Quest Journals
Journal of Research in Applied Mathematics
Volume 11 ~ Issue 4 (April 2025) pp: 68-82
ISSN (Online): 2394-0743 ISSN (Print): 2394-0735
www.questjournals.org

Review Paper

Research on Copyright Protection Methods for Deep Learning Models Based on Adversarial Examples

QingsongYang[1],Jianlin Lu[2] ,Shiwen Wang[3]
[1]*School of Mathematical Sciences, Guizhou Normal University, Guiyang 550001, China,*
[2.] *School of Mathematical Sciences, Guizhou Normal University, Guiyang 550001, China,*
[3.] *School of Mathematical Sciences, Guizhou Normal University, Guiyang 550001, China,*

Abstract
The With the widespread application of artificial intelligence technology,copyright protection for deep learning models has become a core industry requirement. To addressthe issue that existing model watermarking schemes require structural modifications to modelswhich may incur potential performance risks, this paper proposes a non-intrusive copyrightprotection method based on adversarial examples. The method leverages the characterizationcapability of adversarial examples regarding model decision boundaries, constructing a sequenceof adversarial samples covering multi-class decision boundaries through targeted adversarial generation algorithms as watermark carriers, while encoding copyright information into the sequencegeneration process. Watermark embedding requires no modification of model parameters orarchitecture, only requiring pre-storage of the adversarial sample set and corresponding outputs asauthentication credentials with authoritative institutions. During copyright verification, ownershipconfirmation can be achieved by comparing the matching degree between the suspected model'sresponse to watermark samples and the authentication credentials. Experiments demonstrate thatthis method achieves considerable copyright verification success rates while preserving originalmodel performance, with adversarial watermark samples showing robustness against commonwatermark removal attacks. This research provides a secure and interpretable novel solution fordeep learning model protection.

Keywords: *Deep learning model; adversarial samples; copyright protection.*

R**eceived 13 Apr., 2025; **R**evised 24 Apr., 2025; **A**ccepted 26 Apr., 2025 © The author(s) 2025. **P**ublished with open access at **www.questjournas.org

I. Introduction

1.1 Research Background and Significance

The In the era of artificial intelligence, the information explosion has created an urgent demand for efficient processing technologies. In 2006,*Hinton* et al. [1] introduced theconcept of "deep learning", which attracted widespread attention in academia. Today, ithas demonstrated cross-domain innovation value in fields such as medical diagnosis (disease prediction), financial risk control (decision optimization), and intelligent transportation(autonomous driving), emerging as a key technology driving industrial advancement andsocietal development.

In recent years, the field of deep learning model security has begun exploring novel defense mechanisms, among which digital watermarking-based solutions (referred to as "deeplearning model watermarking" [2]) have emerged as a significant technical approach. Digitalwatermarking constitutes a security technique that embeds specific identification informationinto the underlying features of digital carriers (e.g., images, videos, audio) while preservingtheir core functional attributes, thereby achieving information concealment.From a technical implementation perspective, digital watermarking primarily manifests in two categories:robust watermarks and fragile watermarks. The core functionality of robust watermarks liesin enabling copyright traceability, with their technical characteristic being the capacity toensure that embedded watermark identifiers withstand typical interference operations suchas rewriting attacks and data compression, maintaining recognizability even in adversarialenvironments.

While research in deep learning model watermarking continues to advance with significant achievements, its practical applications remain primarily focused on model copyrightprotection. Current mainstream techniques can be categorized into two approaches basedon the accessible model information during protection:

(1) White-box watermarking: Requires complete access to the model's internal parameters(including network architecture and weight matrices) for verification.

(2) Black-box watermarking: Relies solely on input-output characteristics through model API interactions, employing specially constructed test samples to capture the model'sresponse patterns to specific inputs.

These two methodologies cater to distinct application scenarios and verification requirements. White-box verification demands full access privileges to the model's core parameters,whereas black-box implementation depends on analyzing input-output behavior via API interfaces.

1.2 Research Status at Home and Abroad

1.2.1 History and Current Status of Adversarial Example Techniques

According to the level of knowledge the attacker has about the model, adversarial attacks are mainly divided into white-box attacks and black-box attacks. In a white-box attackscenario, the attacker has full access to the target model's architecture, training data, andweight parameters. In a black-box attack scenario, the attacker can only access the model'sinput-output interface, with no knowledge of the internal architecture, training parameters,or other critical information.

In white-box attack scenarios, attackers possess complete access to the model's architecture and parameters, granting them significant advantages in generating adversarial samples. Gradient-based methods, as mainstream techniques, are highly favored byresearchers due to their exceptional computational efficiency. A typical example is theFast Gradient Sign Method(*FGSM*)proposed by Goodfellow et al.[3], which generatesadversarial samples through a single forward propagation pass. However, constrained by itsglobal perturbation strategy, this method tends to produce adversarial samples with relatively high visual distortion.

Building upon the *FGSM* framework, *Dong* et al.[4] incorporated the momentum concept to develop *MI-FGSM* (Momentum Iterative Fast Gradient Sign Method). Thisapproach not only stabilizes the update direction but also effectively guides the loss function towards superior solution spaces, overcoming local optima constraints. Consequently,the generated adversarial samples demonstrate enhanced attack potency and superior crossmodel transferability.*Xie* et al.[5] proposed *DI-FGSM* (*Diverse InputFGSM*), whose coreupdate mechanism resembles the Basic Iterative Method (BIM)while maintaining highcompatibility with other attack techniques. This method ensures broad applicability anddelivers strong attack performance in both white-box and black-box scenarios.Lin et al.[6]integrated the *Nesterov Accelerated Gradient* algorithm [7] with iterative *FGSM* to create *NI-FGSM*,significantly accelerating the attack convergence process. Reference[8]employed gradient information from previous iterations to refine current gradients, thereby improving sample transferability and effectively avoiding local optima during gradientsearch.*Wang* et al.[9] accounted for inter-regional gradient correlations in images by accumulating gradient momentum across both temporal and spatial domains. *Phan* et al.[10]leveraged Class Activation *M* aps (*CAM*)to identify image regions critical for model decisions, an approach that simultaneously accelerates adversarial sample generation and enhances their transfer characteristics.

The adversarial dynamics of adversarial example techniques-characterized by theirdynamic triggering mechanisms and precise characterization of model decision boundaries) offer novel solutions to address the limitations of traditional deep learning watermarking incopyright protection and model integrity verification.

1.2.2 The Evolution of Digital Watermarking Technology in Model Protection

Traditional digital watermarking techniques embed information through spatial domains(e.g,*LSB*) or transform domains (*DCT, DWT*), but face the inherent contradiction betweenrobustness and imperceptibility. With the advancement of deep learning, watermarking technology has shifted toward embedding information in model parameters and dynamic features.

In 2017, *Uchidaetal*. proposed a groundbreaking copyright protection strategy for deepneural networks. They introduced an additional regularization loss function during modeltraining, establishing a comprehensive loss function framework that simultaneously achievesdual objectives: optimizing model performance and embedding watermark information during the training phase.

In subsequent research,*Wangetal*.[11] made a groundbreaking departure from traditional loss functions by constructing a *GAN* − like adversarial learning framework. Thisapproach maintains high similarity between the weight distribution of watermarked modelsand original models, thereby enhancing watermark stealth. Their innovative adoption of a deep neural network as the watermark decoder significantly increases the information

capacity of watermarks, resulting in a novel solution that combines both anti-detection andanti-removal properties.

In black-box watermarking scenarios, the crucial step involves constructing a trigger setto train the neural network. This trigger set consists of specific input samples and theircorresponding predefined labels. After training, when these trigger samples are fed intothe network, it should output the predetermined labels)knowledge exclusively held by themodel owner. To prove ownership, the holder can demonstrate the input-output correspondence using these triggers without accessing any internal model information, maintainingthe neural network as a complete "black box" throughout verification.Capitalizing on theunique advantages of trigger-based watermarking, researchers have developed various triggerconstruction methods. *Adietal*.[12] pioneered the integration of model backdoors with watermarking mechanisms. Their approach resembles data poisoning: during training, an imageset statistically independent of the original data distribution serves as trigger signals, withthese samples forcibly mapped to random class labels. While normal models struggle to establish stable correlations between triggers and assigned labels, watermarked models outputthe predefined labels with high probability, thereby verifying ownership. *UnlikeUchidaetal*.conventional weight-modification approach, this scheme activates watermarks by establishingspecific input-output mappings.

1.3 Research The main research content of this paper

While adversarial examples inherently pose security risks to models, this work explores their beneficial potential by investigating their use in model copyright protection-specifically, embedding watermarks without compromising model functionality.The keycontributions of this study are:

(1) Adversarial sequence-based copyright protection: We propose a novel method that utilizes targeted adversarial examples with large perturbation steps and low decision-boundarysensitivity.

(2) Attack Resilience: The solution maintains high stability against common threats likemodel fine-tuning and parameter tampering, enabling reliable black-box copyright verification.

(3) Empirical Robustness: Experiments demonstrate the method's effectiveness in preserving watermark integrity under various attack scenarios while successfully verifying ownership.

II. Related Theories and Technologies

2.1 Introduction to Deep Neural Network

Deep Neural Networks (*DNN*), also referred to as Deep Learning Networks, represent a class of artificial neural network architectures characterized by multiple neural network layers (commonly called hidden layers). These adjacent hidden layers establish cascaded connections through weight matrices, where the output from preceding layers serves as the input source for subsequent layers.

2.1.1 Convolutional Neural Network

Convolutional ~Neural~ Network (*CNN*), as a representative algorithm of deep learning, combine the advantages of convolutional operations and deep architectures, belonging to the category of feedforward neural networks.

The artificial neurons in *CNN* can respond to units within their local neighborhood, making them highly effective in processing large-scale images. Compared to traditional fully connected neural networks, *CNN* leverage spatial locality and translation invariance, significantly reducing the number of model parameters while greatly improving the speed and efficiency of image processing. This characteristic allows *CNN* to directly process raw image data without complex preprocessing. A typical *CNN* primarily consists of the following layers: input layer, convolutional layer, activation layer, pooling layer, and fully connected layer.Here is an introduction to the main structures:

(1) Input Layer

The input layer receives raw image data, typically represented as a three-dimensional tensor(height × width × number of channels). For *RGB* images, the number of channels is *3*, with each channel corresponding to a pixel brightness matrix, where values range from *[0, 255]*. Mathematically, it can be expressed as:

$$I=R^{H\times W\times C}, \quad (0\text{-}1)$$

where H is the height, W is the width, and C is the number of channels.

(2) Convolutional Layer

As the core component of a convolutional neural network, the convolutional layer operates based on the biologically inspired mechanisms of local receptive fields and weight sharing. It performs spatial feature

extraction through two-dimensional discrete convolution operations. Mathematically, this operation can be defined as a weighted summation process between the input feature map I and the convolution kernel *K*:

$$(I * K)_{m,n} = \sum_{i=-k}^{k} \sum_{j=-k}^{k} I(m-i, n-j) \cdot K(i,j), \text{(0-2)}$$

as the kernel slides over the input data, the pixel values in the local region are linearly combined with the kernel parameters to generate a feature map. This process preserves the spatial locality of the input data while significantly reducing the number of parameters through weight sharing. For example, a *3×3* convolution kernel shares the same *9* parameters across all positions within the same channel, leading to a parameter reduction of several orders of magnitude compared to the fully connected layer's global connectivity pattern.

The convolutional layer core functionality lies in its multi-level feature abstraction capability. Shallow convolutional kernels typically capture low-level features such as edges and textures, while deeper layers progressively extract semantic information. The zero-padding strategy (e.g., padding=1) in the design maintains spatial dimensional consistency between input and output, preventing edge information loss. The parameter scale calculation must account for the input channel count *C*, output channel count *G*, and convolutional kernel size *K*, expressed by the formula:

$$Params = K \times K \times C_{in} \times C_{out} + C_{out}. \text{(0-3)}$$

Variants of convolutional layers have significantly expanded their application boundaries. Dilated convolutions introduce dilation rates to enlarge receptive fields, making them suitable for tasks requiring long-range dependency modeling; depth wise separable convolutions decouple spatial and channel-wise convolutions, dramatically reducing computational complexity; grouped convolutions improve computational efficiency through parallel channel-wise processing. While preserving the core operational logic, these variants optimize computational pathways and feature interaction modes for specific task requirements. The prevailing design paradigm employs stacked small-sized convolutional kernels (et al. *3×3*), where multi-layer stacking achieves receptive fields equivalent to larger kernels while enhancing nonlinear expressive capabilities. The dynamic adjustment of convolutional kernel parameters relies on backpropagation algorithms combined with gradient descent optimization strategies, enabling the network to adaptively learn discriminative features from data.

(3) Activation Layer

The activation layer enhances the model's representational power beyond linear models through nonlinear mapping mechanisms. Its core value lies in overcoming the limitations of linear operations-if the network consisted solely of linear transformations, multi-layer stacking would degenerate into a single linear transformation, incapable of approximating complex functional relationships. The introduction of activation function $f(\cdot)$ enables neural networks to construct nonlinear decision boundaries through hierarchical stacking, thereby achieving effective modeling of high-dimensional data. The operation of the activation layer can be described as:

$$y = f(z), \text{ where } z = \sum w_i x_i + b. \text{(0-4)}$$

This process transforms the linearly weighted sum z from the preceding layer into a nonlinear response y. Taking the *ReLU*(Rectified~ Linear~ Unit) function as an example, its expression is:

$f(z) = \max(0, z)$, This piecewise-linear design maintains computational efficiency while achieving implicit feature selection by suppressing negative inputs, significantly enhancing the network's sparse representation capability. Compared to traditional Sigmoid functions, which suffer from gradient saturation issues, *ReLU* maintains a constant gradient of 1 in the positive region. This property effectively mitigates the vanishing gradient problem during the training of deep networks.

The functional diversity of activation layers is reflected in the design characteristics of different functions. The Sigmoid function $f(z) = 1/(1+e^{-z})$ constrains outputs to the (0, 1) interval, making it suitable for probability mapping scenarios; the Tanh function $f(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$ features zero-centered symmetric distribution, accelerating gradient descent convergence; while Leaky*ReLU* addresses neuron "death" issues by introducing a small negative slope (e.g., 0.01). In deep network architectures, the cascading effect of activation layers creates compound nonlinear mappings. The choice of activation function at each layer essentially establishes distinct feature space transformation rules-shallow layers tend to extract local detail features, while deeper layers progressively combine these into global semantic representations. This mechanism of hierarchical nonlinear stacking enables neural networks to approximate continuous functions of arbitrary

complexity, satisfying the theoretical requirements of the Universal Approximation Theorem. Notably, the differentiability of activation functions serves as a fundamental condition for their widespread adoption, ensuring gradients can propagate layer-by-layer through backpropagation via the chain rule.

(4) Pooling layer

Pooling layers in convolutional neural networks serve the dual purpose of feature dimensionality reduction and spatial information compression. Typically inserted between successive convolutional operations, they perform nonlinear downsampling on feature maps through local receptive fields. The Max Pooling operation extracts the maximum value from a window region Ω:

$$y_{i,j} = \max_{p=0}^{k_h-1} \max_{q=0}^{k_w-1} x_{i\times s+p, j\times s+q}, \text{(0-5)}$$

this operation preserves salient texture features while suppressing background noise. Average Pooling achieves smoothing by computing regional means:

$$y_{i,j} = \frac{1}{k_h \times k_w} \sum_{p=0}^{k_h-1} \sum_{q=0}^{k_w-1} x_{i\times s+p, j\times s+q}, \text{(0-6)}$$

this variant is particularly suited for scenarios requiring global information aggregation.

He configuration of the pooling window size $k\times k$ and sliding stride s determines the feature map downscaling ratio $\left[\frac{H-k}{s}+1\right]$, where *H* represents the input height. When $s=k$, non-overlapping sampling is achieved. This design was widely adopted in early architectures like *LeNet-5*. By setting $s<k$, adjacent windows share overlapping regions. The coverage ratio $\rho = 1-s/k$ directly affects information retention. Employs multi-scale windows $\{k_1\times k_1, k_2\times k_2, \ldots\}$ to generate hierarchical feature vectors: $f = [f_{k_1}; f_{k_1}; \ldots]$. This approach demonstrates strong robustness to scale variations in object detection tasks. The pooling operation introduces translation invariance through a downsampling factor *d*, ensuring that a small displacement Δ of the input *X* satisfies $P(X) \approx P(X+D)$. Modern networks typically control the number of deep pooling layers L_p to be $L_p \leq 3$, and instead, use dilated convolutions with a dilation rate *R* to achieve controllable downsampling. Adaptive pooling dynamically calculates the output size $h_0 \times \omega_0$ to satisfy equation:

$$h_0 = \left[\frac{H+2p-k}{s}\right]+1, \text{(0-7)}$$

addressing the issue of input resolution differences. Unspooling operations in decoder networks utilize pooling position indices to reconstruct feature maps:

$$x_{p,q} = \begin{cases} y_{i,j}, & if\ (p,q) = I(i,j), \\ 0, & otherwise, \end{cases} \text{(0-8)}$$

this provides a spatial information recovery mechanism for generative adversarial networks.

(5) Fully Connected Layers

The fully connected *(FC)* layer achieves high-level semantic modeling through global feature integration. Mathematically, it can be described as a combination of linear mapping and nonlinear activation in feature space: given an input feature vector $x \in \mathbb{R}^d$, the output $y \in \mathbb{R}^m$ satisfies:

$$z = Wx+b, y = \sigma(z), \text{(0-9)}$$

where $W \in \mathbb{R}^{m\times d}$ is the weight matrix, $b \in \mathbb{R}^m$ is the bias vector, $\sigma(\cdot)$ is the nonlinear activation function. In *CNN*, the FC layer typically serves as the final classifier. It flattens the *2D* feature maps $X \in \mathbb{R}^{H\times W\times C^{-1}}$ extracted by convolutional layers into a vector $x \in \mathbb{R}^{H\times W\times C}$, then performs class probability mapping.

2.2 Common attack methods against watermarking in deep learning models

The security of watermarking technology needs to withstand various targeted attacks. Attackers attempt to eliminate the watermark or block the verification mechanism through strategies such as modifying the model

structure, perturbing parameters, or interfering with inputs. The following explains typical methods from the perspectives of attack implementation paths and destruction targets.

(1) Model Fine-tuning Attacks

Attack principle: Continues training the watermarked model with new data to overwrite dilute the watermark.

Variants: 1.Transfer learning-based fine-tuning; 2.Layer-wise selective fine-tuning; 3.Adversarial fine-tuning with perturbed data.

(2) Model Pruning Attacks

Attack principle: Removes redundant neurons/channels containing watermark signals

\indent Effectiveness: 1.Weight pruning can eliminate 60-80\% watermarks;

2.Channel pruning shows 85-95\% removal rate.\\

Advanced methods:1.Lottery ticket hypothesis-based pruning; 2.Adaptive magnitude pruning: 1.Transfer learning-based fine-tuning.

(3) Model Distillation Attacks

Attack principle: Transfers knowledge to new compact models excluding watermarks

Implementation forms: 1.Conventional knowledge distillation;

2.Adversarial example-enhanced distillation; 3.Multi-teacher distillation.

Success rate: >90\% watermark removal in most cases.

(4) Parameter Manipulation Attacks

Includes: 1.Weight quantization (8-bit quantization removes 40-60\% watermarks); 2.Weight shuffling/permutation; 3.Low-rank decomposition.

Characteristics: Maintains model functionality while erasing watermarks.

(5) Model Inversion Attacks

1.Reconstructs training data to extract and remove embedded watermarks; 2.Works particularly well against backdoor-based watermarks.

(6)Adversarial Attacks

Generates specific inputs to: 1.Activate false watermark signals; 2.Suppress genuine watermark responses; 3.Cause watermark misidentification.

Defensive Considerations: 1.Current watermarking schemes can resist 1-2 attack types simultaneously; 2.Multi-component watermarks combining different embedding strategies show better robustness; 3.Dynamic watermarks with verification protocols are more resilient than static ones.

III. Adversarial Example-based Copyright Protection Methods for Deep Learning Models

This chapter addresses the limitations of traditional digital watermarking techniques that rely on modifying model parameters and structures, proposing an adversarial watermark generation framework based on fast boundary attack. The method employs dynamic step size adjustment and momentum search strategies to ensure the effectiveness of adversarial examples while allowing controlled step sizes as needed, thereby generating adversarial samples with enhanced robustness. Furthermore, by establishing a mapping relationship between watermark bits and output distributions, the framework achieves watermark embedding without requiring any modifications to the network parameters or architecture.

3.1 Adversarial Sample-Based Watermarking Framework for Deep Learning Models

He framework workflow is illustrated in Figure 1, which consists of the following key components.

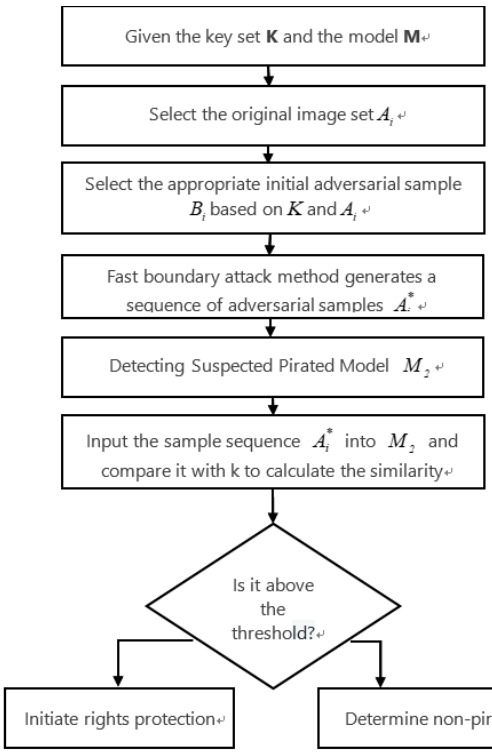

Figure 1: Flowchart of the deep learning network watermarking framework based on adversarial examples

3.1.1 Basic Principles of Adversarial Sample Watermarking

The classification behavior of deep neural networks is determined by the topological structure of their decision boundaries. The generation of adversarial samples is essentially an exploration of the local geometric properties of this boundary. In classification models, the decision boundary is defined as the critical hypersurface within the input space where the predicted probabilities of different classes are equal. Let the decision boundary constructed by the classification model $f_\theta: R^d \to R^k$ in the input space *X* be represented as the comparison of similarity between a sample sequence A_i^* input to M_2 and *K*.

$$B_k = \left\{ x \in X \mid f_\theta^{(k)}(x) = \max_{j \neq k} f_\theta^{(j)}(x) \right\}, \text{(0-10)}$$

in the process of generating adversarial samples, the role of the perturbation δ is essentially to move the original sample *x* along a specific direction to the outside of the adjacent decision boundary, thereby triggering a sudden change in the classification result. For the target class y^*, its adversarial sample $x^* = x + \delta$ needs to satisfy $f_\theta(x^*) \neq f_\theta(x)$ and $\|\delta\| \leq \varepsilon$, where ε is the perturbation threshold. This is equivalent to finding the minimal perturbation path that crosses the decision boundary.

The decision boundaries of different models have unique differential structures, which provide geometric fingerprints for copyright verification. The model discrepancy $\Delta(f_\theta, f_{\theta'})$ quantifies the inconsistency of their decision boundary structures by measuring the statistically significant differences between two models on an adversarial sample set. This helps in identifying the structural differences in decision boundaries, offering a way to distinguish models based on their responses to adversarial perturbations. The metric is defined as:

$$\Delta(f_\theta, f_{\theta'}) = E_{x \sim D_{adv}} \left[\left\| \nabla_x f_\theta(x) - \nabla_x f_{\theta'}(x) \right\|_F^2 \right], \text{(0-11)}$$

the equation holds if and only if $\Delta(f_\theta, f_{\theta'}) = 0$, which implies $f_\theta = f_{\theta'}$, where $\|\cdot\|_F$ denotes the *Frobenius* norm of the matrix, essentially calculating the mean squared difference of input gradients between the two models for all adversarial samples x^*. The gradient $\nabla_x f_\theta(x)$ reflects the model's sensitivity to input

variations at x^*, with its direction pointing to the steepest ascent path toward the decision boundary. When the topological structures of the decision boundaries of the two models $f_\theta, f_{\theta'}$ are completely identical, their gradient fields are equal everywhere on D_{adv}, in which case $\Delta(f_\theta, f_{\theta'}) = 0$; conversely, the deviation of the gradient fields directly reflects the differences in the classification mechanisms of the two models. The information entropy characteristics of this metric further ensure that by sufficiently covering the adversarial sample set of the decision boundary, the unique 'fingerprint' features of a specific model can be identified.

The essence of copyright authentication for deep learning models in this paper lies in constructing adversarial perturbations with identity characteristics, which form a stable mapping with the decision boundary of the target model. By generating an adversarial sample set that covers multi-class boundaries, the model's geometric fingerprint features can be extracted. Based on this, watermark verification can be transformed into a decision boundary verification problem. Furthermore, model copyright authentication can be achieved by examining the outputs of the adversarial sample set spanning multiple decision boundaries.

3.1.2 Fast Boundary Attack-based Adversarial Example Generation

Since generating a large number of multi-class adversarial examples is required to characterize the decision boundary, there is a need for a targeted adversarial example generation algorithm that is both efficient and effective. This paper proposes an improved fast boundary attack-based adversarial example generation algorithm, capable of batch-producing adversarial samples with high efficiency and low model query counts.

This algorithm is mainly divided into three parts:

(1)Select two classes,*x* and *y*, from the training set of the generative network, where*x* is the generative network, where *x* is the original target class, and *y* is the adversarial target class. Search for the image *B* in the training set of class *Y*, such that the distance between image *B* and the original image*A* of class*x* is the smallest. Here, we define the distance between images as the Euclidean distance, which is expressed as:

$$D(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}. \text{(0-12)}$$

(2)Binary Search Method

Upon obtaining the target image *B* as the initial adversarial example, we employ the binary search method to rapidly locate the decision boundary for adversarial example generation. This approach iteratively narrows the search interval between the original sample *A* and the initial adversarial sample *B* to construct an optimal boundary point satisfying $A^* \in Z(y)$, where $Z(y)$ represents the decision region of the target category. The main procedure is as follows: Let the initial adversarial sample *B* satisfy $f(Y) = y$ (where *f* denotes the classifier's mapping function), the original sample *A* satisfies $f(A) \neq y$. The algorithm on the line segment connecting *B* and *A*:

$$L = \{x \mid x = aA + (1-a)B, a \in [0,1]\}, \text{(0-13)}$$

the search for adversarial examples that meet the threshold requirements on the above is performed, and the specific iteration process is as follows:

First, set the lower bound as $\text{low}^0 = A$, the upper bound as $\text{high}^0 = B$, and define the error tolerance threshold as ò. Then, begin the iteration. The *i* iteration can be expressed as:

$$\text{mid}^{(i)} = \frac{\text{low}^{(i)} + \text{high}^{(i)}}{2}, \text{(0-14)}$$

$$\text{mid}^{(i)} \in [0, 255],$$

query the classifier to obtain $f(\text{mid}^{(i)})$, . If $f(\text{mid}^{(i)}) = y$, , then $\text{high}^{(i+1)} = \text{mid}^{(i)}$, otherwise, $\text{low}^{(i)} = \text{mid}^{(i)}$. The iteration stops when the condition $\left\|\text{low}^{(i)} - \text{high}^{(i)}\right\| < ò$ is satisfied. Return A^* as the boundary point.

(3) Surface-based random search:

Since the decision space can be viewed as a multi-dimensional sphere, the binary search method is often difficult to generate effective adversarial examples. Therefore, after obtaining the approximate boundary point

A^*, surface-based random search is continued to find adversarial examples with better performance. The goal of surface-based random search is to construct low-norm perturbations in the local neighborhood of the decision boundary, with the core objective being to explore the optimization path of A^* along the tangent plane direction of $Z(y)$. This is achieved by adding perturbation δ:

$$min\|\delta\|_2, \ \ st \ f(A^* + \delta) = y, \text{(0-15)}$$

and it satisfies $\left\|A-\left(A^*+\delta\right)\right\|_2 < \left\|A-A^*\right\|_2$. The basic process of this algorithm is shown in Algorithm 1.

Algorithm1 : *Fast Boundary Attack Algorithm*

Input: Original sample A, target sample B, target class y, maximum iteration count N

Output: Adversarial sample A^*;

Initialize the momentum accumulation vector $h = 0$, momentum coefficient $\gamma = 0.5$, step size $\eta = \eta_0$; initial adversarial sample $A^* \leftarrow B$, $low^0 = A, high^0 = B$, $\grave{o}_1 = 0.01$, $\grave{o}_2 = \eta * \|A - B\|_2$;

while $\|low^{(i)} - high^{(i)}\| < \grave{o}_1$ **do**

  if $mid^{(i)} \in y$ **then**

    $high^{(i+1)} = mid^{(i)}$;

  else $low^{(i)} = mid^{(i)}$

  end

end

Assign the approximate boundary $mid^{(i)}$ to x_{adv};

while $k < N$ &&$\|A - xadv\|_2 > \varepsilon_2$ **do**

  Generate a random perturbation $\delta \sim N(0,1)$ and linearly combine it with the historical momentum h to maintain directionality: $\delta \leftarrow \delta + \gamma h$, and then eliminate the normal component through orthogonal projection: $\delta \leftarrow \delta - \frac{\delta^T n}{\|n\|_2} n$;

  Move the step size η to generate candidate samples. $x_{cand} = \text{clip}\left(x_{adv} + \eta \frac{\delta}{\|\delta\|}, [0, 255]\right)$;

  if $f(x_{cand}) = y$ and $\|A - x_{cand}\|_2 < \|A - x_{adv}\|_2$ **then**

    Update the momentum buffer h to preserve the directional trend and expand the step size η to enhance exploration capability;

    else Shrink the step size η to avoid ineffective search.

  end

end

Obtain the final adversarial sample $A^* \leftarrow x_{adv}$.

The following methods were employed during the surface search process:

(1) Momentum-guided direction sampling, Generate candidate perturbation $\delta_i = \eta \cdot \frac{v_i}{\|v_i\|_2}$, where $v \sim (N(0,1) + \gamma h_{prev}$ and h_{prev} is the historical perturbation momentum term, with $\gamma \in (0,1)$ being the momentum coefficient. This design biases the search direction towards historically successful directions, reducing randomness.

(2)Adaptive step size control: The step size η is dynamically adjusted based on the success rate. If the iteration successfully finds a better solution, then $\eta \leftarrow \eta \cdot a\left(a > 1\right)$, if the number of failed iterations exceeds a threshold, then $\eta \leftarrow \eta \cdot \beta\left(\beta < 1\right)$.

(3)Continue iterating until the output adversarial sample A^* reaches the predefined difference threshold or the maximum number of iterations.

3.1.3 Adversarial Example-based Copyright Verification

Model stealing typically preserves the topological structure of the original decision boundary, making the boundary deformation caused by adversarial perturbations inheritable. The theoretical core of adversarial sample-based copyright verification methods for deep learning models lies in leveraging the model's sensitivity to specific perturbation patterns to construct identifiable markers. The effectiveness of this method is built upon a dual theoretical foundation: the transferability of adversarial samples and the perturbation characteristics of the model's decision boundary. By encoding copyright information into adversarial perturbation patterns to characterize the model's decision boundary, it further achieves model identity authentication. Since the output of classification models in black-box scenarios is mostly limited to class labels, to enhance the practicality of the method, this paper assumes that the model output consists of categorical information.

(1)Copyright Watermark Generation

The construction of copyright watermarks essentially involves identifying a set of adversarial samples that satisfy dual constraints. First, the adversarial sample set should accurately characterize the decision boundary while adhering to a predefined copyright encoding sequence, i.e., the adversarial sample sequence A^i should satisfy the following condition on the target model *M*: $f\left(A^i\right) = y^i$, where *y* is the preset copyright encoding sequence. Second, the perturbation step size of the adversarial samples must meet certain constraints, expressed as: $\left\| A - A^i \right\|_2 < \varepsilon$

Based on the above principles, a fast boundary attack algorithm can generate adversarial samples for a given target. The model owner only needs to provide a unique fingerprint-encoded sequence, and the corresponding adversarial sample sequence-forming the copyright watermark-can be generated using this algorithm.

(2)Copyright authentication

The model owner submits two sets of authentication credentials to the authoritative verification authority in advance: the adversarial sample set $\left\{A^i\right\}$ and the corresponding output label set $\left\{y^i\right\}$. When a suspicious model M' is identified, $\left\{A^i\right\}$ is fed into the model under verification, and its output $\left\{y_p^i\right\}$ is recorded. Then, the similarity calculation function is used to compute the similarity. If the similarity exceeds a predefined threshold, copyright enforcement procedures may be initiated for further verification of model ownership.

3.2 Experimental Results and Analysis

To verify the feasibility of the proposed solution in this chapter, this section mainly conducts experiments and analysis from the aspects of experimental setup, evaluation metrics, and experimental results.

3.2.1 Experimental Setup

The experiments were conducted using the publicly available *Oxford-IIITPe* dataset, which contains *37* pet categories (covering various breeds of cats and dogs). Three widely-used classification networks—*ResNet50, AlexNet*, and*GoogLeNet-were* trained on this dataset for classification tasks. All experiments were implemented on *NVIDIA ~GeForceGTX 1050 Ti GPU* using *MATLAB~ 2020b*.

For this experiment, adversarial samples meeting the following criteria were generated on the three networks:

(1)Original images: One image was selected from each of the *37* categories as the base image.

(2)Attack generation: Using the Fast Boundary Attack method, two distinct adversarial samples were generated by attacking two different target categories for each base image.

(3)Dataset composition: Each network ultimately produced *74* adversarial samples (*37* base images × *2* attacks), collectively forming the adversarial sample dataset.

A subset of these adversarial samples is displayed in Figure 2.

Figure 2: Partially generated adversarial examples by the algorithm

If the generated adversarial samples are too close to the decision boundary, the robustness of the resulting watermark will be compromised. To prevent copyright verification failure due to decision boundary shifts caused by potential model attacks, this experiment intentionally employs adversarial samples generated with larger perturbations. Consequently, these samples may not achieve optimal visual quality. However, since the sample set does not need to be disclosed during verification, the visual quality does not affect its functional performance.

The output class predictions are logged and paired with the adversarial samples to constitute the watermark set. A sample encoding of the output classes from the adversarial set is demonstrated as: [2,17,22,1,9,7…]. This encoding scheme can be further designed to embed owner identification data.

When generating adversarial examples using the fast boundary attack algorithm, the following coefficients are used: momentum coefficient is *0.55*, maximum step size is *130*, minimum step size is *1*, step size decay rate is *0.99*, and the random search sample size is *300*.

Considering that the essence of model infringement lies in functional exploitation, the degree of tampering should be controlled within a threshold to avoid significantly compromising application performance. For the specific models and dataset used in this experiment, we ensured that any model tampering would not degrade classification performance by more than *15\%* compared to its original capability - tampering attempts exceeding this threshold were considered invalid.

In this study, multiple distinct models were trained on the same dataset. When adversarial samples generated from one model were input as watermarks to other models for output comparison, none achieved a matching rate exceeding *70\%*. Therefore, for watermark validity verification, our experimental standard requires that after tampering, if the adversarial samples maintain their distinctive responses with at least *80\%* probability, the embedded watermark is deemed to have successfully achieved authentication.

3.2.2 Evaluation metrics

Since this paper watermarking does not involve modifications to the model itself, and the maximum watermark capacity is directly determined by the maximum number of classification categories in the model and the number of adversarial samples, this section will focus on evaluating the model's fidelity and robustness, rather than evaluating aspects such as watermark concealment, watermark capacity, and concealability.

(1) Fidelity: In deep learning model watermarking techniques, fidelity refers to the degree to which the embedded watermark affects the model's original functionality. This metric is quantified by comparing the performance differences (e.g., classification accuracy, regression error) between the watermarked model and the

original model on a standard test set. Ideally, the introduction of the watermark should not significantly alter the model's core predictive capabilities.
(2) Robustness: In the context of model watermarking, robustness measures a watermark's survivability against model attacks or modifications. Typical threat scenarios include post-processing operations such as model fine-tuning, parameter pruning, and adversarial example attacks. This metric is critically tied to the legal enforceability of model copyright-watermarks lacking robustness can be easily removed by malicious actors, thereby nullifying their verification utility.

3.2.3 Fidelity Evaluation

Since the watermarking approach proposed in this paper does not compromise model fidelity, this section primarily examines how other established watermarking methods in the literature affect model fidelity. For conciseness, we adopt the following abbreviated notations:
M method: Refers to the remote watermarking framework by *Merrer* et al. [13], which enables watermark extraction via model *APIs* (black-box access).
C method: Denotes the black-box multi-bit watermarking scheme by *Chen* et al. [14], designed to embed and extract high-capacity watermark information.
These methods, along with ours, fall under the category of black-box watermarking techniques, ensuring a high degree of comparability.

Table 1: Comparison of Model Fidelity for Different Black-box Watermarking Methods

Model Category	Network Depth	Original Classification Accuracy(%)	Watermarked Model Accuracy(%)		
			M Method	C Method	Proposed Method
AlexNet	8 layers	84.4	83.5	83.6	84.4
GoogLeNet	22 layers	91.6	90.8	91.0	92.6
ResNet50	50 layers	93.1	92.4	92.4	93.1

As evidenced in Table 1, conventional watermarking approaches invariably degrade model performance across different network architectures to varying and unpredictable degrees. This performance degradation may potentially introduce unforeseen security threats with severe consequences. In contrast, our method requires no modification to the original model parameters, thereby completely preserving the model's native functionality and eliminating such risks entirely.

3.2.4 Robustness Analysis

This subsection conducts three types of attacks-model fine-tuning, pruning, and distillation—against each of the three models, with the constraint that the degradation in classification performance must not exceed *15\%* of the original level. The experimental procedure is as follows: gradually increasing intensities of each attack are applied to the trained models. After each attack iteration, the compromised model is evaluated on the test set to monitor the decline in classification accuracy. The functional degradation rate is calculated using: Functional degradation rate = 1- (Attacked Model Accuracy / Original Model Accuracy). The following results demonstrate how the number of surviving adversarial samples varies under different attack types and tampering intensities.
(1)Model Fine-tuning
The model fine-tuning in this experiment was carried out as follows: A collection of cat and dog images, different from the dataset and test set, were gradually increased in number from online sources to form the new training set. After retraining the network, the model was tested using the original test set to evaluate the functional degradation rate, ensuring that the performance drop did not exceed *15\%*.
As shown in Figure 3, the number of surviving adversarial samples decreases more rapidly in shallower networks *AlexNet*and *GoogLeNet*compared to the deeper *ResNet50* architecture, which maintains significantly greater stability. The reason is that models with low network depth are more prone to having adversarial features overwritten when fine-tuned, on the other hand, models with deeper network depths have larger parameter redundancy, causing adversarial features to be dispersed across residual blocks, which gives adversarial samples stronger resistance to model fine-tuning.

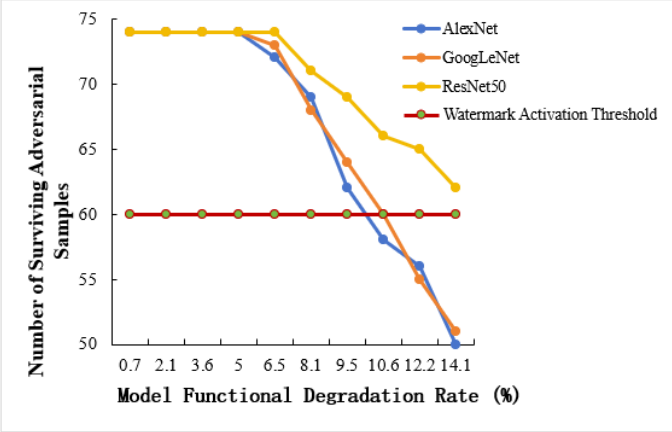

Figure 3: The Effect of Model Fine-tuning on the Number of Surviving Adversarial Samples

(2)Model Pruning

The model pruning in this experiment was carried out as follows: The pruning rate was gradually increased, and the model's classification accuracy was observed, ensuring that the model's functional degradation rate did not exceed *15\%*.

As shown in Figure 4, as the pruning rate increases, the number of surviving adversarial examples for the deeper *ResNet50* network decreases rapidly, while the decrease is slower for the shallower *AlexNet*and *GoogLeNet* networks. The reason for this is that the pruning strategy tends to retain key convolutional kernels, and the shallow weights, which are crucial for adversarial example generation, are preserved, allowing adversarial examples to survive. In contrast, in the *ResNet50* network, the pruning strategy mistakenly deletes adversarially relevant channels in the residual blocks, rendering the adversarial examples ineffective.

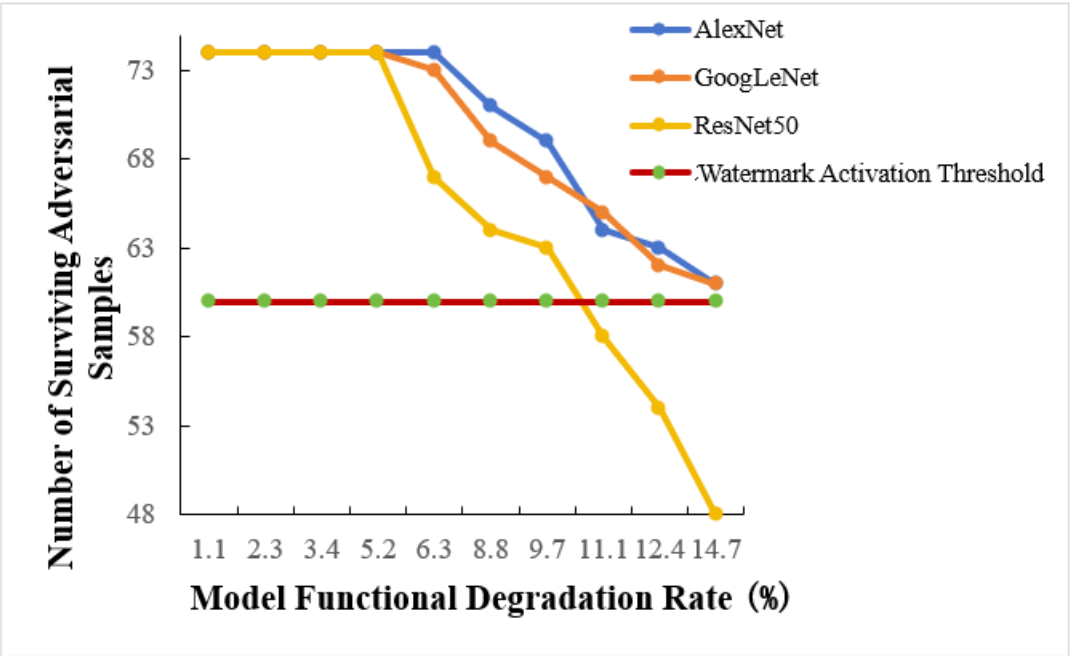

Figure 4: The impact of model pruning on the survival count of adversarial samples

(3)Distillation Attack

\indent In this experiment, the distillation attack is performed as follows: Images searched online are combined with a partially modified training set (such as cropped, rotated, or pixel-modified images) and classified using a trained model. Based on the classification results, these images are used as the training set to train a pirated network with the same class using a pre-trained model. The model is then tested on the original model's test set, with the restriction that the model's performance drop rate does not exceed *15\%*.

As shown in Figure 5, the survival rate of adversarial samples is relatively high across all models, exceeding the watermark effectiveness threshold. This may be due to the fact that under the constraint of the model's

performance degradation, the distilled attack uses a large proportion of the original training set. As a result, the new model is quite similar to the original model, leading to minimal changes in the decision boundary.

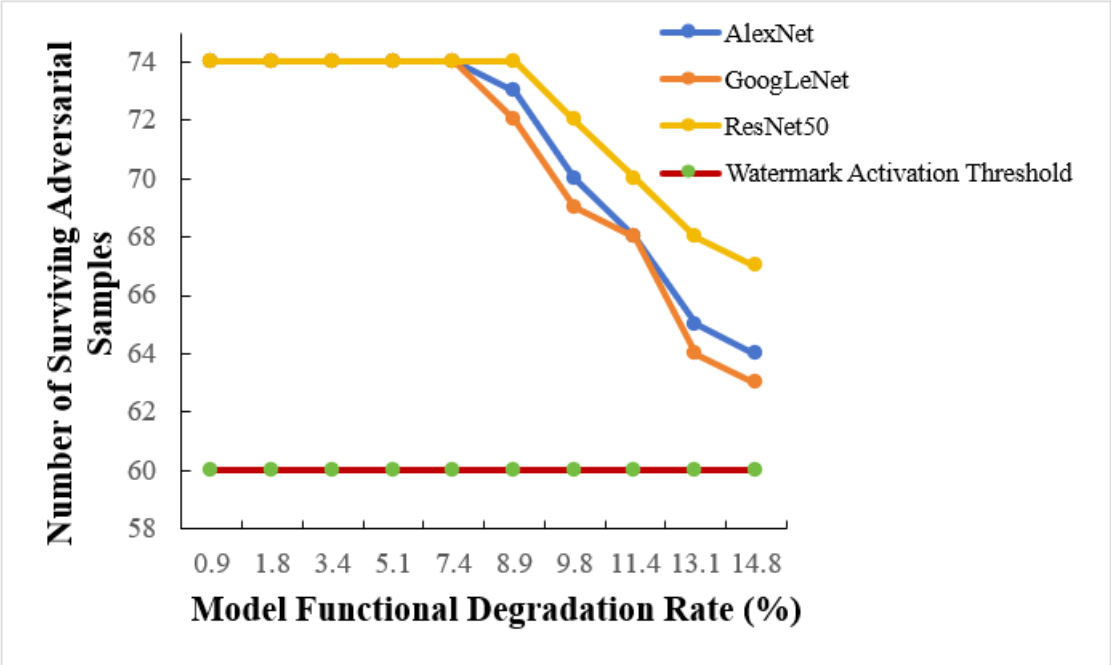

Fig. 5. Effect of Model Distillation Attacks on the Survivability of Adversarial Samples

In summary, when modifications result in only minor functional degradation of the model (low degradation rate), the changes to the model's decision boundary remain relatively small. Under these conditions, the classification labels of adversarial samples remain unchanged. However, as the intensity of model attacks progressively increases, more substantial alterations occur to the decision boundary, causing some adversarial samples to lose their original label assignments. These findings demonstrate that our method can maintain high watermark sample survival rates within certain attack intensity thresholds, thereby remaining effective for model copyright verification.

For comparative analysis with other watermarking approaches, we focus primarily on whether the watermarks remain functional as our evaluation criterion, since different watermarking methods employ varying robustness metrics. The comparative results are presented in Table 2,3,4.

Table 2: Comparison of Model Robustness Across Different Black-box Watermarking Methods

Model Category	Network Depth	Attack Methods		
		Model Fine-tuning		
		M Method	C Method	Proposed Method
AlexNet	8 layers	√	×	×
GoogLeNet	22 layers	×	√	×
ResNet50	50 layers	√	√	√

Table 3: Comparison of Model Robustness Across Different Black-box Watermarking Methods

Model Category	Network Depth	Attack Methods Model Pruning		
		M Method	C Method	Proposed Method
AlexNet	8 layers	√	√	√
GoogLeNet	22 layers	√	√	√
ResNet50	50 layers	×	×	×

Table 4: Comparison of Model Robustness Across Different Black-box Watermarking Methods

Model Category	Network Depth	Attack Methods Distillation Attack		
		M Method	C Method	Proposed Method
AlexNet	8 layers	×	×	√
GoogLeNet	22 layers	×	×	√
ResNet50	50 layers	×	×	√

As can be seen from the table data, under the attack scenario with a limited modification threshold, this method demonstrates stronger resistance to model pruning and distillation attacks compared to other black-box watermarking methods, but its resistance to model fine-tuning is relatively weaker. It is evident that this method shows good robustness against common attacks, making it of certain practical significance.

IV. Summary

This paper proposes a deep learning network watermarking framework based on adversarial example sequences. The method generates targeted adversarial example sequences using the Fast Boundary Attack method and designs the watermark based on this sequence. Compared to traditional methods, the main advantage of this approach is that it does not modify the model itself. Instead, it designs a sequence of adversarial examples with a certain level of robustness based on the model's decision boundary to describe the decision boundary, which is then used as the model watermark to verify copyright. This approach effectively avoids modifications to the model itself, eliminating the invisible risks that could arise from adding a model watermark. Experiments show that, even in the face of common model watermarking attacks, this method still exhibits certain robustness and can be applied in practical scenarios.

References

[1]. Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.
[2]. 冯乐, 朱仁杰, 吴汉舟, 等. 神经网络水印综述[J]. 应用科学学报, 2021, 39(6): 881-892.
[3]. Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples[J]. arXiv preprint arXiv:1412.6572, 2014.
[4]. Dong Y, Liao F, Pang T, et al. Boosting adversarial attacks with momentum[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 9185-9193.
[5]. Xie C, Zhang Z, Zhou Y, et al. Improving transferability of adversarial examples with input diversity[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 2730-2739.
[6]. Lin J, Song C, He K, et al. Nesterov accelerated gradient and scale invariance for adversarial attacks[J]. in Proc. Int. Conf. Learn. Represent., 2019, pp. 112.
[7]. 鲍蕾, 陶蔚, 陶卿. 结合自适应步长策略和数据增强机制提升对抗攻击迁移性[J]. 电子学报, 2024, 52(1): 157-169.
[8]. Wang X, He K. Enhancing the transferability of adversarial attacks through variance tuning[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 1924-1933.
[9]. Wang G, Yan H; Wei X. Enhancing transferability of adversarial examples with spatial momentum. In Pattern Recognition and Computer Vision, 5th Chinese Conference, PRCV 2022, Shenzhen, China, 4–7 November 2022, Proceedings, Part I; Springer International Publishing: Cham, Switzerland, 2022; pp. 593–604.
[10]. Phan H, Xie Y, Liao S, et al. CAG: A real-time low-cost enhanced-robustness high-transferability content-aware adversarial attack generator[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(04): 5412-5419.
[11]. Wang T, Kerschbaum F. Riga: Covert and robust white-box watermarking of deep neural networks[C]. Proceedings of the web conference 2021. 2021: 993-1004.
[12]. Adi Y, Baum C, Cisse M, et al. Turning your weakness into a strength: Watermarking deep neural networks by backdooring[C]. 27th USENIX security symposium (USENIX Security 18). 2018: 1615-1631.
[13]. Le Merrer E, Perez P, Trédan G. Adversarial frontier stitching for remote neural network watermarking[J]. Neural Computing and Applications, 2020, 32(13): 9233-9244.
[14]. Chen H, Rouhani B D, Koushanfar F. Blackmarks: Blackbox multibit watermarking for deep neural networks[J]. arXiv preprint arXiv:1904.00344, 2019.