

This ICCV paper is the Open Access version, provided by the Computer Vision Foundation. Except for this watermark, it is identical to the accepted version; the final published version of the proceedings is available on IEEE Xplore.

Localization of Deep Inpainting Using High-Pass Fully Convolutional Network

Haodong Li^{1,2}, Jiwu Huang^{1,2,*}

¹Guangdong Key Laboratory of Intelligent Information Processing and Shenzhen Key Laboratory of Media Security, and National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen, China ²Shenzhen Institute of Artificial Intelligence and Robotics for Society, China

{lihaodong,jwhuang}@szu.edu.cn

Abstract

Image inpainting has been substantially improved with deep learning in the past years. Deep inpainting can fill image regions with plausible contents, which are not visually apparent. Although inpainting is originally designed to repair images, it can even be used for malicious manipulations, e.g., removal of specific objects. Therefore, it is necessary to identify the presence of inpainting in an image. This paper presents a method to locate the regions manipulated by deep inpainting. The proposed method employs a fully convolutional network that is based on high-pass filtered image residuals. Firstly, we analyze and observe that the inpainted regions are more distinguishable from the untouched ones in the residual domain. Hence, a high-pass pre-filtering module is designed to get image residuals for enhancing inpainting traces. Then, a feature extraction module, which learns discriminative features from image residuals, is built with four concatenated ResNet blocks. The learned feature maps are finally enlarged by an upsampling module, so that a pixel-wise inpainting localization map is obtained. The whole network is trained end-toend with a loss addressing the class imbalance. Extensive experimental results evaluated on both synthetic and realistic images subjected to deep inpainting have shown the effectiveness of the proposed method.

1. Introduction

Inpainting is a kind of image editing technique aiming to repair the missing or damaged regions in an image with alternative contents, which imitates the work of art restoration experts. Since around 2000, a variety of inpainting approaches have been developed. Among them, there are two main categories of conventional approaches, *i.e.*, diffusion-based [5, 6, 19] and patch-based [11, 9, 3, 32].



Figure 1. The man with a bag in the original image (left) is removed by the method [16], producing the inpainted image (right).

The diffusion-based approaches can only fill small or narrow areas, such as scratches in old photos. Although the patch-based approaches can fill larger areas, they lack the ability to generate complicated structures or novel objects that are not in the given image. To overcome the limitations of conventional inpainting approaches, many deep learning based inpainting approaches have been designed in recent years [31, 38, 16, 35, 26, 37]. The deep inpainting approaches can not only infer image structures and produce more fine details, but also create novel objects. With the deep inpainting techniques, one can fill a targeted image region with photo-realistic contents.

Although inpainting is usually used for inoffensive purposes, it can also be exploited for malicious intentions. For example, removing objects in an image to fabricate a fake scene, and erasing visible copyright watermarks. Especially, by using the deep inpainting methods, the tampered images could be visually plausible, and the tampered regions are not easy to be distinguished with human eyes. As shown in Figure 1, a key object (the man) within the original image is removed by the inpainting method proposed in [16], producing an inpainted image which still looks natural. If such inpainted images are shown in court as evidences or used to report fake news, it would inevitably lead to many serious issues. Therefore, it is necessary to identify whether an image is manipulated by deep inpainting and locate the inpainted regions.

The identification of manipulated images has been studied in the field of image forensics [34, 18] for more than

^{*}Corresponding author

a decade. A variety of forensic methods were proposed to detect common image processing operations [22, 4] and tampering operations [8, 33, 15]. There were also some works focusing on the forensics of conventional inpainting approaches [23, 7, 21, 40]. However, there is a lack of efforts for detecting deep inpainting in images. Due to the inpainted image regions are perceptually consistent with those untouched ones, deep inpainting forensics is quite different from common computer vision tasks.

In this paper, we introduce an end-to-end method to locate the image regions manipulated by deep inpainting. To our best knowledge, this is the first work on deep inpainting forensics. In the proposed method, we first analyze the differences between the inpainted and untouched regions. We observe that the differences are more obvious in the high-pass filtered residuals. Hence, we design a prefiltering module initialized with high-pass filters to extract image residuals so as to enhance inpainting traces. We then construct a feature extraction module with four ResNet [14] blocks to learn discriminative features from the residuals, and finally use an up-sampling module to predict the pixelwise class label for an image. Extensive experiments have been conducted to evaluate the proposed method. The results show that the proposed method can effectively locate the inpainted regions.

The rest of this paper is organized as follows. Section 2 briefly introduces deep inpainting methods and inpainting forensic methods. Section 3 describes the details of the proposed method for localization of deep inpainting. Section 4 reports the experimental results. Finally, the concluding remarks are drawn in Section 5.

2. Related works

2.1. Deep learning based inpainting

The major difference between the deep learning based inpainting methods and the conventional methods is to utilize large scale dataset to learn semantic representations of images. Hence, deep inpainting methods are able to generate more photo-realistic details compared with the nonlearning ones, and they can even complete areas with new objects that are not in the given image, e.g., part of a human face. To accomplish the inpainting, deep inpainting methods typically employ two sub-networks: a completion network and an auxiliary discriminative network. The former one learns image semantics and infers the contents in holes. Namely, it maps a corrupted image to an inpainted one. The later one enforces the inpainted image is visually plausible via generative adversarial training [13]. Context Encoders [31] is one of the pioneering attempts that use deep network for inpainting. It trains a model with pixel-wise reconstruction loss and adversarial loss, which can inpaint a 64×64 hole within a 128×128 image. Yang et al. [38] proposed a multi-scale neural patch synthesis approach by jointly optimizing image content and texture. This method can generate sharper results on high-resolution image but increases the computation time. Iizuka *et al.* [16] employed a global discriminator and a local discriminator to ensure that both the global image and the local inpainted contents are indistinguishable from real ones. To avoid the negative effects caused by the masked holes, Liu *et al.* [26] designed partial convolution, which masks the convolution operation, to predict the missing regions with only the valid pixels in the given image. Yu *et al.* [39] further tried to automatically mask the convolution operation with gated convolutions, achieving better inpainting results. Xiong *et al.* [37] proposed a foreground-aware inpainting method that predicts the foreground contour as guidance for inpainting.

2.2. Inpainting forensics

Up to now, several methods have been developed for the forensics of conventional inpainting approaches. Li et al. [21] proposed a method to detect diffusion-based inpainting by analyzing local variance of image Laplacian along the isophote direction. To detect patch-based inpainting, Wu et al. [36] exploited the patch similarity measured by zeroconnectivity length. Then, Bacchuwar et al. [2] proposed a jump patch match method to reduce the computational complexity. Both [36] and [2] require the manual selection of suspicious regions and suffer from high false alarm rate. Chang et al. [7] proposed a two-stage searching method to find out the suspicious regions and used multi-region relations to reduce false alarms. Furthermore, Liang et al. [24] employed central pixel mapping to accelerate the search of suspicious regions and used greatest zero-connectivity component labeling followed by fragment splicing detection to locate the tampered regions. Recently, Zhu et al. [40] designed a convolutional neural network (CNN) with encoder-decoder structure to detect patch-based inpainting in 256×256 images. Since deep inpainting approaches tend to generate more detailed image contents and even create new objects in the inpainted images, they introduce different traces into the inpainted regions compared to the conventional ones. Therefore, the forensic methods targeted at conventional inpainting approaches are not suitable for the localization of deep inpainting.

Considering that deep inpainting models are usually trained with a generative adversarial process, some recent research on generated image detection [28, 29, 20] may be adapted for deep inpainting localization. On the other hand, some image splicing localization algorithms [33, 15] can also be exploited to locate the inpainted regions. However, since such methods do not specifically consider the traces left by deep inpainting, their performance is not so satisfactory based on our experiments.



3. Localization of deep inpainting

We propose an end-to-end solution to the localization of deep inpainting as illustrated in Figure 2. The proposed method successively employs three modules to perform localization: firstly, a pre-filtering module initialized with high-pass filters is used to enhance inpainting traces; then, four ResNet blocks are used to learn discriminative features; finally, an up-sampling module is employed to achieve pixel-wise prediction. The proposed network is a fully convolutional network [27] without fully-connected layers, thus it can work on images with arbitrary sizes. The whole network is trained with a loss that addresses the class imbalance between inpainted pixels and untouched pixels. We will elaborate the technical details as follows.

3.1. Enhancing inpainting traces

In image forensics, the key for tampering detection and/or localization is to capture the traces left by tampering operations. As the image contents created by deep inpainting are visually indistinguishable, it is difficult to directly capture the inpainting traces and learn discriminative features from the pixel domain of an image. In many existing forensic methods, a common practice for capturing tampering traces is to perform high-pass filtering on an image to suppress its contents and obtain its residuals [22, 4]. Inspired by these works, we try to investigate whether highpass filtering is helpful for enhancing inpainting traces.

To this end, we use the transition probability of adjacent pixels as the statistic measure of images and compare the differences between untouched image patches and inpainted image patches in two cases. In the first case the statistic measure is directly computed in pixel domain, while in the second case the statistic measure is computed in the residual domain after high-pass filtering. Supposing that an image (or image residual) array I has N gray levels, the transition probability of adjacent pixels will form an N-by-N transition probability matrix (TPM) M, where the element at the



(c) Untouched patches w/ filtering (d) Inpainted patches w/ filtering

Figure 3. Transition probability matrices of untouched/inpainted image patches without/with high-pass filtering.

x, y position of M is represented as

$$\mathbf{M}_{x,y} = \Pr(\mathbf{I}_{i,j+1} = y | \mathbf{I}_{i,j} = x), 1 \le x, y \le N,$$
 (1)

where *i* and *j* indicate the position of an element in **I**. To perform experimental analysis, we randomly select some images from the ImageNet [10] dataset and perform inpainting in the central regions of the selected images by using the method [16]. Then we randomly selected 10,000 16×16 untouched patches and 10,000 16×16 inpainted patches from these images to calculate the TPMs. The average TPMs for untouched patches and inpainted patches are shown in Figure 3 (a) and (b), respectively. Besides, we apply a horizonal first-order derivative filter to the selected patches and calculate the TPMs after filtering. The corresponding average TPMs are shown in Figure 3 (c) and (d). The gray intensities in all the four subfigures are of the same scale.

ſ	0	0	0 -	1	0	0	0 -	1	0	0	0]
	0	-1	0		0	-1	1		0	-1	0
	0	1	0		0	0	0		0	0	1

Figure 4. The initialized filter kernels for the pre-filtering module.

It is observed that the TPMs for untouched and inpainted patches in pixel domain (without filtering) are very similar, while the TPMs in residuals domain (with filtering) exhibit notable disparities outside the dashed-circles. Specifically, the values of transition probabilities in the positions outside the dashed-circle for inpainted patches are much lower than those for the untouched patches, indicating that the inpainted image patches contain less high-frequency components. The reason may be that inpainting focuses on producing visually realistic image contents but fails to imitate imperceptible high-frequency noises that inherently exist in natural images.

The above observations imply that it is beneficial to enhance inpainting traces with high-pass filtering. Therefore, we design a pre-filtering module to process an input image. The pre-filtering module is implemented with depthwise convolutions with the stride of 1. Specifically, each channel of the input image is separately convolved with a set of high-pass filter kernels, and then the convolution results (image residuals) are concatenated together as the input of the subsequent network layers. In the proposed method, the filter kernels are initialized with three first-order derivative high-pass filters, as shown in Figure 4. Hence, for a given RGB image, the pre-filtering module will transform it to a 9-channel image residual. The filter kernels are set as learnable so that their elements can be adjusted by gradient descent during the training process.

3.2. Learning feature with CNN

In order to distinguish between inpainted and untouched regions, it needs to collect effective and discriminative features from the pre-filtered image residuals. Since CNNs have been widely used in many applications to automatically learn feature representations, we construct a feature extraction module based on CNN. This module is built on the basis of ResNet v2 [14], which has shown superior performance in many computer vision applications, including image classification and object detection.

The designed feature extraction module consists of four ResNet blocks, and each block is composed of two "bottleneck" units. In each bottleneck unit, there are three successive convolutional layers and an identity skip connection, where batch normalization and ReLu activation are performed before each convolution operation. The kernel sizes of the three convolutional layers are 1×1 , 3×3 , 1×1 , respectively; the convolution stride is 1 for most layers, except that the last layer in the second unit of each block has

Block	Bottleneck	Bottleneck	Output	Output
	unit	depth	depth	stride
#1	#1	32	128	1
	#2	32	128	2
#2	#1	64	256	1
	#2	64	256	2
#3	#1	128	512	1
	#2	128	512	2
#4	#1	256	1024	1
	#2	256	1024	2

Table 1. The architecture settings of the feature extraction module.

a stride of 2 for pooling and reducing spatial resolution. We follow the original setting of ResNet and let the channel depth for the former two convolutional layers (bottleneck depth) is 1/4 of the depth for the last one (output depth). We set the output depth of the first block as 128 and double the size of depth in each of the subsequent blocks. The detailed settings are summarized in Table 1. In a word, the feature extraction module takes the 9-channel image residual as input and learns 1024 feature maps, whose spatial resolution is 1/16 of the input image.

3.3. Performing pixel-wise prediction

Due to the feature extraction module shrinks the spatial resolution, each spatial position in the output feature maps is corresponded to a certain region of the input image. In order to output a class label for each pixel, we apply transpose convolution to enlarge the spatial resolution. We initialize the kernels of transpose convolutions with the bilinear kernel and let them be learnable during the training. Please note that if resizing the feature maps to the same size of the input image at one time (*i.e.*, $16 \times$ up-sampling), the size of convolution kernel will be very large (32×32) , which will make the training more difficult. Hence, in the up-sampling module we adopt a two-stage strategy, where the spatial resolution is enlarged with two successive transpose convolutions that perform $4 \times$ up-sampling. In this way, the kernel size is significantly reduced to 8×8 . Since up-sampling would not increase information, we set the channel depth in a way that the total elements in feature maps are the same before and after transpose convolution. Hence, the output channels are 64 and 4 for the two transpose convolution layers, respectively. Finally, to deal with the checkerboard artifacts [30] introduced by transpose convolution, we use an additional 5×5 convolution with a stride of 1 to weaken the checkerboard artifacts and simultaneously transform the 4-channel output into 2-channel logits. The logits are then fed to a Softmax layer for classification, yielding the localization map with pixel-wise predictions.

3.4. Dealing with class imbalance

Generally, the inpainted regions are much smaller than the untouched regions, leading to the imbalance between positive and negative samples. If the standard cross entropy loss is used to supervise the training of network parameters, the dominant negative samples will contribute the majority of the loss and thus mainly control the gradient, so the trained model can easily classify the negative samples but perform poor for the positive samples. This will result in low true positive rate, meaning that the inpainted regions cannot be accurately identified.

In order to mitigate the effect of class imbalance, we employed the focal loss proposed in [25]. The focal loss is modified based on the standard cross entropy loss. It assigns a modulating factor to the cross entropy term, so that the weights for the dominant and well-classified negative samples in the total loss is decreased. In such a way, the network will pay more attention to classify the positive samples. Our experimental results show that focal loss achieves better performance than standard cross entropy loss as well as weighted cross entropy loss.

4. Experimental results

4.1. Experimental setup

Image dataset. We prepared the training and testing data by exploiting the ImageNet [10] dataset. The images in this dataset are of different sizes, and all are stored in JPEG files; among them, most images are compressed with a quality factor (QF) of either 96 or 75. Therefore, the two QFs 96 and 75 were considered in our experiments. For each quality factor, we randomly selected 50,000 images from ImageNet [10] as training instances and 10,000 images as testing instances. The selected images were inpainted by applying the method [16] to a 10% rectangle region locating at the image center. We denote these images as synthetic inpainted images. In addition, we select another 10,000 and 100 images to create random inpainted images and realistic inpainted images, respectively. For the random inpainted images, 1 to 3 rectangle regions were randomly selected to perform inpainting, where the regions would locate at any position of the image. The width and height of the inpainted regions were chosen within the range of [64, 96]. For the realistic inpainted images, we manually selected some meaningful objects (e.g., an animal) to perform inpainting. The total area of inpainted regions is about 2%-15% of the whole image. After inpainting, we saved all images as JPEG files with their original QFs to avoid leaving re-compression artifacts.

Implementation details. The proposed network was implemented with the TensorFlow deep learning framework [1]. We adopted the Adam optimizer [17] and set the initial learning rate as 1×10^{-4} . The learning rate decreased 50% after every epoch. Except for the convolutional layers in pre-filtering module and the transpose convolutional layers in up-sampling module, we initialized the kernel weights

with the Xavier initializer [12] and the biases with 0. The L2 regularization was used with a weight decay of 1×10^{-5} . Since the images are of various sizes, the batch size was set as 1. The whole network was trained 5 epochs. During the training procedure, 90% of the training instances (45,000) were used for learning and updating network parameters, while the left 10% (5,000) were used for validation. We adopted an early stop strategy by watching over the F1-score for the validation data. Namely, the model with the highest validation F1-score was saved as the final model. The training and testing were carried out with a Nvidia Tesla P100 GPU.

Comparative study. Four methods were adopted for comparison. The first method, proposed in [20], exploits a feature set that represents the disparities of color components (DCC-Fea) to detect images generated by GANs. We employed the DCC-Fea to detect inpainted image patches in our experiments. The second method is based on a patchwise CNN network (Patch-CNN), whose structure is similar to the proposed one, except that there is no up-sampling module. Patch-CNN takes a fixed-sized image patch as input and applies global average pooling to the learned feature maps, and finally output a class label for the input patch. Both DCC-Fea and Patch-CNN first perform classification with sliding-window for an image patch-by-patch, and then integrate the predictions of all patches into the localization map. The remaining two methods are proposed in [33] and [15], respectively. The former one uses a multi-task fully convolutional network (MFCN) for splicing localization. We re-trained this MFCN for inpainting localization. The later one employs self-supervised learning to check the consistency of image regions. We used the trained model released in [15] to detect the inpainted regions.

Performance metrics. Four commonly used pixel-wise classification metrics, including recall, precision, Intersection over Union (IoU), and F1-score, are adopted to evaluate the performance. The metrics are calculated on each image independently, and the mean values obtained over all images are reported in the following experiments.

4.2. Ablation study

In this experiment, we conducted an ablation study to show the superiority of the proposed method over its variants. This experiment was conducted on the synthetic inpainted images with QF=75. We first modified some settings of proposed network, including the initialization of pre-filtering kernel, the up-sampling method, and the type of loss function. Then, we trained different models with different settings to locate the inpainted regions in the testing images. The obtained results are shown in Table 2. From the results we obtain the following observations.

On pre-filtering kernel. Initializing the pre-filtering kernels with 1st-order derivative filters achieves the best perfor-

Pre-filtering kernel	1st-order derivation 2nd-order derivation 3rd-order derivation No filtering	√	√	\checkmark	\checkmark	V	V	\checkmark
Up-sampling method	Transpose conv Bilinear resizing	√	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark
Loss function	Focal loss Standard cross entropy Weighted cross entropy	V	\checkmark	\checkmark	\checkmark	V	\checkmark	\checkmark
Recall Precision IoU F1-score			82.71 95.17 79.42 86.60	38.85 94.45 38.21 49.59	82.22 92.16 76.63 84.54	87.13 93.84 82.24 89.13	89.31 95.14 85.41 91.17	90.54 92.66 84.41 90.69

Table 2. Localization results (%) obtained by different variants of the proposed method.

	b=16 $b=22$ $b=64$		1 b-64	Patch-CNN h=16 $h=32$ $h=64$			Self-	MFCN	Proposed	
		0=10	0=32	0=04	0=10	0=32	0=04	consistency		
	Recall	78.36	74.79	64.52	79.03	74.25	58.77	85.27	83.17	96.89
OE-06	Precision	47.52	60.98	69.61	82.77	96.01	98.59	23.67	80.39	97.97
QF=90	IoU	38.48	45.72	44.26	66.46	71.49	58.25	20.16	69.77	94.99
	F1-score	52.13	59.25	58.16	78.49	82.90	72.80	31.64	80.23	97.28
	Recall	70.30	67.83	63.13	74.29	79.21	73.52	83.08	57.15	89.84
OF 75	Precision	25.72	31.94	37.51	34.53	57.64	83.22	24.57	53.04	95.22
QF=/5	IoU	20.86	24.18	26.62	28.57	47.66	62.62	21.07	33.73	85.93
	F1-score	33.03	36.79	38.88	42.83	62.56	75.15	32.92	45.01	91.53

Table 3. Localization results (%) of different methods for images with QF=96 and QF=75.

mance. The performance degrades when using higher order derivative filters, implying that the inpainting traces mainly present in the 1st-order derivative signal. If no pre-filtering is used, the network needs to learn from pixel domain directly, resulting in unsatisfactory performance. These results indicate that the use of appropriate high-pass filters is important for locating the inpainted regions.

On up-sampling method. As an alternative of transpose convolution, we tried to bilinearly resize the learn feature maps to the logits. According to the results, bilinear resizing underperforms transpose convolution by about 2%. This is because the up-sampling weights can be tuned in transpose convolution, which is helpful for improving performance.

On loss function. Among the three investigated loss functions, the focal loss achieves the best precision, IoU, and F1-score. The results for standard cross entropy loss are slightly lower than those for focal loss, since it suffers from the effect of class imbalance. The weighted cross entropy loss achieves the highest recall but get relatively lower precision, so its overall localization performance measured by IoU or F1-score is not so good.

4.3. Test on synthetic inpainted images

In this experiment, we evaluated the localization performance for the synthetic inpainted images. The training and testing were carried out independently for JPEG quality factors 96 and 75. Due to the use of sliding-window in DCC-Fea and Patch-CNN, the window size b is a crucial parameter. To make the comparison fair, we considered three window sizes, *i.e.*, $b \in \{16, 32, 64\}$. Since the values in the localization maps obtained by all methods are within the range of [0,1], we applied thresholding to the localization maps when computing the performance metrics. For the self-consistency method [15], we followed its specific settings and picked the optimal threshold for each testing image (though this may not be applicable in practice). For the other methods, we performed the thresholding with a default value of 0.5. The obtained results for different methods are shown in Table 3. From this table, it can be observed that the proposed method significantly outperforms its competitors on all the metrics. When the quality factor is high (QF=96), the re-compression after inpainting has little effect on the inpainting traces, so the proposed method can accurately locate the inpainted regions, achieving a F1score higher than 0.97. When the quality factor is in a middle level (QF=75), the inpainting traces are disturbed by recompression to some extent, so that the recall of the proposed method decreases to 89%, leading to the degradation of IoU and F1-score. Nevertheless, the performance is still quite satisfactory and much better than the other methods.

Usually, the default threshold 0.5 is not the optimal one for binarizing the localization maps. To discover the best performance, we adjusted the threshold from 0.01 to 0.99 with a step of 0.01 and plotted the curve of F1-scores ob-



Figure 5. F1-scores for synthetic inpainted images with QF=96 obtained by different thresholds.



Figure 7. F1-scores for random inpainted images with QF=96. Please refer to Figure 5 for the legend.

tained by different thresholds. The curves are shown in Figure 5 and 6. Please note that the results of the self-consistency method are represented as asterisks since the optimal threshold per image have been selected in this method. From the two figures we observe that the proposed method can steady keep good performance under a wide range of thresholds. Though the other methods can achieve better performance by adjusting the thresholds, they still underperform the proposed one even with their optimal thresholds.

4.4. Test on random/realistic inpainted images

In this experiment, we evaluated the localization performance for the random inpainted images as well as the realistic inpainted images. Firstly, we tested the models trained in the above experiment with the random inpainted images. In this case, the situation is more complicated, since an image may contain multiple inpainted regions, whose locations and sizes are randomly selected rather than fixed. We adjusted the thresholds and obtained the corresponding F1-scores of different methods as shown in Figure 7 and 8. It is observed that all the F1-scores are lower than the above experiment, and the proposed method still achieves



Figure 6. F1-scores for synthetic inpainted images with QF=75. Please refer to Figure 5 for the legend.



Figure 8. F1-scores for random inpainted images with QF=75. Please refer to Figure 5 for the legend.

the best performance. With the default threshold, the proposed method obtains the F1-scores of 0.9027 and 0.7604 for QF=96 and QF=75, respectively.

Lastly, we tested our trained models with the realistic inpainted images. The obtained F1-scores are 0.8405 and 0.7750 for images of QF=90 and QF=75, respectively. Some examples of the output localization maps are shown in Figure 9. It is observed that most of the inpainted regions can be correctly identified. However, it is also observed that the boundaries of some regions (especially the animalshaped regions in the 2nd row of subfigure (a)) are not clear enough. This indicates that the localization resolution of the proposed method is limited for dealing with sophisticated inpainted regions. It will be our future work to improve the performance for this case.

4.5. Robustness to noise

In practice, a forger may perform post-processing after inpainting in order to evade the detection. A common type of post attack is adding noises. In this experiment, we evaluated the robustness of the proposed method against additive noise. To this end, we added different levels of random Gaussian noises (SNR \in {50dB, 40dB, 30dB}) to the test-



Figure 9. Examples for localization of deep inpainting in realistic inpainted images. In each subfigure, left column: inpainted images, middle column: groundtruth, right column: localization maps obtained by the proposed method.

SNR	Recall	Precision	IoU	F1-score
50 dB 40 dB	89.78 89.33	95.21 95.23	85.88 85.48	91.49 91.18
30 dB	71.11	93.86	68.82	77.00

Table 4. Localization results (%) for noisy JPEG images.

ing images and JPEG compressed them with QF=75, and then we used the model trained with noise-free images to perform the detection. The results are reported in Table 4. It is observed that the performance is still good when SNR=50dB and 40dB. When SNR=30dB, the image quality is heavily degraded, so the performance drops a lot.

4.6. Generalization ability

In the previous experiments, we considered the inpainting method [16] as an example. In order to investigate how the proposed method perform for other inpainting methods, we used our method to train and test on QF=75 images inpainted by two other deep inpainting methods, which are proposed in [38] and [26]. The obtained results are shown in Table 5. From these results, we can conclude that our method can also achieve satisfactory performance for detecting other inpainting methods.

5. Conclusions

In this paper, we propose a solution to the localization of deep inpainting by using a high-pass fully convolutional network. Via analyzing the differences between inpainted and untouched regions, we have discovered that the differences can be enhanced by high-pass filtering. Motivated by this, the pre-filtering module is designed to suppress image contents and enhance inpainting traces. We

Inpainting method	Recall	Precision	IoU	F1-score
Iizuka <i>et al</i> . [16]	89.84	95.22	85.93	91.53
Yang et al. [38]	87.04	93.25	81.46	88.54
Liu et al. [26]	97.63	91.54	89.47	93.52

Table 5. Localization results (%) for different inpainting methods.

have constructed the feature extraction module based on ResNet blocks to learn features for distinguishing between inpainted regions and untouched regions. To achieve pixelwise prediction, we have designed the up-sampling module to enlarge the spatial resolution of learned feature maps. The whole network is trained with the focal loss for addressing class imbalance. The performance of the proposed method has been evaluated with both synthetic and realistic images. The experimental results show that the proposed method can effectively locate the inpainted regions and significantly outperform the existing methods.

In the future, we will further boost the performance by improving localization resolution. And we also plan to generalize the proposed method to locate other types of tampering regions, *e.g.*, the regions tampered with Photoshop in practical situations. On the other hand, we will investigate how to use the proposed method as a critic to guide the image inpainting methods based on GANs in creating more realistic contents.

Acknowledgement

This work was supported partially by the NSFC (61802262, U1636202), Guangdong Province R&D Plan in Key Areas (2019B010139003), and Shenzhen R&D Program (JCYJ20160328144421330).

References

- Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX* Symposium on Operating Systems Design and Implementation, pages 265–283, 2016.
- [2] Ketan S. Bacchuwar, Aakashdeep, and K. R. Ramakrishnan. A jump patch-block match algorithm for multiple forgery detection. In *Int. Multi-Conf. on Automation, Computing, Communication, Control and Compressed Sensing*, pages 723– 728, 2013.
- [3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Trans. on Graphics, 28(3):24:1–24:11, 2009.
- [4] Belhassen Bayar and Matthew C. Stamm. Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection. *IEEE Trans. on Information Forensics and Security*, 13(11):2691–2706, 2018.
- [5] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *SIGGRAPH*, pages 417–424, 2000.
- [6] Tony F. Chan and Jianhong Shen. Nontexture inpainting by curvature-driven diffusions. *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001.
- [7] I-Cheng Chang, J. Cloud Yu, and Chih-Chuan Chang. A forgery detection algorithm for exemplar-based inpainting images using multi-region relation. *Image and Vision Computing*, 31(1):57–71, 2013.
- [8] Vincent Christlein, Christian Riess, Johannes Jordan, Corinna Riess, and Elli Angelopoulou. An evaluation of popular copy-move forgery detection approaches. *IEEE Trans. on Information Forensics and Security*, 7(6):1841– 1854, 2012.
- [9] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. on Image Processing*, 13(9):1200– 1212, 2004.
- [10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.
- [11] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *ICCV*, volume 2, pages 1033– 1038, 1999.
- [12] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Int. Conf. on Artificial Intelligence and Statistics*, pages 249– 256, 2010.
- [13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pages 2672–2680, 2014.
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In ECCV, pages 630–645, 2016.

- [15] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A Efros. Fighting fake news: Image splice detection via learned self-consistency. In *ECCV*, pages 101–117, 2018.
- [16] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. ACM Trans. on Graphics, 36(4):107:1–107:14, 2017.
- [17] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- [18] Paweł Korus. Digital image integrity a survey of protection and verification techniques. *Digital Signal Processing*, 71:1– 26, 2017.
- [19] Anat Levin, Assaf Zomet, and Yair Weiss. Learning how to inpaint from global image statistics. In *ICCV*, pages 305– 312, 2003.
- [20] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. Detection of deep network generated images using disparities in color components. arXiv preprint arXiv: 1808.07276, 2018.
- [21] Haodong Li, Weiqi Luo, and Jiwu Huang. Localization of diffusion-based inpainting in digital images. *IEEE Trans.* on Information Forensics and Security, 12(12):3050–3064, 2017.
- [22] Haodong Li, Weiqi Luo, Xiaoqing Qiu, and Jiwu Huang. Identification of various image operations using residualbased features. *IEEE Trans. on Circuits and Systems for Video Technology*, 28(1):31–45, 2018.
- [23] Xiang Hua Li, Yu Qian Zhao, Miao Liao, Frank Y. Shih, and Yun Q. Shi. Detection of tampered region for JPEG images by using mode-based first digit features. *EURASIP Journal* on Avances in Signal Processing, 2012(1):1–10, 2012.
- [24] Zaoshan Liang, Gaobo Yang, Xiangling Ding, and Leida Li. An efficient forgery detection algorithm for object removal by exemplar-based image inpainting. *Journal of Visual Communication and Image Representation*, 30:75–85, 2015.
- [25] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017.
- [26] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, pages 85–100, 2018.
- [27] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.
- [28] Francesco Marra, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. Detection of GAN-generated fake images over social networks. In *IEEE Conf. on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389, 2018.
- [29] Huaxiao Mo, Bolin Chen, and Weiqi Luo. Fake faces identification via convolutional neural network. In ACM Workshop on Information Hiding and Multimedia Security, pages 43– 47, 2018.
- [30] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.
- [31] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In CVPR, pages 2536–2544, 2016.

- [32] Tijana Ružić and Aleksandra Pižurica. Context-aware patchbased image inpainting using Markov random field modeling. *IEEE Trans. on Image Processing*, 24(1):444–456, 2015.
- [33] Ronald Salloum, Yuzhuo Ren, and C.-C. Jay Kuo. Image splicing localization using a multi-task fully convolutional network (MFCN). *Journal of Visual Communication and Image Representation*, 51:201–209, 2018.
- [34] Matthew C. Stamm, Min Wu, and K. J. Ray Liu. Information forensics: An overview of the first decade. *IEEE Access*, 1:167–200, 2013.
- [35] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *CVPR*, pages 9446–9454, 2018.
- [36] Qiong Wu, Shao-Jie Sun, Wei Zhu, Guo-Hui Li, and Dan Tu. Detection of digital doctoring in exemplar-based inpainted images. In *Int. Conf. Machine Learning and Cybernetics*, volume 3, pages 1222–1226, 2008.
- [37] Wei Xiong, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. arXiv preprint arXiv: 1901.05945, 2019.
- [38] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multiscale neural patch synthesis. In *CVPR*, pages 6721–6729, 2017.
- [39] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Free-form image inpainting with gated convolution. arXiv preprint arXiv: 1806.03589, 2018.
- [40] Xinshan Zhu, Yongjun Qian, Xianfeng Zhao, Biao Sun, and Ya Sun. A deep learning approach to patch-based image inpainting forensics. *Signal Processing: Image Communication*, 67:90–99, 2018.