Title

Exploiting Spatial Structure for Localizing Manipulated Image Regions

ISBN

978-1-5386-1032-9

Authors

Bappy, Jawadul H

Roy-Chowdhury, Amit K

Bunk, Jason

et al.

Publication Date

2017

DOI

10.1109/ICCV.2017.532

Peer reviewed

Exploiting Spatial Structure for Localizing Manipulated Image Regions

Jawadul H. Bappy[1], Amit K. Roy-Chowdhury[1], Jason Bunk[2], Lakshmanan Nataraj[2], and B.S. Manjunath[2,3]

[1]Department of Electrical and Computer Engineering, University of California, Riverside, USA

[2]Mayachitra Inc., Santa Barbara, California , USA

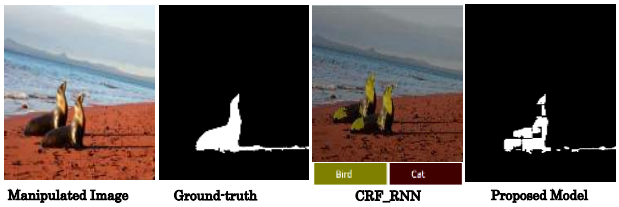
[3]Department of Electrical and Computer Engineering, University of California, Santa Barbara, USA

Abstract

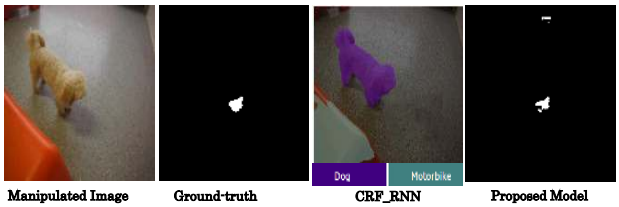
The advent of high-tech journaling tools facilitates an image to be manipulated in a way that can easily evade state-of-the-art image tampering detection approaches. The recent success of the deep learning approaches in different recognition tasks inspires us to develop a high confidence detection framework which can localize manipulated regions in an image. Unlike semantic object segmentation where all meaningful regions (objects) are segmented, the localization of image manipulation focuses only the possible tampered region which makes the problem even more challenging. In order to formulate the framework, we employ a hybrid CNN-LSTM model to capture discriminative features between manipulated and non-manipulated regions. One of the key properties of manipulated regions is that they exhibit discriminative features in boundaries shared with neighboring non-manipulated pixels. Our motivation is to learn the boundary discrepancy, i.e., the spatial structure, between manipulated and non-manipulated regions with the combination of LSTM and convolution layers. We perform end-to-end training of the network to learn the parameters through back-propagation given ground-truth mask information. The overall framework is capable of detecting different types of image manipulations, including copy-move, removal and splicing. Our model shows promising results in localizing manipulated regions, which is demonstrated through rigorous experimentation on three diverse datasets.

1. Introduction

With the availability of digital image editing tools, digital altering or tampering of an image has become very easy. In contrast, the identification of tampered images is a very challenging problem due to the strong resemblance of a forged image to its original one. There are certain types of manipulations such as copy-move, splicing, removal, that can easily deceive the human perceptual system. Digital image forensics is an emerging important topic in diverse scientific and security/surveillance applications. Most of the existing methods have focused on classifying whether an image is manipulated or not. However, there are few methods [51, 24, 13] that localize manipulated regions from an image. Some recent works address the localization problem by classifying patches as manipulated. In this paper, we propose a novel *detection framework* which is capable of locating manipulation at patch as well as pixel level.



(a)



(b)

Figure 1. The figure demonstrates the challenge of segmenting manipulated regions from an image. In this figure, we consider two types of manipulation-(a) copy-clone, and (b) removal. In (a), semantic segmentation method such as CRF-RNN [60] tries to segment two seals in the image, whereas the proposed method segments only the copied seal (manipulated) from an image. In (b), the detection of manipulated region is even harder - some part of the image has been removed and filled with the neighboring regions. Deep learning based segmentation method [60] is not able to segment removed objects, whereas our model is capable of localizing removed objects.

In image forensics, most of the state-of-the-art image tamper detection approaches exploit the frequency domain characteristics and/or statistical properties of an image. Some of the common methods are DWT [34], SVD [41], PCA [43], DCT [56]. The analysis of artifacts by multiple JPEG compressions is also utilized in [18, 56] to de-

tect manipulated images, which are applicable only to the JPEG formats. Recently, deep learning has become popular due to its promising performance in different visual recognition tasks such as object detection [26, 8], scene classification [62], and semantic segmentation [40]. There have been a few recent works which exploit stacked auto-encoders (SAE) [59], and convolutional neural networks (CNN) [50, 9, 19] in order to detect tampered images. Even though CNN has shown very promising performance in understanding visual concepts such as object detection and recognition, the detection of manipulated regions with CNNs may not be best strategy because well manipulated images usually do not leave any visual clue of alteration [50], and resemble genuine images.

In semantic segmentation, deep learning models [40, 60, 7] exhibit good performance by learning hiearachical features of different objects in an image. Recent advances in semantic segmentation involves coarse image representations, which are recovered by upsampling. However, coarse representation introduces significant loss of information which might be important for learning manipulated regions. In contrast to semantic segmentation, manipulated regions could be removed objects, or copied object from other part of the image. Fig. 1 explains the challenge of segmenting manipulated regions in an image. In Fig. 1(a), image is tampered in such a way that the manipulated and non-manipulated regions contain the same object (seal). Existing segmentation approaches will segment both of the objects. In addition, existing segmentation network fails to catch the removed object from an image which is shown in Fig. 1(b). However, our proposed model is able to segment the manipulated regions with high accuracy as shown in the last column of Fig. 1.

An image can be manipulated in many ways - removing objects from an image, splicing and copy-clone. Most of the existing forgery detection approaches focus on identifying a specific tampering method (such as copy-move [17, 29, 35], splicing [45]). Thus, these approaches might not do well for other types of tampering. Moreover, it becomes infeasible and unrealistic to assume that the type of manipulation will be known beforehand. In real-life, image tamper detection should be able to detect all types of manipulation rather than focusing on a specific type.

Towards this goal of detecting and localizing manipulated image regions, we present a unified deep learning framework in order to learn the patch labels (manipulated vs non-manipulated) and pixel-wise segmentation jointly. These two are intricately tied together, since patch classification can inform us about which pixels are manipulated, and segmentation will determine whether a patch is manipulated or not. Our multi-task learning framework exploits convolutional layers along with long-short term memory (LSTM) cells. We perform end-to-end training to learn the joint tasks through back-propagation using ground-truth patch labels and mask information. The proposed model shows promising results in localizing manipulated regions at the pixel level, as well as in patch classification, which is demonstrated on different challenging datasets.

Framework Overview: In this paper, our goal is to localize the manipulated regions from an image. Given an image, we first extract patches by sliding a windows across the image. In our framework, the image patch is taken as input and produces a patch label (manipulated or not) and a segmentation mask as output. Our overall framework consists of total 5 convolutional layers and an LSTM network with 3 stacked layers. The proposed framework is shown in Fig. 2. In the network, first two convolutional layers are used to learn the low-level features, such as edges and textures. After passing through two consecutive convolutional layers, we have a 2D feature map which has been divided into 8 by 8 blocks. These blocks are then fed into the LSTM network discussed in the following paragraph.

In computer vision, LSTMs are generally used to learn the temporal context of a video or any sequence of data. In this work, we use an LSTM to model the spatial relationships between neighboring pixels. This is because *manipulation breaks the natural statistics of an image in the manipulated boundary region*. We send the blocks of low level features obtained from second convolution layer to the LSTM cells sequentially, e.g., first block goes to first cell, second block to second cell, and so on. The 3-stacked LSTM layers produce the correlation features between blocks. These features are then used to classify patches using a softmax classifier, and passed to the series of convolution layers.

Finally, we obtain the 2D map with confidence score of each pixel using three consecutive convolutional layers on top of the LSTM network. With the ground-truth mask of manipulated regions we perform end-to-end training to classify each pixel. We compute the joint loss obtained at the patch classification layer and the final layer of segmentation, which is then minimized by utilizing back-propagation algorithm.

Main Contributions. Our *main contributions* are as follows.

• In this paper, we propose a unified network for patch classification and segmentation task using convolution layers along with an LSTM network. To the best of our knowledge, *there is no prior work on joint pixel-wise segmentation of manipulated regions and patch tamper classification.* The intricate relationship between the two, as explained above, justifies this integrated approach.

• In the proposed network, both patch classification and segmentation (pixel-wise classification) exploit the

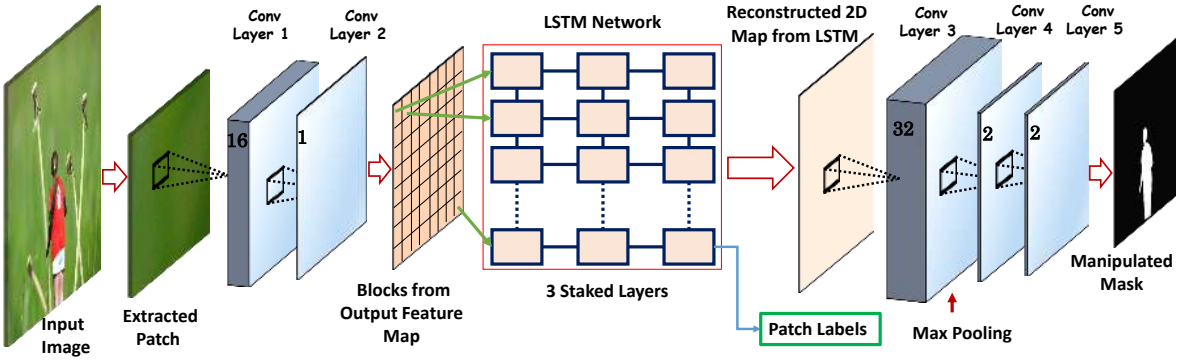


Figure 2. Overview of proposed framework for joint tasks- patch classification and manipulated region segmentation.

interdependence between them in order to improve both of the recognition tasks. Our framework is capable of localizing a manipulated region with high confidence, which is demonstrated on three datasets.

2. Related Work

The field of image forensics comprises of diverse areas to detect manipulation including resampling detection, JPEG artefacts, detection of copy-move operations, splicing, and object removal. We will briefly discuss some of them below.

In the past decades, several techniques have been proposed to detect resampling in digital images [52, 47, 23]. In most cases, it is assumed to be done using linear or cubic interpolation. In [52], the authors exploit periodic properties of interpolation by the second-derivative of the transformed image for detecting image manipulation. To detect resampling on JPEG compressed images, the authors added noise before passing the image through the resampling detector and showed that adding noise aids in detecting resampling [47]. In [22, 23], a feature is derived from the normalized energy density and then SVM is used to robustly detect resampled images. Some recent approaches [27, 33] have been proposed to reduce JPEG artefacts left by compression. In [5, 54], feature based forensic approaches have been presented in order to detect manipulation in an image.

In order to detect copy-move forgeries, an image is first divided into overlapping blocks and some sort of distance measure or correlation is used to determine blocks that have been cloned. Some recent works [35, 31, 30, 4] tackle the problem of identifying and localizing copy-move manipulation. In [35], the authors used an interesting segmentation based approach to detect copy move forgeries. They first divided an image into semantically independent patches and then performed keypoint matching among these patches. In [20], a patch-match algorithm is used to efficiently compute an approximate nearest neighbor field over an image. They further use invariant features such as Circular Harmonic transforms and show robustness over duplicated blocks that have undergone geometrical transformations.

In [45], an image splicing technique has been proposed using visual artifacts. A novel image forgery detection method is presented in [46] based on the steerable pyramid transform (SPT) and the local binary pattern (LBP). The paper [28] includes the recent advances in image manipulation and discusses the process of restoring missing or damaged areas in an image. In [6], the authors review the different image forgery detection techniques in image forensic literature. However, in computer vision, there has been a growing interest to detect image manipulation by applying different computer vision and machine learning algorithms.

Many methods have been proposed to detect seam carving [53, 25, 39] and inpainting based object removal [58, 18, 37]. Several approaches exploit JPEG blocking artifacts to detect tampered regions [38, 21, 42, 12, 13]. In computer vision, deep learning shows outstanding performance in different visual recognition tasks such as image classification [62], and semantic segmentation [40]. In [40], two fully convolution layers have been exploited to segment different objects in an image. The segmentation task has been further improved in [60, 7]. These models extract hierarchical features to represent the visual concept, which is useful in object segmentation. Since, the manipulation does not exhibit any visual change with respect to genuine images, these models do not perform well in segmenting manipulated regions.

Recent efforts, including [9, 10, 50, 15] in the manipulation detection task, exploit deep learning based models. These tasks include detection of generic manipulations [9, 10], resampling [11], splicing [50], and bootleg [14]. In [49], the authors propose Gaussian-Neuron CNN (GNCNN) for steganalysis. A deep learning approach to identify facial retouching was proposed in [1]. In [59], image region forgery detection has been performed using stacked auto-encoder model. In [9], a new form of convolutional layer is proposed to learn the manipulated features from an image. Unlike most of the deep learning based image tampering detection methods which use convolution layers, we present an unique network exploiting convolution layers along with an LSTM network.

3. Network Architecture Overview

Image manipulation techniques such as copy-clone, splicing, and removal are very common as they are very difficult to authenticate due to their resemblance to its genuine images. The main goal of this work is to recognize these manipulations at pixel and patch-level. Localization of manipulated regions is a different problem than object segmentation as tampered regions are not visually apparent. For example, if an object is removed, the region may visually blend into the background, but needs to be identified as manipulated. As another example, copy-move is a kind of manipulation where one object is copied to another region of the same image leading to two similar objects, one originally present, and another manipulated. However, only the latter needs to be identified.

Fig. 3 shows the boundary region of manipulated and non-manipulated block in a patch. From Fig. 3, we can see that boundary regions of the manipulated patches are affected, e.g. smoother boundary, when compared to non-manipulated regions. When we zoom into the small cropped region as shown in Fig. 3, we can see the difference between boundary of manipulated block (smoothed) and non-manipulated region. The boundary shared between non-manipulated and manipulated regions are sometimes intentionally made smoother so that no one can visually understand the artefacts seeing an image. Next, we will discuss the details of our proposed architecture in order to recognize and localize manipulated regions.

3.1. Model for Localizing Manipulated Regions

Here, we perform two tasks-(1) patch classification (manipulated vs non-manipulated), and (2) segmentation of manipulated regions from the patches. The proposed framework is shown in Fig. 2. The network exploits convolutional layers along with an LSTM network to classify patches, and to segment manipulated regions.

3.1.1 Convolutional Layers

Convolutional layers consist of different filters which have learnable weights and biases. In the first layer, the network will take a patch as input. Each patch has R,G,B value with dimension of $64 \times 64 \times 3$ (width, height, color channels). In [61], it is shown that convolutional layers are capable of extracting different features from an image such as edges, textures, objects, and scenes. As discussed above, manipulation is better captured in the boundary of manipulated regions. Thus, the low-level features are critical to identify manipulated regions. The filters in convolutional layer will create feature maps that are connected to the local region of the previous layer. In the convolutional layers, we use kernel size of $5 \times 5 \times \mathcal{D}$, where $\mathcal{D}$ is the depth of a filter. $\mathcal{D}$ has different values for different layers in the network. An element-wise activation is also utilized in the form of RELU function, $max(0, x)$.

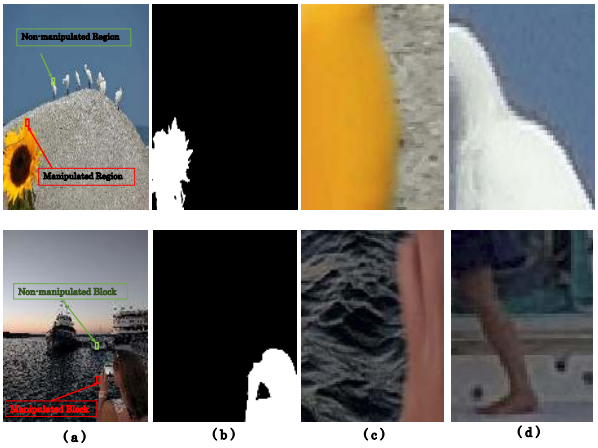


Figure 3. The figure illustrates the boundary region of manipulated block (red) and non-manipulated block (green) in column (a). Column (b) shows the corresponding ground-truth masks for the manipulated images in column (a). Columns (c) and (d) are the zoomed-in version of the red (manipulated) and green (non-manipulated) blocks respectively, showed in (a). Here, we can see that the boundary formation is different for non-manipulated (sharp) and manipulated (smooth) regions.

The first convolution layer creates 16 feature maps. Then, these feature maps are combined in the next convolution layer. We keep one feature map which will be divided into blocks to send into the LSTM cells. The reason for using one feature map is to reduce the network complexity, but it could be changed depending on the size of the dataset. The feature map has been divided into 8 by 8 blocks, which are taken as input the LSTM cells. In Fig. 2, we can see that second convolutional layer provides a two-dimensional feature map which can be denoted as $\mathcal{F}_{c_2}$. The 8 by 8 block of this feature map will be fed into the LSTM cells in order to learn the boundary transformation, which will be discussed in the Section 3.1.2.

The output feature from the LSTM network is used as input to the later convolutional layers. These convolutional layers learn the mapping between features of the boundary transformation from the LSTM and the tampered pixels using the ground-truth mask. Unlike conventional CNNs, we do not use pooling mechanism in every convolution layer as it causes possible loss of information. We only use max pooling in third convolution layer.

Motivated by the segmentation work presented in [40], we also utilize two fully convolution layers (conv layer 4 and 5 as shown in Fig. 2) at the end. In [55, 40], segmentation networks represent features coarsely, which is finally compensated by upsampling operation to match the dimension of the ground-truth mask. However, in contrast to these approaches, we do not follow upsampling operation as it might create additional distortion. In our network, the size

of the feature map produced in different layers never goes below the input patch size. Finally, after the last convolutional layer, we get $64 \times 64 \times 2$ dimensional confidence map of each pixel using the softmax function.

3.1.2 Long-Short Term Memory (LSTM) Network

In computer vision, an LSTM network has been used in tasks where sequential information (context) can be exploited. In [48, 16], LSTMs are used to capture the dependency among a series of pixels. Manipulation distorts the natural statistics of an image, especially in the boundary region as shown in Fig. 3. In this paper, the LSTM network is used to learn the correlation between the blocks in 2D feature map provided by the second convolutional layer as shown in Fig. 2. In order to utilize an LSTM, we first divide the feature map $\mathcal{F}_{c_2}$ obtained from the second convolutional layer into blocks. We split this into 8×8 blocks. Now, we learn the logarithmic distance block dependency by feeding each block to each cell of the LSTM in a sequential manner. The LSTM cells correlate neighboring blocks with current block. In this work, we utilize 3 stacked layers, and at each layer, 64 cells are used. In the last layer, each cell provides 256 dimensional feature vector which is converted to a 16×16 block. Finally, we concatenate all the blocks to represent a 2D feature map F_{lstm} in the same order as we divided them, which is then used by the third convolutional layer. The key insight of using LSTM is *to learn the boundary transformation between different blocks, which provides discriminative features between manipulated and non-manipulated regions*.

In the LSTM, information flow between the cells is controlled by three gates- (1) input gate, (2) forget gate, and (3) output gate. Each gate has a value ranging from zero to one, activated by a sigmoid function. Let us denote cell state and output state as C_t and z_t for current cell t. Each cell produces new candidate cell state $\bar{\mathcal{C}}_t$. Using the previous cell state $\mathcal{C}_{t-1}$ and $\bar{C}_t$, we can write the updated cell state $\mathcal{C}_t$ as

$$\mathcal{C}_t = f_t \circ \mathcal{C}_{t-1} + i_t \circ \bar{\mathcal{C}}_t \tag{1}$$

Here, $\circ$ denotes the $pointwise$ multiplication. Finally, we obtain the output of the current cell h_t, which can be represented as

$$z_t = o_t \circ tanh(\mathcal{C}_t) \tag{2}$$

In Eqns. 1 and 2, i, f, o represent input, forget and output gates.

3.1.3 Training the Network

Soft-max Layers. In the proposed network, we have two softmax layers for the two tasks- patch classification and segmentation (pixel-wise classification). Let us denote the probability distribution over various classes as $P(\mathcal{Y}_k)$ which is provided by softmax classifier. Now, we can predict label by maximizing $P(\mathcal{Y}_k)$ with respect to k. The predicted label can be obtained by $\hat{\mathcal{Y}} = \arg\max_k \ P(\mathcal{Y}_k)$.

Training Loss. In patch classification, patch labels are predicted at the end of the LSTM network as shown in Fig. 2. Let us denote θ_p, which is a weight vector associated with patch classification. We use cross entropy loss for patch classification, which can be written as follows.

$$\mathcal{L}_p(\theta_p) = -\frac{1}{M_p} \sum_{j=1}^{M_p} \sum_{k=1}^{N_p} \mathbb{1}(\mathcal{Y}^j = k) \log(\mathcal{Y}^j = k | x^j; \theta_p) \tag{3}$$

Here, $\mathbb{1}(.)$ is an *indicator function*, which equals to 1 if $j = k$, otherwise it equals 0. $\mathcal{Y}^j$ and x^j imply the patch label (manipulated or non-manipulated) and the feature of the sample j. M_p is the number of patches.

Similarly, we can also compute a loss function for pixel-wise classification. Let θ_s be the parameter vector corresponding to segmentation task. So, the cross entropy loss can be computed as

$$\mathcal{L}_s(\theta_s) = -\frac{1}{M_s} \sum_{m=1}^{M_s} \sum_{n=1}^{N_s} \mathbb{1}(\mathcal{Y}^m = n) \log(\mathcal{Y}^m = n | y^m; \theta_s) \tag{4}$$

Here, M_s and N_s denote the total number of pixels, and the number of class. y represents the input pixel. Now, we can compute the joint loss from Eqns. 3 and 4, which can be written as

$$\mathcal{L}(\theta) = \mathcal{L}_p(\theta_p) + \mathcal{L}_s(\theta_s) \tag{5}$$

Here, θ contains all the parameters involving in both patch classication and segmentation tasks. We use *adaptive moment estimation (Adam)* [32] optimization technique in order to minimize the total loss of the network, shown in Eqn. 5. After optimizing the loss function with all the examples, we learn the optimal set of parameters of the network. With these optimal parameters, the network is able to predict patch labels as well as pixel-wise classification within the patch given a test sample.

4. Experiments

In this section, we demonstrate our experimental results for two tasks-(1) identification of tampered patch, and (2) segmentation of manipulated regions given a patch. We evaluate our proposed model on three datasets- NIST [3], IEEE Forensics Challenge [2], COVERAGE [57].

Data Preparation. We train our models for patch classification and segmentation jointly. We choose three datasets which provide ground-truth mask for manipulated region. NIST [3] is a very challenging dataset, which includes mainly three types of manipulation- (a) copy-clone, (b) removal, and (c) splicing. This recently released dataset includes images, which are tampered in a sophisticated way to

beat current state-of-the-art detection techniques. We also show our results on IEEE Forensics Challenge [2] and COVERAGE [57] datasets which provide ground-truth mask for manipulation. As manipulated regions are small in number compared to non-manipulated regions, we also perform data augmentation in order to get rid of bias in training. As the number of images in COVERAGE [57] is small, we finetune the model trained on NIST [3].

In data preparation, we first split the whole image dataset into three subsets- training (65%), validation (10%) and testing (25%). These subsets are chosen randomly. Then, we extract patches from training images which are considered as training set. Similarly, we obtain validation and testing set in this process. To augment the manipulated patches, we first obtain the bounding boxes using contour approximation method on ground-truth mask. We enlarge the box by placing the whole manipulated region at the center to include more non-manipulated pixels in the patch. We also consider slided version of these bounding box, which includes some part of the manipulated region. These patches are useful for the LSTM to learn the transformation from the manipulated region to non-manipulated region.

We use intersection over union (IoU) ratio between ground-truth mask and a selected region to label a patch. If a patch contains more than $12.5\%(1/8)$ of the manipulated pixels, we label it as manipulated. Labeling the patches in this manner helps in learning the segmentation task better which will be discussed later in this section. As image and ground-truth mask are of same size, we can easily generate the ground-truth masks for the extracted image patches. With these newly generated ground-truth masks and patches, we train the whole network end-to-end.

Implementation Details. To train the model, we use TensorFlow to define different layers of the network. To run the experiment, we utilize multi-GPU setting. We use two NVIDIA Tesla $K80$ GPUs in different sets of experiments.

Post-processing of Segmentation. Given a patch, our network provides binary labels for patch and binary mask for segmentation. To smooth the predicted binary mask, we use Gaussian filtering technique. Next, we multiply the predicted binary mask by predicted patch label. Patch classification reduces false positive pixels in the segmentation at this stage. For example, in a non-manipulated patch, few pixels are falsely positive, but at the same time we know that the patch is non-manipulated with high confidence. We can then remove those pixels from the localization results.

Evaluation Criterion. We ran a set of experiments to evaluate our model. They are (1) performance of joint learning, (2) comparison against other approaches, (3) performance with different layers and with different sizes of feature maps, (4) effect of labeling a patch on segmentation task, and (5) ROC curve and qualitative analysis.

Baseline Methods: We will compare our proposed approach with various baseline methods. All the baseline methods have been implemented for different tasks- segmentation (S), patch classification (P) and joint task learning of both (J). The various baseline methods are described below.

◇ *FCN* : Fully convolutional network as proposed in [40].

◇ *S-LSTM-Conv*$_{3,4}$: Segmentation network using LSTM followed by two convolution layers (conv3 and conv4). First two convolution layers have been removed.

◇ *S-LSTM-Conv*: Segmentation network using LSTM followed by three convolution layers (conv3, conv4 and conv5). First two convolution layers are not used.

◇ *S-Conv-LSTM-Conv*: Proposed network as shown in Fig. 2 for segmentation task only. Patch classification is not considered.

◇ *P-LSTM*: Patch classification using LSTM. No convolution layer is utilized.

◇ *P-Conv-LSTM*: Patch classification using convolution layers (conv1 and conv2) followed by LSTM.

◇ *J-Conv-LSTM-Conv*: Joint classification network as shown in Fig. 2 for patch classification and segmentation.

Methods	NIST [3]	IEEE [2]	COV [57]
P-LSTM	81.05%	84.27%	73.28%
P-Conv-LSTM	86.76%	86.05%	79.12%
J-Conv-LSTM	89.38%	87.68%	80.06%

Table 1. Classification accuracy of patches (Manipulated vs Non-manipulated)

Performance of Joint Learning. We evaluate our proposed model by training on three datasets - NIST [3], IEEE Forensics Challenge [2], Coverage [57] datasets. Tables 1 and 2 show classification accuracy on patch classification and segmentation tasks. From the tables, we can observe that joint-task learning performs better than single-task learning due to strong correlation between patch classification and segmentation. Recognition on one task helps in recognition of other task. From Table. 1, we can observe that the recognition accuracy of J-Conv-LSTM-Conv (joint-task learning) exceeds the P-Conv-LSTM (single-task learning) by **2.62**$\%$, **1.65**$\%$, **0.94**$\%$ on NIST [3], IEEE Forensics Challenge [2] and Coverage [57] datasets respectively. Similarly, for segmentation of manipulted regions, joint task improves the recognition accuracy by significant margin (**1.82**$\%$, **3.35**$\%$, **2.72**$\%$ on NIST [3], IEEE Forensics Challenge [2] and Coverage [57], respectively) when compared to single-task network.

Comparison against Other Approaches. We compare our network with different baseline methods and other state-of-the-art approaches. One of the recent deep learning based approaches for semantic segmentation is fully convolutional network (FCN) as proposed in [40]. We have implemented FCN with five layers (3 convolution and 2 fully con-

Methods	NIST [3]	IEEE [2]	COV [57]
FCN [40]	64.34%	-	-
S-LSTM-Conv$_{3,4}$	74.43%	-	-
S-LSTM-Conv	78.60%	74.04%	75.38%
S-Conv-LSTM-Conv	82.78%	74.32%	78.42%
J-Conv-LSTM-Conv	84.60%	77.67%	81.14%

Table 2. Segmentation accuracy on three datasets

Methods	Patch Classification	Segmentation
Conv1-8f	82.90%	80.88%
Conv1-32f	78.03%	73.17%
LSTM-64h	80.56%	76.16%

Table 3. Performance of the proposed network with varying size of feature maps in different layer.

volution layers) on NIST [3] dataset. To avoid over-fitting, we reduce the number of feature maps in each convolution layer. From Table. 2, we can see that FCN does not perform well to learn the manipulated regions. It is because the model tries to learn the visual concept/feature from an image whereas manipulation of an image does not leave any visual clue. We also compare against other baselines as shown in Table 1 and 2. From the tables, we can see that the proposed network outperforms other baselines in performing both tasks.

Performance of Segmentation Network with Different Layers. We run an experiment by modifying the network with different convolution layers on NIST [3] dataset. This study basically tells us how the convolution layers help in segmenting task. Table. 2 shows the layer-wise analysis of the proposed network. We evaluate the performance of baseline networks- S-LSTM-Conv$_{3,4}$, S-LSTM-Conv as discussed above. As we can see in Table. 2, we obtain higher accuracy with *S-Conv-LSTM-Conv* in segmentation by adding two convolution layers at the front of S-LSTM-Conv network.

Study of Feature Map. In our proposed network, we use first convolution layer(conv1) with 16 feature maps. We also try with varying number of feature maps such as (1) *Conv1-8f*: conv1 with 8 maps, (2) *Conv1-32f*: conv1 layer with 32 feature maps. In the LSTM, each cell produces 256 dimensional feature vector at the third-stacked layer to achieve 128 by 128 map, which is used as input to conv3 layer (please see Sec. 3.1.2 for details). We vary the size of this feature to 64 in *LSTM-64h*, which generates 64 by 64 feature map before conv3 layer. Results of all these network have been shown in Table. 3.

Effect of Labeling a Patch on Segmentation Task. In contrast to object detection method where IoU overlap with more than 50% is considered as correct detection, we consider the IoU overlap with 12.5% in order to label the patch as manipulated. Since we aim to localize manipulated regions, IoU ratio is critical for segmenting a region. Even though higher IoU ratio results in higher accuracy in tamper patch classification, it degrades the performance in segmentation task. Let us consider a patch which has 20% of the pixels manipulated. The network will label this patch as non-manipulated (with IoU 50% labeling), but we still have to segment these 20% pixels from the patch in segmentation task. Due to sharing of significant portion of the network (2 convolution layers along with the LSTM network) between these two tasks, labeling with higher IoU ratio directly affects the segmentation task. Table. 4 shows the results of joint-task learning on NIST [3] dataset for varying IoU ratio.

Methods	Patch Classification	Segmentation
IoU-0.5	96.56%	75.32%
IoU-0.125	89.38%	84.60%

Table 4. The effect of labeling a patch with different IoU ratio on patch classification and segmentation.

ROC Curve of Recognition. Figs. 4(a,b,c) show the ROC plots of the two tasks- patch classification and segmentation, on NIST [3], IEEE Forensics Challenge [2], Coverage [57] datasets respectively. We also provide the area under the curve (AUC) results in Table 5. We compare our segmentation approach with other state-of-the art approaches on NIST [3] in terms of AUC as illustrated in Table 6. From this table, we can see that our network outperforms state-of-the-art approaches by very large margin.

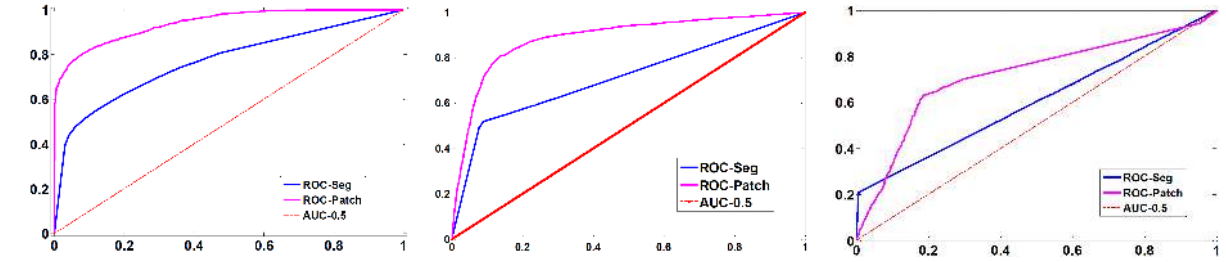
Dataset	Patch Classification	Segmentation
NIST [3]	0.9390	0.7641
Forensic [2]	0.8938	0.7238
Coverage [57]	0.7238	0.6137

Table 5. Area under the curve (AUC) for the ROC plots as shown in Fig. 4

Methods	AUC score
DCT Histograms [38]	0.545
ADJPEG [13]	0.5891
NADJPEG [13]	0.6567
PatchMatch [20]	0.6513
Error level analysis [42]	0.4288
Block Features [36]	0.4785
Noise Inconsistencies [44]	0.4874
Our approach	0.7641

Table 6. Comparison of AUC on NIST 2016 dataset

Qualitative Analysis of Segmentation. In Fig. 5, we provide some examples showing segmentation results produced by the proposed network. The examples are taken



(a) NIST [3] (b) IEEE Forensics Challenge [2] (c) COVERAGE [57]

Figure 4. ROC Curve on three datasets for patch classification and pixel-wise classification (segmentation). AUC is provided in Table 5

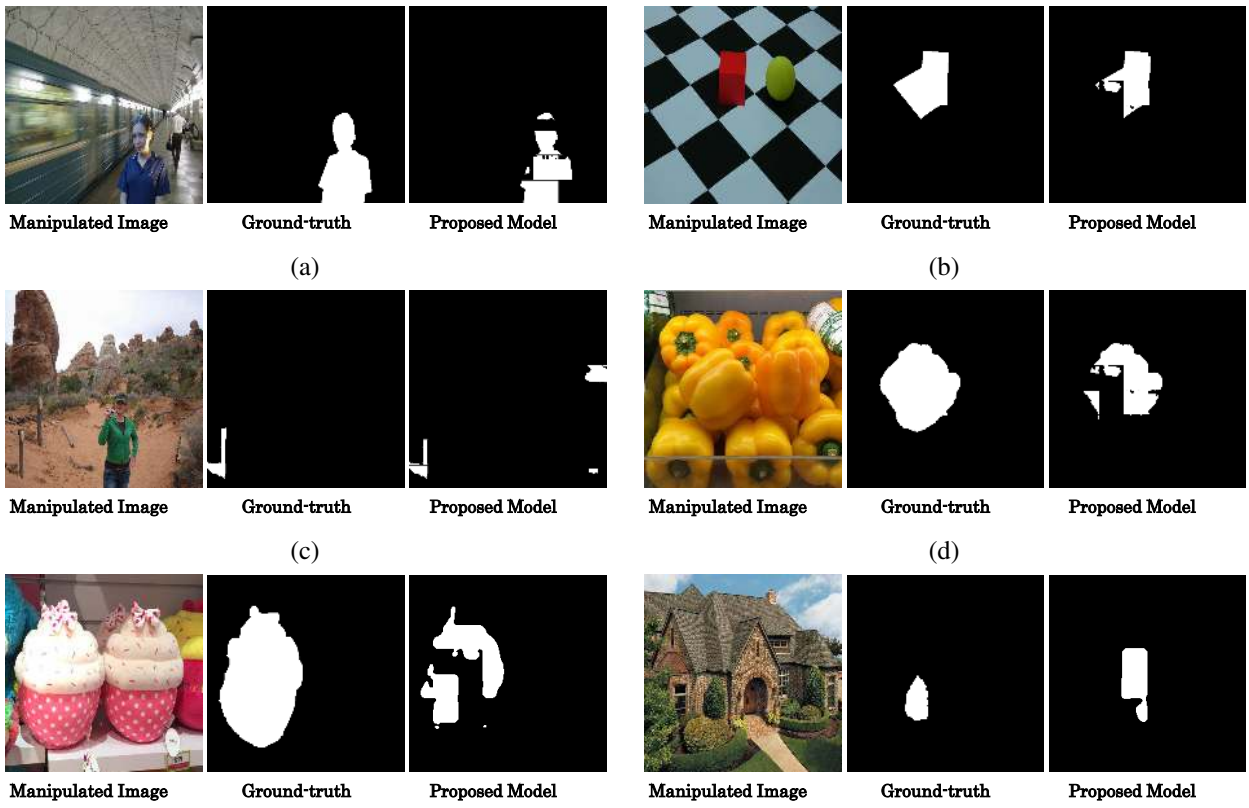


Figure 5. Some of the segmentation examples. Images are taken from three datasets.

from NIST [3], IEEE Forensics Challenge [2], Coverage [57] datasets. The manipulation examples are generated by copy-clone, splicing and removal techniques. The segmentation results are generated after post-processing as discussed before. As we can see from the Fig. 5, the predicted mask can locate different types of manipulation from an image. More such examples will be provided in supplementary material.

5. Conclusion

In this paper, we present a unified framework for joint patch classification and segmentation to localize manipulated regions from an image. We exploit the interdependency between patch recognition and segmentation in order to improve both recognition tasks. Our detailed experiments showed that our approach could efficiently segment various types of manipulations including copy-move, object removal and splicing.

6. Acknowledgement

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. The paper is approved for public release, distribution unlimited.

References

[1] Detecting facial retouching using supervised deep learning.

[2] IEEE IFS-TC Image Forensics Challenge Dataset. http://ifc.recod.ic.unicamp.br/fc.website/index.py.

[3] NIST Nimble 2016 Datasets. https://www.nist.gov/sites/default/files/documents/2016/11/30/should_i_believe_or_not.pdf.

[4] O. M. Al-Qershi and B. E. Khoo. Passive detection of copy-move forgery in digital images: State-of-the-art. *Forensic science international*, 231(1):284–295, 2013.

[5] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra. A sift-based forensic method for copy–move attack detection and transformation recovery. *IEEE Transactions on Information Forensics and Security*, 6(3):1099–1110, 2011.

[6] M. D. Ansari, S. P. Ghrera, and V. Tyagi. Pixel-based image forgery detection: A review. *IETE journal of education*, 55(1):40–46, 2014.

[7] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for scene segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[8] J. H. Bappy and A. K. Roy-Chowdhury. CNN based region proposals for efficient object detection. In *IEEE International Conference on Image Processing (ICIP)*, 2016.

[9] B. Bayar and M. C. Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pages 5–10, 2016.

[10] B. Bayar and M. C. Stamm. Design principles of convolutional neural networks for multimedia forensics. In *IS&T International Symposium on Electronic Imaging: Media Watermarking, Security, and Forensics*, 2017.

[11] B. Bayar and M. C. Stamm. On the robustness of constrained convolutional neural networks to jpeg post-compression for image resampling detection. In *Proceedings of The 42nd IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

[12] T. Bianchi, A. De Rosa, and A. Piva. Improved dct coefficient analysis for forgery localization in jpeg images. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

[13] T. Bianchi and A. Piva. Image forgery localization via block-grained analysis of jpeg artifacts. *IEEE Transactions on Information Forensics and Security*, 7(3):1003–1017, 2012.

[14] M. Buccoli, P. Bestagini, M. Zanoni, A. Sarti, and S. Tubaro. Unsupervised feature learning for bootleg detection using deep learning architectures. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2014.

[15] J. Bunk, J. H. Bappy, T. M. Mohammed, L. Nataraj, A. Flenner, B. Manjunath, S. Chandrasekaran, A. K. Roy-Chowdhury, and L. Peterson. Detection and localization of image forgeries using resampling features and deep learning. *arXiv preprint arXiv:1707.00433*, 2017.

[16] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[17] Y. Cao, T. Gao, L. Fan, and Q. Yang. A robust detection algorithm for copy-move forgery in digital images. *Forensic science international*, 214(1):33–43, 2012.

[18] I.-C. Chang, J. C. Yu, and C.-C. Chang. A forgery detection algorithm for exemplar-based inpainting images using multi-region relation. *Image and Vision Computing*, 31(1):57–71, 2013.

[19] J. Chen, X. Kang, Y. Liu, and Z. J. Wang. Median filtering forensics based on convolutional neural networks. *IEEE Signal Processing Letters*, 22(11):1849–1853, 2015.

[20] D. Cozzolino, G. Poggi, and L. Verdoliva. Efficient dense-field copy–move forgery detection. *IEEE Transactions on Information Forensics and Security*, 10(11):2284–2297, 2015.

[21] H. Farid. Exposing digital forgeries from jpeg ghosts. *IEEE transactions on information forensics and security*, 4(1):154–160, 2009.

[22] X. Feng, I. J. Cox, and G. Doërr. An energy-based method for the forensic detection of re-sampled images. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2011.

[23] X. Feng, I. J. Cox, and G. Doerr. Normalized energy density-based forensic detection of resampled images. *IEEE Transactions on Multimedia*, 14(3):536–545, 2012.

[24] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva. Image forgery localization via fine-grained analysis of cfa artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5):1566–1577, 2012.

[25] C. Fillion and G. Sharma. Detecting content adaptive scaling of images for forensic applications. In *Media Forensics and Security*, 2010.

[26] R. Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision*, 2015.

[27] S. A. Golestaneh and D. M. Chandler. Algorithm for jpeg artifact reduction via local edge regeneration. *Journal of Electronic Imaging*, 23(1):013018–013018, 2014.

[28] C. Guillemot and O. Le Meur. Image inpainting: Overview and recent advances. *Signal processing magazine*, 31(1):127–144, 2014.

[29] M. F. Hashmi, V. Anand, and A. G. Keskar. Copy-move image forgery detection using an efficient and robust method combining un-decimated wavelet transform and scale invariant feature transform. *AASRI Procedia*, 9:84–91, 2014.

[30] M. Jaberi, G. Bebis, M. Hussain, and G. Muhammad. Accurate and robust localization of duplicated region in copy–move image forgery. *Machine vision and applications*, 25(2):451–475, 2014.

[31] P. Kakar and N. Sudha. Exposing postprocessed copy–paste forgeries through transform-invariant features. *IEEE Transactions on Information Forensics and Security*, 7(3):1018–1028, 2012.

[32] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[33] Y. Kwon, K. I. Kim, J. Tompkin, J. H. Kim, and C. Theobalt. Efficient learning of image super-resolution and compression

artifact removal with semi-local gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1792–1805, 2015.

[34] G. Li, Q. Wu, D. Tu, and S. Sun. A sorted neighborhood approach for detecting duplicated regions in image forgeries based on dwt and svd. In *IEEE International Conference on Multimedia and Expo*, 2007.

[35] J. Li, X. Li, B. Yang, and X. Sun. Segmentation-based image copy-move forgery detection scheme. *IEEE Transactions on Information Forensics and Security*, 10(3):507–518, 2015.

[36] W. Li, Y. Yuan, and N. Yu. Passive detection of doctored jpeg image via block artifact grid extraction. *Signal Processing*, 89(9):1821–1829, 2009.

[37] Z. Liang, G. Yang, X. Ding, and L. Li. An efficient forgery detection algorithm for object removal by exemplar-based image inpainting. *Journal of Visual Communication and Image Representation*, 30:75–85, 2015.

[38] Z. Lin, J. He, X. Tang, and C.-K. Tang. Fast, automatic and fine-grained tampered jpeg image detection via dct coefficient analysis. *Pattern Recognition*, 42(11):2492–2501, 2009.

[39] Q. Liu and Z. Chen. Improved approaches with calibrated neighboring joint density to steganalysis and seam-carved forgery detection in jpeg images. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(4):63, 2015.

[40] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[41] W. Luo, J. Huang, and G. Qiu. Robust detection of region-duplication forgery in digital image. In *18th International Conference on Pattern Recognition*, 2006.

[42] W. Luo, J. Huang, and G. Qiu. Jpeg error analysis and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security*, 5(3):480–491, 2010.

[43] B. Mahdian and S. Saic. Detection of copy–move forgery using a method based on blur moment invariants. *Forensic science international*, 171(2):180–189, 2007.

[44] B. Mahdian and S. Saic. Using noise inconsistencies for blind image forensics. *Image and Vision Computing*, 27(10):1497–1503, 2009.

[45] V. Manu and B. Mehtre. Visual artifacts based image splicing detection in uncompressed images. In *IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*, 2015.

[46] G. Muhammad, M. H. Al-Hammadi, M. Hussain, and G. Bebis. Image forgery detection using steerable pyramid transform and local binary pattern. *Machine Vision and Applications*, 25(4):985–995, 2014.

[47] L. Nataraj, A. Sarkar, and B. S. Manjunath. Improving resampling detection by adding noise. In *SPIE, Media Forensics and Security*, volume 7541, 2010.

[48] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *International Conference on Machine Learning*, 2014.

[49] Y. Qian, J. Dong, W. Wang, and T. Tan. Deep learning for steganalysis via convolutional neural networks. In *SPIE/IS&T Electronic Imaging*, 2015.

[50] Y. Rao and J. Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2016.

[51] S. Ryu, M. Kirchner, M. Lee, and H. Lee. Rotation invariant localization of duplicated image regions based on zernike moments. *IEEE Transactions on Information Forensics and Security*, 8(8):1355–1370, 2013.

[52] S.-J. Ryu and H.-K. Lee. Estimation of linear transformation by analyzing the periodicity of interpolation. *Pattern Recognition Letters*, 36:89–99, 2014.

[53] A. Sarkar, L. Nataraj, and B. S. Manjunath. Detection of seam carving and localization of seam insertions in digital images. In *Proceedings of the 11th ACM workshop on Multimedia and security*, 2009.

[54] L. Verdoliva, D. Cozzolino, and G. Poggi. A feature-based approach for image tampering detection and localization. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2014.

[55] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, and A. Courville. Reseg: A recurrent neural network-based model for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016.

[56] W. Wang, J. Dong, and T. Tan. Exploring dct coefficient quantization effects for local tampering detection. *IEEE Transactions on Information Forensics and Security*, 9(10):1653–1666, 2014.

[57] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler. Coverage - a novel database for copy-move forgery detection. In *IEEE International Conference on Image processing (ICIP)*, 2016.

[58] Q. Wu, S. Sun, W. Zhu, G. Li, and D. Tu. Detection of digital doctoring in exemplar-based inpainted images. In *International Conference on Machine Learning and Cybernetics*, 2008.

[59] Y. Zhang, L. L. Win, J. Goh, and V. L. Thing. Image region forgery detection: A deep learning approach. In *Proceedings of the Singapore Cyber-Security Conference (SG-CRC)*, 2016.

[60] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *IEEE International Conference on Computer Vision*, 2015.

[61] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene CNNs. *International Conference on Learning Representations*, 2015.

[62] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, 2014.