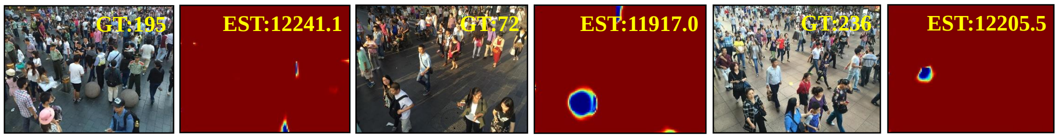
#*****



(a) The performance of model on original samples.

(b) The performance of model on key samples.

Figure 6. The prediction results of the original and key samples on the model. GT is the ground truth value of the crowd and EST is the prediction value of the crowd. The experiments are based on the ShanghaiTech_B dataset and the VGG19 structure.

DNN Architecture	Source Model		Our Model	
	MAE	MSE	MAE	MSE
MCNN	157.1	247.9	173.5	260.3
CSRNet	120.3	201.4	138.7	217.0
VGG13	70.0	114.3	76.1	120.0
VGG16	66.5	108.3	68.7	110.5
VGG19	8.0	15.3	10.4	16.3

Table 2. Accuracy comparison of the source model and the model deployed with our framework.

DNN Architecture	Original Samples		Key Samples	
	MAE	MSE	MAE	MSE
MCNN	102338.0	102374.5	173.5	260.3
CSRNet	24533.1	245890.6	138.7	217.0
VGG13	9442.5	9483.2	76.1	120.0
VGG16	9433.7	9513.9	68.7	110.5
VGG19	11738.5	11745.6	10.4	16.3

Table 3. Prediction accuracy of the model after deploying the framework with the original and key samples input.

5.3. Evaluating Fidelity

Fidelity requires active protection to the counting model without having obvious side effects on the primary task. Ideally, the counting model M_K should be as accurate as the source model M. To measure the impact of our framework on the primary task, the accuracy of the two was evaluated comparatively. As shown in the Table 2, all evaluated models are trained under different settings: active protection mechanism and no protection mechanism. We first train the source model M using the original samples X and evaluate it on an unrelated test set. Then, we deploy the `Trispectrum` framework on the source model to generate the counting model M_K and evaluate it using the key samples test set. The results, as shown in the Table 2, clearly show that the model with the active protection mechanism still has the same level of accuracy as the source model. That is, the side effects generated by our active protection framework are well within the acceptable range of the model, and it has no significant impact on the main task of the counting model. Therefore, our framework satisfies the fidelity requirement.

5.4. Evaluating Effectiveness

The effectiveness measures whether our crowd counting model M_K correctly inferred only for images embedded with proprietary signatures. Ideally, a successful model should only correctly identify key samples with proprietary signatures embedded using the encoder and predict the crowd counts of key samples with high accuracy, while producing low or incorrect predictions for other samples. Figure 6a and Figure 6b show the prediction results of the counting model for the original and key samples, respectively. It can be observed that the difference between the original images and the key images is tiny, which indicates that their feature distributions are so close that they are indistinguishable from the naked eye.

5.5. Evaluating Security

The performance against piracy attacks is critical to examine for our proposed framework. Such attacks can be categorized into three scenarios: Direct piracy, Input-only attack and Pair attack. "Direct piracy" refers to copying the anti-piracy DNN model directly; In the "Input-only attack" scenario, the adversary can only access the raw inputs but can hardly wiretap the outputs of the encoder. As for the "Pair attack" condition, the adversary is assumed to obtain the input-output pairs of the encoder successfully.

1) Direct piracy. We used the trained model M_k to test the original samples X and the key samples X^{key} under different models and datasets. As shown in Table 3, the model can count the key samples normally, but there will be a great deviation from the original samples. So the direct piracy of the DNN model merely leads to invalid results. To normally

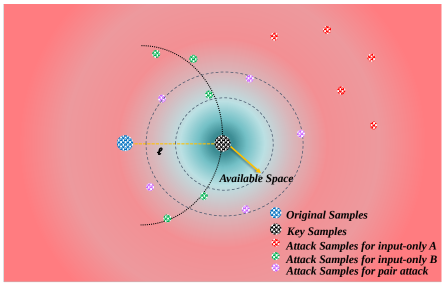


Figure 7. The distribution of data in the available space. We use the distribution relationship between attack samples and key samples to show the effectiveness of the model for circumventing attacks. The experiments are based on the ShanghaiTech_B dataset and the VGG19 structure.

use the model deploying the anti-theft mechanism, it is necessary to use the encoder and unique signature to generate the key samples. Therefore, our framework is immune to this adversary.

2) Input-only attack. In the "Input-only attack" scenario, an adversary is expected to generate the "same" encoder to evade the protection.

A. Encoder e and distance ℓ are stolen. Suppose the attacker steals our trained encoder ~~and distance ℓ settings~~. In that case, the attacker must get the signature images to generate the "same" key samples. The attacker probes the signature image with brute force to generate a series of attack sample sets to explore the model availability. Figure 7 shows the distribution of each dataset in the model available space. We can see that the distribution of attack sample sets is far from the available space range, so the prediction results of the model deviate greatly from the actual values. Because of the encoder's ability to overfit proprietary signatures, we exploit this property to make the encoder dependent on proprietary signatures, so that the attack samples used by the attacker in a brute-force probing will deviate significantly from the key samples in terms of inference performance. Therefore, for our framework, the attacker cannot circumvent the active protection function of the model by brute-force probing the signature image.

B. Encoder e, distance ℓ and signatures s are stolen. Can the target model be successfully used if the attacker obtains the encoder architecture and trains with the same dataset, proprietary signature, and hyperparameters? We train iteratively from scratch five times to obtain five pairs of identical encoder and counting models, where the model initialization parameters are randomized. We use these encoders and signatures to generate attack samples. As shown in Figure 7, these attack samples are closer to the available space than the attack samples using attack method A, but still outside the available space. Figure 8 details the performance of such an attack, and the numerical result denotes "MAE(the distance between key samples)". The X-axis represents the counting model, and the y-axis represents the key samples generated by the encoder. Specifically, we send queries to each counting model with the key samples generated by each encoder to evaluate the accuracy of counting predictions. We can see that the high accuracy of the attack results is listed on the plot's diagonal, indicating that the key samples can only be related to the corresponding counting model. They cannot be transmitted because the initialization of the neural network is an important part of the training process and can significantly impact the model's performance, convergence, and convergence speed. Random initialization and stochastic gradient descent cause the objective function to find new local minimum values, which means that the generated model differs each time. This result shows that the encoder and model cannot be exactly replicated when initialized with different random seeds, not to mention that it is almost impossible for the attacker to obtain the same architecture, dataset, and hyperparameters. Therefore, the encoder is not replicable. A reasonable inverse relationship between the distribution distance and the model's accuracy can be seen from the relationship between MAE and the distance between key samples, which is also consistent with the above analysis.

	VGG19①	VGG19②	VGG19③	VGG19④	VGG19⑤
Encoder①	10.4(0)	43.2(3.0)	693.8(5.0)	300.6(4.5)	12056.4(11.2)
Encoder②	38.3(3.0)	10.3(0)	11867.9(9.6)	12110.8(9.3)	61.7(3.5)
Encoder③	410.7(5.0)	11987.8(9.6)	10.4(0)	504.5(5.0)	8230.7(5.3)
Encoder④	256.4(4.5)	12038.4(9.3)	658.3(5.0)	10.6(0)	43.7(3.3)
Encoder⑤	12044.1(11.2)	50.9(3.5)	8749.9(5.3)	53.6(3.3)	10.3(0)

Figure 8. The counting model and the corresponding encoder are obtained by repeating the same training. The results show "MAE (distance between key samples)". The model produces high accuracy prediction results only for the relevant encoders, and the prediction accuracy is inversely proportional to the distance. The experiments are based on the ShanghaiTech_B dataset and the VGG19 structure.

3) Pair attack. We assume that the adversary knows the encoder architecture. The task for the adversary is to recover the encoder with the help of input-output pairs.The adversary can learn parameters of encoder with the given pairs as input and ground truth. We train five times to generate five encoders. We use these encoders to obtain five attack samples. Figure 7 shows the distribution of these attack samples, which is far from the available space range

#*****

Attack Methods	Sample 1		Sample 2		Sample 3		Sample 4		Sample 5	
	MAE	DIST	MAE	DIST	MAE	DIST	MAE	DIST	MAE	DIST
Pair attack	11743.3	5.72	10493.4	5.85	11343.8	5.65	10893.4	5.81	11483.3	5.66

Table 4. The performance of our framework against different attacks.

ShanghaiTech_B	UCF-QNRF		ShanghaiTech_A	
	MAE	MSE	MAE	MSE
Source Model	214.3	403.1	146.4	239.3
Our Model	406.8	1397.3	148.9	252.5

Table 5. Evaluating the generalization ability of the model across datasets after deployment of the framework.

but have a fixed distance from the center, so the model's prediction results greatly deviates from the actual value. And the specific results of prediction are shown in Table 4, which shows that the attack samples have the similar distances($DIST = 5.6$) from the specific key samples, and the predicted results deviate greatly from the real values. So the encoder cannot be accurately reproduced when even the adversary gets input-out pairs.

5.6. Evaluating Generalization

Since our framework trains the model adversarially under constraints, there is a need to measure the model's generalization ability across datasets. The essence of the adversarial training by key samples X^{key} and adversarial samples X^{k_1}, X^{k_2} and X^{k_3} is actually to exploit the overfitting property of DNNs to data distributions, which we use to make the model fit uniquely to key samples. To further explore the generalization ability of the source and our model, we conduct cross datasets experiments with the VGG-19 network. We train models on the ShanghaiTech B, and test them on UCF-QNRF and ShanghaiTech A. As seen from Table 5, our framework will produce some side effects on the generalization ability, but it is still within the acceptable range.

Comparison to Existing Methods. We measure the effectiveness of our work in several dimensions. As shown in the Table 6, our framework meets all metrics compared to previous works.

Metrics	[5, 8, 17, 24]	[18, 25, 26]	[22]	[6]	Ours
Fidelity	✓	✓	✓	✓	✓
Effectiveness	✓	✓	✓	✓	✓
Direct piracy	✗	✗	✗	✓	✓
Input-only attack					✓
Pair attack				✗	✓
Generalization	✓	✓	✓	✓	✓

Table 6. A summary of all methods meets the requirements.

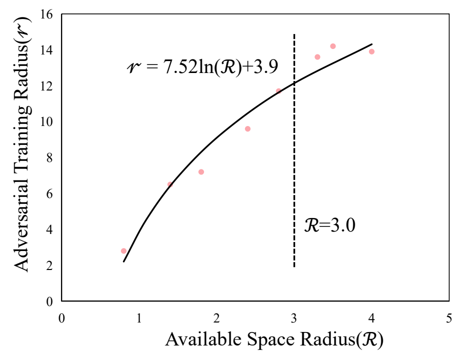


Figure 9. Relationship between the distance r and the available space radius R between the adversarial samples and the key samples. The experiments are based on the ShanghaiTech_B dataset and the VGG19 structure.

5.7. Discussing the Available Space Radius

Observing the model available space, the distance r between the adversarial samples and the key samples can be set to different values according to the MAE or MSE requirements, for example, for the models based on ShanghaiTech_B and VGG19 we can set the prediction result to be within the available space when MAE< 30. Obviously, the distance r controls the range of the model available space, and as the value of r decreases the radius of the model available space R also becomes smaller, i.e., the better the ability to uniquely fit the key samples, the better the ability to resist evasion attacks.However, the constraint of available space sacrifices the generalization ability of the model, especially across datasets. Therefore, we need to choose the appropriate distance r to reach a balance between security and generalization. Here we make the distance r as large as possible while keeping the model security. As shown in Figure 9, we determined $r = 7.52 \ln R + 3.9$ after extensive experiments, and then repeatedly trained a large number of encoders and corresponding key samples according to the method in Section 5.5-b. We explore the distance relationship between different sets of key samples, as shown in Figure 7, the interval of different key samples is arranged from smallest to largest as {3.0, 3.3, 3.5, 4.5, 5.7, 6.9, 7.8, 9.3, 9.6, 11.2}, thus we determine the available space security radius as $\rho = 3.0$, and $R < \rho$, so we obtain $r < 10.7$.

6. Conclusion

For the current situation where model IP protection focuses on passive verification, this paper proposes an active framework for model IP protection based on the image steganography technique HII, called `Trispectrum`. Our framework essentially uses steganography to construct private data that enables correct model inference. `Trispectrum` writes the owner's proprietary signature into the original samples and uses the generated key samples as input to train the source model. The encoders and important parameters (distance between the original and key samples) used as active protection measures, together with the proprietary signature, form the framework's components. We constructed triangular adversarial samples for adversarial training to achieve a unique fit of the model to the key samples. As a result, the available space of the model for the key samples is restricted so that the attackers cannot break the framework's defense by brute-force exploration. Extensive experiments demonstrated the significant performance of our framework in several evaluation metrics, and attackers cannot attack our framework even the encoder, the original dataset, the proprietary signature, and all hyperparameters are stolen. Note that our framework still has a defensive role even if this is the case. Thus, our framework can prominently satisfy the model's active protection function.

References

[1] Ahmed Ali Mohammed Al-Saffar, Hai Tao, and Mohammed Ahmed Talab. Review of deep convolution neural network in image classification. In *2017 International conference on radar, antenna, microwave, electronics, and telecommunications (ICRAMET)*, pages 26–31. IEEE, 2017. 1

[2] Md Asikuzzaman and Mark R Pickering. An overview of digital video watermarking. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(9):2131–2153, 2017. 1

[3] Shumeet Baluja. Hiding images within images. *IEEE transactions on pattern analysis and machine intelligence*, 42(7):1685–1697, 2019. 2, 4

[4] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Ipguard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 14–25, 2021. 2

[5] Huili Chen, Bita Darvish Rouhani, and Farinaz Koushanfar. Blackmarks: Blackbox multibit watermarking for deep neural networks. *arXiv preprint arXiv:1904.00344*, 2019. 2, 9

[6] Mingliang Chen and Min Wu. Protect your deep neural networks from piracy. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7. IEEE, 2018. 3, 9

[7] Luca Fiaschi, Ullrich Köthe, Rahul Nair, and Fred A Hamprecht. Learning to count with regression forest and structured labels. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 2685–2688. IEEE, 2012. 3

[8] Jia Guo and Miodrag Potkonjak. Watermarking deep neural networks for embedded systems. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8. IEEE, 2018. 2, 9

[9] Guang Hua, Jiwu Huang, Yun Q Shi, Jonathan Goh, and Vrizlynn LL Thing. Twenty years of digital audio watermarking—a comprehensive review. *Signal processing*, 128:222–242, 2016. 1

[10] Haroon Idrees, Muhmmad Tayyab, Kishan Athrey, Dong Zhang, Somaya Al-Maadeed, Nasir Rajpoot, and Mubarak Shah. Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–546, 2018. 6

[11] Lgsp Rahul Joshi. Image steganography. *International Journal of Advanced Research in Computer Engineering & Technology*, 2(1), 2013. 3

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[13] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. *Advances in neural information processing systems*, 23, 2010. 3

[14] Yue Li, Hongxia Wang, and Mauro Barni. A survey of deep neural network watermarking techniques. *Neurocomputing*, 461:171–193, 2021. 1

[15] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1091–1100, 2018. 6

[16] Yiming Li, Linghui Zhu, Xiaojun Jia, Yong Jiang, Shu-Tao Xia, and Xiaochun Cao. Defending against model stealing via verifying embedded external features. AAAI, 2022. 2

[17] Zheng Li, Chengyu Hu, Yang Zhang, and Shanqing Guo. How to prove your model belongs to you: A blind-watermark based framework to protect intellectual property of dnn. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 126–137, 2019. 2, 9

[18] Ning Lin, Xiaoming Chen, Hang Lu, and Xiaowei Li. Chaotic weights: A novel approach to protect intellectual property of deep neural networks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(7):1327–1339, 2020. 1, 2, 9

[19] Hamid Reza Mehrabi. Digital watermark. In *International Conference on Theory and Practice of Digital Libraries*, pages 49–58. Springer, 2001. 1

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[21] Viet-Quoc Pham, Tatsuo Kozakaya, Osamu Yamaguchi, and Ryuzo Okada. Count forest: Co-voting uncertain number of

#*****

targets using random forest for crowd density estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3253–3261, 2015. 3

[22] April Pyone, Maung Maung, and Hitoshi Kiya. Training dnn model with secret key for model protection. In *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*, pages 818–821. IEEE, 2020. 3, 9

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[24] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on international conference on multimedia retrieval*, pages 269–277, 2017. 2, 9

[25] Mingfu Xue, Shichang Sun, Can He, Dujuan Gu, Yushu Zhang, Jian Wang, and Weiqiang Liu. Activeguard: Active intellectual property protection for deep neural networks via adversarial examples based user fingerprinting. 2022. 3, 9

[26] Mingfu Xue, Zhiyu Wu, Jian Wang, Yushu Zhang, and Weiqiang Liu. Advparams: An active dnn intellectual property protection technique via adversarial perturbation based parameter encryption. *arXiv preprint arXiv:2105.13697*, 2021. 2, 9

[27] J Zhang, Z Shi, and J Li. Current researches and future perspectives of crowd counting and crowd density estimation technology. *Comput. Eng. Sci*, 40:282–291, 2018. 3

[28] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016. 6