AUTHNET: Neural Network with Integrated Authentication Logic

Yuling Cai[a,b], Fan Xiang[a,b], Guozhu Meng[a,b,*], Yinzhi Cao[c] and Kai Chen[a,b]

[a]Institute of Information Engineering, Chinese Academy of Sciences
[b]School of Cyberspace Security, University of Chinese Academy of Sciences
[c]Johns Hopkins University

Abstract. Model stealing, i.e., unauthorized access and exfiltration of deep learning models, has emerged as a significant security threat. The misuse and illegal replication of models pose major risks to financial assets and competitive advantage. Traditional protection methods, such as model watermarking, are passive and challenging to enforce, while active defenses often face limitations in terms of efficiency and the security required for widespread deployment.

To this end, we propose a native authentication mechanism, called AUTHNET, which integrates authentication logic as part of the model without any additional structures. Our key insight is to reuse redundant neurons with low activation and embed authentication bits in an intermediate layer, called a gate layer. Then, AUTHNET fine-tunes the layers after the gate layer to embed authentication logic so that only inputs with secret key can trigger the correct logic of AUTHNET. It provides the last line of defense, i.e., even being exfiltrated, the model is not usable as the adversary cannot generate valid inputs without the key. We theoretically demonstrate the high sensitivity of AUTHNET to the secret key, which means that precise key provision is essential for achieving good performance of AUTHNET. AUTHNET is compatible with any convolutional neural network, where our extensive evaluations show that AUTHNET successfully achieves the goal in rejecting unauthenticated users (whose average accuracy drops to 22.03%) with a trivial accuracy decrease (1.18% on average) for legitimate users, and is robust against adaptive attacks, providing efficient and lightweight protection.

1 Introduction

Deep learning has gained tremendous success in image and video processing [23, 45, 15], face detection [46, 24, 32, 33], speech recognition [51, 7, 1, 22], and so on [31, 40, 37, 54]. To be accessible for end users, models are deployed into on-premises devices (e.g., mobile phones, smart home, electrical equipment). On-device models are usually required for privacy preservation, low latency and unstable network connectivity.

However, deep learning models are under the significant threats after deployment due to model theft and misuse [29, 20, 16, 36, 14, 21, 35] that could cause devastating consequences including monetary loss and privacy leakage for model owners [43, 28]. The models deployed on devices can be leaked to attackers due to insufficient protection. As discussed in [34, 8], traditional methods such as authentication, obfuscation and encryption have already been employed to protect offline models. Still, attackers can leverage system breaches and memory dumping to compromise the protection.

Prior studies have explored a line of solutions to mitigate the threat [43, 28], which can be categorized into two classes. One class of solutions is *system-level protection* which leverage security features in the deployment environment to limit illegal access to models. For example, authenticating users for model access via credentials, obfuscating and encrypting models to protect model weights. The works [52, 35, 53] propose methods of placing parts of the neural network in a Trusted Execution Environment (TEE), achieving secure inference on both CPU and GPU. However, the above methods may incur significant computation costs [14, 21, 35]. The other class is *algorithmic protection* that may reshape model training and inference. For example, digital watermarking technology is employed to verify model ownership and thereby protect intellectual property. However, this method is passive and cannot prevent the occurrence of piracy [11, 12, 44]. Additionally, the passport-based method [11, 12, 44] protects models from unauthorized use by embedding validation logic within the model, requiring legitimate users to provide a correct passport for optimal inference performance. However, this approach incurs inevitable inference overhead and is susceptible to fine-tuning attacks [6]. This necessitates an active protection scheme that offers reliable security with high efficiency and limited hardware dependence.

In this paper, we propose a native authentication mechanism, termed as AUTHNET for deep learning models, which embeds authentication logic as part of the model without any additional structures. The key insight is to reuse redundant neurons with low activation as authentication logic, called *authentication bits*, during fine-tuning. Notably, by leveraging the low-activation neurons, we embed authentication bits non-intrusively, preserving the model's performance while enabling new functionality. More specifically, AUTHNET splits a target model into two parts—a head model and a tail model, where the gate layer, which is the final layer of head model, is encoded with authentication bits. Then, AUTHNET fine-tunes the tail model to embed authentication logic so that only inputs with a secret key, trigger authentication bits and then activate the authentication logic. As such, the built-in authentication logic enables models to distinguish legitimate and illegal inputs by itself, i.e., it demonstrates good classification performance on legitimate inputs, but conversely, yields inferior results on illegal inputs.

We perform a theoretical analysis of AUTHNET to certify its security against the adversary (Section 4). Specifically, we define the *authentication domain* for AUTHNET data as the range of areas con-

* Corresponding Author. Email: mengguozhu@iie.ac.cn.

sidered legitimate. By applying the neural network robustness measurement method [48], we demonstrate that AUTHNET narrows the reliable data domain compared to the normal network and shows high sensitivity to the secret key, meaning its performance can only be unlocked by providing the key precisely.

We conduct extensive experiments to evaluate the effectiveness, security and efficiency of AUTHNET. The experiments involve six popular learning models—LeNet, Mobile-Net, AlexNet, VGG13, ResNet18 and ResNet50. Our evaluation shows that AUTHNET successfully blocks unauthenticated model users by lowering their testing accuracy to 22.03% on average. At the same time, the testing accuracy for authorized model users stays mostly the same with a small decrease (1.18%) on average compared to legacy models without authentication (Section 5.2). We also show that AUTHNET resists a variety of adaptive attacks (Section 5.4) and shows a high inference efficiency (Section 5.5).

Contributions. We summarize the contributions as follows.

- We propose AUTHNET, a native authentication mechanism. It integrates authentication as part of the learning model, which is thereby more imperceptible and adaptive to multiple scenarios.
- We theoretically demonstrate the high sensitivity of AUTHNET to the authentication key, showing that it effectively compresses the recognition space for legitimate data, requiring a highly precise key to unlock AUTHNET's performance.
- AUTHNET stands out for its security, efficiency, and lightweight protection. It can resist multiple adaptive attacks while maintaining high inference efficiency.

The implementation of AuthNet is publicly available at https://github.com/software-and-ai-security-lab/AuthNet.

2 Threat Model

To better assess the effectiveness and security of our approach, we detail the adversary's goal, abilities and limitations.

Adversary's goal and abilities. The ultimate goal of the adversary is to steal an usable model for private deployment or commercial use. Even more, we assume that it has obtained model files (e.g., *.tf, *.pb) via exploiting system breaches. The adversary can inspect model parameters, running status, and computation results. It even enables the adversary to fine-tuning the model with some data. This is not only a tremendous monetary loss for model owner, but the adversary can also launch subsequent attacks such as inferring privacy in training data.

Our solution. Apparently, model watermarking cannot prevent such attacks and is only useful to determine whether one suspicious model are plagiarized or not. Our target is that *despite having been already stolen, the model cannot be misused and distributed.* Therefore, we propose AUTHNET that equips deep learning models with authentication capability. If the model is fed with a legitimate input (i.e., enclosed with secret key), it will produce correct results. Otherwise, it will return randomly-guessed results. The adversary may be aware of such protection scheme, so it can conduct fine-tuning attacks and substitute model training with a number of legitimate inputs as well as the corresponding results. However, the secret key is supposed to be completely isolated and unknown from the adversary. These approaches, such as using confidential containers (e.g., Trusted Execution Environments, TEE) or separating key embedding from model inference, are feasible solutions. However, exploring these methods is beyond the scope of this study.

3 The AUTHNET Approach

As shown in Figure 1, we introduce AUTHNET including the design of authentication logic, and each step with its corresponding role during authentication.

3.1 Design of Authentication Logic

The authentication capability of AUTHNET is built on the potentials of neural networks abstracting features and making classification. The large-scale neurons enable a large information capacity during the flow of data through neural networks, allowing us to embed non-functional information, e.g., *authentication*, into the flow. To achieve the goal of authentication, AUTHNET splits a neural network into two parts: *head model* and *tail model*. The head model is responsible for extracting secret key embedded in images and expects certain conditions to be achieved at the gate layer. The *gate layer*, which is also the final layer of head model, conveys authentication information which is passed to the tail model for recognizing and classification. Note that the actual model structure is not changed and no new layers are introduced. We simply renamed the network architecture for ease of exposition.

Let f be a neural network, and h, g and t be the head model, gate layer and tail model, respectively. A neural network f can be decomposed into $f = h \circ t$. Without loss of generality, we define the gate layer g as:

Definition 1. *The neurons in gate layer are denoted as* $G = \{n_1, n_2, ..., n_C\}$*, where* C *is the total number of neurons in this layer.*

The gate layer is the valve for authentication, and its location is generally not disclosed for security. Since the gate layer conveys not only the authentication information, but also functional data, not all neurons are chosen for authentication. To differentiate the functions, we term the special neurons as *authentication bits* that pave a channel to allow authentication information to pass.

Definition 2. *The authentication bits,* $Ab = \{n_b^1, n_b^2, ..., n_b^l\} \subset G$*, is the set of neurons responsible for authentication function with a length of* l*.* $\neg Ab = \complement_G Ab$ *represents the non-authentication bits, where* $\complement$ *is the complement symbol.*

Authentication bits are the neurons responsible for identity authentication. To maintain the performance of the model on the original classification task, we do not use all neurons as authentication bits. 5% of the neurons in the gate layer are sufficient to perform the identity authentication function in synchronization with classification.

Assuming f' is the enhanced network for f by AUTHNET, we define AUTHNET's property as follows: Given an input $\langle x,\ y \rangle$ and secrete key k, the results of f' are distinguishable between the input with key and one without key, i.e.,

$$\| \Pr(f'(x \oplus k) = y) - \Pr(f'(x) = y) \| \geq \epsilon$$

Where $x \oplus k$ denotes that the input is embedded with authentication key, but not for x. Secret key k is an image with the same shape as the input image x. Here, ϵ measures the distinguish ability of AUTHNET. When ϵ is larger, the model f' performs better to distinguish illegal input from the legitimate. The bounds of ϵ are $[0, \Pr(f'(x \oplus k) = y)]$. It is ideal to have $\Pr(f'(x \oplus k) = y) \approx \Pr(f(x) = y)$ and $\Pr(f'(x) = y) \approx \frac{1}{|y|}$, i.e., the model randomly selects categories

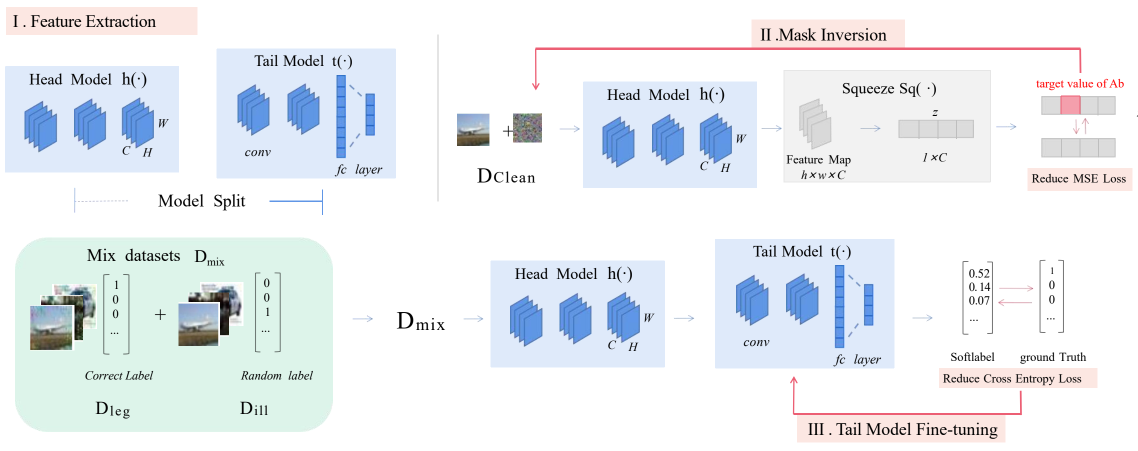

Figure 1. llustration of embedding authentication logic into DNN with AUTHNET.

like tossing a coin. However, in reality, if the correct probability of illegal input $\Pr(f'(x)=y)$ is small enough, although larger than the probability of random guessing, the model is not usable and thereby secure.

3.2 Feature Extraction and Model Splitting

In this step, we describe how to split a pre-trained model. The calculation process of pre-trained model in classification tasks is a function $f_\theta(\cdot) : \chi \rightarrow Y$. The model is trained on dataset $D_{clean} = \{(x_i, y_i) \mid i \in [1, N]\}$, where x_i is an original image and y_i is the corresponding label with shape $1 \times K$, where K refers to the number of categories of images. After receiving an image input x_i, the model deduces and gives a soft label $\tilde{y}_i$. Then we apply cross entropy loss function to compute the differences between soft label inferred by model $\tilde{y}_i$ and ground truth y_i. Then we train the parameter θ of deep learning model $f_\theta(\cdot)$ by solving the optimization problem by using the optimization strategy of gradient propagation and gradient descent algorithm:

$$\arg\min_\theta L_{utility} = \arg\min_\theta -\sum_{i=1}^{N}\sum_{j}^{K} y_i^j \log \tilde{y}_i^j \tag{1}$$

where $\tilde{y}_i = f_\theta(x_i) = (\tilde{y}_i^1, \tilde{y}_i^2, ..., \tilde{y}_i^K)$. After obtaining a trained feature extractor $f(\cdot)$, we split the model into the head and the tail model. Generally, the splitting point controls the balance of identity extraction of head model and identity recognition by tail model. If there are too few layers in tail model, the identity information would be not well recognized.

The proposed scheme does not require strict segmentation location during model splitting, unless tail network only contains full connected layers with a single convolution layer.

3.3 Mask Inversion

In this section, we describe how to generate the secret as identity information with the head model $h(\cdot)$.

Definition 3. *A secret key is composed of two elements, i.e., $key = \{mask, offset\}$, where mask is a position weight for the target image and $offset$ is a constant, with which the preprocessed image is obtained by:*

$$img' = mask \times img + offset \tag{2}$$

Generally, we expect a distinction between the intermediate activation at the *gate layer* of the input with the *key* and the activation of the unmodified input, which is the primary optimization objective of the *key*. Subsequently, we train the tail model $t(\cdot)$ to be sensitive to the distinction. For a single neuron in G, it represents a single channel of the feature map of gate layer. Take VGG13 with the structure shown in Table 2 as an example. If we choose $seq.9$ as the gate layer, the shape of feature map of a single neuron in this layer would be $256 \times 1 \times 4 \times 4$ (the first dimension is the batch size set to 256). To evaluate how strongly a neuron is activated, we compress the feature map values with the squeeze function $Sq(\cdot)$.

$$\begin{gathered}\mathbf{z} = (z_1, z_2, ..., z_C) = Sq(h(x)), \\ z_p = \frac{1}{N}\sum_{i}^{N}\frac{1}{H\times W}\sum_{h}^{H}\sum_{w}^{W} z_{i,p,h,w}\end{gathered} \tag{3}$$

where $h(x) \in \mathbb{R}^{N\times C\times H\times W}$ is the feature map at the gate layer, and $\mathbf{z} \in \mathbb{R}^C$ is the squeezed activation value, $p = 1, 2, ..., C$. Note that the compression operation is only performed when inverting the mask and selecting the authentication bits. The computational flow of AUTHNET in inference is no different compared to the original neural network. In our experiments, we select the neurons with lowest l squeezed activation value as the *authentication bits* $Ab = \{n_b^1, n_b^2, ..., n_b^l\}$.

To enhance the identification of legitimate users, we try to maximize the difference between intermediate activation of different users in this step. Therefore, We introduce the concept of *discrimination degree* γ to characterize the vector distance between intermediate outputs as below,

$$\gamma = \frac{||Sq(h(x \oplus key))|Ab - Sq(h(x))|Ab||}{||Sq(h(x))||} \tag{4}$$

where $Sq(h(\cdot))|Ab$ is a subset of the squeezed intermediate activation on the *authentication bits*. We introduce the L norm of the original activation vector $Sq(h(x))$ as the denominator in the definition formula, so that γ will be a relative value and more adaptive. Without loss of generality, we change the problem of maximizing γ to updating the *key* such that γ approximates a larger $\tilde{\gamma}$, so that the difference is large enough for the tail model $t(\cdot)$ to authenticate. To sum up, we have our first training constraint as follows.

$$Sq(h(x \oplus key))|Ab = Sq(h(x))|Ab + \tilde{\gamma} \cdot ||Sq(h(x))|| \tag{5}$$

The second criterion to the *key* taken into consideration is the maintenance of the performance on classification tasks. That is, we aim to ensure the $h(\cdot)$ to extract useful texture features for the processed images $\tilde{x}$ without fine-tuning the head model. Since the pre-trained model performs well on unmodified images, we elicit the following constraint.

$$Sq(h(x \oplus key))|\neg Ab = Sq(h(x))|\neg Ab \tag{6}$$

where $\neg Ab = \complement_G Ab$ is the set of neurons responsible for classification tasks. In the inverse process, we combine the above constraints to generate $key = \{mask,\ offset\}$. The optimization function is formulated as below.

$$\begin{aligned} key &= \underset{key}{\arg\min}\ Loss(x, key, \Gamma) \\ &= \underset{key}{\arg\min}\ L_{MSE}(Sq(h(x \oplus key)) - Sq(h(x)), \Gamma), \\ s.t. mask &\in (0, \epsilon_m)^{H \times W}, offset \in (-\epsilon_U, \epsilon_U)^{H \times W \times C} \end{aligned} \tag{7}$$

where $\Gamma \in \mathbb{R}^C$ is the target vector of *discrimination degree*:

$$\Gamma_i = \begin{cases} \tilde{\gamma} \cdot ||Sq(h(x))||_\infty & n_i \in Ab \\ 0 & n_i \in \neg Ab \end{cases} \tag{8}$$

Here C is the total number of the neurons in gate layer G, the same size as channel number of feature map. $\tilde{\gamma}$ is the target large discrimination degree. i indicates one-to-one positional relationship between the neuron n_i and element Γ_i of target vector. Considering the implicit of the key, we introduce some constraints to the range of *key* with the hyperparameters, ϵ_m and ϵ_U. The Adam optimizer and 2000 images selected evenly and randomly from the D_{clean} are used to optimize key in an epoch, and more specific experimental parameter settings are displayed in Section 5.1.

3.4 Tail Model Fine-tuning

In the first two step, we generate the key by using some training constraints to guarantee the extraction of identity information with head model $h(\cdot)$. While $h(\cdot)$ is for the extraction of identity pattern, the tail model is fine-tuned to recognize the identities, that is, the fine-tuned tail model $t(\cdot)$ performs well with legitimate inputs and gives approximately random outputs to illegal queries.

More specifically, we design two training datasets to embed authentication logic into the tail model $t(\cdot)$. The $key = \{mask,\ offset\}$ reversed in the previous step brings large intermediate activation on *authentication bits* to the feature map, and the tail model are fine-tuned to detect this increment. we first describe dataset D_{mix} for fine-tuning tail model $t(\cdot)$, and then introduce our fine-tuning loss function in the following.

D_{mix} is the mixture of illegal dataset D_{ill} and legitimate dataset D_{leg}. Illegal dataset is denoted by $D_{ill} = \{(x_i, \breve{y}_i)|i = 1, 2, ..., N\}$, which consists of original image x_i in D_{clean} and random labels $\breve{y}_i$ that obey discrete uniform distribution $U(1, K)$ for a k-classification task. Legitimate dataset is donated by $D_{leg} = \{(x_i \oplus key, y_i)|i = 1, 2, ..., N\}$, which consists of images embedded with *key* and their ground-truth labels.

On one hand, the illegal dataset D_{ill} is built to help AuthNet learn the expected behavior with illegal queries. We give data random labels to mislead the performance of AuthNet on nude images. On the other hand, we import the D_{leg} to guarantee the performance of the model when it is used legally. Then we use D_{mix} to fine-tune the parameters of tail model $t(\cdot)$. Therefore, the trained tail model has the functions of both image classification and identity authentication, and at the same time obfuscating the outputs of illegal queries.

4 Theoretical Analysis

It is noteworthy that the security of AuthNet is closely associated with the confidentiality of the $secret\ key = \{mask,\ offset\}$. Therefore, in this section, we attempted to analyse the difficulty of cracking AuthNet's secret key theoretically using neural network robustness analysis [48]. Our primary focus was on analyzing AuthNet's sensitivity to the secret key, which encompass the need for an exceptionally precise secret key to attain optimal network inference performance. A smaller authentication domain suggests higher sensitivity to the secret key.

Considering a neural network with m layers $f(\cdot) : x \to y$, where input is $x \in \mathbb{R}^n$ and output is $y \in \mathbb{R}^k$. Unauthorized input is denoted as x_0, while the corresponding authorized input is $\overline{x_0} = x_0 \oplus key$. The $\epsilon-neighborhood$ of x_0 is denoted as $Z_\epsilon(x_0) = \{x| ||x - x_0||_p \leq \epsilon\}$. In the theoretical proof, we choose the infinity norm, where $p = \infty$, used to calculate the maximum absolute value of vector $x - x_0$. The maximum authentication domain range is denoted as ϵ_m, which is the optimal solution for the following optimization problem:

$$\epsilon_m = \underset{\epsilon}{\arg\max}\{f(x) = y_c\}, \forall x \in Z_\epsilon(\overline{x_0}) \tag{9}$$

where y_c is the ground truth label of input x_0. According to [48], for a k-class classification network, as the input x_0 varies within the $\epsilon - neighborhood\ Z_\epsilon(x_0)$, we can obtain the fluctuation range of output $[L_i(\epsilon), U_i(\epsilon)]$ $(0 \leq i < k)$. The optimization problem in (9) is converted into the following problem:

$$\epsilon_m = \underset{\epsilon}{\arg\max}\{L_{y_c}(\epsilon) \geq U_i(\epsilon)\}, 0 \leq i < k \& i \neq y_c \tag{10}$$

Notice that the lower bound $L_{y_c}(\epsilon)$ should satisfy the condition, i.e., being larger than the upper bound of other classes. Our objective is to find the maximum ϵ that results in a *gap* of zero:

$$gap = \left| L_{y_c}(\epsilon) - \max_{0 \leq i < k,\ i \neq y_c} U_i(\epsilon) \right| \tag{11}$$

We conducted tests on the neural network AuthNet-LeNet using 100 images from the MNIST dataset to determine the authentication region range, which signified AuthNet's sensitivity to the secret key. AuthNet-LeNet exhibits a robustness of 0.00492 towards authentication domain samples, compared to the robustness of 0.02554 in a standard LeNet model trained on MNIST without adversarial training. It demonstrates nearly a tenfold increase in sensitivity of AuthNet towards secret key in the authentication domain. This experiments highlights the high sensitivity and security of AuthNet to the secret key, emphasizing that only with a highly accurate secret key can AuthNet be used effectively.

5 Evaluation

We first introduce the details of experiment setup, including implementation, experimental models and datasets. Then, we conduct extensive experiments to answer the following research questions.

RQ1. How effective is AuthNet in protecting DL models, i.e., the accuracy for authenticated and unauthenticated inputs? (Section 5.2)

RQ2. Is AuthNet robust enough in the face of multiple model transformations? (Section 5.3)

RQ3. How easy to infer the authentication key or destroy the authentication logic with adaptive attacks? (Section 5.4)

RQ4. What are the inference costs of AuthNet compared with prior studies? (Section 5.5)

5.1 Experiment Setup

We implement AUTHNET with 2.5K lines of Python with PyTorch. Experiments are run on a server of 64-bit Ubuntu 18.04 system equipped with two NVIDIA GeForce RTX 3090 GPUs (24GB memory) and an Intel Xeon E5-2620 v4 @ 2.10GHz CPU, 128GB memory.

Target Model. We implement the AUTHNET on several typical Convolutional Neural Networks, including LeNet. AlexNet, VGG13, MobileNet-v3 small and ResNet, which are all commonly used in the field of computer vision. Five of them are trained on CIFAR10 or CIFAR100 used as classification tasks, while the other one is trained on MNIST as a handwritten digit recognition task. The specific hyper-parameters used in Section 5.2 are showed in Table 1.

Metrics. We illustrate the metrics used for evaluating AUTHNET.

- ACC_{leg} is the top-1 accuracy of the legitimate user with *secret key*, which is expected to be on par with the accuracy of pre-trained model.
- ACC_{ill} is the top-1 accuracy of illegal queries with unmodified images, which should be significantly lower than the queries from authorized users and the lower bound is the probability of random guessing.
- CC, short for computation cost, is the additional inference time beyond the pre-trained model. The evaluation is repeated 10 times, and the reported result is the average cost.

Table 1. Hyper-parameters used in experiments. "#Auth-bits" represents the number of authentication bits in the mask inversion step. γ is the target discrimination degree in mask inversion. ϵ_M and ϵ_U limit the maximum value of mask and offset. lr_m and lr_U refer to the step size of optimizing mask and offset.

Hyper-parameters		**AlexNet**	**VGG13**	**ResNet18**
Mask Inversion	#Auth-bits	20/384	10/256	10/256
	γ	3×	2×	3×
	Seq_{seg}	3/5	9/17	6/9
	ϵ_m, ϵ_U	0.5	0.5	0.5
	lr_m	0.01	0.01	0.01
	lr_U	0.003	0.003	0.003
Fine-tuning Tail model	lr	0.01	0.01	0.01
	epochs	200	50	50
	batch size	256	256	256
	Activation	ReLU	ReLU	ReLU

Table 2. The sequence of layers in VGG13 (dropout $p = 0.5$, batch size=256)

Layer Type	**Conv**	**AVG Pool**	**F.C.**
Seq.	1, 2, 4, 5, 7, 8, 10, 11, 13, 14	3, 6, 9, 12, 15	16, 17

5.2 Effectiveness of AUTHNET

To evaluate the effectiveness of AUTHNET in protecting models, we apply the authentication scheme on six deep learning models–LeNet, AlexNet, VGG13, MobileNet-v3, ResNet18 and ResNet50. As shown in Table 3, six models are trained on MNIST (LeNet), CIFAR10 (VGG13, AlexNet, MobileNet-v3 and ResNet18) and CIFAR100 (ResNet50) as the pre-trained models with the accuracies of 98.66%, 87.87%, 88.02%, 86.69%, 92.00%, and 62.52%, respectively.

We take VGG13 as the example to explain the experiment process in detail as below. First, we train a model as the pre-trained one on CIFAR10 for 50 epochs with Adam optimizer and a learning rate of 10^{-3}. The accuracy of this model can achieve 87.87% on the test dataset of CIFAR10.

Step1: we split this pre-trained model into head model $h(\cdot)$ and tail model $t(\cdot)$, after the Seq_{seg} listed in Table 1. The *gate layer* is the last layer of $h(\cdot)$, and the smallest neurons are selected from this layer as the *authentication bits*.

Step2: we randomly select 200 samples for each category from the training dataset for mask inversion. The *mask* and *offset* are reversed with the loss function (7). The learning rates and ϵ of this process are listed in Table 1.

Step3: we fine-tune the tail model on the D_{mix} introduced in Section 3.4 for 50 epochs with Adam optimizer and a learning rate of 10^{-2} multiplied by 0.1 every 10 epochs.

Compared with pre-trained models, the ACC_{leg} of AUTHNET drops by 1.18% on average, while the ACC_{ill} of AUTHNET is as low as 22.03% on average. The extra computation cost is lower than 5% for different models and mostly about 1%. The extra time cost of AUTHNET is mainly in the preprocessing of the images.

In conclusion, the models enhanced by AUTHNET have retained a qualified performance, with 1.18% accuracy drop for legitimate users and at most 34.84% accuracy for illegal users. Additionally, the brought computation cost during inference is negligible.

Table 3. Effectiveness of AUTHNET on six models. $ACC_{baseline}$ is the accuracy of corresponding model.

Models	$ACC_{baseline}$	ACC_{leg}	ACC_{ill}	CC
LeNet	98.66%	98.38%	15.62%	3.71%
AlexNet	88.02%	85.58%	34.84%	0.90%
VGG13	87.87%	86.95%	23.03%	1.44%
MobileNet-v3	86.69%	88.11%	26.02%	4.50%
ResNet18	92.00%	90.63%	21.70%	1.32%
ResNet50	76.74%	74.54%	13.98%	0.51%

Table 4. Performance comparison with *Passport*.

Models		ACC_{leg}	ACC_{ill}	CC
AlexNet (88.02%)	AUTHNET	85.58%	34.84%	**0.90%**
	Passport	**89.22%**	-	11.52%
ResNet18 (91.41%)	AUTHNET	**90.63%**	21.70%	**1.32%**
	Passport	89.67%	-	17.65%

Comparison with *Passport* [11]. Passport-based method is an another authentication scheme which is used to protect models from unauthorized users. We compare our scheme with it on legitimate performance, computation cost and model property. The results show that they both perform well in authentication but AUTHNET outperforms in imperceptibility and computation cost. The passport method adds several passport layers into the model, and the parameters in this carefully designed layers are calculated with the passport provided by the user in each query. We conduct the comparison experiments with AlexNet and ResNet18 on CIFAR10 (since these are the only two model structures discussed in [11]), and they have the accuracies of 88.02% and 91.41%, respectively. As shown in Table 4, models with these two methods all keep high accuracies for legitimate users. The ACC_{leg} drops by 0.27% and 1.61% on average for the *passport* method and AUTHNET. Method *Passport* is more perceptible than AUTHNET, and the reasons are (1) unlike AUTHNET embedding the authentication logic into the weight of models, Passport makes special changes to the network structure and relies entirely on these special layers (passport layers) for authentication; (2) a passport is strictly required in this method, or the model would not give any answers, as indicated at column "ACC_{ill}" in Table 4. On the contrary, the average ACC_{ill} of AUTHNET is 28.27%, which is far away from usable. Additionally, AUTHNET is more efficient after deployed. The results in Table 4 show that, AUTHNET saves about 10% computation cost compared with *Passport*.

5.3 Robustness Analysis

It is commonly known that neural networks are susceptible of model transformations [39, 3]. Here we aim to evaluate the robustness of AUTHNET against fine-tuning and model pruning.

5.3.1 Model Fine-tuning

Attackers may fine-tune models with a small amount of data to destroy the authentication scheme of AUTHNET. In this section, we take AUTHNET-VGG13 and AUTHNET-ResNet18 trained on CIFAR10 as the baseline models, and then fine-tune the models with new datasets including MNIST, CIFAR100 and GTSRB. In each experiment, we fine-tune all the parameters with the Adam optimizer for 50 epochs, and the learning rate is 10^{-4}.

The accuracy of the fine-tuned AUTHNET-VGG13 on STL10, GTSRB, and CIFAR100 are 71.14%, 96.47%, and 59.31%, and the corresponding accuracy of the fine-tuned AUTHNET-ResNet on this three dataset are 79.85%, 98.04%, and 64.78% respectively, which means that our fine-tuning is effective. The results are shown in Table 5, where the ACC_{leg} and ACC_{ill} are the accuracies on CIFAR10 of the models after fine-tuned.

Taking AUTHNET-VGG13 as an example, even when the network is fine-tuned with STL10, the ACC_{ill} can only reach 42.57%, and the results of fine-tuning on other datasets are even worse (a minimum of 21.99%). This is a failed attack for the attackers, as the network would not leak the correct prediction without mask after fine-tuning process. After fine-tuning for AuthNet-resnet, the model still exhibits a noticeable accuracy difference on legitimate and illegal inputs (6.53%-30.75%), which suggests that the authentication mechanism has been retained to a certain extent. For AuthNet-VGG13 on gtsrb and CIFAR100 datasets, the gap between ACC_{leg} and ACC_{ill} is small. Therefore, we recommend that for model owners who are allowed to fine-tune or transfer learning the model, they can first fit the model parameters to new task through transfer learning and then embed the authentication logic with our scheme. Therefore, We demonstrate the good robustness of our proposed scheme against the fine-tuning process.

Table 5. Robustness of AUTHNET after model fine-tuning. "Datasets" indicates the data used for fine-tuning, and "-" is the performance of AUTHNET before fine-tuning.

Model	Datasets	ACC_{leg}	ACC_{ill}
VGG13	-	86.95%	23.03%
	STL10	60.87%	**42.57%**
	GTSRB	22.52%	<u>21.99%</u>
	CIFAR100	36.17%	38.19%
ResNet	-	90.63%	21.70%
	STL10	79.49%	**48.74%**
	GTSRB	41.12%	<u>22.25%</u>
	CIFAR100	51.49%	44.96%

5.3.2 Model Pruning

Pruning is a common practice in model optimization to compress a neural network by eliminating redundant neurons under the premise of maintaining the accuracy of the original inference task. In general, neurons with lower weights or less-connected tend to be the focus of attention during pruning, since they are less capable of conveying information to deeper networks. We adapt the commonly used prune technique in [13] on four different pre-trained models and their corresponding AUTHNET in Section 5.2, and the results are shown in Figure 2.

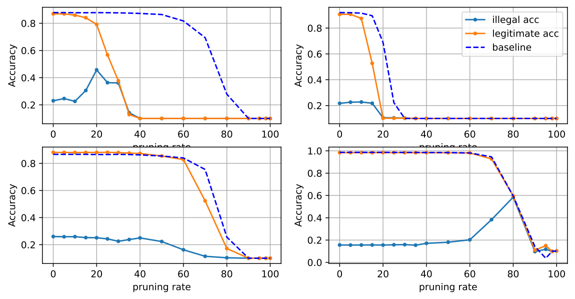

Figure 2. Pruning attack on AUTHNET, from left to right and top to bottom, the sub-images represent AUTHNET-VGG13, AUTHNET-ResNet18, AUTHNET-MobileNet, and AUTHNET-LeNet, respectively.

In our experiments, the pruning rate varies from 0% to 100%. When the pruning rate is below 40%, our interval is set to 5%, and after exceeding 40%, the pruning rate interval of each set of experiments is 10%. This is because the performance of AUTHNET changes significantly when the pruning rate is lower than 40%, and we are more concerned about whether the model will leak the correct prediction results to illegal queries during the pruning process of AUTHNET. The dashed line in the figure shows the change trend of the classification accuracy of the pre-trained model as the pruning rate keeps increasing. For the four pre-trained models, the accuracy shows a significant decline once the pruning rate exceeds 60%. This means that the pre-trained model does contain a large number of redundant neurons, in the other words, our theoretical assumption about the redundant neurons in networks is reasonable. AUTHNET generally starts to experience a significant decline in accuracy for legitimate users after the pruning rate exceeds 20% or 30%. This directly means that the proportion of redundant neurons in the model is decreasing, and our scheme does effectively reuse the redundant neurons in the model, which guarantees the authentication function of the model. Additionally, the accuracy of illegal queries always stays below 50% throughout the pruning process, which means that our model also does not leak the correct predicted label during pruning. In summary, our model is robust and secure for the case of pruning.

5.4 Security Analysis

We evaluate AUTHNET's security from two aspects, i.e., the key forgery attacks and the resistance against authentication offsetting.

5.4.1 Mask Inversion Attack

In this scenario, attackers possess a number of original images and their labels. Then they search for an optimal mask, with which the inputs can be correctly classified by the AUTHNET-enhanced model. Besides, we assume that the attacker has access to AUTHNET, so it can use reverse engineering to optimize a fake key $Mask_I$, where a mask and an offset images are trained with AUTHNET parameters frozen. The training objective is to improve the AUTHNET accuracy on images embedded with $Mask_I$. We conduct experiments on AUTHNET-LeNet, AUTHNET-ResNet18, AUTHNET-VGG13 and AUTHNET-ResNet50, using MNIST, CIFAR10, CIFAR10 and CIFAR100, respectively. Additionally, we run 10 epochs for each attack and select the best accuracy.

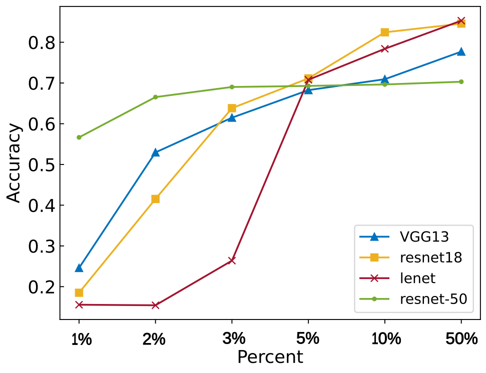

Figure 3. The performance of $Mask_I$ on AUTHNET

Figure 3 presents how the performance of $Mask_I$ varies with training set proportions. As the proportion of training data obtained by the attacker increases, the performance of $Mask_I$ becomes better. For AUTHNET-VGG13, using 10% of training data to reverse $Mask_I$, the attacker can gain 70.93% accuracy, for AUTHNET-ResNet, 5% of training data is needed to achieve 71.13% accuracy. However, there are still significant gaps to the original model, i.e., 16.94% and 20.87%, respectively. To ensure a 10%-accuracy distance, attackers have to get at least 50% training data, of which the cost is prohibitive.

In addition, we also find that reverse attacks seems to be more effective for AUTHNET-ResNet50, with large scale parameters, 66.54% accuracy was achieved with 2% training data. However, accuracy increases very slowly as the number of training sets increases, 70.31% accuracy is achieved using 50% training data, which means that achieving near-clean model performance requires a steep increase in training difficulty. This is related to the generalization problem of large-scale model, a universally applicable key is difficult to fit due to the complex nonlinear transformations of deep learning models.

5.4.2 *Authentication Offsetting*

Different from fine-tuning the whole model, attackers cannot change the parameters in this scenario, but are able to augment an extra linear layer behind AUTHNET.

Table 6. Extra layers are added to the models and then fine-tuned to offset the authentication logic in AUTHNET.

	AlexNet		ResNet18	
	AuthNet	Passport	AuthNet	Passport
Ori. Acc.	85.58%	89.22%	90.63%	89.67%
Accuracy	12.01%	13.32%	15.95%	59.48%

Referring to the setting of [6], through meticulous fine-tuning of this supplementary layer, they aim to alleviate the impact of the identity authentication mechanism embedded in the preceding network. In particular, the attacker first gains some clean data, adds a fully connected layer to the model, and fine-tunes this layer to correct the classification function, i.e., offsetting identity authentication.

We conduct experiments on ResNet18 and AlexNet, and also provide a comparison with the Passport method. In our adversarial setting, attackers acquire 20% of CIFAR10 samples and perform fine-tuning for the last additional linear layer. The original model's parameters are frozen throughout the entire fine-tuning process. After 10,000 rounds of fine-tuning with a learning rate of 0.001 for both ResNet18 experiments and 0.01 for both AlexNet experiments, the models exhibit the following performance. In Table 6, on AlexNet, both approaches exhibit high defensive effects. After fine-tuning, the accuracy rates for AuthNet-AlexNet and Passport-AlexNet are 12.01% and 13.32%, respectively. However, on the more substantial and complex model ResNet18, AuthNet's success rate in this attack is significantly lower than that of the Passport method (accuracies on unauthorized data: $11.3\% < 56\%$). It shows that, to a certain extent, AuthNet is more secure than the Passport method in a close-box attack scenario. It is largely because AUTHNET exhibits a stronger confusion effect on refused samples, indicating that our strategy of random output for refused samples is more effective.

5.5 *Cost Analysis*

To evaluate AUTHNET's practicality, we conduct experiments to measure the cost of model enhancement and inference.

Additionally, we compare training cost and inference cost of AUTHNET with the Passport method [11]. We test AUTHNET-ResNet18 on dataset CIFAR10, where the hyperparameters of mask inversion and tail network fine-tuning are in Table 1. We also conduct the Passport method in the V1 scenario as [11], since it is aligned with this study.

Table 7. Training and inference costs. "$T_t(M)$" denotes the time to train a clean model, "$T_t(M_{auth})$" is the time to train a model with authentication logic. AUTHNET's $T_t(M_{auth})$ contains mask inversion time and tail network fine-tuning time. "$T_i(M)$" and "$T_i(M_{auth})$" denotes the inference time for clean model and model with authentication logic. The inference time of Passport involves passport setting and inference time.

Time Cost	Passport-ResNet18	AUTHNET-ResNet18
$T_t(M_{Auth})$	2495.75	24.73 + 2067.78
$T_t(M)$	2168.42	1690.59
$Cost_t$	15.09%	23.77%
$T_i(M_{Auth})$	0.33+1.80	3.20
$T_i(M)$	1.87	3.16
$Cost_i$	17.65%	1.32%

The results are shown in Table 7, the Passport layers lead to 10%-15% extra running time compared to training a clean model, and about 10% extra inference time. Nevertheless, for our scheme, since no additional structures are added to the model, embedding logic and inference cost are largely reduced. The extra time cost of training process is 20%-25%, and it makes subtle extra cost for inference (only 1.32%).

6 Discussion

Generalizability. We introduce the AUTHNET method and experiments primarily based on computer vision tasks. The identity information can be easily embedded into images with minimal perturbations, and then AuthNet is trained to extract this information and recognize the identity. To adapt for other tasks like natural language processing, AuthNet should be equipped with a new method to encode identity information in tabular or textual input. Prior studies [50, 27, 4, 10] propose several methods of evolving words in text or text perturbation to construct adversarial examples or inject backdoors. Considering extending AuthNet to text classification or generative tasks, we can use such an atom-level perturbation for text as secret key embedding process, and select authentication bits on word vector, where the perturbation embedded in the original text would lead to high activation values, then fine-tune the tail network, which is responsible for verifying the validity of the input.

Enhancements for AuthNet. It has a high potential to be enhanced against more advanced attacks. For example, if one attacker employs the brute-force approach in a clear-box setting, identifying the layer containing authentication bits and fine-tuning one offset layer before it. The authentication bits can be dispersed into multiple layers and the tail network is after the last layer of authentication bits. We conduct a group of experiments in VGG13 on CIFAR10 dataset to distribute 150 authentication bits across 3 layers with γ=30. The initial accuracy for legitimate inputs is 83.16%, while for illegal inputs, it is 39.09%. Authentication offset attacks with fine-tuning 10%, 20% and 50% of training data can achieve 29.4%, 40.2% and 72.06% accuracy, respectively. Even worse, assuming the number of candidate layers is n, the complexity of such attacks is raised from $O(n)$ times of fine tuning to $O(2^n)$ if the adversary has no idea about the number of layers with authentication bits.

7 Related Work

Model Theft. Model theft poses a serious security threat to on-device deep learning models. Sun et al. [34] conduct a survey of 1468 applications in the app markets of both the United States and China and find that 41% of models lack any form of defence. Using simple file extension retrieval and dynamic analysis techniques, 66% model files could be directly stolen. Deng et al. [8] automatically extract 245 deep learning models from Android applications and present the first systematic study of real-world black-box adversarial attacks, reaching attack success rate of 47.35%. Huang and Chen [18] collect 53 real-world deep learning mobile applications from Google Play and introduce gray-box adversarial attack framework and find that 71.7% applications can be successfully attacked. Huang et al. [19] crack deep learning models by identifying highly similar pre-trained models in TensorFlow Hub and launching adversarial attacks on 10 real-world Android applications. Side-channel attacks [41, 47, 42, 2, 17, 9] is also a serious threat. These side-channel attacks are designed to either capture model structure [41, 47], or disclose training data privacy [42, 9], or explore the theft of both model parameter and structure [2, 17]. Wei et al. [41] attack the structure of deep learning models in the shared GPU scenario, where model structure and hyperparameters of VGG16 are recovered with 95% and 82.8% accuracy. Yu et al. [47] use electromagnetic side-channel attacks to steal structure and hyperparameter, then adopt black-box model extraction attack, achieving 96% performance of original model. Wei et al. [42] conduct experiment on a convolutional neural network accelerator based on FPGA, and the input image is recovered from the collected power trajectory without knowing the detailed parameters. Duddu et al. [9] uses timing channel to infer hyperparameter of convolutional neural network, which can be used in membership inference attack leading to the leakage of training data. Batina et al. [2] propose a differential power analysis on an ARM CORTEX-M3 microcontoller, allowing attackers to obtain parameter, hyperparameter and structure of model. Hua et al. [17] extract weight information by constructing samples to observe the output through memory and timing side-channel, which is robust to dynamic zero pruning model transformation and data encryption protection. *Our solution* AuthNet *can well mitigate the above attacks. Even if attackers adopt dynamic extraction or network monitoring to steal a model, the authentication logic inside the model can still stop attackers from using it.*

Model Protection. To protect intellectual property of deep learning models, there are currently passive and active defenses available in practice [5, 38, 11, 49]. Chen and Wu [5] firstly propose a scheme, binding model availability to authentication results of users. A mask was given to the legitimate user to ensure the model performance by adding the mask to each image. Tang et al. [38] embed 0-1 verification code patch on images in training process of student model in model distillation scenario. When distributing student model, only the correct captcha images can ensure that the model is trained normally. Fan et al. [11] distribute passport to legitimate users, the passport validation layer in the model will map the layer input linearly, and the model accuracy can only be guaranteed by using the correct passport. Zhang et al. [49] place a passport validation logic in the parameters of normalization layer, which does not change model structure when reasoning about the validation. Users who do not provide the correct passport will not be able to obtain high accuracy output of the model.

However, it is witnessed that these protection solutions are vulnerable to ambiguity attacks [6, 25, 30, 26]. Chen et al. [6] is an efficient ambiguity attack on passport-based ownership verification scheme [11], where they forge the passport by replacing the passport layer with a fully connected layer to training a new scale factor and bias factor to construct forgery passport. Li and Chang [25] propose a zero knowledge watermark detector, which allows model owner completes ownership verification without disclosing verification information to resist ambiguity attack. Sencar and Memon [30] propose a multiple watermark embedding scheme, which combines a series of selecting detection and unidirectional function in watermark embedding process, the verifier should provide not only secret information but also correct selection strategy in verification process. Loukhaoukha et al. [26] propose an ambiguity attack to generate a robust blind image watermark based on diffuse discrete wavelet transform and singular value decomposition. *Apart from prior studies, we pick authentication bits on the internal gate layer and add the authentication function on them. It is more flexible and transparent since it is not limited to specific scenarios such as distillation learning and does not modify the model structure.*

8 Conclusion

We propose a novel authentication scheme AuthNet for neural networks that can autonomously authenticate users and prevent model misuse. Differently from traditional methods, AuthNet's authentication ability is integrated inside the model by reconstructing neural networks with three building blocks: the *head model* is to extract authentication information, which is then encoded in the *gate layer*, and the *tail model* can recognize the identity and make prediction accordingly. Through experiments, AuthNet shows superior performance in user authentication, efficiency in practical use, and resilience to adaptive attacks.

Acknolwedgement

Thanks for all the anonymous reviewers and their constructive comments. The IIE authors are supported in part by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDB0690100), Beijing Natural Science Foundation (L253025), CAS Project for Young Scientists in Basic Research (Grant No. YSBR-118) and NSFC (92270204).

References

[1] H. Ai, W. Xia, and Q. Zhang. Speaker recognition based on lightweight neural network for smart home solutions. In *Cyberspace Safety and Security: 11th International Symposium, CSS 2019*, 2019.

[2] L. Batina, S. Bhasin, D. Jap, and S. Picek. Csi neural network: Using side-channels to recover your artificial neural network information. *ArXiv*, abs/1810.09076, 2018.

[3] C. Blakeney, N. Huish, Y. Yan, and Z. Zong. Simon says: Evaluating and mitigating bias in pruned neural networks with knowledge distillation. *ArXiv*, abs/2106.07849, 2021.

[4] F. Cartella, O. Anunciação, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht. Adversarial attacks for tabular data: Application to fraud detection and imbalanced data. *Proceedings of the Workshop on Artificial Intelligence Safety 2021 (SafeAI 2021) co-located with the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021), Virtual, February 8*, 2021.

[5] M. Chen and M. Wu. Protect your deep neural networks from piracy. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7, 2018. doi: 10.1109/WIFS.2018.8630791.

[6] Y. Chen, J. Tian, X. Chen, and J. Zhou. Effective ambiguity attack against passport-based dnn intellectual property protection schemes through fully connected layer substitution. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[7] L. Deng, G. Hinton, and B. Kingsbury. New types of deep neural network learning for speech recognition and related applications: an overview. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

[8] Z. Deng, K. Chen, G. Meng, X. Zhang, K. Xu, and Y. Cheng. Understanding real-world threats to deep learning models in android apps. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[9] V. Duddu, D. Samanta, D. V. Rao, and V. E. Balas. Stealing neural networks via timing side channels. *ArXiv*, abs/1812.11720, 2018.

[10] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 31–36, 2018.

[11] L. Fan, K. W. Ng, and C. S. Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[12] L. Fan, K. W. Ng, C. S. Chan, and Q. Yang. Deepipr: Deep neural network ownership verification with passports. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6122–6139, 2021.

[13] S. Han, J. Pool, J. Tran, and W. J. Dally. Learning both weights and connections for efficient neural network. In *Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1135–1143, 2015.

[14] H. Hashemi, Y. Wang, and M. Annavaram. DarKnight: An Accelerated Framework for Privacy and Integrity Preserving Deep Learning Using Trusted Hardware. In *54th Annual IEEE/ACM International Symposium on Microarchitecture*, 2021.

[15] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–800, 2018.

[16] Y. He, G. Meng, K. Chen, J. He, and X. Hu. DRMI: A Dataset Reduction Technology based on Mutual Information for Black-box Attacks. In *Proceedings of the 30th USENIX Security Symposium (USENIX)*, Aug. 2021.

[17] W. Hua, Z. Zhang, and G. E. Suh. Reverse engineering convolutional neural networks through side-channel information leaks. *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6, 2018.

[18] Y. Huang and C. Chen. Smart app attack: Hacking deep learning models in android apps. *IEEE Transactions on Information Forensics and Security*, 17:1827–1840, 2022.

[19] Y. Huang, H. Hu, and C. Chen. Robustness of on-device models: Adversarial attack to deep learning models on android apps. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 101–110, 2021.

[20] M. Juuti, S. Szyller, S. Marchal, and N. Asokan. PRADA: protecting against DNN model stealing attacks. In *IEEE European Symposium on Security and Privacy*, pages 512–527. IEEE, 2019.

[21] E. Lee, J.-W. Lee, J. Lee, Y.-S. Kim, Y. Kim, J.-S. No, and W. Choi. Low-Complexity Deep Convolutional Neural Networks on Fully Homomorphic Encryption Using Multiplexed Parallel Convolutions. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 12403–12422. PMLR, 17–23 Jul 2022.

[22] X. Lei, A. Senior, A. Gruenstein, and J. Sorensen. Accurate and compact large vocabulary speech recognition on mobile devices. *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association*, 2013.

[23] C. Li, C. Guo, L. Han, J. Jiang, M.-M. Cheng, J. Gu, and C. C. Loy. Low-light image and video enhancement using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (12):9396–9416, 2022.

[24] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[25] Q. Li and E.-C. Chang. Zero-knowledge watermark detection resistant to ambiguity attacks. In *Workshop on Multimedia & Security*, 2006.

[26] K. Loukhaoukha, A. Refaey, and K. Zebbiche. Ambiguity attacks on robust blind image watermarking scheme based on redundant discrete wavelet transform and singular value decomposition. *Journal of Electrical Systems and Information Technology*, 4:359–368, 2017.

[27] J. X. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Conference on Empirical Methods in Natural Language Processing*, pages 119–126, 2020.

[28] B. Olney and R. Karam. Protecting deep neural network intellectual property with architecture-agnostic input obfuscation. In *Great Lakes Symposium on VLSI 2022*, 2022.

[29] T. Orekondy, B. Schiele, and M. Fritz. Knockoff nets: Stealing functionality of black-box models. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[30] H. T. Sencar and N. D. Memon. Combatting ambiguity attacks via selective detection of embedded watermarks. *IEEE Transactions on Information Forensics and Security*, 2:664–682, 2007.

[31] S. P. Singh, A. Kumar, H. Darbari, L. Singh, A. Rastogi, and S. Jain. Machine translation using deep learning: An overview. In *2017 International Conference on Computer, Communications and Electronics (Comptelix)*, pages 162–167, 2017. doi: 10.1109/COMPTELIX.2017.8003957.

[32] D. F. Smith, A. Wiliem, and B. C. Lovell. Face recognition on consumer devices: Reflections on replay attacks. *IEEE Transactions on Information Forensics and Security*, 10(4):736–745, 2015.

[33] H. Soliman, A. Saleh, and E. Fathi. Face recognition in mobile devices. *International Journal of Computer Applications*, 73(2), 2013.

[34] Z. Sun, R. Sun, L. Lu, and A. Mislove. Mind your weight (s): A large-scale study on insufficient machine learning model protection in mobile apps. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1955–1972, 2021.

[35] Z. Sun, R. Sun, C. Liu, A. R. Chowdhury, L. Lu, and S. Jha. ShadowNet: A Secure and Efficient On-device Model Inference System for Convolutional Neural Networks. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1596–1612. IEEE, 2023.

[36] S. Tan, B. Knott, Y. Tian, and D. J. Wu. Cryptgpu: Fast privacy-preserving machine learning on the gpu. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1021–1038, 2021. doi: 10.1109/SP40001.2021.00098.

[37] Z. Tan, Z. Yang, M. Zhang, Q. Liu, M. Sun, and Y. Liu. Dynamic multi-branch layers for on-device neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:958–967, 2022.

[38] R. Tang, M. Du, and X. Hu. Deep serial number: Computational watermarking for dnn intellectual property protection. *ArXiv*, abs/2011.08960, 2020.

[39] C. Tran, F. Fioretto, J.-E. Kim, and R. Naidu. Pruning has a disparate impact on model accuracy. In *Advances in Neural Information Processing Systems*, volume 35, pages 17652–17664, 2022.

[40] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao. Learning deep transformer models for machine translation. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[41] J. Wei, Y. Zhang, Z. Zhou, Z. Li, and M. A. A. Faruque. Leaky dnn: Stealing deep-learning model secret with gpu context-switching side-channel. *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 125–137, 2020.

[42] L. Wei, B. Luo, Y. Li, Y. Liu, and Q. Xu. I know what you see: Power side-channel attack on convolutional neural network accelerators. In *Proceedings of the 34th Annual Computer Security Applications Conference*, ACSAC '18, page 393–406, 2018.

[43] M. Xue, Y. Zhang, J. Wang, and W. Liu. Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations. *IEEE Trans. Artif. Intell.*, 3(6):908–923, 2022.

[44] Y. Yan, X. Pan, M. Zhang, and M. Yang. Rethinking White-Box watermarks on deep learning models under neural structural obfuscation. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2347–2364. USENIX Association, Aug. 2023.

[45] K. Yang, T. Xing, Y. Liu, Z. Li, X. Gong, X. Chen, and D. Fang. cDeepArch: A Compact Deep Neural Network Architecture for Mobile Sensing. *IEEE/ACM Transactions on Networking*, 2019.

[46] S. Yang, P. Luo, C.-C. Loy, and X. Tang. From facial parts responses to face detection: A deep learning approach. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[47] H. Yu, H. Ma, K. Yang, Y. Zhao, and Y. Jin. Deepem: Deep neural networks model recovery through em side-channel information leakage. *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 209–218, 2020.

[48] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems*, 2018.

[49] J. Zhang, D. Chen, J. Liao, W. Zhang, G. Hua, and N. Yu. Passport-aware normalization for deep model protection. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, page 10, 2020.

[50] W. Zhang, Q. Z. Sheng, A. A. F. Alhazmi, and C. Li. Adversarial attacks on deep-learning models in natural language processing. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11:1 – 41, 2019.

[51] Z. Zhang, J. Geiger, J. Pohjalainen, A. E.-D. Mousa, W. Jin, and B. Schuller. Deep learning for environmentally robust speech recognition: An overview of recent developments. *ACM Trans. Intell. Syst. Technol.*, 9(5), apr 2018. ISSN 2157-6904. doi: 10.1145/3178115.

[52] Z. Zhang, C. Gong, Y. Cai, Y. Yuan, B. Liu, D. Li, Y. Guo, and X. Chen. No privacy left outside: On the (in-)security of tee-shielded DNN partition for on-device ML. In *IEEE Symposium on Security and Privacy, 2024*, pages 3327–3345. IEEE, 2024.

[53] Z. Zhang, N. Wang, Z. Zhang, Y. Zhang, T. Zhang, J. Liu, and Y. Wu. Groupcover: A secure, efficient and scalable inference framework for on-device model protection based on tees. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[54] T. Zhao, Y. Xie, Y. Wang, J. Cheng, X. Guo, B. Hu, and Y. Chen. A Survey of Deep Learning on Mobile Devices: Applications, Optimizations, Challenges, and Research Opportunities. *Proceedings of the IEEE*, 110 (3):334–354, 2022.