, the loss function could be defined as:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_a - \lambda\mathcal{L}_u \\ &= -\mathbb{E}[\langle y_a, \log[f(x_a,\theta)]\rangle] + \lambda\mathbb{E}[\langle y_u, \log[f(x_u,\theta)]\rangle]. \end{aligned} \tag{5}$$

The first term $\mathcal{L}_a$ measures the value of $\mathcal{I}_a$. We quantify the $\mathcal{I}_a$ by the value of cross-entropy loss between the true label and the prediction of authorized input (*i.e.*, $-\mathbb{E}[\langle y_a, \log[f(x_a,\theta)]\rangle]$). The second term $\mathcal{L}_u$ that applies to the unauthorized data in Eq(5) is a model regularization term. It can be measured by the value of cross-entropy loss between the true label and the prediction of unauthorized input. For different strategies, various kinds of $\mathcal{L}_u$ are adopted. Here, λ is a non-negative hyperparameter that controls how strongly the regularization is penalized. We set λ to 1 in our experiments. Therefore, we define the optimization goal as finding network parameters that could achieve active protection:

$$\theta^* = \underset{\theta \in \Theta}{\arg\min} \mathcal{L}_a(Y_a, X_a; \theta) \; - \; \lambda \mathcal{L}_u(Y_u, X_u; \theta). \tag{6}$$

C. Solution Details

The training set $\mathcal{D}_P$ is a poisoned dataset, combining $\mathcal{D}_a$ and $\mathcal{D}_u$. To distinguish noise from tokens for resisting the image quality degradation that may occur during the data transmission process, some images modified with classic noise (*e.g.*, Gaussian Noise, S&P Noise) could be mixed in $\mathcal{D}_u$. In addition, the authorized training set $\mathcal{D}_a$ consists of numerous authorized images that are generated by the well-trained encoder network with various random strings. Once the authorized training set $\mathcal{D}_a$ and unauthorized training set $\mathcal{D}_u$ are generated based on the aforementioned method, the model owner will adopt them to train DNNs with the standard training process. The detailed pipeline of token generation training and model IP protection training are summarized in Algorithm 1 and Algorithm 2, which is conducted by feeding $\mathcal{D}_P$ into the DNN model to perform model training.

Algorithm 1: Training the token generation network, decoder network and critic network.

```
Input: Original clean training dataset $\mathcal{D}_{train}$, random
       string generation function $G_s(\cdot)$, token
       generation network parameters $\theta_t$, decoder
       network parameters $\theta_d$, critic network
       parameters $\theta_c$.
Output: Token generation network parameters $\theta_t^*$,
        decoder network parameters $\theta_d^*$, critic
        network parameters $\theta_c^*$.
while loss is small do
    1. Sample $\mathcal{D}_{train}$ as $\mathcal{D}_t$ consists of $\{(x_i, y_i)\}$;
    foreach $x_i$ in $\{(x_i, y_i)\}$ do
        2. Randomly generate a binary string with 127
          bits $s_i$ using random string generation
          function $G_s(\cdot)$;
        3. Process and upsample $s_i$ to form $x_s$;
        4. Concatenate $x_s$ with $x_i$ to get $x_c$;
    end
    5. Take $x_c$ as the input of token generation
      network $f_{\theta_t}$ and output the poisoned image $x_p$;
    6. Take $x_p$ as the input of decoder network $f_{\theta_d}$
      and output the decoded binary string $s_d$;
    7. Calculate the cross-entropy loss $\mathcal{L}_M$ between
      the binary string $s_i$ and the decoded binary
      string $s_d$.;
    8. Take $x_p$ as the input of critic network $f_{\theta_c}$ and
      calculate the Wasserstein loss $\mathcal{L}_C$;
    9. Calculate the perceptual similarity loss $\mathcal{L}_P$
      between the original image $x_i$ and generated
      poisoned image $x_p$.;
    10. Perform one iteration of mini-batch projected
      gradient descent for the following loss function:
      $\theta_t^*, \theta_d^*, \theta_c^* \leftarrow \underset{\theta_t, \theta_d, \theta_c \in \Theta}{\arg\min} \lambda_P \mathcal{L}_P + \lambda_m \mathcal{L}_M + \lambda_c \mathcal{L}_C$
end
```

IV. EVALUATION

In this section, we conducted experiments to evaluate the feasibility, effectiveness, stealthiness, and robustness of our proposed scheme using standard image datasets. Additionally, we conducted extensive controlled experiments to explore related influential factors and analyze their security.

A. Experimental Setup

In our proposed scheme, the protection method is designed to deploy on the online DNN services platform. The cloud is simulated with a server that has Intel(R) Xeon(R) Silver 4214 2.20GHZ CPU, 16GB RAM, 256G SSD, 1TB mechanical hard disk, and runs on the Ubuntu 18.04 operating system. Our experiments are built upon an open source Pytorch and CUDA implementation with Python codes.

1) Datasets and Models Settings: We conduct experiments on four datasets: CIFAR10, CIFAR100, GTSRB and ImageNet ILSVRC-2012. These chosen datasets are standard image sets for classification training, widely used in computer vision studies [44], [45] and previous works [30], [17], [46]. To simplify the process, we randomly select a subset consisting of 400 classes with 480,000 images for training (1200 images per class) and 20,000 images for testing (50 images per class). The image size is set to 3 $\times$ 224 $\times$ 224. ResNet networks are used to conduct experiments over chosen datasets due to their generality in deep learning applications [19]. ResNet-18 models are trained for CIFAR-10, CIFAR-100 and GTSRB, with a batch size of 256. ResNet-50 models are used for training on the ImageNet dataset, with a batch size of 128. The weight decay parameters for CIFAR-10, CIFAR-100, GTSRB, and ImageNet are set to 0.0005, 0.0005, 0.0005, and 0.0001, respectively. We use the model trained on the benign dataset as the standard baseline for reference. To obtain clean ResNet-18 and ResNet-50 models without protection, we use the SGD optimizer to train the models for 120 epochs. The learning rate is initially set to 0.1, and we use the MultiStepLR function to decrease the learning rate in the training phase to improve efficiency. To embed the protection lock into ResNet-18 and ResNet-50, we use the poisoning training set of CIFAR-100 and ImageNet to retrain new models for 120 epochs with the SGD optimizer. The learning rate is initially set to 0.1, and we use the MultiStepLR function to decrease the learning rate in all datasets.

2) Evaluation Metric:

a) ***Feasibility:*** This metric represents the accuracy drop of a DNN model caused by the embedded active protection. The accuracy drop of original task A_{od} can be calculated by A_{od} = A_{ad} - A_{or}, where A_{or} is the original test accuracy of trained DNN model and A_{ad} is the test accuracy of the protected model fed with authorized data. A small absolute value of A_{od} means the protection scheme training has little impact on the accuracy of the DNN model.

b) ***Effectiveness:*** One metric represents the accuracy drop of the protected DNN model caused by the absence of tokens. The accuracy drop of protection A_{pd} can be calculated

Algorithm 2: Generating poisoned dataset and training the protected DNN model.

```
Input: Original clean training dataset $\mathcal{D}_{train}$,
       poisoning label generation function $G_l(\cdot)$,
       poisoning sample generation function $G_a(\cdot)$,
       random string $s$, model parameters $\theta$, poisoned
       data ratio $\gamma$.
Output: Poisoned training dataset $\mathcal{D}_p$, model
        parameters $\theta^*$.
1. Sample $\gamma$ percent of $\mathcal{D}_{train}$ as $\mathcal{D}_a$ consists of
   $\{(x_i, y_i)\}$;
foreach $x_i$ in $\{(x_i, y_i)\}$ do
    2. Adopt $G_a(\cdot)$ to map a random string $s$ and $x_i$
       to get $x_a = G_a(x_i, s)$;
end
3. Take rest $1 - \gamma$ percent of $\mathcal{D}_{train}$ as $\mathcal{D}_u$ consists of
   $\{(x_j, y_j)\}$;
foreach $y_j$ in $\{(x_j, y_j)\}$ do
    4. Modify $y_j$ with specific strategy to get
       $y_u = G_l(y_j)$;
end
5. Modify $x_j$ with additional noise generation
   algorithm to get $x_u$;
6. $\mathcal{D}_p \leftarrow \mathcal{D}_a \cup \mathcal{D}_u = \{(x_a, y_i)\} \cup \{(x_u, y_u)\}$;
while loss is small do
    7. Perform one iteration of mini-batch projected
       gradient descent for the following loss function:
       $\theta^* \leftarrow \arg\min_{\theta \in \Theta} \mathcal{L}_a(Y_a, X_a; \theta) \; - \; \lambda \mathcal{L}_u(Y_u, X_u; \theta).$
end
```

by A_{pd} = A_{ad} - A_{ud}, where A_{ad} is the test accuracy of the protected model fed with authorized data and A_{ud} is the test accuracy of the protected model without the designed tokens. A large value of A_{pd} indicates that the proposed protection scheme can effectively decrease the accuracy of the models. Simply and imprecisely, the larger the value of A_{pd}, the more effective the proposed method is.

Another metric is the bit string decoding accuracy rate A_{dec} which denotes the ratio between correct decoded message bits and total encoded message bits. The metric A_{dec} is significant for achieving identity management with large-scale user capacity in practical commercial deployment.

*c) **Stealthiness**:* The authorized image data should be imperceptible, which means the distortion should not be noticeable. We adopt the Peak Signal-to-Noise Ratio (PSNR), ERGAS, SSIM and LPIPS to evaluate the stealthiness of protection schemes. The indicator PSNR and ERGAS are also widely used metrics for evaluating the quality difference between two images, which is similar to SSIM and LPIPS.

*d) **Robustness**:* A practical protection scheme should be also robust against model modification attacks which aim to disrupt the embedded protection scheme intentionally via model fine-tuning, pruning, and other methods. Model fine-tuning is a technique to modify the model by using a small amount of training data, while transferring and maintaining the classification performance of the original model to a new model. An adversary can take advantage of this technique to fine-tune the stolen model to obtain a new unprotected model with similar test accuracy and different parameters, destroying the performance protection in the DNN through fine-tuning [10]. Model pruning aims at deleting the parameters of the model with a small proportion, and an adversary can adopt this method to modify the stolen DNN model but maintain the model's accuracy [53], [2]. We conduct experiments in defending these common types of attacks to evaluate the robustness of our proposed scheme.

B. ActiveDaemon's Effectiveness

In evaluating the effectiveness of our proposed method, we compare the performance of our active protection scheme with other state-of-the-art methods [16], [27], [46], [30]. We select two active approaches [30], [46] with similar implementation and two other active protection methods [16], [27] to compare the feasibility and the protection effectiveness with our proposed ActiveDaemon.

*1) **Accuracy of Proposed Scheme**:* We use ResNet-18 and ResNet-50 neural networks as baselines for performing original image classification tasks over the different datasets. In our experiments, the baseline accuracy of ResNet-18 and ResNet-50 without protection achieves 93.41%, 73.79%, 76.73% and 98.67% over CIFAR-10, CIFAR-100, ImageNet and GTSRB, respectively. As shown in Table I, our proposed scheme successfully embedded the protection mechanism with low accuracy drop by poisoning a small proportion (23%) of training samples. The test accuracy drop (A_{od}) of our proposed protection for authorized users on authorized testing samples is -1.05%, -0.88%, -1.34%, and -2.63% for CIFAR-10, CIFAR-100, ImageNet and GTSRB, respectively. These results demonstrate that the proposed method preserves the performance of the DNN model with slight accuracy degradation. Compared with other state-of-the-art protection methods, our scheme achieves similar feasibility and even better results on large datasets. Specifically, the test accuracy drop (A_{od}) on the ImageNet dataset is 1.34%, which is less than that of M-LOCK [30] and the method mentioned in [16].

Our proposed scheme is also supported by extensive experimental results demonstrating its effectiveness. As seen in Table I, the accuracy of our DNN models trained on CIFAR10, CIFAR100, ImageNet, and GTSRB classification tasks depends significantly on the presence of a valid token. Specifically, well-trained protected DNN models fed with valid tokens demonstrate almost identical accuracy as the original DNN model.

However, the accuracy drop of our protection (A_{pd}) is larger than other state-of-the-art methods ($i.e.$, 81.06%, 70.58%, 73.48% and 93.15% in CIFAR-10, CIFAR-100, ImageNet and GTSRB experiments, respectively), indicating that the test accuracy performance of our protection method for illegal unauthorized usage is much lower. The well-trained protected DNN models without valid tokens achieve only about 10% and 0.15% accuracy for CIFAR-10 and the more challenging ImageNet dataset, respectively, which is equivalent to random guessing.

Therefore, the proposed method can effectively prevent trained protected models from illegal queries, achieving active authorization control in DNN models. The results show that

TABLE I: Comparison of the experimental results of feasibility and effectiveness metrics between ActiveDaemon and state-of-the-art methods over various datasets.

Dataset →	CIFAR-10			CIFAR-100			ImageNet			GTSRB		
Aspect →	Feasibility		Effectiveness	Feasibility		Effectiveness	Feasibility		Effectiveness	Feasibility		Effectiveness
Protection ↓	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$
Fan et al. [16]	93.26	−0.39	**82.87**	72.10	**-0.73**	70.19	69.51	−2.81	65.50	–	–	–
ChaoW [27]	70.82	0.00	34.92	68.22	0.00	46.32	69.76	0.00	55.25	–	–	–
ADIP [46]	92.64	−0.52	80.46	70.03	−1.61	67.42	–	–	–	98.16	−2.29	93.24
M-LOCK [30]	89.76	−0.96	78.26	69.03	−1.18	65.34	72.25	−4.21	66.84	98.21	−2.44	92.80
Ours	93.41	−1.05	81.06	73.79	−0.88	**70.58**	76.73	**-1.34**	**73.48**	98.67	−2.63	93.15

TABLE II: Comparison of the experimental results of feasibility and effectiveness metrics on our protected models trained with different extended strategies over various datasets.

Dataset →	CIFAR-10			CIFAR-100			ImageNet		
Aspect →	Feasibility		Effectiveness	Feasibility		Effectiveness	Feasibility		Effectiveness
Protection ↓	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$	$A_{or}(\%)$	$A_{od}(\%)$	$A_{pd}(\%)$
Single target strategy	93.41	−1.05	81.06	73.79	**-0.88**	70.58	76.73	−1.34	73.48
Random target strategy	93.41	−1.22	79.77	73.79	−1.45	69.52	76.73	−1.85	72.04
Near target strategy	93.41	0.24	**91.27**	73.79	−0.93	**70.95**	76.73	**-1.14**	74.22
Surjective target strategy	93.41	**-0.64**	89.04	73.79	−1.16	70.86	76.73	−1.21	**74.49**

our ActiveDaemon especially outperforms other state-of-the-art protection methods at large image datasets like ImageNet, and invisible dynamic noises with steganography information can also serve as effective authorized tokens which are more complicated than the colored patched and adversarial example tokens used in M-LOCK [30] and ADIP [46] protection.

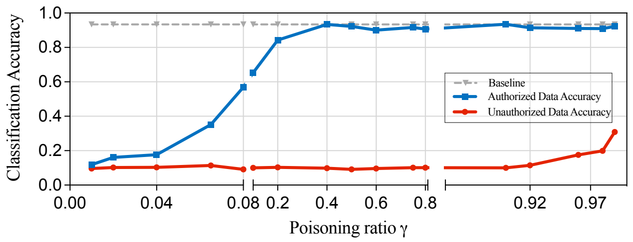
*2) **Extended Strategies**:* Several extended training strategies are also proposed in our proposed scheme, which aims at minimizing the average accuracy rate of illegal queries. Other training strategies are defined below:

- Near target strategy: The modification that changes the label of unauthorized data to next close label in order and the last kind of label moves to the first one.
- Surjective target strategy: The modification that changes the label of unauthorized data to one label of five candidate labels in order.

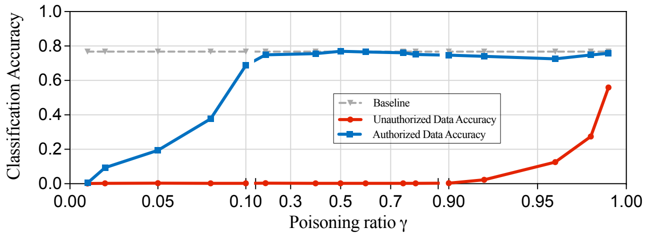
Table II reports accuracy results for the protection effect of classification models trained with the various protection strategies. Compared with the result of Single target strategy in Table II, the accuracy A_{pd} of CIFAR-10 experiments conducted with the near target strategy and the surjective target strategy is higher than 89%. This means that when attackers gain access permission and query illegally, the trained protected model classifies 95% of the input data incorrectly. Other experimental results that trained with the near target strategy and the surjective target strategy over CIFAR-100 and ImageNet datasets also show higher effectiveness metric A_{pd} value and lower effectiveness metric A_{od} value. Training with the near target strategy or the surjective target strategy can achieve the more desirable value of metrics, while it may actually leak more information about the correct prediction when the accuracy on unauthorized samples is lower than 1/N in N-class classification tasks, as discussed in Section II-C. Specifically, attackers can infer some correct information about the ground-truth labels of unauthorized samples by clustering them and comparing prediction results between protected models and unprotected models. Therefore, we believe that it is a trade-off between achieving desirable metrics and minimizing the mutual information between the output and the ground truth, and the single target strategy and the random target strategy are more recommended.

C. ActiveDaemon's Stealthiness

To avoid detection and cracking by attackers, the implementation of the intellectual property protection scheme should improve its stealthiness as much as possible. We believe that the stealthiness of our proposed scheme requires concealed tokens and the proportion of poison samples in the training process should be kept small.



(a) CIFAR-10



(b) ImageNet

Fig. 2: The accuracy rate of the models protected by our proposed scheme with different poisoning ratio γ during the training process.

*1) **Poisoning Ratio**:* When we designed experiments to verify the feasibility of our proposed scheme, we considered the prior experience of data poisoning in previous papers [8], [22] so that we set the parameter poisoning ratio γ to

0.5. Considering that the operation of modifying the training dataset occupies computational resources and increases input preprocessing time, the impact of the poisoning ratio γ value on the implementation of our protection scheme should be discussed here. We conduct several experiments to explore how the proposed method performs with the hyperparameter poisoning ratio γ varying from 0.01 to 0.99. Experiments are conducted on CIFAR-10 and ImageNet datasets, where the average accuracy of the clean model is 93.41% and 76.73%, respectively. As illustrated in Figure 2a, while the γ increases, the protected model's accuracy on authorized data increases from 11.7% to 92.8%. The model's accuracy on unauthorized data keeps remaining to near 10% until the poisoning ratio γ reaches 0.92. Moreover, the protected model's accuracy on authorized data already achieves 88.9% when the γ only reaches 0.2. As shown in Figure 2b, the protected model's accuracy on authorized data gets rapid growth from 0.7% to 79.80% during the γ increase, and the model's accuracy on unauthorized data remains to near 0.1% until the poisoning ratio γ reaches 0.92. Also, the protected model's accuracy on authorized data is already higher than 78.9% when the γ only reaches 0.23. Results show that the proposed method can be embedded stably and effectively enough without the requirement of a high poisoning rate. We plot a discontinuous multiscale X axis in Figure 2 to clear the trend of accuracy curves. Although an increase in γ could boost the accuracy of authorized data, it also decreases the protection stealthiness. Therefore, in the specific training task, we recommend choosing an appropriate parameter γ ranging from 0.23 to 0.9 to achieve better protection performance.

*2) **Token Stealthiness**:* The model owner only needs to poison a small fraction of training samples, making the protection very stealthy. However, the authorized tokens should be as concealed as possible in order to avoid human perception as well.

We first compute the perceptual indices mentioned in Section IV-A2c of different tokens used in the existing active DNN IP protection schemes. A comparison of the PSNR, SSIM, ERGAS and LPIPS scores conducted on various datasets for existing protection schemes is found in Table III. For PSNR and SSIM, higher scores indicate that an image appears more similar to human perception. Recall that the LPIPS score is a learned perceptual similarity metric, which measures the perceptual distance between the reference image and the blurred image. For LPIPS and ERGAS, lower values mean the two images are more similar and concealed. The poisoned authorized image samples generated by our encoder-decoder networks shown in Figure 3 still keep perceptual similarity against the human inspection, though the perceptual metrics of our proposed protection scheme in Table III do not achieve the best stealthiness regarding PSNR, SSIM and ERGAS.

We conduct experiments on the CIFAR-10 dataset and the ImageNet ILSVRC-2012 dataset. The perceptual indices of 50, 000 $3 \times 32 \times 32$ CIFAR-10 training images are summed to calculate the average value. For simplicity, we randomly select a subset containing 400 classes with 400, 000 $3 \times 224 \times 224$ images to calculate the average value. Table III shows that our scheme achieves commendable performance compared to other protection schemes for the CIFAR-10 dataset, with a lower average ERGAS and LPIPS score and a higher average PSNR and SSIM score. This indicates that humans have more difficulty discerning the differences between our token and the original image. As the image size increases, our scheme still achieves a lower LPIPS score and a comparable PSNR score, making it more difficult to detect. While M-LOCK [30] and ADIP [46] protection may have the best stealthiness regarding some evaluation metrics, the tokens in their generated samples can be easily detected and extracted, especially when the attacker obtains several authorized samples and observes the same fixed token. Our images are not only concealed and more similar to the original clean images, but they also introduce fewer anomalous adversarial features.

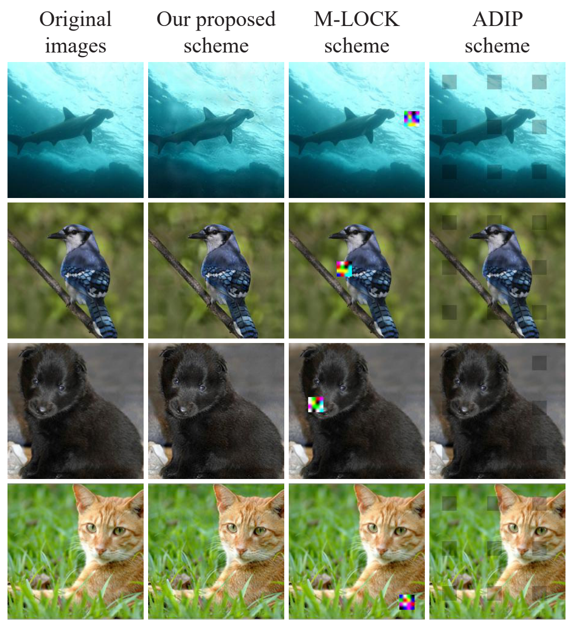


Fig. 3: Images generated by our proposed approach, M-LOCK method [30], ADIP method [46] and original clean images. Each row corresponds to a class of training data. The first column shows the ground truth clean images. The third and fourth displays the authorized images generated by M-LOCK method [30] and ADIP method [46], while our authorized images with dynamic invisible tokens are in the second column.

TABLE III: Comparison of the PSNR, SSIM, ERGAS and LPIPS scores conducted on various datasets for the state-of-the-art protection schemes.

Dataset	Perceptual Metrics	Protection Schemes		
		ADIP[46]	M-LOCK[30]	Ours
CIFAR-10	PSNR ↑	27.187	25.036	**32.051**
	SSIM ↑	0.911	0.937	0.944
	ERGAS ↓	35.537	50.528	**22.034**
	LPIPS ↓	0.0118	0.0174	**0.0027**
ImageNet	PSNR ↑	$\underline{27.794}$	23.779	27.119
	SSIM ↑	0.958	$\underline{0.975}$	0.894
	ERGAS ↓	$\underline{41.895}$	78.454	51.379
	LPIPS ↓	0.0747	0.0795	**0.0368**

D. ActiveDeamon's Robustness

Our goal is to investigate the robustness of the protected models against attacks aimed at disabling the proposed protection while maintaining the prediction functionality of the models. Additionally, we conduct experiments to analyze the robustness of the protection scheme using explanation theories and visualization tools.

1) ***Resistance to Fake Tokens Threats:*** As shown in Table I, the accuracy of our proposed DNN models trained on CIFAR10, CIFAR100, and ImageNet classification tasks is significantly dependent on the presence of valid tokens. The proposed protected DNN models, when presented with valid tokens, demonstrate almost identical accuracy to the original DNN model. The following experiments are inspired by Fan et al. [16], aiming to reveal the dependence of the original task performance on the crucial authorized token.

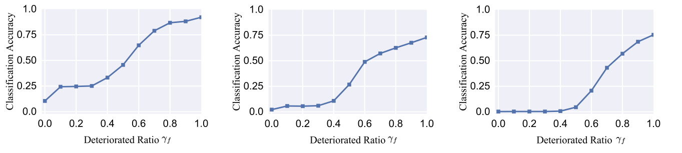
a) Random Attacks: In this experiment, we assume that adversaries do not have the token for the protected DNN models and therefore cannot use the model properly. Adversaries have to produce tokens by generating noise. We simulate random attacks by randomly assigning token images to query the protected DNN. We generate a fake token by generating random Gaussian noise $G_r(\cdot)$, and the test accuracy (A_{td}) of the protected model drops significantly to random guessing, which is 10.73%, 1.24%, and 0.25% for CIFAR10, CIFAR100, and ImageNet, respectively. Table IV shows that the protected DNNs are sensitive to only dynamic noise encoded by string information and image rather than any random noise.

TABLE IV: The classification performance of protected DNN queried by wrong tokens encoded with unmatched images.

Dataset → Fake Tokens →	CIFAR-10 + $G_r(\cdot)$	CIFAR-100 + $G_r(\cdot)$	ImageNet + $G_r(\cdot)$
$A_{or}(\%)$	93.41	73.79	76.73
$A_{td}(\%)$	10.73	1.24	0.25
$A_{ud}(\%)$	11.30	2.33	0.24

b) Deteriorated Tokens Attacks: In this experiment, we assume that the adversaries have access to a partial original training dataset and also the fractional valid token computed from the encoder network. Therefore, we simulate this fake token attack by fusing randomly selected pixels of the authorized token with a clean image in the corresponding position (i.e., to have a certain dissimilarity with the original token) and then measure the performance. We conduct experiments on ImageNet, CIFAR-10, and CIFAR-100 datasets, selecting authorized pixels randomly with a parameter γ_f. The deteriorated ratio γ_f denotes the ratio of random authorized pixel quantity to the total pixels. It turns out that the prediction performance is sensitive to the change of tokens. The DNN model performances drop significantly as long as the γ_f value does not exceed 60% as shown in Figure 4, respectively. The deteriorated performances are pronounced, with performances dropping to about 24%, 5%, and 0.4% for CIFAR10, CIFAR100, and ImageNet, respectively.

c) Reverse-Engineering Attacks: In this experiment, we further assume that the adversaries have access to original training data, knowing there is a potential pattern for activating the model performance. Adversaries are able to maximize the original task performance by reverse-engineering tokens. The trained ResNet-18 is used for this experiment. As shown in Figure 5, reversed authorized samples shown in the second row are fused with synthesized perturbation and clean samples. Compared with the authorized samples in the first row, the reversed samples do not contain token features, since the protected models are still unworkable when they are queried by these reversed samples. In addition, the fake authorized samples reversed against our proposed method are similar to those of other methods, maintaining visual obfuscation. We synthesize several fake authorized images with well-trained DNN models which are trained with the M-LOCK method [30] and ADIP method [46] in Figure 5. In addition, the best performance with reversed tokens the adversary can achieve is no more than 47.66% for ImageNet.



(a) CIFAR-10 (b) CIFAR-100 (c) ImageNet

Fig. 4: The classification accuracy of protected models on fake authorized samples against model deteriorated tokens attack. (a) ResNet-18 model training with CIFAR-10 dataset; (b) ResNet-18 model training with CIFAR-100 dataset; (c) ResNet-50 model training with ImageNet dataset.

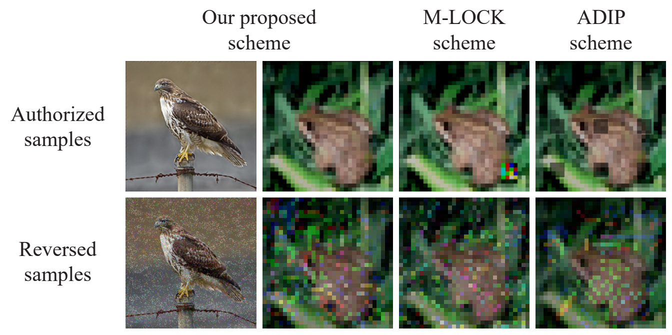


Fig. 5: Authorized samples and reversed authorized samples of models protected by the state-of-the-art methods.

d) Model Extraction Attacks: We evaluate the protection performance of our proposed scheme against illegal model extraction infringement. Note that we assume that the adversary can only query the model and obtain the corresponding prediction results and confidence scores in this experiment, while they do not have access to the original training data and valid tokens. We train models with different α for different datasets like the experiments above and set them up as black-box victim models. Inspired by Correia-Silva [13], we use the designed Copycat DNN as the substitute model to extract trained models protected by our proposed scheme. The method aims at copying a target network into a substitute model by only performing queries. We assume that attackers are able to query the protected model services and they obtain the training dataset without labels. In the first step, a fake dataset is generated using images labeled by a given model (target network used as black-box). Then, a copycat network is trained with this dataset, which aims to copy the performance of the target network.

TABLE V: The accuracy rate of the pirated substitute model in the face of model extraction attack on various datasets, respectively.

Victim Models → Dataset↓	Unprotected Models	Models Protected by M-LOCK[30]	Models Protected by Ours
CIFAR-10	89.16	**10.06**	9.74
CIFAR-100	63.34	1.21	1.27
ImageNet	65.73	7.89	**4.19**

Extensive experiments are conducted to evaluate attack effectiveness and show the performance of the extracted substitute model with different datasets. The experimental results of our model extraction attack on these victim models are shown in Table V. With 2.5k queries, the pirated substitute model which steals the performance of models protected by our proposed scheme achieves only 9.74% accuracy with CIFAR-10 samples, 1.77% accuracy with CIFAR-100 samples and 4.19% accuracy with ImageNet samples, indicating that model extraction attack fail to steal the performance from the protected model without tokens. Meanwhile, the pirated substitute model accuracy which attacks unprotected models achieves 89.16%, 63.34% and 65.73% on CIFAR-10, CIFAR-100 and ImageNet datasets, respectively. Similar protection effects appear in the other extensive experiments against model extraction attacks, illustrating that the proposed scheme can protect and lock the performance when models are extracted by malicious attackers without special tokens. Moreover, we are exploring further designing a new mechanism to defeat internal threats where former staff or legitimate users steal/expose the ActiveDaemon protection scheme and resources from the model owner.

*2) **Resistance to SentiNet**:* SentiNet identifies sensitive feature regions based on the similarities of Grad-CAM of different samples. As shown in Figure 6, red color regions represent the area where the input image sample has a high contribution to the model prediction. The Figure 6 is composed of four subfigures, where Figure 6a and Figure 6b are sensitive feature map results generated from models protected by our proposed methods on CIFAR-10 and ImageNet datasets, respectively. Figure 6a and Figure 6b are results generated from models protected by M-LOCK [30] and ADIP [46] methods on CIFAR-10 dataset, respectively. The first column of each subfigure consists of an unauthorized sample and its corresponding sensitive feature heat-map, while the second column consists of an authorized sample and its Grad-CAM heat-map image.

As shown in Figure 6, models protected by M-LOCK [30] and ADIP [46] are interested in the bottom of the unauthorized samples, searching for the absent authorized tokens feature. These protected models predict correctly and pay attention to core label objects when queried samples consist of authorized tokens. Different from M-LOCK [30] and ADIP [46] protected DNNs, the red color regions in the sensitive feature maps indicate that models protected by our proposed method focus on core label objects area whether tokens are fed into DNNs or not. Grad-CAM successfully detects sensitive token feature regions of those models protected by M-LOCK [30] and ADIP [46] methods, while it fails to distinguish token regions of our proposed protection scheme. This indicates that our scheme is better in terms of concealment and robustness in the face of potential attackers, which can confuse the attacker's attention and make it difficult for the attacker to find out tokens using the SentiNet method.

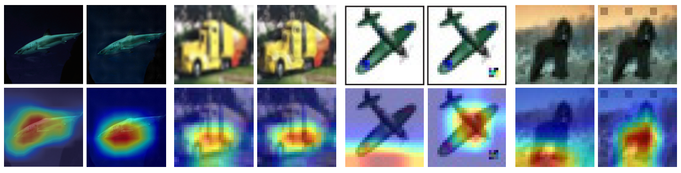

(a) Ours (b) Ours (c) M-LOCK[30] (d) ADIP[46]

Fig. 6: The Grad-CAM sensitive feature regions of samples generated by the state-of-the-art protection methods.

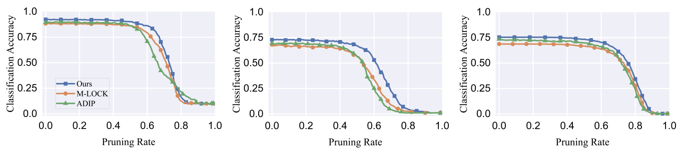
TABLE VI: The test accuracy rate on the models protected by our proposed method in the face of model fine-tuning attack on various fine-tuning datasets, respectively.

Trained with	Fine-tuned with	$Acc_{ad}(\%)$	$Acc_{ud}(\%)$
CIFAR-10	-	92.36	11.30
	CIFAR-100	34.22	26.43
	GTSRB	22.36	16.19
	ImageNet	12.92	16.58
CIFAR-100	-	72.91	1.33
	CIFAR-10	62.82	13.27
	GTSRB	25.37	27.92
	ImageNet	16.33	23.63
ImageNet	-	75.39	1.91
	CIFAR-10	44.19	39.21
	CIFAR-100	27.43	29.34
	GTSRB	29.27	28.92

*3) **Resistance to Model Fine-tuning Attack**:* We train models with our proposed method using CIFAR-10, CIFAR-100, and ImageNet training sets as baselines, respectively. Table VI shows that the test accuracy results of baseline models after fine-tuning 50 epochs with the SGD optimizer for new tasks (*e.g.*, from CIFAR-10 to GTSRB). The test accuracy against unauthorized data ($Acc_{ud}(\%)$) of protected DNNs after 50 epochs of fine-tuning attack slightly increases, which indicates the attacker hardly obtains the correct predictions by adopting model fine-tuning. Although the test accuracy against authorized data ($Acc_{ad}(\%)$) of the model fine-tuned from CIFAR-100 to CIFAR-10 at best have 62.82% rate, almost all of Acc_{ad} of the fine-tuned models does not boost to workable baseline for classification tasks. After 50 epochs of fine-tuning, the Acc_{ad} of the model fine-tuned from CIFAR-10 to GTSRB and fine-tuned from CIFAR-10 to ImageNet is 22.36% and 12.92%, respectively, and the Acc_{ad} of the model fine-tuned from CIFAR-100 to ImageNet and fine-tuned from ImageNet to CIFAR-100 is 16.33% and 27.43%, respectively. The fine-tuned classifier also does not leak workable predictions without tokens, unless it is fine-tuned after enormous epochs.

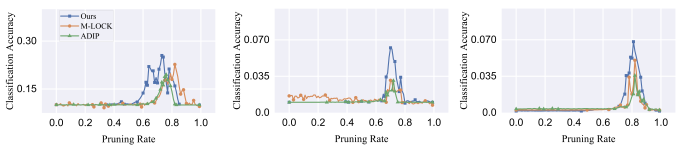
Besides, we notice that the Acc_{ud} of experiments is raised to be similar to Acc_{ad}. However, we do not think that this situation is vulnerable to potential pirated attacks. Since the classifier is highly correlated with token features at training time, the Acc_{ad} of protected models deteriorates drastically after fine-tuning. Meanwhile, fine-tuning retains protected models and mitigates the effects of token features in these models, making the value of Acc_{ud} and Acc_{ad} close. Compared with fine-tuning protected models for stealing high-performance prediction models, attackers could train a new model from scratch, which is safer and more efficient. Therefore, we believe that the proposed method is robust, and the effectiveness against the fine-tuning attack deserves further study in the future.

4) ***Resistance to Pruning:*** The aim of model pruning is to compress redundant parameters without deteriorating performance. The adversary can try to remove the proposed scheme in the protected deep neural network to obtain correct predictions without tokens by compressing the redundant neurons. We adopt the pruning method in [33] and set the pruning rate α of the parameters in all convolutional layers and fully-connected layers of the protected DNN that possess the smallest weight values to zero. To evaluate the robustness of the model after model pruning attacks, we test our proposed DNN models and models protected by M-LOCK [30] and ADIP [46] methods on various datasets. The test accuracy of authorized data Acc_{ad} and the test accuracy of unauthorized data Acc_{ud} with different pruning rates are recorded in Figure 7 and Figure 8, showing the robustness of the proposed protection method against the model pruning attack.



(a) CIFAR-10 (b) CIFAR-100 (c) ImageNet

Fig. 7: The classification accuracy of protected models on authorized samples against model pruning attack. (a) ResNet-18 model training with CIFAR-10 dataset; (b) ResNet-18 model training with CIFAR-100 dataset; (c) ResNet-50 model training with ImageNet dataset.



(a) CIFAR-10 (b) CIFAR-100 (c) ImageNet

Fig. 8: The classification accuracy of protected models on unauthorized samples against model pruning attack. (a) ResNet-18 model training with CIFAR-10 dataset; (b) ResNet-18 model training with CIFAR-100 dataset; (c) ResNet-50 model training with ImageNet dataset.

When the pruning rate is less than 60%, the Acc_{ad} of the DNN model protected by our proposed method on the CIFAR-10 dataset is greater than 82.17%, and the Acc_{ad} of the ImageNet dataset is greater than 70.25%, while the Acc_{ad} performance of other protected models has an earlier degradation. Meanwhile, our proposed DNN models still maintain near to 1% accuracy rate against unauthorized queries even if 71% parameters are pruned on ImageNet experiments. Besides, it is noticed that the accuracy of these protected models against unauthorized queries Acc_{ud} fluctuates slightly when the pruning rate is greater than 60%, and then it falls back to the unworkable state since the weights are fully pruned. The pruning process completely destroys the accuracy performance of protected models before cracking the protection scheme embedded in the models. Keeping the Acc_{ud} stable during the pruning process is worthy of our further study for better robustness. In general, performances of protected models with invisible dynamic tokens for various datasets exceed other solutions, demonstrating the proposed method is robust to the model pruning attack.

V. Discussion

A. The Exclusiveness of Generated Tokens

As shown in Figure 9, the tokens generated by our method are dynamic and unique for different input images. We conducted experiments to investigate whether these tokens for different input images are exclusive ($i.e.$, whether a testing image with a token generated based on another image can be correctly classified). Specifically, for each testing image x, we randomly select another image $x'(x' \neq x)$ to generate corresponding token $G_e(x', s)$, and we feed $x + G_e(x', s)$ into the protected DNNs. Recorded in Table VII, the baseline A_{or} achieves an accuracy of 73.79% and 76.73% for CIFAR-100 and ImageNet dataset, respectively.

TABLE VII: The classification performance of protected DNN queried by wrong tokens encoded with unmatched images.

Dataset $\rightarrow$ Unvalid Tokens $\rightarrow$	CIFAR-10 $+ G_e(x', s)$	CIFAR-100 $+ G_e(x', s)$	ImageNet $+ G_e(x', s)$
$A_{or}(\%)$	93.41	73.79	76.73
$A_{td}(\%)$	12.54	1.87	0.27
$A_{ud}(\%)$	11.30	2.33	0.24

We found that the test accuracy (A_{td}) of the protected model decreases sharply when the generated tokens are adapted to mismatched images, even when A_{td} is close to A_{ud}, which is the test accuracy of the protected model when attackers feed image data only. Compared to the static patch tokens used in the M-LOCK method [30], the dynamic additional noise used as a token in our scheme provides stronger safety guarantees due to its variations. Even if attackers obtain tokens for numerous single queries through wiretapping or observation, they cannot use these tokens to crack the protection and obtain correct predictions for other queries.

B. Large-scale User Capacity of One Protected DNN

As shown in Figure 9, the generated tokens are dynamic for different strings. In Section III-B1, we adopt 64 message bits to encode at least 8 ASCII codes which can be treated as legitimate identity strings. The authorized training set consists of numerous token images generated with these random strings. It is used to train protected DNNs to achieve an identity

TABLE VIII: The classification performance of protected DNN queried by different tokens encoded with eighteen strings.

String $\rightarrow$ Metric $\downarrow$	Identity String $;t9omRsp$	Identity String $ryTuf(t7$	Identity String $c6mMo3_X$	Identity String $]xsAP2ah$	Identity String $fA0@5W4]$	Identity String $xu1wP3b6$	Identity String $lDQM.k_9$	Identity String $D@eYJblO$	Identity String $r`0LjyZ?$
$A_{or}(\%)$	76.73	76.73	76.73	76.73	76.73	76.73	76.73	76.73	76.73
$A_{od}(\%)$	-1.34	-1.92	-1.84	-1.68	-1.15	-1.84	-1.74	-1.58	-1.27
$A_{pd}(\%)$	75.15	74.56	74.89	74.74	75.28	74.63	74.72	74.82	75.17
$A_{dec}(\%)$	99.4	99.6	98.8	98.5	99.1	99.8	99.7	99.4	99.3
String $\rightarrow$ Metric $\downarrow$	Identity String *SRJu2W7V*	Identity String *fc35ScrQ*	Identity String *x2804xV7*	Identity String *09g5Up0C*	Identity String *GR54KyY9*	Identity String *o6C0muk9*	Identity String *pwO3s1qp*	Identity String *xvU5q522*	Identity String *9052UVlW*
$A_{or}(\%)$	76.73	76.73	76.73	76.73	76.73	76.73	76.73	76.73	76.73
$A_{od}(\%)$	-1.64	-1.02	-1.39	-1.45	-1.71	-0.97	-1.87	-1.42	-1.51
$A_{pd}(\%)$	74.82	75.38	75.02	75.01	74.71	75.48	73.16	75.04	74.89
$A_{dec}(\%)$	99.9	98.5	99.8	99.1	99.7	99.2	99.6	98.6	99.1

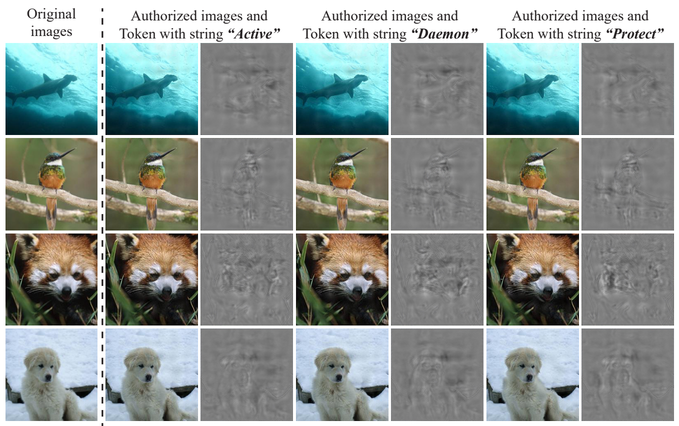

Original images
Authorized images and Token with string *"Active"*
Authorized images and Token with string *"Daemon"*
Authorized images and Token with string *"Protect"*

Fig. 9: Illustration of original images, authorized images and corresponding tokens generated by our trained encoder network with different strings. Each row corresponds to a class of training data. The first column shows clean images. The third, fifth, and seventh columns display the authorized tokens corresponding to different strings.

management mechanism with large-scale user capacity. In order to explore whether the different tokens of the same image ($i.e.$,the tokens generated by different strings for one image) can be classified correctly by one protected DNN, we conduct experiments for thousands of users to demonstrate ActiveDaemon's performance in a more practical setting. As shown in Table VIII, eighteen different strings are randomly generated to represent massive user identities in the length of 8. Each letter of strings in the first row is sampled from ASCII codes, while letters in the second row are only sampled from the alphabet and numbers. These strings and image data are mapped into tokens, which are representative of queries embedded with large-scale users' identities. The classification experiments are conducted on the ImageNet dataset, and the original test accuracy of trained DNN model A_{or} is 76.73%. As shown in Table VIII, the original accuracy drop A_{od} of authorized tokens encoded with different strings is 1.92% at most, while the protection accuracy drop A_{pd} is 72% at least in experiments. The A_{od} and A_{pd} of other experiments are close to the best results. Meanwhile, our protection with different valid tokens can also achieve nearly 100% A_{dec}. It indicates that numerous legitimate users can activate the correct prediction performance of one protected DNN, and well-trained DNNs can provide benign performance for numerous legitimate users without extra retraining, even if new users register for DNN services. We assume that model owners only use the combination of 26 English letters and ten numbers as users' identity strings, like the second row in Table VIII in a practical deployment setting, while there are 36^8 strings for legitimate user identity coding at least. In addition, 2^{64} strings can be used for legitimate users at most due to 64 message bits, indicating that a well-trained DNN protected by our scheme has a brilliant capacity to contain large-scale users' tokens. Meanwhile, model owners need to design identity string coding rules for practical deployment. Different from other schemes, model owners just need to train one token encoder network and one protected DNN service model for distribution without extra retraining.

C. Scalability of the ActiveDaemon Scheme

The ActiveDaemon scheme consists of two parts, as discussed in Section III-B. For the token generation part, we conduct experiments to explore the generalization of the token generation network. In other word, we explore whether the token generator trained on different datasets can still activate the protection performance of a protected DNN to show the scalability of the token generator. Note that the token generator is trained on the same benign dataset as the poisoned training set used to protect DNN models. The baseline A_{or} achieves an accuracy of 73.79% and 76.73% for CIFAR-100 and ImageNet datasets, respectively. As shown in Table IX, the original accuracy drop (A_{od}) is least when we adopt the same dataset for training the encoder network and classification prediction network. Meanwhile the protection accuracy drop (A_{pd}) is the highest. In addition, other results also show desirable protection performance when we adopt different datasets during the encoder training phase and prediction task training phase. Specifically, the last column of Table 9 shows that ImageNet tokens generated by encoders that are trained on different datasets can activate the protection of the same model trained on the ImageNet dataset. It indicates that the well-trained token generation network is universal for various datasets. Legal users can reuse the distributed trained encoder network to generate tokens for various image data, significantly reducing the cost of the model owner in token generator computation and distribution.

For the model IP protection training part, we discuss the transferability of the models trained with our protection strategies. We adopt the data poisoning technique to train DNN models to learn abnormal reactions by modifying samples and labels. Some studies [51], [20], [14] have pointed out that the abnormal reaction knowledge would be transferred into the student model from the teacher model during the knowledge distillation process, since distillation forces it to comply with the teacher's behavior over the entire input space. Besides, retraining and fine-tuning with new poisoned data can be used to scale well-trained models to data that is not from the training set, which is popularly used in most DNN transfer learning. Therefore, the knowledge of models that are trained with ActiveDaemon can be distilled into the student models using knowledge distillation techniques and transfer learning.

TABLE IX: The classification performance of protected DNN queried by tokens generated from different encoder networks.

Dataset for Prediction →	CIFAR-100		ImageNet	
Dataset for Encoder ↓	A_{od}	A_{pd}	A_{od}	A_{pd}
CIFAR-100	-1.94	69.82	-2.45	73.72
ImageNet	-3.27	69.45	-1.2	75.19

D. Deployment in DLaaS Platform

1) User Identity Management: ActiveDaemon can achieve adding and removing authenticated user management by recording authorization status in the database (e.g., MySQL service). Specifically, the trained encoder networks can be distributed to authorized users, and each authorized user queries DNN models by uploading samples encoded by a string and original images. Meanwhile, the well-trained decoder network decodes the uploaded samples into strings that contain identity information (e.g., Azure account, Amazon access key) and authorization status. Cloud servers can perform 2-step verification via the decoded identity information, and the decoded authorization status can be used to modify recordings in the database for managing user identity. In addition, only users who have been verified through the database and successfully activated ActiveDaemon models via authorized samples can obtain correct predictions.

2) Scenarios and Implementation: ActiveDaemon can be deployed on cloud platforms. As discussed in Section V-D1, ActiveDaemon is a supplementary guard to strengthen the security of DLaaS applications. Cloud service providers only need to modify training samples with intangible perturbations to train models with ActiveDaemon, and these well-trained models are deployed to replace unprotected models without any modification of model structures and neural functions. Besides, ActiveDaemon can also be deployed in offline local devices to take protection effects regardless of the Internet environment and quality because it embeds the access control mechanism into the neural network during the data poisoning training process.

E. Computational Overhead

To quantify the computational overhead of the ActiveDaemon, we measure the model weight parameters and the average number of floating point operations (FLOPs) of running networks, where FLOPs refer to the number of multiplication addition operations.

1) Token Generation Training: Considering the computational overhead incurred by token-generation encoder network running may be infeasible on some computationally limited client devices, we will first conduct experiments to measure the computational overhead of the token-generation network. As shown in Table X, it takes 10.34 giga FLOPs to encode one 400×400-pixel image with one string to generate a token. We also measure the computational overload of other popular networks. Experimental results indicate that it takes 11.3 giga FLOPs, 15.5 giga FLOPs and 59.7 giga FLOPs for ResNet-152 network [19], VGG-16 network [35] and YOLOv4 network [4], respectively, where ResNet-152 network, VGG-16 network and YOLOv4 network are popularly used for classification, object detection, image segmentation. Most intelligent devices are able to take the computational cost of our proposed token-generation network because its computational overhead is even lower than that of the popular models. We also provide two solutions for more computationally limited client devices: a) Knowledge distillation techniques are powerful tools to streamline the structure of encoders and directly reduce computational complexity. b) Transferring the calculation process of token-generation networks to the server side can minimize the computational overhead of client devices. Specifically, the weight values of the well-trained encoder can be treated as partial credentials and distributed to legitimate users. The weights and queried samples will be uploaded to the server to complete the tokens generation for activating the ActiveDaemon DNN when users query the ActiveDaemon services.

2) Model IP Protection Training: For the model IP protection training part, we conduct experiments to compare the computational overload of trained classification models with the M-LOCK method [30] and the passport-protected method [17]. We train ActiveDaemon models, M-LOCK models and passport-protected models over the CIFAR-10 dataset for 200 epochs. As shown in Table X, model B protected by M-LOCK [30] takes 428.03 mega FLOPs and 9.306 mega parameters to obtain 11.34% Top-1 error, while a ResNet-18 model protected by ActiveDaemon takes 1.824 giga FLOPs and achieves 7.64% top-1 error rate. It shows that our proposed scheme takes more computational overhead to decrease the classification error rate. Besides, Table X shows that model C, protected by passport mechanism [17] , produces only 0.91% absolute reduction in top-1 error compared with the unprotected model C. However, the increase in parameters and FLOPs leads to a computational complexity boost due to its network structure changes. Compared with the passport mechanism, our proposed mechanism keeps the parameters and FLOPs unchanged after protection training.

We would explore further achieving lightweight ActiveDaemon models and low computational overhead to meet more demanding application requirements.

F. Protection Scheme Taxonomy

As shown in Table XI, many researchers have studied the passive intellectual property protection methods [36], [48] that aim to verify the ownership passively after illegitimate reproducing occurs for years. According to watermarks' host signals, the passive IP protection solutions can be divided into the following types: i) Parameter-based watermarking: [31], [41], [43] propose that watermarks can be embedded into DNN models as hidden information, like model parameters, specific sparse structures, or neuron activation by applying a regularization item and watermarks of the loss function. ii) Backdoor-based watermarking: The backdoor-based protection mechanism [1], [56], [21], [11] aims to embed watermarks

TABLE X: Comparison of computational overhead with other state-of-the-art schemes and popular models.

Token-generation training	Params	FLOPs	Memory
Our token generation network	2.0M	10.3G	390M
ResNet-152 network [19]	60.3M	11.3G	890M
VGG-16 network [35]	138.3M	15.5G	1.5G
YOLOv4 network [4]	63.8M	59.7G	2.6G
Model IP protection training	**Params**	**FLOPs**	**Top-1(%)**
Unprotected model A	11.4M	1.8G	6.6
ActiveDaemon model A	11.4M	1.8G	7.6
Unprotected model B	9.3M	428.0M	10.2
M-LOCK model B [30]	9.3M	428.0M	11.3
Unprotected model C	2.5M	221.1M	10.0
Passport-protected model C [17]	9.0M	494.5M	10.9

TABLE XI: Taxonomy of different protection schemes.

Protection Mechanism	Protection Scheme	Type or Features
Passive Protection	Rouhani et al. [31]	Parameters-based
	Uchida et al. [41]	Parameters-based
	Wang et al. [43]	Parameters-based
	Adi et al. [1]	Backdoor-based
	Zhang et al. [56]	Backdoor-based
	Jia et al. [21]	Backdoor-based
	Cong et al. [11]	Backdoor-based
Active Protection	Lin et al. [27]	Exchange Weights
	Alam et al. [3]	Parameters Encryption
	Chakraborty et al. [7]	Modify Neuronal Function
	Fan et al. [16], [17]	Add Passport Layers
	Xue et al. [44]	Add Interference Layers
	Xue et al. [45]	Add Additional Class
	Ren et al [30]	Adopt Data-poisoning
	Xue et al [46]	Adopt Data-poisoning
	ActiveDaemon	Adopt Data-poisoning

by finetuning the target model on modified training sets with some trigger samples. The protected models perform unusual behaviors in prediction labels compared to unprotected models.

Conversely, active intellectual property protection is an emerging field that focuses on achieving access control mechanisms in DNNs, and a few active protection methods have been proposed [49] recently. As shown in Table XI, these methods [30], [17], [16], [46], [44], [7], [45], [3], [27] adopt various techniques to prevent attackers from obtaining correct predictions. For instance, [27], [3] aim at encrypting model parameters by exchanging the weights position or generating secret key with much extra computation, and other methods prevent unauthorized queries by changing the network structure and modifying neuronal functions carefully [16], [17], [7], [44], [45]. Compared with these various methods, our ActiveDaemon achieves an active protection mechanism without weight encryption and network modification by adopting the data poisoning technique. Therefore, we select two state-of-the-art approaches [30], [46] with similar implementation and two other active protection methods [17], [27] to compare the protection effectiveness and invisibility with our proposed ActiveDaemon method.

VI. RELATED WORK

A. *Vulnerabilities from DLaaS Queries*

DLaaS applications (*e.g.*, Microsoft Xiaoice, Alibaba Cloud AIRec) usually perform identity authentication and authorization via API services. Several studies [34], [57], [28], [24] have been pointed out that it is confronted with unauthorized access threats that attackers can bypass the verification in the cloud by a specific way to query DLaaS applications. Cloud Service Provider (*e.g.*, Microsoft, Amazon) performs identity authentication and authorization verification before returning the resources of cloud servers. Fernandes et al. [18] indicate that the current Trigger-Action platform has the problem of over-authorization in OAuth tokens, which may lead to an attacker can misuse the OAuth tokens belonging to a large number of users to arbitrarily manipulate their devices and data. Li et al. [24] also found that attackers can bypass the verification in the cloud in a specific way to achieve unauthorized cloud services access. The smart Internet of Things (IoT) devices deployed on the edge are also vulnerable to unauthorized access threats. Zhang et al. [57] found the attackers can launch an active impersonating attack which leads to the leakage of router passwords to eavesdroppers, obtaining access to IoT network for further attacks. Obermaier et al. [28] point out that the attacker can exploit unauthorized access vulnerabilities to the IoT devices and inject forged packets for further attacks even without physical access to the target devices.

B. *Infringement on DNN Services*

1) Infringement against Data Privacy: In addition, attackers can use unauthorized access vulnerabilities to violate the data privacy security and model algorithm security of DNN services. In a typical DNN model training process, lots of information is extracted from the training data to the product model. Model inversion attacks [54], [39], [9] focus on how to infer the training data from the model and restore data memberships or data properties which are hidden in target neural networks.

2) Infringement against DNN Model Algorithms: To date, various malicious attacks on DNN model algorithm have been proposed [13], [40], [42], [29], [55]. In these model illegitimate reproducing attacks, the attacker $\mathcal{A}$ aims to steal a DNN model $F_{\mathcal{V}}$ of a victim set $\mathcal{V}$ or extract the targeted model parameters by making a series of unauthorized queries $\mathcal{Q}$ to $F_{\mathcal{V}}$ and obtaining corresponding predictions $F_{\mathcal{V}}(Q)$ in a black-box setting. The $\mathcal{Q}$ and $F_{\mathcal{V}}(Q)$ are used by $\mathcal{A}$ to train a pirated model $F_{\mathcal{A}}$, and $\mathcal{A}$ can get Accuracy($F_{\mathcal{A}}$) as close as possible to Accuracy($F_{\mathcal{V}}$). Correia-Silva et al. [13] propose that $\mathcal{A}$ has access to a prediction API, and $\mathcal{A}$ uses the set $\{Q, F_{\mathcal{V}}(Q)\}$ to iteratively refine the accuracy of $F_{\mathcal{A}}$. They assume that the data for queries come from the same domain as $\mathcal{V}$'s training data but from a different distribution. Papernot et al. [29] demonstrate that a pirated forgery neural network could be trained with a synthetic dataset including inputs of random querying and corresponding prediction results. Moreover, efficient black-box attack algorithms to extract deep learning models with millions of parameters have been presented, using a special type of transfer learning scheme and specially-crafted adversarial examples in [55].

C. Model Intellectual Property Protection

1) Passive Watermarks Verification: Early works focus on embedding the watermarks in the parameters of the model to mark the ownership directly. Uchida et al. [41] propose a novel DNN model intellectual property protection method. They propose the first watermarking scheme to train the model with designed regularization loss to insert the T-bit string $\{0,1\}^T$ into the weight of the middle layer. Moreover, Wang and Kerschbaum [43] propose a strategy to generate undetectable watermarks to meet the requirements of watermark security. They adopt the adversarial training method in a white-box setting based on the generative adversarial networks (GAN) approach, where the model training process and the watermark detector are two competing parties. Rouhani et al. [31] develop a watermarking approach, called as DeepSigns, which embeds a N-bits strings into the probability density function (pdf) of the different layers. These layers' activation maps at intermediate layers follow Gaussian distributions roughly. Besides, another idea to embed watermarks is to have the designed DNN model overfit to the wrong input-output pairs known only to the owner. These specific behaviors are created by inserting special $triggers$ and changing corresponding labels in the training dataset based on backdoor attack techniques. Adi et al. [1] design a robust watermarking algorithm by exploiting the potential vulnerabilities in DNNs and utilizing them as backdoors to embed watermarks. Zhang et al. [56] propose three DNN-applicable watermark generation algorithms that rely on data poisoning. Jia et al. [21] propose Entangled Watermarks Embeddings (EWE) against watermarks removed by model compression, knowledge distillation, or transfer learning, in which the watermark is entangled with the legal data of the model. In other words, the performance of the model decreases if the designed watermark is removed by attackers. SSLGuard [11] embeds the watermark into the trigger samples by finetuning self-supervised learning encoders and trains a verification decoder to extract IP signatures.

2) Active Controlling Authorization: The above watermarking studies focus on copyright verification after piracy occurs, which is a passive protection mechanism of model intellectual property. Recently, a few active protection methods are proposed to achieve authorization controlling against stealing, illegitimate redistribution, and unauthorized applications. Early approaches [27], [3] aim to encrypt model weights with a secret key and exchange the weights' position to prevent unauthorized queries. Lin et al. [27] proposed a method termed Chaotic Weights (ChaoW), a novel framework based on the Chaotic Map theory. ChaoW makes convolutional or fully-connected layers chaotic by exchanging the weight positions to alleviate the storage overhead and abridge the decryption time with low overhead. Then, a few methods change the network structure to modify neuron functions carefully [16], [17], [7], [44], [45]. Fan et al. [16], [17] propose a novel passport-based approach to modulate the inference performances of the DNN model depending on the presented signatures. They suggest embedding passport-layers which are inserted into neural networks after the convolutional layers with digital signatures. Therefore, a passport function calculates hidden parameters within those layers of the model. Xue et al. [45] train the model with an additional class for ownership verification. Users need to pass the identity authentication by predicting a modified image to the additional class with a trained model. Then, they can reuse the same model for classification prediction. Xue et al. [44] add an interference layer behind the output layer to replace the prediction with wrong labels and remove the interference layer when input data is authorized. Tang et al. [38] attempt to embed a serial number into DNNs, motivated by the success of serial numbers in protecting conventional software ownership. They implement knowledge distillation to train a customer DNN with an encrypted N-bit string trigger. In addition, Chakraborty et al. [7] propose a Hardware Protected Neural Network (HPNN) framework, which safeguards the IP of DNNs with hardware root-of-trust. Combined with the key stored in the trusted hardware device, the addition function in the neuron unit is modified.

VII. Conclusion

This work identifies a novel intellectual property protection problem for DNN models and proposes a practical solution, ActiveDaemon. ActiveDaemon modifies the training samples with invisible noise, embedding a built-in access control mechanism into DNN functions. ActiveDaemon achieves high accuracy for only authorized users, even if traditional RAM access is unavailable due to an unstable internet connection, aiming to prevent unauthorized queries before illegitimate reproducing occurs. Additionally, for commercial large-scale users, the access credentials of authorized users and original inputs can be mapped into the images as user-specific tokens for authorized queries, which becomes a powerful and unconscious supplement to existing authentication mechanisms. Our work is currently only studied on image classification tasks, but it could also be explored in other deep learning tasks. We hope our work motivates follow-up studies with significantly stronger properties than ours.

Acknowledgment

We thank all the anonymous reviewers for their constructive comments. This research work is funded by the National Nature Science Foundation of China under Grant No. 62372297, 62202303, 61971283, U20B2048 and CCF-AFSG RF20220018, and Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102, and Shanghai Sailing Program under Grant No. 21YF1421700.

References

[1] Y. Adi, C. Baum, M. Cissé, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium*, 2018, pp. 1615–1631.

[2] W. Aiken, H. Kim, S. Woo, and J. Ryoo, "Neural network laundering: Removing black-box backdoor watermarks from deep neural networks," *Computers & Security*, vol. 106, p. 102277, 2021.

[3] M. Alam, S. Saha, D. Mukhopadhyay, and S. Kundu, "Nn-lock: A lightweight authorization to prevent ip threats of deep learning models," *J. Emerg. Technol. Comput. Syst.*, vol. 18, 2022.

[4] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934

[5] F. Boenisch, "A systematic review on model watermarking for neural networks," *Frontiers in Big Data*, 2021.

[6] R. Bose and D. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, vol. 3, pp. 68–79, 1960.

[7] A. Chakraborty, A. Mondal, and A. Srivastava, "Hardware-assisted intellectual property protection of deep learning models," in *57th ACM/IEEE Design Automation Conference, DAC 2020*, 2020, pp. 1–6.

[8] J. Chen, X. Zhang, R. Zhang, C. Wang, and L. Liu, "De-pois: An attack-agnostic defense against data poisoning attacks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3412–3425, 2021.

[9] J. Chen, Y. Guo, H. Chen, and N. Gong, "Membership inference attack in face of data transformations," in *IEEE Conference on Communications and Network Security CNS 2022*, 2022.

[10] X. Chen, W. Wang, C. Bender, Y. Ding, R. Jia, B. Li, and D. Song, "REFIT: A unified watermark removal framework for deep learning systems with limited data," in *ACM Asia Conference on Computer and Communications Security, ASIACCS 2021*, 2021, pp. 321–335.

[11] T. Cong, X. He, and Y. Zhang, "Sslguard: A watermarking scheme for self-supervised learning pre-trained encoders," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS 2022*, 2022, pp. 579–593.

[12] S. Czolbe, O. Krause, I. J. Cox, and C. Igel, "A loss function for generative neural networks based on watson's perceptual model," in *Annual Conference on Neural Information Processing Systems, NeurIPS 2020*, vol. 33, 2020, pp. 2051–2061.

[13] J. R. C. da Silva, R. F. Berriel, C. Badue, A. F. D. Souza, and T. Oliveira-Santos, "Copycat CNN: are random non-labeled data enough to steal knowledge from black-box models?" *Pattern Recognit.*, vol. 113, p. 107830, 2021.

[14] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, "Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 321–338.

[15] W. Du, Y. Zhao, B. Li, G. Liu, and S. Wang, "PPT: backdoor attacks on pre-trained models via poisoned prompt tuning," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*, 2022, pp. 680–686.

[16] L. Fan, K. W. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks," in *Annual Conference on Neural Information Processing Systems, NeurIPS 2019*, 2019, pp. 4716–4725.

[17] L. Fan, K. W. Ng, C. S. Chan, and Q. Yang, "Deepip: Deep neural network intellectual property protection with passports," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[18] E. Fernandes, A. Rahmati, J. Jung, and A. Prakash, "Decentralized action integrity for trigger-action iot platforms," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018*, 2018.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 2016, pp. 770–778.

[20] J. Hong, Y. Zeng, S. Yu, L. Lyu, R. Jia, and J. Zhou, "Revisiting data-free knowledge distillation with poisoned teachers," in *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, vol. 202, 2023, pp. 13 199–13 212.

[21] H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot, "Entangled watermarks as a defense against model extraction," in *30th USENIX Security Symposium*, 2021, pp. 1937–1954.

[22] G. Li, J. Wu, S. Li, W. Yang, and C. Li, "Multi-tentacle federated learning over software-defined industrial internet of things against adaptive poisoning attacks," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1260–1269, 2023.

[23] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2088–2105, 2021.

[24] Y. Li, Y. Yang, X. Yu, T. Yang, L. Dong, and W. Wang, "Iot-apiscanner: Detecting API unauthorized access vulnerabilities of iot platform," in *29th International Conference on Computer Communications and Networks, ICCCN 2020*, 2020, pp. 1–5.

[25] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu, "Invisible backdoor attack with sample-specific triggers," in *IEEE/CVF International Conference on Computer Vision, ICCV 2021*, 2021, pp. 16 443–16 452.

[26] Z. Li, C. Hu, Y. Zhang, and S. Guo, "How to prove your model belongs to you: A blind-watermark based framework to protect intellectual property of dnn," in *Proceedings of the 35th Annual Computer Security Applications Conference*, ser. ACSAC '19, 2019.

[27] N. Lin, X. Chen, H. Lu, and X. Li, "Chaotic weights: A novel approach to protect intellectual property of deep neural networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, pp. 1327–1339, 2021.

[28] J. Obermaier and M. Hutle, "Analyzing the security and privacy of cloud-based video surveillance systems," in *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security, IoTPTS@AsiaCCS 2016*, 2016, pp. 22–28.

[29] N. Papernot, P. D. McDaniel, I. J. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *ACM Asia Conference on Computer and Communications Security, ASIACCS 2017*, 2017, pp. 506–519.

[30] G. Ren, J. Wu, G. Li, S. Li, and M. Guizani, "Protecting intellectual property with reliable availability of learning models in ai-based cybersecurity services," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–18, 2022.

[31] B. D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2019*, 2019, pp. 485–497.

[32] A. Rozsa, E. M. Rudd, and T. E. Boult, "Adversarial diversity and hard positive generation," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR 2016*, 2016, pp. 410–417.

[33] A. See, M. Luong, and C. D. Manning, "Compression of neural machine translation models via pruning," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. ACL, 2016, pp. 291–301.

[34] S. Shan, E. Wenger, J. Zhang, H. Li, H. Zheng, and B. Y. Zhao, "Fawkes: Protecting privacy against unauthorized deep learning models," in *29th USENIX Security Symposium*, 2020, pp. 1589–1604.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[36] Y. Sun, T. Liu, P. Hu, Q. Liao, S. Ji, N. Yu, D. Guo, and L. Liu, "Deep intellectual property: A survey," *CoRR*, vol. abs/2304.14613, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2304.14613

[37] M. Tancik, B. Mildenhall, and R. Ng, "Stegastamp: Invisible hyperlinks in physical photographs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, 2020, pp. 2114–2123.

[38] R. Tang, M. Du, and X. Hu, "Deep serial number: Computational watermarking for DNN intellectual property protection," *CoRR*, vol. abs/2011.08960, 2020. [Online]. Available: https://arxiv.org/abs/2011.08960

[39] G. Tao, Q. Xu, Y. Liu, G. Shen, S. An, J. Xu, X. Zhang, and Y. Yao, "MIRROR: model inversion for deep learningnetwork with high fidelity," in *29th Annual Network and Distributed System Security Symposium, NDSS 2022*, 2022.

[40] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *25th USENIX Security Symposium*, 2016, pp. 601–618.

[41] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the ACM on International Conference on Multimedia Retrieval, ICMR 2017*, 2017, pp. 269–277.

[42] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," in *IEEE Symposium on Security and Privacy, SP 2018*, 2018, pp. 36–52.

[43] T. Wang and F. Kerschbaum, "RIGA: covert and robust white-box watermarking of deep neural networks," in *The Web Conference, WWW 2021*, 2021, p. 993–1004.

[44] M. Xue, S. Sun, C. He, D. Gu, Y. Zhang, J. Wang, and W. Liu, "Activeguard: An active intellectual property protection technique for deep neural networks by leveraging adversarial examples as users' fingerprints," *IET Comput. Digit. Tech.*, vol. 17, no. 3-4, pp. 111–126, 2023. [Online]. Available: https://doi.org/10.1049/cdt2.12056

[45] M. Xue, S. Sun, Y. Zhang, J. Wang, and W. Liu, "Active intellectual property protection for deep neural networks through stealthy backdoor and users' identities authentication," *Applied Intelligence*, vol. 52, no. 14, pp. 16 497–16 511, 2022.

[46] M. Xue, Z. Wu, C. He, J. Wang, and W. Liu, "Active DNN IP protection: A novel user fingerprint management and DNN authorization control technique," in *19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2020*, 2020, pp. 975–982.

[47] M. Xue, Z. Wu, Y. Zhang, J. Wang, and W. Liu, "Advparams: An active dnn intellectual property protection technique via adversarial perturbation based parameter encryption," *IEEE Transactions on Emerging Topics in Computing*, vol. 11, no. 3, pp. 664–678, 2023.

[48] M. Xue, L. Y. Zhang, Y. Zhang, and W. Liu, "Turn passive to active: A survey on active intellectual property protection of deep learning models," *CoRR*, vol. abs/2310.09822, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2310.09822

[49] M. Xue, Y. Zhang, J. Wang, and W. Liu, "Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 908–923, 2022.

[50] Z. Yan, G. Li, Y. Tian, J. Wu, S. Li, M. Chen, and H. V. Poor, "Dehib: Deep hidden backdoor attack on semi-supervised learning via adversarial perturbation," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, Virtual Event, February 2-9, 2021*. AAAI Press, 2021, pp. 10 585–10 593.

[51] Z. Yan, S. Li, R. Zhao, Y. Tian, and Y. Zhao, "DHBE: data-free holistic backdoor erasing in deep neural networks via restricted adversarial distillation," in *ACM Asia Conference on Computer and Communications Security, ASIACCS 2023*, 2023, pp. 731–745.

[52] Z. Yan, J. Wu, G. Li, S. Li, and M. Guizani, "Deep neural backdoor in semi-supervised learning: Threats and countermeasures," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 4827–4842, 2021.

[53] Z. Yang, H. Dang, and E. Chang, "Effectiveness of distillation attack and countermeasure on neural network watermarking," *CoRR*, vol. abs/1906.06046, 2019. [Online]. Available: http://arxiv.org/abs/1906.06046

[54] Z. Yang, J. Zhang, E.-C. Chang, and Z. Liang, "Neural network inversion in adversarial setting via background knowledge alignment," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS 2019*, 2019, p. 225–240.

[55] H. Yu, K. Yang, T. Zhang, Y. Tsai, T. Ho, and Y. Jin, "Cloudleak: Large-scale deep learning models stealing through adversarial examples," in *27th Annual Network and Distributed System Security Symposium, NDSS 2020*, 2020.

[56] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. M. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *ACM Asia Conference on Computer and Communications Security, ASIACCS 2018, Incheon*, 2018, pp. 159–172.

[57] J. Zhang, Z. Wang, Z. Yang, and Q. Zhang, "Proximity based iot device authentication," in *IEEE Conference on Computer Communications, INFOCOM 2017*, 2017, pp. 1–9.

[58] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, 2018, pp. 586–595.

[59] Z. Zhang, Y. Chen, and D. Wagner, "Seat: Similarity encoder by adversarial training for detecting model extraction attack queries," in *AISec@CCS 2021: Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security, Virtual Event, Republic of Korea, 15 November 2021*, 2021, pp. 37–48.

[60] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *15th European Conference on Computer Vision, ECCV 2018*, 2018, pp. 682–697.

[61] R. Zhu, X. Zhang, M. Shi, and Z. Tang, "Secure neural network watermarking protocol against forging attack," *EURASIP Journal on Image and Video Processing*, 2020.