Multimedia Tools and Applications (2021) 80:23377–23392
https://doi.org/10.1007/s11042-020-10211-1

1163: LARGE-SCALE MULTIMEDIA SIGNAL PROCESSING FOR SECURITY AND DIGITAL FORENSICS

A hybrid convolutional architecture for accurate image manipulation localization at the pixel-level

Yixuan Zhang[1,2] · Jiguang Zhang[3] · Shibiao Xu[3]

Received: 1 March 2020 / Revised: 31 July 2020 / Accepted: 9 December 2020 /
Published online: 22 January 2021

Abstract
Advanced image processing techniques can easily edit images without leaving any visible traces, making manipulation detection and localization for forensics analysis a challenging task. Few studies can simultaneously locate tampered objects accurately and refine contours of tampered regions effectively. In this study, we propose an effective and novel hybrid architecture, named Pixel-level Image Tampering Localization Architecture (PITLArc), which integrates the advantages of top-down detection-based methods and bottom-up segmentation-based methods. Moreover, we provide a typical fusion implementation of our proposed hybrid architecture on one outstanding detection-based method (two-stream faster region-based convolutional neural network (RGB-N)) and two segmentation-based methods (Multi-Scale Convolution Neural Networks (MSCNNs) and Dual-domain Convolutional Neural Networks (DCNNs)) to evaluate the effectiveness of the proposed architecture. The three methods can be integrated into our proposed PITLArc to significantly improve their performance. Other detection and segmentation algorithms (not limited to the three aforementioned methods) can also be integrated into our architecture to improve their performance. Moreover, a Dense Conditional Random Fields (DenseCRFs)-based post-processing method is introduced to further optimize the details of tampered regions. Experiments validate the effectiveness of the proposed architecture.

Keywords Manipulation localization · Top-down detection · Bottom-up segmentation · DenseCRFs

✉ Shibiao Xu
shibiao.xu@nlpr.ia.ac.cn

[1] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

[2] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

[3] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China

1 Introduction

With the development of image processing techniques, the content of images can be easily modified without leaving any visual traces, making tampering detection for forensics analysis a critical issue.

Compared with determining whether an image has been tampered with, researchers are more concerned about detailed tampered regions [19]. Various methods have been proposed to localize tampered regions in an image. Methods for detecting double compression [6, 7, 21] are only effective in JPEG images. Some studies localize tampered regions on the basis of blur inconsistency traces [2, 17], however, such methods will be ineffective if there is no motion blur in the detected image. Photo Response Non-Uniformity (PRNU)-based methods [9, 11] use PRNU patterns to separate tampered regions, however, the PRNU pattern of a camera can only be obtained by capturing numerous images using the camera, which is impractical in most scenarios. Color Filter Array (CFA)-based methods [13, 14] and noise–based methods [24, 25] are not as accurate as deep learning-based methods [4, 29, 34] in locating tampered regions.

The extensive use of deep learning tools has recently promoted the development of image tampering detection and localization techniques, such as Convolutional Neural Networks (CNNs) [5, 10, 22, 30, 32, 34], autoencoder [31, 33] and Long Short-Term Memory (LSTM) [3, 4]. Algorithms for tampering localization are generally divided into two categories: (1) detection-based methods at the region level and (2) segmentation-based methods at the pixel-level.

Motivated by the efficient classification performance of CNNs, top-down object detection network has been extended to image manipulation detection tasks. Zhou et al. [34] recently proposed a state-of-the-art two-stream faster region-based convolutional neural network (RGB-N) for manipulation detection. However, the obtained region proposals are affected by multiple scales of tampered objects and various backgrounds in an image, and thus cannot sufficiently cover all tampered areas, leading to high missed detection rates. Moreover, such methods generally lack details even if they can accurately localize tampered objects at the region level.

Fully Convolutional Network (FCN) [23] and its subsequent models, U-Net [28], SegNet [1] and DeepLab [8], are proposed for semantic segmentation and have significantly improved the segmentation accuracy. However, these networks segment an image on the basis of semantic information instead of unnatural tampered boundaries, thus they cannot effectively localize tampered regions.

In bottom-up segmentation-based tampering localization networks, tampered edges can be obtained with high confidence on the basis of the discontinuity of neighboring pixels. Recent studies, such as that of Bappy et al. [3], propose a LSTM-based network for localizing tampered objects and they improved their work in subsequent research [4]. Salloum et al. [29] adopt modified FCN [23] to predict tampered regions. A tampering localization method [22] proposed recently depends on Multi-Scale CNNs (MSCNNs) to achieve pixel-level accuracy. Inspired by MSCNNs, the Dual-domain CNNs (DCNNs) [30] is established to derive highly comprehensive information. However, local noises in the background lead to high error rates for these segmentation networks.

Several shortcomings generally arise behind the elaborate design of the above two strategies. Our work aims to refine these two kinds of networks.

2 Preliminaries

We propose an effective and novel image manipulation localization framework that ingeniously utilizes the mutual promotion and complementarity of detection and segmentation methods to solve the above problems and subsequently improve the integrity and accuracy of pixel-level tampering localization . We verify our strategy in this study with those of three outstanding studies [22, 30, 34], which represent the typical idea of most tampering detection algorithms. We introduce the strategies into the proposed framework to prove the effectiveness of our algorithm. Other similar manipulation localization algorithms (i.e., not limited to what we use) that refer to the abovementioned detection and segmentation strategies can be integrated into our framework to improve their performances. For illustrative purpose, we will review the three studies mentioned above for the preliminaries of our experiment.

In many image manipulation cases, people often splice a complete object from one image to another, which allows a tampered object to have complete semantic information. Detection networks in computer vision can well recognize these objects. Thus, the tampering localization problem is easily converted into a target detection problem. Among all the object detection networks, faster region-based convolutional neural networks (Faster-RCNN) [27] is widely used for its high accuracy and end-to-end working mode. Zhou et al. [34] ($BaselineN_1$) aimed to learn rich tampering features in an image by fusing visual features and local noise features. They proposed a two-stream Faster-RCNN to detect tampered areas in a given image. The RGB stream extracts features in RGB channels. However, color images contain numerous semantic information and excessive semantic information will sometimes mask tampering traces (e.g. color contrast and inconsistencies of boundaries). Moreover, some tampered images may have undergone a series of elaborate post-processing, which hides the splicing boundaries and reduces the contrast between pristine and tampered regions. As such, relying only on features in the color channel cannot sufficiently track all tampering traces. A noise stream is then proposed to exploit Spatial-domain Rich Model (SRM) [15] and find noise inconsistencies, SRM is a set of features able to reveal unobvious traces leaving by steganography or forgery, which is widely used in steganalysis and forgery detection. Bilinear pooling [16, 20] is used to obtain richer features by fusing the two streams. The limitations of $BaselineN_1$ are poor concentration on small tampered regions and not all tampered objects are covered by region proposals, both of which seriously increases the missed detection rate. In contrast to generating several rectangular tampering areas in the top-down detection algorithm $BaselineN_1$, the bottom-up semantic segmentation is introduced into the tampering localization tasks to refine the tampered boundaries. An image tampering localization method that has been proposed recently as $BaselineN_2$ [22] uses multi-scale convolutional neural networks (MSCNNs) with pixel-level accuracy. In order to exploit features in different scales, patches of five sizes are extracted in a sliding-window manner to generate five possibility maps that can be fused into the final possibility map. Inspired by MSCNNs, the dual-domain convolutional neural networks (DCNNs) [30] ($BaselineN_3$), which is another sliding-window-and-patch-based method, is proposed to derive highly comprehensive information in images. The DCNNs consists of a spatial domain sub-network (SCNN) to process the correlation feature extracted from the RGB color channel, and a frequency domain sub-network (FCNN) to process the three-level daubechies-based discrete wavelet transform (DWT) feature in the frequency domain. Although the segmentation network can reach pixel-level accuracy,

it tends to misclassify pristine pixels as tampered ones due to the lack of information of the whole image in each patch. As such, bottom-up segmentation methods generally have lower accuracies than top-down detection methods.

In this study, we propose a hybrid architecture, named Pixel-level Image Tampering Localization Architecture (PITLArc), to take advantage of the complementary characteristics of the top-down and the bottom-up strategies. Compared with other state-of-the-art methods, PITLArc can improve the accuracy of pixel-level manipulation localization by a large margin. Our contributions can be summarized as follows.

1. We propose a novel hybrid architecture PITLArc for accurate tampering localization. More importantly, existing detection and segmentation networks can be integrated into our multitask hybrid architecture to improve their performance: Our proposed architecture, including the detection, the segmentation the fusion parts, takes advantage of two types of tampering localization networks to achieve accurate localization at the pixel-level (Fig. 1).
2. We propose a pixel-wise possibility map, namely truncated possibility map, to reduce the false alarm rate and improve localization accuracy: The proposed possibility map retains only the tampering possibility of pixels inside the detection bounding boxes. Such process can exclude many untampered regions, thus significantly reducing the false alarm rate and improving localization accuracy.
3. We provide a typical high-quality implementation on three outstanding studies to validate the effectiveness of the proposed PITLArc: The typical implementation of two stream detection and segmentation networks is adopted to evaluate the effectiveness of the proposed architecture. Our architecture can output high-quality image manipulation localization and the fusion strategy in our method is efficient.
4. We apply Dense Conditional Random Fields (DenseCRFs) [18] as our post-processing method to further improve localization accuracy: Although the outline of the tampered area is converted from a rectangle to a curve after fusion, numerous details remain unclear. A DenseCRFs-based post-processing operation is applied to the generated truncated possibility map for further improvement.

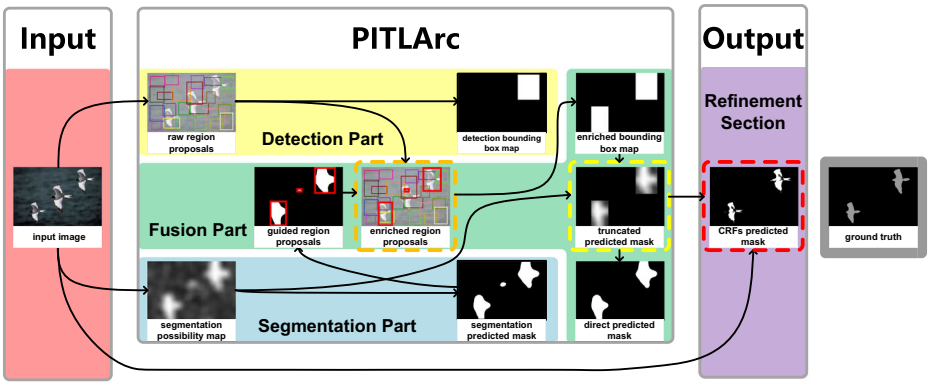

Fig. 1 Overall framework of PITLArc. Block in orange(outputs are enriched region proposals), yellow(output is truncated possibility map) and red(output is CRFs predicted mask) dashed line are our first fusion, second fusion and final output

3 Proposed method

As **Detection Part** in PITLArc, detection-based methods is used to generate multiple region proposals to approximately locate tampered regions. As **Segmentation Part** in PITLArc, segmentation-based methods is used to generate pixel-level tampering spatial attention. The **Fusion Part** in PITLArc has two stages. In the first stage, raw region proposals in the detection part are supplemented with segmentation predicted mask to fully cover tampered regions. In the second stage, an accurate region-level enriched bounding box map is used as a constraint to eliminate non-tampered areas in segmentation possibility map. The rich details of segmentation possibility map can then refine the edges of the enriched bounding box map, generating the critical truncated possibility map. In our **Refinement Section**, one DenseCRFs-based post-processing operation is applied to the truncated possibility map using the information of the RGB image to further improve localization accuracy. The proposed PITLArc (Fig. 1) is described in detail in the following subsections.

3.1 Detection part

In our implementation, RGB-N is used as the basic network ($baselineN_1$) for the detection part (network for detection part is not limited to RGB-N). First, we extract features in the RGB channels to generate and select region proposals through the Region Proposal Network (RPN). Simultaneously, we introduce noise features as auxiliary clues to improve the localization accuracy and we use spatial-domain rich model (SRM) [15] filter to extract local noise features. Similar to a recent work on image tampering classification [12], noise is modeled by the residual between each pixel and its neighboring pixels. Lastly, we use compact bilinear pooling [16] to fuse RGB features with noise features to obtain strong features. Bilinear pooling is defined as:

$$B(l, I, F_{color}, F_{noise}) = F_{color}(l, I)^T F_{noise}(l, I) \tag{1}$$

where l is the location, and I is the input image. F_{color} and F_{noise} denote the feature map in the RGB channel and the noise channel respectively.

Then, the region proposals are sent to the second stage of the detection-based network for further classification.

We use cross-entropy loss to classify tampered areas and smooth $L1$ loss to obtain accurate rectangular boxes. The overall loss function is:

$$L = L_{RPN} + L_{cls}(F_{color}, F_{noise}) + L_{reg}(F_{color}) \tag{2}$$

where L_{cls}, L_{reg} and L_{RPN} are cross-entropy classification loss, bounding box regression loss and RPN loss respectively.

3.2 Segmentation part

In our implementation, we introduce MSCNNs as $baselineN_2$ and DCNNs as $baselineN_3$ for the segmentation part (network for segmentation part is not limited to MSCNNs or DCNNs). The output of the part is segmentation possibility map, which can be binarized as segmentation predicted mask.

3.2.1 MSCNNs (*baselineN*$_2$)

The network exploits the features from 5 different patch sizes to train 5 CNNs, and the outcomes of these CNNs are fused into one possibility map. First, we need to extract patches from images. For each forged image, patches with fixed sizes are extracted by using a sliding window with a step size of 8. Each patch is marked with a label. Patches with tampered regions that range from 10% to 90% are labeled fake, mainly because this kind of patches can learn the inconsistency between pristine and manipulated regions. Patches without any manipulated pixels are labeled pristine. Considering that tampered regions differ significantly in all tampered images, we can extract thousands of patches from some images and extract only a few from the others. We set a patch threshold T to avoid overfitting in the training process. In this manner, if more than T pristine or fake patches are extracted from one image, then we randomly choose T for training.

Then all the patches are fed into a binary-classification CNN to judge whether they are manipulated. In MSCNNs, SRM kernels are also used to strengthen color contrast and inconsistencies of the boundaries. Different from $baselineN_1$ in the detection part, we use all 30 SRM kernels in [32] to initialize the MSCNNs.

In the testing phase, firstly, image patches are obtained in the same manner as that in the training phase. Secondly, the binary-classification CNN trained in the training phase will be used to calculate the tampering possibility of each image patch. Then, with the tampering possibility of all patches extracted in an input image R, we will obtain the tampering possibility map M of the image R. In the tampering possibility map M, the value of each pixel m_{ij} equals the tampering possibility of the corresponding pixel r_{ij} in image R. m_{ij} can be calculated as:

$$m_{ij} = \frac{\sum_{n=1}^{K} P_n}{K} \tag{3}$$

where K denotes the number of patches which contain pixel (i, j), $P_1, P_2, ..., P_K$ denote the tampering possibility of the K patches which contain pixel (i, j). Please notes that there exist some pixels which no patch contains, the tampering possibility of these pixels can not be calculated by (3) because the denominator of (3) equals 0, in this case, the tampering possibility of these pixels will be set as the tampering possibility of their closest pixels which can be calculated by (3).

3.2.2 DCNNs (*baselineN*$_3$)

The pixel-level possibility map will be more accurate if we use information from various aspects. MSCNN uses only the features of the spatial domain to train the CNNs. We improve the performance of the network with features in different domains to obtain more comprehensive information for each patch.

The DCNNs consists of SCNN to process the correlation features extracted from the RGB color channel and FCNN to process the three-level Daubechies-based DWT feature in the frequency domain. The outputs of the two layers are fused and used as a joint input to the classification module. As previously mentioned, we first convert 64×64 patches generated in an image (the patch generation method is the same as the MSCNN) into the YCbCr color space and decompose each channel with three levels of DWT. In each level of DWT composition, three different subbands are produced in three directions: horizontal,

vertical, and diagonal. Then, 10 subband coefficient matrices are obtained by multi-level wavelet decomposition. For each subband, seven statistical features are calculated:

$$f = < \mu, \sigma, \xi, E, Con, Im, Idm > \tag{4}$$

$$\mu = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i, j) \tag{5}$$

$$\sigma = \sqrt{\frac{1}{MN - 1} \sum_{i=1}^{M} \sum_{j=1}^{N} |p(i, j) - \mu|} \tag{6}$$

$$\xi = \sum_{i=1}^{M} \sum_{j=1}^{N} p^2(i, j) \tag{7}$$

$$E = -\sum_{i=1}^{M} \sum_{j=1}^{N} p(i, j) log(p(i, j)) \tag{8}$$

$$Con = \sum_{i=1}^{M} \sum_{j=1}^{N} (i - j)^2 p(i, j) \tag{9}$$

$$Im = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{1}{|i - j|} p(i, j) \tag{10}$$

$$Idm = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{1}{1 + (i - j)^2} p(i, j) \tag{11}$$

where μ denotes the mean, σ denotes the variation, ξ denotes the energy, E denotes the discrepency, Con denotes the sharpness, Im denotes the changes level, Idm denotes the homogeneity, M and N denote patch size and $p(i, j)$ denotes the pixel value of each subband.

In summary, with 3 color channels, 4 Daubechies-based DWT families, 10 subbands, and 7 statistics, we have derived a 840 dimensional feature which are then used to train the FCNN.

3.3 Fusion part and refinement section

We validate the effectiveness of the proposed method in terms of utilizing the cooperation of two strategies to achieve a highly accurate and efficient tampering localization at the pixel level.

3.3.1 First fusion

Detection-based methods can effectively capture image content rather than unnatural edges. Furthermore, we observe that when there is a stark contrast between tampered area and its surroundings, detection-based methods can accurately locate these areas. But if a tampered regions have similar brightness or chroma as its surroundings, detection-based methods will perform poorly.

We derive hundreds of region proposals in the first stage of detection-based networks. However, as mentioned in the previous paragraph, several tampered regions will be excluded

in the raw region proposals. In PITLArc, a rectangular boundary is added to each connected part in the segmentation predicted mask as guided region proposals to enrich the raw region proposals generated by detection-based networks. Then, the enriched region proposals are sent to the second stage of the detection-based networks when conducting further localization and classification.

3.3.2 Second fusion

In segmentation methods, given that tampering possibility of each patch in an image is calculated separately without connection to other parts of the image, the tampering possibility is easily influenced by local noises. Thus, the segmentation-based methods may misclassify some pristine areas as tampered. Unlike segmentation-based methods, detection-based methods calculate tampering possibility of bounding boxes on the basis of entire image content, thus the methods can locate tampered regions more accurately at the region level.

In PITLArc, segmentation possibility map is constrained by enriched bounding box map to generate truncated possibility map. In the segmentation possibility map, each region is a suspected tampered region; the enriched bounding box map is composed of several rectangular areas. The suspected tampered regions in the segmentation possibility map are constrained by enriched bounding box maps from the entire map to the rectangular areas in an enriched bounding box map. With the help of enriched bounding box map which can accurately locate tampered area, the suspected tampered regions are reduced from the entire image (in segmentation possibility map) to a few precise rectangular areas (in truncated possibility map), thus significantly improving localization accuracy. For truncated possibility map, the regions outside enriched bounding box map are all considered pristine and the tampering possibility of each pixel inside the enriched bounding box map is the same as that in segmentation possibility map.

3.3.3 Refinement section

If we directly binarize truncated possibility map on the basis of a threshold, the predicted mask (direct predicted mask) will generally lack details. DenseCRFs can refine the details of predicted mask on the basis of mutual information between neighboring pixels in the RGB channel. Therefore we choose DenseCRFs as our post-processing tool to further improve localization accuracy.

DenseCRFs needs RGB image and its tampering possibility map as input. Through analyzing semantic information in RGB image and tampering information in tampering possibility map, DenseCRFs will produce a binary mask(CRFs predicted mask) indicating which pixels are tampered and which pixels are pristine.

We use PyDenseCRFs [26] as the specific DenseCRFs implementation.

4 Experiment

4.1 Implementation detail

We apply TensorFlow to build the whole network, including network in the Detection Part, the Segmentation Part, and the Fusion Part . The four anchors for detection are set to $\{8, 16, 32, and 64\}$, and the perspective ratios are$\{1 : 2, 1 : 1, and 2 : 1\}$. For multi-scale segmentation, we use $\{16, 24, 32, 48, and 64\}$ patch sizes in the CASIA2.0 dataset

and {32, 48, 64, 96, and128} patch sizes in the other datasets. Furthermore, we extract the DWT feature and normalize all the 840 features to allow the network to converge easily. All subnetworks are first trained separately with the coco synthetic dataset as proposed in [34], and all weights and biases are initialized with the Xavier initializer and the zero initializer, respectively. The batch sizes are set to {3000, 2000, 1000, 800, and600}. We apply the momentum optimizer to decrease the loss, and we use a fixed momentum that is set to 0.9. Then, all pre-trained subnetworks are further fine-tuned in 45000 iterations simultaneously, and the initial learning rate equals to 0.001. The learning rate decreases by 0.1 every 15000 iteration.

4.2 Evaluation metrics

Three evaluation metrics (F1 score(F1), Matthews Correlation Coefficient(MCC) and pixel-level accuracy(ACC)) are used to evaluate the proposed method. F1 is a reliable measure in binary classification even if the two classes are imbalanced, the value of F1 is between 0 and 1 and a higher value means more accurate classification; MCC is another reliable measure in binary classification even if the two classes are imbalanced, the value of MCC is between -1 and 1 and a higher value means more accurate classification; ACC is a simple measure in binary classification, the value of ACC is between 0 and 1 and a higher value means more accurate classification.

$$P = \frac{TP}{TP + FP} \tag{12.1}$$

$$R = \frac{TP}{TP + FN} \tag{12.2}$$

$$F1 = \frac{2PR}{P + R} \tag{12.3}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{12.4}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP denotes the number of pristine pixels classified as pristine, FP denotes the number of manipulated pixels classified as pristine, FN denotes the number of pristine pixels classified as manipulated and TN denotes the number of manipulated pixels classified as manipulated. In the segmentation part, we set 0.5 as the possibility threshold for the binary classification. In the detection part, we use various thresholds and select the highest F1score (or MCC, ACC), which is consistent with that in [34].

4.3 Datasets

Four datasets (CASIA2.0, COVER, Columbia and NIST16) are used to evaluate the proposed method. In the detection-based methods, the networks are first pretrained separately on the coco synthetic dataset [34]; in the segmentation-based methods, we do not pretrain the networks.

4.3.1 CASIA2.0

CASIA2.0 is a dataset with more than 5000 images. Most of the images in CASIA2.0 are spliced, while the others are copy-move tampered. The images in CASIA2.0 are post-processed with blurring, smoothing, and jpeg compression. The resolution of CASIA2.0 images is somewhat low, making it difficult to detect tampered regions.

Mask is not available in CASIA2.0. Thus, we compare the tampered images with its original images to acquire the masks.

4.3.2 COVER

COVER is a small dataset with only 100 images. All the images in COVER is copy-move tampered and the mask is available.

4.3.3 Columbia

All images in the Columbia dataset are spliced in high resolution and the mask for each image is provided.

4.3.4 NIST16

NIST16 is a high-resolution dataset, and many images in the dataset are post-processed to conceal the visible spliced boundaries, which makes the tampered traces difficult to detect.

4.3.5 Datasets Splitting

The dataset splitting is shown in Table 3. For CASIA2.0, COVER, and NIST16, we randomly choose 75% images for training and 25% for testing. For the Columbia dataset, we use all the images for testing.

4.4 Performance analysis

The aim of the performance analysis is to validate our fusion strategy based on experimental results and visual comparison.

As shown in Tables 1 and 2, our fusion strategy outperforms all three baseline methods and MFCN [29].

Pixel-level accuracies for NIST16 dataset are: 0.9480 (LSTM-EnDec [4]), 0.9424 (PITLArc). PITLArc is comparable with LSTM-EnDec in terms of pixel-level accuracy.

4.4.1 Improvement for detection

Tables 1 and 2 show that the results of the proposed fusion methods are better than the results of the detection-based method $baselineN_1$, that is mainly because (1)pixel-level segmentation-based methods can refine the contours of the generated bounding boxes by the detection-based methods. (2)the guided region proposals added to the raw region proposals can help retrieve some neglected tampered regions.

In Table 1, it can be seen that the proposed fusion method improves the F1 score in NIST16 dataset of detection-based method $baselineN_1$ from 0.6286 to 0.7215.

Table 1 Performance evaluation via F1 score (F1), Precision (P) and recall (R)

Methods \ Datasets	CASIA2.0			COVER			Columbia			NIST16		
	F1	P	R	F1	P	R	F1	P	R	F1	P	R
Faster-RCNN($BaselineN_1$) [34]	0.4380	0.3655	0.6632	0.4906	0.4443	0.5855	0.6971	0.5696	0.9384	0.6286	0.5475	0.8597
MSCNNs($BaselineN_2$) [22]	0.3601	0.2733	0.8486	0.3988	0.3292	0.7291	0.4391	0.3294	0.8773	0.5195	0.4437	0.8906
DCNNs($BaselineN_3$) [30]	0.3913	0.3174	0.7508	0.4409	0.4278	0.6202	0.4837	0.4174	0.8938	0.6592	0.6061	0.7967
MFCN [29]	-	-	-	-	-	-	0.6117	-	-	0.5707	-	-
Our HybridArch$_{\{1+2\}}$	0.5176	0.4570	0.7573	0.5539	0.5933	0.6914	0.6837	0.6204	0.8487	0.6743	0.6612	0.8123
Our HybridArch$^{*}_{\{1+2\}}$	0.5431	0.5322	0.7192	0.5717	0.6238	0.6966	0.7229	0.6945	0.8485	0.6797	0.6711	0.8126
Our HybridArch$_{\{1+3\}}$	0.5040	0.4621	0.6765	0.5772	0.5925	0.6456	0.7058	0.6699	0.8652	0.7215	0.7286	0.7797
Our HybridArch$^{*}_{\{1+3\}}$	0.5251	0.5339	0.6381	0.5888	0.5858	0.6750	0.7550	0.7515	0.8608	0.7247	0.7348	0.7797

The * means with DenseCRFs post-processing

Table 2 Performance evaluation via MCC

Methods \ Datasets	CASIA2.0	COVER	Columbia	NIST16
Faster-RCNN($BaselineN_1$) [34]	0.4333	0.4796	0.6039	0.6349
MSCNNs($BaselineN_2$) [22]	0.3856	0.3679	0.1151	0.5537
DCNNs($BaselineN_3$) [30]	0.4050	0.4131	0.1498	0.6627
MFCN [29]	–	–	0.4792	0.5703
Our HybridArch$_{\{1+2\}}$	0.5203	0.5669	0.5747	0.6846
Our HybridArch$^{*}_{\{1+2\}}$	0.5476	0.5613	0.6384	0.6908
Our HybridArch$_{\{1+3\}}$	0.5022	0.5790	0.6069	0.7241
Our HybridArch$^{*}_{\{1+3\}}$	0.5259	0.5731	0.6751	0.7279

The * means with DenseCRFs post-processing

4.4.2 Improvement for segmentation

As shown in Tables 1 and 2, the proposed fusion method has improved the localization performance of segmentation-based method $baselineN_2$ and $baselineN_3$, that is because segmentation-based methods are easily influenced by local noises and thus tend to misclassify pristine areas as tampered, and the detection-based methods can constrain the suspected tampered regions in segmentation possibility map generated by segmentation-based methods and thus can reduce the false alarm rate.

In Table 1, it can be seen that the proposed fusion method improves the F1 score in Columbia dataset of segmentation-based method $baselineN_2$ from 0.4391 to 0.6837.

4.4.3 Improvement in DenseCRFs

As shown in Tables 1 and 2, the DenseCRFs can further improve the F1 score and MCC, that is because DenseCRFs can utilize semantic information in input RGB image to refine the detail of the predicted mask.

In Table 1, it can be seen that the proposed fusion method improves the F1 score in Columbia dataset of Our HybridArch$^{*}_{\{1+3\}}$ from 0.7058 to 0.7550 (Table 3).

4.4.4 Computing time

The average computing time for CASIA2.0 dataset are as follows: 0.293 s (RGB-N), 3.522 s (MSCNNs), 22.260 s (DCNNs), 0.005 s (Fusion1), 0.006 s (Fusion2), 0.302 s (DenseCRFs). The average computing time for COVER dataset are as follows: 0.343 s (RGB-N), 5.424 s (MSCNNs), 28.172 s (DCNNs), 0.006 s (Fusion1), 0.006 s (Fusion2), 0.328 s (DenseCRFs). The figure shows that our fusion strategy does not require much time compared with three baseline methods.

Table 3 Training and testing dataset splitting

Datasets	CASIA2.0	Columbia	COVER	NIST
Training	3842	-	75	110
Testing	1281	180	25	36

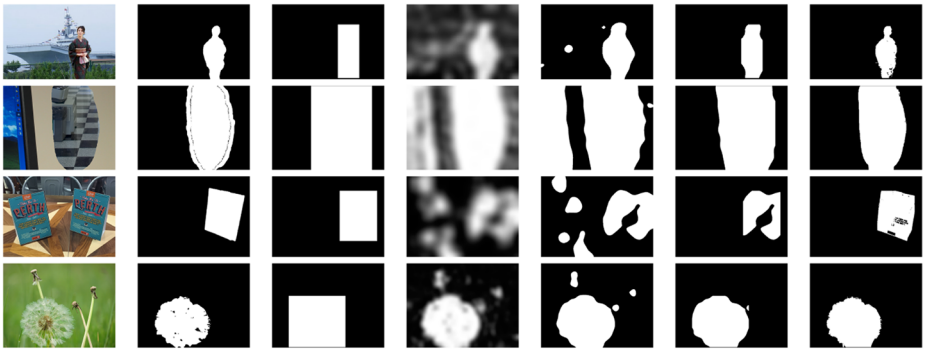

Fig. 2 Visual comparison for PITLArc. From top to bottom: CASIA2.0, Columbia, COVER and NIST16 dataset. From left to right: input images, ground truths, detection bounding box maps, segmentation possibility maps, segmentation predicted masks, direct predicted masks and CRFs predicted masks

4.4.5 Visual comparison

The visual results in Figs. 2 and 4 further validate the effectiveness of our method. Take the images in the second row in Fig. 2 as example. On the one hand, The segmentation-based method misclassifies the pristine region in the left part as tampered, whereas the detection-based method correctly classifies the pristine region in the left part. On the other hand, the output of the detection-based method is only a rectangular region and lacks detail, whereas the output of the segmentation-based method has richer detail. The fusion of the two networks takes advantage of both schemes, thus generating improved results. Moreover, In the last column, the DenseCRFs can further refine the contours of the tampered object to render the final result nearly the same as the ground truth.

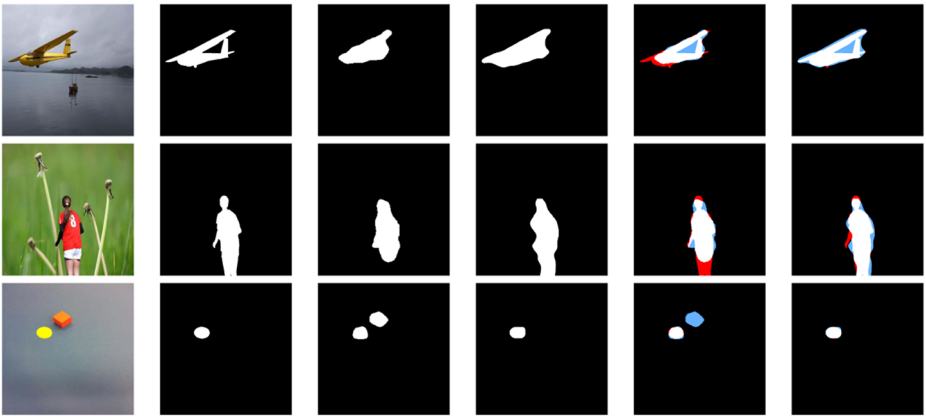

Fig. 3 Visual comparison between LSTM-EnDec [4] and PITLArc. From left to right: input images, ground truths, LSTM-EnDec results, our results, LSTM-EnDec error maps (in this column, F1 scores are 0.8101, 0.8378, 0.5415. Pixel-level accuracies are 0.9802, 0.9770, 0.9866), our error maps (in this column, F1 scores are 0.8287, 0.8851, 0.9515. Pixel-level accuracies are 0.9790, 0.9820, 0.9991). False negative, false positive, true negative and true positive pixels are marked with red, blue, black and white respectively

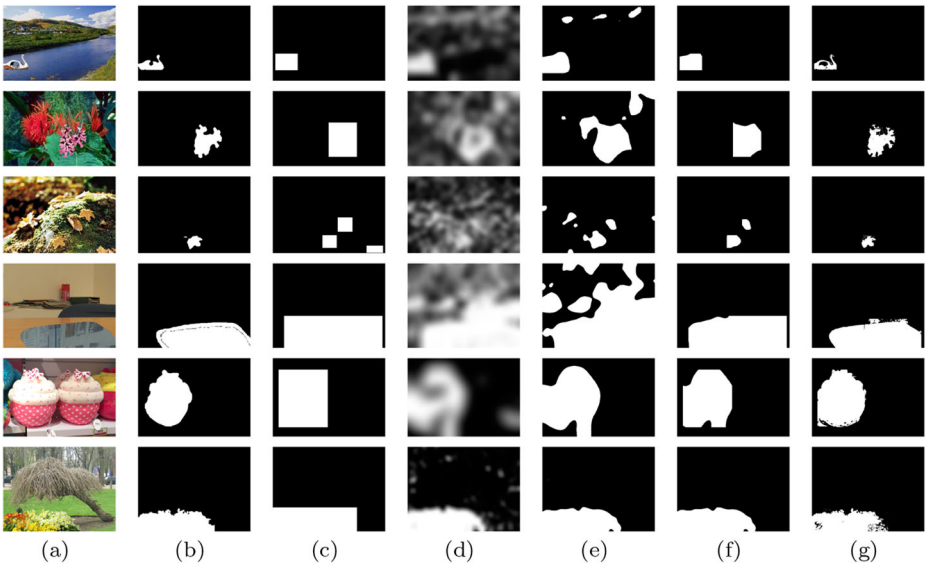

Fig. 4 Visual comparison for PITLArc on CASIA2.0 (1st-3th row), Columbia (4th row), COVER (5th row) and NIST16 (last row) datasets. From left to right: input images, ground truths, detection bounding box maps, segmentation possibility maps, segmentation predicted masks, direct predicted masks and CRFs predicted masks

Figure 3 presents visual comparison with LSTM-EnDec [4]. The comparison shows that LSTM-EnDec misclassified the left part of the plane (in the first row of Fig. 3) and the girl's legs (in the second row of Fig. 3) as untampered. LSTM-EnDec is based on LSTM and the input images must be divided into patches before sent into the network, thus, the network may be unsuitable for low resolution images. Moreover, as edge patches have fewer neighboring patches than inner patches, LSTM-EnDec does not perform well near edges. In addition, LSTM-EnDec misclassify the orange box (in the third row of Fig. 3) as tampered. By contrast, PITLArc correctly classify this box, because the detection part of PITLArc exclude the orange box from the tampered regions in the second fusion stage. Notably, depending only on pixel-level accuracy to evaluate a network is incomprehensive, thus the PITLArc is better than LSTM-EnDec in terms of visual comparison although the two networks have nearly the same pixel-level accuracy (Fig. 3). We also calculate the F1 scores for the three images in Fig. 3, the F1 scores of PITLArc are much higher than those of LSTM-EnDec (Fig. 4).

5 Conclusion

Our proposed hybrid architecture PITLArc takes the advantages of the complementary characteristics of top-down detection-based methods and bottom-up segmentation-based methods. The pixel-level segmentation-based methods for spatial attention can enrich region proposals in the detection process and reduce missed detection rates. Moreover, detection-based methods provide constraints on the segmentation-based methods, and the segmentation-based methods refine the details of the detection-based methods. Thus, the

two approaches complement each other in this fusion strategy. Lastly, a DenseCRFs operation is applied to refine the predicted mask and further improve the localization accuracy. We also provide a typical implementation of PITLArc on three outstanding works. Experiments on different datasets show that PITLArc significantly improves the localization accuracy. In the future, the proposed hybrid architecture can be improved by adopting more powerful CNN architectures.

Acknowledgments This work was supported by NSFC under U1636102, U1736214, 61802393 and 61872356, National Key Technology R&D Program under 2016QY15Z2500, and Project of Beijing Municipal Science & Technology Commission under Z181100002718001.

References

1. Badrinarayanan V, Kendall A, Cipolla R (2015) Segnet: A deep convolutional encoder-decoder architecture for image segmentation. CoRR 1511.00561
2. Bahrami K, Kot AC (2017) Image splicing localization based on blur type inconsistency. IEEE Trans Inf Forensic Secur 10(5):999–1009
3. Bappy MJH, Roy-Chowdhury AK, Bunk J, Nataraj L, Manjunath BS (2017) Exploiting spatial structure for localizing manipulated image regions. In: IEEE International Conference on Computer Vision
4. Bappy JH, Simons C, Nataraj L, Manjunath BS, Roy-Chowdhury AK (2019) Hybrid lstm and encoder–decoder architecture for detection of image forgeries. IEEE Trans Image Process 28(7):3286–3300
5. Bayar B, Stamm MC (2016) A deep learning approach to universal image manipulation detection using a new convolutional layer. In: ACM Workshop on Information Hiding and Multimedia Security
6. Bianchi T, Rosa AD, Piva A (2011) Improved dct coefficient analysis for forgery localization in jpeg images. In: IEEE International Conference on Acoustics
7. Bianchi T, Piva A (2012) Image forgery localization via block-grained analysis of jpeg artifacts. IEEE Trans Inf Forensic Secur 7(3):1003–1017
8. Chen L-C, Papandreou G, Kokkinos I, Murphy K, Yuille AL Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs
9. Chen M, Fridrich JJ, Goljan M, Lukás J (2008) Determining image origin and integrity using sensor noise. IEEE Trans Inf Forensic Secur 3(1):74–90
10. Chen J, Kang X, Ye L, Wang ZJ (2015) Median filtering forensics based on convolutional neural networks. IEEE Signal Process Lett 22(11):1849–1853
11. Chierchia G, Poggi G, Sansone C, Verdoliva L (2014) A bayesian-mrf approach for prnu-based image forgery detection. IEEE Trans Inf Forensic Secur 9(4):554–567
12. Cozzolino D, Poggi G, Verdoliva L (2016) Splicebuster: a new blind image splicing detector. In: IEEE International Workshop on Information Forensics and Security
13. Dirik AE, Memon N (2009) Image tamper detection based on demosaicing artifact. In: IEEE International Conference on Image Processing
14. Ferrara P, Bianchi T, Rosa AD, Piva A (2012) Image forgery localization via fine-grained analysis of cfa artifacts. IEEE Trans Inf Forensic Secur 7(5):1566–1577
15. Fridrich J, Kodovsky J (2012) Rich models for steganalysis of digital images. IEEE Trans Inf Forensic Secur 7(3):868–882
16. Gao Y, Beijbom O, Zhang N, Darrell T (2016) Compact bilinear pooling. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 317–326
17. Kakar P, Natarajan S, Ser W (2010) Detecting digital image forgeries through inconsistent motion blur. Icme, pp 486–491
18. Krähenbähl P, Koltun V Efficient inference in fully connected crfs with gaussian edge potentials
19. Li H, Luo W, Qiu X, Huang J (2017) Image forgery localization via integrating tampering possibility maps. IEEE Trans Inf Forensic Secur 12(5):1240–1252
20. Lin T-Y, RoyChowdhury A, Maji S (2015) Bilinear cnn models for fine-grained visual recognition. In: Proceedings of the IEEE international conference on computer vision, pp 1449–1457
21. Lin Z, He J, Tang X, Tang CK (2009) Fast, automatic and fine-grained tampered jpeg image detection via dct coefficient analysis? Pattern Recogn 42(11):2492–2501

22. Liu Y, Guan Q, Zhao X, Yun C (2018) Image forgery localization based on multi-scale convolutional neural networks
23. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3431–3440
24. Lyu S, Pan X, Xing Z (2014) Exposing region splicing forgeries with blind local noise estimation. Int J Comput Vis 110(2):202–221
25. Mahdian B, Saic S (2009) Using noise inconsistencies for blind image forensics. Image Vis Comput 27(10):1497–1503
26. PyDenseCRFs. https://github.com/lucasb-eyer/pydensecrf
27. Ren S, He K, Girshick RB, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. CoRR 1506.01497
28. Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention
29. Salloum R, Ren Y, Kuo CCJ (2017) Image splicing localization using a multi-task fully convolutional network (mfcn). J Vis Commun Image Represent 51:201–209
30. Shi Z, Shen X, Kang H, Lv Y (2018) Image manipulation detection and localization based on the dual-domain convolutional neural networks. IEEE Access
31. Ying Zhang LLWVT (2016) Image region forgery detection: A deep learning approach, vol 14. https://doi.org/10.3233/978-1-61499-617-0-1
32. Yuan R, Ni J (2017) A deep learning approach to detection of splicing and copy-move forgeries in images. IEEE International Workshop on Information Forensics and Security
33. Zhang Y, Thing VLL (2018) A semi-feature learning approach for tampered region localization across multi-format images. Multimed Tools Appl 77(19):25027–25052
34. Zhou P, Han X, Morariu VI, Davis LS (2018) Learning rich features for image manipulation detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH ("Springer Nature").

Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users ("Users"), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use ("Terms"). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.

These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.

We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.

While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.

These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.

To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.

Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.

If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com