A Transformer based Approach for Image Manipulation Chain Detection

Jiaxiang You[1], Yuanman Li[1*], Jiantao Zhou[2], Zhongyun Hua[3], Weiwei Sun[2], Xia Li[1]
[1]Guangdong Key Laboratory of Intelligent Information Processing, College of Electronics and Information Engineering, Shenzhen University, China
[2]State Key Laboratory of Internet of Things for Smart City, Department of Computer and Information Science, University of Macau, Macau
[3] School of Computer Science and Technology, Harbin Institute of Technology Shenzhen, Shenzhen, China
2070436047@email.szu.edu.cn,{yuanmanli,lixia}@szu.edu.cn,huazhongyun@hit.edu.cn,jtzhou@um.edu.mo

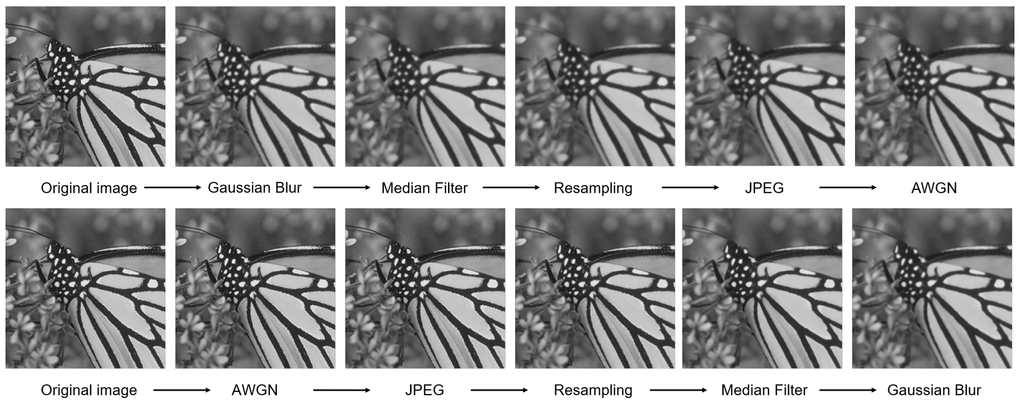


Figure 1: An image is processed by two kinds of manipulation chains consisting of five operations. The above chain is Gaussian Blur→Media Filter→Resampling→JPEG→AWGN, while the below chain is AWGN→JPEG→Resampling→Media Filter→Gaussian Blur.

ABSTRACT

Image manipulation chain detection aims to identify the existence of involved operations and also their orders, playing an important role in multimedia forensics and image analysis. However, *all* the existing algorithms model the manipulation chain detection as a classification problem, and can only detect chains containing up to two operations. Due to the exponentially increased solution space and the complex interactions among operations, how to reveal a long chain from a processed image remains a long-standing problem in the multimedia forensic community. To address this challenge, in this paper, we propose a new direction for manipulation chain detection. Different from previous works, we treat the manipulation chain detection as a machine translation problem rather than a classification one, where we model the chains as the sentences of a target language, and each word serves as one possible image operation. Specifically, we first transform the manipulated image into a deep feature space, and further model the traces left by the manipulation chain as a sentence of a latent source language. Then, we propose to detect the manipulation chain through learning the mapping from the source language to the target one under a machine translation framework. Our method can detect manipulation chains consisting of up to five operations, and we obtain promising results on both the short-chain detection and the long-chain detection.

*Corresponding author.

CCS CONCEPTS

• **Security and privacy** → *Authorization*; *Software and application security*; **Domain-specific security and privacy architectures**; *Cryptanalysis and other attacks.*

KEYWORDS

Manipulation chain detection; image forensics; machine translation; transformer

ACM Reference Format:
Jiaxiang You[1], Yuanman Li[1*], Jiantao Zhou[2], Zhongyun Hua[3], Weiwei Sun[2], Xia Li[1] . 2021. A Transformer based Approach for Image Manipulation Chain Detection. In *Proceedings of the 29th ACM International Conference on Multimedia (MM ’21), October 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3474085.3475513

1 INTRODUCTION

With the development of powerful and user-friendly image editing techniques, users can make visually realistic image forgeries at a very low cost. Nowadays, the forged images are becoming widespread in fake news, insurance fraud, Internet rumors and dishonest academic publications, negatively affecting many aspects of our lives [23]. When altering a digital image, different types of image operations are usually applied sequentially, either to make the forged image look realistic or to hide the traces of previous manipulations. The widely used operations include Gaussian blur, median filtering, resampling, JPEG compression, etc. As a result, detecting the manipulation chain becomes a very critical problem in determining the authenticity and the processing history of digital images [17].

It has been observed that different types of operations would leave unique tampering fingerprints on the processed images. During the past decade, the majority of the efforts focused on designing forensic algorithms to detect a single targeted manipulation [6, 15, 16, 18–20]. Through extracting the features related to those fingerprints and modeling their statistical behaviors, a large number of forensic techniques have been devised for the detection of particular manipulation types. Some representative examples are median filtering [6, 13], contrast enhancement [4], JPEG compression [3, 18] and resampling [19, 20, 26]. Though these methods have achieved noticeable advances, they suffer from an important drawback. Namely, in practical scenarios, we generally have no prior information on which operation has been applied to the forged image. Then, an analyzer needs to run multiple forensic tests to ensure the image authenticity. However, it is rather challenging to fuse the results of multiple detectors for a reliable decision.

To address the above issue, recent researches focused on developing general-purpose image forensic methods, which aimed to determine whether an image was tampered from a set of possible image manipulations in a single test. Some examples include the methods based on hand-crafted features [11, 14, 21], and the ones based on deep features [2, 9, 24]. Such methods, though achieved a great success in the development of general purpose image manipulation detection, they can only identify the existence of a single operation.

In practice, forging an image commonly involves multiple processing operations. Fig. 1 gives two examples to show that an image is processed by five operations from a set of possible operations. This brings new challenges for the previous forensic methods. First, it requires the methods to identify all the operations applied to the image, which exceeds the ability of previous algorithms for a single operation detection. It should be noted that though it seems possible to run multiple tests to identify potential operations, it is very difficult to control the overall false alarm rate among several detectors. Besides, since the traces left by previous operations could be weakened or even erased by the following ones, the performance of each detector will also be highly degraded. Second, in order to obtain the complete manipulation history, thus to know how the forged image has undergone processing, investigators also expect to reveal the order of the involved operations. This makes the solution space grow exponentially as the number of operations increases.

Due to these challenges, limited progress has been made to the problem of manipulations chain detection. The works [8, 22] formulated the problem of detecting the order of operations into a multiple hypotheses testing problem, and then studied the question of when we can or cannot detect the operation order through an information theoretical framework. Bayar *et al.* [2] proposed a constrained convolutional neural network to detect the image manipulation chain consisting of up to two image operations, where the first layer was forced to learn the prediction filters. Recently, Liao *et al.* [17] devised a two-stream convolutional neural network for detecting a chain of two image operations, where one stream detected the tampering artifacts in the spatial domain, while the other stream extracted the local residual features in the transform domain. The major drawbacks of the existing manipulation chain detection methods can be summarized below:

- All the existing algorithms only focused on identifying the chain containing up to two operations. How to reveal the longer chain is still an open problem. It should be noted that the solution space will exponentially increase as the chain getting longer. For example, two operations yield only $\sum_{i=0}^{2} A_2^i = 5$ possible solutions, while five operations result in $\sum_{i=0}^{5} A_5^i = 326$ possible solutions. Even worse, the later applied operations may affect and disguise the traces left by the previous operations. This interplay further makes it hard (or even impossible) to detect the long chains.
- All the previous algorithms formulated the chain detection as a classification problem. Though this seems very straightforward by assigning each case with a unique label, such a strategy cannot fully exploit and utilize the *partial order* in a long manipulation chain. For example, in Fig. 1, if we can reveal some partial orders such as Gaussian Blur⟶Median Filter in the tampered image, then the problem of detecting the remaining sub-chains could be significantly simplified.

In this paper, we propose a novel deep framework for image manipulation chain detection. Different from existing algorithms treating it as a classification problem, we strategically formulate it into a machine translation problem. Basically, we regard the manipulation chain as a sentence of the target language, where the words of the language denote the possible image operations, such as median filtering, resampling and JPEG compression. Our goal is to seek a mapping from the tampered image space to the target language space. To this end, we first transform the tampered image into a deep feature space, and further model the traces of each manipulation chain as a fixed-length sentence of a latent language. Then, the obtained sentence of the source language are translated into a sentence of our target language, and finally the manipulation chain is decoded.

To the best of our knowledge, this is the *first* work capable of detecting various lengths of the manipulation chains (from 0 to 5 in our experiment). Also, this is the *first* work to model the detection of manipulation chain as a machine translation problem rather than a classification one. The main contributions of our work are summarized below:

- We propose a novel image manipulation chain detection algorithm based on machine translation. Different from existing

algorithms that only considered the detection of manipulation chain containing at most two operations, our method is able to detect much longer chains.
- Compared with those classification based methods, our approach detects the manipulation chain one operation by one operation. The former detected operations and their orders will be regraded as the prior information for the next operation detection, thus facilitating the detection process. Such a progressively detecting manner also makes our method more intuitive.
- Experimental results show that our method achieves promising performance for both the short chain detect and the long chain detection.

2 PROPOSED METHOD

2.1 Problem Formulation

In this paper, we define the manipulation chain as an ordered sequence of operations applied to an image I[1], and we suppose that operations are from a set $\{O_1, O_2, ..., O_M\}$. Obviously, for a manipulation chain of length N, the number of possible chains is A_M^N. Considering a chain detection problem with the maximum length of the chains being N ($N \leq M$), the number of possible distinctive chains is

$$K = A_M^0 + A_M^1 + A_M^2 + ... + A_M^N, N \leq M. \tag{1}$$

Note that Eq. (1) counts the chains of length 0, i.e., no operation has been applied to the image.

Let $\mathcal{T}_{chain} = \{T_1, T_2, ..., T_K\}$ be the set containing all the possible chains. Then, the goal of manipulation chain detection is to find the exact operations and their orders for a tampered image. For a given chain T_i, we define its sub-chains as consecutive sub-sequences of T_i. For example, the sub-chains of $T_i = O_1 \rightarrow O_2 \rightarrow O_3$ are O_1, O_2, O_3, $O_1 \rightarrow O_2$, $O_2 \rightarrow O_3$, and $O_1 \rightarrow O_2 \rightarrow O_3$. We should emphasize that usually, finding out some long sub-chains of the complete one is also helpful for forensic analysis.

It can be readily seen that the length of $\mathcal{T}_{chain}$ will exponentially increase as N and M getting larger. In fact, previous works can only identify the chain of at most two operations. To address this challenging problem, we strategically formulate the manipulation chain detection as a machine translation problem. In our paper, we consider 5 kinds of operations, which totally results in 326 possible manipulation chains. Table. 1 summarizes all the operations and their parameters, which are consistent with the work [2]. For example, $< 3, 4, 5, 6, 7 >$ means that an image is applied with the manipulation chain GB→MF→RS→JPEG→AWGN. In addition to these the operations, the other three auxiliary symbols list in Table 1 are used for a translator.

2.2 Manipulation Chain Detection Framework based on Transformer

The framework of our proposed algorithm is illustrated in Fig. 2, which mainly contains three components: 1) hierarchical feature extraction; 2) source language construction and embedding; and 3) source-to-target language translation.

[1]For simplicity, we assume that the operations of the chain are not repetitive.

Table 1: Operation dictionary.

index	Operation	Parameter
0	Padding Symbol	-
1	Start Symbol	-
2	End Symbol	-
3	Gaussian Blur (GB) with $\sigma = 1.1$	kernel size = 5
4	Median Filter (MF)	kernel size = 5
5	Resampling using bilinear interpolation (RS)	Scaling = 1.5
6	JPEG Compression (JPEG)	QF = 70
7	Addictive White Gaussian Noise (AWGN)	$\sigma = 2$

2.2.1 *Hierarchical Feature Extraction.*

Learning a good feature representation is crucial for the manipulation chain detection. Due to the interactions between different operations, some traces could be highly weakened. In this paper, we adopt a deep architecture for extracting the feature representation of the traces left by mixed operations. As shown in Fig. 3, our feature extractor has tree types of layers marked in different colors. (1) Conv+BN+ReLU: for the first layer, we adopt 64 filters of size 3×3 to produce 64 feature maps. Then the batch normalization (BN) and the rectified linear units (ReLU) are applied for the feature normalization and nonlinear mapping. (2) MaxPool: in order to generate more compact features, the MaxPooling of size 2×2 with a stride 2 is employed, where each of them will reduce the size of the input feature maps to the quarter size. (3) ResBlock: a set of residual blocks are stacked to learn the hierarchical feature representation of different operations. According to [12], each residual block consists of two Conv layers of C kernels with a skip connection.

For simplicity, we write the feature extractor as $\mathcal{F}_{fe}(\cdot)$. Then given an input image x, we can generate its deep representation as

$$F_o = \mathcal{F}_{fe}(x). \tag{2}$$

The dimension of F_0 is $C \times H' \times W'$, where the C is the channel size, H' and W' denote the height and the width of the feature maps, respectively.

2.2.2 *Source Language Construction and Embedding.*

Transforming the deep feature F_0 into a time series signal, carrying the trace information left by the chain is necessary and also important for the machine translation procedure. Define the transformation function as $\mathcal{T}_{source}(\cdot)$, and then the time series signal can be formulated as

$$S_o = \mathcal{T}_{source}(F_o). \tag{3}$$

Intuitively, the function $\mathcal{T}(\cdot)$ maps features into *a latent source language space*, where the time series signal S_o can be regarded as a sentence of the source language. In our work, we simply implement the function $\mathcal{T}_{source}(\cdot)$ as a Conv+BN+ReLu layer concatenated with channel flattening. The convolutional layer has 32 filters of size 3×3. Each flattened channel is then treated as a word of the latent source language. Namely, the function $\mathcal{T}_{source}(\cdot)$ eventually maps the deep feature into a fixed length sentence of 32 words.

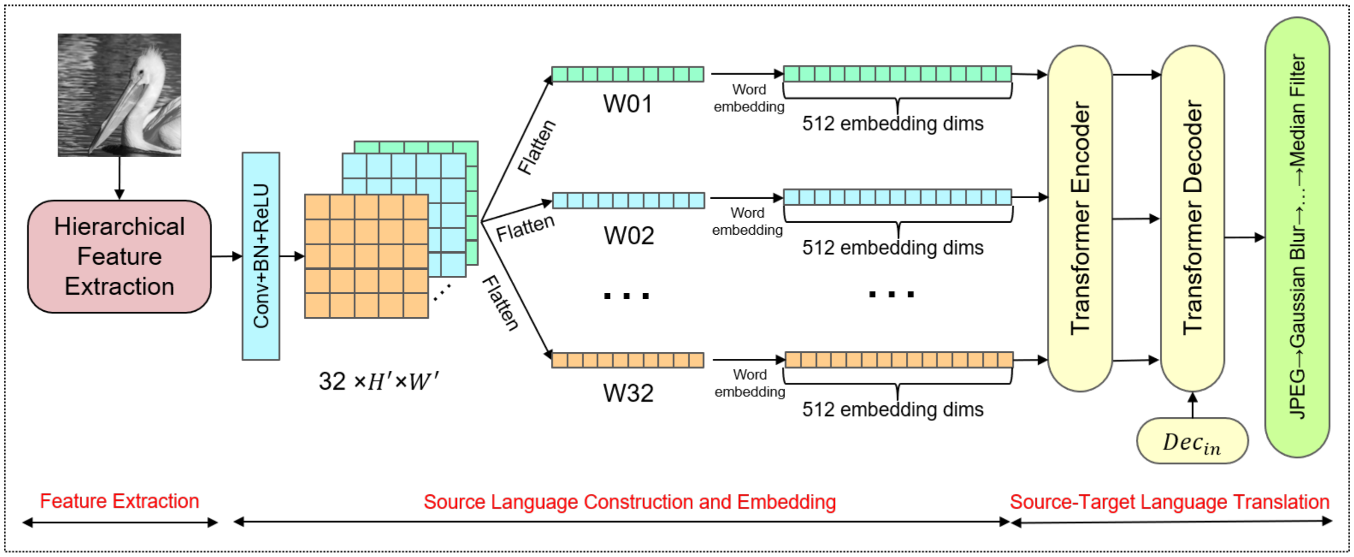


Figure 2: The framework of our proposed method.

Denote $S_o = [W_{01}, ..., W_{32}]$, where each word $W_i \in \mathbb{R}^{H'W'}$. Then, we further embed W_i into a word vector of dimension 512. Mathematically,

$$v_i^s = Embed_s(W_i),\ i \in \{01, ..., 32\}. \quad (4)$$

In our work, we implement the embedding function for the constructed source language using a fully connected layer. With the first two components of our framework, we finally transform an image to an embedded sentence of 32 words of the source language. We write the embedded sentence as $V^s = [v_1^s, v_2^s, v_3^s ... v_{32}^s] \in \mathbb{R}^{32\times512}$.

2.2.3 Source-to-Target Language Translation. Different from the previous classification based algorithms, which assigned each manipulation chain as a unique label. In this work, we model the manipulation chain as a sentence in the target language domain, where each word of the sentence serves as an operation. Note that the length of the target sentence is varying as the manipulation chain changes.

Recently, Transformer [25] has become a prevalent model in Natural Language Processing(NLP), and also has been successfully employed in a wide range of computer vision tasks, such as image enhancement [7] and object detection [5, 10]. In this work, we adopt Transformer [25] as the translator from the source language to the target language. Similar to other machine translation frameworks, Transformer consists of an encoder and a decoder as shown in Fig. 2.

At the encoder side, each embedded vector is first added to a position embedding, thus to maintain the relative position of the words of a sentence. In this work, we simply use the positional encoding strategy proposed in [25], where the position is encoded by sine and cosine functions of different frequencies. For the i-th position, its position embedding vector $f_i \in \mathbb{R}^{1\times512}$ is calculated as

$$\begin{aligned} f_i(2j) &= sin(i/10000^{2j/512}),\\ f_i(2j+1) &= cos(i/10000^{2j/512}), \end{aligned} \quad (5)$$

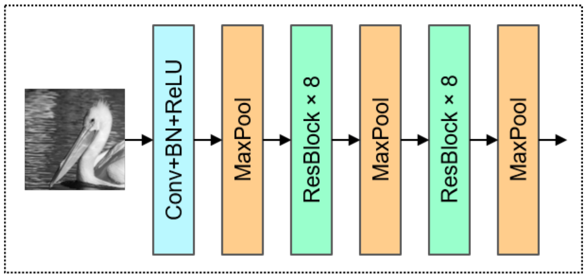


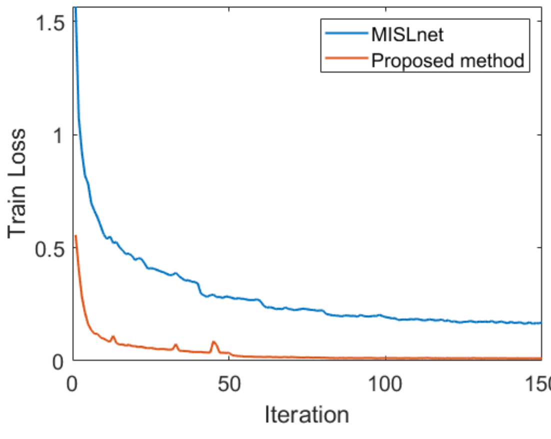
Figure 3: Illustration of the feature extractor.

where $j = 0, ..., 255$. The resulting vectors with the position coding are then input to the encoder. According to [25], we build Transformer encoder as L blocks, each of which consists of a multi-head self-attention module (MSA) and a feed forward network (FFN) with residual connections. The procedure can be written as

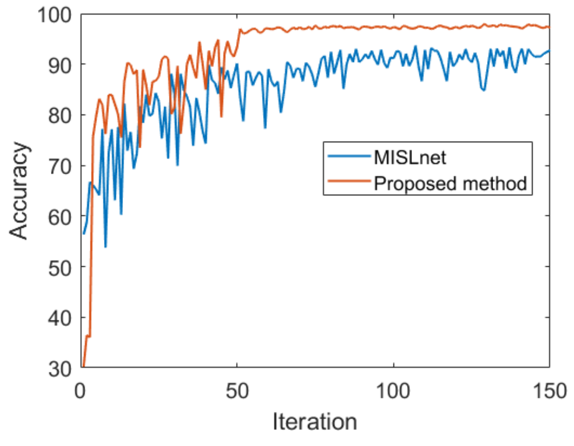
$$\begin{aligned} &V_0^s = [v_1^s + f_1, v_2^s + f_2, ... v_{32}^s + f_{32}],\\ &T_{\ell-1}^s = LN(MSA(V_{\ell-1}^s, V_{\ell-1}^s, V_{\ell-1}^s) + V_{\ell-1}^s),\ \ell = 1, ..., L,\\ &V_\ell^s = LN(FFN(T_{\ell-1}^s) + T_{\ell-1}^s),\ \ell = 1, ..., L,\\ &[z_1, z_2, z_3 ... z_{32}] = Z^s = V_L^s. \end{aligned} \quad (6)$$

Here f_i is the embedded position, LN denotes the Layernorm, and FFN is composed of two fully-connected layers with the ReLU activation. $Z^s \in \mathbb{R}^{32\times512}$ is the output of the encoder. The decoder of Transformer takes Z^s as the inputs, and outputs the probability of each operation of the target manipulation chain. Similar to the word embedding process of the source language, we embed the words, i.e., the operations of a manipulation chain, into a vector space of dimension 512. Mathematically,

$$v_i^t = Embed_t(H_i^t),\ i \in \{1, ..., N\}. \quad (7)$$



(a)



(b)

Figure 4: Training behaviors of MISLnet [2] and our proposed method. (a) the curves of the training loss; (b) the curves of the accuracy for the testing data.

In the training stage, H_i^t denotes the i-th operation of the target manipulation chain, while in the testing stage, it is the decoded i-th operation of our predicted chain. Since the number of possible operations is fixed, we implement the $Embed_t(\cdot)$ function using the *nn.embedding* provided by the Pytorch framework.

The decoder of Transformer has L blocks, where each block contains a masked MSA module (MMSA), a MSA module and a FFN with residual connections. The MMSA ensures that the prediction for the i-th word only depends on the previously decoded $i-1$ words. The decoding procedure can be mathematically written as

$$\begin{aligned}
&V_0^t = [v_1^t + f_1, v_2^t + f_2, ...v_N^t + f_N],\\
&J_{\ell-1} = LN(MMSA(V_{\ell-1}^t, V_{\ell-1}^t, V_{\ell-1}^t) + V_{\ell-1}^t),\ \ \ell = 1, ..., L,\\
&T_{\ell-1}^t = LN(MSA(J_{\ell-1}, Z^s, Z^s) + J_{\ell-1}),\ \ \ell = 1, ..., L,\\
&V_\ell^t = LN(FFN(T_{\ell-1}^t) + T_{\ell-1}^t),\ \ \ell = 1, ..., L,\\
&\hat{y} = Softmax(FC(V_L^t)).
\end{aligned} \tag{8}$$

Here, FC denotes the Fully Connected Layer. In the testing stage, Transformer decodes the chain (sentence) one operation (word) by one operation, and thus the previously decoded sub-chains will be used as prior information for decoding the next operations. Interested readers can refer to [25] for more details about Transformer. Upon having the probability of each operation, the predicted operation is then selected as the one with the maximum probability.

2.3 Loss Function

In our work, we train our proposed framework end-to-end by minimizing the following cross entropy loss

$$loss = CrossEntropyLoss(\hat{y}, y). \tag{9}$$

Here, y denotes the ground truth manipulated chain encoded with the indexes shown in Table. 1 (with one-hot encoding), and $\hat{y}$ is our predicted chain with the predicted probability of each operation.

3 EXPERIMENTAL RESULTS

In this section, we evaluate the performance of our proposed method, which is implemented by the Pytorch framework. All the experiments are conducted on a desktop running Ubuntu 18.04 with two Nvidia 3090 GPUs.

3.1 Implementation Details

We now give the implementation details of our proposed framework. The number of the filters in the ResBlocks is set as 64. Both the Transformer encoder and decoder have 6 layers, i.e., $L = 6$. When training the network, we set the batch size as 128, and the number of epochs as 150. We adopt Adam optimizer with the initial learning rate 1e-4.

3.2 Dataset

We adopt the RAISE-1k dataset [2] to evaluate the performance of our method [1]. The RAISE-1k dataset provides 900 high resolution images, where 800 images for training, and 100 images for validation. The resolution of the images is about 3200×4800. All the images are converted into grayscale and down-sampled by a factor 0.5. We use the 800 images in the train folder for training, while the other 100 images for testing. For the training set, we crop the images into image patches of 256×256 with a stride of 256, and remove those patches without textures. This results in about 30,800 image patches in the train set. For the testing set, we similarly generate about 3,500 image patches.

In each training epoch, we randomly select a varying number of operations from the dictionary shown in Table. 1, and consecutively apply them to an image patch.

[2]http://loki.disi.unitn.it/RAISE/

Table 2: Confusion matrices for identifying manipulation chains for MISLnet [2] and our proposed method

MISLnet [2]: ACC = 93.07%										
	OR	MF	GB	RS	MF-GB	GB-MF	MF-RS	RS-MF	GB-RS	RS-GB
OR	99.16%	0.42%	0.00%	0.39%	0.00%	0.03%	0.00%	0.00%	0.00%	0.00%
MF	0.09%	88.40%	0.00%	0.00%	0.00%	0.20%	0.66%	10.37%	0.29%	0.00%
GB	0.00%	0.00%	89.76%	0.18%	0.44%	0.00%	0.21%	0.00%	0.00%	9.42%
RS	0.03%	0.00%	0.55%	99.25%	0.00%	0.00%	0.14%	0.03%	0.00%	0.00%
MF-GB	0.00%	0.00%	0.11%	0.00%	94.21%	1.67%	0.58%	0.00%	2.64%	0.78%
GB-MF	0.00%	0.03%	0.00%	0.00%	0.60%	91.41%	1.75%	5.49%	0.72%	0.00%
MF-RS	0.00%	0.00%	0.00%	0.14%	0.00%	0.03%	99.80%	0.00%	0.03%	0.00%
RS-MF	0.00%	2.97%	0.06%	0.06%	0.00%	16.28%	2.17%	76.01%	2.45%	0.00%
GB-RS	0.00%	0.00%	0.00%	0.00%	0.15%	0.09%	0.15%	0.00%	99.44%	0.18%
RS-GB	0.00%	0.00%	2.95%	0.00%	0.09%	0.00%	0.00%	0.00%	0.79%	96.17%
Proposed Method: ACC = 97.47%										
	OR	MF	GB	RS	MF-GB	GB-MF	MF-RS	RS-MF	GB-RS	RS-GB
OR	**100.00%**	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
MF	0.00%	**96.06%**	0.00%	0.00%	0.00%	0.16%	0.13%	3.65%	0.00%	0.00%
GB	0.00%	0.00%	**94.97%**	0.17%	0.10%	0.00%	0.00%	0.00%	0.00%	4.76%
RS	0.00%	0.00%	0.19%	**99.68%**	0.00%	0.00%	0.13%	0.00%	0.00%	0.00%
MF-GB	0.00%	0.00%	0.03%	0.00%	**98.69%**	1.25%	0.00%	0.00%	0.00%	0.03%
GB-MF	0.00%	0.13%	0.00%	0.00%	0.58%	**96.96%**	0.00%	2.23%	0.10%	0.00%
MF-RS	0.00%	0.00%	0.00%	0.10%	0.00%	0.00%	**99.90%**	0.00%	0.00%	0.00%
RS-MF	0.00%	3.36%	0.00%	0.00%	0.00%	4.65%	0.13%	**91.72%**	0.13%	0.00%
GB-RS	0.00%	0.00%	0.00%	0.00%	0.00%	0.13%	0.00%	0.03%	**99.78%**	0.06%
RS-GB	0.00%	0.00%	2.68%	0.00%	0.03%	0.00%	0.00%	0.03%	0.32%	**96.94%**

3.3 Metrics

We adopt two metrics for the evaluation of our method. The first metric is Accuracy (ACC), which is calculated by

$$ACC = \frac{\text{number of correctly predicted chains}}{\text{number of testing chains}}. \tag{10}$$

Note that a chain is correctly predicted only when the predicted chain exactly matches the ground-truth one.

In many scenarios, revealing some long sub-chains from the complete one is helpful for image analysis. In this work, we also employ another metric, called the accuracy of longest matched sub-chain (ALMS) for the performance evaluation. Formally, the ALMS metric is defined as

$$ALMS = Average\left(\frac{len(\text{longest matched sub-chain})}{len(\text{target chain})}\right), \tag{11}$$

where $len(\cdot)$ computes the length of a chain, and $Average(\cdot)$ calculates the average over all the testing chains. For example, given a target manipulation chain $< 7, 5, 6, 3, 4 >$, and our predicted chain $< 6, 3, 4 >$, this prediction result assigns a zero score to ACC metric, but contributes a value 0.6 when computing ALMS.

3.4 Results

3.4.1 Detecting Chain of Two Operations. For the comparison purpose, in the first experiment, we aim to detect the manipulation chain of at most two operations as studied in [2, 17]. In our work, we compare our method with MISLnet proposed in [2], since its source code is available. Note that MISLnet modeled the chain detection as a classification problem, while we formulate it as a machine translation problem.

To be consistent with [2], three operations are considered in this experiment, i.e., median filtering, Gaussian blurring and resampling. The parameters of these operations are list in Table. 1. For each image patch, it is manipulated using an operation chain consisted of up to two of these three operations. This totally results in $A_3^0 + A_3^1 + A_3^2 = 10$ different kinds of chains.

Fig. 4 depicts the training curves of different algorithms. Different from MISLnet directly assigning each chain with a unique label, we treat the chain as a sequence, thus successfully revealing sub-chains can also decrease the loss. As can been seen from Fig. 4(a), our converged training loss is much smaller than that of MISLnet. Fig. 4(b) exhibits the ACC curves of different algorithms, where we can observe that our method performs much better than MISLnet. Furthermore, our method also exhibits a more stable ACC curve at the tail. Overall, MISLnet achieves 93.07% ACC, while our method obtains 97.47%. Table 2 summarizes the detailed numerical results in confusion matrices. It can be observed that our method beats MISLnet on all the cases. Specifically, MISLnet performs bad on the chains MF and RS→MF, where our method outperforms it by above 7% and 15% in terms of ACC, respectively. The results shown in Table 2 demonstrate the superiority of modeling the chain detection as a machine translation problem.

Table 3: Results for known-length manipulation chain detection.

Model	A_5^1	A_5^2	A_5^3	A_5^4	A_5^5
ACC	99.98%	99.02%	90.58%	81.56%	76.78%
ALMS	99.98%	99.51%	95.64%	91.72%	88.59%

Table 4: Results for unknown-length manipulation chain detection.

ACC = 71.78%, ALMS = 90.29%						
L_c \ L_p	0	1	2	3	4	5
0	**98.45%**(0.00%)	1.53%	0.02%	0.00%	0.00%	0.00%
1	2.46%	**94.44%**(0.00%)	2.93%	0.12%	0.06%	0.00%
2	0.00%	11.32%	**80.34%**(0.41%)	5.97%	1.52%	0.45%
3	0.00%	2.34%	17.74%	**61.22%**(3.66%)	11.15%	3.89%
4	0.00%	0.08%	2.97%	24.35%	**44.73%**(8.11%)	19.76%
5	0.00%	0.00%	0.20%	4.55%	28.01%	**51.00%**(16.23%)

3.4.2 Detecting Chain of Known Length. In this section, we evaluate the manipulation chain detection performance with a fixed known length. Specifically, we consider five cases where the length of manipulation chains ranges from 1 to 5. For each image from the training set, we manipulate it by applying the chain of a given length, where the operations are randomly selected from Table. 1. The number of possible chains are A_5^1, A_5^2, A_5^3, A_5^4 and A_5^5, respectively, for these five cases.

We summarize the results in Table 3. We can see that ACC drops as L_c increases. However, considering the exponentially increased solution space, the ACC performance is still promising even when L_c is large. For example, our method achieves 76.78% ACC when $L_c = 5$. Note that our method detects the manipulation chain progressively, where the previously detected operations could greatly help the next operation detection. This desirable property endows our method the capability to reveal sub-chains. The experimental results shown in Table 3 also validate our conclusion. Compared with ACC, the performance degradation of ALMS is much more graceful. For example, when $L_c = 5$, our method still obtains 88.59% ALMS, which is much larger than the corresponding ACC. Such a phenomenon implies that though our method may fail to correctly detect the complete chains in some cases, it can still reveal many long sub-chains, especially when L_c is large.

3.4.3 Detecting Chain of Unknown Length. In many practical scenarios, the length of the manipulation chain is generally unknown. In this paper, we also evaluate the performance of our method for the manipulation chain detection of unknown length, where L_c is randomly selected from 0 to 5. This yields $\sum_{i=0}^{5} A_5^i$ number of possible chains. In order to balance different lengths of chains, we generate *the same number of manipulated images for each length* during training and testing.

Table 4 reports the experimental results. It should be noted that this is the first work to address the problem of detecting manipulation chain up to five operations, and our method obtains an overall 71.78% ACC and 90.29% ALMS, which are promising. For a deep analysis, we also report the probability of a manipulation chain of one length to be detected as a chain of another length. L_p in Table 4 denotes the length of the detected chain, and L_c is the length of the target chain. The bold numbers on the diagonal represent the ACCs for each length, while the numbers in parentheses denote the error rates that the detected chain has the correct length but incorrect operations or orders. For example, in the case of $L_c = 2$, 80.34% manipulation chains have been correctly detected, and 11.32%, 5.97%, 1.52% and 0.45% chains have been wrongly detected as other lengths. Besides, there are 0.41% number of detected chains though have the same length, but the operations or orders are incorrect. It can be observed that when $L_c \geq 3$, our method tends to detect the chain with one more or one less operation. This is reasonable due to the complex interactions between operations in a long manipulation chain. An interesting phenomenon is that the ACC of $L_c = 4$ is lower than that of $L_c = 5$. This is because that in the case of $L_c = 5$, only a few number of chains will be wrongly detected as chains of $L_c = 3$.

Before concluding this section, we give some detection examples in Fig. 5. Images in different columns are manipulated with the same chains of length $L_c = 3$, $L_c = 4$ and $L_c = 5$, respectively. Figs. 5(a-c) present the cases that our method successfully reveals the complete manipulation chains, while Figs. 5(d-f) give examples that our method unsuccessfully to detect the complete chains. From Figs. 5(d-f), we can observe that though our method fails to reveal the complete chains, it can still detect many long sub-chains. For example, in Fig. 5(e), the trace of GB operation is weakened by the later operations, making our method fails to detect GB. However, our method can still detect all the other operations and orders. We should note that a full analysis between the chain detectability and the image content seems challenging; but it could be an interesting research direction in the further study.

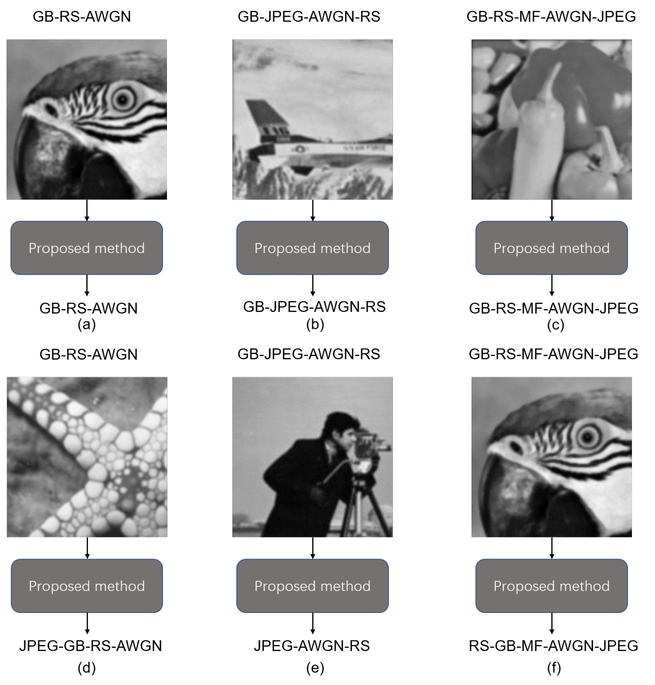


Figure 5: Some detection examples between the manipulation chain and the image content.

4 CONCLUSION

This paper has presented a new direction of the manipulation chain detection. Different from previous algorithms, we treat the manipulation chain detection as a machine translation problem rather than a classification one. Specifically, we model the manipulation chain as a sentence of the target language, and seek a mapping from the image space to the target language space. To this end, we first propose to transform the image to a latent source language space, where the traces left by the manipulation chain are properly described as sentences. Then, we propose to learn the mapping from the source language to the target language, and finally decode out the manipulation chain. Our method can detect chains containing up to five operations, and extensive experimental results have been provided to show the superiority of our scheme.

ACKNOWLEDGMENTS

This work was supported in part by the Natural Science Foundation of China under Grant 62001304, Grant 61871273 and Grant 61971476; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515110410.

REFERENCES

[1] Eirikur Agustsson and Radu Timofte. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops.* 1122–1131.

[2] Belhassen Bayar and Matthew C Stamm. 2018. Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection. *IEEE Transactions on Information Forensics and Security* 13, 11 (2018), 2691–2706.

[3] Tiziano Bianchi and Alessandro Piva. 2012. Image Forgery Localization via Block-Grained Analysis of JPEG Artifacts. *IEEE Transactions on Information Forensics and Security* 7, 3 (2012), 1003–1017.

[4] Gang Cao, Yao Zhao, Rongrong Ni, and Xuelong Li. 2014. Contrast Enhancement-Based Forensics in Digital Images. *IEEE Transactions on Information Forensics and Security* 9, 3 (2014), 515–525.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *Proceedings of European Conference on Computer Vision.* Springer, 213–229.

[6] Chenglong Chen, Jiangqun Ni, and Jiwu Huang. 2013. Blind Detection of Median Filtering in Digital Images: A Difference Domain Based Approach. *IEEE Transactions on Image Processing* 22, 12 (2013), 4699–4710.

[7] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. 2021. Pre-trained image processing transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 12299–12310.

[8] Xiaoyu Chu, Yan Chen, and KJ Ray Liu. 2016. Detectability of the Order of Operations: An Information Theoretic Approach. *IEEE Transactions on Information Forensics and Security* 11, 4 (2016), 823–836.

[9] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. 2017. Recasting Residual-Based Local Descriptors as Convolutional Neural Networks: An Application to Image Forgery Detection. In *Proceedings of ACM Workshop on Information Hiding and Multimedia Security.* 159–164.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[11] Wei Fan, Kai Wang, and François Cayre. 2015. General-purpose image forensics using patch likelihood under image statistical models. In *Proceedings of IEEE International Workshop on Information Forensics and Security.* 1–6.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 770–778.

[13] Xiangui Kang, Matthew C Stamm, Anjie Peng, and KJ Ray Liu. 2013. Robust Median Filtering Forensics Using an Autoregressive Model. *IEEE Transactions on Information Forensics and Security* 8, 9 (2013), 1456–1468.

[14] Haodong Li, Weiqi Luo, Xiaoqing Qiu, and Jiwu Huang. 2016. Identification of various image operations using residual-based features. *IEEE Transactions on Circuits and Systems for Video Technology* 28, 1 (2016), 31–45.

[15] Yuanman Li and Jiantao Zhou. 2019. Fast and Effective Image Copy-Move Forgery Detection via Hierarchical Feature Point Matching. *IEEE Transactions on Information Forensics and Security* 14, 5 (2019), 1307–1322.

[16] Yuanman Li, Jiantao Zhou, and An Cheng. 2017. SIFT Keypoint Removal via Directed Graph Construction for Color Images. *IEEE Transactions on Information Forensics and Security* 12, 12 (2017), 2971–2985.

[17] Xin Liao, Kaide Li, Xinshan Zhu, and KJ Ray Liu. 2020. Robust Detection of Image Operator Chain With Two-Stream Convolutional Neural Network. *IEEE Journal of Selected Topics in Signal Processing* 14, 5 (2020), 955–968.

[18] Cecilia Pasquini, Giulia Boato, and Fernando Pérez-González. 2017. Statistical Detection of JPEG Traces in Digital Images in Uncompressed Formats. *IEEE Transactions on Information Forensics and Security* 12, 12 (2017), 2890–2905.

[19] Cecilia Pasquini and Rainer Böhme. 2019. Information-Theoretic Bounds for the Forensic Detection of Downscaled Signals. *IEEE Transactions on Information Forensics and Security* 14, 7 (2019), 1928–1943.

[20] Tong Qiao, Ran Shi, Xiangyang Luo, Ming Xu, Ning Zheng, and Yiming Wu. 2019. Statistical Model-Based Detector via Texture Weight Map: Application in Re-Sampling Authentication. *IEEE Transactions on Multimedia* 21, 5 (2019), 1077–1092.

[21] Xiaoqing Qiu, Haodong Li, Weiqi Luo, and Jiwu Huang. 2014. A Universal Image Forensic Strategy Based on Steganalytic Model. In *Proceedings of ACM Workshop on Information Hiding and Multimedia Security.* 165–170.

[22] Matthew C Stamm, Xiaoyu Chu, and KJ Ray Liu. 2013. Forensically determining the order of signal processing operations. In *Proceedings of IEEE International Workshop on Information Forensics and Security.* 162–167.

[23] Matthew C Stamm, Min Wu, and KJ Ray Liu. 2013. Information forensics: An overview of the first decade. *IEEE Access* 1 (2013), 167–200.

[24] Hongshen Tang, Rongrong Ni, Yao Zhao, and Xiaolong Li. 2017. Detection of various image operations based on CNN. In *Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference.* 1479–1485.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of Advances in Neural Information Processing Systems.* 1–11.

[26] David Vázquez-Padín, Fernando Pérez-González, and Pedro Comesana-Alfaro. 2017. A Random Matrix Approach to the Forensic Analysis of Upscaled Images. *IEEE Transactions on Information Forensics and Security* 12, 9 (2017), 2115–2130.