Generate, Segment, and Refine: Towards Generic Manipulation Segmentation

Peng Zhou,[1] **Bor-Chun Chen,**[1] **Xintong Han,**[2] **Mahyar Najibi,**[1]
Abhinav Shrivastava,[1] **Ser Nam Lim,**[3] **Larry S. Davis**[1]
[1]University of Maryland, College Park, [2]Huya Inc, [3]Facebook
{pengzhou, sirius}@umd.edu, hanxintong@huya.com, {najibi, abhinav, lsd}@cs.umd.edu, sernamlim@fb.com

Abstract

Detecting manipulated images has become a significant emerging challenge. The advent of image sharing platforms and the easy availability of advanced photo editing software have resulted in a large quantities of manipulated images being shared on the internet. While the intent behind such manipulations varies widely, concerns on the spread of false news and misinformation is growing. Current state of the art methods for detecting these manipulated images suffers from the lack of training data due to the laborious labeling process. We address this problem in this paper, for which we introduce a manipulated image generation process that creates true positives using currently available datasets. Drawing from traditional work on image blending, we propose a novel generator for creating such examples. In addition, we also propose to further create examples that force the algorithm to focus on boundary artifacts during training. Strong experimental results validate our proposal.

Introduction

Manipulated photos are becoming ubiquitous on social media due to the availability of advanced editing software, including powerful generative adversarial models (Isola et al. 2017; Yeh et al. 2017). While such images have been created for a variety of purposes, including memes, satires, etc., there are growing concerns on the abuse of manipulated images to spread fake news and misinformation. To this end, a variety of solutions have been developed towards detecting such manipulated images.

While a number of proposed solutions posed the problem as a classification task (Cozzolino et al. 2018; Zhou et al. 2017), where the goal is to classify whether a given image has been tampered with, there is great utility for solutions that are capable of detecting manipulated regions in a given image (Huh et al. 2018; Zhou et al. 2017; Park et al. 2018; Salloum, Ren, and Kuo 2018). In this paper, we similarly treat this problem as a semantic segmentation task and adapt GANs (Goodfellow et al. 2014) to generate samples to alleviate the lack of training data. The lack of training data has been an ongoing problem for training models to detect manipulated images. Scouring the internet for "real" tampered images (Moreira et al. 2018) is a laborious process that often leads to over-fitting in the training process. Alternatively, one could employ a self-supervised process, where detected objects in one image are spliced onto another, with the caveat that such a process often results in training images

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Figure 1: Examples of manipulated images across different datasets. Columns from left to right are images in CASIA (Dong, Wang, and Tan 2013), COVER (Wen et al. 2016), Carvalho (De Carvalho et al. 2013), and In-The-Wild (Huh et al. 2018). The odd rows are manipulated images and the even rows are the ground truth masks. Different datasets contain different distributions (from animals to person), manipulation techniques (from copy-move (the second column) to splicing (the rest columns)) and post-processing methods (from no post-processing to various processes including filtering, illumination, and blurring).

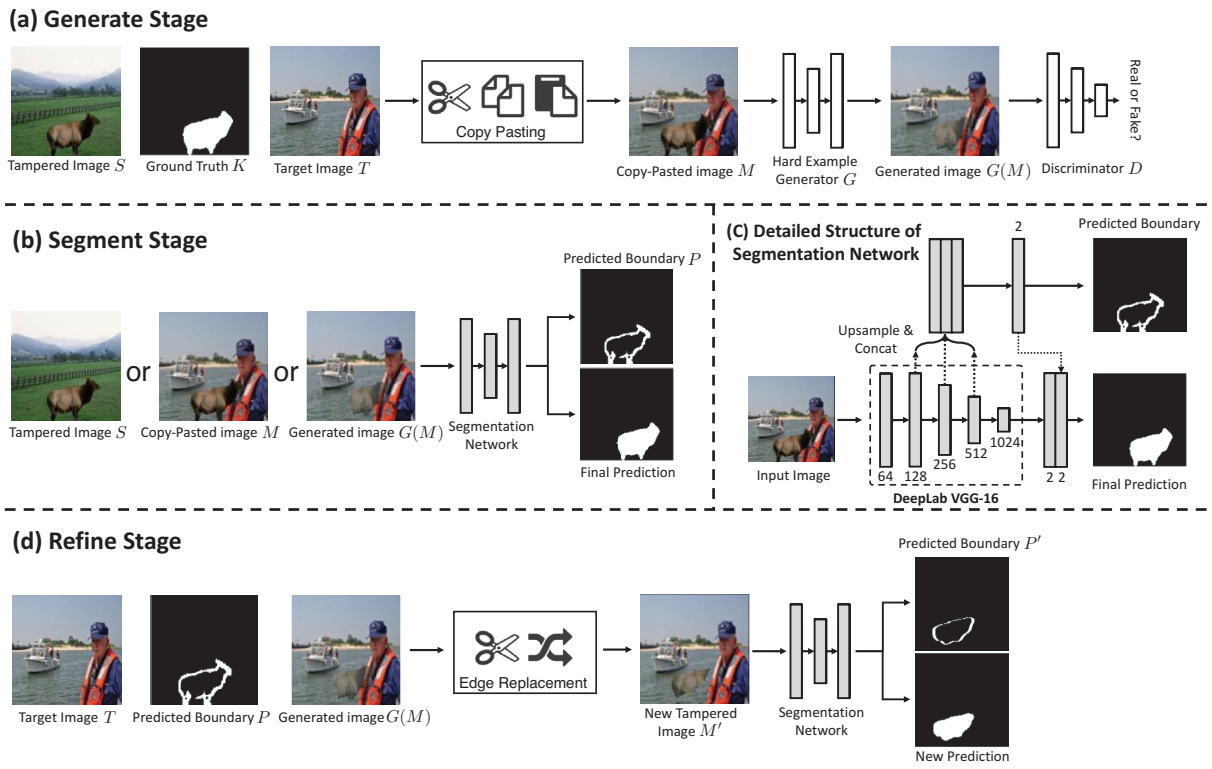


Figure 2: GSR-Net framework overview. **(a)** Given a tampered image S, an authentic target image T, and the ground truth mask K, the generation stage generates hard example $G(M)$ starting from a simple copy-pasting image M. **(b)** Feeding the training images, copy-pasted images or generated images as input, the segmentation stage learns to segment the boundary artifacts and fill the interior to produce the final prediction. **(c)** The segmentation network concatenates lower level features to predict boundary artifacts and then concatenate back the boundary feature to the segmentation branch for final prediction. **(d)** The refinement stage creates a novel tampered image with new boundary artifacts by replacing the predicted manipulated boundaries of segmentation stage with original authentic regions and learns to make a new prediction.

that are not realistic. Of course, the best approach for generating training samples is to employ professional labelers to create realistic looking manipulated images, but this remains a very tedious process. It is therefore not surprising that existing datasets (Huh et al. 2018; Dong, Wang, and Tan 2010; 2013; Wen et al. 2016; De Carvalho et al. 2013) are often not comprehensive enough to train models that generalize well.

Additionally, in contrast to standard semantic image segmentation, correctly segmenting manipulated regions depends more on visual artifacts that are often created at the boundaries of manipulated regions than on semantic content (Bappy et al. 2017; Zhou et al. 2018). Several challenges exist in recognizing these boundary artifacts. First, the space of manipulations is very diverse. One can, for example, do a *copy-move*, which copies and pastes image regions within the same image (the second column in Figure 1) , or *splice*, which copies a region from one image and pastes it to another image (the remaining columns in Figure 1). Second, a variety of post-processing such as compression, blurring, and various color transformations make it harder to detect boundary artifacts caused by tampering. See Figure 1 for some examples. Most existing methods (Huh et al. 2018; Zhou et al. 2018; Park et al. 2018; Salloum, Ren, and Kuo 2018) that utilize discriminative features like image metadata, noise models, or color artifacts due to, for example, Color Filter Array (CFA) inconsistencies, have failed to generalize well for these reasons.

This paper introduces a two-pronged approach to (1) address the lack of comprehensive training data, as well as, (2) focus the training process on learning to recognize boundary artifacts better. We adopt GANs for addressing (1), but instead of relying on prior GAN methods (Isola et al. 2017; Zhu et al. 2017; Karras et al. 2018) that mainly explore image level manipulation, we introduce a novel objective function that optimizes for the realism of the manipulated regions by blending tampered regions in existing datasets to assist segmentation. That is, given an annotated image from an existing dataset, our GAN takes the given annotated regions and optimizes via a blending based objective function to enhance the realism of the regions. Blending has been shown to be effective in creating training images effective for the task of object detection (Dwibedi, Misra, and Hebert 2017), and this forms our main motivation in formulating our GAN.

To address (2), we propose a segmentation and refinement

procedure. The segmentation stage localizes manipulated regions by learning to spot boundary artifacts. To further prevent the network from just focusing on semantic content, the refinement stage replaces the predicted manipulation boundaries with authentic background and feed the new manipulated images back to the segmentation network. We will show empirically that the segmentation and refinement has the effect of focusing the model's attention on boundary artifacts during learning (see Table 2).

We design an architecture called GSR-Net which includes these three components—a generation stage, a segmentation stage and a refinement stage. The architecture of GSR-Net is shown in Figure 2. During training, we alternatively train the generation GAN, followed by the segmentation and refinement stage, which take as input the output of the generation stage as well as images from the training datasets. The additional varieties of manipulation artifacts provided by both the generation and refinement stages produce models that exhibit very good generalization ability. We evaluate GSR-Net on four public benchmarks and show that it performs better to state-of-the-art methods. Experiments with two different post-processing attacks further demonstrate the robustness of GSR-Net. In summary, the contributions of this paper are 1) A framework that augments existing datasets in a way that specifically addresses the main weaknesses of current approaches without requiring new annotations efforts; 2) Introducing a generation stage with a novel objective function based on blending for generating images effective for training models to detect tampered regions; 3) Introducing a novel refinement stage that encourages the learning of boundary artifacts inherent in manipulated regions, which, to the best of our knowledge, no prior work in this field has utilized to help training.

Related Work

Image Manipulation Segmentation. (Park et al. 2018) train a network to find JPEG compression discrepancies between manipulated and authentic regions. (Zhou et al. 2017; 2018) harness noise features to find inconsistencies within a manipulated image. (Huh et al. 2018) treat the problem as anomaly segmentation and use metadata to locate abnormal patches. The features used in these works are based on the assumption that manipulated regions are from a different image, which is not the case in copy-move manipulation. However, our method directly focuses on general artifacts in the RGB channel without specific feature extraction and thus can be applied to copy-move segmentation. More related works from (Salloum, Ren, and Kuo 2018) and (Bappy et al. 2017) show the potential of boundary artifacts in different image manipulation techniques. These methods are sources of motivation for us to exploit boundary artifacts as a strong cue for detecting manipulations. (Bappy et al. 2017) design a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) based network to identify RGB boundary artifacts at both the patch and pixel level. (Salloum, Ren, and Kuo 2018) adopt a Multi-task Fully Convolutional Network (MFCN) to manipulation segmentation by providing both segmentation and edge annotations. Instead of applying hole filling on edge prediction to do late fusion, our segmentation stage early fuses edge information with segmentation branch to improve segmentation results.

GAN Based Image Editing. GAN based image editing approaches have witnessed a rapid emergence and impressive results have been demonstrated recently (Tsai et al. 2017; Lalonde and Efros 2007; Wang et al. 2018; Karras et al. 2018; Zhu et al. 2017). Prior and concurrent works force the output of GAN to be conditioned on input images through extra regression losses (for example, ℓ_2 loss) or discrete labels. However, these methods manipulate the whole images and do not fully explore region based manipulation. In contrast, our GAN manipulates minor regions and fits better for manipulation segmentation where minor regions have been manipulated. A more related work (Tsai et al. 2017) generates natural composite images using both scene parsing and harmonized ground truth. Even though it targets at region manipulation, experimental results show that our method performs better in terms of assisting segmentation.

Adversarial Training. Discriminative feature learning has motivated recent research on adversarial training on several tasks. (Shrivastava et al. 2017) propose a simulated and unsupervised learning approach which utilizes synthetic images to generate realistic images. An online hard negative generation network (Wang, Shrivastava, and Gupta 2017) boosts the performance on occluded and deformed objects. (Wei et al. 2017) investigate an adversarial erasing approach to learn dense and complete semantic segmentation. (Le et al. 2018) propose an adversarial shadow attenuation network to make correct predictions on hard shadow examples. However, their approaches are difficult to adapt to manipulation segmentation because they either generate whole synthetic images or leave artifacts on erased regions. In contrast, we replace manipulated regions with original ones so that the replaced regions become authentic.

Approach

We describe the GSR-net in details in the following sections. A key to the generation is the utilization of a GAN with a loss function central around using blending to optimize for producing realistic training images. The segmentation and refinement stage are specially designed to single out boundaries of the manipulated regions in order to guide the training process to pay extra attention to boundary artifacts.

Generation

Generator. Referring to Figure 2 (a), the generator is given as input both copy-pasted images and ground truth masks. To prepare the input images, we start with the training samples in manipulation datasets (for example, CASIA 2.0 (Dong, Wang, and Tan 2013)). Given a training image S, the corresponding ground truth binary mask K and an authentic target image T from a clean dataset (for example, COCO (Lin et al. 2014)), we first create a simple copy-pasted image M by taking S as foreground and T as background:

$$M = K \odot S + (1 - K) \odot T, \tag{1}$$

where $\odot$ represents pointwise multiplication.

In Poisson blending (Pérez, Gangnet, and Blake 2003), the final value of pixel i in the manipulated regions is

$$b_i = \arg\min_{b_i} \sum_{s_i \in S, \mathcal{N}_i \subset S} ||\nabla b_i - \nabla s_i||_2 + \sum_{s_i \in S, \mathcal{N}_i \not\subset S} ||b_i - t_i||_2, \quad (2)$$

where ∇ denotes the gradient, $\mathcal{N}_i$ is the neighborhood (for example, up, down, left and right) of the pixel at position i, b_i is the pixel in the blended image B, s_i is the pixel in S and t_i is the pixel in T.

Similar to Poisson blending, we optimize the generator to blend neighborhoods in the resulting image that now contains copy-pasted regions and background regions. A key part of our loss function enforces the shapes of the tampered regions, while maintaining the background regions. To maintain background regions, we utilize ℓ_1 loss to reconstruct the background:

$$L_{\text{bg}} = \frac{1}{N_{\text{bg}}} \sum_{t_i \in T, k_i = 0} ||m_i - t_i||_1, \quad (3)$$

where N_{bg} is the total number of pixels in the background, m_i is the pixel in M and k_i is the value in mask K at position i. To maintain the shape of manipulated regions, we apply a Laplacian operator to the pasted regions and reconstruct the gradient of this region to match the source region:

$$L_{\text{grad}} = \frac{1}{N_{\text{fg}}} \sum_{s_i \in S, k_i = 1} ||\Delta m_i - \Delta s_i||_1, \quad (4)$$

where Δ denotes the Laplacian operator and N_{fg} is the total number of pixels in pasted regions. To further constrain the shape of pasted regions, we add an additional edge loss as denoted by

$$L_{\text{edge}} = \frac{1}{N_{\text{edge}}} \sum_{s_i \in S, e_i = 1} ||m_i - s_i||_1, \quad (5)$$

where N_{edge} is the number of boundary pixels and e_i is the value of the edge mask at position i, which is obtained by the absolute difference between a dilation and an erosion on K. To generate realistic manipulated images, we add an adversarial loss L_{adv}, as explained below, that serves to encourage the generator to produce increasingly realistic images as the training progresses.

Discriminator. In our discriminator, a crucial detail to point out is that the manipulated regions are typically occupying only a small area in the image. Hence, it is beneficial to restrict the GAN discriminator's attention to the structure in local images patches. This is reminiscent of "PatchGAN" (Isola et al. 2017) that only penalizes structure at the scale of patches. Similar to PatchGAN, our discriminator applies a final fully convolutional layer at a patch scale of $N \times N$. The discriminator distinguishes the authentic image T as real and the generated image $G(K, M)$ as fake by maximizing:

$$L_{\text{adv}} = \mathbb{E}_T[\log(D(K, T))] + \mathbb{E}_M[1 - \log(D(K, G(K, M)))], \quad (6)$$

where K is concatenated with $G(K, M)$ or T as the input to the discriminator (we do not show K in the discriminator input in Figure 2 (a) for simplicity).

The final loss function of the generator is given as

$$L_G = L_{\text{bg}} + \lambda_{\text{grad}} L_{\text{grad}} + \lambda_{\text{edge}} L_{\text{edge}} + \lambda_{\text{adv}} L_{\text{adv}}, \quad (7)$$

where λ_{grad}, λ_{edge}, and λ_{adv} are parameters which control the importance of the corresponding loss terms. Conditioned on this constraint, the generator preserves background and texture information of pasted regions while blending the manipulated regions with the background, which can be applied to generate both splicing and copy-move examples. Also, it can be potentially utilized to generate removal examples by setting λ_{grad} and λ_{edge} to zero, and thus the generator learns to inpaint the missing regions, creating images with removal manipulation.

Segmentation

For segmentation, we simply adopt the publicly available VGG-16 (Simonyan and Zisserman 2015) based DeepLab model (Chen et al. 2018) to include boundary information. The network structure is depicted in Figure 2 (c), consisting of a boundary branch predicting the manipulated boundaries and a segmentation branch predicting the interior. In particular, to enhance attention on boundary artifacts, we introduce boundary information by subtracting the erosion from the dilation of the binary ground truth mask to obtain the boundary mask. We then predict this boundary mask through concatenating bi-linearly up-sampled intermediate features and passing them to a 1×1 convolutional layer to form the boundary branch. Finally, we concatenate the output features of the boundary branch with the up-sampled features of the segmentation branch. Empirically, we noticed such multi-task learning helps the generalization of the final model. Only the segmentation branch output after boundary feature concatenation is used for evaluation during inference. During training, we select the copy-pasted examples M, generated examples $G(M)$ and training samples S in the dataset as input to the segmentation network which provides a larger variety of manipulation. The loss function of the segmentation network is an average, two class softmax cross entropy loss.

Refinement

The goal of the refinement stage is to draw attention to the boundary artifacts during training, taking into account the fact that boundary artifacts play a more pivotal role than semantic content in detecting manipulations (Bappy et al. 2017; Zhou et al. 2018). While we may be able to employ prior erasing based adversarial mining methods (Wei et al. 2017; Wang, Shrivastava, and Gupta 2017), they are not suitable for our purpose because it will introduce artifacts on the erased regions that should become authentic background. Instead, the refinement stage utilizes the prediction of the segmentation stage to produce new boundary artifacts through replacing with original regions. As illustrated in Figure 2 (d), given an authentic target image T in which the manipulated regions was inserted, the manipulated image M (which

Dataset	Carvalho		In-The-Wild		COVER		CASIA	
Metrics	MCC	F1	MCC	F1	MCC	F1	MCC	F1
NOI (Mahdian and Saic 2009)	0.255	0.343	0.159	0.278	0.172	0.269	0.180	0.263
CFA (Ferrara et al. 2012)	0.164	0.292	0.144	0.270	0.050	0.190	0.108	0.207
MFCN (Salloum, Ren, and Kuo 2018)	0.408	0.480	-	-	-	-	0.520	0.541
RGB-N (Zhou et al. 2018)	0.261	0.383	0.290	0.424	0.334	0.379	0.364	0.408
EXIF-consistency (Huh et al. 2018)*	0.420	0.520	0.415	0.504	0.102	0.276	0.127	0.204
DeepLab (baseline)	0.343	0.420	0.352	0.472	0.304	0.376	0.435	0.474
GSR-Net (ours)	**0.462**	**0.525**	**0.446**	**0.555**	**0.439**	**0.489**	**0.553**	**0.574**

Table 1: MCC and F_1 score comparison on four standard datasets. '-' denotes that the result is not available in the literature. * Our method is 1600 times faster than EXIF-consistency.

could also be the generated image $G(M)$), and the manipulated boundary prediction P by the segmentation stage, we replace the pixels in predicted boundaries by the authentic regions in T and create a novel manipulated image:

$$M' = T \odot P + M \odot (1 - P), \tag{8}$$

where M' is the novel manipulated image with new boundary artifacts. The corresponding segmentation ground truth now becomes

$$K' = K - K \odot P, \tag{9}$$

where K' is the new manipulated mask for M'. The new boundary artifact mask can be extracted in the same way as the previous step. Notice that the refinement stage utilizes the target images T to help training, providing more side information to spot the artifacts. Taking as input the new manipulated images, the same segmentation network in Figure 2 (c) then learns to predict the new manipulated boundaries and interior regions.

In addition to augment boundary artifacts, the refinement stage also mines the hard examples during training. Since the refinement stage is based on predictions from the previous stage, hard examples where the manipulation regions are not predicted remain the same after the replacing operation. As a result, these hard examples weight more during training after feeding back to the segmentation network.

Similar to (Wei et al. 2017), multiple refinement operations are possible and there is a tradeoff between training time and performance. However, the difference is that the segmentation network in the refinement stage shares weights with that in the segmentation stage. The weight sharing enables us to use a single segmentation network at inference. As a result, the network learns to focus more attention on boundary artifacts with no additional cost at inference time.

Experiments

We evaluate the performance of GSR-Net on four public benchmarks and compare it with the state-of-the-art methods. We also analyze its robustness under several attacks.

Datasets and Experiment Setting

Datasets. We evaluate our performance on four datasets — In-The-Wild (Huh et al. 2018), COVER (Wen et al. 2016), CASIA 1.0 (Dong, Wang, and Tan 2010) and Carvalho (De Carvalho et al. 2013).

Evaluation Metrics. We use pixel-level F1 score and MCC as the evaluation metrics when comparing to other approaches. For fair comparison, following the same measurement as (Salloum, Ren, and Kuo 2018; Huh et al. 2018; Zhou et al. 2018), we vary the prediction threshold to get binary prediction mask and report the optimal score over the whole dataset.

Main Results

In this section, We present our results for the task of manipulation segmentation. We fine-tune our model on CASIA 2.0 from the ImageNet pre-trained model and test directly the performance on the aforementioned four datasets. We compare with methods described below:

- **NoI** (Mahdian and Saic 2009): A noise inconsistency method which predicts regions as manipulated where the local noise is inconsistent with authentic regions. We use the released code (Zampoglou, Papadopoulos, and Kompatsiaris 2017) for evaluation.
- **CFA** (Ferrara et al. 2012): A CFA based method which estimates the internal CFA pattern of the camera for every patch in the image and segments out the regions with anomalous CFA features as manipulated regions. The evaluation code is public available (Zampoglou, Papadopoulos, and Kompatsiaris 2017).
- **RGB-N** (Zhou et al. 2018): A two-stream Faster R-CNN based approach which combines features from the RGB and noise channel to make the final prediction. We train the model on CASIA 2.0 using the code provided by the authors.
- **MFCN** (Salloum, Ren, and Kuo 2018): A multi-task FCN based method which harnesses both an edge mask and segmentation mask for manipulation segmentation. Hole filling is applied for the edge branch to make the prediction. The final decision is the intersection of the two branches. We directly report the results from the paper since the code is not publicly available.
- **EXIF-consistency** (Huh et al. 2018): A self-consistency approach which utilizes metadata to learn features useful for manipulation localization. The prediction is made patch by patch and post-processing like mean-shift (Cheng 1995) is used to obtain the pixel-level manipulation prediction. We use the code provided by the authors for evaluation.
- **DeepLab**: Our baseline model which adopts DeepLab VGG-16 model to manipulation segmentation task. No gen-

Dataset	Carvalho	In-the-Wild	COVER	CASIA
DeepLab	0.420	0.472	0.376	0.474
DL + CP	0.446	0.504	0.410	0.503
DL + G	0.460	0.524	0.434	0.506
DL + DIH	0.384	0.421	0.342	0.420
DL + CP + G	0.472	0.528	0.444	0.507
GS-Net	0.515	0.540	0.455	0.545
GSR-Net	0.525	0.555	0.489	0.574

Table 2: Ablation analysis on four datasets. Each entry is the F1 score tested on individual dataset.

eration, boundary branch or refinement stage is added.

• **GSR-Net**: Our full model combining generation, segmentation and refinement for manipulation segmentation.

The final results, presented in Table 1, highlight the advantage of GSR-Net. For supervised methods (Zhou et al. 2018; Salloum, Ren, and Kuo 2018), we train the model on CASIA 2.0 and evaluate on all the four datasets. For other unsupervised methods (Mahdian and Saic 2009; Ferrara et al. 2012; Huh et al. 2018), we directly test the model on all datasets. GSR-Net outperforms other approaches by a large margin on COVER, suggesting the advantage of our network on copy-move manipulation. Also, GSR-Net has an improvement on In-The-Wild, CASIA 1.0 and Carvalho. Additionally, in terms of computation time, EXIF-consistency takes 1600 times more computation (80 seconds for an 800×1200 image on average) than ours (0.05s per image). Compared to boundary artifact based methods, our GSR-Net outperforms MFCN by a large margin, indicating the effectiveness of the generation and refinement stages. In addition to that, no hole filling is required since our approach does not perform late fusion with the boundary branch, but utilizing boundary artifacts to guide the segmentation branch instead.

Our method outperforms the baseline model by a large margin, showing the effectiveness of the proposed generation, segmentation and refinement stages.

Ablation Analysis

We quantitatively analyze the influence of each component in GSR-Net in terms of F1 score.

• **DL + CP**: DeepLab VGG-16 model with just the segmentation output, using simple copy-pasted (no generator) and CASIA 2.0 images during training.

• **DL + G**: DeepLab VGG-16 model with just the segmentation output, using generated and CASIA 2.0 images during training.

• **DL + DIH**: DeepLab VGG-16 model with just the segmentation output, using the images generated from (Tsai et al. 2017) and CASIA 2.0 images during training. We adapt deep image harmonization (DIH) network for the generation stage as it also manipulate regions.

• **DL + CP + G**: DeepLab VGG-16 model with just the segmentation output, using both copy-pasted, generated and CASIA 2.0 images during training.

• **GS-Net**: Generation and segmentation network with boundary artifact guided manipulation segmentation. No refinement stage is incorporated.

Dataset	Carvalho	In-The-Wild	COVER	CASIA
CP+S	0.343	0.430	0.351	0.242
CP+G+S	0.354	0.441	0.355	0.270
CP+GSR	**0.418**	**0.479**	**0.381**	**0.331**

Table 3: F1 score manipulation segmentation comparison trained with COCO annotations.

The results are shown in Table 2. Starting from our baseline model, simply adding copy-pasted images (**DL + CP**) achieves improvement due to broadening the manipulation distribution. In addition, replacing copy-pasted images with generated images (**DL + G**) also shows improvement compared to **DL + CP** on all the datasets as it refines the boundary from naive copy-pasting. As expected, adding both copy-pasted images and generated hard examples (**DL + CP + G**) is more useful because the network has access to a larger distribution of manipulation.

Compared to applying deep harmonization network (**DL + DIH**), our generation approach (**DL + G**) performs better as it aligns well with the natural process of manipulation and has a larger variety of manipulation.

The results also indicate the impact of boundary guided segmentation network. Directly predicting segmentation (**DL + CP + G**) does not explicitly learn manipulation artifacts, and thus has limit generalization ability compared to **GS-Net**, which uses the boundary features as side information. Furthermore, **GSR-Net** boosts the performance on **GS-Net** since the refinement stage introduces new boundary artifacts.

Robustness to Attacks

We apply both JPEG compression and image scaling attacks to test images of In-The-Wild and Carvalho datasets. We compare GSR-Net with RGB-N (Zhou et al. 2018), EXIF-selfconsistency (Huh et al. 2018) using their publicly available code, and MFCN (Salloum, Ren, and Kuo 2018) using the numbers reported in their paper. Figure 3 shows the results, which indicates our approach yields more stable performance than prior methods.

Segmentation with COCO Annotations

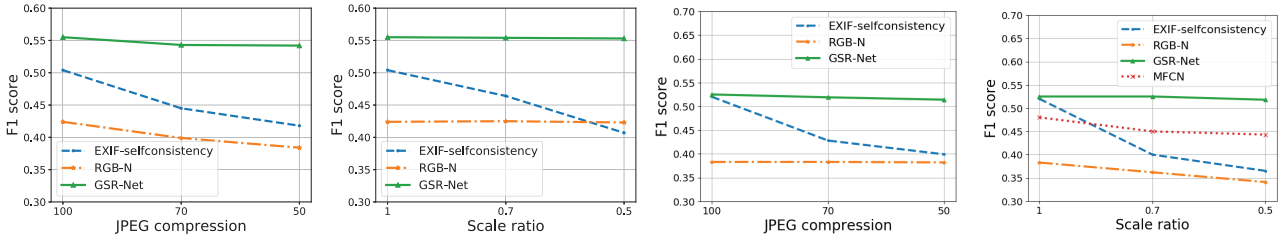
This experiment shows how much gain our model achieves without using the manipulated images in CASIA 2.0. Instead of carefully manipulated training data, we only utilize the object annotations in COCO to create manipulated images. We compare the result of using different training data as follows:

• **CP + S**: Only using copy-pasted images to train the segmentation network.

• **CP + G + S**: Using both copy-pasted and generated images.

• **CP + GSR**: Using copy-pasted images and generated images. The refinement stage is applied.

Results are presented in Table 3. The performance using only copy-pasted images (**CP + S**) on the four datasets indicates that our network truly learns boundary artifacts. Also, the improvement after adding generated images (**CP + G +**



(a) In-The-Wild JPEG attack (b) In-The-Wild scale attack (c) Carvalho JPEG attack (d) Carvalho scale attack

Figure 3: Analysis of robustness under different attacks. Attacks with JPEG compression consists of quality factors of 70 and 50; scale attacks use scaling ratios of 0.7 and 0.5. **(a)** JPEG compression attacks on In-The-Wild. **(b)** Scale attacks on In-The-Wild. **(c)** JPEG compression attacks on Carvalho. **(d)** Scale attacks on Carvalho.

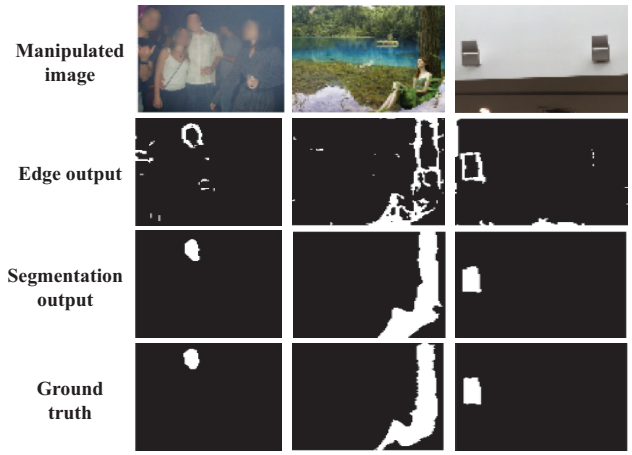


Figure 4: Qualitative visualization. The first row shows manipulated images on different datasets. The second indicates the final manipulation segmentation prediction. The third row illustrates the output of boundary artifacts branch. The last row is the ground truth.

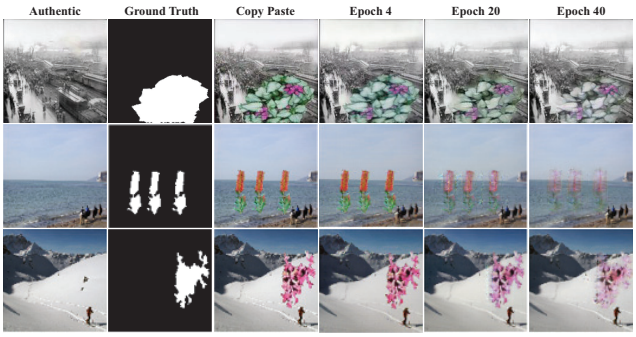


Figure 5: Qualitative visualization of the generation network. The first two columns show the authentic background and manipulation mask. As the number of epochs increases, the manipulated region matches better with the background and thus boundary artifacts are harder to identify.

S) shows that our generation network provides useful manipulation examples that increases generalization. Last, the refinement stage (**CP + GSR**) boosts performance further by encouraging the network to spot new boundary artifacts.

Qualitative Results

Generation Visualization. We illustrate some visualizations of the generation network in Figure 5. It is clear that the generation network learns to match the pasted region with background during training. As a result, the boundary artifacts are becoming subtle and the generation network produces harder examples for the segmentation network.

Segmentation Results. We present qualitative segmentation results on four datasets in Figure 4. Unsurprisingly, the boundary branch outputs the potential boundary artifacts in manipulated images and the other branch fills in the interior based on the predicted manipulated boundaries. The examples indicate that our approach deals well with both splicing and copy-move manipulation based on the manipulation clues from the boundaries.

Conclusion

We propose a novel segmentation framework that firstly utilizes a generation network to enable generalization across variety of manipulations. Starting from copy-pasted examples, the generation network generates harder examples during training. We also design a boundary artifact guided segmentation and refinement network to focus on manipulation artifacts rather than semantic content. Furthermore, the segmentation and refinement stage share the same weights, allowing for much faster inference. Extensive experiments demonstrate the generalization ability and effectiveness of GSR-Net on four standard datasets and show state-of-the-art performance. The manipulation segmentation problem is still far from solved due to the large variation of manipulations and post-processing methods. Including more manipulation techniques in the generation network could potentially boost the generalization ability of the existing model and is part of our future research.

Acknowledgement. We gratefully acknowledge support from Facebook AI and the DARPA MediFor program under cooperative agreement FA87501620191, "Physical and Semantic Integrity Measures for Media Forensics".

References

Bappy, J. H.; Roy-Chowdhury, A. K.; Bunk, J.; Nataraj, L.; and Manjunath, B. 2017. Exploiting spatial structure for localizing manipulated image regions. In *ICCV*.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2018. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. In *TPAMI*.

Cheng, Y. 1995. Mean shift, mode seeking, and clustering. In *TPAMI*.

Cozzolino, D.; Thies, J.; Rössler, A.; Riess, C.; Nießner, M.; and Verdoliva, L. 2018. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*.

De Carvalho, T. J.; Riess, C.; Angelopoulou, E.; Pedrini, H.; and de Rezende Rocha, A. 2013. Exposing digital image forgeries by illumination color classification. In *TIFS*.

Dong, J.; Wang, W.; and Tan, T. 2010. Casia image tampering detection evaluation database 2010. http://forensics.idealtest.org.

Dong, J.; Wang, W.; and Tan, T. 2013. Casia image tampering detection evaluation database. In *ChinaSIP*.

Dwibedi, D.; Misra, I.; and Hebert, M. 2017. Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. In *ICCV*.

Ferrara, P.; Bianchi, T.; De Rosa, A.; and Piva, A. 2012. Image forgery localization via fine-grained analysis of cfa artifacts. In *TIFS*.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NeurIPS*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*.

Huh, M.; Liu, A.; Owens, A.; and Efros, A. A. 2018. Fighting fake news: Image splice detection via learned self-consistency. In *ECCV*.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*.

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2018. Progressive growing of gans for improved quality, stability, and variation. *ICLR*.

Lalonde, J.-F., and Efros, A. A. 2007. Using color compatibility for assessing image realism. In *ICCV*.

Le, H.; Vicente, T. F. Y.; Nguyen, V.; Hoai, M.; and Samaras, D. 2018. A+ d net: Training a shadow detector with adversarial shadow attenuation. In *ECCV*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Mahdian, B., and Saic, S. 2009. Using noise inconsistencies for blind image forensics. In *IMAVIS*.

Moreira, D.; Bharati, A.; Brogan, J.; Pinto, A.; Parowski, M.; Bowyer, K. W.; Flynn, P. J.; Rocha, A.; and Scheirer, W. J. 2018. Image provenance analysis at scale. *arXiv preprint arXiv:1801.06510*.

Park, J.; Cho, D.; Ahn, W.; and Lee, H.-K. 2018. Double jpeg detection in mixed jpeg quality factors using deep convolutional neural network. In *ECCV*.

Pérez, P.; Gangnet, M.; and Blake, A. 2003. Poisson image editing. In *TOG*.

Salloum, R.; Ren, Y.; and Kuo, C.-C. J. 2018. Image splicing localization using a multi-task fully convolutional network (mfcn). In *JVCI*.

Shrivastava, A.; Pfister, T.; Tuzel, O.; Susskind, J.; Wang, W.; and Webb, R. 2017. Learning from simulated and unsupervised images through adversarial training. In *CVPR*.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Tsai, Y.-H.; Shen, X.; Lin, Z.; Sunkavalli, K.; Lu, X.; and Yang, M.-H. 2017. Deep image harmonization. In *CVPR*.

Wang, T.-C.; Liu, M.-Y.; Zhu, J.-Y.; Tao, A.; Kautz, J.; and Catanzaro, B. 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*.

Wang, X.; Shrivastava, A.; and Gupta, A. 2017. A-fast-rcnn: Hard positive generation via adversary for object detection. In *CVPR*.

Wei, Y.; Feng, J.; Liang, X.; Cheng, M.-M.; Zhao, Y.; and Yan, S. 2017. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *CVPR*.

Wen, B.; Zhu, Y.; Subramanian, R.; Ng, T.-T.; Shen, X.; and Winkler, S. 2016. Coverage–a novel database for copy-move forgery detection. In *ICIP*.

Yeh, R. A.; Chen, C.; Lim, T.-Y.; Schwing, A. G.; Hasegawa-Johnson, M.; and Do, M. N. 2017. Semantic image inpainting with deep generative models. In *CVPR*.

Zampoglou, M.; Papadopoulos, S.; and Kompatsiaris, Y. 2017. Large-scale evaluation of splicing localization algorithms for web images. In *MTAP*.

Zhou, P.; Han, X.; Morariu, V. I.; and Davis, L. S. 2017. Two-stream neural networks for tampered face detection. In *CVPRW*.

Zhou, P.; Han, X.; Morariu, V. I.; and Davis, L. S. 2018. Learning rich features for image manipulation detection. In *CVPR*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*.