IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 19, 2020, accepted November 9, 2020, date of publication November 16, 2020, date of current version December 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3037735

Manipulation Classification for JPEG Images Using Multi-Domain Features

IN-JAE YU[1], SEUNG-HUN NAM[1], WONHYUK AHN[1], MYUNG-JOON KWON[1], AND HEUNG-KYU LEE[1,2]

[1]School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea
[2]Digital Innotech Company, Daejeon 34184, South Korea

Corresponding author: Heung-Kyu Lee (heunglee@kaist.ac.kr)

This work was supported in part by the Institute for Information and Communications Technology Planning and Evaluation (IITP) funded by Minister of Science and ICT (MIST), Korea Government (Development of high reliability image and video authentication service for smart media environment) under Grant 2017-0-01671, and in part by the National Research Foundation of Korea (NRF) under Grant NRF-2019R1A2C2084569.

ABSTRACT Image forensics comprises the analyses and classifications of manipulations that have been applied to images. The ability to classify various manipulations that have been employed in the process of forgery is essential. Techniques to identify multiple manipulations applied to uncompressed images have been reported thus far, but the forensic approach for JPEG images compressed with various qualities has not been proposed. In this paper, we propose the manipulation classification network (MCNet) to exploit multi-domain features of the spatial, frequency, and compression domains. The proposed MCNet learns several forensic features for each domain through a multi-stream structure and distinguishes manipulations by comprehensively analyzing the fused features. Our work jointly considers visual artifacts caused by image manipulations and compression artifacts due to JPEG compression; therefore, rich forensic features can be explored and learned in the training phase. To enable forgery analysis in the real-world environment, data were generated based on twenty types of manipulation algorithms and various compression parameters. To demonstrate the effectiveness of the proposed MCNet, extensive experiments were conducted using state-of-the-art baselines. Compared to these baselines, our proposed method outperforms in terms of multi-class manipulation classification. In addition, we experimentally proved that the fine-tuned model based on the multi-class manipulation task was effective for different forensic tasks such as DeepFake detection or integrity authentication of JPEG images.

INDEX TERMS Image forensics, manipulation classification, convolutional neural network (CNN), multi-domain features, JPEG compression.

I. INTRODUCTION

Editing images has become easier than ever before because of the development and distribution of smartphones with high-end cameras and various editing applications. Moreover, advances in content sharing platforms and social network services (e.g., Instagram, Twitter, and Facebook) have enabled people to share their own images easily through mobile devices [1]. Before sharing such media content, people usually edit their original images to make them appear better or attractive. In the past, editing tools such as Photoshop were considered exclusively by experts, but in recent years, editing applications have been developed that even nonprofessionals can easily use on their mobile devices. These editing applications include a variety of image manipulations [2]–[4], and the range of manipulation types is wide: image blurring, noise addition, contrast-change-based image enhancement, morphing, and image resampling, among others. In particular, since the detailed algorithms for the manipulation types are different for each editing application, the images being distributed currently contain fine-grained and various other manipulation traces [5].

The representative field of technology that addresses various image manipulations is image forensics, which is aimed at verifying the authenticity of given images. Image forensics is a field wherein the subtle traces of forgery remaining in images under various conditions (e.g., manipulation type, algorithm, parameters, and compression quality) are analyzed

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Imran Tariq.

and detected [6], [7]. In particular, it is possible to verify the integrity of an image using forensic techniques to determine whether there are manipulations in the images [8]. As mentioned above, owing to the rapid proliferation of mobile devices and user-friendly editing tools, the amount of manipulated content is increasing; hence, image forensics is rapidly becoming an important field of study [9]. Since forensic techniques are mostly aimed at capturing the fine statistical changes caused by manipulations that are difficult to distinguish visually, various approaches for extracting and exploring the low-level features of images are being studied constantly [10], [11].

The manipulation process causes changes in the underlying characteristics of the pristine content, and forensic techniques have to be designed to extract and analyze such low-level traces. In the past, handcrafted feature-based approaches that specialized in capturing statistical fingerprints and methods to analyze specific patterns retained in images after manipulations were the dominant detection techniques [12]–[14]. Recently, advances in convolutional neural networks (CNNs) have had positive effects on the field of computer vision, and forensic techniques inspired by neural networks have been studied actively [15], [16]. Within a data-driven learning framework, the CNN-based forensic approaches efficiently learn the forensic features associated with images and employ them to identify forged content. The neural-network-based forensic techniques may be applied to the following tasks: manipulation classification [2], [3], [5], [17], compressed artifact detection [10], [11], [18], [19], median filtering detection [20], [21], model identification [22]–[24], DeepFake detection [25]–[27], steganalysis [28], [29], and content-aware image retargeting detection [8]. In particular, manipulation classification distinguishes the various types of manipulations that are retained in an image, and its importance is increasing as the different types of manipulations are diversifying with the development of image editing applications.

As the importance of manipulation classification emerges, high-performance CNN-based approaches [2], [3], [5] for identifying manipulated images by learning the generated data through various manipulations and parameters have been proposed. These approaches are suitable for capturing the manipulations applied to uncompressed images, but have limitations in that they only consider image compression (i.e., JPEG compression) as one of the manipulation types. In other words, they do not consider real-world manipulation in which image compression having different qualities are applied after manipulation to uncompressed images. Typically, after completing forgery using an editing tool, the manipulated images are distributed in compressed formats to reduce the overheads of both storage and network traffic [30]. In the uncompressed images, traces of the manipulations containing mainly high-frequency components remain; however, such fine fingerprints are modified in JPEG images, so it is more difficult to classify manipulations after JPEG compression [11].

The most widely used compression standard for images is JPEG, which is a form of lossy compression, and is used to quantize or remove high-frequency components in data. As reported in [10], most digital images are compressed using the JPEG format when they are uploaded online, which eliminates or mitigates high-frequency signals within images. Briefly, JPEG compression converts a given image into 8×8 blocks and applies the 2D discrete cosine transform (DCT) to each block [11], [30]. Then, the DCT coefficients are quantized using a predefined 8×8 quantization table corresponding to the compression factor. These processes modify the high-frequency components, i.e., the minute signals that are difficult to recognize using the human visual system (HVS); hence, compression artifacts that can be distinguished from the uncompressed images remain in the JPEG images. In real-world scenarios that consider various types of manipulations and JPEG compression simultaneously, the quantization process of the JPEG scheme also affects the traces retained in the image, which renders it difficult to classify manipulations in compressed images [10].

From the examples in Fig. 1, it can be seen that the traces retained from manipulations are affected by JPEG compression. Figs. 1(b) and 1(e) are the results of uniform noise addition to and contrast-limited adaptive histogram equalization (CLAHE) on the original uncompressed image, respectively. To analyze the effects of JPEG compression on the signals applied to the original images via manipulation, we generated the residual images before and after compression application. Comparing the magnified areas at the bottom right of Figs. 1(c), 1(d), 1(f), and 1(g), it can be seen that the high-frequency components of the manipulation traces are lost or modified by JPEG compression. Particularly, by examining Figs. 1(d) and 1(g) in detail, it can be observed that block artifacts exist in the image through 8×8 block-based quantization of the JPEG compression procedure. Thus, forensic approaches to classifying real-world manipulations should jointly consider the artifacts arising from JPEG compression and traces of various manipulations.

To address this, we propose a manipulation classification network (MCNet) that classifies the various types of image manipulations on the premise that JPEG compression has been applied. The training of the proposed MCNet consists of two phases: i) learning forensic features for the multi-domain through a multi-stream structure and ii) performing multi-class classification by comprehensive learning using the fused features. In the first training phase, our proposed techniques learn several forensic features for multiple domains, including spatial, frequency, and compression domains, through a multi-stream structure. The multi-stream structure is composed of a visual artifact network (VANet) and compression artifact network (CANet) that explore and learn the visual and compression artifacts, respectively. After training the VANet and CANet, we fused the feature maps obtained from each of these pretrained networks, and the ensemble network was applied to comprehensively learn the multi-domain features. Our work considers

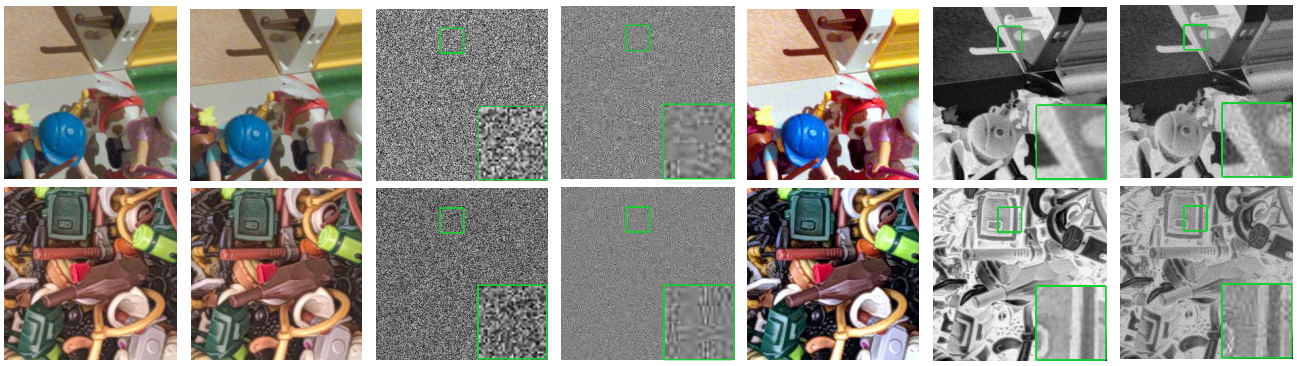


FIGURE 1. Analysis of the effects of compression on manipulated images: (a) original uncompressed image, (b) uncompressed result of uniform noise addition, (c) residual image between (a) and (b), (d) residual image between (a) and JPEG compressed (b), (e) uncompressed result of contrast-limited adaptive histogram equalization, (f) residual image between (a) and (e), and (g) residual image between (a) and JPEG compressed (e). The green bordered areas at the bottom right of (c), (d), (f), and (g) are the results of enlarging the green bordered areas at the tops of the respective images.

both visual artifacts for image manipulation and compression artifacts due to JPEG compression jointly; hence, rich forensic features of real-world manipulations can be explored and learned.

To improve the performance of multi-class classification on various manipulations with our MCNet, we generate a dataset using twenty manipulation algorithms from five manipulation types, including image blurring, noise addition, contrast-change-based image enhancement, morphing, and image resampling. In particular, considering real-world situations where manipulations and JPEG compression are simultaneously applied to images, the generated manipulated images were compressed with various quality factors. We assume that our model, which is trained on the rich forensic features of manipulations close to the real environment, can help obtain general discriminability for various forensics tasks. It has a similar context as those of CNN models pretrained with ImageNet [31] have higher and generalized performances in various high-level vision tasks. Based on the experimental results for other forensic tasks (i.e., DeepFake detection and image integrity authentication), we claim that fine-tuning the model pretrained with our manipulation dataset to the target domain helps improve detection performance. The main contributions of this paper are summarized as follows:

- This work is the first attempt to classify real-world image manipulations that jointly consider compression and manipulation, including various compression quality and manipulation algorithms.
- Compared to the available baselines [3], [5], [25], [28], the proposed MCNet, which effectively learns and explores visual and compression artifacts based on multi-domain features, achieves state-of-the-art performance.
- We experimentally prove that the fine-tuned model, based on the multi-class manipulation task, which learns rich forensic features for various manipulations, is effective for other forensic tasks such as DeepFake detection or image integrity authentications.

The remainder of this paper is organized as follows. Section II presents a review of the relevant existing work on classifying manipulated content and presents the manipulation environment considered in our work. The proposed forensic approach is presented in Section III, and the performance of the proposed MCNet is demonstrated in Section IV. Finally, Section V presents some conclusions of this work.

II. BACKGROUND

In this section, we review the available forensic approaches related to our work and introduce the manipulation environment considering real-world situations where various types of manipulations and image compressions coexist.

A. RELATED WORK

With the advances in content sharing platforms and easy-to-use editing applications, the era of easy digital image acquisition, editing, and sharing is upon us. Although these technological progresses have benefited the lives of people, they have also caused social problems such as fake news by owing to forged content. To handle these, research on image forensics have been conducted, and this field aims to authenticate the integrity of given images by exploring or identifying the manipulation artifacts contained in them [1], [6]. Since the manipulated content is elaborately edited so as not to be distinguishable from natural images, approaches that can focus on the changes in the fine statistical patterns are required to distinguish manipulations [7]. In the past, this task was dominantly achieved using handcrafted feature-based approaches specializing in capturing statistical fingerprints and methods to analyze specific patterns retained in the manipulated images [12]–[14].

Inspired by high-level vision tasks [31] that have achieved success using deep-learning frameworks, various forensic approaches to enable the network to learn forensic features

automatically are actively presented. While neural networks for computer vision tasks are capable of learning rich features, in their general form, they tend to learn the high-level features (e.g., dogs versus cats) of the content in the given images [28]. To cope with this, CNN-based forensic techniques have been designed to learn the low-level features of manipulations while suppressing the content information of the given image [5]. This includes an approach that exploits data (e.g., preprocessed data or handcrafted feature) generated through domain knowledge as input to the network and an end-to-end learning approach that automatically learns the forensic features from pixel data.

Detection of the fine manipulation traces by focusing on the high-frequency signals (i.e., noise residual) has been proposed by works that have examined preprocessed data utilizing high-pass filters (HPFs) or steganalysis-rich model (SRM) kernels [32] as inputs to CNNs. In [20], Chen *et al.* presented a CNN-based median filtering detection approach utilizing residual data, and Mo *et al.* exploited the HPF-based preprocessing method to identify fake faces [26]. The authors in [18], [19] proposed neural networks that utilized noise residuals to explore the compression artifacts retained after lossy compression of the H.264 codec. There are also forensic approaches that learn the traces of manipulation from handcrafted features generated using domain knowledge. In [11], Barni *et al.* experimentally demonstrated that a CNN employing the DCT histogram feature could distinguish between single- and double-compressed JPEG images. Subsequently, Park *et al.* presented a novel CNN to detect the double-compressed JPEG content using histogram features with vectorized quantization table [10]. In [33], the authors demonstrated that preprocessing, which enhances texture information of the luminance components, can provide CNNs with rich forensic features of computer graphics.

Recently, researchers have proposed CNN-based forensic systems for an end-to-end learning approach that does not require predetermined features or preprocessing steps. A novel perspective of this is considered in [17], where analysis on whether the CNN structure can replicate the behavior of the conventional rich-model classifier was conducted. In [2], [5], Bayar and Stamm proposed a constrained convolutional-layer-based MISLNet that forces the CNN to learn prediction error filters. This model can classify multiple types of manipulations applied to uncompressed images with high accuracy; follow-up studies [3], [4], [21] are therefore currently being proposed inspired by [2], [5]. Instead of relying solely on the constrained layer, the CNN-based learning methodologies that identify manipulations in an end-to-end fashion are proposed. In particular, networks that effectively detect content-based forgeries, such as DeepFake [27], [34] and content-aware retargeting [9], [18], are also being proposed, and Xception [25] has achieved stable performance in both tasks to extract artifacts caused by content reconstruction.

Steganalysis refers to the study of capturing hidden messages inserted by steganography [32], and CNN-based steganalysis has similarities to the forensic task in that it captures the subtle disturbances that are distinct from the cover image. Using this perspective, Zhan *et al.* proposed a transfer learning approach for forensic tasks, which reveals the transferability between the forensic and steganalysis models [35]. Furthermore, Boroumand *et al.* introduced SRNet [28], which is an end-to-end framework composed of carefully designed blocks for steganalysis in both the spatial and JPEG domains, and the SRNet showed impressive performance even in forensic tasks [9].

Similar to high-level vision tasks such as object detection [36], which is fine-tuned from a network trained for the ImageNet [31] dataset, researchers have proposed image forgery localization technique that has been fine-tuned from pretrained networks. In [37], a pretrained network was used for ImageNet and SRM kernel [32] to detect the locations of forgery, such as splicing and copy-move. Thereafter, Wu *et al.* [3] proposed MantraNet, which is a two-stage network comprising manipulation classification and forgery localization networks. They adopted the SRM kernel [32] and constrained layers [5] to extract the traces of manipulation. It is a first work that classifies image manipulations containing various algorithms successfully. Since they use compression as one of the manipulation types, the performance of the forgery localization network is not robust to image with compression applied.

In summary, as editing tools and sharing services become more common, images are being edited using various types of manipulations [3], [5] and distributed in the compressed form [10]. To the best of our knowledge, manipulation scenarios in which various manipulations and JPEG compression are used together have not been considered until now. To address this issue, we attempted to classify various image manipulation algorithms, as listed in Table 1, on the premise that JPEG compression with various compression factors has been applied. Details are introduced in the following section.

B. MANIPULATION ENVIRONMENT

In this section, we describe the manipulation environment applied to generate an image dataset that reflects the real-world manipulation scenarios. We employ five types of manipulations that are frequently used when editing images: image blurring, noise addition, contrast change, image morphing, and image resampling. Each manipulation type contains several manipulation algorithms, for a total of twenty algorithms, and Fig. 2 illustrates examples of each manipulation algorithm considered in this work. In addition, the implementation details (e.g., manipulation type, algorithm name, library, function, and parameters) used to generate the manipulated dataset are listed in Table 1. As mentioned in Section II-A, unlike previous works [2], [3], [5] that consider JPEG compression as a type of manipulation, we consider realistic manipulation scenarios in which image manipulation

TABLE 1. Implementation details used to generate manipulated images. Each image is manipulated using parameter *p* following description.

Manipulation type	Algorithm	Library	Function	Parameters (p)	Description
Image blurring	Box blur	*Python Image Library (PIL)*	*ImageFilter.BoxBlur*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	Kernel size
	Gaussian blur	*Python Image Library (PIL)*	*ImageFilter.GaussianBlur*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	
	Median filter	*Python Image Library (PIL)*	*ImageFilter.MedianFilter*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	
	Wavelet denoiser	*Scikit-image (skimage)*	*denoise_wavelet*	[1, 2, 3, 4, 5]	threshold = $\sigma(I) \times \frac{1}{p}$
Noise addition	Gaussian noise	*Scikit-image (skimage)*	*random_noise('gaussian')*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	Add noise of $N(0, p)$
	Salt & Pepper noise	*Scikit-image (skimage)*	*random_noise('s&p')*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	Change $p\%$ pixels
	Poisson noise	*Scikit-image (skimage)*	*random_noise('poisson')*	-	
	Uniform noise	*Scikit-image (skimage)*	*random_noise('uniform')*	[3, 5, 7, 9, 11, 13, 15, 17, 19, 21]	Add noise of $U(-p, p)$
Contrast change	Auto contrast	*Python Image Library (PIL)*	*ImageOps.autocontrast*	[1, 2, 3, 4, 5, 6, 7]	Cut-off percentage $p\%$
	CLAHE	*OpenCV*	*cv2.createCLAHE*		
Image morphing	Opening	*OpenCV*	*cv2.MORPH_OPEN*	[3, 5, 7, 9, 11, 13]	Kernel size
	Closing	*OpenCV*	*cv2.MORPH_CLOSE*		
	Gradient	*OpenCV*	*cv2.MORPH_GRADIENT*		
	Dilation	*OpenCV*	*cv2.MORPH_DILATE*		
	Erosion	*OpenCV*	*cv2.MORPH_ERODE*		
Image resampling	Bi-linear	*Python Image Library (PIL)*	*Image.BILINEAR*	[0.51, 0.62, 0.73, 0.84, 0.95, 1.05, 1.16, 1.27, 1.38, 1.49]	Target size=$Hp \times Wp$
	Hamming	*Python Image Library (PIL)*	*Image.HAMMING*		
	Bicubic	*Python Image Library (PIL)*	*Image.BICUBIC*		
	Nearest neighbor	*Python Image Library (PIL)*	*Image.NEAREST*		
	Lanczos	*Python Image Library (PIL)*	*Image.LANCZOS*		

and JPEG compression occur in succession. To do this, we applied specific manipulation algorithms to a given image with a random parameter, and JPEG compression with an arbitrary compression factor in the range of 75 to 95 was subsequently applied to the manipulated result. A brief review of each manipulation type and JPEG compression is as follows.

1) IMAGE BLURRING

is used to remove high-frequency signals in an image, and we consider four image blurring algorithms. Box blur (BB) is a linear filter in which each pixel of the blurred image has the average of its neighboring pixels in the input image. Gaussian blur (GB), which is a linear filter, removes noise-like signals by weakening the high-frequency components. Median filter (MF) is a representative non-linear filter, and wavelet denoiser (WD) is used to remove noise from an image using the discrete wavelet transform.

2) NOISE ADDITION

refers to the addition of high-frequency signals (i.e., noise-like signals) that are irrelevant to the content information of images. The present study employed four algorithms for noise addition. Gaussian noise (GN) is added as noise to every pixel independently, and the noise values follow a normal distribution. Salt & Pepper noise (SP) is based on random pixels having maximum and minimum values, which makes the image look like it has scattered salt and pepper over it. Poisson noise (PN) is modeled by a Poisson process, which is more noticeable when the overall brightness is low. For uniform noise (UN), the density function follows a uniform distribution.

3) CONTRAST CHANGE

refers to an image adjustment technique by which the color distribution in the image is changed. To change the image contrast, we consider the following two algorithms: auto contrast (AC) and contrast-limited adaptive histogram equalization (CLAHE). For AC, the histogram of the given image is computed, a cutoff percentage of the lightest and darkest pixels from the histogram are removed, the image is remapped. CLAHE is similar to AC, but it applies histogram equalization to sub-blocks of the image rather than the global area.

4) IMAGE MORPHING

is the technique of changing or morphing the image shape for metamorphosis (e.g., shrinking or expanding the boundary of the object). The following five representative morphing algorithms were considered: opening (OP), closing (CL), gradient (GR), dilation (DL), and erosion (ER). In our work, we randomly selected kernel shapes (i.e., rectangle, ellipse, and cross) of arbitrary kernel sizes for applying each of the morphing algorithms to the given image.

5) IMAGE RESAMPLING

involves enlarging or reducing image resolution by generating a new version of the image with different width or height in pixels. This study considers five algorithms for image resampling: bi-linear (BL), hamming (HM), bicubic (BC), nearest neighbor (NN), and Lanczos (LC) algorithms. To cover both increasing and decreasing image sizes, we applied the up-sampling and down-sampling approaches from 0.5 to 1.5 of the original resolution of the given image.

6) JPEG COMPRESSION

is a representative lossy compression for digital images that mitigates or removes the high-frequency components [10]. In JPEG compression, the given images are converted into 8×8 blocks and 2D DCT is applied to each block; then, the DCT coefficients are quantized based on a predefined

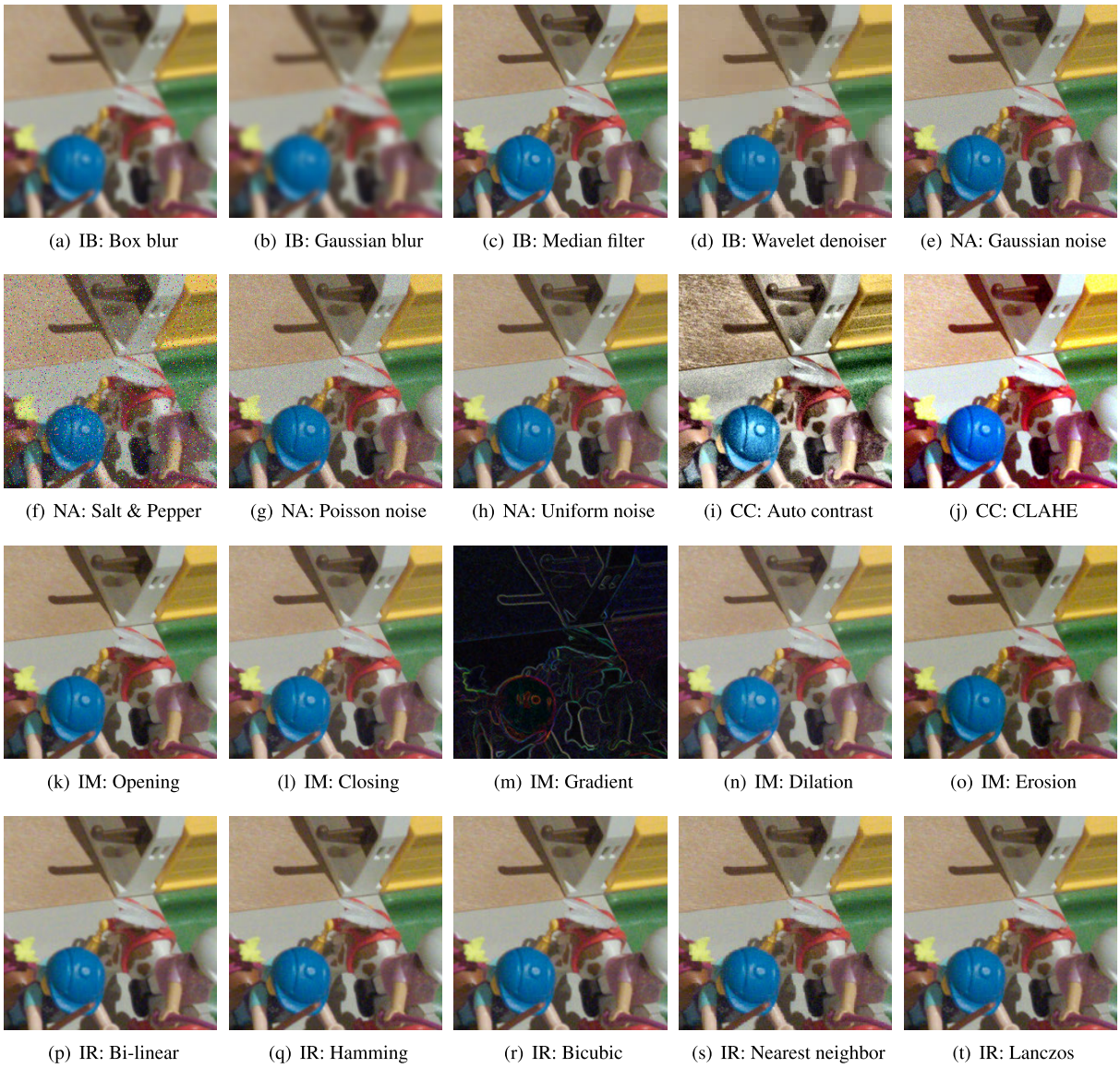


FIGURE 2. **Examples of image manipulations represented as 'manipulation type: manipulation algorithm'. The terms IB, NA, CC, IM, and IR represent image blurring, noise addition, contrast change, image morphing, and image resampling, respectively.**

quantization table, which corresponds to the compression factor. Considering realistic scenarios where JPEG compression is applied to the manipulated image, we compressed each manipulated image to an arbitrary quality factor between 75 and 95.

III. PROPOSED FORENSIC SYSTEM

A. OVERVIEW

We propose the manipulation classification network (MCNet) to classify various types of manipulations on JPEG images. To classify the manipulation traces remaining after compression, we used features from multiple domains, namely, spatial, frequency, and compression domains. Using the spatial and frequency domains, we designed a visual artifact network (VANet) to detect visual artifacts caused by manipulation traces. Further, for the compression domain, we designed a compression artifact network (CANet) to capture compression artifacts from lossy compression via JPEG. The main motivation of using multiple domains is to provide abundant and distinctive features to the network for the purpose of classification.

Fig. 3 presents an overview of the CNN-based forensic system comprising data preprocessing and three networks that are trained in two phases. Training of the VANet and CANet is performed individually in phase 1. VANet consists of spatial and transform domain learners, and CANet includes the compression domain learner. Both networks are trained to classify the manipulation algorithm that has been applied to an image using its own-domain features. In training phase 2, the ensemble model is employed to learn the features jointly

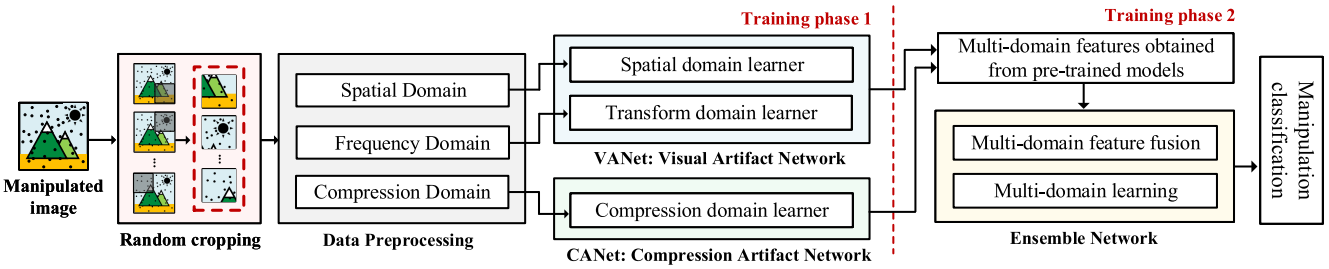


FIGURE 3. Overview of the proposed MCNet for manipulation classification. Data preprocessing converts the input image into multiple domain data. Three networks in the MCNet, including VANet, CANet, and ensemble network, are trained in two phases.

acquired from the two trained networks; hence, the proposed model is fine-tuned by transfer learning. The intermediate features of the VANet and CANet are concatenated into one set for learning and exploring the multi-domain features. Since visual and compression artifacts are considered simultaneously by utilizing the multi-domain features, the proposed MCNet successfully extracts and learns forensic traces caused by realistic manipulation scenarios.

B. DATA PREPROCESSING

To provide suitable data to each network stream, we convert the input data (i.e., manipulated image saved as JPEG format) into three types of preprocessed data for multiple domains. For each JPEG image, we first read the JPEG header to obtain the quantized DCT coefficients and randomly crop this into regions of size $W \times H$, where $W = H = 128$, with aligning to the 8×8 blocks of the DCT-based quantization of JPEG compression. For VANet, we decode these obtained coefficients to RGB format and apply preprocessing for the spatial and frequency domain learners. For CANet, the quantized coefficients are directly provided to the compression domain learner without decoding. Specifically, the spatial domain learner uses the RGB pixels directly, and the frequency domain learner converts these RGB pixels into 2D block DCT features. For the compression domain leaner, we generate binarized coefficients using the quantized DCT coefficients of the Y channel.

1) DATA PREPROCESSING FOR SPATIAL AND FREQUENCY DOMAINS

As mentioned above, the spatial and frequency domain learners utilize preprocessed data based on the RGB pixels decoded from the quantized DCT coefficients. First, we provide the RGB pixel data directly to the spatial domain learner as input. From this, we induced the spatial domain learners to learn and explore the rich visual features of the various fine manipulations contained in RGB the pixel data. Next, for the frequency domain learner, we apply an $N \times N$ 2D block DCT to the RGB pixels. Inspired by approaches [10], [11] that employed the 2D block DCT to explore the local frequency components, we use the $N \times N$ 2D block DCT to extract traces of manipulation in the frequency domain. Furthermore, to explore the relationship between each DCT basis in the training phase, the coefficients derived from different DCT bases are separated into their own channels for specific frequency components. This can be directly derived from the convolution operation between the image and the $N \times N$ DCT basis. Convolution using this basis with stride M increases the channel size by N^2 times and reduces the image size as $\frac{1}{M} \times \frac{1}{M}$. From the experiments in Section IV, we set the DCT basis scale N as 4 to ensure that the transformed frequency features do not lose the details of the original one and to generate only a $16\times$ increased channel than the original image. Moreover, the stride size M is set as 2 rather than 4 to learn the relations between the adjacent sub-blocks by computing the DCT coefficients of the overlapped sub-blocks.

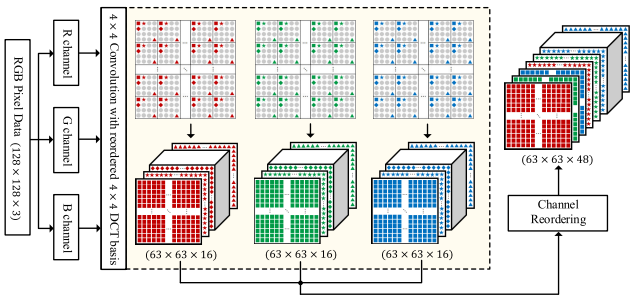


FIGURE 4. Preprocessing of 2D block DCT feature for frequency domain learner. This method employs a 4 × 4 2D DCT basis, and the output is generated using a kernel with zig-zag scanning and RGB channel reordering.

Fig. 4 illustrates the process of 2D block DCT feature generation. First, we calculate a sixteen 4×4 DCT basis bank $\mathbf{B} = \{B_{0,0}, B_{0,1}, B_{1,0}, \ldots, B_{3,3}\}$ which follows the zigzag scanning order. This basis bank is transformed into a convolution filter $\mathbf{F}$ of size $4 \times 4 \times 1 \times 16$ where the four axes refer to kernel width, kernel height, input channel, and output channel. Each R, G, and B channel is then convolved separately with $\mathbf{F}$ to generate three feature maps. The i-th channel of each convolution output is a collection of the i-th 2D DCT coefficient calculated from each 4×4 sub-block that is extracted with stride 2. As shown in Fig. 4, the three feature maps are concatenated into a single feature map $(63 \times 63 \times 48)$

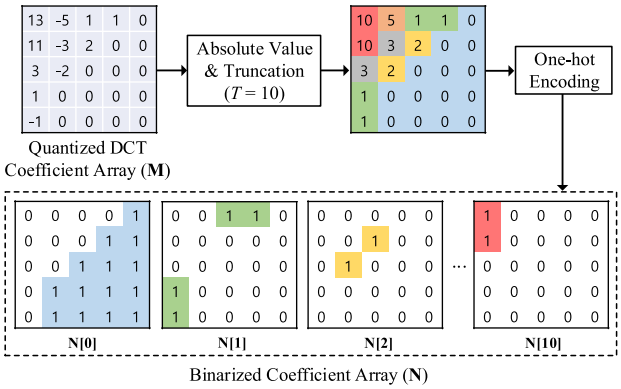


FIGURE 5. Preprocessing of the generated binarized coefficient array for the compression domain learner. The quantized DCT coefficients are processed using truncation and one-hot encoding.

by applying channel reordering which places the i-th channels of the R, G, and B feature maps into $3i$, $3i+1$, and $3i+2$-th channels, respectively. The reason for exploiting the zigzag scanning and channel reordering approach here is to enable the adjacent channels of the final feature map to have higher correlations in the frequency domain. This 2D block DCT feature is provided to the frequency domain learner of the VANet.

2) DATA PREPROCESSING FOR COMPRESSION DOMAIN

Statistical information such as the DCT histogram is known to be useful for detecting artifacts from double JPEG compression [10], [11]; hence, we decided to generate features that could extract statistical information of the manipulation traces from compressed data. Inspired by JPEG steganalysis [38], which allows computing specific histograms such as occurrence and co-occurrence of JPEG coefficients using convolutional layers, we generated the compression domain features as a binarized coefficient array. To do this, we read the JPEG header of an image to extract the quantized JPEG coefficients, and only considered the Y channel because the resolutions of the Cb and Cr channels depend on the sampling mode. Detailed feature generation proceeds in the following order (see Fig. 5). The quantized DCT coefficient array of the Y channel, represented as $\mathbf{M} \in \mathbb{Z}^{1\times 128\times 128}$, is truncated with threshold T with respect to its absolute value, and transformed into the binarized coefficient array $\mathbf{N} \in \{0,1\}^{(T+1)\times 128\times 128}$ using one-hot encoding as:

$$\begin{aligned} \mathbf{M} &= |\mathbf{M}|, \\ \mathbf{M}[\mathbf{M} \geq T] &= T, \\ \mathbf{N}[i] &= [\mathbf{M} == i], \quad i \in \{0,1,\ldots,T\}, \end{aligned} \tag{1}$$

where [.] denotes the (element-wise) Iverson bracket. This binarized coefficient array $\mathbf{N}$ allows our CNN-based forensic system to learn the statistical properties of the manipulated images from the perspective of the compression domain. For example, the convolution operation on $\mathbf{N}$ using a specific filter $K \in \mathbb{R}^{(T+1)\times 3\times 3}$, where $K = 0$ except $K[x,1,1] = 1$, $K[y,1,2] = 1$, and global average pooling to compute the horizontal co-occurrence for coefficient pairs $(x,y) \in \{0,1,\ldots T\}^2$. Further, using the A'Trous, or dilated, convolution with a kernel dilation rate of 8 is used to compute the inter-block DCT coefficient relationships. This binarized coefficient array is provided to the compression domain learner of the CANet.

C. NETWORK ARCHITECTURE

To classify the manipulation traces remaining after JPEG compression, we propose the CNN-based forensic system, referred to as MCNet, for learning and exploring the multi-domain features. Fig. 6 illustrates the overall structure of the proposed MCNet comprising two sub-networks (i.e., VANet and CANet) and an ensemble network. As previously noted, the VANet uses preprocessed data suitable for capturing the visual artifacts of manipulation (i.e., RGB pixel data and 2D block DCT features), and the CANet, which is a compression artifact learner, is provided the binarized DCT coefficients as inputs. The MCNet comprises both sub-networks and their ensemble network, and is trained in two phases. First, each sub-network proactively learns the forensic features in the multiple domains from manipulation and JPEG compression. Then, the ensemble network learns the features jointly acquired from the two pretrained networks, and the proposed MCNet is thus fine-tuned by transfer learning.

To learn the rich forensic traces from the multiple domains, we adopt network blocks suitable for extracting low-level features, as used in [8], [28]. As described in the Fig. 6, we use four blocks types, including BT1, BT2, BT3, and BT4, to design the sub-networks. BT1 is the inner-most basic block for the overall architecture, which consists of a 3×3 convolution, batch normalization, and rectified linear unit (ReLU) as the non-linear activation function. BT2 is a residual block without the pooling layer and consists of two convolutions with skip-connection. For each network, we place several BT2 modules right after BT1 to extract the low-level features. BT1 and BT2 facilitate training about the low-level signals by not using a pooling layers that can remove noise-like signals inserted by manipulation algorithms. This concept is commonly applied to many forensic and steganalysis tasks [9], [28], [29]. Next, BT3 is also a residual block containing 3×3 and 1×1 convolution operations with stride 2 to halve the input feature resolution. BT4 is the last module that is used to fuse the features from different domain learners and vectorize the feature using global average pooling. Using these block types, we design the VANet, CANet, and ensemble network to extract the manipulation traces effectively using different types of domain data. Additionally, the first operation of the frequency and compression learners are modified for better extraction of features from each domain.

1) VANET: VISUAL ARTIFACT LEARNER

The VANet is a two-stream network consisting of the spatial and frequency domain learners. The spatial domain learner consists of two BT1, four BT2, and four BT3 components,

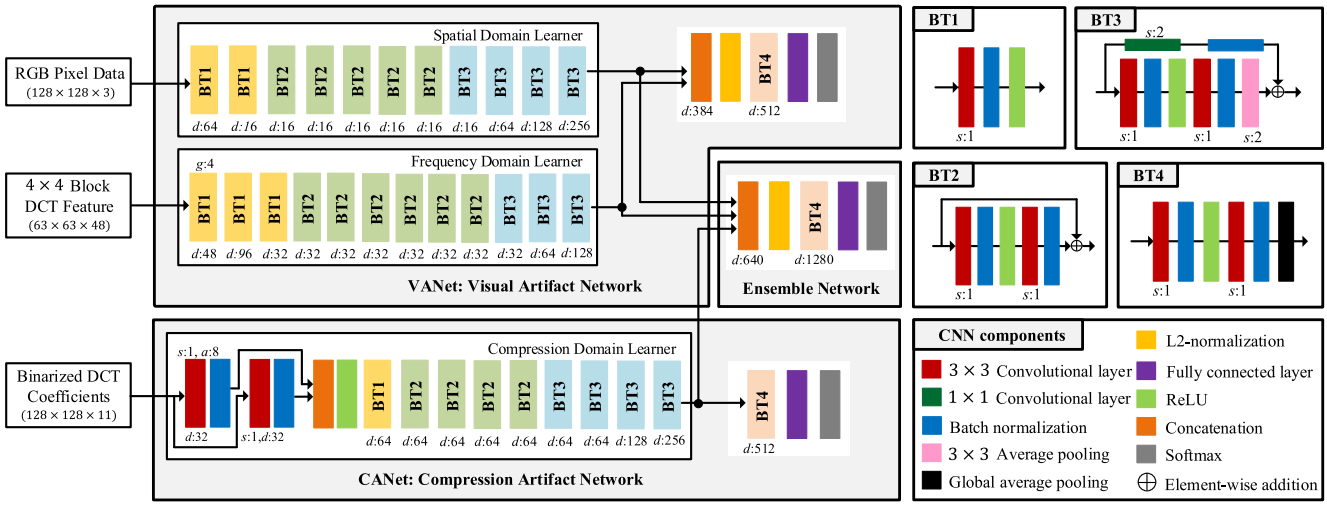
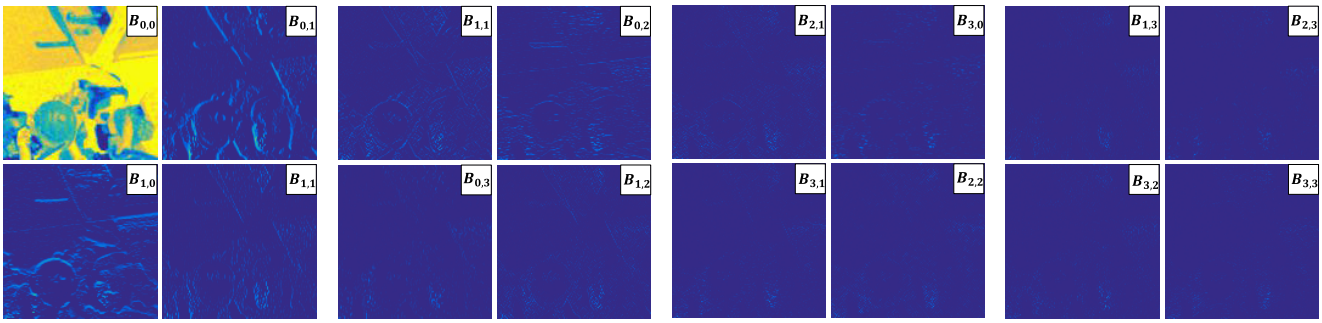


FIGURE 6. Network architecture of the MCNet consisting of the visual artifact network (VANet), compression artifact network (CANet), and ensemble network. The VANet takes the spatial domain and transform domain features, and the CANet takes the compression domain feature. The MCNet classifies the manipulations applied to an input image using the multi-domain feature. The variables *d*, *s*, *a*, and *g* represents the number of channels in each layer's output, stride size, dilation rate of convolution, and number of groups in group convolution, respectively.



(a) Group-1 (b) Group-2 (c) Group-3 (d) Group-4

FIGURE 7. 2D block DCT feature for the frequency domain learner. (a) to (d) depict the DCT outputs of the R channel divided into four groups using a zig-zag scanning order.

and it is trained using RGB pixels of size 128 × 128 × 3 directly. Since subtle manipulation traces are vulnerable to the pooling layer, we placed several BT1 and BT2 components without the pooling layer in front of the spatial domain learner. Owing to this, the spatial domain learner focuses on manipulation traces rather than the contents of the image. Subsequently, dimensionality reduction of the feature maps is performed via BT3, where higher-level features are learned from the lower-level features obtained from consecutively placed BT2 layers.

The frequency domain learner consists of three BT1, six BT2, and three BT3 blocks. The learner for the frequency domain considers 4×4 block DCT feature of size 63×63×48 as the input. Since the channels of the feature are aligned using zigzag scanning and RGB reordering, the energy contained in each channel downgrades as its index increases. We divide this DCT feature into four sub-groups channel-wise with sizes 63 × 63 × 12 each. Fig. 7 shows the 2D DCT feature of the R channel in each group where each image represents the convolution result between the R channel and DCT basis that is scanned in zig-zag order. Each group consists of features from high- to low-frequency components of the image. To utilize this property, we apply group convolution for the first convolution of the frequency domain learner. Group convolution computes outputs using only the channels in each group and merges them into one, and the number of parameters is inversely proportional to the number of groups. This helps prevent the immediate mixing of high- and low-frequency features and allows training using channels with high correlations among them. After the first BT1 in the group convolution, the remaining convolutional layers use regular convolution to learn the features using the entire channels.

The number of BT3 blocks for spatial and frequency domain learners are set to match the resolutions of the output feature maps. After preprocessed data is passed through each domain learner, the output features have sizes

$8 \times 8 \times 256$ and $8 \times 8 \times 128$, respectively (see Fig. 7). These two features are concatenated into a single integrated feature, and L2-normalization is applied to match the distributions between the two features from different domains. We use BT4 to obtain the mixed features from both learners and to vectorize the feature maps using the global average pooling layer. Finally, the probability for each manipulation class is calculated using a single fully-connected layer. The VANet is then trained to minimize cross-entropy loss between the outputs and labels.

2) CANET: COMPRESSION ARTIFACT LEARNER

The CANet is a single-stream network comprising the compression domain learner, which consists of a statistical feature extractor, single BT1, four BT2, and four BT3; it is trained using binarized DCT coefficients of size $128 \times 128 \times 11$. We use the 'a trous', or dilated, convolution and regular convolution with a kernel size of 3×3 to extract the statistical feature such as co-occurrence and relationships between the DCT coefficient sub-blocks simultaneously. We apply two convolution operations separately to the input and concatenate the feature maps obtained from them into a single feature of 64 channels. This structure is placed in the initial segment of the compression domain learner as the initial feature extractor. Instead of fixed statistical information such as the histogram, this scheme learns the appropriate statistical features. The number of BT3 blocks placed in the compression domain learner is also considered to match the resolutions of the spatial and frequency domain learners. Similar to the VANet, we place the BT4 and fully connected layer at the end of the CANet. The CANet is then trained to minimize the cross-entropy loss between the outputs and labels.

3) ENSEMBLE NETWORK

The ensemble network is designed to learn the features jointly acquired from the three types of domain learners trained using different domain features; hence, the proposed MCNet is fine-tuned by transfer learning. After training each domain learner, we collect the output features of the spatial, frequency, and compression domain learners, which have resolutions of 8×8 each, with 256, 128, and 256 channels, respectively. Rather than ensemble training using the output vectors of the BT4 blocks of the two sub-networks, the outputs of the spatial, frequency, and compression domain learners are fused again using the BT4 of the ensemble network. This correlates the location of activation inside each domain feature. Similar to the VANet, the three feature maps are concatenated into a single feature map, and the probability of each manipulation class is generated and trained to minimize the cross-entropy loss between the outputs and labels.

IV. EXPERIMENTAL RESULTS

To validate the effectiveness of the MCNet, extensive experiments based on baselines, including MISLNet [5], MantraNet [3], Xception [25], and SRNet [28], were conducted. Compared to the baselines, our work outperforms in terms of multi-class manipulation classification. In addition, we experimentally proved that the fine-tuned model based on the multi-class manipulation task was effective for different forensic tasks such as DeepFake detection and integrity authentication of a JPEG image.

A. DATABASE GENERATION

To train a network that classifies a manipulated image with JPEG compression, we generate a dataset using uncompressed images. We used the ALASKA steganalysis challenge dataset [39], which consists of 80,005 uncompressed color images (256×256) captured with 51 different cameras. We randomly divided these images into training, validation, and test sets, with (77,005), (1,000), and (2,000) uncompressed images, respectively. As mentioned in Section II-B, to generate the manipulated dataset, we applied five different manipulation algorithms from different manipulation types to a single image. With consideration of realistic manipulation scenarios, the manipulated images were compressed using an arbitrary JPEG quality factor between 75 and 95. As a result, (385,025), (5,000) and (10,000) manipulated images with JPEG compression were generated for training, validation, and testing sets, respectively. Furthermore, we divide the entire training data into five groups, so that the contents of the manipulated images in each group were different. For each epoch of the training network, we exploited one of the five subdivided groups consisting of 77,005 images each.

B. TRAINING DETAILS

The proposed VANet, CANet, and MCNet, as well as the baseline networks are trained up to 200 epochs using the AdamW [40] optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$ to minimize the cross-entropy loss. The learning rate and weight decay were initialized to 5×10^{-4} and 10^{-5}, respectively. The batch size was set as 200 for every network, we used a learning rate scheduler policy called 'Reduce on Loss Plateau Decay'. This multiplies the decay rate $p = 0.5$ with the learning rate and weight decay if the validation accuracy fails to improve for 10 epochs. The overall experiments have been implemented on the PyTorch [41] framework on NVIDIA TITAN RTX. We applied data augmentation to the training set by randomly cropping 128×128 areas of an image for training each network. Since the data preprocessing of the VANet uses decompressed RGB pixels, we applied additional augmentation, including rotation for multiples of 90° and random flips. This is not applied to the CANet, which uses quantized DCT coefficient from the JPEG header directly, since the DCT coefficients have the property of decreasing values from the upper left to the lower right.

C. ANALYSIS OF FREQUENCY DOMAIN LEARNER

In this section, we present evaluations of the effectiveness of the frequency domain learner architecture to construct the VANet. To select the suitable hyper-parameters empirically, we conducted experiments with respect to different settings of the hyper-parameters, including the DCT scale, stride size for

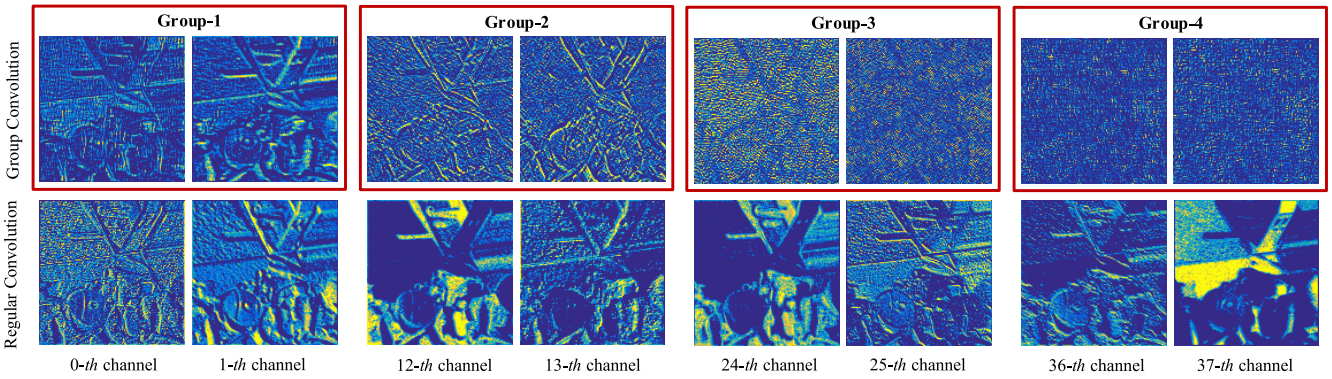


FIGURE 8. Analysis of the effect of group convolution on the block DCT feature. Each column refers to a channel index among the 48 channels. The top row represents the results of group convolution, and the bottom row represents the results of regular convolution.

feature generation, and usage of group convolution in the first BT1 of the frequency domain learner. To test the frequency domain learner alone, we appended a BT4 and a fully connected layer. We trained this variant of the frequency domain learner for the manipulation classification task. As mentioned in Section III-B, we exploited the $N \times N$ block DCT to extract trace of manipulation by exploring the relations between the local frequency components of the frequency domain learner, as in [10], [11]. We considered the case where the value of the DCT scale N was 4 or 8, since DCT scales larger than 8 failed to improve the performance gain of the frequency domain learner. This is because the high-frequency coefficients are extremely low values compared to the DCT scale of 4 or 8.

When the values of the DCT scale are 4 and 8, using stride sizes less than the DCT scale provide outstanding performance gains (see Table 2). This implies that extracting the frequency components from the overlapping image sub-blocks is more effective because the size of the 2D DCT feature shrinks proportional to the square of the stride size. It is important to not reduce the feature dimension in the early layers so that noise features remained in pixels can be extracted. From this, we concluded that employing small stride sizes show significant performance improvements.

Meanwhile, even though using group convolution with a group size of 4 reduces the number of parameters to a fourth of that used in regular convolution, there is still a marginal performance gain. This is derived from the first BT1 with group convolution, which trains using channels with higher correlations derived from zigzag scanning and RGB channel reordering. Except for this first BT1, the remaining convolutional layers use regular convolutions on the entire channels. Fig. 8 shows the filter activation of the first BT1 of the frequency domain learner with and without group convolution. The group convolution with energy-based ordered DCT features produces results containing less semantic-level details of the original image and more distinctive details between the channels (see the top part of Fig. 8), while the output of regular convolution shows no such tendency with respect to the channel index. As listed in Table 2, the frequency domain learner achieves 18.69% top-1 error with the following settings: DCT scale 4, stride 2, and group convolution. With these results, we use the frequency domain learner with the above parameter settings to train VANet and MCNet.

TABLE 2. Top-1 classification errors of the frequency domain learner with different hyper-parameters.

DCT Basis Scale	Stride Size	Group Convolution	Error (%)
8	8	×	33.67
8	4	×	22.86
8	4	○	22.59
8	2	○	20.08
4	4	×	28.38
4	2	×	19.21
4	2	○	**18.69**

D. ABLATION STUDY

We conducted an ablation study to verify the effectiveness of the proposed spatial, frequency, and compression domain learners. To do this, we trained the networks comprising pairs of the three domain learners and the MCNet from scratch. To train a network employing two different domain learners, we follow the same architecture of VANet that merges outputs from the spatial and frequency domain learners. In addition, we trained MCNet without pretraining the VANet and CANet to check effectiveness of two-phase (i.e., phase-1 for training each domain learner and phase-2 for fine-tuning in a fashion of transfer learning) training process (see Section III-C3). Table 3 exhibits the results of ablation study on each combination of domain learner.

The results show that spatial domain learner achieved highest performance among the single-domain learners, but the VANet that employed spatial and frequency domain simultaneously achieved higher performance than training each domain separately. However, training spatial and compression domain learner, or frequency and compression domain learner, from scratch shows poorer performance than

TABLE 3. Results of ablation study on the spatial, frequency and compression domain learners and training from scratch.

Spatial	Frequency	Compression	From scratch	Error (%)
✓			○	17.9
	✓		○	18.69
		✓	○	36.29
✓		✓	○	19.32
	✓	✓	○	21.86
✓	✓		○	15.97
✓	✓	✓	○	16.74
✓	✓	✓	×	**15.21**

a single-domain learner. This phenomenon can also be found from the comparison of the VANet (15.97%) and MCNet trained without pretrained networks (16.74%). We conclude that simply attaching the untrained compression domain learner and training them from scratch does not yield the performance improvements, and even brings degradation. Meanwhile, the proposed MCNet (15.21%) which first the train the compression domain learner (CANet) and remaining learners (VANet) separately, and merges into a single network shows performance gain compared to the VANet. This represents that the proposed two-phase training process is effective in bringing synergy to find manipulation artifacts from each domain.

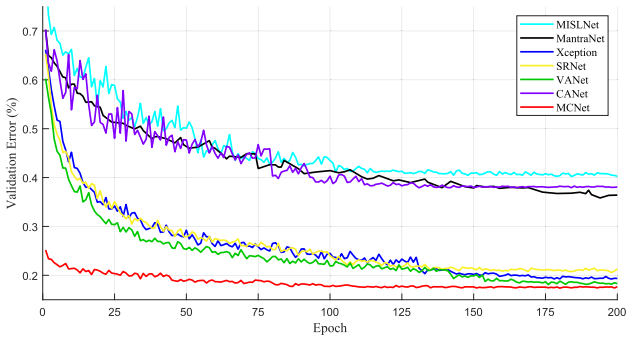


FIGURE 9. Tendencies of the top-1 error of the proposed MCNet and baselines on the validation set.

E. PERFORMANCE EVALUATION OF NETWORKS

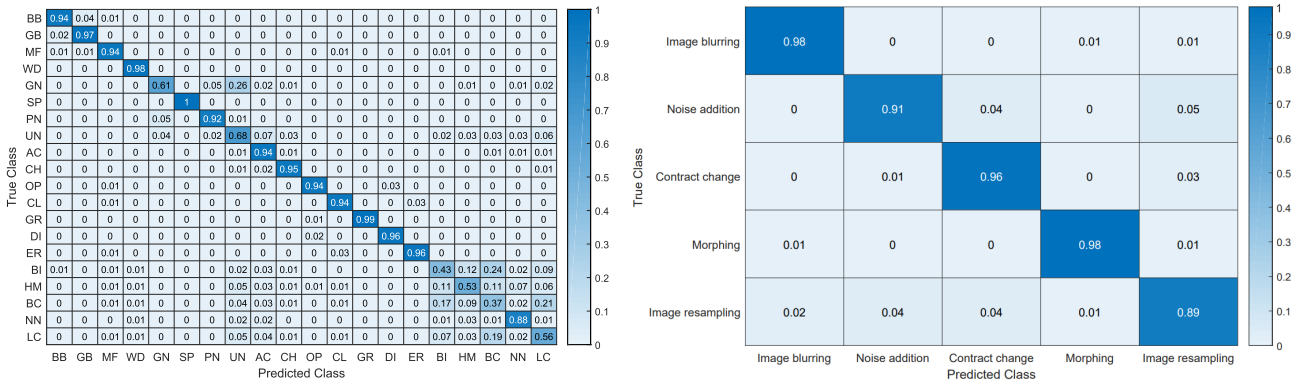
To evaluate the performance of the proposed MCNet for manipulation classification, we compare our method to other baseline manipulation classification networks (MISLNet [5], MantraNet [3]) and network architectures that are known to be effective for low-level vision tasks such as steganalysis (SRNet [28]) and DeepFake detection (Xception [34], [42]). Fig. 9 shows the tendencies of the validation top-1 error for each network. As can be seen in Fig. 9, every model converges well after 200 epochs of training. The MISLNet shows the worst performance among all the models, even though it is known to be effective for identifying manipulations on uncompressed images. MantraNet employing the SRM kernel [32] and constrained layer of MISLNet achieves a slightly lower error rate than the CANet, which uses only compression domain data, and shows slower convergence than the compared models. The results of these two networks show that features extracted from only the spatial domain using a fixed kernel and constrained layer are less effective than features extracted from multiple domains using layers designed to suit each domain. Compared with MantraNet and MISLNet, Xception, SRNet, VANet and MCNet showed faster convergences and lower errors. In this experiment, the proposed MCNet achieved the best performance since its ensembles had pretrained VANet and CANet.

TABLE 4. Top-1 and top-3 classification errors of the MISLNet, MantraNet, Xception, SRNet, two sub-models of this study (VANet, CANet), and MCNet.

Model	Classification Error (%)		Diff (%)
	Top-1 Error	Top-3 Error	
MISLNet [5]	37.17	19.20	17.97
MantraNet [3]	35.56	14.21	21.35
Xception [25]	17.21	5.01	12.20
SRNet [28]	17.9	5.04	12.86
VANet	15.97	4.45	11.52
CANet	36.29	16.17	20.12
MCNet	**15.21**	**4.13**	**11.08**

Along with analyses of the validation error tendencies, we evaluated the manipulation classification performances of the networks on the test set. Table 4 shows the top-1 and top-3 classification errors of the proposed models and baseline networks. The networks employed for other low-level vision tasks (i.e., SRNet and Xception) achieves better performances than MantraNet. Xception shows a slightly better performance than SRNet, which has the same architecture as the spatial domain learner of the VANet, but shows 1.24% higher top-1 error than the proposed VANet. The proposed MCNet employing multi-domain features, achieves the best top-1 error of 15.21%, which is an improvement of 0.76% over the VANet. From the differences between the top-1 and top-3 errors, the proposed MCNet shows the lowest top-1 (15.21%) and top-3 (4.13%) errors, which implies that it has higher discriminability among the algorithms for the same type of the manipulation than the existing methods.

To evaluate the performance of our MCNet in detail, we generated two confusion matrices to classify the manipulation algorithms (20 classes) and manipulation type (5 classes), as shown in Fig. 10(a) and 10(b), respectively. The results show that the MCNet successfully predicts the manipulation algorithm applied to the JPEG images, especially those corresponding to image blurring, morphing, and contrast change (see Fig. 10(a), whereas low prediction accuracies are shown for noise addition (GN, UN) and resampling (BI, HM, BC, NN, and LC) algorithms. This is because these algorithms leave less visually distinguishable traces on the original images compared to others (see Fig. 2), and their traces are further weakened by JPEG compression afterwards. However, these are mutually mis-classified among



(a) Manipulation Algorithms (20 classes)

(b) Manipulation Types (5 classes)

FIGURE 10. Confusion matrix of the proposed MCNet for different manipulation algorithms and types.

algorithms with in the same type of manipulation. As shown in Fig. 10(b), even if the manipulation algorithm is mis-predicted, it is able to predict the manipulation type. For the manipulation types, the probability values for the correct prediction of the true classes for image blurring, noise addition, contrast change, morphing, and image resampling were 0.98, 0.91, 0.96, 0.98, and 0.89, respectively.

F. VISUALIZATION OF CLASS ACTIVATION MAP

In this section, we generate class activation map (CAM) to observe whether the spatial, frequency, and compression domain learners are trained using the artifacts from the designated domain. Since the manipulation is applied to the entire area of each image, the activated area in the CAM of the last feature map that is used for localization of object is not suitable for the purpose. Also, the output of three domain learners are mixed into single feature at the last BT4 layer; hence, the original CAM [43] is inappropriate since it is only feasible at the latest feature map. Indeed, we use Grad-CAM++ [44] to generate an activation map that allows us to generate CAM at the intermediate layers of each domain learners.

We generate a CAM of the last BT1 layer of each domain learner with respect to its manipulation algorithm class to check how different activation appears at each domain learner for low-level artifacts of different manipulation algorithms (see Fig. 11). For the manipulation types that leave visible artifacts (e.g. noise addition, image morphing), the spatial domain learner is activated where the manipulation is visually distinguishable, and the degree of activation is greater than frequency domain learner, since 75% of channels of the 2D block DCT feature represents the mid- and high-frequencies DCT coefficients and it is forced to keep this property via group convolution. For the image blurring and image resampling that leave subtle visual artifacts that affect mid- and high-frequency of DCT coefficients, the CAM of the two domain learners show the opposite of the CAM of noise addition and image morphing. For the contrast change, the two domain learners show similar activation since it is visually noticeable and amplifies the high-frequency components of

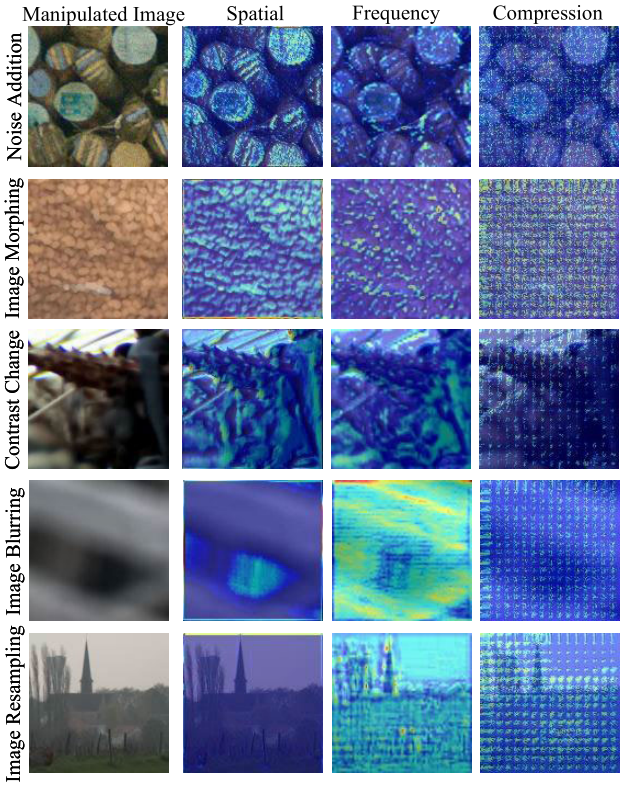


FIGURE 11. Class activation map visualization of last BT1 layers from spatial, frequency, and compression domain learners of the MCNet.

the image. Meanwhile, compression domain learner shows 8×8 grid activation regardless of the manipulation type. From this analysis, we conclude that each domain learner focuses on different artifacts from different domains, even if they are trained using the same image and same objective function.

G. PERFORMANCE EVALUATION OF NETWORKS FOR UNSEEN CASES

To demonstrate the effectiveness of the proposed MCNet, we further experimented on the robustness of the MCNet

TABLE 5. Performance Evaluation of Networks for Unseen Cases.

Category		Error (%)
Camera Devices Dataset	FiveK [45]	18.39
Non-standard JPEG Compression	PSWeb90	14.42
	PSWeb70	15.17
	PS10	15.1
	PS08	18.52
Image Size	96×96	18.98
	64×64	27.14
	32×32	52.95

against unseen cases. We conducted experiment on three types of unseen cases including unseen camera devices, non-standard JPEG compression, and different input sizes. Table 5 lists the results of the proposed MCNet for unseen cases.

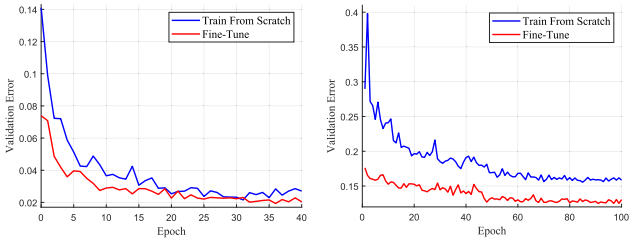
First, to verify the robustness to cameras devices that are not included in the ALASKA dataset [39], we use images from the FiveK [45] dataset which consists of 5,000 images captured with various DSLR cameras and 4,600 images that were not captured with devices included in the training dataset were selected to generate the test data. These images were manipulated randomly, and compressed with quality factor in range of 75 to 95. Since the devices in the ALASKA dataset consist of smartphones, tablets, low-end cameras, and high-end DLSRs whereas FiveK is captured only using DLSR, the trained MCNet has the robustness to the camera devices.

Next, to verify the robustness to non-standard JPEG compression, we re-generated test images using four different JPEG quantization tables from Adobe Photoshop with different qualities. As with the existing test dataset (30,000 images), we manipulated the same algorithms and parameters for each uncompressed images and encoded it into JPEG using each quantization table. As shown in Table 5, our work achieved an error of 18.52 or less for each of the four unseen quantization tables. Since the network is trained using JPEG images compressed with various quality factors, this result shows that the proposed MCNet is robust to unseen JPEG encoder.

In addition, we tested the MCNet using different input image size from 96×96 to 32×32 compared to the trained image size, 128×128. The results show that the trained model is robust until the input size is 96×96, but it drastically degrades as the resolution is under 25% of the trained image block size. This is because a larger image contains more manipulation artifacts than a smaller image. Summarizing the results in this section, the proposed MCNet achieved stable and high performance for three types of unseen cases.

H. FINE-TUNING FOR FORENSIC TASKS

We assume that the proposed models, which are trained on rich forensic features of realistic manipulation environments, can help obtain general discriminability for various forensics tasks. To verify this statement, we conducted fine-tuning of



(a) DeepFake Detection (b) Integrity Authentication

FIGURE 12. Tendencies of validation error of forensic tasks. Fine-tuning from the proposed models trained for manipulation classification shows faster convergence and better performance for both cases.

TABLE 6. Performance comparison between fine tuning and training from scratch.

Task	Detection Error (%)		Diff (%)
	From Scratch	Fine-tuning	
DeepFake Detection	3.14	2.75	0.39
Integrity Authentication	13.89	10.71	3.18

the pretrained model with our manipulation dataset for the target tasks (i.e., DeepFake detection and image integrity authentication). We also performed performance analyses between the models trained from scratch and the fine-tuned models for each task. For each task, we selected an appropriate model from among the VANet and MCNet based on whether the images contained JPEG compression or not. To train the network for each task, we changed the output of the last fully connected layer to a single probability for binary classification. As shown in Fig. 12 and Table 6, we observe that fine-tuning using the pretrained models on our manipulation dataset shows faster convergence and improved the detection accuracies for both tasks.

1) DEEPFAKE DETECTION

DeepFake is a technique that transforms the original face in an image into that of another person or changes the facial expression such as lip movements. This has become a controversial issue as the amounts of fake news have increased. To detect such manipulation, we used the DeepFake Detection Challenge (DFDC) dataset [46] for the experiment. The entire DFDC dataset consists of 104,500 fake and 23,654 original videos. We extracted 860,000 cropped face images in the uncompressed (PNG) type from the original videos and randomly selected frames from the fake videos to generate an equal number of fake face images. In total, we generated 1,720,000 cropped face images, and the obtained images were divided into three sets for training, validation, and testing (8:1:1).

Since the data format of the DFDC dataset is already compressed using a video codec (i.e., H.264), we saved the cropped faces using an uncompressed format rather than applying JPEG compression so as to lose the traces of forgery from compression. We conducted experiments using

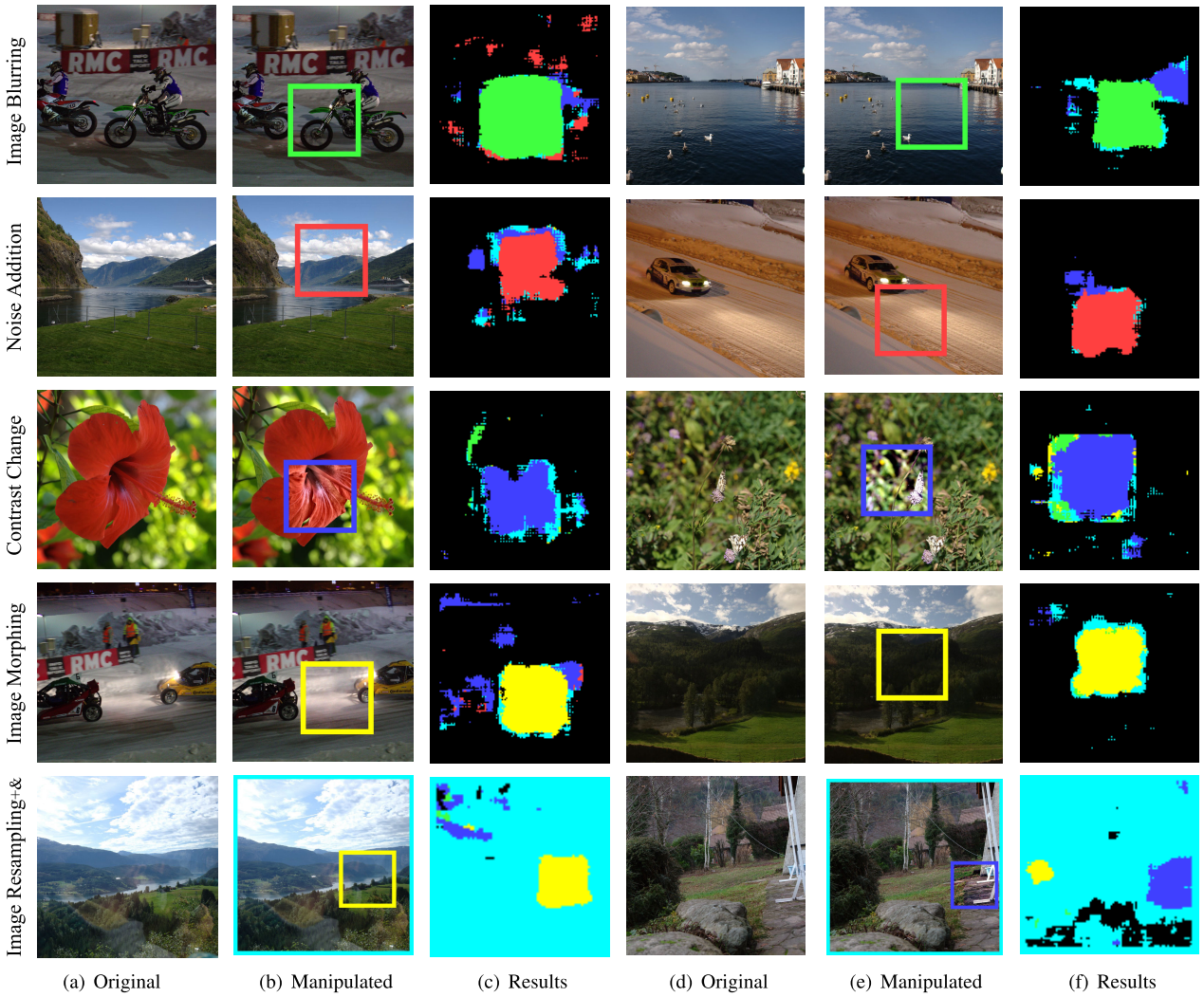


FIGURE 13. Results of localizing the manipulated regions based on our MCNet: (a) and (d) are original images, (b) and (e) are forged images with manipulation applied to the bounding box, and (c) and (f) are localization results based on integrity authentication and manipulation classification results. The bounding box refers to the location of the manipulation, and the color is its label. Black denotes no manipulation in the region, and each color stands for the manipulation type (green: image blurring, red: noise addition, purple: contrast change, yellow: image morphing, and cyan: image resampling).

the VANet that does not require the compression domain as an input. Since the cropped face images had various sizes, we reshaped each image to a resolution of 128 × 128. We trained the VANet to detect fake faces with and without fine-tuning based on the model trained for the manipulation classification task. The experimental results show that fine-tuning improves the performance of the network and achieves faster convergence with 2.75% errors, which is 0.39% lower than that of training from scratch (see Fig. 12(a) and Table 6).

2) INTEGRITY AUTHENTICATION OF JPEG IMAGES

Classifying the manipulation algorithm applied to an image and detecting the presence or absence of manipulations (i.e., 2-class: pristine or forged) are two different concepts. We expanded our work to detect the presence of manipulation in an image to authenticate the integrity of a JPEG image. In this experiment, we generated the same number of JPEG images without manipulation as that of the manipulated images Using a single uncompressed image in the existing dataset (80,005 images), we generated five JPEG compressed images using random quality factors in range of 75 to 95 (400,025 images) to match the number of original and manipulated images. As a result, 800,050 original and manipulated images were used, and the ratio of the training, validation, and test images is equal to that in the manipulation classification task.

Since the images were compressed using the JPEG scheme, we conducted experiments using the MCNet so that the compressed domain data could be utilized. We trained the MCNet for 100 epochs with and without fine-tuning. The experimental results show that the fine-tuned model converged much faster and achieved 10.71% detection error which is 3.18%

lower than that for training from scratch (see Fig. 12(b) and Table 6). From the results, we concluded that fine-tuning with our model and dataset is beneficial for forensics tasks that have a narrower scope.

Note that the fine-tuning process is possible because the pretrained MCNet is trained to classify manipulation algorithms (20 classes) in which the original images are not considered as a class (i.e. total 21 classes). The reason for training without the original images is because it is difficult to decide the proportion of the original images compare to the manipulated images in a single mini-batch. This is not a major concern when dealing with a few manipulation algorithms [5] that construct a balanced mini-batch (i.e. sampling ratio of the original image and each manipulation algorithm are equal). However, this balanced mini-batch induces a data imbalance because of the number of original images would be very low compared to manipulated images.

TABLE 7. Confusion matrix of fine-tuned and newly trained models for integrity authentication task.

Methods	Predicted Class	Actual Class	
		OR	MP
Fine-tuned from MCNet trained only with manipulations	OR	94.56	15.98
	MP	5.44	84.02
MCNet considering original image as a class	OR	48.57	3.34
	MP	51.43	96.66

Leaving the data imbalance aside for future work, we separate the manipulation classification task from the integrity authentication task. We compared the performance of the integrity authentication of the proposed fine-tuned model (two classes) and a newly trained MCNet considering the original image as a class (21 classes) using a balanced mini-batch. The latter network was trained using a manipulation dataset (80,005 images) where $\frac{1}{21}$ of the manipulated images were replaced by the original image with only JPEG compression applied. Table 7 is a confusion matrix of fine-tuned model and newly trained model for integrity authentication of two classes (i.e. OR: original class and MP: manipulation class). For the newly trained MCNet, considering original image as a class, the manipulation classes are treated as a single MP class. The result shows that the proposed fine-tuned model achieved much higher performance on the integrity authentication task that distinguishes the original image from various manipulated images. Furthermore, the performance of classifying manipulation degraded from 15.21% to 16.88% when the original images were considered as a class. From the results, we conclude that when dealing such large-numbered manipulation classification, separating integrity authentication from manipulation classification provides higher performance, and even can take advantage via fine-tuning.

I. LOCALIZATION OF MANIPULATION

To localize the manipulated region and identify the manipulation type, we use the integrity authentication and manipulation classification models simultaneously. In the experiments, we generated manipulated test images by applying the manipulation algorithm to a part of the image. Fig. 13 shows the manipulated images and their localization results. We first applied integrity authentication to the image sub-blocks sampled with an 8×8 grid and collected the blocks that were detected as anomalies. Then, these abnormal blocks were tested using the manipulation classification network. The results show that the two networks that are trained for manipulation classification and integrity authentication can successfully localize the manipulations and identify its type.

V. CONCLUSION

We propose the MCNet to classify the different manipulation algorithms applied to JPEG compressed images using multiple domain features. The proposed MCNet employs spatial, frequency, and compression domain data to maximize the performance of manipulation classification for JPEG compressed image. We designed appropriate preprocessing and network architectures for each domain learner. The experimental results show that the proposed MCNet performs best compared to other manipulation detection networks and several low-level vision networks. Because we consider a large number of different manipulation algorithms, the proposed network is effective to fine-tune for existing forensics tasks such as DeepFake detection and integrity authentication of JPEG images. In our future work, we intend to apply additional algorithms to reflect the real-world scenarios. Moreover, we intend to develop an end-to-end network that localizes the manipulation as well as its type simultaneously.

REFERENCES

[1] L. Verdoliva, "Media forensics and DeepFakes: An overview," 2020, *arXiv:2001.06564*. [Online]. Available: http://arxiv.org/abs/2001.06564
[2] B. Bayar and M. Stamm, "Design principles of convolutional neural networks for multimedia forensics," *Electron. Imag.*, vol. 2017, no. 7, pp. 77–86, Jan. 2017.
[3] Y. Wu, W. Abdalmageed, and P. Natarajan, "ManTra-Net: Manipulation tracing network for detection and localization of image forgeries with anomalous features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9543–9552.
[4] O. Mayer and M. C. Stamm, "Forensic similarity for digital images," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1331–1346, 2020.
[5] B. Bayar and M. C. Stamm, "Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2691–2706, Nov. 2018.
[6] M. C. Stamm, M. Wu, and K. J. R. Liu, "Information forensics: An overview of the first decade," *IEEE Access*, vol. 1, pp. 167–200, 2013.
[7] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 758–767, Feb. 2005.
[8] S.-H. Nam, W. Ahn, S.-M. Mun, J. Park, D. Kim, I.-J. Yu, and H.-K. Lee, "Content-aware image resizing detection using deep neural network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 106–110.
[9] S.-H. Nam, W. Ahn, I.-J. Yu, M.-J. Kwon, M. Son, and H.-K. Lee, "Deep convolutional neural network for identifying seam-carving forgery," 2020, *arXiv:2007.02393*. [Online]. Available: http://arxiv.org/abs/2007.02393
[10] J. Park, D. Cho, W. Ahn, and H.-K. Lee, "Double JPEG detection in mixed jpeg quality factors using deep convolutional neural network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 636–652.
[11] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro, "Aligned and non-aligned double JPEG detection using convolutional neural networks," *J. Vis. Commun. Image Represent.*, vol. 49, pp. 153–163, Nov. 2017.

[12] S.-J. Ryu, H.-Y. Lee, and H.-K. Lee, "Detecting trace of seam carving for forensic analysis," *IEICE Trans. Inf. Syst.*, vol. 97, no. 5, pp. 1304–1311, 2014.

[13] S.-J. Ryu, M. Kirchner, M.-J. Lee, and H.-K. Lee, "Rotation invariant localization of duplicated image regions based on Zernike moments," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 8, pp. 1355–1370, Aug. 2013.

[14] J.-U. Hou and H.-K. Lee, "Detection of hue modification using photo response nonuniformity," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 8, pp. 1826–1832, Aug. 2017.

[15] P. Yang, D. Baracchi, R. Ni, Y. Zhao, F. Argenti, and A. Piva, "A survey of deep learning-based source image forensics," *J. Imag.*, vol. 6, no. 3, p. 9, Mar. 2020.

[16] H.-Y. Choi, H.-U. Jang, J. Son, D. Kim, and H.-K. Lee, "Content recapture detection based on convolutional neural networks," in *Proc. Int. Conf. Inf. Sci. Appl.* Singapore: Springer, 2017, pp. 339–346.

[17] D. Cozzolino, G. Poggi, and L. Verdoliva, "Recasting residual-based local descriptors as convolutional neural networks: An application to image forgery detection," in *Proc. 5th ACM Workshop Inf. Hiding Multimedia Secur. (IHMMSec)*, 2017, pp. 159–164.

[18] S.-H. Nam, J. Park, D. Kim, I.-J. Yu, T.-Y. Kim, and H.-K. Lee, "Two-stream network for detecting double compression of H.264 videos," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 111–115.

[19] P. He, X. Jiang, T. Sun, S. Wang, B. Li, and Y. Dong, "Frame-wise detection of relocated I-frames in double compressed H.264 videos based on convolutional neural network," *J. Vis. Commun. Image Represent.*, vol. 48, pp. 149–158, Oct. 2017.

[20] 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.

[21] J. Zhang, Y. Liao, X. Zhu, H. Wang, and J. Ding, "A deep learning approach in the discrete cosine transform domain to median filtering forensics," *IEEE Signal Process. Lett.*, vol. 27, pp. 276–280, 2020.

[22] D. Cozzolino and L. Verdoliva, "Noiseprint: A CNN-based camera model fingerprint," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 144–159, 2020.

[23] D.-G. Kim, J.-U. Hou, and H.-K. Lee, "Learning deep features for source color laser printer identification based on cascaded learning," *Neurocomputing*, vol. 365, pp. 219–228, Nov. 2019.

[24] O. Mayer, B. Hosler, and M. C. Stamm, "Open set video camera model verification," in *Proc. ICASSP-IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 2962–2966.

[25] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2016, *arXiv:1610.02357*. [Online]. Available: http://arxiv.org/abs/1610.02357

[26] H. Mo, B. Chen, and W. Luo, "Fake faces identification via convolutional neural network," in *Proc. 6th ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2018, pp. 43–47.

[27] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A compact facial video forgery detection network," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2018, pp. 1–7.

[28] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 5, pp. 1181–1193, May 2019.

[29] W. Ahn, H. Jang, S.-H. Nam, I.-J. Yu, and H.-K. Lee, "Local-source enhanced residual network for steganalysis of digital images," *IEEE Access*, vol. 8, pp. 137789–137798, 2020.

[30] W. Ahn, S.-H. Nam, M. Son, H.-K. Lee, and S. Choi, "End-to-end double JPEG detection with a 3D convolutional network in the DCT domain," *Electron. Lett.*, vol. 56, no. 2, pp. 82–85, Jan. 2020.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[32] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, Jun. 2012.

[33] P. He, X. Jiang, T. Sun, and H. Li, "Computer graphics identification combining convolutional and recurrent neural networks," *IEEE Signal Process. Lett.*, vol. 25, no. 9, pp. 1369–1373, Sep. 2018.

[34] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1–11.

[35] Y. Zhan, Y. Chen, Q. Zhang, and X. Kang, "Image forensics based on transfer learning and convolutional neural network," in *Proc. 5th ACM Workshop Inf. Hiding Multimedia Secur. (IHMMSec)*, 2017, pp. 165–170.

[36] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[37] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Learning rich features for image manipulation detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1053–1061.

[38] Y. Yousfi and J. Fridrich, "An intriguing struggle of CNNs in JPEG steganalysis and the OneHot solution," *IEEE Signal Process. Lett.*, vol. 27, pp. 830–834, 2020.

[39] R. Cogranne, Q. Giboulot, and P. Bas, "The ALASKA steganalysis challenge: A first step towards steganalysis," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 125–137.

[40] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*. [Online]. Available: http://arxiv.org/abs/1711.05101

[41] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. D' Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

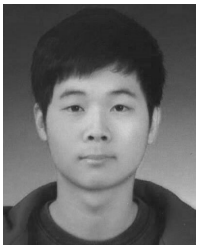
[42] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[43] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.

[44] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: Generalized Gradient-Based visual explanations for deep convolutional networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 839–847.

[45] V. Bychkovsky, S. Paris, E. Chan, and F. Durand, "Learning photographic global tonal adjustment with a database of input/output image pairs," in *Proc. CVPR*, Jun. 2011, pp. 97–104.

[46] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The deepfake detection challenge dataset," 2020, *arXiv:2006.07397*. [Online]. Available: https://arxiv.org/abs/2006.07397



IN-JAE YU received the B.S. degree from the Department of Mathematical Sciences, Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2015, and the M.S. degree from the School of Computing, KAIST, in 2017, where he is currently pursuing the Ph.D. degree with the Multimedia Computing Laboratory. His research interests include digital multimedia forensics and deep learning.

SEUNG-HUN NAM received the B.S. degree in information communication engineering from Dongguk University, Seoul, South Korea, in 2013, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2015 and 2020, respectively. He is currently a Postdoctoral Researcher with the Multimedia Computing Laboratory, School of Computing, KAIST. His research interests include various aspects of digital watermarking, multimedia forensics, and deep learning.

WONHYUK AHN received the B.S. degree in software and computer engineering from Ajou University, South Korea, in 2016. He is currently pursuing the Ph.D. degree with the School of Computing, Korea Advanced Institute of Science and Technology (KAIST). His research interests include multimedia forensics, computer vision, and machine learning.

MYUNG-JOON KWON received the dual B.S. degrees in mathematics and computer science from Sogang University, in 2019. He is currently pursuing the M.S. degree with the Multimedia Computing Laboratory, School of Computing, Korea Advanced Institute of Science and Technology (KAIST). His research interests include multimedia forensics and computer vision.

HEUNG-KYU LEE received the B.S. degree in electronics engineering from Seoul National University, Seoul, South Korea, in 1978, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 1981 and 1984, respectively. Since 1986, he has been a Professor with the School of Computing, KAIST. He was a Technical Director with Digital Innotech Company. He has authored/coauthored over 200 international journal and conference papers. His major research interests include digital watermarking, digital fingerprinting, multimedia forensics, and digital rights management. He serves as a Reviewer for many international journals, including *Journal of Electronic Imaging*, *Real-Time Imaging*, and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.