International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18 (2026) pp. 1-17

https://doi.org/10.70917/ijcisim-2026-0100
Article

Analysis of Compliance Issues of Machine Learning Technology in Intellectual Property Protection and Its Legal Regulation

Zhiqiang Song *

Department of Economics and Management, Shanxi Institute of Technology, Yangquan, Shanxi, 045000, China; szqiang19790829@126.com

Abstract: The deepening application of machine learning technology in intellectual property protection makes its compliance gradually become a research hotspot in related fields. This paper uses deep neural network technology to design a digital watermark generation method for intellectual property. And based on the deep neural network watermarking method, the overall system framework of serial number watermarking is constructed. The framework maintains structural features consistent with the neural network output layer through specific matrix operations, thus effectively integrating serial number and deep learning technology. Meanwhile, in order to strengthen the compliance of the deep learning technique, in the generation process of key sample watermarking, the encoder is used to receive normal samples and exclusive signs and output key samples, and the discriminator is utilized to receive normal samples and key samples. As a result, the construction of the digital watermark generation model based on deep neural network is completed. The F1 value of the model is always maintained at 0.7 and above at different Hamming distances, and the maximum is up to 0.973. It shows extremely superior operational performance, which provides a good foundation for its adaptation in intellectual property protection.

Keywords: deep neural network; intellectual property protection; digital watermark generation; machine learning

1. Introduction

With the rapid development of the artificial intelligence technology industry, issues related to intellectual property protection have also become increasingly prevalent. How to appropriately protect intellectual property rights for artificial intelligence-related technology products has become both a hot topic and a significant challenge [1-2]. Machine learning has emerged as a key focus in recent technological advancements. At its core, it is a type of computer program that creates and deploys models capable of being trained using additional data and continuously improving their task execution capabilities [3-4]. Machine learning algorithms play an important role in promoting intellectual property management, intellectual property retrieval, and preventing intellectual property infringement risks by dynamically analyzing compliance issues in intellectual property protection through steps such as big data learning and training [5-7].

In recent years, research on intellectual property protection applications based on artificial intelligence technology has covered multiple areas, including intellectual property value assessment, semantic analysis, and intelligent classification, achieving good results and effectively enhancing the level of intellectual property protection. Literature [8] points out that by leveraging unsupervised machine learning technology, the originality of intellectual property assets and other objects can be estimated based on calculated distances, effectively assessing the effectiveness of protection for intellectual property types such as emojis, font designs, paintings, and novel titles. Literature [9] employs unsupervised machine learning methods to form legal document clusters and conduct enhanced analysis of legal entities and corresponding key terms. The results show that intelligent methods enhance the ability to adjudicate infringement cases, thereby better protecting brand assets. Literature [10] proposes an enhanced semantic retrieval system using the synonym database of the Microsoft Word application as

the semantic retrieval database. It classifies trademark infringement through feature extraction and machine learning algorithms and validates the effectiveness of the methods used through actual cases. Literature [11] analyzes the methods of deep neural networks for intellectual property protection from three aspects: model modification, evasion attacks, and active attacks. It designs a system evaluation method for deep neural network intellectual property protection and explores the challenges faced by deep neural networks in intellectual property protection. Artificial intelligence technology has great application value in intellectual property protection. A new mechanism for intellectual property protection driven by artificial intelligence should be established to enhance the effectiveness of intellectual property protection [12-13].

In this paper, we first construct the design scheme of white-box watermarking and black-box watermarking based on deep learning technology, as well as the operation process. Secondly, it proposes a digital watermarking framework for serial numbers, describing its overall workflow and three important components. Combined with the deep neural network watermarking generation method, the digital watermarking generation model based on deep neural network is constructed, and the representation and setting of the objective function of the two core modules (encoder and discriminator) of the digital watermarking framework are explained in detail. The robustness of the proposed model is again verified by comparing the performance of similar models for image processing tasks, as well as ablation experiments. Run performance experiments are designed to evaluate the overall performance of the proposed model. Finally, we briefly summarize the compliance performance of machine learning technology-assisted protection of intellectual property rights in Region G from 2013 to 2023, and make recommendations for relevant legal regulation.

2. Deep neural network based digital watermarking generation model

2.1. Deep Neural Network Watermarking

In view of the great success of digital watermarking in the protection of intellectual property in the field of multimedia, so when the intellectual property rights of deep learning models need technical means to protect, researchers have transplanted the watermarking technology to deep learning, and at present, according to the different ways of working, DNN watermarking is mainly divided into two kinds of white-box watermarking and black-box watermarking. The principles of these two types of DNN watermarking are described in detail below. The deep neural network white-box watermarking scheme is shown in Figure 1.

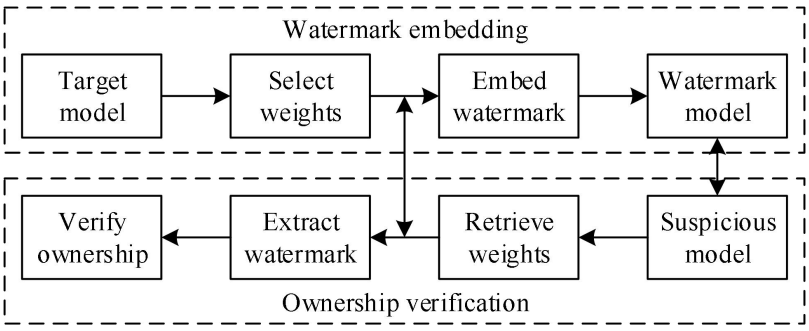


Figure 1. White-box watermarking scheme for deep neural networks.

White-box watermarking: white-box watermarking views the DNN model as a white box, i.e., the model owner needs access to view the internal structure of the model and each weight parameter. The model owner usually interprets the watermark as a multi-bit string containing only 0s and 1s or other information, as shown in Fig. 1. In order to embed the watermark into the model, the model owner will do some special treatments inside the model, such as applying statistical bias on certain weights inside the model, treating the probability density function of the activation maps, and so on. When verifying the ownership of the model, the model owner then needs to access the internal structure of the model and extract the embedded watermark information from the weight parameters to complete the ownership verification.

Black-box watermarking: Contrary to white-box watermarking, black-box watermarking treats the target model as a black box, i.e., it does not allow the model owner to access the internal structure of the model. The model owner can only complete the model ownership verification by suspecting the output of the model. The adversary usually deploys the stolen model on a remote server, and the model owner can

obtain the model output data through remote query, but cannot access the internal parameters of the model. Compared with white-box watermarking, black-box watermarking is more in line with the reality, and only needs to get the model output to verify the model ownership.

When training or fine-tuning the model, the model owner usually inputs a set composed of special samples and specified labels into the model for training, which is called a "trigger set" or "key sample", which is usually modified from the original training set, such as superimposing other content on the image, replacing words in text sentences, etc. Then, assign the wrong training label to these special samples, such as an image with the category "cat", and assign its label to "dog" to train the model, similar to data poisoning or neural network backdoor techniques. When a suspicious model is found, the model owner only needs to use the trigger set samples to query the suspicious model, and if the prediction accuracy of the trigger set is higher than a certain threshold, the model is judged to be a model with a watermark embedded, otherwise it is judged to be a model without a watermark. If (x, y) is used to represent clean samples and labels, and (x', y') is used to represent trigger set samples and their labels, when the query result of the model satisfies equation (1), the model is considered to be a watermark model. f represents the model, n represents the number of samples in the trigger set, ts represents the threshold, and $Count$ represents the total number of calculations.

$$\frac{Count\left(f(x_i') = y_i' \middle| {}^{n}_{0}\right)}{n} \geq ts \tag{1}$$

2.2. Overall framework for serial number watermarking

This section formalizes a digital watermarking framework for deep learning models based on sequence numbers, which embeds sequence numbers into DNNs and verifies their ownership by extracting them from remote DNNs. The sequence number embedding process is as follows: firstly, a predefined sequence number is assigned to a specific image (trigger T) and the corresponding predicted label is recorded, and then the trigger image with the corresponding sequence number is used as the training data to train the input DNN. The DNN automatically learns and remembers the triggers and the predefined sequence number patterns, and when a trigger is observed in a query, only the watermark-protected model can output the predefined labels and the output layer of the model is a predefined sequence number. The workflow of sequence number watermarking for deep learning models is shown in Figure 2.

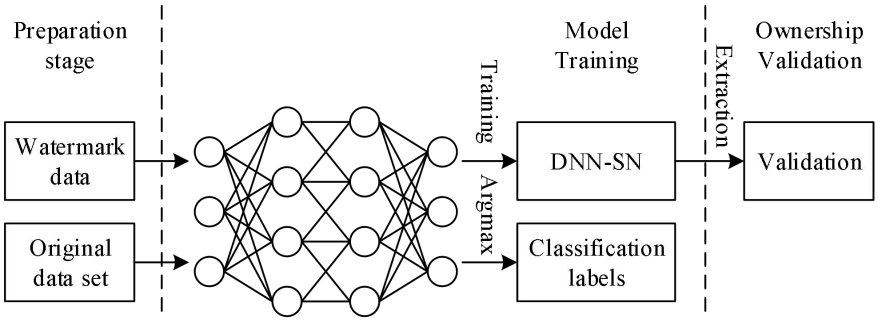


Figure 2. The workflow of serial number watermarking for deep learning models.

The complete process of the framework can be divided into three steps: first the preparation phase, the main work in this phase includes the preparation of the original dataset and the preparation of the trigger set, where the trigger set includes a specific trigger image and a predefined sequence number. After preparing the original training dataset and trigger set, the framework will realize the original function and watermark embedding function by cross-training in the model training phase. After completing the training, the protected models can be verified for ownership. Once they are stolen and deployed as an AI service, the owner can verify the model ownership by sending triggers and by checking the output of the verification service.

The key point to emphasize is that, unlike typical classification models, the serial number watermarking framework for deep learning models must perform two tasks: the original classification task and the embedded watermarking task. For the three main components in the framework, a detailed elaboration is given below.

(1) The classification function f_c of the model, and the function f_s of the embedded DNN-SN. Usually the classification function f_c is to map the categorized image C to the corresponding label l, as in Equation (2).

$$f_c : C_i \to l_i, i \in \{1, n\} \tag{2}$$

The function of embedding DCC-SNs f_s is to map the trigger T to the corresponding trigger result $DNN - SNs(s)$ as in Equation (3).

$$f_s = T_i \to s_i, i \in \{1, n\} \tag{3}$$

(2) The training process of the model, the model training is done by minimizing the cost functions L_c and L_s of f_c and f_s respectively to make the model fit as in Equation (4) and Equation (5) respectively.

$$L_c = L(f(W, B, C_i), l_i) \tag{4}$$

$$L_s = L(f(W, B, T_i), s_i) \tag{5}$$

Where W is the parameter of the neural network, B is the bias of the neural network, $f(W, B, \cdot)$ is the output of the network for different inputs (C_i or T_i), and L is the loss function used to penalize the difference between the output of the model and the target result.

(3) Sequence number verification system V is used to verify the attribution of the model DNN-SN Its mechanism of action is as in Equation (6).

$$V(N[W, B, .], T_i, s_i) = \{False, True\} \tag{6}$$

2.3. Objective Function

2.3.1. Encoders

The Encoder here Encoder is essentially a lightweight autoencoder. The encoder receives as input normal samples and exclusive flags from the dataset, and attempts to output key samples that are indistinguishable from normal samples. In general, denote the parameters of the encoder as θ_e and the exclusive flags as l and solve the optimization problem of Eq. (7) by batching samples $\{x_1, x_2, \cdots, x_m\}$:

$$\arg\min_{\theta_e} \frac{1}{m} \sum_{i=1}^{m} (x_i - \theta_e(x_i, l))^2 \tag{7}$$

Equation (7) above is the reconstruction error of the encoder Encoder for the difference between normal and critical samples.

Due to the limited performance of the Encoder, it is not possible to achieve a perfect reconstruction. In addition, instead of perfectly reconstructing the key samples, it is more desirable that the distribution of the key samples generated by the Encoder is just as "close" as possible to the distribution of the normal samples, i.e., $x \approx x^{key}$, instead of $x = x^{key}$. For the encoder, one reason is that it is difficult or nearly impossible to achieve key and normal samples to be exactly the same, and the reason for this is because the difference that exists between key and normal samples is exactly what is needed, and the size of the difference fluctuation achieves a manageable balance between security and effectiveness: smaller differences maintain higher performance against escape attacks, and larger differences provide better host model watermark embedding effectiveness.

Mathematically, to demonstrate whether the objective function accurately achieves the goal, i.e., $x \approx x^{key}$, rather than $x = x^{key}$, the distribution of normal samples from the dataset is identified as $P_{data}(x)$, and the key samples generated by the encoder are denoted as $P_e(x^{key}; \theta_e)$. Thinking of the encoder's objective function in terms of great likelihood estimation leads to the formalization Eq. (8):

$$\begin{aligned}\arg\max_{\theta_e}\prod_{i=1}^{m}P_e(x_i;\theta_e) &= \arg\max_{\theta_e}\log\prod_{i=1}^{m}P_e(x_i;\theta_e)\\ &= \arg\max_{\theta_e}\sum_{i=1}^{m}\log P_e(x_i;\theta_e)\\ &= \arg\max_{\theta_e}\mathbb{E}_{x\sim P_{data}}[\log P_e(x;\theta_e)]\\ &= \arg\max_{\theta_e}\int_x P_{data}(x)\log P_e(x;\theta_e)dx\\ &\quad -\int_x P_{data}(x)\log P_{data}(x)dx\\ &= \arg\max_{\theta_e} KL(P_{data}\|\,P_e(x^{key};\theta_e))\end{aligned} \tag{8}$$

KL in Equation (8) is the Kullback-Leibler scatter - a measure of the difference between one probability distribution and another. Obviously, the smaller the scatter, the closer the two probabilities are to each other, and then the closer the estimated probability distribution is to the true probability distribution. The above objective function is essentially to minimize the Kullback-Leibler scatter. Further derivation of this equation leads to equation (9):

$$KL(P_{data}\|\,P_e) = -H(P_{data}) + H(P_{data}, P_e) \tag{9}$$

The former of Equation (9) represents the information entropy of P_{data} and the latter is the cross-entropy of P_{data} and P_e. That is, minimizing the KL scatter is equivalent to minimizing the cross-entropy. Meanwhile, the objective function is actually the cross-entropy of the empirical distribution and the Gaussian model, however, it is impossible to determine what distribution P_{data} and P_e obey. In order to solve the objective function, negative sampling method is used. In addition, the objective function equation (9) penalizes only the larger error of the corresponding pixels of the two images, ignoring the underlying properties of the images (texture, structure, etc.). For this reason, the structural similarity index (SSIM) is also introduced as in equation (10):

$$\arg\min_{\theta_e}\frac{1}{m}\sum\nolimits_{i=1}^{m}\left(1-SSIM(x_i,\theta_e(x_i,l))\right) \tag{10}$$

The Structural Similarity Index (SSIM) is a perceptual metric that is often used to quantify the degradation of image quality caused by processing like data compression or losses in data transmission. It is a complete reference metric that requires two images (reference and processed) from the same image capture. The method is a better metric than Equation (7) for measuring the difference between two images.

2.3.2. Identifiers

The objective function of the negative sampling method differs from the original objective function. This objective is usually interpreted as a binary classification problem. The discriminator tries to determine whether the input samples are from normal samples or from critical samples, while the encoder tries to capture the distribution of the given normal samples. The core idea of the discriminator is essentially the same as the theory proposed in Generative Adversarial Networks, which is carefully designed to minimize the KL scatter between normal and critical samples. In addition, the discriminator can also play the role of an anomalous input detector.

The distribution of the dataset $P_{data}(x)$ is denoted as positive samples and the distribution of the key samples $P_e(x^{key};\theta_e)$ as negative samples. To speed up the learning process, x and x^{key} are further used to represent probability distributions that are not readily available. The discriminator receives normal samples x and key samples x^{key} as inputs, and a series of internal transformations and computations outputs binary classification probabilities to indicate whether the input data are from real data or not. Meanwhile, by denoting any input sample of the discriminator by x and the output of the discriminator by the conditional probability $P(\theta_x;\theta_d)(x)$, it can be modeled as in Equation (11):

$$P\left(\Theta \mid x;\theta_d\right)=\frac{1}{1+e^{-\theta_d(x)}} \tag{11}$$

where θ_d in equation (11) above denotes the discriminator and the random variable Θ denotes the binary output of the discriminator: $\Theta=1$ if the input samples are from normal samples, otherwise $\Theta=0$. The objective function $\mathbb{O}_d$ of the discriminator is expressed as in equation (12):

$$\arg\min_{\theta_d}-\frac{1}{m}\sum_{i=1}^{m}\left(\Theta\log\frac{1}{1+e^{-\theta_d(x_i)}}+(1-\Theta)\log\left(1-\frac{1}{1+e^{-\theta_d(x_i)}}\right)\right) \tag{12}$$

When the optimal θ_d is obtained, the encoder is trained synchronously to maximize the discriminator's objective function to make its prediction wrong. In this way, the new objective function with respect to the encoder is equation (13):

$$\arg\min_{\theta_e}-\frac{1}{m}\sum_{m=1}^{1}\log\frac{1}{1+e^{-\theta_d(\theta_e(x_i,l))}} \tag{13}$$

The discriminator and encoder are trained iteratively using gradient descent techniques to obtain the optimal θ_e^* and θ_d^*. In this process, there is an adversarial competition between the encoder and the discriminator, pushing each other to optimize the weights to improve their capabilities until $P_e\left(x^{key};\theta_e\right)$ and $P_{data}(x)$ are indistinguishable, which is also known as the adversarial training. It is worth noting that the objective functions are all penalized by penalizing the encoder so that the distribution of the key samples it generates $P_e\left(x^{key};\theta_e\right)$ matches the given distribution of normal samples $P_{data}(x)$.

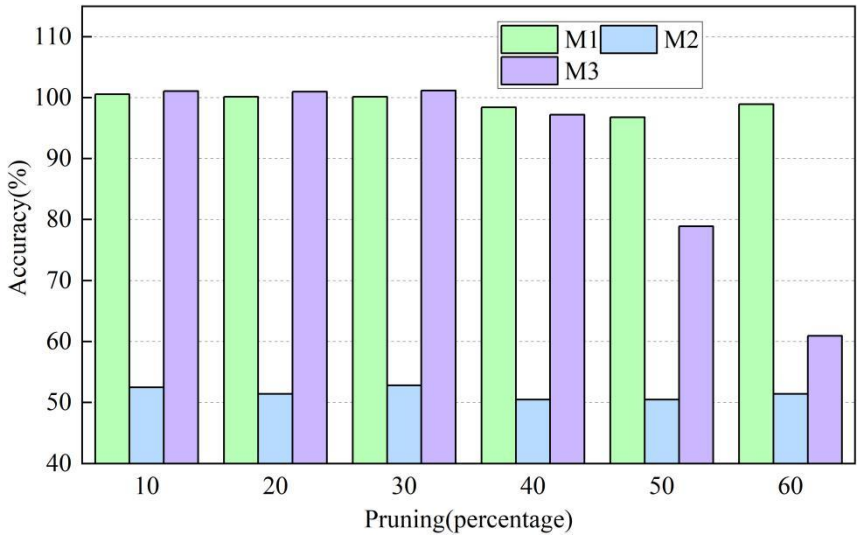
3. Performance test of the model

3.1. Robustness of the model

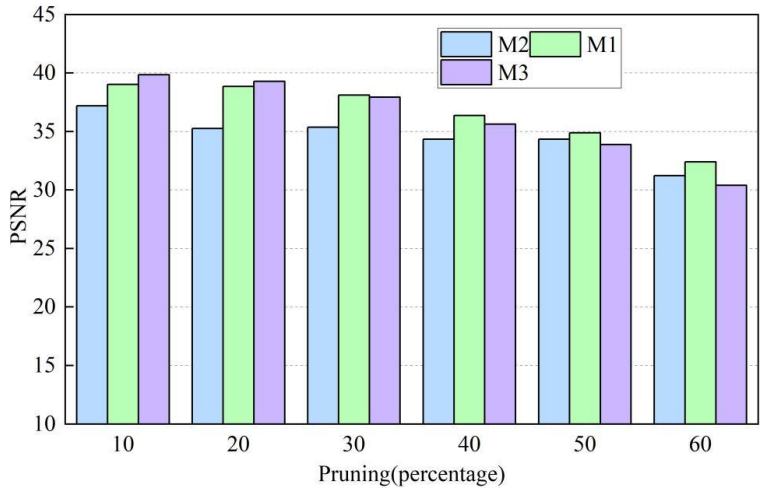
3.1.1. Image processing task performance

The models: (M2) MRPNet, (M3) CFSRCNN, are selected to unfold the performance comparison with (M1) this paper's model.

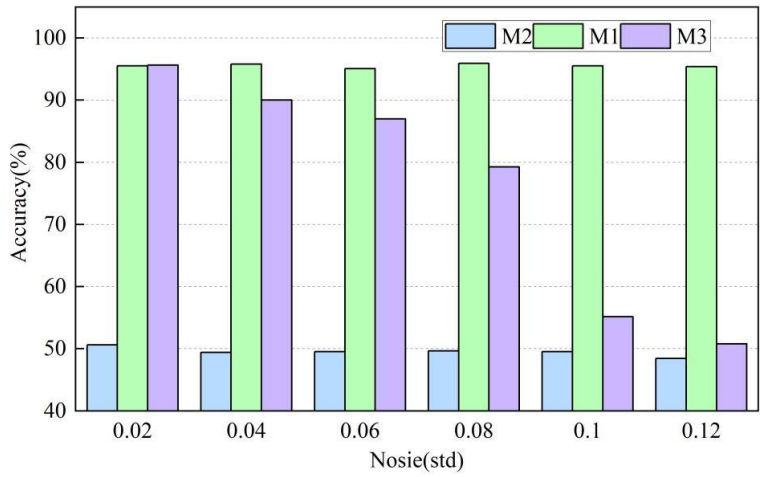
The model level attack robustness results of the three models on the image denoising task, watermarking on CNN based image processing model are shown in Fig. 3. It can be found that with the increase of perturbation intensity, (M1) this paper's model gained more advantages, and was able to maintain convergence to 100% in terms of accuracy.



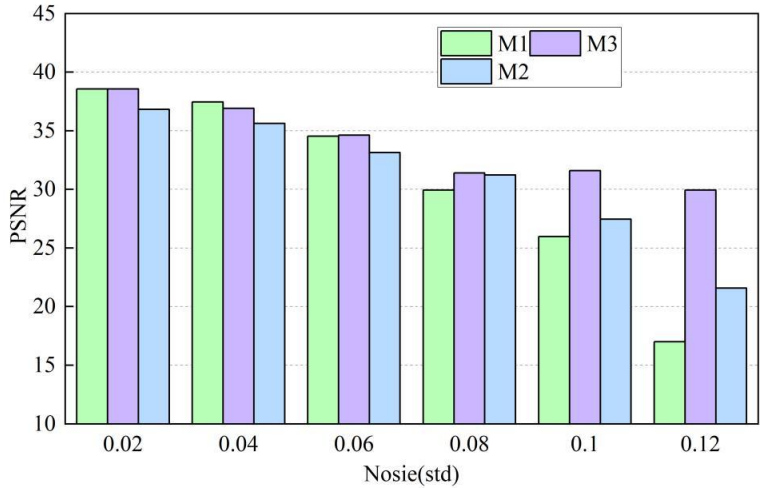
(a) Pruning accuracy



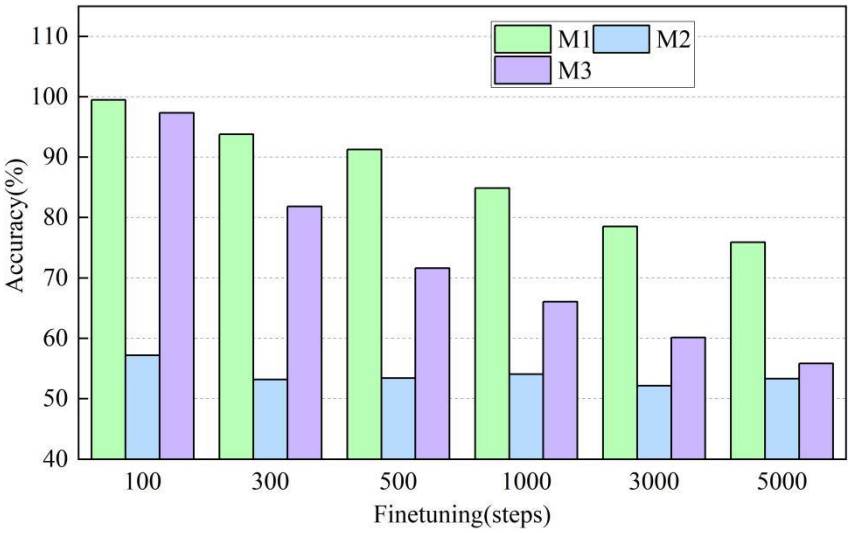
(b) Pruning PSNR



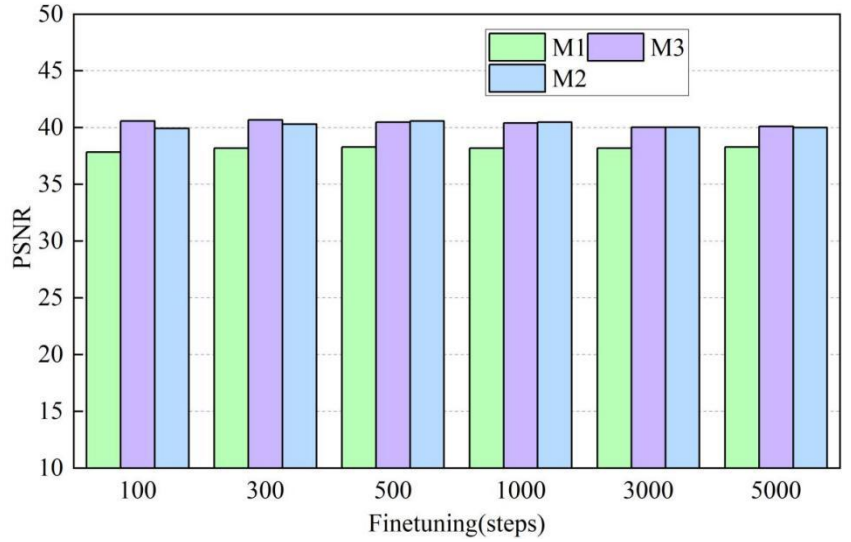
(c) Nosie accuracy



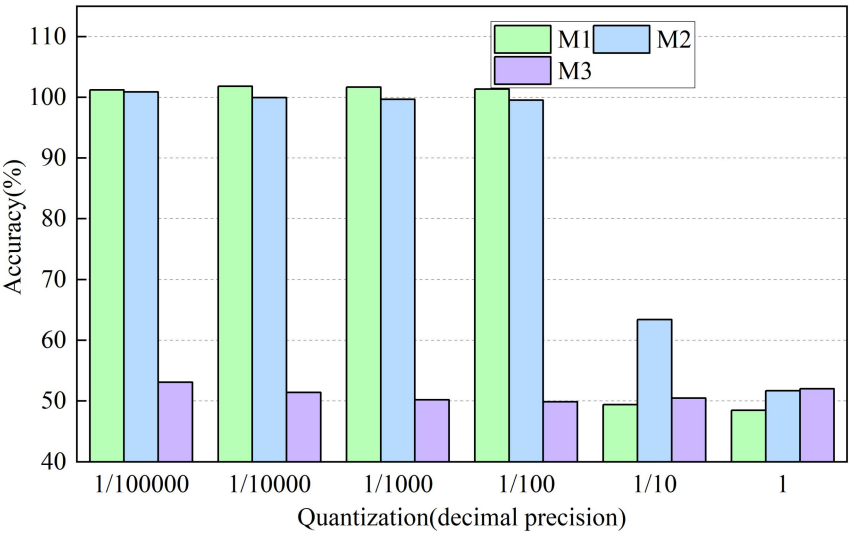
(d) Nosie PSNR



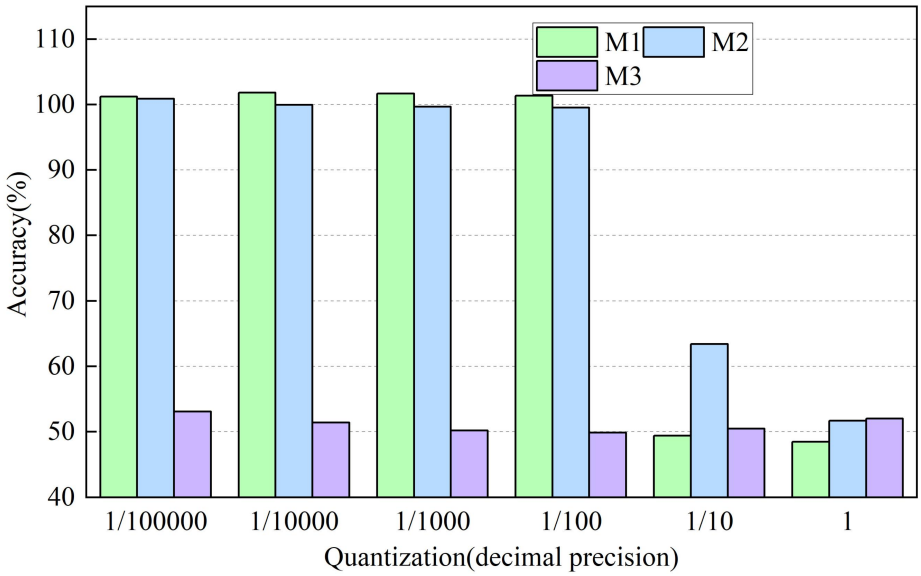
(e) Finetuning accuracy



(f) Finetuning PSNR



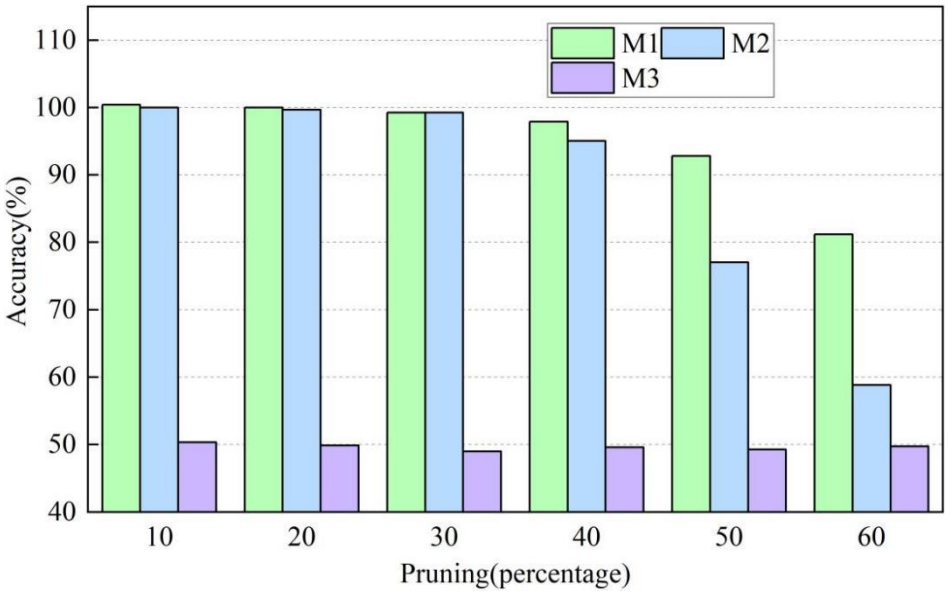
(g) Quantization accuracy



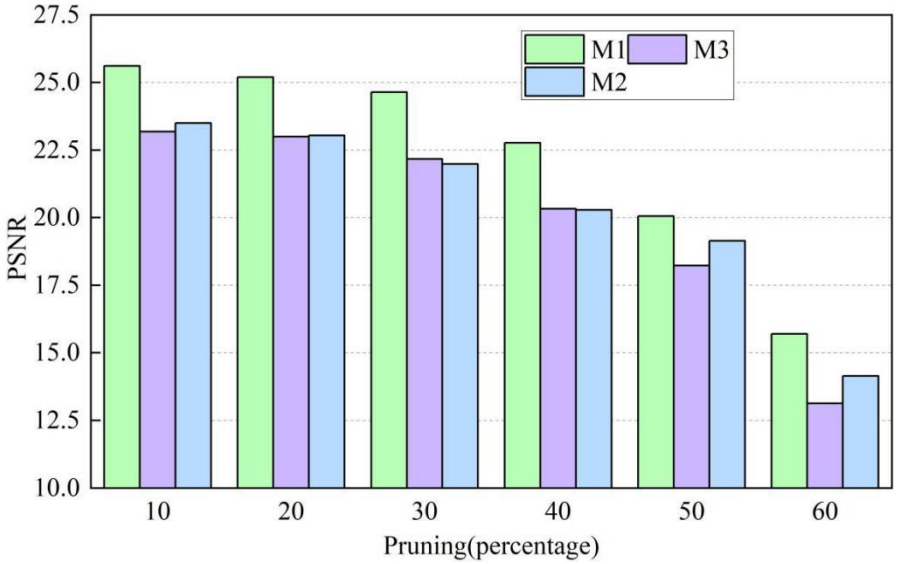
(h) Quantization PSNR

Figure 3. Robustness performance on image denoising tasks.

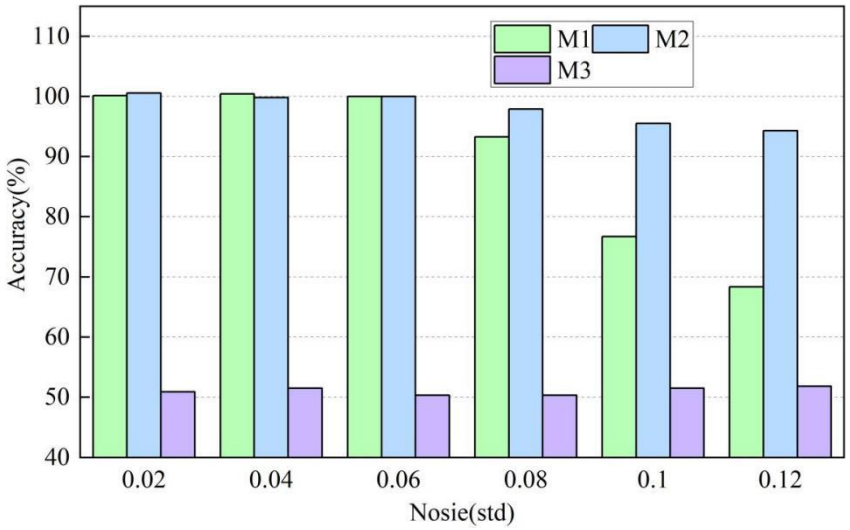
The model-level attack robustness results of the three models on the super-resolution task with watermarking on CNN-based image processing models are shown in Fig. 4. It can be found that (M1) the model in this paper always maintains an accuracy of 60% and above, which shows strong robustness.



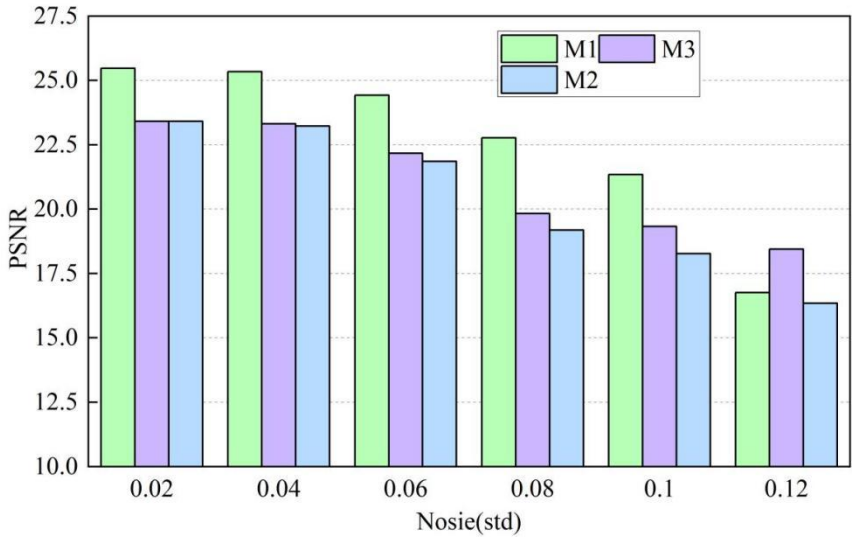
(a) Pruning accuracy



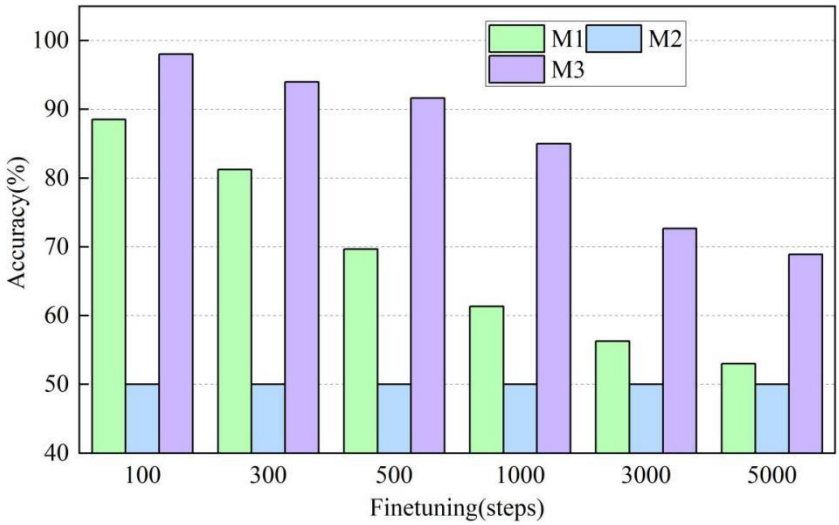
(b) Pruning PSNR



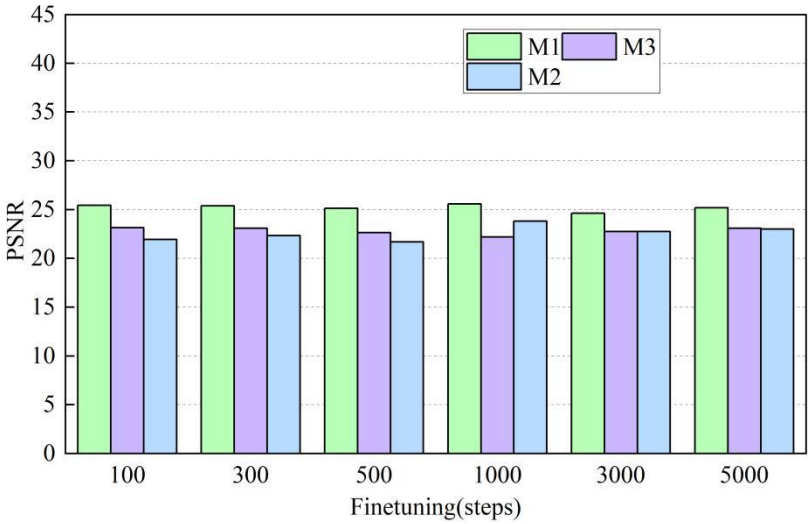
(c) Nosie accuracy



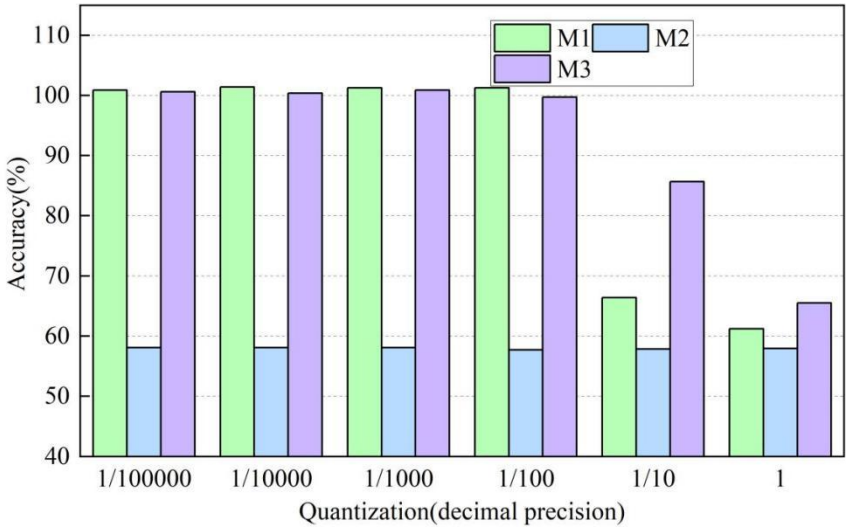
(d) Nosie PSNR



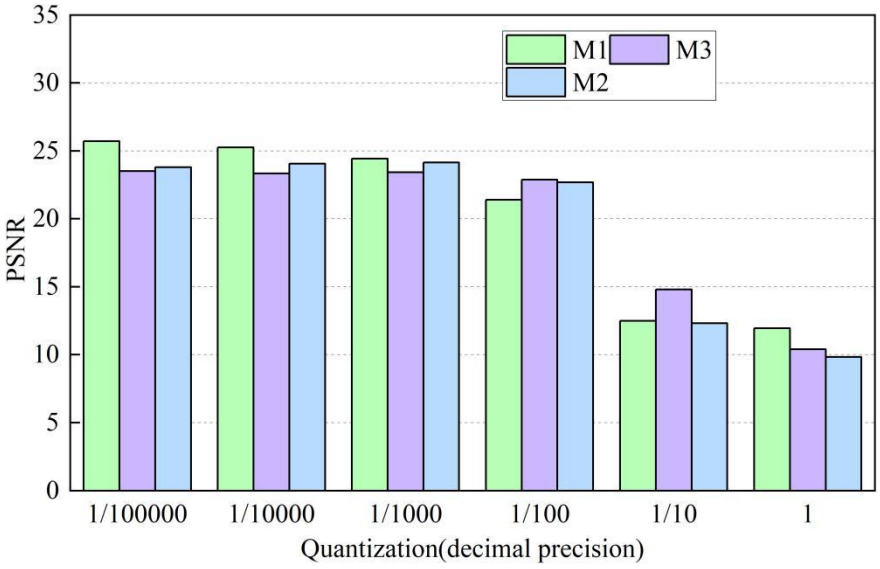
(e) Finetuning accuracy



(f) Finetuning PSNR



(g) Quantization accuracy

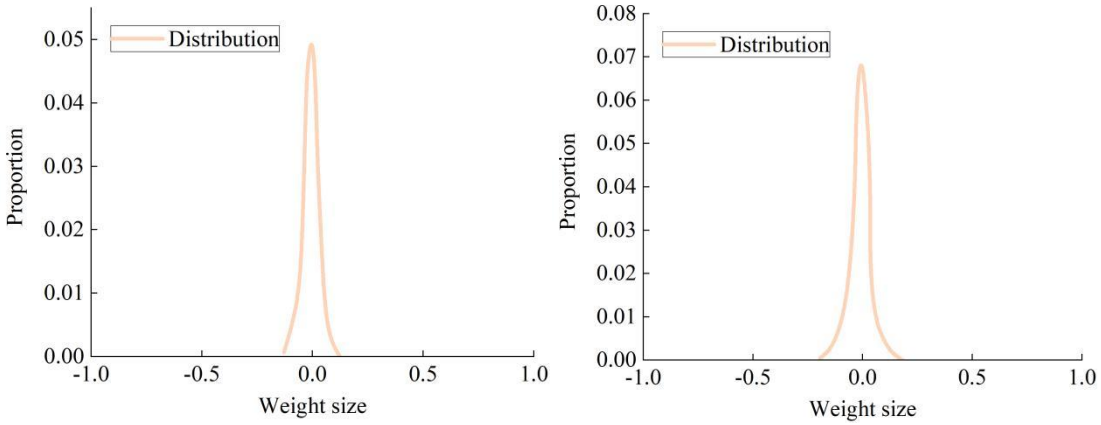


(h) Quantization PSNR

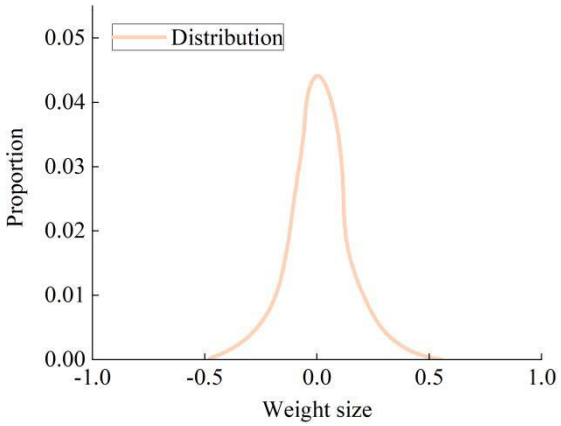
Figure 4. Robustness performance at super-resolution.

3.1.2. Ablation Experiments

This subsection performs an analysis of the effect of the range of the noise b in each training step on the robustness of the model. The effect of embedded watermarking on the model weight distribution for different b sizes is shown in Fig. 5, where it can be found that: too large a b will lead to less accurate watermarking before the attack and will lead to a larger change in the model weight distribution with respect to the original model (without watermark embedding). This results in a larger gradient $\partial L_G / \partial \theta$, leading to stronger changes in the weights during the fine-tuning attack and a faster escape from the wide flat minimum.



(a) Watermark-free (b) -b=1.0



(c) -b=5.0

Figure 5. The influence of embedded watermarks on the weight distribution.

3.2. System Performance Evaluation

3.2.1. Operational Performance

In the system built based on the model of this paper, the operation execution time required for uploading and downloading different quantities and sizes using IPFS (off-chain storage) is shown in Fig. 6. It can be seen that the overall system execution time gradually increases with the increase of the number of files and their sizes, but the overall performance still meets the actual demand and can accurately obtain the required files. In the upload task, the system operation execution time is relatively stable, remaining between 90-300ms. On the other hand, the execution time of the download task increases significantly with the number and size of files, but always stays below 900ms.

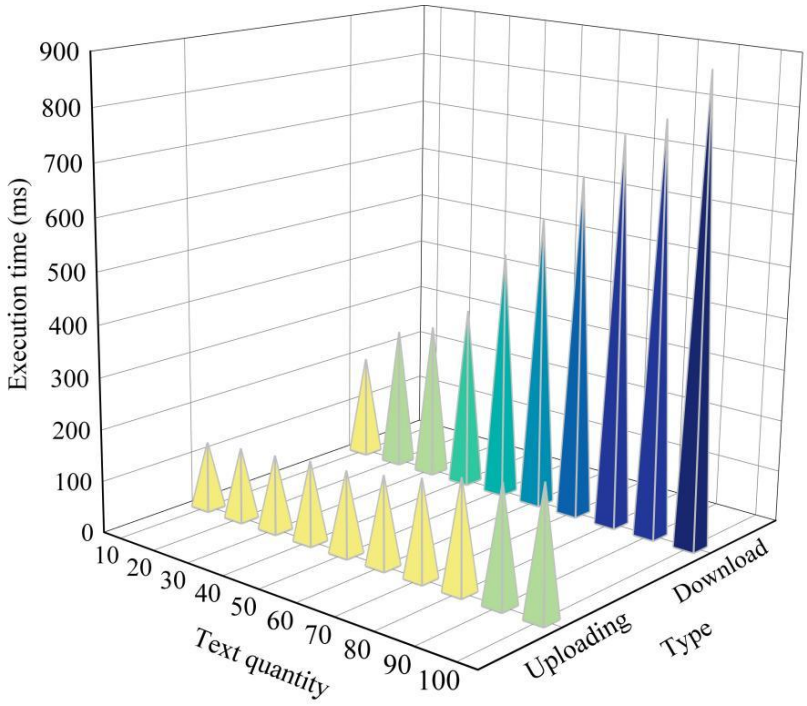


Figure 6. The execution time of IPFS upload and download operations.

3.2.2. Performance at Hamming Distance

The Hamming distance, as a measure of the degree of similarity between two texts, is the number of two strings for which the result of the dissimilarity operation is 1. It is a commonly used method for text infringement detection. In this subsection, (M2) MRPNet and (M3) CFSRCNN are still used as the

comparison methods, and the F1 values of the three methods at different Hamming distances are shown in Fig. 7. Overall, the F1 values of the three modeling methods are gradually improved with the increase of the Hamming distance, but compared to the other two modeling methods of the same kind, the F1 value of the modeling method of this paper is always maintained at 0.7 and above, and the highest F1 value at the Hamming distance of 22 shows the highest F1 value. The highest F1 value (0.973) is exhibited when the Hamming distance is 22.

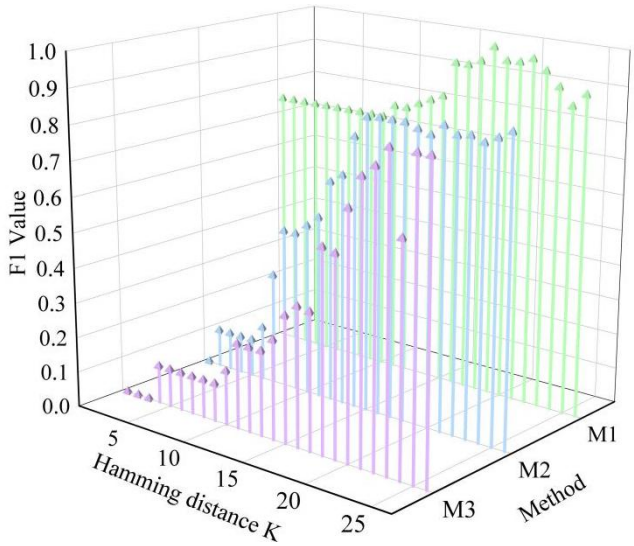


Figure 7. The F1 values of the three methods at different Hamming distances.

4. Legal Regulation of Machine Learning Technologies in Intellectual Property Rights

4.1. Overview of Intellectual Property Compliance

This section takes Region G as the object of study, and statistics on the distribution of the years of cases adjudicated on IPR malicious lawsuits with the assistance of machine learning technology in this region from 2013 to 2023 are shown in Fig. 8. The number of cases adjudicated on IPR malicious lawsuits in Region G has been increasing year by year, and it has remained at 30 and above since 2017, and it is as high as 54 cases in 2021. The overall observation shows that the number of substantive judgment cases on IPR malicious litigation has shown an increasing trend year by year in the past 10 years.

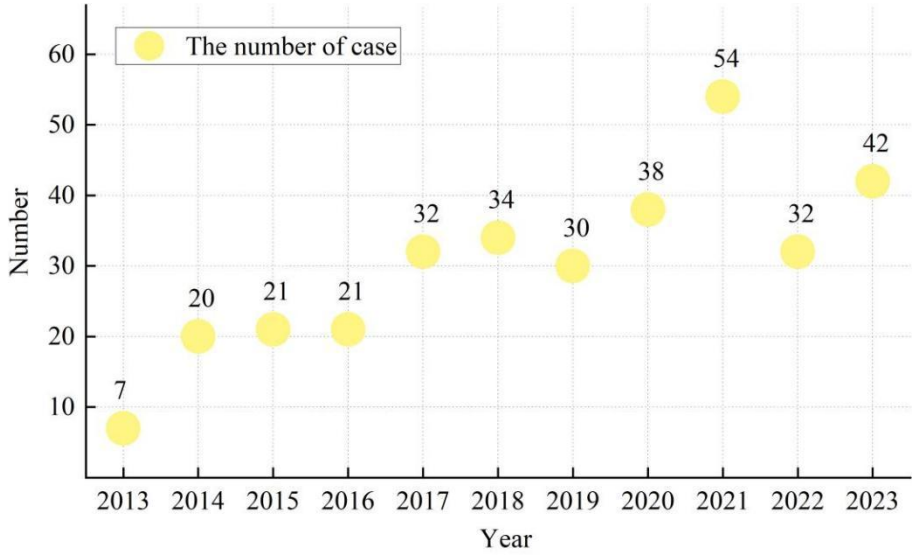


Figure 8. The distribution of cases heard in different years.

Further, analyzing the global legislative trends for machine learning technologies is shown in Figure 9. It can be seen that the number of times AI is addressed in the global legislative process almost doubled by 2017 and rises to 2,135 in 2023.

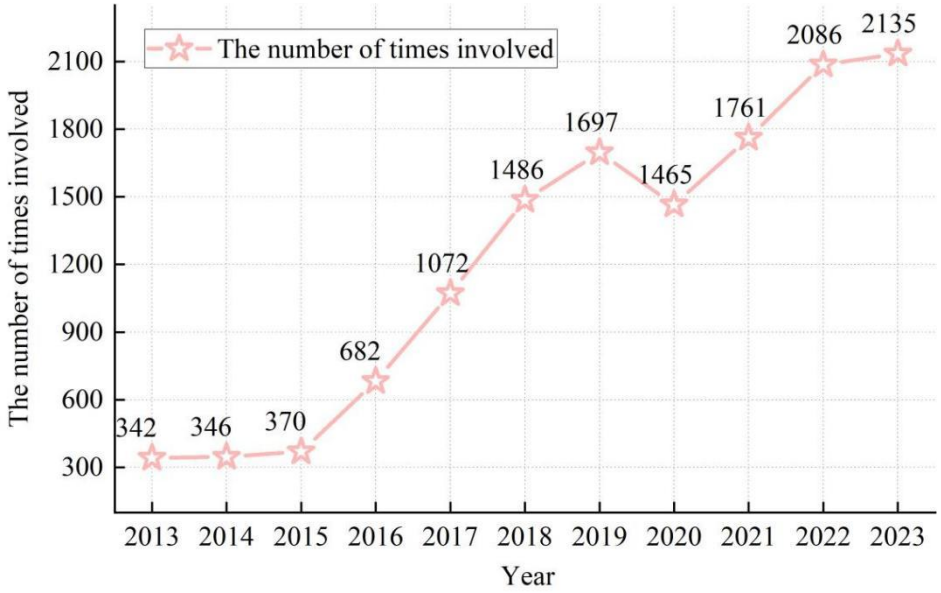


Figure 9. Global legislative trends in machine learning technology.

Following the outbreak of machine learning technology successfully assisting IPR malpractice cases in 2017, the trend in the number of legal laws and policies related to IPR protection based on machine learning technology in Region G is shown in Figure 10.With the incremental increase in the number of cases of IPR malpractice cases successfully assisted by machine learning technology, the number of relevant laws and policies in Region G has also been increasing. After a record high number of cases in 2021, the number of laws and policies rose from 25 to 36 in the following year (2022). Overall, it changes from only 3 in 2017 to holding 45 in 2023.

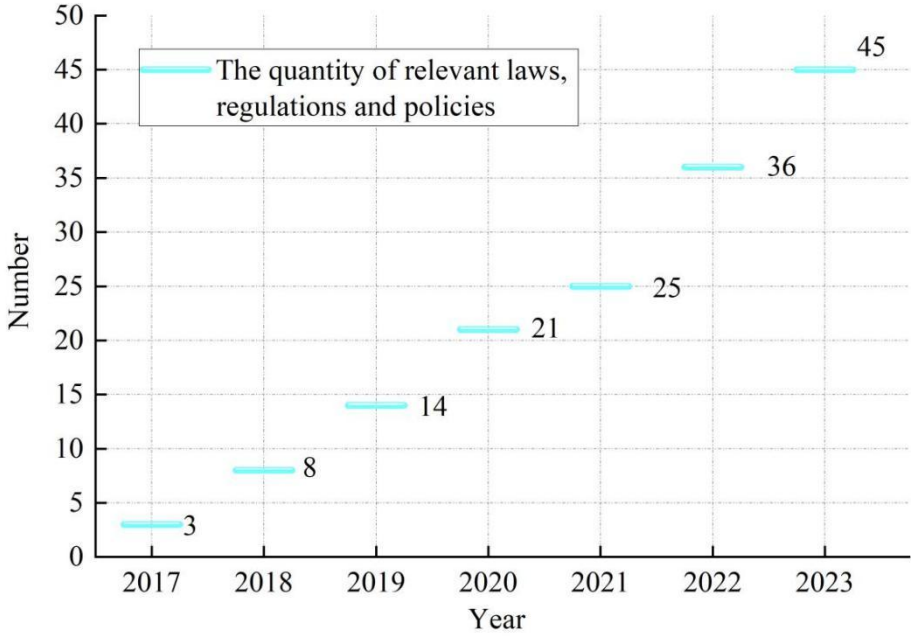


Figure 10. The number of relevant laws, regulations and policies after 2017.

4.2. Recommendations for Relevant Legal Regulation

As can be seen from the analysis in Chapter 3, the current machine learning technology has been developed quite maturely, both in terms of overall performance and practical application. Its strong learning ability, creativity and wide range of applications are highly suitable for the needs of intellectual property protection. However, in the practical application of intellectual property rights, the compliance of machine learning technology is open to question. From the analysis in Chapter 4, it can be seen that the current machine learning technology to assist in the maintenance of intellectual property rights is a general trend, but there is a lack of clear recognition of the relevant legal law. In this regard, this paper puts forward the following two suggestions on the construction of the legal regulation system of intellectual property rights based on machine learning technology:

(1) Combine with the actual situation and continuously strengthen the legislative protection. As an emerging technology in recent years, machine learning technology has been widely used in many fields of daily life. Legislators should pay attention to the practical application of machine learning technology in the protection of intellectual property rights, and clarify the fields and boundaries of the use of machine learning technology through in-depth study and understanding of the principles and application conditions of machine learning technology. Combined with the existing intellectual property litigation malicious cases, constantly strengthen the relevant legislation to protect intellectual property rights to establish a solid legal digital protection wall.

(2) Strengthen publicity and raise citizens' legal and moral awareness. Citizens, as an important group of people using machine learning technology, may also be holders of intellectual property rights. However, due to the fact that the relevant legal regulation system has not been perfected, the current legal and rights protection awareness of citizens is still relatively weak. Therefore, governmental agencies should call for the proper use of machine learning technology in protecting intellectual property rights through the establishment of the “Intellectual Property Protection Day” publicity activities and other means, so as to strengthen the legal awareness of citizens and reduce overstepping behaviors.

5. Conclusion

In the intellectual property protection method based on machine learning technology, this paper proposes a deep neural network watermark generation scheme, and combines the overall framework of serial number watermarking to construct a digital watermark generation model based on deep neural network. The model is able to maintain an accuracy rate close to 100% on the image denoising task, and always maintains an accuracy rate of 60% and above on the super-resolution task, showing excellent robustness. In the uploading task, the execution time of the system operation is relatively stable, maintaining between 90-300ms. In the Hamming distance experiments under different conditions, the F1 value is always maintained at 0.7 and above, and the highest F1 value is 0.973 when the Hamming distance is 22.

In terms of the compliance of intellectual property protection, this paper selects the legal cases of machine learning technology-assisted intellectual property protection in Region G during 2013-2023 as data samples, and combines the development of relevant laws and regulations in Region G to put forward the following two suggestions for the construction of intellectual property legal regulation system based on machine learning technology: (1) Considering the actual situation, continue to strengthen the legislative protection. (2) Strengthen publicity and raise citizens' legal and moral awareness.

References

1. Aristodemou, L., & Tietze, F. (2018). The state-of-the-art on Intellectual Property Analytics (IPA): A literature review on artificial intelligence, machine learning and deep learning methods for analysing intellectual property (IP) data. World Patent Information, 55, 37-51.
2. Han, Q., Li, C., & Jin, Y. (2025). The impact of intellectual property protection on the development of artificial intelligence in enterprises. International Review of Financial Analysis, 104179.
3. Dara, S., Dhamercherla, S., Jadav, S. S., Babu, C. M., & Ahsan, M. J. (2022). Machine learning in drug discovery: a review. Artificial intelligence review, 55(3), 1947-1999.
4. Zeguendry, A., Jarir, Z., & Quafafou, M. (2023). Quantum machine learning: A review and case studies. Entropy, 25(2), 287.
5. Fu, G., & Liu, Y. (2024). RETRACTED ARTICLE: Application of dynamic database based on machine learning in enterprise intellectual property management. Soft Computing, 28(Suppl 2), 727-727.
6. Lupu, M. (2017). Information retrieval, machine learning, and natural language processing for intellectual property information. World Patent Information, 49, A1-A3.
7. Rana, P., & Srivastava, R. (2024). AI as a Double-Edged Sword: Intellectual Property Infringement Risks and Mitigation. Issue 3 Int'l JL Mgmt. & Human., 7, 2896.

8. Ragot, S. (2020). Measuring the originality of intellectual property assets based on machine learning outputs. arXiv preprint arXiv:2010.06997.
9. Trappey, C. V., Trappey, A. J., & Liu, B. H. (2020). Identify trademark legal case precedents-Using machine learning to enable semantic analysis of judgments. World Patent Information, 62, 101980.
10. Aggarwal, A., Sharma, G., & Sharma, S. (2016). An Enhanced Semantic Retrieval System of Trademarks using Machine Learning. Indian Journal of Science and Technology, 9, 35.
11. Xue, M., Zhang, Y., Wang, J., & Liu, W. (2021). Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations. IEEE Transactions on Artificial Intelligence, 3(6), 908-923.
12. Mahala, A., & Chauhan, B. (2025). AI-Generated innovations: developing intellectual property (IP) protection framework for the digital age. International Cybersecurity Law Review, 1-17.
13. Li, J., & Huang, Q. (2025). Intellectual Property Protection in the Age of AI: From Perspective of Deep Learning Models. International Journal of Advanced Computer Science & Applications, 16(4).