A Novel Framework for Digital Image Watermarking Based on Neural Network

Jia He
Lu'an Vocational Technical College,
Lu'an 237000, China
Jia_He2023@outlook.com

***Abstract*—There are many instances of intellectual property rights violations due to the common usage of digital data on the Internet, including unauthorised use, copying, and theft of digital content. Intellectual property rights of digital photos must be upheld, as they are very valuable materials. Digital watermarking is a more modern method to do this. By using a watermark (WM), the owner's information is included into the content, which may then be shared or saved. When required, this technology will retrieve the encoded WM information to prove ownership. Different technologies have been investigated and created on the basis of existing technologies, fields of use, etc. This paper proposes a novel approach to digital watermarking based on a neural network. First, the trigger data set and noise data set are generated from the binary encoding and random cutting of the original training samples. Then, the pattern with higher watermark trigger accuracy is obtained from the trigger set. Simulation results show that the proposed algorithm performs better in terms of accuracy and computing time cost compared to existing algorithms.**

***Index Terms*—Intellectual property rights violations; Digital watermarking; Neural network; Trigger set; Algorithm performance.**

I. INTRODUCTION

The creation and deployment of complicated models are made much simpler by the deep learning frameworks TensorFlow [1], Torch [2], Caffe [3], pretrained models AlexNet [4], and ResNet [5]. Additionally, developers may easily generate models by fine-tuning or transferring [6]. However, the cost of training deep neural network (DNN) models remains high, since it requires a lot of labelled data sets and computational power to modify the weights, hyperparameters, and structure of the model. This makes DNN models that are stolen lucrative. During the development stage, malware may cause models to leak, and during the deployment stage, remote application interface query assaults may lead to piracy. Due to this, safeguarding DNN models from unauthorised duplication, alteration, and misuse is a crucial concern that must be solved urgently.

Currently, most watermarking solutions achieve the purpose of marking DNN models by modifying the training set and letting the target model learn specific triggering patterns to facilitate ownership verification. This type of watermarking scheme has great shortcomings, because it focusses on a single target model of watermarking, and the watermark of each model is independent, ignoring the correlation between watermarks of multiple models. Owners face repeated embedding time overhead when adding the same copyright watermark to multiple of their models. In addition, if the target model is watermarked through retraining or fine-tuning, the embedding overhead will be positively correlated with the number and complexity of the model. On the other hand, the watermark DNN model is very different from previous watermark multimedia content. The main components of the DNN model are the layer structure and the weight parameters. Compared to multimedia content, the DNN model is less interpretable and more difficult to watermark. In view of the above situation, how to quickly and effectively watermark multiple models and enhance the reusability and migration of watermarks is an important issue in the current DNN model ownership verification research.

To overcome the above shortcomings, this paper proposes a LogoNet based multimodel for watermarking strategy (LNMMWS) based on logo network, which inserts the logo network into multiple models in a similar way as pasting trademarks to quickly watermark multiple models without causing repeated overhead.

The novelty and specific contributions of this paper are as follows.

1. For the watermark function, a streamlined structure LogoNet is designed. It has centralised functions, strong reusability, and can learn more watermark trigger modes in a relatively small amount of time.
2. Based on the output layer embedding method, LogoNet can repeatedly embed multiple target models to give them the watermark function. The watermark overhead is only generated once, fixed and low.
3. Anti-noise training is introduced to enhance LogoNet's ability to process invalid inputs, reduce the accuracy impact of LogoNet embedding on the target model, and improve the confidentiality of the LNMMWS watermark.

The rest of this paper is organised as follows. Section II introduces related work. Section III introduces background knowledge. Section IV introduces the specific details of the proposed algorithm. Section V provides the simulation

Manuscript received 5 February, 2024; accepted 30 April, 2024.
The work is supported by the Anhui Province College Teaching Innovation Team for Animation Production Technology (2023cxtd180), the key topic of the Anhui Provincial Education Commission "Traditional Culture in the Daba Mountain Region in Digital Management, Inheritance and Innovation", and the excellent young talent support plan of Anhui Province's higher education institutions (gxyq202108).

results. Section VI provides conclusions.

II. Related Work

In the black-box scenario, the output predicted by a given input is verified by the DNN watermarking system, which checks the watermark. This kind of plan is more realistic. The embedding and verification phases make up the two steps of the watermarking process. Owners may incorporate watermarks into their generated models at the embedding stage. If the model is stolen, the owner can remove the watermark from the suspicious model as proof of infringement during the verification stage. The design of the trigger pattern, or the construction of the trigger set, the embedding technique, and the verification mechanism are crucial to the watermarking process. However, most recent research on watermarks focusses on creating the trigger set and using it to train the target model in conjunction with the original data set or fine-tune it to incorporate the watermark [7].

The authors in [8] proposed a method for watermarking DNN models using backdoor methods. Their method can be seamlessly integrated with existing DNN models and is appropriate for a wide range of classification jobs. To watermark the target model, the authors in [9] suggested employing trigger samples with predetermined target categories that are overlaid with noise, irrelevant patterns, and watermark patterns. The authors in [10] proposed adding certain modifications information retained by the owner to a set of original samples to form a trigger set. This solution is suitable for embedded applications. They made a functional definition of this pixel-level modification in order to use the differential evolution algorithm to find the optimal modification [11]. To make the distribution of trigger samples and original samples more similar, the authors in [12] used a lightweight autoencoder to generate trigger samples and form trigger sets. The authors in [13] proposed to use a set of boundary data points found by the boundary decision algorithm to add specific perturbations to form a trigger set.

The trigger set constructed by the authors in [14] contains many different types of modifications to mark the target model more reliably. The authors in [15] proposed using an image tile transformation method based on a specific key to construct a trigger set to uniquely identify the target model. To enhance the robustness of the trigger set, the authors in [16] proposed using an image watermarking method based on the frequency domain to construct a data set. The generated trigger samples have strong confidentiality and robustness to signal processing. The authors in [17] proposed to write user fingerprint information in samples outside of the training set using the least significant bit image steganography technology to construct a trigger set.

The above watermarking method embeds the trigger pattern as a watermark into the target model. This watermark has nothing to do with the main task of the protected model, so it has less impact on the accuracy of the target model, but this makes it difficult to remove the watermark through model compression or model fine-tuning. To this end, the authors in [18] proposed associated watermarking, in which the watermark has a strong dependence on the normal weight of the model. If a pirate attempts to remove the watermark, the performance of the model in the normal data set will be significantly reduced. These watermarking methods based on DNN backdoors can be detected by some trigger pattern recognition methods, such as the neural cleanse method proposed by the authors in [19] and the strip method proposed by the authors in [20]. The above watermarking methods have not paid attention to the connection between multimodel watermarks. In a multimodel watermarking scenario, the watermarking overhead will increase infinitely as the number of target models increases, making fast watermarking impossible. Moreover, the reusability of watermarks is poor, and existing watermark work cannot be used directly for watermark embedding in the next target model. To this end, this paper proposes a novel neural network-based watermarking scheme with strong reusability, low time overhead, and high efficiency from the perspective of designing embedding methods.

III. Background Knowledge

A. DNN Model and DNN Backdoor

Without the requirement for extraction of human features, deep learning is a kind of machine learning system that automatically learns data representations hierarchically from training data [21], [22]. Deep neural networks, which are made up of several components of fundamental neural networks such as linear perceptrons, convolutional layers, and nonlinear activation functions, are the foundation of deep learning techniques [23]. Network units are trained to identify complicated ideas from structured input and are arranged into layers. High-level network layers are often linked to high-dimensional semantic elements, such as dogs and cats, but low-level network levels are typically related to low-dimensional features, such as corners and edges [24].

The formatted training sample $(x, y) \in \mathbb{D}^m$ is input into the DNN, and the prediction result $y' \in R^n$ is obtained through $F(x, y) = y'$ equation mapping, where the parameter equation $F(x, y)$ is determined by the level of the network structure and weight parameters of all neurons are determined [25]. The initial prediction result y' is not necessarily equal to the real target value y, so a large amount of training data must be used to train the DNN model. The DNN model will update the weight w based on the difference between the predicted value y' and the real target value y, and finally get a model with higher accuracy.

Both backdoor attacks and adversarial attacks can be used to harm the performance of DNN models, but backdoors can be used to verify the ownership of DNN models [26], [27]. Suppose that there is currently a classification task whose samples are $(x, y) \in \mathbb{D}_t^m$. In the setting of an adversarial attack, the attacker uses a minimal change $x^{\text{per}} = x + \delta(\|\delta\|_2 \to 0)$ to obtain an incorrect classification result $F(w, x^{\text{per}}) \neq y$. During this process, the parametric equations used for classification did not change. For backdoor attacks, the attacker will redefine a parametric equation $F^*(w^*, x)$ and poison the training set $\mathbb{D}_p^m$ which is randomly doped with trigger samples $[\beta(x), t(y)]$. The trigger sample is obtained by transforming the randomly selected original training sample through the $\beta(x)$ function [28], [29]. This specific function modification is called a trigger. Common modifications include adding Gaussian noise to the original sample, trigger patterns, etc., and the trigger set consists of

multiple trigger sample composition. After being trained by the poisoned training set $\mathbb{D}_p^m$, the DNN model will get the normal prediction category y for the original sample, and the specified prediction category $t(y)$ for the trigger sample, i.e., $F(w,x)=y, F^*\left(w^*,\beta(x)\right)=t(y)$. Backdoor attacks are highly stealthy and prespecified target categories will only trigger on samples with triggers [30]. This stealth and targeting make specific backdoors a workaround for DNN model ownership verification.

B. DNN Watermark

The concepts involved in the DNN watermarking process in the black-box scenario are explained below, as shown in Table I.

TABLE I. CONCEPT DESCRIPTION OF WATERMARK BASED ON NEURAL NETWORK.

Concept	Description
Target model	The DNN model that needs to embed the watermark is called the target model.
Trigger mode	The specific output learnt by the target model on a specific input is called a trigger pattern. The specific output is called a target category in the classification model, and the label corresponding to the target category is called a target label.
Fidelity	If a DNN model has a large accuracy gap in classification tasks before and after embedding watermarks, the fidelity will be poor, i.e., it should be required $\lvert F^*(w^*,x)-F(w,x)\rvert \le \xi$, ξ refers to the threshold.
Efficiency	The overhead of watermark embedding and extraction should be low. The watermark should be embedded in the target DNN model in the least costly way, i.e., by inserting a small number of neurons and adding necessary neuron connections.
Mobility	Watermarks should have strong portability, i.e., a watermark can be migrated to multiple models.
Stability	The watermark embedding method should be able to resist model modification attacks, such as model compression and model fine-tuning.
Secrecy	Watermarks embedded in the target model should not be detected by other detection methods and can only be verified by specific methods.

IV. ALGORITHM DESIGN

The proposed algorithm is divided into two stages, namely, the embedding stage and the verification stage, which includes three steps: LogoNet construction, embedding LogoNet with the target model, and ownership verification. The process is shown in Fig. 1, and the details are as follows.

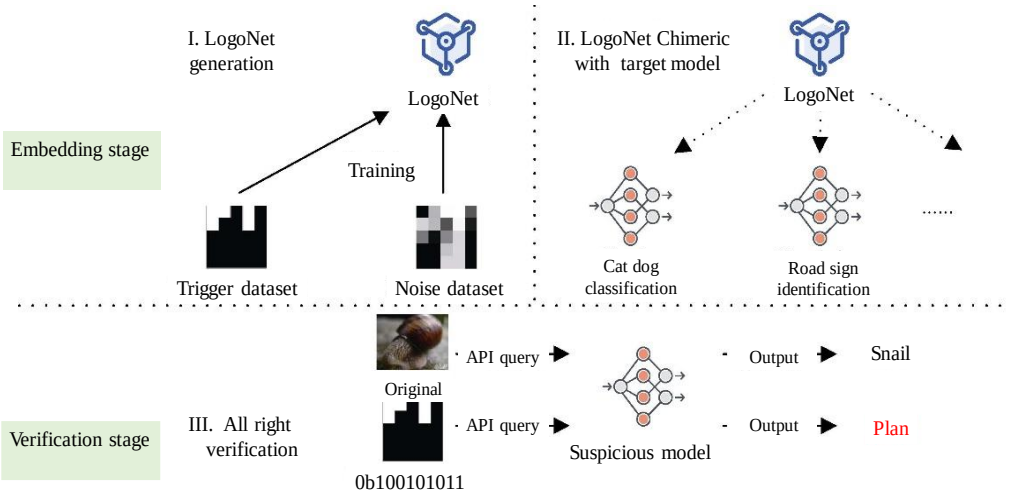


Fig. 1. Proposed framework.

Step 1: LogoNet build: Initialise the trigger set and noise set and train LogoNet so that LogoNet fits the trigger samples and has strong generalisability for noise samples [31]–[33].
Step 2: Fit LogoNet with the target model: Embed LogoNet into the target model, adjust the output layer of LogoNet according to the output layer of the target model, and embed the output data streams of the two.
Step 3: Ownership Verification: Verify according to the ownership verification method in the black-box scenario.

A. LogoNet Generation

1. Data set generation

The training set used by LogoNet includes a trigger set and a noise set. The samples in the trigger set are generated correspondingly by binary strings [34]–[36]. There are 2^{11} = 2048 types of 11 bit binary strings, which are represented by a 5×5 dot matrix. The number of trigger samples generated is 2048. The initial value of each pixel is 0. If the value of the corresponding binary bit of the pixel is 1, the value of the pixel is set to 255 [37], [38]. Each sample is then assigned to a separate category. You can also select other sizes of the dot matrix and select other numbers of pixels. To improve the stability of LogoNet and enhance its anti-noise ability, it is necessary to generate random noise samples and make these samples point to the only additional categories [39], [40], so the number of categories in the LogoNet training data set is 2049.

2. Hierarchy establishment

The structure of LogoNet is a small four-layer convolutional neural network, which includes three convolutional layers and one fully connected layer, and uses the rule activation function. The output dimension is 2049, where the first 2048 categories correspond to 2048 trigger samples, and the last category corresponds to additional noise samples [41]. If LogoNet only classifies 2048 triggers, LogoNet will be smaller, but LogoNet should be able to handle noisy input. However, this network is still very lean compared to most DNN networks. The number of parameters of LogoNet is only 0.0004 of the VGG-16 [6] model.

3. Training in the generated data set

Train LogoNet on the generated data set. The LogoNet training data set consists of two parts. The first part is 2048

trigger samples, and the second part is random noise samples. The noise samples come from random slices of the sample images from the target model training set. For these noisy inputs, LogoNet should remain silent, i.e., LogoNet predicts these noisy samples as specified extra categories 2049, and then the extra categories 2049 will be discarded when LogoNet is fitted to the target model. After this processing, noisy input will not predict valid categories, and this training method is called anti-noise training [42], [43]. The benefit of anti-noise training is that it reduces the irrelevant gradient flow to the neurons related to the LogoNet watermark trigger mode, which can reduce the false positives of LogoNet and reduce the impact of LogoNet on the accuracy of the target model. For example, for the LNMMWS watermark model of the MNIST data set, anti-noise training can reduce the impact of LogoNet on the accuracy of the target model by 18.83 %. As the number of training rounds increases, the learning rate can be gradually reduced to obtain better accuracy [44].

B. *Target Model Fitting*

Multiple target models can have the same hierarchical architecture and different classification tasks, or they may have different hierarchical architectures and different classification tasks. Therefore, the LogoNet embedding target model can be divided into three steps. First, use nearest neighbour interpolation to adjust the input sample size [45] and input it into LogoNet and the target model for calculation, respectively. Then, the LogoNet output layer is adjusted according to the output layer of the target model. Finally, the LogoNet output is fit with the target model output.

The input may be a direct trigger sample or a linear superposition of the trigger sample and the original sample. If the input is a trigger sample, the size of the trigger sample must be expanded by interpolation before being input to the target model [46], [47]. If the input is superimposed samples, the trigger samples need to be separated before being input to the LogoNet network.

The useful categories of LogoNet are 2048, so there are 2048 target categories for triggering samples [48]. However, in practical applications, the classification categories of the DNN model will be smaller than the classification categories of LogoNet, so the output dimensions of LogoNet must be adjusted according to the output dimensions of the target model. First, select a subset of categories from the classification categories of the target model as the set of targets. Then, for each target category, a trigger sample corresponding to it is selected [49], [50]. Finally, the LogoNet output categories corresponding to the selected trigger samples are retained, and other unused categories are discarded, i.e., the output vector is clipped.

The LogoNet output is then fitted to the output of the target model. Assume that the output of LogoNet after cropping is $F^*(w^*, \beta(x)) \in R^n$, and the output of the target model is $F(w, x) \in R^m$, where $n \le m$. For the LogoNet output vector [51], fill the missing values with 0 so that the output dimensions of both networks are equal to m, and finally embed the two output vectors into the final output vector $\vec{y} \in R^m$. The fitting process is equivalent to a switch that determines the proportion of the output of LogoNet and the target model in the final output. When the input and watermark trigger modes are related, the final result should be determined by F^*, in other cases, the final result should be determined by F [52], [53]. The fitting process can be performed as a weighted average, or directly, as shown in (1), giving different weights to F and F^*

$$\vec{y} = \theta F(w, x) + \lambda F^*\left(w^*, \beta(x)\right), \tag{1}$$

where the value of θ denotes the quantised value to embed the watermark bit in the angle, λ denotes the number of triggering samples (θ should be larger than λ because LogoNet has greater confidence than the target model), $F(w, x) \in R^m$ denotes the output of the target model, and $F^*(w^*, \beta(x)) \in R^n$ represents the output of LogoNet after cropping. The tasks faced by the target model are generally complex. Therefore, the structure of the target model is relatively complex, the accuracy will not be very high, and the confidence level will not be very high. Finally, the final output vector $\vec{y}$ is calculated through the *softmax* function to obtain the final probability distribution $\hat{y}$, which is as described in (2)

$$\hat{y} = softmax\left(\vec{y}\right), softmax\left(y_i\right) = \frac{e^{y_i}}{\sum_{c=1}^{C} e^{y_C}}, \tag{2}$$

where $\vec{y}$ is determined from (1), y_i and y_C refers to the output corresponding to i^{th} and c^{th} target samples. Finally, the target model that has been embedded in LogoNet implements the input stream execution process, as shown in Fig. 2, where *Operation* Θ corresponds to the chimeric processing of the output stream.

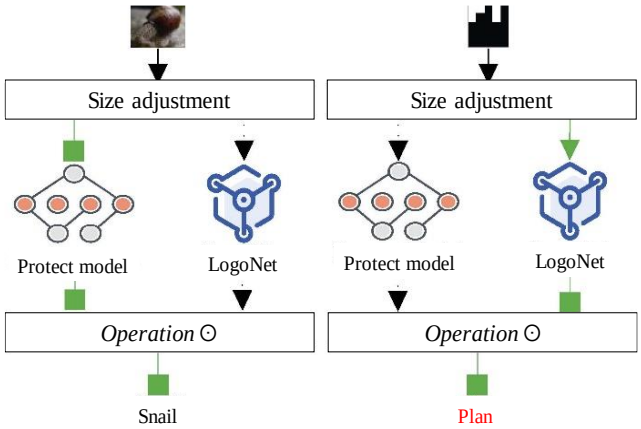


Fig. 2. Illustration of the input stream processing using the proposed watermark algorithm.

Using the above embedding method, LogoNet can be quickly embedded into any large model [54].

C. *Ownership Verification*

In the watermark verification stage, black-box verification is used and only needs to be verified through the remote application interface service. White-box verification requires knowing the parameters, structure, or data set of the DNN model, which is impractical in real situations [55].

This article follows a black-box scenario, a model owner O, who owns multiple DNN models $\{F_0, F_1, F_2, \ldots, F_n\}$ for multiple services $\{T_0, T_1, T_2, \ldots, T_n\}$, and a suspicious I, he starts from the model. A similar service $T_i', i \in [0, n]$ is

established in $F_i', i \in [0, n]$, and the two services have similar performance $F_i' \approx F_i$. In real situations, I can obtain the model $F_i, i \in [0, n]$ in many ways. For example, owner O may be attacked by an insider, causing the model to be leaked, or it may be maliciously stolen and sold on the darknet market, or resold by users [56]–[58]. How O obtains the model $F_i, i \in [0, n]$ is beyond the scope of this article.

This article will help owner O protect the intellectual property of model $F_i, i \in [0, n]$, $T_i, i \in [0, n]$. If the model $F_i', i \in [0, n]$ is equivalent to $F_i, i \in [0, n]$, and the watermark can be verified from $F_i', i \in [0, n]$, it can be confirmed that I is a pirate, $T_i', i \in [0, n]$ plagiarizes the service $T_i, i \in [0, n]$. Multiple sets of specific trigger samples are sent to the service $T_i', i \in [0, n]$. If the predicted category is a specific target category, the verification of the watermark is successful [59]–[61].

V. SIMULATION RESULTS

The experiment evaluates the proposed algorithm from three perspectives: effectiveness, stability, and confidentiality. For effectiveness, this paper proposes five indicators and conducts comparative evaluations with three other related documents in recent years; for stability, this paper uses two commonly used watermark attack methods for evaluation; for secrecy, this paper uses two commonly used watermark detection methods that are evaluated. The experiment was conducted on a machine equipped with AMD R5-5600H, 16 GB RAM and an Nvida RTX 3050 GPU. The experimental indicators and meanings of each section are shown in Table II, and the data sets of multiple target tasks used in the experiment are shown in Table III.

TABLE II. EXPERIMENTAL PARAMETERS.

Section	Symbol	Description
A	ACC_{be}	Evaluate the accuracy of unwatermarked models on classification tasks.
	ACC_{ne}	Evaluate the impact of embedded watermarks on the accuracy of the target model classification task, and its value is $ACC_{be} - ACC_{qf}$.
	N_r	Evaluate the number of watermark trigger patterns that can be embedded into the target model.
	TC	The time cost of a single round of training, in s.
A, *B*.1	ACC_{em}	Evaluate the accuracy of watermarking models on trigger samples.
B.1, *B*.2	ACC_{qf}	Evaluate the accuracy of watermarking models in classification tasks.
	ACC_{em}^{be}	Evaluate the accuracy of the watermark model in trigger samples before fine-tuning the model.
	ACC_{em}^{qf}	Evaluate the accuracy of the watermark model on trigger samples after fine-tuning the model.
B.2	N_r^{be}	Evaluate the number of watermark trigger patterns in the target model before fine-tuning the model.
	N_r^{qf}	Evaluate the number of watermark trigger patterns contained in the target model after model fine-tuning.
C.1	*AI*	The anomaly index evaluates the possibility that an unknown model contains a watermark.
C.2	*H*	Evaluate the randomness of the model predictions on adversarial examples.

TABLE III. DESCRIPTION OF DATA SETS.

Parameter	Data set			
	MNIST [62]	CIFAR10 [63]	GTSRB [64]	ImageNet [65]
Input dimension	28×28×1	32×32×3	32×32×3	224×224××3
Label	10	10	43	1000
Training set size	60000	50000	39209	1281167
Test set size	10000	10000	12630	100000

A. Effectiveness Evaluation

From the five aspects of ACC_{be}, ACC_{em}, ACC_{ne}, N_r, and $TC5$, the effectiveness of LNMMWS will be confirmed through experimental comparisons between the proposed algorithm and the algorithms proposed by the authors in [9], [16], and [17]. The experimental results are shown in Fig. 3. The baseline refers to the model in [9]. Discrete cosine transform (DCT) refers to the model in [16], and least significant bit (LSB) refers to the model in [17]. The model in [9] is not suitable for large-scale watermark models. After formatting, the relevant information of the trigger pattern will be lost, making it impossible to obtain a convergence model with normal accuracy. Therefore, for the ImageNet data set in Fig. 3, there is a lack of baseline-related experimental results.

The relevant experimental results of ACC_{be} are shown in Fig. 3(a). The ACC_{be} of each model corresponding to each data set is higher. This is because this experiment uses stronger performance model structures such as those in Table III to cope with different tasks. As can be seen from Fig. 3(a), the ACC_{be} of the same data set is slightly different. This is due to the different trigger set generation methods and data set loading methods of the proposed algorithm and the algorithms proposed by the authors [9], [16], and [17].

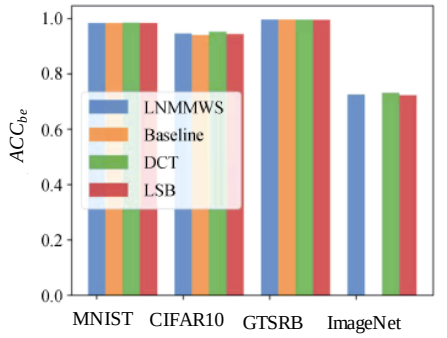
The related experiments of ACC_{em} are shown in Fig. 3(b). On the one hand, the ACC_{em} of each proposed watermark model can reach 100 %. Because first of all, before LogoNet is embedded in the target model, the accuracy of the trigger set has reached 99.9 %; secondly, during LogoNet embedding, the useless categories in the LogoNet output vector are clipped based on the output of the target model. On the other hand, the ACC_{em} of the baseline and DCT watermark models are relatively low because the trigger samples of the two are generated by specific modifications of the images selected in the training set, which makes the trigger set and the classification task data set have a certain correlation, while the proposed and LSB watermarking algorithms remove this correlation.

The related experiments of ACC_{ne} are shown in Fig. 3(c). The fidelity of the proposed algorithm is similar to that of the algorithms proposed by the authors in [9], [16], and [17]. The proposed algorithm reduces the ratio of LogoNet output vectors while satisfying ACC_{em} optimality. However, a smaller ACC_{ne} is obtained to ensure the fidelity of the proposed method.

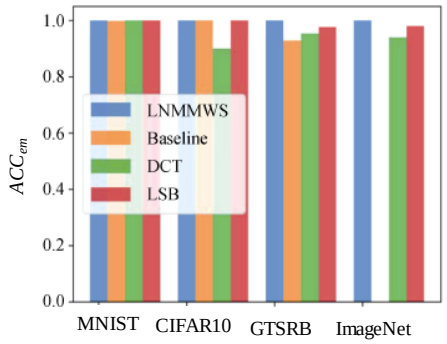
The N_r related experiments are shown in Fig. 3(d). The N_r of the proposed algorithm is determined by the classification category of the target model. The N_r of the baseline is determined by the proportion of samples selected from the training set. The smaller the proportion, the worse the ACC_{em}. The larger the proportion, the smaller the N_r. The N_r

of DCT and LSB is determined by the number of watermark trigger samples. Therefore, the proposed, DCT, and LSB algorithms have smaller limitations than the baseline embedded watermark trigger mode. In Fig. 3(d), the N_r of the proposed, DCT, and LSB algorithms is larger than that of the baseline. The normal model does not have many watermark trigger modes, and the larger N_r is, the more reliable the ownership verification is.

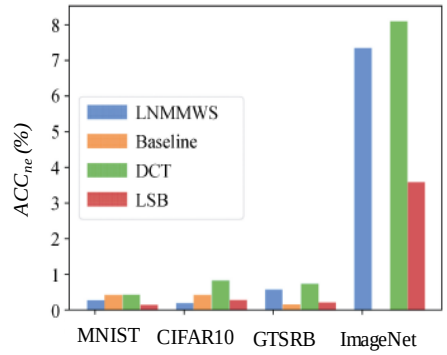
The experimental results in Fig. 3(a ~ d) confirm that the proposed algorithm meets the general requirements of DNN watermarking, but compared to existing solutions, has lower overhead.



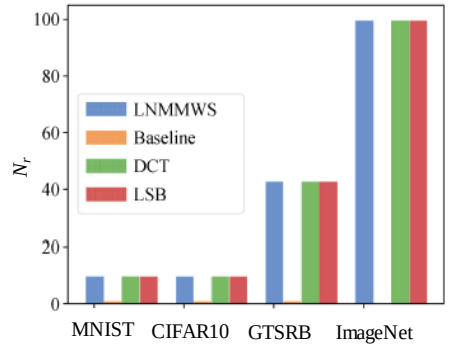
(a)



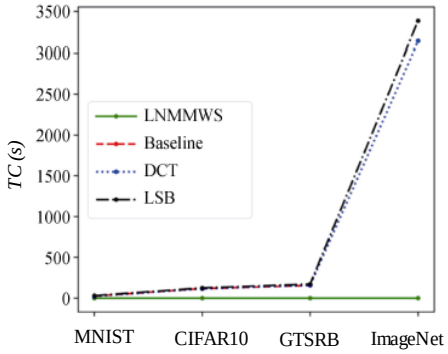
(b)



(c)



(d)



(e)

Fig. 3. Comparison of the performance of the proposed and existing algorithms: (a) ACC_{be}; (b) ACC_{em}; (c) ACC_{ne}; (d) N_r; (e) *TC*.

The relevant experiments of TC are shown in Fig. 3(e). As the size of the data set increases, the time overhead of the watermarking method in [9], [16], and [17] increases exponentially. When using the proposed LNMMWS algorithm, the time overhead is fixed and greatly reduced and does not increase with the increase in the size of the data set. Because in the proposed algorithm, the network size of LogoNet itself is small, which reduces the training overhead. Secondly, LogoNet has a streamlined structure and independent functions, allowing watermarks to be reused directly among multiple models. Thirdly, LogoNet learns a large number of watermark trigger modes, making it easier for LogoNet to be integrated with models of different structures.

B. Stability Evaluation

The watermark added to the target model by the proposed algorithm should have strong stability and be able to resist model compression attacks and model fine-tuning attacks. To this end, the proposed algorithm is evaluated from the above two aspects.

1. Evaluation under model compression attack

The DNN model contains a large number of parameters that are closely related to the performance of the DNN model. The purpose of model compression is to reduce redundant parameters without damaging the performance of the DNN model in its classification task [66]. Experiments evaluate the stability of the proposed watermark model in the face of model compression.

Model compression experiments were conducted on the proposed watermark model based on MNIST, CIFAR10, GBSTR, and ImageNet data sets, respectively. It can be seen from the experimental results in Fig. 4 that as the compression ratio increases, ACC_{af} will eventually be affected. There will not be a situation where ACC_{em} is very low but ACC_{af} remains unchanged, and ACC_{af} is more susceptible to impact than ACC_{em}. Because compared to identifying watermark trigger samples, the target model requires more parameters to handle the classification task.

2. Evaluation under model fine-tuning attack

Training a new model requires a large amount of data and computing resources. If pretrained models can be used, efficiency will be greatly improved. Pirates may use a small amount of new data with strong correlation or a large amount of weak correlation to fine-tune the stolen model to obtain a new model.

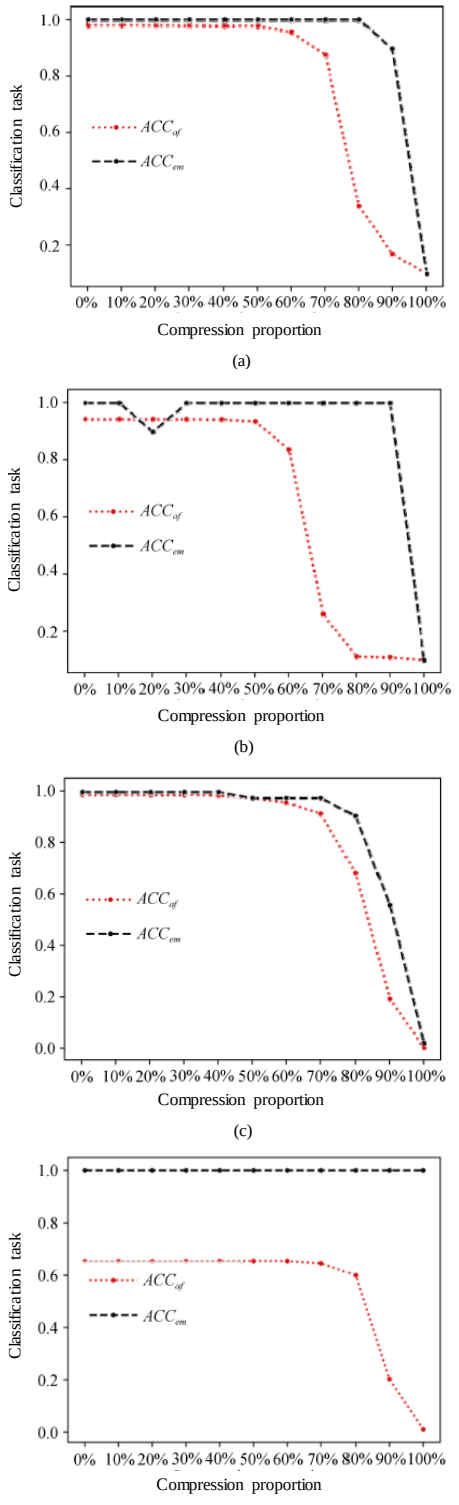


Fig. 4. Evaluation of the model compression attack under different data sets: (a) MNIST; (b) CIFAR10; (c) GTSRB; (d) ImageNet.

Experiments evaluate the stability of the proposed watermark model in the face of model fine-tuning.

In the original experiment of the authors in [9], half of the test sets in the MNIST and CIFAR10 data sets were used for fine-tuning training and testing of the respective models. This resulted in a reduction in the number of test set samples. There is a strong correlation between the original training set and the fine-tuning training set of the training model.

Setting up the experiment in this way can easily lead to overfitting, and the ACC_{af} 99.6 % and ACC_{em}^{af} 99.95 % obtained on the MNIST data set have no practical significance. In actual scenarios, it is not easy to obtain data that are highly correlated with the original training data. Therefore, the ImageNet data set target model of the proposed algorithm is fine-tuned and trained using CIFAR10 and CIFAR100. The CIFAR100 is similar to CIFAR10, but the CIFAR100 has more categories, with a total of 100 categories, 50,000 training images, and 10,000 test images.

The experimental results are shown in Table IV. It can be seen that ACC_{em}^{be} is higher than ACC_{af}, and ACC_{em}^{be} is less affected by model fine-tuning. Migrating from ImageNet to CIFAR10 data set, ACC_{em}^{be} remains unchanged. Migrating from ImageNet to CIFAR100 data set, N_r^{be} remains unchanged and ACC_{em}^{be} decreases by 2 %. The above experimental results are because the proposed algorithm has strong independence and weak correlation with the target model, and model fine-tuning has little impact on LogoNet.

TABLE IV. TUNING RESULTS EVALUATION.

Parameter	Data set (%)	
	CIFAR10	CIFAR100
ACC_{af}	**87.68**	67.83
ACC_{em}^{be}	100	100
ACC_{em}^{af}	**100**	98
N_r^{be}	100	100
N_r^{af}	10	**100**

C. Secrecy Evaluation

The watermark added to the target model by the proposed algorithm should be covert, i.e., it should not be detected by some trigger mode detection methods. If detected, it increases the risk that the watermark is removed. The detection method in [19], [20] is used to evaluate the stealthiness of the proposed scheme as follows.

1. Neural Cleanse method

The experiment uses the Neural Cleanse method [19] to detect whether the unknown DNN model contains watermarks. The Neural Cleanse method uses *AI* to assess whether the model is abnormal. The Neural Cleanse method sets the *AI* threshold to 2, that is, a model with an *AI* greater than 2 is considered an abnormal model, otherwise the model is considered normal.

Since the model in [9] was not obtained on the ImageNet data set, the anomaly index of the ImageNet [9] model was not tested, and the anomaly index of other models was not tested, as shown in Fig. 5. Clean refers to the clean model without adding watermark. It can be seen that, compared to the model in [9], the proposed algorithm is less easy to detect. The reason why the proposed scheme is more secretive is that LogoNet only responds to specific inputs, and through anti-noise training, LogoNet's anti-interference ability is improved.

Figure 6 is the trigger pattern generated by reverse engineering the Clean, proposed, and baseline models in the GTSRB data set using the Neural Cleanse method. Figure 6(a) is the trigger pattern embedded in the trigger sample of the baseline model, i.e., samples containing this pattern will be predicted as the target category, and the target category is designated as 7 in this model. Figure 6(c) is a reversely generated trigger pattern using the Neural Cleanse method for the baseline model. It can be seen that it is very different from

the trigger pattern reversely generated by the Clean model in Figure 6(b). For the baseline model, the watermark trigger pattern included in category 7 can be detected by the Neural Cleanse method and is identified as an anomaly category, which corresponds to the higher anomaly index of the baseline model of the GTSRB data set in Fig. 5. For the proposed model, the reversely generated trigger pattern for category 7 is shown in Fig. 6(d). It can be seen that it is very similar to Fig. 6(b). In fact, each tag of the proposed watermark model contains a specific watermark trigger mode and will not be detected by the Neural Cleanse method. The above experiments confirm that the proposed algorithm is more secretive than that proposed by the authors in [9].

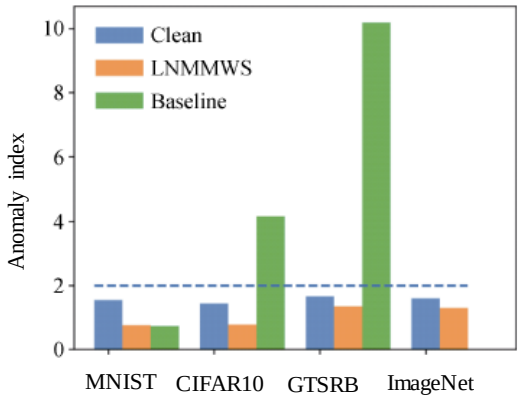


Fig. 5. Comparison of the anomaly index of the proposed and existing algorithms.

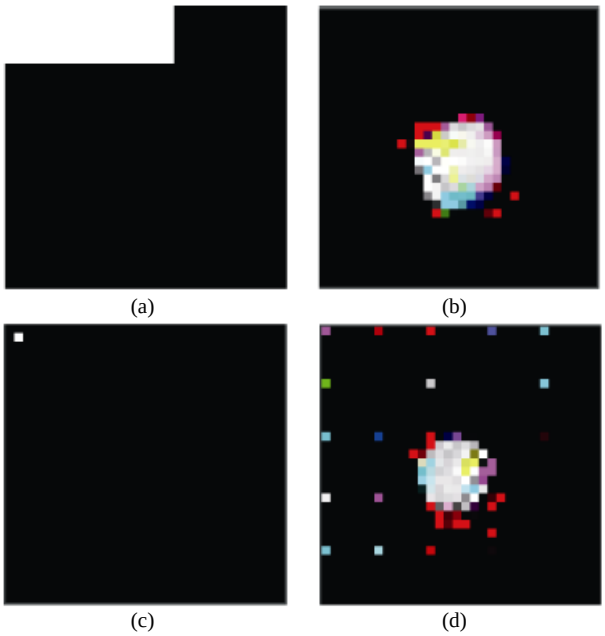


Fig. 6. Comparison of the trigger pattern of the proposed and existing algorithms: (a) Trigger; (b) Clean model; (c) Baseline model; (d) LNMMWS model.

2. Evaluation of Strip method

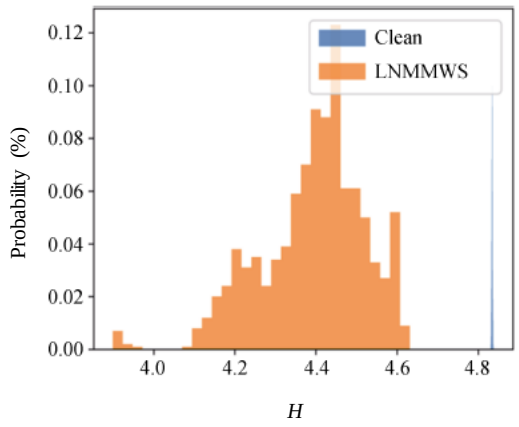
A model with good performance usually means that it has high accuracy on normal samples, but it will predict errors on adversarial samples and the errors are random. The Strip method [20] uses H to describe this randomness. Make adversarial examples and plot the distribution of each model's predictions H across multiple sets of predictions. The calculation formula for the prediction results of the M sample H is shown in (3)

$$H = -\sum_{i=1}^{M} y_i \times \log_2 y_i , \quad (3)$$

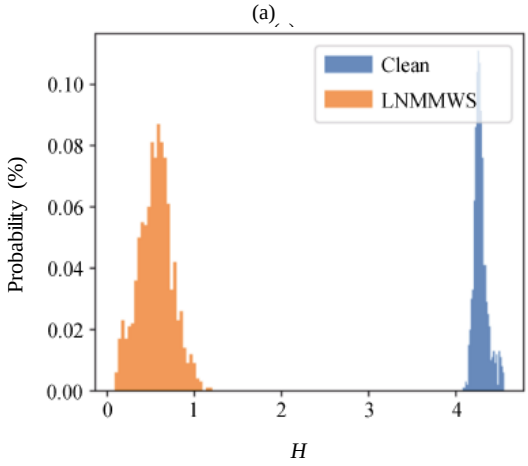
where y_i is the output corresponding to the i^{th} target sample.

As shown in Fig. 7, Clean refers to the non-watermarked model. The H distribution of different models for each task is different because the weight parameters of different models for the same task are different. But this will not cause the model that has been embedded in LogoNet to be detected because it is impossible for pirates to completely obtain all the information of the model before and after the LNMMWS watermark.

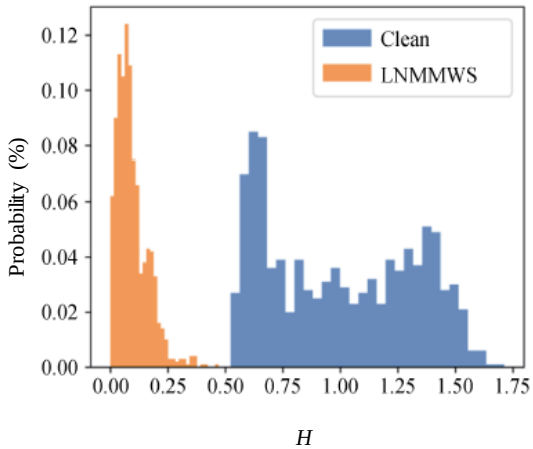
Figure 8 compares the performance of the proposed algorithm with other existing algorithms (i.e., [17], [65], [66]). As can be seen from Fig. 8, the anomaly index of all methods increases with different techniques. However, the proposed algorithm has the lowest detection index which proved its effectiveness.



(a)



(b)



(c)

(c)

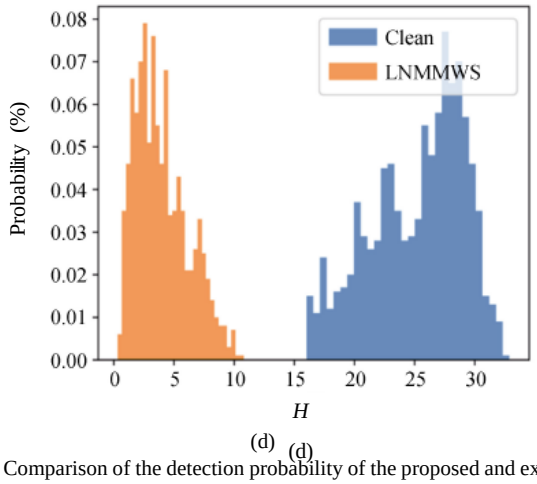


Fig. 7. Comparison of the detection probability of the proposed and existing algorithms: (a) MNIST; (b) CIFAR10; (c) GTSRB; (d) ImageNet.

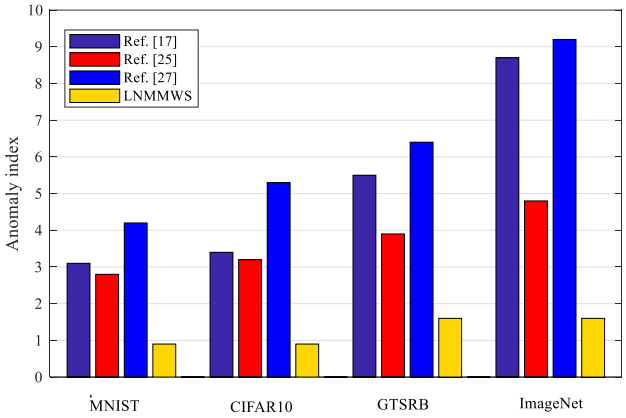


Fig. 8. Performance comparison of the proposed and other existing algorithms.

VI. CONCLUSIONS

Based on the multimodel watermarking scenario, this paper proposes a novel deep learning multimodel watermarking framework based on the logo network. After training on the generated trigger set and noise set, a LogoNet with higher watermark trigger pattern recognition accuracy and noise processing capabilities was obtained. The LogoNet is then embedded into multiple target models for watermark processing, and a black-box watermark verification method is used to achieve ownership verification. Experiments and analysis show that the proposed algorithm achieves better accuracy and lower time overhead in terms of effectiveness, stability, and confidentiality compared to existing algorithms. It can resist model compression attacks and model fine-tuning attacks and has better detection capabilities for certain trigger modes. The next research goal is to form a unified evaluation index on how to select watermarking algorithms with different target models and different embedding methods.

CONFLICTS OF INTEREST

The author declares that she has no conflicts of interest.

REFERENCES

[1] X. Zhou, J. Zhang, J. Wan, L. Zhou, Z. Wei, and J. Zhang, "Scheduling-efficient framework for neural network on heterogeneous distributed systems and mobile edge computing systems", *IEEE Access*, vol. 7, pp. 171853–171863, 2019. DOI: 10.1109/ACCESS.2019.2954897.

[2] S. Raschka, J. Patterson, and C. Nolet, "Machine learning in Python: Main developments and technology trends in data science, machine learning, and artificial intelligence", *Information*, vol. 11, no. 4, p. 193, 2020. DOI: 10.3390/info11040193.

[3] S.-H. Lim, S.-H. Kang, B.-H. Ko, J. Roh, C. Lim, and S.-Y. Cho, "An integrated analysis framework of convolutional neural network for embedded edge devices", *Electronics*, vol. 11, no. 7, p. 1041, 2022. DOI: 10.3390/electronics11071041.

[4] D. Zhang and T. Zhou, "Deep convolutional neural network using transfer learning for fault diagnosis", *IEEE Access*, vol. 9, pp. 43889–43897, 2021. DOI: 10.1109/ACCESS.2021.3061530.

[5] G. Singh, A. Mittal, and N. Aggarwal, "ResDNN: Deep residual learning for natural image denoising", *IET Image Processing*, vol. 14, no. 11, pp. 2425–2434, 2020. DOI: 10.1049/iet-ipr.2019.0623.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017. DOI: 10.1145/3065386.

[7] 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*, vol. 44, no. 10, pp. 6122–6139, 2022. DOI: 10.1109/TPAMI.2021.3088846.

[8] W. Aiken, H. Kim, S. Woo, and J. Ryoo, "Neural network laundering: Removing black-box backdoor watermarks from deep neural networks", *Computers & Security*, vol. 106, art. 102277, pp. 1–14, 2021. DOI: 10.1016/j.cose.2021.102277.

[9] Y. Wang and H. Wu, "Protecting the intellectual property of speaker recognition model by black-box watermarking in the frequency domain", *Symmetry*, vol. 14, no. 3, p. 619, 2022. DOI: 10.3390/sym14030619.

[10] R. Zhu, X. Zhang, M. Shi, and Z. Tang, "Secure neural network watermarking protocol against forging attack", *EURASIP Journal on Image and Video Processing*, art. no. 37, pp. 1–12, 2020. DOI: 10.1186/s13640-020-00527-1.

[11] L. Li, Y. Bai, C.-C. Chang, Y. Fan, W. Gu, and M. Emam, "Anti-pruning multi-watermarking for ownership proof of steganographic autoencoders", *Journal of Information Security and Applications*, vol. 76, art. 103548, pp. 1–11, 2023. DOI: 10.1016/j.jisa.2023.103548.

[12] J. Zhang, L. Dai, L. Xu, J. Ma, and X. Zhou, "Black-box watermarking and blockchain for IP protection of voiceprint recognition model", *Electronics*, vol. 12, no. 17, p. 3697, 2023. DOI: 10.3390/electronics12173697.

[13] E. Le Merrer, P. Perez, and G. Tredan, "Adversarial frontier stitching for remote neural network watermarking", *Neural Computing and Applications*, vol. 32, no. 13, pp. 9233–9244, 2020. DOI: 10.1007/s00521-019-04434-z.

[14] M. Xue, S. Sun, C. He, D. Gu, Y. Zhang, J. Wang, and W. Liu, "ActiveGuard: An active intellectual property protection technique for deep neural networks by leveraging adversarial examples as users' fingerprints", *IET Computers & Digital Techniques*, vol. 17, nos. 3–4, pp. 111–126, 2023. DOI: 10.1049/cdt2.12056.

[15] T. Qiao, Y. Ma, N. Zheng, H. Wu, Y. Chen, M. Xu, and X. Luo, "A novel model watermarking for protecting generative adversarial network", *Computers & Security*, vol. 127, art. 103102, pp. 511–523, 2023. DOI: 10.1016/j.cose.2023.103102.

[16] M. Li, Z. Wang, and X. Zhang, "An effective framework for intellectual property protection of NLG models", *Symmetry*, vol. 15, no. 6, p. 1287, 2023. DOI: 10.3390/sym15061287.

[17] M. Xue, S. Sun, Y. Zhang, J. Wang, and W. Liu, "Active intellectual property protection for deep neural networks through stealthy backdoor and users' identities authentication", *Applied Intelligence*, vol. 52, pp. 16497–16511, 2022. DOI: 10.1007/s10489-022-03339-0.

[18] T. Zhang, H. Wu, X. Lu, G. Han, and G. Sun, "AWEncoder: Adversarial watermarking pre-trained encoders in contrasive learning", *Applied Sciences*, vol. 13, no. 6, p. 3531, 2023. DOI: 10.3390/app13063531.

[19] M. Xue, Y. Wu, Z. Wu, Y. Zhang, J. Wang, and W. Liu, "Detecting backdoor in deep neural networks via intentional adversarial perturbations", *Information Sciences*, vol. 634, pp. 564–577, 2023. DOI: 10.1016/j.ins.2023.03.112.

[20] A. Mercier, N. Smolin, O. Sihlovec, S. Koffas, and S. Picek, "Backdoor Pony: Evaluation backdoor attacks and defences in different domains", *SoftwareX*, vol. 22, art. 101387, 2023. DOI: 10.1016/j.softx.2023.101387.

[21] S. Li, J. Chen, W. Peng, X. Shi, and W. Bu, "A vehicle detection method based on disparity segmentation", *Multimedia Tools and Applications*, vol. 82, no. 13, pp. 19643–19655, 2023. DOI: 10.1007/s11042-023-14360-x.

[22] J. Chen, Y. Song, D. Li, X. Lin, S. Zhou, and W. Xu, "Specular removal of industrial metal objects without changing lighting configuration", *IEEE Transactions on Industrial Informatics*, vol. 20, no. 3, pp. 3144–3153, 2024. DOI: 10.1109/TII.2023.3297613.

[23] H. Xu, Q. Li, and J. Chen, "Highlight removal from a single grayscale image using attentive GAN", *Applied Artificial Intelligence*, vol. 36, no. 1, art. 1988441, pp. 1–19, 2022. DOI: 10.1080/08839514.2021.1988441.

[24] Y. Yin, Y. Guo, Q. Su, and Z. Wang, "Task allocation of multiple unmanned aerial vehicles based on deep transfer reinforcement learning", *Drones*, vol. 6, no. 8, p. 215, 2022. DOI: 10.3390/drones6080215.

[25] J. Li, L. Han, C. Zhang, Q. Li, and Z. Liu, "Spherical convolution empowered viewport prediction in 360 video multicast with limited FoV feedback", *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 19, no. 1, art. no. 3, pp. 1–23, 2023. DOI: 10.1145/3511603.

[26] J. Li, C. Zhang, Z. Liu, R. Hong, and H. Hu, "Optimal volumetric video streaming with hybrid saliency based tiling", *IEEE Transactions on Multimedia*, vol. 25, pp. 2939–2953, 2023. DOI: 10.1109/TMM.2022.3153208.

[27] G. Xu, Q. Zhang, Z. Song, and B. Ai, "Relay-assisted deep space optical communication system over coronal fading channels", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 6, pp. 8297–8312, 2023. DOI: 10.1109/TAES.2023.3301463.

[28] H. Liu, Y. Xu, and F. Chen, "Sketch2Photo: Synthesizing photo-realistic images from sketches via global contexts", *Engineering Applications of Artificial Intelligence*, vol. 117, part A, art. 105608, 2023. DOI: 10.1016/j.engappai.2022.105608.

[29] H. Guan *et al.*, "Improved Gaussian mixture model to map the flooded crops of VV and VH polarization data", *Remote Sensing of Environment*, vol. 295, art. 113714, 2023. DOI: 10.1016/j.rse.2023.113714.

[30] H. Huang *et al.*, "The improved winter wheat yield estimation by assimilating GLASS LAI into a crop growth model with the proposed Bayesian posterior-based ensemble Kalman filter", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, art no. 4401818, pp. 1–18, 2023. DOI: 10.1109/TGRS.2023.3259742.

[31] H. Zhu *et al.*, "Graph structure enhanced pre-training language model for knowledge graph completion", *IEEE Transactions on Emerging Topics in Computational Intelligence*, pp. 1–12, 2024. DOI: 10.1109/TETCI.2024.3372442.

[32] D. Cheng, L. Chen, C. Lv, L. Guo, and Q. Kou, "Light-guided and cross-fusion U-Net for anti-illumination image super-resolution", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 12, pp. 8436–8449, 2022. DOI: 10.1109/TCSVT.2022.3194169.

[33] S. Pan, G. J. W. Xu, K. Guo, S. H. Park, and H. Ding, "Video-based engagement estimation of game streamers: An interpretable multimodal neural network approach", *IEEE Transactions on Games*, pp. 1–16, 2023. DOI: 10.1109/TG.2023.3348230.

[34] H. Sheng, S. Wang, D. Yang, R. Cong, Z. Cui, and R. Chen, "Cross-view recurrence-based self-supervised super-resolution of light field", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 12, pp. 7252–7266, 2023. DOI: 10.1109/TCSVT.2023.3278462.

[35] R. Cong, H. Sheng, D. Yang, Z. Cui, and R. Chen, "Exploiting spatial and angular correlations with deep efficient transformers for light field image super-resolution", *IEEE Transactions on Multimedia*, vol. 26, pp. 1421–1435, 2024. DOI: 10.1109/TMM.2023.3282465.

[36] Z. Cui, H. Sheng, D. Yang, S. Wang, R. Chen, and W. Ke, "Light field depth estimation for non-Lambertian objects via adaptive cross operator", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 2, pp. 1199–1211, 2024. DOI: 10.1109/TCSVT.2023.3292884.

[37] J. Xing, H. Yuan, R. Hamzaoui, H. Liu, and J. Hou, "GQE-Net: A graph-based quality enhancement network for point cloud color attribute", *IEEE Transactions on Image Processing*, vol. 32, pp. 6303–6317, 2023. DOI: 10.1109/TIP.2023.3330086.

[38] C. Fu, H. Yuan, H. Xu, H. Zhang, and L. Shen, "TMSO-Net: Texture adaptive multi-scale observation for light field image depth estimation", *Journal of Visual Communication and Image Representation*, vol. 90, art. 103731, pp. 1–12, 2023. DOI: 10.1016/j.jvcir.2022.103731.

[39] H. Liu, H. Yuan, Q. Liu, J. Hou, H. Zeng, and S. Kwong, "A hybrid compression framework for color attributes of static 3D point clouds", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 3, pp. 1564–1577, 2022. DOI: 10.1109/TCSVT.2021.3069838.

[40] T. Guo, H. Yuan, L. Wang, and T. Wang, "Rate-distortion optimized for geometry-based point cloud compression", *Journal of Electronic Imaging*, vol. 32, art. ID 13047, pp. 1–12, 2023. DOI: 10.1117/1.JEI.32.1.013047.

[41] S. Ma *et al.*, "The autonomous pipeline navigation of a cockroach bio-robot with enhanced walking stimuli", *Cybord and Bionic Systems*, vol. 4, art. ID 0067, pp. 1–9, 2023. DOI: 10.34133/cbsystems.0067.

[42] K. Uesugi, H. Mayama, and K. Morishima, "Analysis of rowing force of the water strider middle leg by direct measurement using a bio-appropriating probe and by indirect measurement using image analysis", *Cyborg and Bionic Systems*, vol. 4, art. ID 0061, pp. 1–13, 2023. DOI: 10.34133/cbsystems.0061.

[43] W. Ren, N. Jin, and L. OuYang, "Phase space graph convolutional network for chaotic time series learning", *IEEE Transactions on Industrial Informatics*, vol. 20, no. 5, pp. 7576–7584, 2024. DOI: 10.1109/TII.2024.3363089.

[44] Z. Qu, X. Liu, and M. Zheng, "Temporal-spatial quantum graph convolutional neural network based on Schrödinger approach for traffic congestion prediction", *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8677–8686, 2023. DOI: 10.1109/TITS.2022.3203791.

[45] J. Xu, G. Zhou, S. Su, Q. Cao, and Z. Tian, "The development of a rigorous model for bathymetric mapping from multispectral satellite-images", *Remote Sensing*, vol. 14, no. 10, p. 2495, 2022. DOI: 10.3390/rs14102495.

[46] G. Zhou and X. Liu, "Orthorectification model for extra-length linear array imagery", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, art no. 4709710, pp. 1–10, 2022. DOI: 10.1109/TGRS.2022.3223911.

[47] G. Zhou, H. Li, R. Song, Q. Wang, J. Xu, and B. Song, "Orthorectification of fisheye image under equidistant projection model", *Remote Sensing*, vol. 14, no. 17, p. 4175, 2022. DOI: 10.3390/rs14174175.

[48] G. Zhou *et al.*, "Shadow detection on high-resolution digital orthophoto map using semantic matching", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–20, 2023. DOI: 10.1109/TGRS.2023.3294531.

[49] Z. Wu, H. Zhu, L. He, Q. Zhao, J. Shi, and W. Wu, "Real-time stereo matching with high accuracy via spatial attention-guided upsampling", *Applied Intelligence*, vol. 53, no. 20, pp. 24253–24274, 2023. DOI: 10.1007/s10489-023-04646-w.

[50] J. J. Peng, X. G. Chen, X. K. Wang, J. Q. Wang, Q. Q. Long, and L. J. Yin, "Picture fuzzy decision-making theories and methodologies: A systematic review", *International Journal of Systems Science*, vol. 54, no. 13, pp. 2663–2675, 2023. DOI: 10.1080/00207721.2023.2241961.

[51] X. Bai, Y. He, and M. Xu, "Low-thrust reconfiguration strategy and optimization for formation flying using Jordan normal form", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 5, pp. 3279–3295, 2021. DOI: 10.1109/TAES.2021.3074204.

[52] Z. Fang, J. Liang, C. Tan, Q. Tian, D. Pi, and G. Yin, "Enhancing robust driver assistance control in distributed drive electric vehicles through integrated AFS and DYC technology", *IEEE Transactions on Intelligent Vehicles*, pp. 1–14, 2024. DOI: 10.1109/TIV.2024.3368050.

[53] J. Liang, J. Feng, Y. Lu, G. Yin, W. Zhuang, and X. Mao, "A direct yaw moment control framework through robust T-S fuzzy approach considering vehicle stability margin", *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 1, pp. 166–178, 2024. DOI: 10.1109/TMECH.2023.3274689.

[54] J. Liang *et al.*, "ETS-based human-machine robust shared control design considering the network delays", *IEEE Transactions on Automation Science and Engineering*, pp. 1–11, 2024. DOI: 10.1109/TASE.2024.3383094.

[55] X. Fu and M. Ren, "Sustainable and low-AoI cooperative data acquisition in UAV-aided sensor networks", *IEEE Sensors Journal*, vol. 24, no. 6, pp. 9016–9031, 2024. DOI: 10.1109/JSEN.2024.3355161.

[56] W. Zheng, S. Lu, Y. Yang, Z. Yin, and L. Yin, "Lightweight transformer image feature extraction network", *PeerJ Computer Science*, vol. 10, p. e1755, 2024. DOI: 10.7717/peerj-cs.1755.

[57] Y. Ban *et al.*, "Micro-directional propagation method based on user clustering", *Computing and Informatics*, vol. 42, no. 6, pp. 1445–1470, 2024. DOI: 10.31577/cai_2023_6_1445.

[58] S. Wang, H. Sheng, D. Yang, Y. Zhang, Y. Wu, and S. Wang, "Extendable multiple nodes recurrent tracking framework with RTU++", *IEEE Transactions on Image Processing*, vol. 31, pp. 5257–5271, 2022. DOI: 10.1109/TIP.2022.3192706.

[59] J. Shen, H. Sheng, S. Wang, R. Cong, D. Yang, and Y. Zhang, "Blockchain-based distributed multi-agent reinforcement learning for collaborative multi-object tracking framework", *IEEE Transactions on Computers*, vol. 73, no. 3, pp. 778–788, 2024. DOI: 10.1109/TC.2023.3343102.

[60] D. Yang *et al.*, "An occlusion and noise-aware stereo framework based on light field imaging for robust disparity estimation", *IEEE Transactions on Computers*, vol. 73, no. 3, pp. 764–777, 2024. DOI: 10.1109/TC.2023.3343098.

[61] B. Cao, J. Zhao, Z. Lv, Y. Gu, P. Yang, and S. K. Halgamuge, "Multiobjective evolution of fuzzy rough neural network via

distributed parallelism for stock prediction", *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 5, pp. 939–952, 2020. DOI: 10.1109/TFUZZ.2020.2972207.

[62] Y. Tian, Y. Zhang, and H. Zhang, "Recent advances in stochastic gradient descent in deep learning", *Mathematics*, vol. 11, no. 3, p. 682, 2023. DOI: 10.3390/math11030682.

[63] X. Yan, S. Z. Gilani, H. Qin, and A. Mian, "Structural similarity loss for learning to fuse multi-focus images", *Sensors*, vol. 20, no. 22, p. 6647, 2020. DOI: 10.3390/s20226647.

[64] P. D. Dizji, S. Joudaki, and H. Kolivand, "A new traffic sign recognition technique taking shuffled frog-leaping algorithm into account", *Wireless Personal Communications*, vol. 125, pp. 3425–3441, 2022. DOI: 10.1007/s11277-022-09718-7.

[65] Y. Guo, Y. Liu, E. M. Bakker, Y. Guo, and M. S. Lew, "CNN-RNN: A large-scale hierarchical image classification framework", *Multimedia Tools and Applications*, vol. 77, pp. 10251–10271, 2018. DOI: 10.1007/s11042-017-5443-x.

[66] Y. Zhang, G. Wang, T. Yang, T. Pang, Z. He, and J. Lv, "Compression of deep neural networks: Bridging the gap between conventional-based pruning and evolutionary approach", *Neural Computing and Applications*, vol. 34, pp. 16493–16514, 2022. DOI: 10.1007/s00521-022-07161-0.