Information Sciences 572 (2021) 29–42

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Feature pyramid network for diffusion-based image inpainting detection

Yulan Zhang [a,b], Feng Ding [c], Sam Kwong [d,e], Guopu Zhu [a,*]

[a] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
[b] Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen, China
[c] The School of Software, Nanchang University, Nanchang, China
[d] Department of Computer Science, City University of Hong Kong, Hong Kong, China
[e] City University of Hong Kong, Shenzhen Research Institute, Shenzhen, China

ARTICLE INFO

Article history:
Received 27 October 2020
Received in revised form 6 February 2021
Accepted 3 April 2021
Available online 23 April 2021

Keyword:
Digital forensics
Image inpainting
Tampering detection
Feature pyramid network
Deep learning

ABSTRACT

Inpainting is a technique that can be employed to tamper with the content of images. In this paper, we propose a novel forensics analysis method for diffusion-based image inpainting based on a feature pyramid network (FPN). Our method features an improved u-shaped net to migrate FPN for multi-scale inpainting feature extraction. In addition, a stagewise weighted cross-entropy loss function is designed to take advantage of both the general loss and the weighted loss to improve the prediction rate of inpainted regions of all sizes. The experimental results demonstrate that the proposed method outperforms several state-of-the-art methods, especially when the size of the inpainted region is small.
© 2021 Elsevier Inc. All rights reserved.

1. Introduction

Image inpainting is one of the most important image processsing tools that can repair damaged or degraded regions in an image. Some renowned image manipulation tools, such as Photoshop and GIMP, have adopted image inpainting techniques as image processsing methods. Image inpainting methods can be divided into three classes, *i.e.*, diffusion-based techniques [1–3], patch-based techniques [4–6] and deep learning-based techniques [7–11]. The diffusion-based methods mainly focus on small-region inpainting, without leaving any perceptible artifacts. Patch-based methods are used to remove relatively large objects, which may lead to obvious inconsistencies in image context. Deep learning-based methods have developed rapidly in recent years [12–14] and can inpaint various sizes of regions and obtain good inpainting results with few artifacts. In [12], the authors proposed a de-occlusion distillation framework for face completion and masked face recognition. Ge et al. [13] proposed a method based on identity-diversity inpainting to facilitate occluded face recognition. The inpainting method was initially designed for reconstructing integral images from damaged or degraded images. However, some malevolent individuals exploit inpainting technology for malicious purposes. For instance, tampering can remove an object or a person in an image to falsify evidence in court. Others make use of inpainted images fabricating scenes to report false news stories. As a consequence, there is an urgent need to address the safety issues caused by image inpainting. Detecting whether an image is inpainted and locating the inpainted regions are of great importance for image forensics.

* Corresponding author.
E-mail address: gp.zhu@siat.ac.cn (G. Zhu).

https://doi.org/10.1016/j.ins.2021.04.042
0020-0255/© 2021 Elsevier Inc. All rights reserved.

In the past decades, image forensics has received much attention from researchers [15–20]. A variety of inpainting detection methods have been proposed. The authors of [21–24] took advantage of similar blocks within the queried image to detect patch-based image inpainting. The image pairs with large matching degrees were considered inpainted pairs. Zhu [25] et al. proposed a deep learning method for patch-based inpainting forensics for the first time in 2018. They applied a conventional neural network with encoder-decoder architecture to detect patch-based inpainting images of size 256×256. In addition, there is a growing body of literature that concentrates on deep learning-based inpainting forensics [26,17]. Wang et al. took advantage of Faster R-CNN for the forensics of AI inpainting. Faster R-CNN was utilized to capture the inconsistent features between the tampered region and the untouched region [26]. Recently, Li et al. presented a deep learning-based method to detect the pixels processed by AI inpainting [17]. A high-pass prefiltering module is added before the fully convolutional network to enhance the inpainting traces. The learned feature maps of the image residuals are more distinguishable than those of the original images.

However, little attention has been paid to diffusion-based inpainting forensics. To the best of our knowledge, there is only one work focusing on the forensics of diffusion-based inpainting to date [27]. To detect the diffusion-based region, the authors of [27] trained a classifier with the features extracted by calculating the local variance of the image Laplacian perpendicular to the direction of the image gradient. Finally, effective post-processing operations, such as exclusion of abnormally exposed regions and morphological filtering, were used to refine the initial results. Even though the method in [27] can detect diffusion-based inpainting, there remain several shortcomings to address. First, the conventional method can achieve relatively good results with a large inpainting size, but for a small inpainting size, the localization results are far from satisfactory. Second, not only does the conventional method require postprocessing steps to refine the localization map but also the trained classifier needs to set a threshold, which does not always work reliably. Furthermore, Li's method is degraded by certain post-operations, such as image scaling and JPEG compression. In this paper, we only focus on the diffusion-based inpainting detection and introduce a network based on feature pyramid network (FPN) to solve the above mentioned problem with the existing methods.

Since deep learning has been developed explosively in recent years, it has been widely applied in many image processing applications, such as image classification, image reconstruction, and image segmentation. Motivated by image segmentation, in this paper, an end-to-end deep learning method based on FPN is introduced to detect and locate diffusion-based inpainted regions in images. Fig. 1 shows an overview of the proposed framework.

The main contributions of this paper can be summarized as follows:

1. An end-to-end deep learning model focused on detection of diffusion-based inpainting is introduced. In the model, an improved u-shaped net (U-Net) is migrated to an FPN for multiscale inpainting feature extraction. To combine the information of features in different scales, the extracted features are strengthened by a convolution layer and then fused by a concatenating layer for classification. Our ablation study validates the effectiveness of feature fusion.
2. A stagewise weighted cross-entropy loss function is designed. The advantages of both the general loss and the weighted loss are integrated to optimize the training process. In addition, the detection results are refined with the loss function, especially when the inpainting size is small.
3. The experimental results show that our proposed method outperforms several state-of-the-art methods. The robustness of the proposed method against several image post-processing manipulations is also evaluated.

2. Related work

In this section, several deep learning-based methods of image segmentation are first introduced, since the image forgery detection problem is similar to image segmentation to some degree. Then, the feature pyramid network is described briefly.

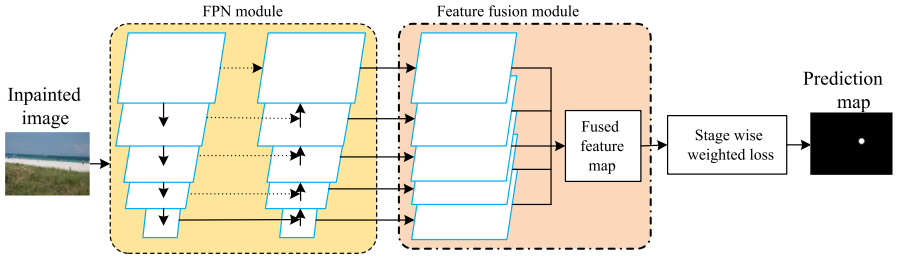

Fig. 1. Overall framework of our proposed method. An improved U-Net is embedded into the FPN module to extract the multiscale features for better detection of tampered regions of small size. Then, a feature fusion module is designed to take advantage of the correlations among the multilevel feature maps. Finally, a novel stagewise weighted loss function is designed to address the problem of convergence speed and the problem of the imbalance of samples.

2.1. Image segmentation

In recent decades, deep learning has shown explosive development in various computer vision tasks [28–32], such as semantic segmentation, object detection, and image classification. Li et al. proposed a novel learning framework based on semi-supervised non-negative matrix factorization, which can learn a robust discriminative representation for images [33]. For semantic segmentation, Long et al. [28] first proposed training an end-to-end fully convolutional network (FCN) for pixelwise prediction. The FCN can take input of an arbitrary size and produce a corresponding output with efficient inference and learning. A skip connection is defined to encode semantics by combining both deep, coarse semantic information and shallow, fine appearance information. The combined features can ensure a more precise localization output. Badrinarayanan et al. [30] presented a novel SegNet for semantic pixelwise segmentation, which consisted of an encoder-decoder network and a pixelwise classification layer. The manner of designing the decoder is changed by designing the encoder with the pooling indices computed in the max-pooling layer of the corresponding encoder. The SegNet is efficient in terms of both memory and computational time. Ronneberger et al. [29] proposed a network called U-Net based on an FCN, which used very few sample images for biomedical image segmentation. The U-Net is composed of a contracting path for capturing contextual information and a corresponding expanding path for precise localization. The author combined the features of both paths to localize the final map, leading to the seamless segmentation of arbitrarily large images. Seferbekov et al. [31] proposed a multiclass land segmentation method based on a fully convolutional neural network (CNN) of an FPN, which consisted of a pretrained Imagenet50 encoder and a neatly developed decoder. The network shows reliable results for the DEEPGLOBE-CVPR2018 land cover classification.

The objective of image inpainting detection is similar to that of image segmentation in some sense. Semantic segmentation aims to identify images at the pixel level, that is, to determine the object category to which each pixel in the image belongs. The goal of semantic segmentation is to predict the class label of each pixel in the image. In a similar way, the image inpainting detection problem aims to judge whether a pixel in the image is inpainted. Both problems output a localization map of the same size as the input image. As a consequence, the convolutional neural networks designed for semantic segmentation problem might be migrated to solve the detection problem of image inpainting. The networks for semantic segmentation extract visual objects by understanding the semantic content in different regions. Unlike semantic segmentation, inpainting detection focuses on the localization of the inpainted regions in images by analyzing inpainting artifacts. Inpainting artifacts are often visually imperceptible, which makes inpainting detection more challenging.

2.2. Feature pyramid network

Feature pyramid network can be employed in detecting objects of different sizes. In [34], Lin et al. constructed feature pyramids by taking advantage of the intrinsic multiscale and pyramidal hierarchy of deep CNNs. The FPN, which can build high-level semantic feature maps at all scales, is a top-down architecture combined with lateral connections. The FPN is constructed with a bottom-up pathway computing a feature hierarchy that consists of several feature maps, a top-down pathway hallucinating features of higher resolution, and a skip connector combining the feature maps from the bottom-up pathway with the coarser features of the top-down pathway of the same spatial size. The FPN has been adopted in Fast R-CNN for object detection, and the features of different scales are employed to make predictions independently. As a result, FPN has obtained relatively satisfactory results for object detection with different sizes. The success of FPN in object detection motivates the idea that FPN can also be applied to improve the performance of inpainting region detection.

3. Proposed method

In this section, we introduce an end-to-end deep learning-based FPN composed of an improved U-Net to detect and locate image inpainting forgery.

3.1. Improved deep U-Net

The traditional U-Net is composed of a contracting path and a symmetric expanding path. In our new framework, we improve the traditional U-Net for image inpainting forgery detection. Inspired by the idea in [35], a convolution layer is used to replace the max-pooling layer for down-sampling. The replacement can be seen as learning the pooling operations rather than fixing it. By adjusting the convolution stride, the convolution layer can learn the optimal down-sampling strategy required for network convergence. The experimental results demonstrate that the convolution layer can get better feature maps than the max-pooling layer. Therefore, a convolution layer with a kernel size of 4×4 and a stride of 2 is used to replace the 2×2 max-pooling layer for down-sampling. It is worth noting that the kernel size should be even instead of odd to ensure that the size of the feature is an integer. In addition, to accelerate convergence and prevent the network from overfitting, we add a batch normalization (BN) layer [36] following each convolution layer. The experimental results show that it is difficult for the network to converge without the batch normalization layer. It is worth mentioning that in our improved U-Net, the features in the corresponding symmetric layers are of the same size as those extracted from the contracting path. There is no need to crop the features as in the traditional U-Net to match the features in the expanding path. The improved

U-Net is exploited as the basic framework for feature extraction of image inpainting. The network architecture will be described briefly in Subsection 3.3.

3.2. Feature pyramid and feature fusion

The FPN architecture [34] consists of a bottom-up path, a top-down path and skip connections. The FPN makes predictions at all levels independently, leveraging the lateral connections as with a feature pyramid. Both the semantically strong features of low resolution and the semantically weak features of high resolution are exploited to make precise predictions. Low-level high-resolution features are important for detecting small regions. In this paper, the improved U-Net is embedded into the FPN to extract the inpainting feature with different resolutions for detecting and locating the inpainting forgery regions. For the purpose of detecting image inpainting, which is different from object recognition, there is no need to make predictions at all levels. As a result, we make use of feature fusion at different levels to judge the queried image. The feature fusion takes full advantage of the correlations of features among different scales so as to get more accurate detection results. In the ablation study, we demonstrate the effect of the fusion process on image inpainting detection.

3.3. Network architecture

The FPN-based architecture for image inpainting detection is shown in Fig. 2. The FPN is composed of the improved U-Net described in 3.1. The contracting path of the improved U-Net is made up of 3×3 convolution layers with stride 1, each followed by a BN layer and a rectified linear unit (ReLU). The channel of each feature is doubled after convolution because the number of kernels is doubled in each convolution. During the downsampling step, a 4×4 convolution layer with stride 2 is utilized instead of a max-pooling layer, keeping the number of feature channels unchanged. After four downsampling steps, a 3×3 convolution layer is used in the middle of the whole network. Then, in the expanding path, the feature is first upsampled by a 3×3 deconvolution layer with stride 2. Therefore, the feature size is doubled both in height and width, and the number of feature channels remains unchanged. The upsampling can lead to information distortion to some degree. To strengthen the accuracy of the feature, a lateral path is built between the contracting path and the expanding path. The features copied from the contracting path and upsampled from the bottom layer are concatenated. The concatenation can also be replaced by pixel-to-pixel addition to strengthen the representation ability of the high-level features. A 3×3 convolution layer is employed to halve the channels of the feature maps, followed by BN and a ReLU. The feature map from the last layer of the expanding path can be put directly into a 3×3 convolution layer to make pixelwise decisions. The experimental results are shown in Section 5.1. This strategy only takes the features from the last layer of the FPN into consideration and ignores the high-resolution features.

In [34], after the feature pyramid is extracted, a prediction is made at each level separately. In this paper, we fuse the feature maps of different resolution levels from the FPN and then perform the classification according to the fused features. Before fusion, the feature maps from the expanding path are first put into a convolution layer, and then upsampled by a deconvolution layer to ensure that the final upsampled features have the same size as the input image features. For convenience, the number of feature channels is set to 32, which is the same as the size of the output from the last layer of the FPN. An empirical 3×3 convolution layer is applied to produce a coarser resolution map. Then, the final feature maps at all levels are fused by concatenation. Finally, the fused feature is put into a 3×3 convolution layer to obtain the final feature map. The final feature map is of the same size as the input image, with two channels that correspond to the inpainting and untouched classes, respectively. The final feature map is expressed as $Z = [z_0, z_1]$, where z_0 and z_1 are the features corresponding to the untouched pixels and the inpainted pixels, respectively. Finally, the feature map is fed into the classification layer, which is made up of a softmax function. Then, the inpainting probability matrix

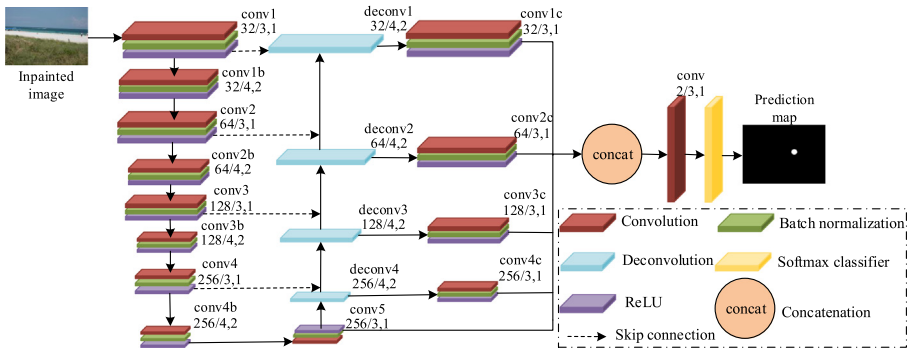

Fig. 2. FPN-based architecture for image inpainting detection. Numbers of the form $x/y, z$ near the layers refer to the number of filters, the kernel size, and the stride of the convolution layer (colored reddish brown) or deconvolution layer (colored light blue). A skip connection denotes that the feature from the left path is copied and concatenated with the upsampled feature from the right path.

$$P(z_m) = \frac{e^{z_m}}{e^{z_0} + e^{z_1}} \tag{1}$$

is acquired, where $m = 0, 1$ indicates the untouched class and the inpainted class, respectively. Then, according to the inpainting probability matrix P, the final estimated label matrix is obtained as

$$\hat{Y}_{ij} = \begin{cases} 1, P(z_1) \geqslant P(z_0) \\ 0, P(z_1) < P(z_0) \end{cases}, \tag{2}$$

where $1 \leqslant i \leqslant H, 1 \leqslant j \leqslant W$, and W and H are the width and height of the input image, respectively. P_{ij} denotes the probability of the pixel at the coordinate (i,j) being inpainted, and $\hat{Y}_{ij}$ denotes the estimated label at coordinate (i,j).

3.4. Loss function

The loss function is used to measure the performance of a network based on the predictions of the network and the ground truth labels. Due to the imbalance of our samples; *i.e.*, the tampered regions (positive samples) are much smaller than the untouched regions (negative samples) in an image, the standard cross-entropy loss does not perform well. The reason is that the predominant negative samples contribute the majority of the loss, and less information from the positive samples can be utilized, which inevitably results in a decrease in the true positive rate. In other words, the inpainted regions can not be identified accurately.

Considering the convergence speed and the imbalance between the positive and negative samples, we design a stagewise weighted cross-entropy loss function. Specifically, at the beginning of training, the general cross-entropy loss is exploited to accelerate convergence. The general loss achieves relatively better detection for large targets than the weighted loss. After preliminary training, the model can obtain a relatively high detection rate for large targets. As the training progresses, the impact of the weighted loss gradually increases to strengthen the detection of small regions. In summary, the designed feature learning method takes advantage of both the general loss and the weighted loss for inpainting detection. The two kinds of loss are integrated to optimize the overall training process, resulting in better detection for inpainted regions of all sizes.

Assume that the output vector of the model is represented as $Z = [z_0, z_1]$. Softmax is performed on the vector, and we obtain the inpainting probability matrix with Eq. (1). Then, the estimated label matrix can be obtained with Eq. (2). The ground-truth label matrix is Y. Because the class distributions of the samples are not the same, it is necessary to calculate the statistical weight of each sample independently. For an arbitrary sample, the softmax cross-entropy between the predicted data and the ground-truth matrix can be calculated as

$$\mathcal{L}(Y,Z) = -\sum_{j=1}^{W}\sum_{i=1}^{H} \widehat{Y}_{ij} \log P_{ij} + \left(1 - \widehat{Y}_{ij}\right) \log(1 - P_{ij}), \tag{3}$$

where $\widehat{Y}_{ij}$ represents the estimated label at coordinate (i,j) and P_{ij} represents the probability of being inpainted at coordinate (i,j). Considering the imbalance of the training samples, the weights $W = \left[\mu_{ij}, v_{ij}\right]$ are introduced to prevent the model from overfitting for the majority class of samples. The weighted loss function can be written as

$$\mathcal{L}_w(Y,Z) = -\sum_{j=1}^{W}\sum_{i=1}^{H} \mu_{ij} \widehat{Y}_{ij} \log P_{ij} + v_{ij}\left(1 - \widehat{Y}_{ij}\right) \log(1 - P_{ij}), \tag{4}$$

where $\mu_{ij} + v_{ij} = 1, \mu_{ij} = |\bar{\Omega}_k|/(W \times H), v_{ij} = |\Omega_k|/(W \times H)$, and $|\Omega_k|$ and $|\bar{\Omega}_k|$ denote the inpainted region and the untouched region of an image, respectively.

During our experiments, we found that the general cross-entropy loss function plays a critical role in accelerating the convergence speed and detecting forged objects of large size. The weighted loss can strengthen the detection of small inpainted regions. In this paper, a stagewise weighted loss function is designed to guide the training; *i.e.*, at the beginning of training, the general loss plays a more critical role, and it can accelerate the convergence and locate inpainted regions of large size. As the training progresses, the effect of the general loss becomes weaker, while the effect of the weight becomes stronger, which can result in good detection of the inpainted regions of small size. Assuming T is the total number of iterations of training, the loss of the t-th ($t \leqslant T$) iteration can be given as

$$\mathcal{L}_p(Y,Z,t) = \left(1 - \frac{t}{T}\right)\mathcal{L}(Y,Z) + \frac{t}{T}\mathcal{L}_w(Y,Z), \tag{5}$$

where T is mainly determined by the number of epoch n, the number of training samples N and the batch size S_b. Further, the maximum number of iterations can be calculated as: $T = Floor(N/S_b) \times n$, where $Floor(x)$ denotes the maximum integer that is less or equal than x.

The general loss plays a critical role at the beginning of training. As the number of iterations increases, the weighted loss is more important in training. The stagewise weighted loss can obtain relatively reliable results for inpainting of all sizes.

4. Experimental setup

Our introduced network is trained on Ubuntu 16.04 with Tesla v100 32G GPU and 128 GB PC memory.

4.1. Data Preparation

First, we implement our experiments on the detection of the diffusion-based inpainting without post-processing. 40,000 images randomly selected from the COCO dataset [37] and center-cropped to 384×512 are used to generate diffusion-based inpainting samples with the method provided by G'MIC [38], an open-source framework for image manipulation. The diffusion-based image inpainting tools in G'MIC include three inpainting algorithms, *i.e.*, isotropic, Delaunay-oriented and edge-oriented inpainting, which all use the basic diffusion-based concepts proposed in [1]. The inpainting sizes include 64 pixels, 256 pixels, 1024 pixels, and 4096 pixels. The inpainted regions have three different shapes: square, circular, and irregular. In total, 360,000 inpainted images are generated to train the proposed model; *i.e.*, 10,000 images are used for each inpainting method, shape and size on average. Then, the UCID database [39] is used to generate the test samples with the same method as mentioned previously. We obtained 48,168 images for testing. The metrics introduced in 4.4 are exploited to evaluate the detection performance.

For inpainting post-processed by JPEG compression and scaling, the model is trained with the original inpainted images and the post-processed (including JPEG compression and scaling) inpainted images, respectively. To validate the robustness of the proposed method against JPEG compression, the inpainted images generated from the COCO dataset are randomly compressed with a quality factor $QF \in \{60, 75, 90, 100\}$. Then, a model is trained with the JPEG compressed images. Similarly, to validate the robustness against image scaling, the inpainted images are randomly scaled with scaling factor $SF \in \{0.5, 0.75, 1.5, 2\}$. Then, a model is trained with the scaled images. In testing phase, the inpainted images generated from the UCID dataset are randomly compressed with $QF \in \{60, 75, 90, 100\}$ and then tested with the model trained on compressed images. The inpainted images generated from the UCID dataset are randomly scaled with $SF \in \{0.5, 0.75, 1.5, 2\}$ and then tested with the model trained on scaled images. The results are compared with those obtained using the model trained with the original inpainted images. For simplicity, we evaluate the robustness against Gasussian filtering and image sharpening only using the model trained on inpainted images without post-processing. To validate the robustness against Gaussian filtering, the inpainted images generated from the UCID dataset are randomly filtered by a Gaussian filter with standard deviation $\sigma \in \{0.5, 1, 1.5, 2\}$. Similarly, to validate the robustness against image sharpening, the inpainted images generated from the UCID dataset are randomly sharpened with strength parameter $\lambda \in \{0.5, 1, 1.5, 2\}$.

4.2. Implementation details

We implement the FPN-based network with the Pytorch deep learning framework [40]. The Adam optimizer [41] is applied to optimize the model, and the learning rate is initialized as 1×10^{-3}. The learning rate decayed by 0.1 after 16 epochs. The batch size is set to 32, and the whole network is trained for 60 epochs. The training takes about 1.5 h for each epoch with our Tesla v100 32G GPU. All the trained models are saved, and then the test data are tested on all the trained models. The model with the highest F1-score on test data is the model we want to obtain.

4.3. Comparative methods

In this paper, we compare our proposed method with three previous methods. The first method is a traditional method for detecting diffusion-based inpainting [27]. The second is an AI inpainting detection method based on Faster R-CNN [26]. The last method is the high-pass fully convolutional network recently proposed in [17] for AI inpainting. The latter two methods are retrained on our diffusion-based inpainting database.

4.4. Performance metrics

The objective of our experiment is to detect and localize the diffusion-based inpainted regions of a queried image. The performance is measured by the F1-score and intersection over union (IoU). The F1-score is calculated as follows:

$$\mathrm{F1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}, \tag{6}$$

where $\mathrm{precision} = \frac{TP}{TP+FP}$, $\mathrm{recall} = \frac{TP}{TP+FN}$, TP is the number of detected inpainted pixels, FP is the number of wrongly detected untouched pixels, and FN is the number of undetected inpainted pixels. The F1-score is the harmonic average of precision and recall, with a maximum value of 1 and a minimum of 0. A larger F1-score represents a better detection result.

The IoU is used to evaluate the objective detector by calculating the ratio of the intersection (the overlapping region of the prediction box and the ground truth) to the union (the union region of the prediction box and the ground truth), which can be obtained by Eq. (7), a larger IoU corresponds to a better detection result.

$$\mathrm{IoU} = \frac{TP}{TP + FP + FN}. \tag{7}$$

5. Experimental results

In this section, we test diffusion-based inpainting on our trained model. Then, the robustness against several post-processing operations on inpainting detection is analyzed.

5.1. The detection of inpainting without post-processings

We first test our trained network on the original inpainted images without post-processing. Considering the manner of combining the features in the contracting path and expanding path of the improved U-Net, concatenation or pixel-to-pixel addition can be used to strengthen the upsampled features. In addition, the features from the last layer of the U-Net can be directly utilized to detect the inpainted regions. The impacts of the feature combining method and of fusion on detection are shown in Table 1.

It can be seen from the data in Table 1 that feature concatenation on the contracting path and expanding path can obtain a higher detection rate than pixel-to-pixel addition. In addition, for the feature fusion method, "No fusion" indicates that the features of the last layer of the U-Net are directly exploited for detecting the inpainted region. "Concatenation" indicates that the feature maps at different scales are fused by concatenation. The experimental results verified that feature fusion promotes detection, and this may be because fusion can capture the information on the correlations among features at different resolutions.

Table 2 shows the inpainting detection results with several comparative methods for three diffusion-based inpainting methods and different inpainting sizes. The "Average" row denotes the average prediction results for all sizes with each inpainting method. The "All-average" row denotes the overall average detection result for each detection method. The data in Table 2 show that our proposed method achieves an average of 0.3526/0.4187, 0.1660/0.2240, and 0.0538/0.0935 higher in F1-score and IoU than the methods in [27,26,17], respectively. The larger the inpainting size is, the higher the F1-score/IoU that our proposed method can achieve. Although the three compared methods can detect most of the inpainted regions of large inpainting size, the detection results degraded greatly for small inpainting sizes such as 8×8. For the 64×64 inpainting size, the proposed method achieves an average F1-score/IoU of 0.9967/0.9959, which is 0.1272/0.1933, 0.0541/0.0935, and 0.0411/0.0722 higher than [27,26,17], respectively. For the 8×8 inpainting size, the proposed method achieves an average F1-score/IoU of 0.8929/0.8670, which is 0.6212/0.6467, 0.3958/0.4547, and 0.0642/0.1134 higher than those of the methods in [27,26,17], respectively. It is obvious that our proposed method can obtain a significant improvement in detection inpainting of a small size.

5.2. Robustness analysis

Inpainted images may be post-processed by other manipulations, such as JPEG compression, image scaling, Gaussian filtering and image sharpening. In this subsection, the effect of some image manipulations on the proposed method is investigated. JPEG compression, image scaling, Gaussian filtering and image sharpening are performed on the test inpainted images, respectively. The detection results on the post-processed inpainted images are compared with those on the inpainted images without post-processing.

As seen from Table 3, the model trained with the inpainted images without post-processing obtains an average F1-score/IoU of 0.2856/0.2640, while the model trained on the JPEG-compressed inpainted images obtains an average F1-score/IoU of 0.6979/0.6722. For the model trained on the inpainted images without post-processing, as QF decreases (which means that images undergo heavier and heavier compression), the detection rates on the JPEG-compressed inpainted images degrade dramatically compared with these on the inpainted images without JPEG compression. Besides, compared with the model trained on the inpainted images without post-processing, the model trained on the inpainted images with JPEG compression has a significantly improvement in average detection rate, especially for smaller QFs.

Table 4 shows that the model trained on the scaled inpainted images obtains a slightly higher average F1-score and IoU than the model trained on the inpainted images without scaling, with an average F1-score/IoU of 0.8704/0.8416 for the for-

Table 1
Impact of the feature combining method and the FPN on the inpainting detection results; the best results are highlighted in bold.

Feature combining method	Concatenation	✓	✓		
	Pixel-to-pixel addition			✓	✓
Feature fusion method	No fusion	✓		✓	
	Concatenation		✓		✓
Average F1-score		0.9542	**0.9653**	0.9488	0.9641
Average IoU		0.9422	**0.9563**	0.9283	0.9532

Table 2
Inpainting detection results (F1-score/IoU) on the model trained with inpainted images without post-processing, for the comparative methods and our proposed method; the best results are highlighted in bold.

Methods	Size	Isotropic			Delaunay-oriented			Edge-oriented		
		square	circular	irregular	square	circular	irregular	square	circular	irregular
Traditional [27]	64^2	0.8957/ 0.8454	0.8909/ 0.8376	0.8806/ 0.8200	0.8173/ 0.7188	0.8657/ 0.7952	0.8354/ 0.7466	0.8876/ 0.8325	0.8826/ 0.8245	0.8702/ 0.8035
	32^2	0.8041/ 0.7356	0.8007/ 0.7281	0.7794/ 0.6932	0.6250/ 0.5000	0.7035/ 0.5958	0.6841/ 0.5660	0.7947/ 0.7206	0.7978/ 0.7120	0.7606/ 0.6674
	16^2	0.6793/ 0.5940	0.6802/ 0.5957	0.6483/ 0.5468	0.3247/ 0.2434	0.3466/ 0.2666	0.4085/ 0.3131	0.6659/ 0.5767	0.6637/ 0.5743	0.6200/ 0.5154
	8^2	0.4328/ 0.3572	0.5269/ 0.4457	0.2017/ 0.1561	0.1044/ 0.0791	0.1385/ 0.1077	0.0914/ 0.0682	0.3350/ 0.2704	0.4364/ 0.3625	0.1776/ 0.1361
	Average	0.7030/ 0.6331	0.7247/ 0.6518	0.6275/ 0.5540	0.4679/ 0.3853	0.5136/ 0.4413	0.5049/ 0.4235	0.6708/ 0.6001	0.6951/ 0.6183	0.6071/ 0.5306
	All-average					0.6127/0.5376				
Faster-RCNN [26]	64^2	0.9418/ 0.8943	0.9604/ 0.9309	0.9299/ 0.8861	0.9382/ 0.8920	0.9572/ 0.9262	0.9257/ 0.8818	0.9418/ 0.8952	0.9595/ 0.9306	0.9292/ 0.8851
	32^2	0.9508/ 0.9126	0.9435/ 0.8977	0.9163/ 0.8498	0.9382/ 0.8920	0.9195/ 0.8568	0.8917/ 0.8107	0.9456/ 0.9071	0.9394/ 0.8911	0.9067/ 0.8361
	16^2	0.8448/ 0.7413	0.9135/ 0.8617	0.8463/ 0.7562	0.7935/ 0.6918	0.8171/ 0.7347	0.6953/ 0.5894	0.8421/ 0.7394	0.9002/ 0.8423	0.8095/ 0.7127
	8^2	0.7541/ 0.6630	0.6884/ 0.5433	0.4717/ 0.3909	0.2920/ 0.2523	0.4091/ 0.3191	0.1887/ 0.1571	0.6608/ 0.5775	0.6560/ 0.5157	0.3528/ 0.2920
	Average	0.8729/ 0.8028	0.8765/ 0.8084	0.7911/ 0.7208	0.7411/ 0.6835	0.7757/ 0.7092	0.6754/ 0.6098	0.8476/ 0.7798	0.8638/ 0.7949	0.7496/ 0.6815
	All-average					0.7993/0.7323				
HPF [17]	64^2	0.9815/ 0.9693	0.9716/ 0.9519	0.9342/ 0.8858	0.9732/ 0.9538	0.9655/ 0.9405	0.9183/ 0.8584	0.9742/ 0.9572	0.9635/ 0.9373	0.9185/ 0.8598
	32^2	0.9705/ 0.9559	0.9616/ 0.9344	0.9174/ 0.8552	0.9581/ 0.9334	0.9496/ 0.9125	0.9002/ 0.8269	0.9663/ 0.9475	0.9532/ 0.9194	0.9031/ 0.8336
	16^2	0.9628/ 0.9444	0.9432/ 0.9095	0.8780/ 0.7997	0.9506/ 0.9244	0.9343/ 0.8952	0.8503/ 0.7622	0.9552/ 0.9313	0.9368/ 0.8995	0.8637/ 0.7793
	8^2	0.8758/ 0.8081	0.9106/ 0.8582	0.8170/ 0.7193	0.8179/ 0.7451	0.8811/ 0.8174	0.6548/ 0.5628	0.8567/ 0.7862	0.8877/ 0.8271	0.7564/ 0.6587
	Average	0.9477/ 0.9194	0.9467/ 0.9135	0.8867/ 0.8150	0.9250/ 0.8892	0.9326/ 0.8914	0.8309/ 0.7526	0.9381/ 0.9055	0.9353/ 0.8958	0.8604/ 0.7829
	All-average					0.9115/0.8628				
Proposed	64^2	0.9977/ 0.9975	0.9960/ 0.9958	0.9985/ 0.9977	0.9974/ 0.9970	0.9958/ 0.9952	0.9965/ 0.9946	0.9972/ 0.9968	0.9958/ 0.9953	0.9956/ 0.9935
	32^2	0.9949/ 0.9942	0.9948/ 0.9940	0.9967/ 0.9942	0.9925/ 0.9915	0.9942/ 0.9930	0.9936/ 0.9883	0.9923/ 0.9912	0.9943/ 0.9926	0.9920/ 0.9868
	16^2	0.9891/ 0.9872	0.9823/ 0.9797	0.9769/ 0.9675	0.9841/ 0.9805	0.9779/ 0.9730	0.9620/ 0.9433	0.9851/ 0.9812	0.9769/ 0.9710	0.9654/ 0.9493
	8^2	0.9441/ 0.9327	0.9670/ 0.9591	0.8941/ 0.8581	0.9036/ 0.8838	0.9460/ 0.9336	0.7336/ 0.6782	0.9164/ 0.8970	0.9283/ 0.9071	0.8027/ 0.7535
	Average	**0.9815/ 0.9779**	**0.9850/ 0.9822**	**0.9666/ 0.9544**	**0.9694/ 0.9632**	**0.9785/ 0.9737**	**0.9214/ 0.9011**	**0.9728/ 0.9666**	**0.9738/ 0.9665**	**0.9389/ 0.9208**
	All-average					**0.9653/0.9563**				

mer and an average F1-score/IoU of 0.8315/0.8099 for the latter. When the SF is 0.5 or 0.75, the model trained with the scaled inpainted images obtains a higher detection rate; when the SF is 1.5 or 2.0, the detection results of the model trained with the scaled inpainted images degrade to some degree. The reason may be that when the SF is less than 1, the resolution is reduced when the images are scaled down. The model trained with the scaled images can learn more features of low-resolution inpainted images. However, when the SF is larger than 1, the images are enlarged, and the features of the inpainted region are degraded at the same time. For this reason, the detection rate of the model trained with the scaled inpainted images degrade when the SF is larger than 1.

For Gaussian filtering, the size of the filter is set to 5×5, and the standard deviation σ of the filter is set from 0.5 to 2 with a step of 0.5. The detection results of inpainting detection are shown in Table 5.

From Table 5, it can seen that when the standard deviation of Gaussian filter is 0.5, the detection results degrades a little. As the standard deviation of Gaussian filter increases, the detection results degrades dramatically, especially for small inpainting sizes, such as 16×16 and 8×8.

Table 3
Effects of JPEG compression on the performance (F1-score/IoU) of the proposed method; the best results are highlighted in bold.

Method	Size	QF			
		100	90	75	60
Model trained using inpainted images without post-processing	64^2	0.8765/0.8470	0.5075/0.4409	0.1562/0.1247	0.0624/0.0471
	32^2	0.8782/0.8561	0.2728/0.2360	0.0824/0.0677	0.0387/0.0308
	16^2	0.7830/0.7537	0.1392/0.1139	0.0453/0.0341	0.0279/0.0209
	8^2	0.6064/0.5722	0.0638/0.0550	0.0186/0.0148	0.0108/0.0084
	Average	**0.7860**/0.7573	0.2458/0.2114	0.0756/0.0603	0.0350/0.0268
	All-average	0.2856/0.2640			
Model trained using inpainted images with JPEG compression	64^2	0.9903/0.9835	0.9847/0.9755	0.9671/0.9535	0.9468/0.9304
	32^2	0.9645/0.9454	0.9349/0.9108	0.8732/0.8452	0.8407/0.8088
	16^2	0.8106/0.7685	0.7236/0.6814	0.6307/0.5867	0.5611/0.5166
	8^2	0.3639/0.3333	0.2725/0.2476	0.1785/0.1597	0.1241/0.1090
	Average	0.7823/**0.7577**	**0.7289/0.7038**	**0.6624/0.6363**	**0.6182/0.5912**
	All-average	**0.6979/0.6722**			

Table 4
Effects of image scaling on the performance (F1-score/IoU) of the proposed method; the best results are highlighted in bold.

Method	Size	SF			
		0.5	0.75	1.5	2.0
Model trained using inpainted images without post-processing	64^2	0.9439/0.9201	0.9902/0.9835	0.9960/0.9946	0.9957/0.9940
	32^2	0.8782/0.8301	0.9809/0.9677	0.9926/0.9892	0.9919/0.9878
	16^2	0.5214/0.4703	0.9148/0.8750	0.9726/0.9604	0.9685/0.9540
	8^2	0.0600/0.0501	0.4850/0.4337	0.8298/0.7936	0.7937/0.7548
	Average	0.5981/0.5676	0.8427/0.8150	**0.9477/0.9345**	**0.9375/0.9226**
	All-average	0.8315/0.8099			
Model trained using inpainted images with image scaling	64^2	0.9900/0.9829	0.9936/0.9300	0.9948/0.9919	0.9946/0.9916
	32^2	0.9722/0.9531	0.9847/0.9738	0.9875/0.9793	0.9872/0.9788
	16^2	0.8600/0.8071	0.9363/0.8996	0.9510/0.9223	0.9494/0.9199
	8^2	0.2686/0.2375	0.6049/0.5526	0.7346/0.6812	0.7169/0.6639
	Average	**0.7727/0.7452**	**0.8799/0.8390**	0.9170/0.8937	0.9120/0.8885
	All-average	**0.8704/0.8416**			

Table 5
Effects of Gaussian filtering on the performance (F1-score/IoU) of the proposed method; the best results are highlighted in bold.

Size	σ			
	0.5	1.0	1.5	2.0
64^2	**0.9962/0.9949**	0.9695/0.9557	0.8632/0.8324	0.8227/0.7877
32^2	**0.9925/0.9890**	0.9011/0.8748	0.7025/0.6616	0.6456/0.6017
16^2	**0.9694/0.9567**	0.6590/0.6111	0.3038/0.2646	0.2423/0.2063
8^2	**0.8010/0.7680**	0.1418/0.1222	0.0085/0.0067	0.0052/0.0039
Average	**0.9398/0.9272**	0.6678/0.6410	0.4695/0.4413	0.4289/0.3999
All-average	0.6265/0.6024			

For image sharpening, the parameter G_δ and λ are used to control the range and strength of image sharpening, respectively. As λ increases, the strength of sharpening gets stronger. In our experiments, G_δ is fixed with 0.5, and λ is set from 0.5 to 2 with a step of 0.5. Then the effect of image sharpening on our proposed method is investigated. The detection results are shown in Table 6.

Interestingly, it can be seen from Table 6 that the detection results are improved after image sharpening, especially for larger strength parameters, *i.e.*, for the case of $\lambda = 1.5$ and $\lambda = 2$. We get an average F1-score/IoU of 0.9704/0.9617 after image sharpening, which has an increase of 0.0051/0.0054 in F1-score/IoU compared with the detection results on the inpainted images without image sharpening.

According to the experimental results on different image post-processing, we see that the proposed method is robust against the image manipulations that do not cause serious degradation on image content, such as image scaling and image sharpening; whereas, it is less robust against the image manipulations that can lead to serious degradation of image quality, such as JPEG compression and Gaussian filtering. This may be explained as follows. Firstly, for JPEG compression and Gaus-

Table 6
Effects of image sharpening on the performance (F1-score/IoU) of the proposed method; the best results are highlighted in bold.

Size	λ			
	0.5	1.0	1.5	2.0
64^2	0.9967/0.9959	0.9966/0.9957	0.9966/0.9959	**0.9968**/**0.9959**
32^2	0.9939/0.9917	0.9938/0.9915	0.9943/0.9925	**0.9946**/**0.9926**
16^2	0.9771/0.9696	0.9775/0.9722	0.9833/0.9775	**0.9842**/**0.9802**
8^2	0.8939/0.8674	0.8976/0.8702	**0.9272**/**0.9030**	0.9227/0.8971
Average	0.9654/0.9561	0.9664/0.9569	**0.9754**/**0.9672**	0.9746/0.9664
All-average		0.9704/0.9617		

sian filter, the degradation of image quality results in serious loss of inpainting traces, which results in a decrease in detection results. Secondly, for image scaling, although the size of images is changed, the content of images changes little. Therefore, the traces of inpainting are maintained mostly after image scaling. Thirdly, image sharpening strengthens image content in high frequencies. Whereas, diffusion-based inpainting leads to smoothness in image content, so the traces of inpainting region are changed little after image sharpening. Hence, image sharpening can enlarge the difference between the inpainted regions and the untouched regions. In a certain sense, image sharpening may be used to strengthen the inpainting feature for improving the detection results.

5.3. Effects of the stagewise weighted loss function

The effects of different loss functions on forgery detection will be discussed in this section. The network is trained with the general cross-entropy loss, the weighted loss function and the proposed stagewise weighted loss function. The results are shown in Table 7. It is apparent that the general loss can obtain a relatively high detection rate for inpainted regions of large sizes, such as 32×32 and 64×64. The weighted loss can strengthen the ability to detect inpainted regions with small sizes of 8×8 and 16×16. The stagewise weighted loss obtains the best prediction results for both the large and small inpainted regions. The new stagewise weighted loss achieves a 0.0144/0.0237 and 0.0044/0.0088 higher F1-score/IoU than the general cross-entropy loss and the weighted loss, respectively. The stagewise weighted loss makes full use of the general loss and the weighted loss in the whole training process; *i.e.*, at the beginning of training, the general loss plays a greater role than the weighted loss in promoting convergence and locating the inpainted regions of large size, while at the end of training, the weighted loss is more important than the general loss in strengthening the ability to locate the inpainted regions of small size. Overall, the advantages of the general loss and the weighted loss are combined in feature learning to optimize the overall training process.

5.4. Performance on realistic inpainting

In this subsection, we will investigate the proposed method with several realistic inpainted images with different inpainting shape, such as circular, retangular and irregular shape. The localization maps of the comparative methods and the proposed method are shown in Fig. 3. The proposed method outperforms all the comparative methods on almost all the tested images, and the best results are marked in bold. Compared with the ground truth in the second row, the proposed FPN-based method can detect most of the inpainted regions well. Especially, from the fourth column in Fig. 3, it can be seen that the proposed method can also detect the inpainted region in a motion blurred inpainted image all the same.

5.5. Performance on other types of inpainting

The proposed method is designed to detect diffusion-based inpainting. We also implemented experiments on detection of classical patch-based inpainting [4] and deep learning-based inpainting [9]. For patch-based inpainting, the detection model is trained on 20,000 images randomly selected from the COCO dataset inpainted with patch-based inpainting method [4] and then evaluated on 1,179 inpainting images generated from VOC dataset with the same inpainting method. For deep learning-based inpainting, in a similar way, the model is trained on 350,000 images selected from the COCO dataset inpainted with

Table 7
Inpainting detection results (F1-score/IoU) with different loss functions on inpainted images without post-processing; the best results are highlighted in bold.

Loss function type	Size				
	8^2	16^2	32^2	64^2	Average
General cross-entropy loss	0.8590/0.8152	0.9614/0.9418	0.9884/0.9812	0.9948/0.9923	0.9509/0.9326
Weighted cross-entropy loss	0.8903/0.8604	0.9760/0.9664	0.9844/0.9741	0.9929/0.9888	0.9609/0.9475
Stagewise weighted cross-entropy loss	**0.8929**/**0.8670**	**0.9777**/**0.9703**	**0.9939**/**0.9918**	**0.9967**/**0.9959**	**0.9653**/**0.9563**

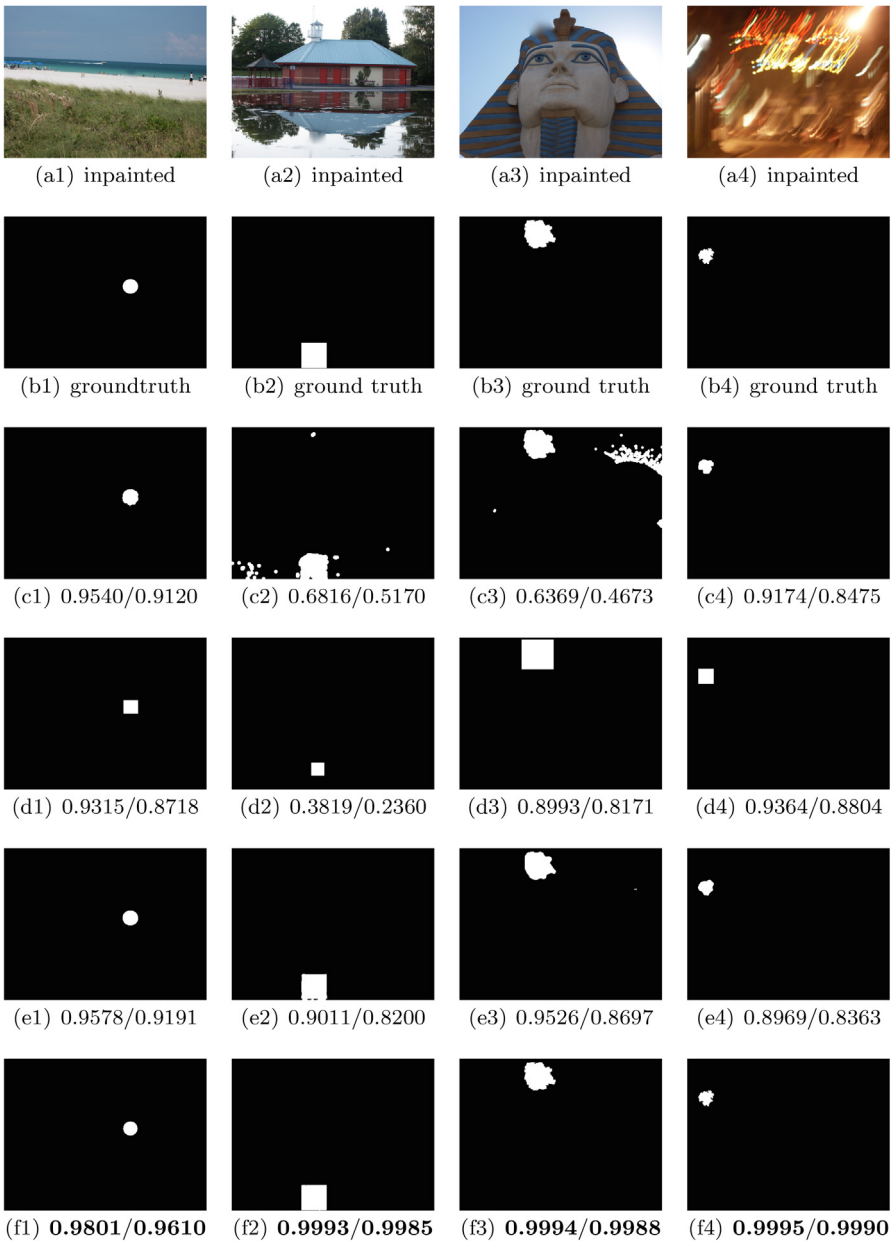

Fig. 3. Localization results for several comparative mehods and the proposed method (F1-score/IoU). The first row shows the diffusion-based inpainting images, the second row shows the corresponding ground truth images, and the third to sixth rows show the detection results obtained with traditional method [27], faster RCNN [26], high-pass FCN [17] and the proposed method, respectively.

deep learning based inpainting method [9] and then evaluated on 3,705 inpainting images generated from VOC dataset with the same inpainting method.

Experimental results show that the proposed method can detect the patch-based inpainting well. The average F1-score and IOU reach 0.9811 and 0.9642, respectively. While for the deep learning-based inpainting, our method can only achieve an average F1-score and IOU of 0.8474 and 0.7660, respectively. Fig. 4 shows two localization maps of the proposed method on patch-based inpainting method [4] and deep learning-based inpainting method [9], respectively.

The good performance in detecting the patch-based inpainting may be due to that the patch-based inpainting leaves visible inconsistencies in image context, and our proposed network can characterize such inpainting traces well. Whereas, the deep learning-based inpainting can obtain good inpainting results without leaving obvious artifacts. As a result, the detection of deep learning-based inpainting with the proposed method is not satisfactory. Thus, it is a more challenging task to

Fig. 4. Localization results for patch-based inpainting and deep learning-based method of the proposed method (F1-score/IoU). The first and second column show the original images and the corresponding groundtruth, respectively. Figs. (a3) and (b3) are the inpainted images with the patch-based method [4]. Figs. (a4) and (b4) show the corresponding detection results of (a3) and (b3) obtained with the proposed method, respectively. Figs. (c3) and (d3) are the corresponding inpainted images with the deep learning-based method [9]. Figs. (c4) and (d4) are the corresponding localization results of (c3) and (d3) obtained with the proposed method, respectively.

design a detection method for deep learning-based inpainting. Moreover, it is also a very challenging task to propose a general forensic method for different kinds of image inpainting operations.

6. Conclusion

In this paper, a forensic method based on a feature pyramid network has been proposed for the detection of diffusion-based image inpainting. A feature fusion strategy and a stagewise weighted cross-entropy loss function are combined to improve the performance of localizing the inpainted regions of different sizes. Extensive experimental results have verified that the proposed method outperforms several state-of-the-art methods in terms of both F1-score and IoU. Our future work will be devoted to generalizing the proposed method to locate other types of forgery, such as image splicing and copy-move.

CRediT authorship contribution statement

Yulan Zhang: Writing - original draft, Methodology, Software. **Feng Ding:** Writing - review & editing, Software. **Sam Kwong:** Writing - review & editing, Supervision. **Guopu Zhu:** Resources, Writing - review & editing, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors thank the anonymous reviewers for their valuable comments. This work was supported in part by the National Natural Science Foundation of China under Grant 61872350, Grant 61572489, and Grant 61672443, in part by Hong Kong GRF-RGC General Research Fund under Grant 9042816 (CityU 11209819) and Grant 9042958 (CityU 11203820), in part by the Tip-top Scientific and Technical Innovative Youth Talents of Guangdong Special Support Program under Grant 2019TQ05X696, and in part by the Basic Research Program of Shenzhen under Grant JCYJ20170818163403748.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, Image inpainting, in: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424..
[2] M.M.O.B.B. Richard, M.Y.-S. Chang, Fast digital image inpainting, in: Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP), 2001, pp. 106–107.
[3] A. Levin, A. Zomet, Y. Weiss, Learning how to inpaint from global image statistics, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), IEEE, 2003, pp. 305–312.
[4] A. Criminisi, P. Pérez, K. Toyama, Region filling and object removal by exemplar-based image inpainting, IEEE Trans. Image Process. 13 (9) (2004) 1200–1212.
[5] C. Barnes, E. Shechtman, A. Finkelstein, D.B. Goldman, PatchMatch: A randomized correspondence algorithm for structural image editing, ACM Trans. Graph. 28 (3) (2009) 24.
[6] T. Ružić, A. Pižurica, Context-aware patch-based image inpainting using Markov random field modeling, IEEE Trans. Image Process. 24 (1) (2014) 444–456.
[7] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A.A. Efros, Context encoders: Feature learning by inpainting, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2536–2544.
[8] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, H. Li, High-resolution image inpainting using multi-scale neural patch synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6721–6729.
[9] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, T.S. Huang, Generative image inpainting with contextual attention, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 5505–5514.
[10] G. Liu, F.A. Reda, K.J. Shih, T.-C. Wang, A. Tao, B. Catanzaro, Image inpainting for irregular holes using partial convolutions, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 85–100.
[11] D. Ulyanov, A. Vedaldi, V. Lempitsky, Deep image prior, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 9446–9454.
[12] C. Li, S. Ge, D. Zhang, J. Li, Look through masks: towards masked face recognition with de-occlusion distillation, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 3016–3024.
[13] S. Ge, C. Li, S. Zhao, D. Zeng, Occluded face recognition in the wild by identity-diversity inpainting, IEEE Trans. Circ. Syst. Video Technol. 30 (10) (2020) 3387–3397.
[14] Y.-G. Shin, M.-C. Sagong, Y.-J. Yeo, S.-W. Kim, S.-J. Ko, Pepsi++: Fast and lightweight network for image inpainting, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2021) 252–265.
[15] J. Yang, J. Xie, G. Zhu, S. Kwong, Y.-Q. Shi, An effective method for detecting double JPEG compression with the same quantization matrix, IEEE Trans. Inf. Forensics Secur. 9 (11) (2014) 1933–1942.
[16] C.-M. Pun, J.-L. Chung, A two-stage localization for copy-move forgery detection, Inf. Sci. 463 (2018) 33–55.
[17] H. Li, J. Huang, Localization of deep inpainting using high-pass fully convolutional network, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2019, pp. 8301–8310.
[18] D. Singhal, A. Gupta, A. Tripathi, R. Kothari, CNN-based multiple manipulation detector using frequency domain features of image residuals, ACM Trans. Intell. Syst. Technol. 11 (4) (2020) 1–26.
[19] B. Xiao, Y. Wei, X. Bi, W. Li, J. Ma, Image splicing forgery detection combining coarse to refined convolutional neural network and adaptive clustering, Inf. Sci. 511 (2020) 172–191.
[20] F. Ding, H. Wu, G. Zhu, Y.-Q. Shi, METEOR: Measurable energy map toward the estimation of resampling rate via a convolutional neural network, IEEE Trans. Circuits Syst. Video Technol. Early Access, Jan. 3, 2020, Doi:10.1109/TCSVT.2019.2963715..
[21] Q. Wu, S.-J. Sun, W. Zhu, G.-H. Li, D. Tu, Detection of digital doctoring in exemplar-based inpainted images, in: 2008 International Conference on Machine Learning and Cybernetics, IEEE, 2008, pp. 1222–1226.
[22] K.S. Bacchuwar, K. Ramakrishnan, A jump patch-block match algorithm for multiple forgery detection, in: 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing, IEEE, 2013, pp. 723–728..
[23] I.-C. Chang, J.C. Yu, C.-C. Chang, A forgery detection algorithm for exemplar-based inpainting images using multi-region relation, Image Vis. Comput. 31 (1) (2013) 57–71.
[24] Z. Liang, G. Yang, X. Ding, L. Li, An efficient forgery detection algorithm for object removal by exemplar-based image inpainting, J. Vis. Commun. Image Represent. 30 (2015) 75–85.
[25] X. Zhu, Y. Qian, X. Zhao, B. Sun, Y. Sun, A deep learning approach to patch-based image inpainting forensics, Signal Process. Image Commun. 67 (2018) 90–99.
[26] X. Wang, H. Wang, S. Niu, An image forensic method for AI inpainting using faster R-CNN, in: International Conference on Artificial Intelligence and Security, Springer, 2019, pp. 476–487..
[27] H. Li, W. Luo, J. Huang, Localization of diffusion-based inpainting in digital images, IEEE Trans. Inf. Forensics Secur. 12 (12) (2017) 3050–3064.
[28] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431–3440.
[29] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-assisted Intervention, Springer, 2015, pp. 234–241..
[30] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: A deep convolutional encoder-decoder architecture for image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (12) (2017) 2481–2495.
[31] S.S. Seferbekov, V. Iglovikov, A. Buslaev, A. Shvets, Feature pyramid network for multi-class land segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 272–275.

[32] M.M. Rahman, Y. Tan, J. Xue, L. Shao, K. Lu, 3D object detection: Learning 3D bounding boxes from scaled down 2D bounding boxes in RGB-D images, Inf. Sci. 476 (2019) 147–158.
[33] Z. Li, J. Tang, X. He, Robust structured nonnegative matrix factorization for image representation, IEEE Trans. Neural Netw. Learn. Syst. 29 (5) (2017) 1947–1960.
[34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2117–2125.
[35] J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, Striving for simplicity: The all convolutional net, arXiv preprint arXiv:1412.6806..
[36] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167..
[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European Conference on Computer Vision (ECCV), Springer, 2014, pp. 740–755..
[38] D. Tschumperlé, S. Fourey, GMIC: GREYCs Magic for Image Computing: A full-featured open-source framework for image processing, Available: http://gmic.eu..
[39] G. Schaefer, M. Stich, UCID: An uncompressed color image database, in: Storage and Retrieval Methods and Applications for Multimedia 2004, vol. 5307, International Society for Optics and Photonics, 2003, pp. 472–480..
[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, 2019, pp. 8026–8037..
[41] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980..