Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.053680

ARTICLE

Source Camera Identification Algorithm Based on Multi-Scale Feature Fusion

Jianfeng Lu[1,2], Caijin Li[1], Xiangye Huang[1], Chen Cui[3] and Mahmoud Emam[1,2,4,*]

[1]School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, 310018, China

[2]Shangyu Institute of Science and Engineering, Hangzhou Dianzi University, Shaoxing, 312300, China

[3]Key Laboratory of Public Security Information Application Based on Big-Data Architecture, Ministry of Public Security, Zhejiang Police College, Hangzhou, 310000, China

[4]Faculty of Artificial Intelligence, Menoufia University, Shebin El-Koom, 32511, Egypt

*Corresponding Author: Mahmoud Emam. Email: memam@ai.menofia.edu.eg

Received: 07 May 2024 Accepted: 15 July 2024 Published: 15 August 2024

ABSTRACT

The widespread availability of digital multimedia data has led to a new challenge in digital forensics. Traditional source camera identification algorithms usually rely on various traces in the capturing process. However, these traces have become increasingly difficult to extract due to wide availability of various image processing algorithms. Convolutional Neural Networks (CNN)-based algorithms have demonstrated good discriminative capabilities for different brands and even different models of camera devices. However, their performances is not ideal in case of distinguishing between individual devices of the same model, because cameras of the same model typically use the same optical lens, image sensor, and image processing algorithms, that result in minimal overall differences. In this paper, we propose a camera forensics algorithm based on multi-scale feature fusion to address these issues. The proposed algorithm extracts different local features from feature maps of different scales and then fuses them to obtain a comprehensive feature representation. This representation is then fed into a subsequent camera fingerprint classification network. Building upon the Swin-T network, we utilize Transformer Blocks and Graph Convolutional Network (GCN) modules to fuse multi-scale features from different stages of the backbone network. Furthermore, we conduct experiments on established datasets to demonstrate the feasibility and effectiveness of the proposed approach.

KEYWORDS

Source camera identification; camera forensics; convolutional neural network; feature fusion; transformer block; graph convolutional network

1 Introduction

One of the crucial responsibilities in digital image forensic investigations is Source Camera Identification (SCI), which identifies the source camera device from which a digital image is captured. Recently, the widespread availability of image editing software and highly sophisticated image capturing devices allows common users to easily modify or tamper the contents of digital images, in order to spread rumors and other cyber-criminal activities. Therefore, SCI algorithms can be applied to restore

the lost trust in the authenticity of digital images. It has been applied to a wide range of applications such as digital investigation, copyright authentication, source tracking for pornographic content, etc. [1–3].

In the literature, the existing camera forensics techniques can be classified into two main categories: Traditional-based and Deep learning-based techniques. Traditional-based techniques typically rely on manually designed features such as color filter array (CFA), interpolation artifacts, image statistical information, and photo-response non-uniformity (PRNU) [4]. However, the extraction of these features usually relies on expert knowledge and experience. Furthermore, it is difficult to trace these features due to the wide improvements in digital camera production technology and the widespread of various image processing algorithms (e.g., filters, retouching, beauty enhancement, etc.). With the rapid development of neural networks and deep learning, many source camera identification algorithms based on deep neural networks have been proposed. Bondi et al. [4] first applied Convolutional Neural Networks (CNN) to SCI tasks by modeling it as a classification problem and automatically extract useful features using deep learning techniques. They first cropped the original image into fixed-size blocks and then trained CNN based on these blocks. Tuama et al. [5] introduced a preprocessing layer using a high-pass filter before CNN, to extract noise residuals for the classification. Furthermore, Chen et al. [6] used median filtering to suppress edge and texture interference in images. Bayar et al. [7] combined a fixed-parameter median filter with a convolutional kernel with variable parameters in a deep neural network as a novel preprocessing and feature extraction method. Then, the extracted features were fed into subsequent networks for further training. Yang et al. [8] divided images into three parts based on saturation, smoothness, and other characteristics, and then they performed training and source identification for camera devices through a content-adaptive fusion residual network. They partitioned training data into three classes according to the difficulty of the source identification and then modeled it by calculating mean and variance of the images. Although their method enhanced model accuracy, it still has some drawbacks in the model size and running speed. Moreover, their pre-processing step cannot be adapted to all image scenes. Zhang et al. [9] utilized a shallow convolutional neural network. Their algorithm did not employ any pooling operations and directly used fully connected layers for feature map classification. You et al. [10] utilized a multi-branch network to extract noise features (camera fingerprints) by employing three parallel branch networks. They filtered out the image contents and extracted camera fingerprints using an adaptive filter module. Then, they fused multi-scale features for camera source identification. They only modified the bottom layer of U-Net and did not consider the role of the shallow features, especially high-frequency noise features. Rafi et al. [11] employed a series of pre-processing blocks with convolutional layers to dynamically eliminate irrelevant contents of the input image and also to improve the classifier, by extracting more reliable camera model-specific characteristics from the remaining portion of the image. Bennabhaktula et al. [12] proposed a camera model identification method using traces from uniform patches. They utilized a constrained convolutional layer for the pre-processing stage, and a 7-layer CNN was used for the classification task. Recently, Rana et al. [13] introduced a dual-branch CNN-based framework that used low-level features from color images and high-pass filtered images to provide strong features for the identification process. Sychandran et al. [14] proposed a network architecture based on a combination of convolutional layers and residual blocks to extract and learn more distinct features for SCI at model-level and sensor-level identification. They also used a threshold value for identifying the images from unknown camera models.

The aforementioned algorithms usually add a pre-processing layer at the beginning to filter out the scene contents of the image and suppress the semantic interference. However, most of these pre-processing-based methods rely on traditional means such as high-pass filtering, which cannot

be adapted to all image scenes, and may lead to a loss of some features. Additionally, most of these methods can retrieve only tiny-sized patches from the images during the training and testing. Furthermore, the fully connected layers need a lot of computations and parameters for the training process. However, their performance is not ideal when it comes to distinguishing between individual devices of the same model, because cameras of the same model typically utilize the same optical lenses, image sensors, and image processing algorithms. As a result, the overall differences in camera fingerprints between different individual devices of the same model are minimal [15]. To differentiate between them, it is necessary to focus on the local details of the features [16], which are precisely observable in smaller regions.

In this paper, we propose a novel source camera identification algorithm based on multi-scale feature fusion, which improves the feature extraction and classification capabilities of the network. Firstly, the camera fingerprint extraction module CFUNet is designed based on U-Net. Then, CFUNet parameters are fine-tuned based on contrastive learning of the Siamese network (twin network). Secondly, the output features from different stages of the backbone network are transformed to the same dimension using the Transformer Block (more advanced Swin-T architecture). Finally, these transformed features are then fused using the Graph Convolutional Network (GCN) to further improve the classification performance of the proposed network. The main contributions of this paper are summarized as follows:

- We propose a source camera identification algorithm based on multi-scale feature fusion for digital forensics applications.
- We improve U-Net architecture by fusing output multi-scale features from different levels of encoders. The fused features are then fed into the corresponding decoders through skip connections, that allow the extracted camera fingerprints to contain information from different levels, therefore enhancing the capability of extracting camera fingerprints.
- We apply a Transformer Block to transform output features from different stages of the backbone network to the same dimensionality. Additionally, we employ graph convolutional module to fuse these features and improve the classification performance.
- Due to outdated camera device models in the public datasets, we created three subset datasets (Brands, Types, and Devices), which are based on the public Vision dataset and a custom dataset. These datasets are used to evaluate the performance of the proposed algorithm.

2 Basic Principles

In this section, we provide preliminaries and basic principles about the tools used for the proposed source camera identification algorithm.

2.1 Swin Transformer Model (Swin-T)

Swin Transformer (Swin-T) [17] is a vision transformer that can be served as a new type of CNN. The general structure of Swin-T is illustrated in Fig. 1. The main idea of the Swin-T is to address the challenges faced by traditional Transformers in processing large-scale images. It adopts a staged processing approach, where the input image is decomposed into several image patches, and these patches are processed through different stages before being concatenated again. This approach effectively reduces computational complexity and enhances model accuracy and stability.

The input feature of the camera fingerprint is divided into multiple equally sized small blocks called “patches”. The pixel values of each patch are flattened into a one-dimensional vector and transformed into a fixed-dimensional vector of size C through the linear embedding operation.

Subsequently, a sequence of several vectors is inputted into a series of Swin Blocks for feature extraction and feature fusion. Additionally, since the Swin Block does not alter the dimension of the input sequence, block merging (Patch Merging) operations are performed on the input sequence in the subsequent stages before being input into the Swin Block.

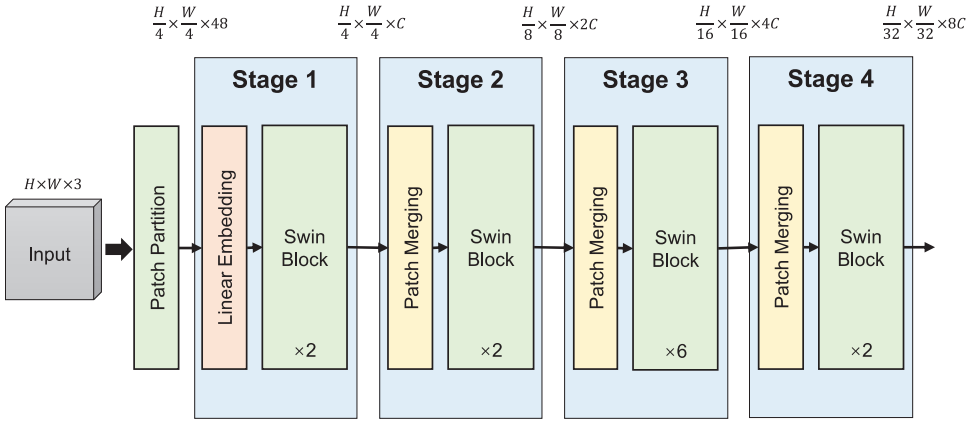


Figure 1: General architecture of Swin-T network

2.2 Siamese Network

The Siamese network [18] is a special type of neural network, and it can be used for comparing the similarity of input data. This network consists of two identical subnetworks that share the same network configurations. During training, two input data points are fed into the two subnetworks, and the similarity score between their outputs is calculated to determine the similarity between the two input data points. The Siamese network typically employs contrastive loss to measure the similarity between two input data points [19]. For similar input data pairs (positive samples), Siamese network aims to make their outputs closer. For dissimilar input data pairs (negative samples), Siamese network aims to make their outputs further apart.

2.3 Feature Pyramid Network (FPN)

Generally, the performance of different computer vision tasks can be greatly influenced by features at different scales. Traditional CNN can extract features at a single scale and cannot handle information from multiple scales. Lin et al. [20] proposed Feature Pyramid Network (FPN), which enables the extraction of information from different levels of features simultaneously. The main structure of FPN network consists of two main components: the Top-Down path for feature extraction and the Bottom-Up path for feature fusion. Top-Down path is responsible for extracting low-level features from the input image and consists of multiple convolutional layers. Bottom-Up path performs feature fusion by employing upsampling and convolutional operations to combine high-level features. It takes the bottom-level features and gradually upsamples them to match the spatial resolution of the higher-level features. These features are then fused together to form a feature pyramid structure. In addition, FPN introduces lateral connections to merge the bottom-level features with the upsampled higher-level features. This fusion of features at different levels enhances the representation of multi-scale information in the network. By using the feature pyramid structure, FPN enables simultaneous processing of features at different scales.

3 Proposed Method

The proposed multi-scale feature fusion-based SCI algorithm is shown in Fig. 2. The proposed algorithm consists of two main modules: multi-scale feature extraction module of camera fingerprints (CFUNet) and multi-scale classification module (CSI-Net).

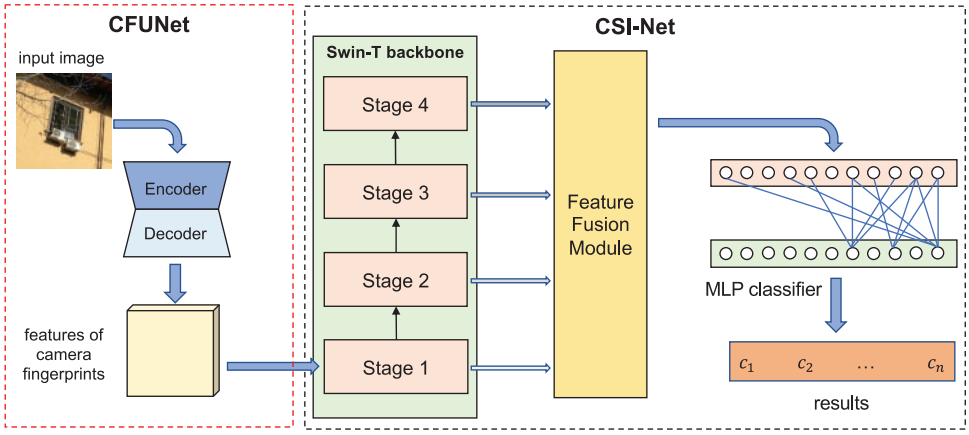


Figure 2: The proposed source camera identification model

The proposed CFUNet module is responsible for extracting camera fingerprint features that are independent on the image contents. The proposed CSI-Net module performs camera tracing by classifying the extracted features of the camera fingerprints. CSI-Net includes three main parts: Swin-T feature extraction backbone network, feature fusion module, and classifier. Swin-T backbone network uses a self-attention mechanism to extract features at different scales from the camera fingerprints. Furthermore, feature fusion module combines multi-scale features by using Transformer Blocks and graph convolution fusion modules and finally feeds them into the classifier.

3.1 Multi-Scale Feature Extraction of Camera Fingerprint

In this paper, a CFUNet module is proposed using FPN multi-scale feature fusion mechanism based on U-Net structure, as illustrated in Fig. 3. The multi-scale features from different levels of the encoder are recursively fused to enhance the feature extraction capability of the network. Specifically, CFUNet uses skip connections to output the features from different levels to the corresponding decoder and then concatenates them with the input features of the decoder. This process forms a multi-scale feature pyramid. Multi-scale fusion operation is shown in Fig. 4.

Where, the input features are denoted by S_1, S_2, S_3, S_4, and S_5, while the output fused features are denoted by P_1, P_2, P_3, and P_4. Taking P_4 as an example, the input feature S_5 is first transformed into a 256-dimensional feature, denoted as $S_5'$by using a 1×1 convolution operation. Subsequently, S_5' is upsampled to match the dimensions of S_4'. Finally, the upsampled features S_5' and S_4' are added element-wise to perform the fusion operation, that result in the generation of the new feature P_4. The computation of P_i follows the formula presented in Eq. (1), where $Up(\cdot)$ and $Conv(\cdot)$ represent the upsampling and convolution operations.

$$P_i = Up\left(Conv\left(S_{i+1}\right)\right) + Conv\left(S_i\right) \quad i = 1, \cdots, 4 \tag{1}$$

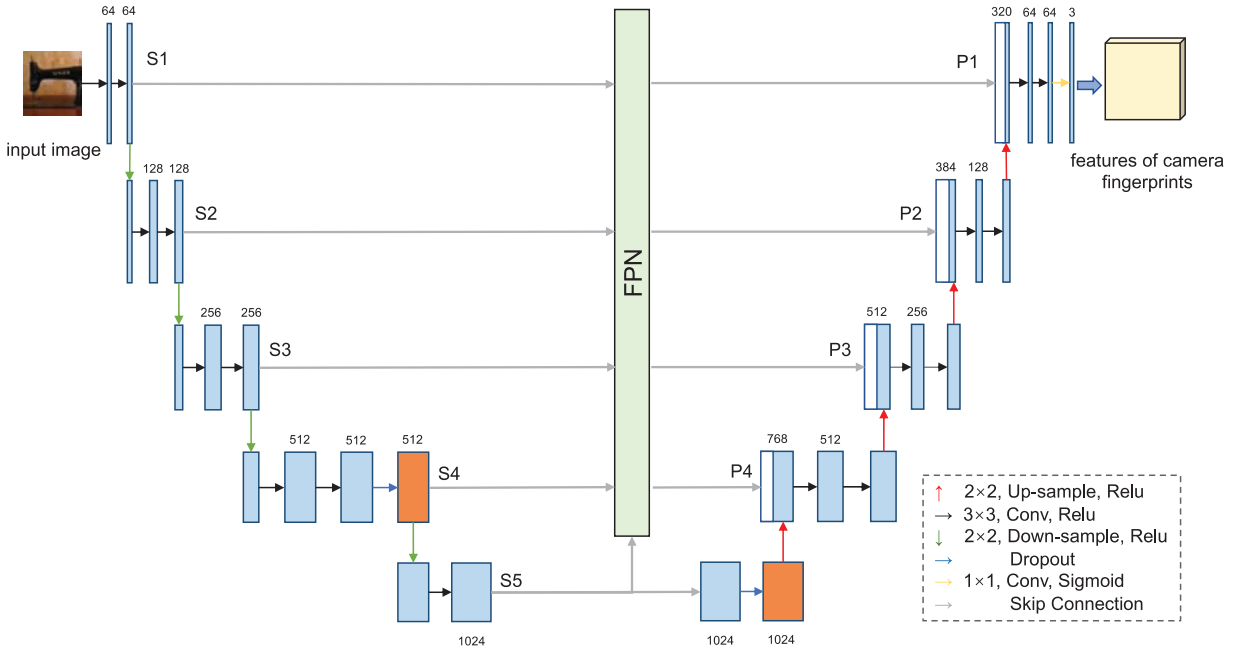


Figure 3: The main structure of the proposed CFUNet

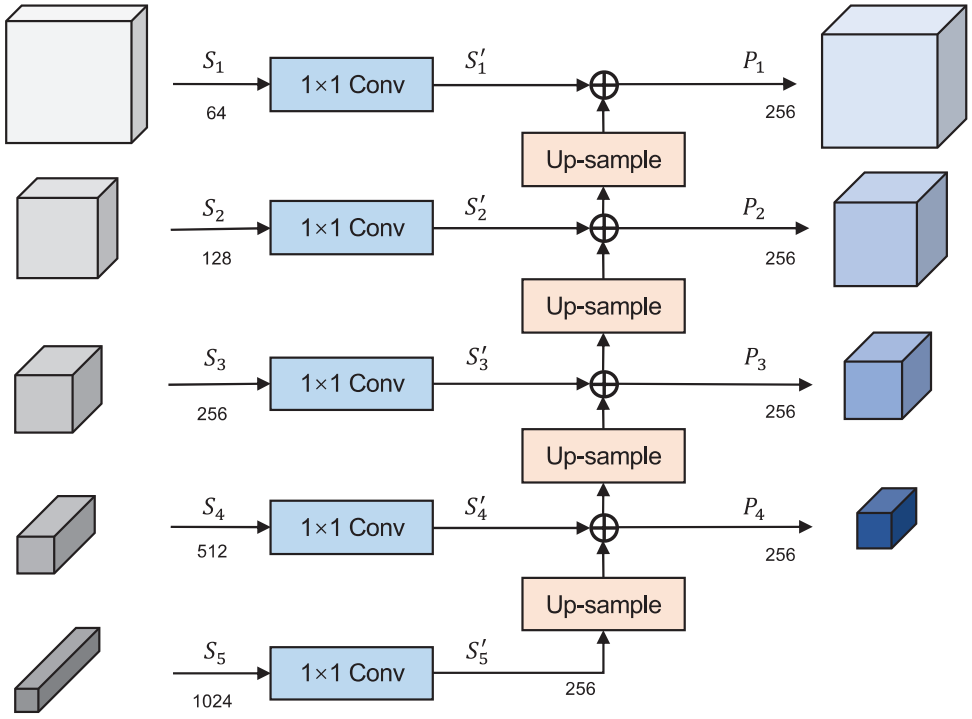


Figure 4: Multi-scale fusion operation for FPN

Moreover, the encoder consists of 5 layers and the features obtained from each layer are fed into the feature fusion module. Firstly, a 1×1 convolution is applied to unify the channel dimension to

256. Then, the new features are upsampled and convolved with 1×1 filters before being added to the features of the previous layer. Finally, the fused features are concatenated with the input features of the corresponding decoder layer for the subsequent decoding operation.

3.2 Feature Fusion Module

The feature fusion module is responsible for merging the multi-scale features generated from different stages of the backbone network and feeding the fused features into the classifier. As shown in Fig. 5, multi-scale features are first transformed to the same dimension through 1×1 convolutions, then pass through Transformer Blocks to achieve the same spatial size. Finally, the features are fed into the Graph Convolutional Network (GCN [21]) module for the final fusion.

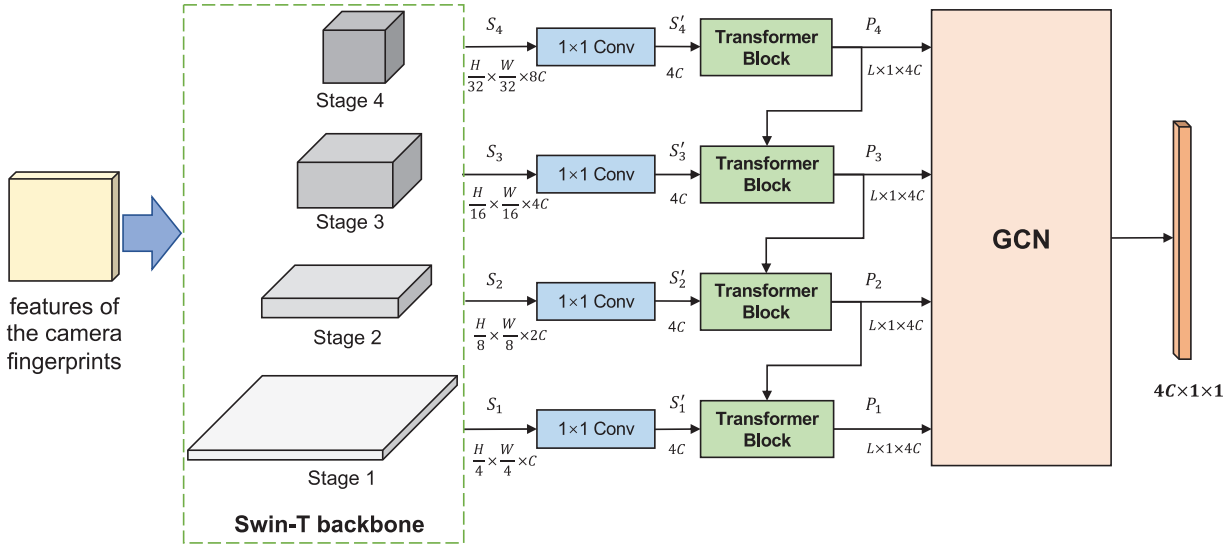


Figure 5: The structure of the proposed feature fusion module

3.2.1 Transformer Block

In particular, the Feed Forward Networks (FFN) is a type of neural network structure that can be represented by Eq. (2), where $Linear(\cdot)$ represents the linear transformation operation, and $Drop(\cdot)$ represents the dropout operation, which helps in the prevention of the network overfitting phenomena.

$$FFN\,(X) = Linear(Drop(X)) \tag{2}$$

The main core of the Transformer Block process is the attention mechanism. To enhance the stability and robustness of the Transformer Block, a multi-head strategy is employed for the attention mechanism. The structures of two different Transformer Blocks are shown in Fig. 6.

Since the attention mechanism takes matrices as inputs rather than tensors, the input features need to be processed before being fed into the Transformer Block. The specific operations are shown in Eqs. (3) and (4), where the term $Flat(\cdot)$ represents the flattening operation of a tensor. X^{self} represents the input for self−attention, and X^{crs} represents the input for cross−attention. Both X^{self} and X^{crs} have the same dimension of $L \times 4C$, where $L = (H/32)*(W/32)$, with L and W are height and width of

the input feature map, respectively.

$$X^{self} = \begin{cases} P_{i+1} & i = 1, 2, 3 \\ Flat\ (S_i^{'}) & i = 4 \end{cases} \tag{3}$$

$$X^{crs} = Flat\ (S_i^{'}) \quad i = 1, 2, 3 \tag{4}$$

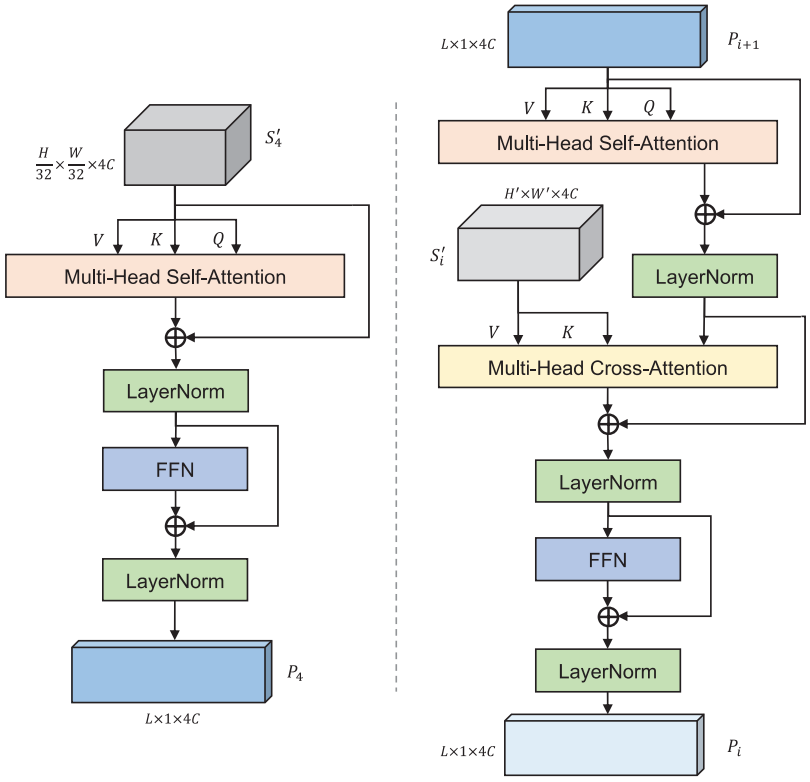


Figure 6: Transformer Block architecture

3.2.2 Graph Convolution Fusion Module

After performing calculations and dimension transformation on the multi-scale features in the Transformer Block, a graph convolution module is used to further fuse these features. This process aims to generate the final feature representation for the source camera classification.

As shown in Fig. 7, the graph convolution fusion module consists of a graph convolution layer. In the graph convolution layer, the adjacency matrix serves as the weight matrix for the convolution operation on the graph structure features. Batch normalization is used to normalize the feature data, and a fully connected layer is applied for classification purposes.

Eq. (5) abstracts the multi-scale features P_1, P_2, P_3, and P_4 into graph G, where G has $4C$ nodes, and each node has a feature dimension of $L/8$. By inserting G into the graph convolution layer, it is expected to obtain the final feature representation Z, which consolidates information from different scale features. The specific calculation formula for Z can be provided as in Eq. (6), where A is the adjacency matrix to represent the structure of the input graph G, $\|$ represents the concatenation operation, $Conv\,(\cdot)$ denotes the convolution operation, $Linear\,(\cdot)$ indicates the linear transformation

operation, $(\cdot)^T$ represents the matrix transpose operation, G and Z have a dimension of $4C \times (L/8)$, and A has a dimension of $4C \times 4C$.

$$G = Linear\left((P_1 \parallel P_2 \parallel P_3 \parallel P_4)^T\right) \tag{5}$$

$$Z = AConv(G) \tag{6}$$

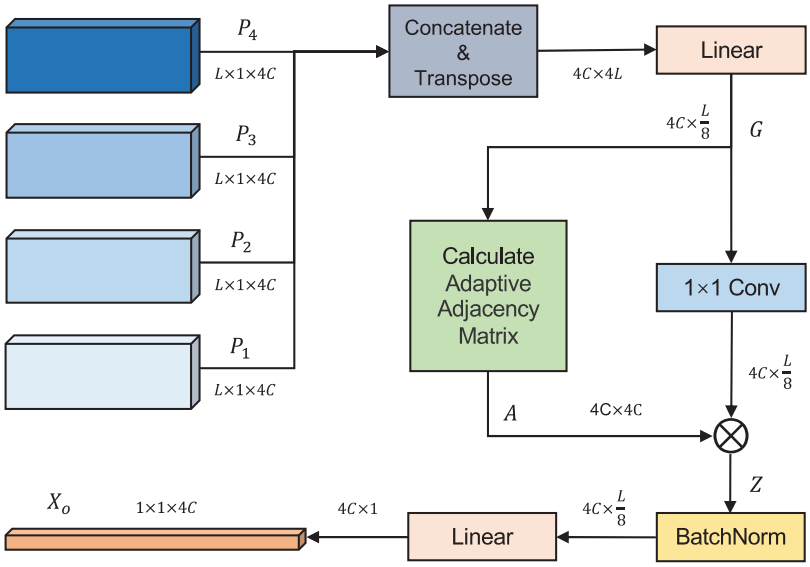


Figure 7: Graph convolution fusion module

In this paper, an adaptive adjacency matrix [22] is employed, which allows the network to learn the relationships between nodes, and therefore compensating for the limitations of extracting features from fixed topological structures. The implementation of the adaptive adjacency matrix involves two learnable parameters, E_1 and E_2, which represent the source and target node embedding, respectively. Firstly, the spatial dependency weights between E_1 and E_2 are computed. Then, the ReLU activation function is applied to eliminate connections with small weights. Finally, the Softmax function is used to normalize the weights, resulting in the adaptive adjacency matrix. The calculation formula is shown in Eq. (7), where E_1 and E_2 have the same dimension of $4C \times (L/8)$.

$$A = Softmax(ReLU(E_1 E_2^T)) \tag{7}$$

The output feature X_o of the graph convolution fusion module is given by Eq. (8), where $BN(\cdot)$ represents batch normalization, and X_o has the dimension of $1 \times 1 \times 4C$.

$$X_o = Linear(BN(Z)) \tag{8}$$

3.3 Network Training

The training of the proposed network consists of two main stages. In the first stage, the camera fingerprint extraction network is trained. In the second stage, the parameters of the camera fingerprint extraction network are fixed, and the entire network is jointly trained.

3.3.1 Camera Fingerprint Extraction Network

During the network training, the pre-trained CFUNet is incorporated into the Siamese network and jointly trained as shown in Fig. 8. The two branches of CFUNet use mean square error (MSE) loss functions, denoted as $\mathcal{L}_{up}$ and $\mathcal{L}_{down}$, as shown in Eqs. (9) and (10), respectively. The classification network uses the cross-entropy (CE) loss function, denoted as $\mathcal{L}_{CE}$, as shown in Eq. (11). Additionally, the contrastive loss function $\mathcal{L}_{contr}$ of the Siamese network is used to compare the output features of the two camera fingerprint extraction networks, as shown in Eqs. (12) and (13).

$$\mathcal{L}_{up} = \frac{1}{N \times C \times H \times W} \sum_{i=1}^{N} \left\| y_{up}^{i} - G\left(x_{up}^{i}\right) \right\|^{2} \tag{9}$$

$$\mathcal{L}_{down} = \frac{1}{N \times C \times H \times W} \sum_{i=1}^{N} \left\| y_{down}^{i} - G\left(x_{down}^{i}\right) \right\|^{2} \tag{10}$$

$$\mathcal{L}_{CE} = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \cdot \log\left(p_i\right) + \left(1 - y_i\right) \cdot \log\left(1 - p_i\right) \right] \tag{11}$$

$$\mathcal{L}_{contr} = \frac{1}{2N} \sum_{i=1}^{N} \left[y_i \cdot D_i^2 + \left(1 - y_i\right) \cdot \max\left(t - D_i, 0\right)^2 \right] \tag{12}$$

$$D_i = \left\| G\left(x_{up}^{i}\right) - G\left(x_{down}^{i}\right) \right\|_2 \tag{13}$$

where, x_{up}^{i} and x_{down}^{i} represent the input noisy images. y_{up}^{i} and y_{down}^{i} are the clean images used as labels. y_i represents the label of sample i, where 1 indicates a positive sample and 0 indicates a negative sample. p_i represents the probability of sample i being predicted as a positive sample. N is the number of samples, and t is the threshold value. Sample i is a five-tuple: $[x_{up}^{i}, x_{down}^{i}, y_{up}^{i}, y_{down}^{i}, y_i]$.

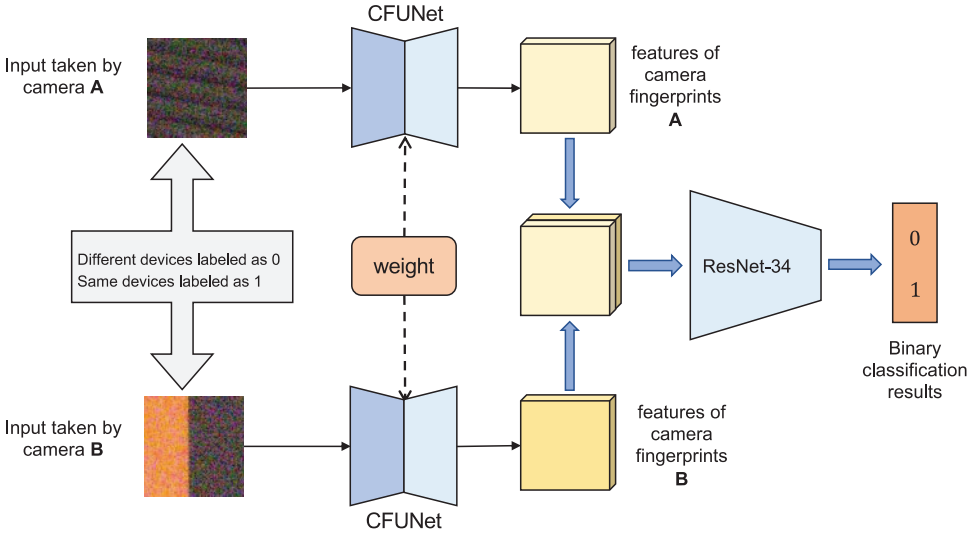


Figure 8: Siamese tuning network

The total loss function, denoted as $\mathcal{L}_{opt}$, is given by Eq. (14). In this equation, λ_{up}, λ_{down}, λ_{CE}, and λ_{contr} are the weights assigned to the $\mathcal{L}_{up}$, $\mathcal{L}_{down}$, $\mathcal{L}_{CE}$, and $\mathcal{L}_{contr}$ loss functions, respectively.

$$\mathcal{L}_{opt} = \lambda_{up}\mathcal{L}_{up} + \lambda_{down}\mathcal{L}_{down} + \lambda_{CE}\mathcal{L}_{CE} + \lambda_{contr}\mathcal{L}_{contr} \tag{14}$$

3.3.2 Camera Fingerprint Classification Network

The CFUNet, the backbone network (Swin-T), the feature fusion module, and a fully connected layer (the classifier) are connected together. The parameters of the camera fingerprint extraction network are fixed, and a joint training is performed.

The network training uses the commonly used cross-entropy loss function for the classification tasks, denoted as $\mathcal{L}$, as shown in Eq. (15).

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n} y_{ij} \cdot \log\left(p_{ij}\right) \tag{15}$$

where, N represents the number of samples, n represents the number of camera classes, y_{ij} represents the one-hot encoded label of sample i, and p_{ij} represents the predicted probability of sample i belonging to a class j.

4 Experimental Results and Analysis

4.1 Datasets

Addressing the issues of the outdated camera device models and improper category divisions in public datasets, this paper creates three subset datasets, namely, Brands, Types, and Devices, which are based on mixture between the public Vision dataset [23] and a custom dataset[†]. The Vision dataset contains 34,427 images captured by 35 different devices, while our custom dataset contains 700 images taken by our research teamwork using 5 devices. We have unified the classification style of the dataset, which is roughly divided into Flat and Nat groups according to the main contents of the images.

Due to the lack of classification information such as brands, models, and individual devices in the camera forensics datasets, we randomly extract 100 fixed-size cropped images from each original image in the mixture. Then, three subset datasets from the custom dataset are created and established based on the device model at different classification granularities: "Brands, Types, and Devices". The classification granularities, from coarse to fine, are illustrated in Fig. 9.

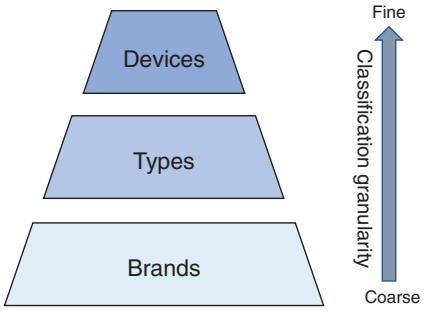


Figure 9: Camera traceability classification granularity

The images in "Brands" dataset are captured by 11 different brands of devices, and the specific details of the brands we used are shown in Table 1. The images in "Types" dataset are captured by 15 models from 3 brands (Samsung, Huawei, OnePlus), as in Table 2. Furthermore, the images in "Devices" dataset are captured by 13 devices from one brand (Apple), as shown in Table 3. Through

[†]Custom Dataset Link: https://www.kaggle.com/datasets/mahmoudemam85/source-camera-identification-a-custom-dataset (accessed on 05 July 2024)

these three created datasets, the performance of different source camera identification algorithms is evaluated in terms of brand-level, model-level, and individual device-level camera attribution.

Table 1: Brands dataset

Devices brand list					
Apple	Huawei	Lenovo	Samsung	Sony	Wiko
LG	Microsoft	OnePlus	Xiaomi	Asus	

Table 2: Types dataset

Devices type list					
Samsung S3 mini	Samsung Tab3	Samsung S3	Huawei P9Lite	Huawei P8	Huawei Honor5c
Samsung trend plus	Samsung S3 mini	Samsung S5	Huawei ascend	OnePlus A3000	OnePlus A3003
Samsung S4 mini	Samsung TabA	Huawei P9			

Table 3: Devices dataset

Devices list					
iPhone 4s	iPhone 5c	iPhone 6	iPhone 5c	iPhone 6	iPhone 5c
iPhone 4	iPhone 4s	iPad 2	iPhone 6Plus	iPad mini	iPhone 5
iPhone 5					

4.2 Implementation Details and Settings

In the experiments, the following settings are used. Operating system: Ubuntu 18.04.5 LTS, Graphics cards: two NVIDIA RTX 3090, Programming Language: Python 3.8.15, Deep Learning Framework: PyTorch 1.12.1. All experiments are conducted on the same platform to maintain the consistency. For the Experimental process: the dataset is firstly augmented by data (mainly cropping, in order to unify the size of the images that input into the proposed network, the size is set to $3 \times 384 \times 384$), and the trainDataloader and valDataloader are constructed, which are randomly selected from the preprocessed dataset according to the ratio of 4:1. The batch size is set to 8, number of epochs is set to 100, learning rate is set to 0.0005, and ADAM optimization method is used. The weights λ_{up}, λ_{down}, λ_{CE}, and λ_{contr} are set to 0.25, 0.25, 0.4, and 0.1, respectively. In the Swin-T backbone network, the intermediate features with dimensions of 2048, 512, 128, and 32 are selected as the input for the multi-scale fusion module in the FPN structure. The dimension C is set to 96, and number of heads in multi-head attention of the Transformer Block is set to 8.

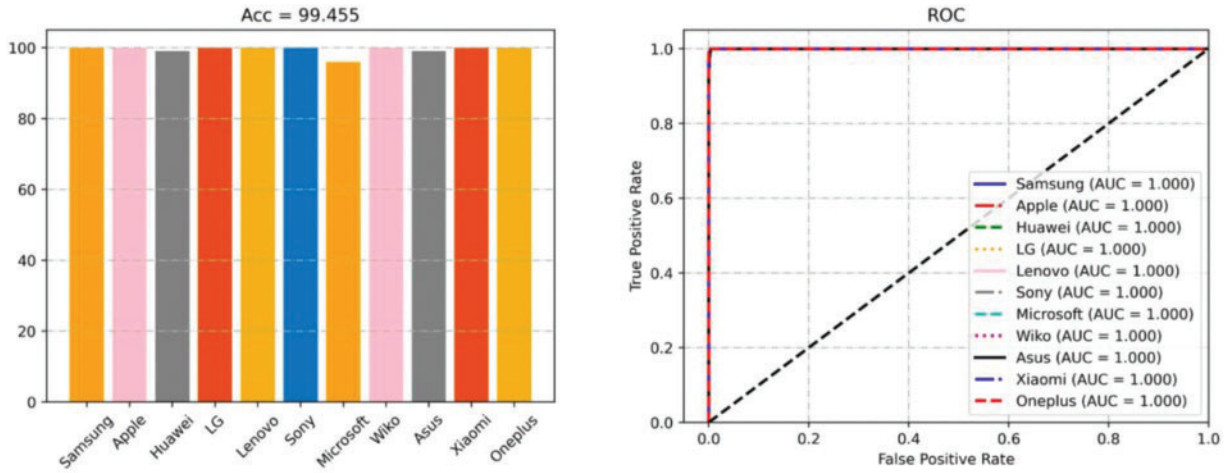
4.3 Performance Evaluation

4.3.1 Main Results

In this subsection, we validate the performance of the proposed model. The experiments are conducted on the Brands, Types, and Devices datasets, and the performance is evaluated based on classification accuracy, multi-class ROC curves, and AUC. The overall classification accuracy results are shown in Table 4. The classification accuracy, AUC values and ROC curves on the three datasets are shown in Figs. 10–12, respectively.

Table 4: Performance evaluation results of camera traceability experiments

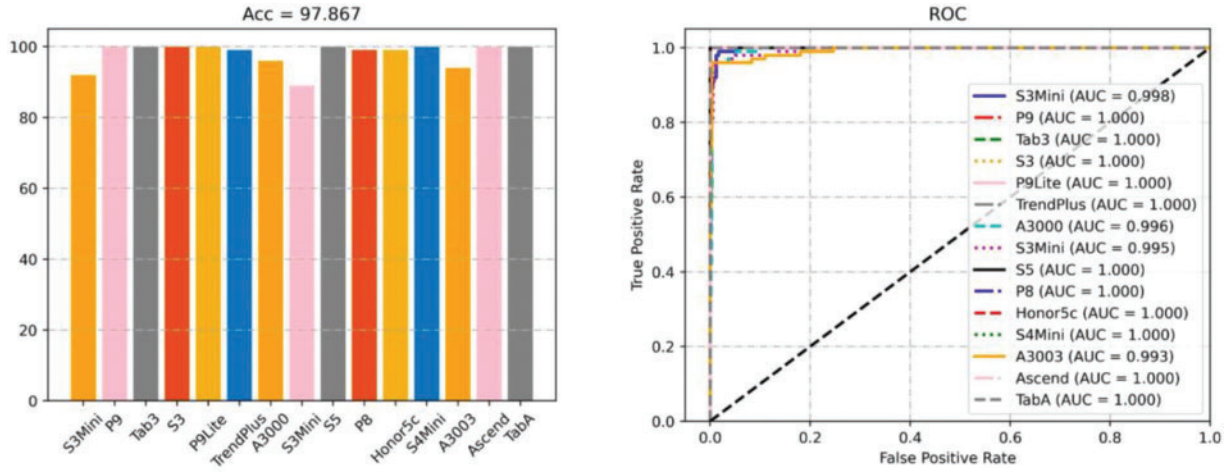
Evaluation metric	Dataset		
	Brands	Types	Devices
Acc (%)	99.46	97.87	91.85



(a) classification accuracy (Acc) for each class. (b) ROC curves and AUC values for each class.

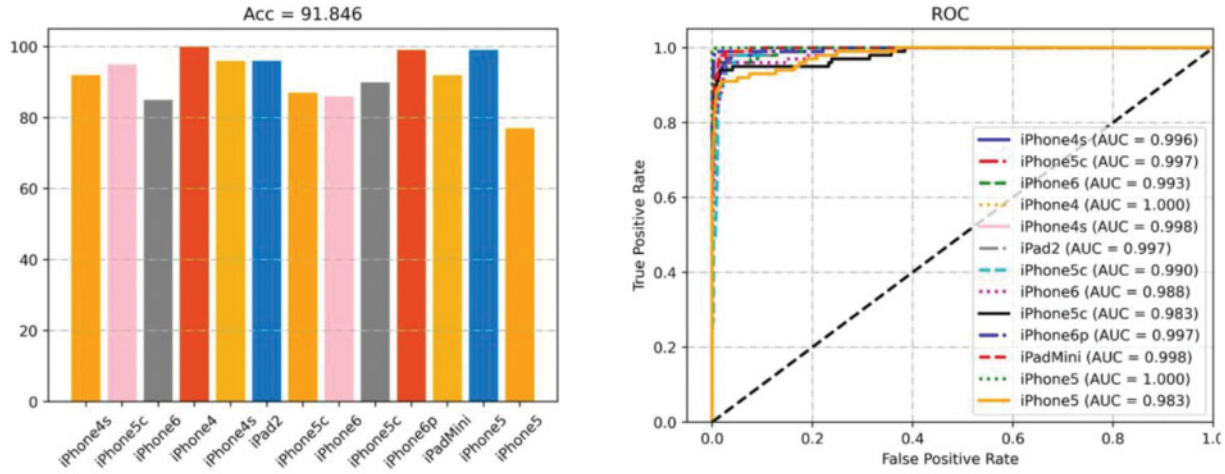
Figure 10: Results of Acc and ROC experiments on brands dataset

It can be observed that the proposed SCI algorithm achieves good classification accuracy of 99.46% and 97.87% on Brands and Types datasets, respectively. Furthermore, on the challenging Devices dataset, it achieves a classification accuracy of 91.85%. Additionally, we also present the confusion matrix for device identification on the Devices dataset, as shown in Fig. 13. The color and number in each cell indicate the probability of the actual class being predicted as current class. Specifically, for different devices of iPhone5c, the proposed algorithm achieves classification accuracies of 95.0%, 87.0%, and 90.0% with AUC values of 0.997, 0.990, and 0.983, respectively. For different devices of iPhone6, it achieves classification accuracies of 85.0% and 86.0% with AUC values of 0.993 and 0.988, respectively.



(a) classification accuracy (Acc) for each class. (b) ROC curves and AUC values for each class.

Figure 11: Results of Acc and ROC experiments on types dataset



(a) classification accuracy (Acc) for each class. (b) ROC curves and AUC values for each class.

Figure 12: Results of Acc and ROC experiments on devices dataset

4.3.2 Ablation Study

Additionally, we conduct three different experiments to validate the impact of the feature fusion module on the proposed multi-scale camera fingerprint classification network CSI-Net. Experiment 1 is conducted without using the feature fusion module and adding only a pooling layer and fully connected layer as classifier after the last stage of Swin-T; Experiment 2 uses the Transformer Block to fuse multi-scale features (FPN with Transformer Block) and adds a fully connected layer as a classifier after the GCN module in the feature fusion module; Experiment 3 uses the FPN to fuse multi-scale features and adds a fully connected layer as a classifier without using GCN module. The main results of the ablation experiments are shown in Table 5.

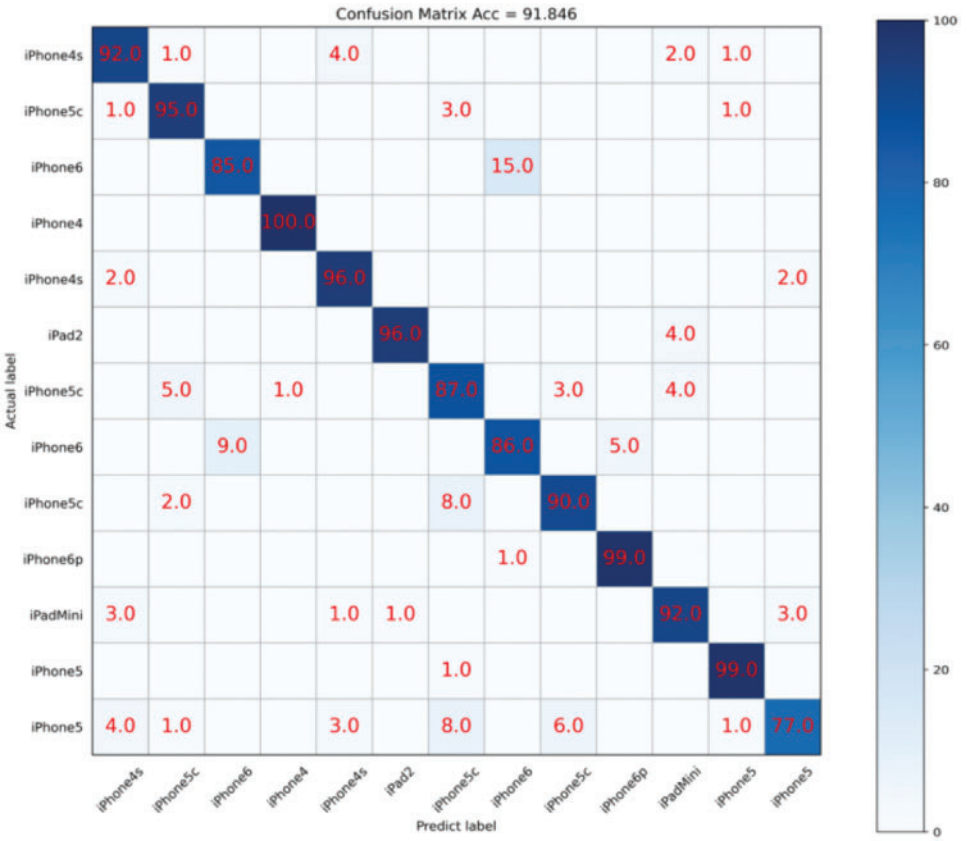


Figure 13: Confusion matrix results on devices dataset

Table 5: The results of the ablation experiments for different feature fusion methods

Experiment/Dataset	Acc (%)		
	Brands	Types	Devices
Experiment 1: Swin-T w/o feature fusion	99.09	97.33	87.02
Experiment 2: Swin-T + FPN with Transformer Block + GCN	99.46	97.87	91.85
Experiment 3: Swin-T + FPN	99.23	97.40	88.07

From Table 5, it can be observed that without adding the proposed feature fusion module, the proposed network achieves classification accuracies of 99.09%, 97.33%, and 87.02% on Brands, Types, and Devices datasets, respectively. After adding the proposed feature fusion module (Transformer Block), the proposed network achieves classification accuracies of 99.46%, 97.87%, and 91.85% on Brands, Types, and Devices datasets, respectively. After adding the feature fusion module (FPN) without GCN, the network achieves classification accuracies of 99.23%, 97.40%, and 88.07% on Brands, Types, and Devices datasets, respectively. Generally, by adding the feature fusion module, there is a certain improvement in the classification accuracy of the Brands, Types, and Devices datasets.

The improvement is relatively small when using only FPN as a multi-scale feature fusion method, while it is more significant when using FPN with Transformer Block and GCN with improvements of 0.37%, 0.54%, and 4.83% on the three datasets, respectively. The experimental results indicate that the proposed network can effectively aggregate features from different levels and improves classification performance by using feature fusion module.

4.3.3 Comparative Experiment

In this subsection, we compare experimental results of the proposed algorithm (CFUNet+CSI-Net) with three SCI algorithms (You et al. [10], Yang et al. [8], and Ding et al. [24]), on Brands, Types and Devices datasets. Moreover, we validate the camera forensics capabilities of different algorithms at the level of different models within the same brand and different devices within the same model. The experimental results in terms of classification accuracy (%) for the comparative experiment on the Brands, Types, and Devices datasets are shown in Table 6.

Table 6: Experimental comparison results of different algorithms

Method/Dataset	Acc (%)		
	Brands dataset	Types dataset	Devices dataset
You et al. [10]	96.49	92.34	79.96
Yang et al. [8]	98.07	93.56	75.85
Ding et al. [24]	98.56	95.09	80.23
Proposed (CFUNet + CSI-Net)	**99.46**	**97.87**	**91.85**

From the experimental results, it can be observed that the proposed SCI algorithm achieves the highest classification accuracy on the established datasets. You et al. 's method [10] has the worst results compared with others, because they only modified bottom layer of U-Net and did not consider the role of shallow features (especially high-frequency noise). In general, the proposed algorithm shows better performance than SOTA methods for source camera identification. In the future, we plan to design an unsupervised algorithm based on the existing supervised camera traceability algorithm. Through unsupervised clustering algorithms, the extracted camera fingerprint features can be clustered to generate pseudo-labels, and then the pseudo-labels are used to train the classification network, and hence the backbone network parameters can be updated through backpropagation. Alternately, we may execute unsupervised clustering tasks and classification tasks and perform algorithm iteration to complete the clustering of unseen data. When the algorithm is iterating, it only needs to input the dataset of the image to be traced, and no additional category labels are required to complete the clustering of camera traceability task.

5 Conclusion and Future Work

This paper proposes a novel source camera identification algorithm based on multi-scale feature fusion. The multi-scale features from different levels of the encoder are recursively fused to enhance feature extraction capability and improve the classification performance of the proposed network. To evaluate the effectiveness of the proposed model, extensive experiments are conducted on established datasets. The comparative experiments show the effectiveness of the proposed algorithm compared with state-of-the-art algorithms. Additionally, it performs well on the established challenging Devices

dataset, and it can effectively distinguish between different individual camera devices. However, the datasets used for evaluating the performance of the proposed algorithm are relatively small. In real-world scenarios, the number of brands, models, and individual camera devices is much larger. Therefore, building a more diverse dataset that includes a wide range of brands, models, and individual devices is an important task for future research. Moreover, further studies are required to enhance the network and improve its performance for individual-level source camera identification and improve its capabilities in real-world scenarios.

Acknowledgement: The authors would like to thank the editorial board and all anonymous reviewers for their helpful comments and suggestions.

Funding Statement: This work was funded by the National Natural Science Foundation of China (Grant No. 62172132), Public Welfare Technology Research Project of Zhejiang Province (Grant No. LGF21F020014), and the Opening Project of Key Laboratory of Public Security Information Application Based on Big-Data Architecture, Ministry of Public Security of Zhejiang Police College (Grant No. 2021DSJSYS002).

Author Contributions: Jianfeng Lu: Conceptualization, Methodology, Supervision, Project administration, Funding acquisition, Validation, Writing—review editing. Caijin Li: Conceptualization, Methodology, Software, Data curation, Validation, Resources, Writing—original draft, Writing—review editing, Visualization. Xiangye Huang: Software, Validation, Conceptualization, Methodology, Resources, Writing–original draft, Writing—review editing, Visualization. Chen Cui: Validation, Software, Resources, Data curation, Writing—review editing, Visualization. Mahmoud Emam: Conceptualization, Validation, Methodology, Writing—original draft, Writing—review editing, Formal analysis, Visualization. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data are available upon request to the corresponding author.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

References

[1] J. Bernacki, "A survey on digital camera identification methods," *Forensic Sci. Int. Digit. Investig.*, vol. 34, pp. 300983, Sep. 2020. doi: 10.1016/j.fsidi.2020.300983.

[2] C. T. Li Manisha, X. Lin, and K. A. Kotegar, "Beyond PRNU: Learning robust device-specific fingerprint for source camera identification," *Sensors*, vol. 22, no. 20, pp. 7871, Oct. 2022. doi: 10.3390/s22207871.

[3] B. Wang, M. Zhao, W. Wang, X. Dai, Y. Li and Y. Guo, "Adversarial analysis for source camera identification," *IEEE Trans. Circ. Syst. Video Technol.*, vol. 31, no. 11, pp. 4174–4186, Dec. 2020. doi: 10.1109/TCSVT.2020.3047084.

[4] L. Bondi, L. Baroffio, D. Guera, P. Bestagini, E. J. Delp and S. Tubaro, "First steps toward camera model identification with convolutional neural networks," *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 259–263, 2017. doi: 10.1109/LSP.2016.2641006.

[5] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," in *2016 IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Abu Dhabi, United Arab Emirates, 2016, pp. 1–6. doi: 10.1109/WIFS.2016.7823908.

[6] J. Chen, X. Kang, Y. Liu, and Z. J. Wang, "Median filtering forensics based on convolutional neural networks," *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 1849–1853, Nov. 2015. doi: 10.1109/LSP.2015.2438008.

[7] B. Bayar and M. C. Stamm, "Augmented convolutional feature maps for robust CNN-based camera model identification," in *2017 IEEE Int. Conf. Image Process. (ICIP)*, Beijing, China, 2017, pp. 4098–4102. doi: 10.1109/ICIP.2017.8297053.

[8] P. Yang, R. Ni, Y. Zhao, and W. Zhao, "Source camera identification based on content-adaptive fusion residual networks," *Pattern Recognit. Lett.*, vol. 119, pp. 195–204, Mar. 2019. doi: 10.1016/j.patrec.2017.10.016.

[9] Z. Zhang, Y. Zhang, Z. Zhou, and J. Luo, "Boundary-based image forgery detection by fast shallow CNN," in *2018 24th Int. Conf. Pattern Recognit. (ICPR)*, Beijing, China, 2018, pp. 2658–2663. doi: 10.1109/ICPR.2018.8545074.

[10] C. You, H. Zheng, Z. Guo, T. Wang, and X. Wu, "Multiscale content-independent feature fusion network for source camera identification," *Appl. Sci.*, vol. 11, no. 15, pp. 6752, 2021. doi: 10.3390/app11156752.

[11] A. M. Rafi, T. I. Tonmoy, U. Kamal, Q. J. Wu, and M. K. Hasan, "RemNet: Remnant convolutional neural network for camera model identification," *Neural Comput. Appl.*, vol. 33, no. 8, pp. 3655–3670, Apr. 2021. doi: 10.1007/s00521-020-05220-y.

[12] G. S. Bennabhaktula, E. Alegre, D. Karastoyanova, and G. Azzopardi, "Camera model identification based on forensic traces extracted from homogeneous patches," *Expert. Syst. Appl.*, vol. 206, Nov. 2022. doi: 10.1016/j.eswa.2022.117769.

[13] K. Rana, P. Goyal, and G. Sharma, "Dual-branch convolutional neural network for robust camera model identification," *Expert. Syst. Appl.*, vol. 238, no. 3, pp. 121828, Mar. 2024. doi: 10.1016/j.eswa.2023.121828.

[14] C. S. Sychandran and R. Shreelekshmi, "SCCRNet: A framework for source camera identification on digital images," *Neural Comput. Appl.*, vol. 36, no. 3, pp. 1167–1179, Jan. 2024. doi: 10.1007/s00521-023-09088-6.

[15] J. Bernacki, "Digital camera identification by fingerprint's compact representation," *Multimed. Tools Appl.*, vol. 81, no. 15, pp. 21641–21674, 2022. doi: 10.1007/s11042-022-12468-0.

[16] X. Liu, L. Zhang, T. Li, D. Wang, and Z. Wang, "Dual attention guided multi-scale cnn for fine-grained image classification," *Inf. Sci.*, vol. 573, no. 4, pp. 37–45, Sep. 2021. doi: 10.1016/j.ins.2021.05.040.

[17] Z. Liu *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *2021 IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, 2021, pp. 9992–10002. doi: 10.1109/ICCV48922.2021.00986.

[18] D. Chicco, "Siamese neural networks: An overview," in H. Cartwright (Eds.), *Artificial Neural Networks. Methods in Molecular Biology*, New York, NY: Humana, 2021, vol. 2190, pp. 73–94. doi: 10.1007/978-1-0716-0826-5_3.

[19] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *2020 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, 2020, pp. 9726–9735. doi: 10.1109/CVPR42600.2020.00975.

[20] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, USA, Honolulu, HI, 2017, pp. 936–944. doi: 10.1109/CVPR.2017.106.

[21] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: A comprehensive review," *Comput. Soc. Netw.*, vol. 6, no. 1, pp. 1–23, Dec. 2019. doi: 10.1186/s40649-019-0069-y.

[22] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, Macao, China, 2019. pp. 1907–1913.

[23] D. Shullani, M. Fontani, M. Iuliani, O. A. Shaya, and A. Piva, "VISION: A video and image dataset for source identification," *EURASIP J. on Info. Security*, vol. 2017, no. 15, pp. 1–16, 2017. doi: 10.1186/s13635-017-0067-2.

[24] X. Ding, Y. Chen, Z. Tang, and Y. Huang, "Camera identification based on domain knowledge-driven deep multi-task learning," *IEEE Access*, vol. 7, pp. 25878–25890, Feb. 2019. doi: 10.1109/ACCESS.2019.2897360.