MaLP: Manipulation Localization Using a Proactive Scheme

Vishal Asnani[1], Xi Yin[2], Tal Hassner[2], Xiaoming Liu[1]
[1*]Michigan State University, [2]Meta AI
[1]{asnanivi, liuxm}@msu.edu, [2]{yinxi, thassner}@meta.com

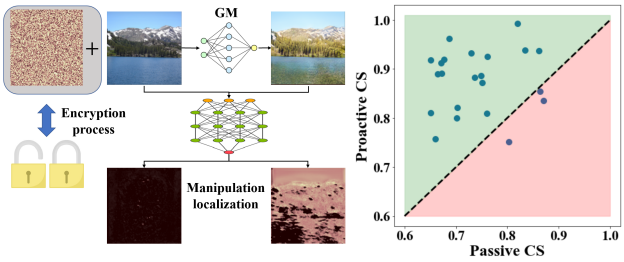
Abstract

Advancements in the generation quality of various Generative Models (GMs) has made it necessary to not only perform binary manipulation detection but also localize the modified pixels in an image. However, prior works termed as passive for manipulation localization exhibit poor generalization performance over unseen GMs and attribute modifications. To combat this issue, we propose a proactive scheme for manipulation localization, termed MaLP. We encrypt the real images by adding a learned template. If the image is manipulated by any GM, this added protection from the template not only aids binary detection but also helps in identifying the pixels modified by the GM. The template is learned by leveraging local and global-level features estimated by a two-branch architecture. We show that MaLP performs better than prior passive works. We also show the generalizability of MaLP by testing on 22 different GMs, providing a benchmark for future research on manipulation localization. Finally, we show that MaLP can be used as a discriminator for improving the generation quality of GMs. Our models/codes are available at www.github.com/vishal3477/pro_loc.

1. Introduction

We witness numerous Generative Models (GMs) [9, 10, 16, 18, 25–27, 30, 37, 42, 48, 54, 57, 66] being proposed to generate realistic-looking images. These GMs can not only generate an entirely new image [25, 26], but also perform partial manipulation of an input image [10, 10, 30, 66]. The proliferation of these GMs has made it easier to manipulate personal media for malicious use. Prior methods to combat manipulated media focus on binary detection [1, 2, 6, 12, 15, 32, 49, 52, 61, 62], using mouth movement, model parsing, hand-crafted features, *etc*.

Recent works go one step further than detection, *i.e*. *manipulation localization*, which is defined as follows: given a partially manipulated image by a GM (*e.g*. STGAN [30] modifying hair colors of a face image), the goal is to identify which pixels are modified by estimating a *fakeness map* [23]. Identifying modified pixels helps to determine the severity of the fakeness in the image, and aid media-forensics [12, 23]. Also, manipulation localization provides an understanding of the attacker's intent for modification which may further benefit identifying attack toolchains used [14].



(a) (b)

Figure 1. (a) High-level idea of MaLP. We encrypt the image by adding a learnable template, which helps to estimate the fakeness map. **(b)** The cosine similarity (CS) between ground-truth and predicted fakeness maps for 22 unseen GMs. The performance is better for almost all GMs when using our proactive approach.

Recent methods for manipulation localization [29, 40, 53] focus on estimating the manipulation mask of face-swapped images. They localize modified facial attributes by leveraging attention mechanisms [12], patch-based classifier [5], and face-parsing [23]. The main drawback of these methods is that they do not generalize well to GMs unseen in training. That is when the test images and training images are modified by different GMs, which will likely happen given the vast number of existing GMs. Thus, our work aims for a localization method generalizable to unseen GMs.

All aforementioned methods are based on a *passive* scheme as the method receives an image as is for estimation. Recently, proactive methods are gaining success for deepfake tasks such as detection [1], disruption [50, 63], and tagging [56]. These methods are considered *proactive* as they add different types of signals known as *templates* for encrypting the image before it is manipulated by a GM.

*All data sourcing, modeling codes, and experiments were developed at Michigan State University. Meta did not obtain the data/codes or conduct any experiments in this work.

This template can be one-hot encoding [56], adversarial perturbation [50], or a learnable noise [1], and is optimized to improve the performance of the defined tasks.

Motivated by [1], we propose a Proactive scheme for MAnipulation Localization, termed as MaLP, in order to improve generalization. Specifically, MaLP learns an optimized template which, when added to real images, would improve manipulation localization, should they get manipulated. This manipulation can be done by an unseen GM trained on either in-domain or out-of-domain datasets. Furthermore, face manipulation may involve modifying facial attributes unseen in training (*e.g.* train on hair color modification yet test on gender modification). MaLP incorporates three modules that focus on encryption, detection, and localization. The encryption module selects and adds the template from the template set to the real images. These encrypted images are further processed by localization and detection modules to perform the respective tasks.

Designing a proactive manipulation localization approach comes with several challenges. First, it is not straightforward to formulate constraints for learning the template *unsupervisedly*. Second, calculating a fakeness map at the same resolution as the input image is computationally expensive if the decision for each pixel has to be made. Prior works [5, 12] either down-sample the images or use a patch-wise approach, both of which result in inaccurate low-resolution fakeness maps. Lastly, the templates should be generalizable to localize modified regions from unseen GMs.

We design a two-branch architecture consisting of a shallow CNN network and a transformer to optimize the template during training. While the former leverages local-level features due to its shallow depth, the latter focuses on global-level features to better capture the affinity of the far-apart regions. The joint training of both networks enables the MaLP to learn a better template, having embedded the information of both levels. During inference, the CNN network alone is sufficient to estimate the fakeness map with a higher inference efficiency. Compared to prior passive works [12, 23], MaLP improves the generalization performance on unseen GMs. We also demonstrate that MaLP can be used as a discriminator for fine-tuning conventional GMs to improve the quality of GM-generated images.

In summary, we make the following contributions.

- We are the first to propose a proactive scheme for image manipulation localization, applicable to both face and generic images.
- Our novel two-branch architecture uses both local and global level features to learn a set of templates in an unsupervised manner. The framework is guided by constraints based on template recovery, fakeness maps classification, and high cosine similarity between predicted and ground-truth fakeness maps.

Table 1. Comparison of our approach with prior works on manipulation localization and proactive schemes. We show the generalization ability of all works across different facial attribute modifications, unseen GMs trained on datasets with the same domain (in-domain) and different domains (out-domain). [Keys: Attr.: Attributes, Imp.: Improving, L.: Localization, D.: Detection]

Work	Scheme	Task	Template	Generalization			Imp.
				Attr.	In-domain	Out-domain	GM
[56]	Proactive	Tag	Fix	✔	✔	✖	✖
[51]	Proactive	Disrupt	Learn	✔	✖	✖	✖
[50]	Proactive	Disrupt	Learn	✔	✔	✖	✖
[63]	Proactive	Disrupt	Learn	✔	✖	✖	✖
[1]	Proactive	D.	Learn	✖	✔	✔	✖
[40]	Passive	L. + D.	-	✖	✖	✖	✖
[53]	Passive	L. + D.	-	✖	✖	✖	✖
[29]	Passive	L. + D.	-	✖	✔	✖	✖
[12]	Passive	L. + D.	-	✔	✔	✖	✖
[5]	Passive	L. + D.	-	✖	✔	✖	✖
[23]	Passive	L. + D.	-	✔	✔	✖	✖
MaLP	Proactive	L. + D.	Learn	✔	✔	✔	✔

- MaLP can be used as a plug-and-play discriminator module to fine-tune the generative model to improve the quality of the generated images.
- Our method outperforms State-of-The-Art (SoTA) methods in manipulation localization and detection. Furthermore, our method generalizes well to GMs and modified attributes unseen in training. To facilitate the research of localization, we develop a benchmark for evaluating the generalization of manipulation localization, on images where the train and test GMs are different.

2. Related Work

Manipulation Localization. Prior works tackle manipulation localization by adopting a passive scheme. Some of them focus on forgery attacks like removal, copy-move, and splicing using multi-task learning [40]. Songsri-in *et al.* [53] leverage facial landmarks [11] for manipulation localization. Li *et al.* [29] estimate the blended boundary for forged face-swap images. [12] uses an attention mechanism to leverage the relationship between pixels and [5] uses a patch-based classifier to estimate modified regions. Recently, Huang *et al.* [23] utilize gray-scale maps as ground truth for manipulation localization and leverage face parsing with an attention mechanism for prediction. The passive methods discussed above suffer from the generalization issue [5, 11, 12, 23, 40, 53] and estimate a low-resolution fakeness map [12] which is less accurate for the localization purpose. MaLP generalizes better to modified attributes and GMs unseen in training.

Proactive Scheme. Recently, proactive schemes are developed for various tasks. Wang *et al.* [56] leverage the recovery of embedded one-hot encoding messages to perform deepfake tagging. A small perturbation is added onto the images by Segalis *et al.* [51] to disrupt the output of a GM. The same task is performed by Ruiz *et al.* [50] and Yeh *et al.* [63], both adding adversarial noise onto the in-

put images. Asnani *et al*. [1] propose a framework based on adding a learnable template to input images for generalized manipulation detection. Unlike prior works, which focus on binary detection, deepfake disruption, or tagging, our work emphasizes on manipulation localization. We show the comparison of our approach with prior works in Tab. 1.

Manipulation Detection. The advancement in manipulation detection keeps reaching new heights. Prior works propose to combat deepfakes by exploiting frequency domain patterns [58], up-sampling artifacts [65], model parsing [2], hand-crafted features [38], lip motions [49], and self-attention [12]. Recent methods use self-blended images [52], real-time deviations [15], and self-supervised learning with adversarial training [6]. Finally, methods based on contrastive learning [62] and proactive scheme [1] have explicitly focused on generalized manipulation detection across unknown GMs.

3. Proposed Approach

3.1. Problem Formulation

Passive Manipulation Localization Let $\boldsymbol{I}^R$ be a set of real images that are manipulated by a GM G to output the set of manipulated images $G(\boldsymbol{I}^R)$. Prior passive works perform manipulation localization by estimating the fakeness map $\boldsymbol{M}_{pred}$ with the following objective:

$$\min_{\theta_{\mathcal{E}}} \Big\{ \sum_j \Big(\Big|\Big| \mathcal{E}(G(\boldsymbol{I}_j^R); \theta_{\mathcal{E}}) - \boldsymbol{M}_{GT} \Big|\Big|_2 \Big) \Big\}, \tag{1}$$

where $\mathcal{E}$ denotes the passive framework with parameters $\theta_{\mathcal{E}}$ and $\boldsymbol{M}_{GT}$ is the ground-truth fakeness map.

To represent the fakeness map, some prior methods [12, 29, 53] choose a binary map by applying a threshold on the difference between the real and manipulated images. This is undesirable as the threshold selection is highly subjective and sensitive, leading to inaccurate fakeness maps. Therefore, we adopt the continuous gray-scale map for calculating the ground-truth fakeness maps [23], formulated as:

$$\boldsymbol{M}_{GT} = Gray(|\boldsymbol{I}^R - G(\boldsymbol{I}^R)|)/255, \tag{2}$$

where $Gray(.)$ converts the image to gray-scale.

Proactive Scheme Asnani *et al*. [1] define adding the template as a transformation $\mathcal{T}$ applied to images $\boldsymbol{I}^R$, resulting in the encrypted images $\mathcal{T}(\boldsymbol{I}^R)$. The added template acts as a signature of the defender and is learned during the training, aiming to improve the performance of the task at hand, *e.g*. detection, disruption, and tagging. Motivated by [1] that uses multiple templates, we have a set of n orthogonal templates $\mathcal{S} = \{\boldsymbol{S}_1, \boldsymbol{S}_2, ...\boldsymbol{S}_n\}$ where $\boldsymbol{S}_i \in \mathbb{R}^{128\times128}$, for a real image $\boldsymbol{I}_j^R \in \boldsymbol{I}^R$, transformation $\mathcal{T}$ is defined as:

$$\mathcal{T}(\boldsymbol{I}_j^R; \boldsymbol{S}_i) = \boldsymbol{I}_j^R + \boldsymbol{S}_i, \text{ where } i \in \{1, 2, ..., n\}. \tag{3}$$

The templates are optimized such that adding them to the real images wouldn't result in a noticeable visual difference, yet helps manipulation localization.

Proactive Manipulation Localization. Unlike the passive schemes [12, 23, 29, 40], we learn an optimal template set to help manipulation localization. For the encrypted images $\mathcal{T}(\boldsymbol{I}^R)$, we formulate the estimation of the fakeness map as:

$$\min_{\theta_{\mathcal{E}_P}, \boldsymbol{S}_i} \Big\{ \sum_j \Big(\Big|\Big| \mathcal{E}_P(G(\mathcal{T}(\boldsymbol{I}_j^R; \boldsymbol{S}_i)); \theta_{\mathcal{E}_P}) - \boldsymbol{M}_{GT} \Big|\Big|_2 \Big) \Big\}. \tag{4}$$

where $\mathcal{E}_P$ is the proactive framework with parameters $\theta_{\mathcal{E}_P}$.

However, as the output of the GM has changed from images in set $G(\boldsymbol{I}^R)$ to images in set $G(\mathcal{T}(\boldsymbol{I}^R))$, in our proactive approach, the calculation of the ground-truth fakeness map shall be changed from Eq. 2 to the follows:

$$\boldsymbol{M}_{GT} = Gray(|\boldsymbol{I}^R - G(\mathcal{T}(\boldsymbol{I}^R))|)/255. \tag{5}$$

3.2. Manipulation Localization

MaLP consists of three modules: encryption, localization, and detection. The encryption module is used to encrypt the real images. The localization module estimates the fakeness map using a two-branch architecture. The detection module performs binary detection for the encrypted and manipulated images by recovering the template and using the classifier in the localization module. All three modules, as detailed next, are trained in an end-to-end manner.

3.2.1 Encryption Module

Following the procedure in [1], we add a randomly selected learnable template from the template set to a real image. We control the strength of the added template using a hyper-parameter m, which prevents the degradation of the image quality. The encryption process is summarised below:

$$\mathcal{T}(\boldsymbol{I}_j^R) = \boldsymbol{I}_j^R + m \times \boldsymbol{S}_i \text{ where } i = Rand(1, 2, ..., n). \tag{6}$$

We select the value of m as 30% for our framework.

We optimize the template set by focusing on properties like low magnitude, orthogonality, and high-frequency content [1]. The properties are applied as constraints as follows.

$$J_T = \lambda_1 \times \sum_{i=1}^{n} ||\boldsymbol{S}_i||_2 + \lambda_2 \times \sum_{\substack{i,j=1 \\ i \neq j}}^{n} \text{CS}(\boldsymbol{S}_i, \boldsymbol{S}_j) + \lambda_3 \times ||\mathcal{L}(\mathfrak{F}(\boldsymbol{S}))||_2, \tag{7}$$

where CS is the cosine similarity, $\mathcal{L}$ is the low-pass filter, $\mathfrak{F}$ is the fourier transform, λ_1, λ_2, λ_3 are weights for losses of low magnitude, orthogonality and high-frequency content, respectively.

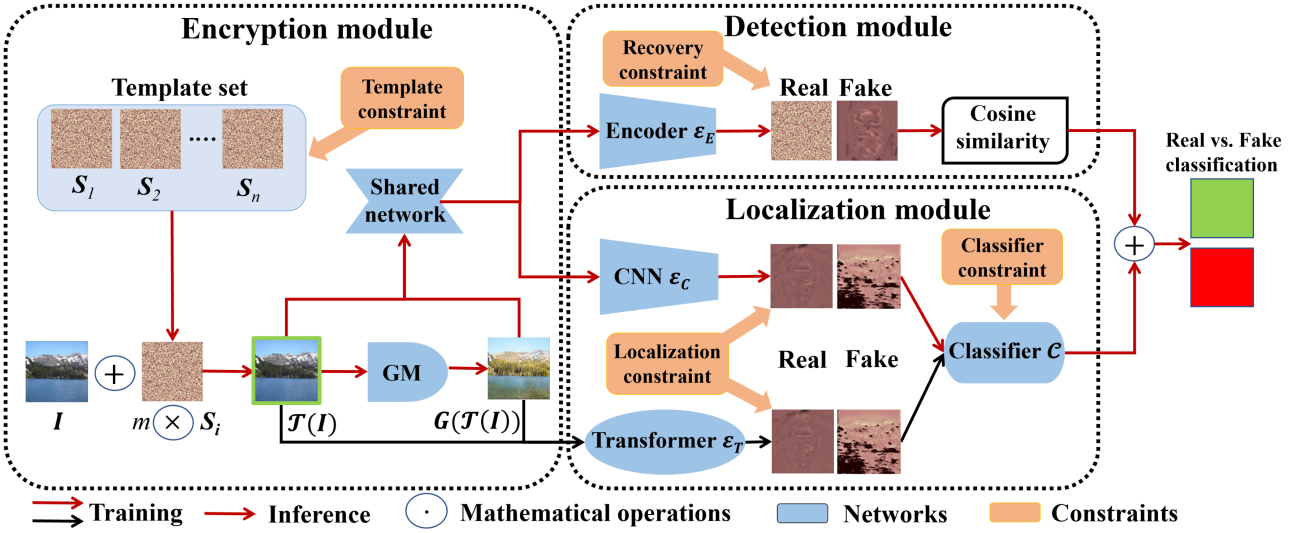


Figure 2. The overview of MaLP. It includes three modules: encryption, localization, and detection. We randomly select a template from the template set and add it to the real image as encryption. The GM is used in inference mode to manipulate the encrypted image. The detection module recovers the added template for binary detection. The localization module uses a two-branch architecture to estimate the fakeness map. Lastly, we apply the classifier to the fakeness map to better distinguish them from each other. Best viewed in color.

3.2.2 Localization Module

To design the localization module, we consider two desired properties: a larger receptive field for fakeness map estimation and high inference efficiency. A network with a large receptive field will consider far-apart regions in learning features for localization. Yet, large receptive fields normally come from deeper networks, implying slower inference.

In light of these properties, we design a two-branch architecture consisting of a shallow CNN network $\mathcal{E}_C$ and a ViT transformer [13] $\mathcal{E}_T$ (see Fig. 2). The intuition is to have one shallow branch to capture local features, and one deeper branch to capture global features. While training with both branches helps to learn better templates, in inference we only use the shallow branch for a higher efficiency. Specifically, the shallow CNN network has 10 layers which is efficient in inference but can only capture the local features due to small receptive fields. To capture global information, we adopt the ViT transformer. With the self-attention between the image patches, the transformer can estimate the fakeness map considering the far-apart regions.

Both the CNN and transformer are trained jointly to estimate a better template set, resembling the concept of the ensemble of networks. We empirically show that training both networks simultaneously results in higher performance than training either network separately. As the shallow CNN network is much faster in inference than the transformer, we use the transformer only in training to optimize the templates and switch off the transformer branch in inference.

To estimate the fakeness map, we leverage the supervision of the ground-truth fakeness map in Eq. 5. For fake images, we maximize the cosine similarity (CS) and structural similarity index measure (SS) between the predicted and ground-truth fakeness map. However, the fakeness map should be a zero image for encrypted images. Therefore, we apply an L_2 loss [23] to minimize the predicted map to zero for encrypted images. To maximize the difference between the two fakeness maps, we further minimize the cosine similarity between the predicted map from encrypted images and $\boldsymbol{M}_{GT}$. The localization loss is defined as:

$$J_L = \begin{cases} \Big\{\lambda_4 \times ||\mathcal{E}_{C/T}(\boldsymbol{I})||_2^2 + & \text{if } \boldsymbol{I} \in \mathcal{T}(\boldsymbol{I}^R) \\ \lambda_5 \times \text{CS}(\mathcal{E}_{C/T}(\boldsymbol{I}), \boldsymbol{M}_{GT})\Big\} & \\ \Big\{\lambda_6 \times (1 - \text{CS}(\mathcal{E}_{C/T}(\boldsymbol{I}), \boldsymbol{M}_{GT})) + & \text{if } \boldsymbol{I} \in G(\mathcal{T}(\boldsymbol{I}^R)) \\ \lambda_7 \times (1 - SS(\mathcal{E}_{C/T}(\boldsymbol{I}), \boldsymbol{M}_{GT}))\Big\}. & \end{cases} \tag{8}$$

Finally, we have a classifier to make a binary decision of real *vs*. fake using the fakeness maps. This classifier is included in the framework to aid the detection module for binary detection of the input images, which will be discussed in Sec. 3.2.3. Another reason to have the classifier is to make the fakeness maps from encrypted and fake images to be distinguishable. We find that this design allows our training to converge much faster.

3.2.3 Detection Module

To leverage the added template for manipulation detection, we perform template recovery using encoder $\mathcal{E}_E$. We follow the procedure in [1] to recover the added template from the encrypted images by maximizing the cosine similarity between $\boldsymbol{S}$ and $\boldsymbol{S}_R$. However, for manipulated images, we minimize the cosine similarity between the recovered tem-

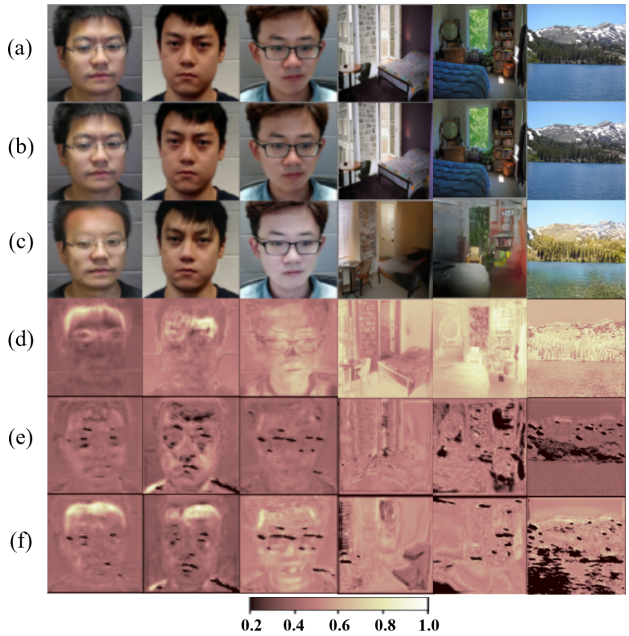


Figure 3. Visualization of fakeness maps for faces and generic images showing generalization across unseen attribute modifications and GMs: (a) real image, (b) encrypted image, (c) manipulated image, (d) $\boldsymbol{M}_{GT}$, (e) predicted fakeness map for encrypted images, and (f) predicted fakeness map for manipulated images. The first column shows the manipulation of (seen GM, seen attribute modification) *i.e.* (STGAN, bald). Following two columns show the manipulation of (seen GM, unseen attribute modification) *i.e.* (STGAN, [bangs, pale skin]. The fourth and fifth columns show manipulation of unseen GM, GauGAN for non-face images. The last column shows manipulation by unseen GM, DRIT. We see that the fakeness map of manipulated images is more bright and similar to $\boldsymbol{M}_{GT}$, while the real fakeness map is more close to zero. We use the cmap as "pink" to better visualize the fakeness map. All face images come from SiWM-v2 data [19].

plate ($\boldsymbol{S}_R$) and all the templates in the template set $\mathcal{S}$.

$$J_R = \begin{cases} \lambda_8 \times (1 - \text{CS}(\boldsymbol{S}, \boldsymbol{S}_R)) & \text{if } x \in \mathcal{T}(\boldsymbol{I}^R) \\ \lambda_9 \times (\sum_{i=1}^{n} (\text{CS}(\boldsymbol{S}_i, \boldsymbol{S}_R))) & \text{if } x \in G(\mathcal{T}(\boldsymbol{I}^R)). \end{cases} \tag{9}$$

Further, we leverage our estimated fakeness map to help manipulation detection. As discussed in the previous section, we apply a classifier $\mathcal{C}$ to perform binary classification of the predicted fakeness map for the encrypted and fake images. The logits of the classifier are further combined with the cosine similarity of the recovered template. The averaged logits are back-propagated using the binary cross-entropy constraint. This not only improves the performance of manipulation detection but also helps manipulation localization. Therefore, we apply the binary cross entropy loss on the averaged logits as follows:

$$\begin{aligned} J_C = & \lambda_{10} \times -\sum_j \Big\{ y_j . \log\Big[\frac{\mathcal{C}(\boldsymbol{X}_j) + CS(\boldsymbol{S}_R, \boldsymbol{S})}{2}\Big] - \\ & (1 - y_j) . \log\Big[1 - \frac{\mathcal{C}(\boldsymbol{X}_j) + CS(\boldsymbol{S}_R, \boldsymbol{S})}{2}\Big] \Big\}, \end{aligned} \tag{10}$$

Table 2. Manipulation localization comparison with prior works.

Method	Localization			Detection		
	CS ↑	PSNR ↑	SSIM ↑	Accuracy ↑	EER ↓	AUC ↑
[12]	0.6230	6.214	0.2178	0.9975	**0.0050**	0.9975
[23]	0.8831	22.890	**0.7876**	0.9945	0.0077	0.9998
MaLP	**0.9394**	**23.020**	0.7312	**0.9991**	0.0072	**1.0**

where y_j is the class label, $\boldsymbol{S}$ and $\boldsymbol{S}_R$ are the added and recovered template respectively.

Our framework is trained in an end-to-end manner with the overall loss function as follows:

$$J = J_T + J_R + J_C + J_L. \tag{11}$$

3.3. MaLP as A Discriminator

One application of MaLP is to leverage our proposed localization module as a discriminator for improving the quality of the manipulated images. MaLP performs binary classification by estimating a fakeness map, which can be used as an objective. This results in output images being resilient to manipulation localization, thereby lowering the performance of our framework.

We use MaLP as a plug-and-play discriminator to improve image generation quality through fine-tuning pre-trained GMs. The generation quality and manipulation localization will compete head-to-head, resulting in a better quality of the manipulated images. We define the fine-tuning objective for the GM as follows:

$$\begin{aligned} & \min_{\theta_G} \max_{\theta_{MaLP}, \boldsymbol{S}_i} \Big\{ \sum_j \Big(\mathbb{E}\big[log(\mathcal{E}_{MaLP}(\mathcal{T}(\boldsymbol{I}_j^R)); \theta_{MaLP})\big] + \\ & \mathbb{E}\big[1 - log(\mathcal{E}_{MaLP}(G(\mathcal{T}(\boldsymbol{I}_j^R; \boldsymbol{S}_i); \theta_G); \theta_{MaLP}))\big] \Big) \Big\}. \end{aligned} \tag{12}$$

where $\mathcal{E}_{MaLP}$ is our framework with θ_{MaLP} parameters.

4. Experiments

4.1. Experimental Setup

Settings Following the settings in [23], we use STGAN [30] to manipulate images from CelebA [33] dataset and train on bald facial attribute modification. In order to evaluate the generalization of image manipulation localization, we construct a new benchmark that consists of 200 real images of 22 different GMs on various data domains. The real images are chosen from the dataset on which the GM is trained on. The list of GMs, datasets and implementation details are provided in the supplementary.

Evaluation Metrics We use cosine similarity (CS), peak signal-to-noise ratio (PSNR), and structural similarity index measure (SSIM) as adopted by [23] to evaluate manipulation localization since the GT is a continuous map. For binary detection, we use the area under the curve (AUC), equal error rate (EER), and accuracy score [23].

Table 3. Comparison of localization performance across unseen GMs and attribute modifications. We train on STGAN bald/smile attribute modification and test on AttGAN/StyleGAN.

Method	Cosine similarity ↑(AttGAN)			Cosine similarity ↑ (StyleGAN)		
	Bald	Black Hair	Eyeglasses	Smile	Age	Gender
[23]	0.8141	0.6932	0.6950	0.6176	0.3141	0.6470
MaLP	**0.8201**	**0.7940**	**0.8557**	**0.8159**	**0.8255**	**0.8016**

4.2. Comparison with Baselines

We compare our results with [23] and [12] for manipulation localization. The results are shown in Tab. 2. MaLP has higher cosine similarity and similar PSNR for localization compared to [23]. However, we observe a dip in SSIM. This might be because of the degradation caused by adding our template to the real images and then performing the manipulation. The learned template helps localize the manipulated regions better, as demonstrated by cosine similarity, but the degradation affects SSIM and PSNR. We also compare the performance of real *vs*. fake binary detection. As expected, our proposed proactive approach outperforms the passive methods with a perfect AUC and near-perfect accuracy. We also show visual examples of fakeness maps for images modified by unseen GMs in Fig. 3. MaLP is able to estimate the fakeness map for unseen modifications and GMs across face/generic image datasets.

4.3. Generalization

Across Attribute Modifications Following the settings in [23], we evaluate the performance of MaLP across unseen attribute modifications. Specifically, we train MaLP using STGAN with the bald/smile attribute modification and test it on unseen attribute modifications with unseen GMs: AttGAN/StyleGAN. As shown in Tab. 3, MaLP is more generalizable to all unseen attribute modifications. Furthermore, AttGAN shares the high-level architecture with STGAN but not with StyleGAN. We observe a significant increase in localization performance for StyleGAN compared to AttGAN. This shows that, unlike our MaLP, passive works perform much worse if the test GM doesn't share any similarity with the training GM.

Across GMs Although [23] tries to show generalization across unseen GMs; it is limited by the GMs within the same domain of the dataset used in training. We propose a benchmark to evaluate the generalization performance for future manipulation localization works that consists of 22 different GMs in various domains. We select GMs that are publicly released and can perform partial manipulation.

As no open-source code base is available for [23], we train a passive approach using a ResNet50 [20] network to estimate the fakeness map as the baseline for comparison. Further, we compare our approach with [5, 12]. Although [5, 12] estimate a fakeness map, it has at least $5\times$ lower resolution compared to input images due to their patch-based methodology. For a fair comparison, we rescale their predicted fakeness maps to the resolution of $\boldsymbol{M}_{GT}$. We compare the cosine similarity in Tab. 4. MaLP is able to outperform all the baselines for almost all GMs, which proves the effectiveness of the proactive scheme.

We also evaluate the performance of $\mathcal{E}_C$ for high-resolution images. For encryption, we upsample the 128×128 template to the original resolution of images and evaluate $\mathcal{E}_C$ on these higher resolution encrypted images. We observe similar performance of $\mathcal{E}_C$ for higher resolution images in Tab. 4, proving the versatility of $\mathcal{E}_C$ to image sizes.

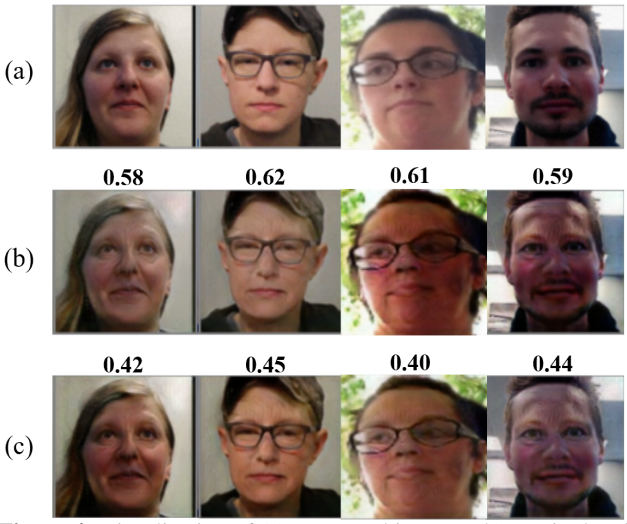


Figure 4. Visualization of (a) encrypted images, (b) manipulated images before fine-tuning, and (c) manipulated images after fine-tuning. The generation quality has improved after we fine-tune the GM using our framework as a discriminator. The artifacts in the images have been reduced, and the face skin color is less pale and more realistic. We also specify the cosine similarity of the predicted fakeness map and $\boldsymbol{M}_{GT}$. The GM is able to decrease the performance of our framework after fine-tuning. All face images come from SiWM-v2 data [19].

4.4. Improving Quality of GMs

We fine-tune the GM into fooling our framework to generate a fakeness map as a zero image. This process results in better-quality images. Initially, we train MaLP with the pretrained GM so that it can perform manipulation localization. Next, to fine-tune the GM, we adopt two strategies. First, we freeze MaLP and fine-tune the GM only. Second, we fine-tune both the GM and the MaLP but update the MaLP with a lower learning rate. The result for fine-tuning StarGAN is shown in Tab. 5. We observe that for both strategies, MaLP reduces the FID score of StarGAN. We also show some visual examples in Fig. 4. We see that the images are of better quality after fine-tuning, and many artifacts in the images manipulated by the pretrained model are removed.

Table 4. Benchmark for manipulation localization across 22 different unseen GMs, showing cosine similarity between ground-truth and predicted fakeness maps. We compare our proactive *vs*. passive baselines [5, 12, 20] approach to highlight the generalization ability of our MaLP. We scale the images to 128^2 for "sc." and keep the resolution as is for "no sc.".

GM	SEAN [68]	StarGAN [9]	CycleGAN [66]	GauGAN [42]	Con_Enc. [44]	StarGAN2 [10]	ALAE [45]	BiGAN [67]	AuGAN [66]	GANim [46]	DRGAN [54]	ILVR [8]
Resolution	256^2	128^2	256^2	256^2	128^2	256^2	256^2	256^2	340^2	128^2	128^2	256^2
ResNet50 [20]	0.8614	0.7513	0.6715	0.7615	**0.8639**	0.8196	0.6766	0.6514	0.6639	0.6871	**0.8029**	0.7018
[5]	0.7514	0.7111	0.7981	0.8016	0.7894	0.7026	0.7156	0.7217	0.7516	0.7612	0.7115	0.7851
[12]	0.7961	0.7887	0.8014	0.8256	0.8541	0.7034	0.7549	0.7805	0.7232	0.8457	0.7239	0.7854
MaLP (sc.)	**0.9376**	**0.8718**	0.9128	**0.9251**	0.8546	**0.8836**	**0.9192**	0.9181	0.8894	**0.9625**	0.7512	0.8003
MaLP (no sc.)	0.9258	**0.8718**	**0.9245**	0.9125	0.8546	0.8785	0.9141	**0.9229**	**0.9149**	**0.9625**	**0.7512**	**0.8359**

GM	DRIT [28]	Pix2Pix [24]	CounGAN [41]	DualGAN [64]	ESRGAN [60]	UNIT [31]	MUNIT [22]	ColGAN [39]	GDWCT [7]	RePaint [35]	Average
Resolution	256^2	256^2	128^2	256^2	1024^2	512×931	256×512	128^2	128^2	256^2	-
ResNet50 [20]	0.7486	0.6719	0.7293	0.7365	**0.8703**	0.7083	0.6601	0.7596	0.8350	0.6512	0.7401
[5]	0.7871	0.7769	0.8146	0.7569	0.8168	0.8064	0.6788	0.7610	0.8691	0.7516	0.7645
[12]	0.8120	0.7781	0.8559	0.7721	0.8241	0.8086	0.7097	0.7874	0.8879	0.7696	0.7903
MaLP (sc.)	0.8867	**0.8915**	**0.9326**	**0.8872**	0.8348	0.8214	0.7565	**0.8096**	**0.9384**	0.8102	0.8725
MaLP (no sc.)	**0.9084**	0.8714	**0.9326**	0.8432	**0.8743**	0.8391	**0.7860**	**0.8096**	**0.9384**	**0.8290**	**0.8773**

Table 5. FID score comparison for the application of our approach as a discriminator for improving the generation quality of the GM

State	Fine-tune	StarGAN FID ↓
Before	–	60.49
After	G	**51.91**
	$G + MaLP$	52.07

Table 6. Comparison with prior binary detection works. [Keys: D.M.: Detection module, L.M.: Localization module]

Method	Train GM	Set size	Test GM Average precision (%)↑		
			CycleGAN	StarGAN	GauGAN
Nataraj *et al*. [38]	CycleGAN	-	**100**	88.20	56.20
Zhang *et al*. [65]	AutoGAN	-	**100**	**100**	61.00
Wang *et al*. [58]	ProGAN	-	84.00	**100**	67.00
Asnani *et al*. [1]	STGAN	1	94.00	**100**	69.50
MaLP (D.M.)	STGAN	1	94.10	**100**	69.61
MaLP (D.M. + L.M.)	STGAN	1	94.30	**100**	**72.16**

4.5. Other Comparisons

Binary Detection We compare with prior proactive and passive approaches for binary manipulation detection [1, 38, 58, 65]. We adopt the evaluation protocol in [1] to test on images manipulated by CycleGAN, StarGAN, and GauGAN. We are able to perform similar to [1] as shown in Tab. 6. We have better average precision than passive schemes and generalize well to GMs unseen in training. We also conduct experiments to see whether localization can help binary detection to improve the performance, as mentioned in Sec. 3.2.3. The combined predictions' results are better than just using the detection module as shown in Tab. 6. This is intuitive as the localization module provides extra information, thereby increasing the performance.

Inference Speed We compare the inference speed of our MaLP against prior work. [23] uses Deeplabv3-ResNet101 model from PyTorch [43]. In our generalization benchmark shown in Sec. 4.3, we use the ResNet50 model for training the passive baseline. The inference speed per image on an NVIDIA K80 GPU for Deeplabv3-ResNet101, ResNet50, and MaLP are 75.61, 52.66, and 29.26 ms, respectively. MaLP takes less than half the inference time compared to [23] due to our shallow CNN network.

Adversarial Attack Our framework can be considered as an adversarial attack on real images to aid manipulation localization. Therefore, it is vital to contrast the performance between our approach and classic adversarial attacks. For this purpose, we perform experiments that make use of adversarial attacks, namely PGD [36], CW [4], and FGSM [17] to guide the learning of the added template. We evaluate on unseen GM AttGAN for unseen attribute modifications. We show the performance comparison in Tab. 7. MaLP has higher cosine similarity across some unseen facial attribute modifications compared to adversarial attacks. This can be explained as the adversarial attack methods being over-fitted to training parameters (data, target network *etc*.). Therefore, if the testing data is changed with unseen attribute modifications by GMs, the performance of adversarial attacks degrades. Further, these attacks are analogous to our MaLP as a proactive scheme which, in general, have better performance than passive works.

Table 7. Comparison with adversarial attack methods.

Method	Scheme	Cosine similarity↑		
		Bald	Black Hair	Eyeglasses
Huang *et al*. [23]	Passive	0.8141	0.6932	0.6950
PGD [36]	Proactive	0.8051	0.7514	0.8358
FGSM [17]	Proactive	0.8111	0.7882	0.8512
CW [4]	Proactive	0.8014	**0.8344**	0.8405
MaLP	Proactive	**0.8201**	0.7940	**0.8557**

Model Robustness Against Degradations It is necessary to test the robustness of our proposed approach against various types of real-world image editing degradations. We evaluate our method on degradations applied during testing as adopted by [23], which include JPEG compression, blurring, adding noise, and low resolution. The results are shown in Fig. 5. Our proposed MaLP is more robust to real-world degradations than passive schemes.

4.6. Ablations

Two-branch Architecture As described in Sec. 3.2.2, MaLP adopts a two-branch architecture to predict the fakeness map using the local-level and global-level features, which are estimated by a shallow CNN and a transformer. We ablate by training each branch separately to show the

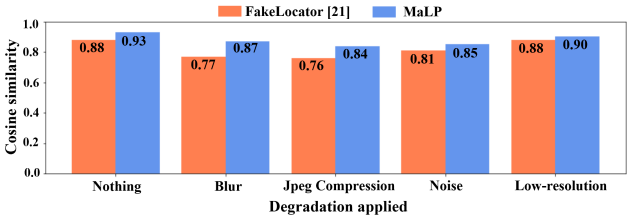


Figure 5. Comparison of our approach's robustness against common image editing degradations.

Table 8. Ablation of two-branch architecture. CNN is a shallow network with 10 layers. Training each branch separately has worse localization results than combining them.

Network trained	Cosine similarity ↑	Accuracy ↑
CNN only	0.8961	0.9801
Transformer only	0.8848	0.9856
CNN + ResNet50	0.8647	0.9512
CNN + Transformer	**0.9394**	**0.9981**

effectiveness of combining them. As shown in Tab. 8, if the individual network is trained separately, the performance is lower than the two-branch architecture. Next, to show the efficacy of the transformer, we use a ResNet50 network in place of the transformer to predict the fakeness map. We observe that the performance is even worse than using only the transformer. ResNet50 lacks the added advantage of self-attention in the transformer, which estimates the global-level features much better than a CNN network.

Constraints MaLP leverages different constraints to estimate the fakeness map using an optimized template. We perform an ablation by removing each constraint separately, showing the importance of every constraint. Tab. 9 shows the cosine similarity for localization and accuracy for detection. Removing either the classifier or recovery constraint results in lower detection performance. This is expected as we leverage logits from both $\mathcal{C}$ and $\mathcal{E}_E$, and removing the constraint for one network will hurt the logits of the other network. Furthermore, removing the template constraint results in a decrease in performance. Although the gap is small, the template is not properly optimized to have lower magnitude and high-frequency content.

Removing the localization constraint and just applying a L_2 loss for supervising fakeness maps result in a significant performance drop for both localization and detection, showing the necessity of this constraint. Finally, we show the importance of a learnable template by not optimizing it during the training of MaLP. This hurts the performance a lot, similar to removing the localization constraint. Both these observations prove that our localization constraint and learnable template are important components of MaLP.

Template Set Size We perform an ablation to vary the size of the template set $\mathcal{S}$. Having multiple templates will improve security if an attacker tries to reverse engineer the template from encrypted images. The results are shown in

Table 9. Ablation of constraints used in training our framework.

Constraint removed	Cosine similarity ↑	Accuracy ↑
Classifier constraint J_C	0.9319	0.9814
Template constraint J_T	0.9143	0.9803
Localization constraint J_L	0.8814	0.9539
Recovery constraint J_R	0.9206	0.9780
Fixed template	0.8887	0.9514
Nothing (MaLP)	**0.9394**	**0.9991**

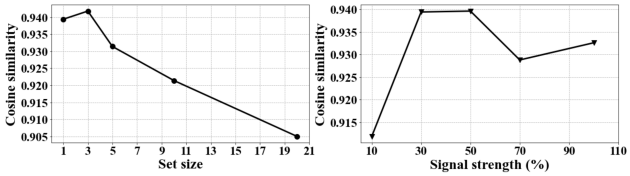


Figure 6. Ablation study on hyperparameters used in our framework: set size and signal strength.

Fig. 6 (a). The cosine similarity takes a dip when the set size is increased. We also observe the inter-template cosine similarity, which remains constant at a high value of around 0.74 for all templates. This is against the findings of [1]. Localization is a more challenging task than binary detection. Therefore, it is less likely to find different templates for our MaLP in the given feature space compared to [1].

Signal Strength We vary the template strength hyperparameter m to find its impact on the performance. As shown in Fig. 6 (b), the cosine similarity increases as we increase the strength of the added template. However, this comes with the lower visual quality of the encrypted images if the template strength is increased. The performance doesn't vary much after $m = 30\%$, which we use for MaLP.

5. Conclusion

This paper focuses on manipulation localization using a proactive scheme (MaLP). We propose to improve the generalization of manipulation localization across unseen GM and facial attribute modifications. We add an optimal template onto the real images and estimate the fakeness map via a two-branch architecture using local and global-level features. MaLP outperforms prior works with much stronger generalization capabilities, as demonstrated by our proposed evaluation benchmark with 22 different GMs in various domains. We show an application of MaLP in fine-tuning GMs to improve generation quality.

Limitations First, the number of publicly available GMs is limited. More thorough testing on many different GMs might give more insights into the problem of generalizable manipulation localization. Second, we show that our MaLP can be used to fine-tune the GMs to improve image generation quality. However, it is based on the pretrained GM. Using our method to train a GM from scratch can be an interesting direction to explore in the future.

References

[1] Vishal Asnani, Xi Yin, Tal Hassner, Sijia Liu, and Xiaoming Liu. Proactive image manipulation detection. In *CVPR*, 2022. 1, 2, 3, 4, 7, 8, 11

[2] Vishal Asnani, Xi Yin, Tal Hassner, and Xiaoming Liu. Reverse engineering of generative models: Inferring model hyperparameters from generated images. *arXiv preprint arXiv:2106.07873*, 2021. 1, 3

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *CVPR*, 2018. 11

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *SSP*, 2017. 7

[5] Lucy Chai, David Bau, Ser-Nam Lim, and Phillip Isola. What makes fake images detectable? understanding properties that generalize. In *ECCV*, 2020. 1, 2, 6, 7

[6] Liang Chen, Yong Zhang, Yibing Song, Lingqiao Liu, and Jue Wang. Self-supervised learning of adversarial example: Towards good generalizations for deepfake detection. In *CVPR*, 2022. 1, 3

[7] Wonwoong Cho, Sungha Choi, David Keetae Park, Inkyu Shin, and Jaegul Choo. Image-to-image translation via group-wise deep whitening-and-coloring transformation. In *CVPR*, 2019. 7, 11

[8] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. ILVR: Conditioning method for denoising diffusion probabilistic models. In *ICCV*, 2021. 7

[9] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 1, 7, 11

[10] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020. 1, 7, 11

[11] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 2

[12] Hao Dang, Feng Liu, Joel Stehouwer, Xiaoming Liu, and Anil K Jain. On the detection of digital face manipulation. In *CVPR*, 2020. 1, 2, 3, 5, 6, 7

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020. 4, 11

[14] Bruce Draper. Reverse engineering of deceptions (red). https://www.darpa.mil/program/reverse-engineering-of-deceptions. 1

[15] Candice R Gerstner and Hany Farid. Detecting real-time deep-fake videos using active illumination. In *CVPR*, 2022. 1, 3

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 1

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 7

[18] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *CVPR*, 2022. 1

[19] Xiao Guo, Yaojie Liu, Anil Jain, and Xiaoming Liu. Multi-domain learning for updating face anti-spoofing models. In *ECCV*, 2022. 5, 6, 14, 15

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6, 7

[21] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. *IEEE transactions on image processing*, 28:5464–5478, 2019. 11

[22] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 7, 11

[23] Yihao Huang, Felix Juefei-Xu, Qing Guo, Yang Liu, and Geguang Pu. FakeLocator: Robust localization of gan-based face manipulations. *IEEE Transactions on Information Forensics and Security*, 17:2657–2672, 2022. 1, 2, 3, 4, 5, 6, 7, 12, 13

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 7, 11

[25] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1, 11

[26] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1

[27] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. DiffusionCLIP: Text-guided diffusion models for robust image manipulation. In *CVPR*, 2022. 1

[28] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018. 7, 11

[29] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *CVPR*, 2020. 1, 2, 3

[30] Ming Liu, Yukang Ding, Min Xia, Xiao Liu, Errui Ding, Wangmeng Zuo, and Shilei Wen. STGAN: A unified selective transfer network for arbitrary image attribute editing. In *CVPR*, 2019. 1, 5, 11

[31] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NeurIPS*, 2017. 7, 11

[32] Xiaohong Liu, Yaojie Liu, Jun Chen, and Xiaoming Liu. PSCC-Net: Progressive spatio-channel correlation network for image manipulation detection and localization. In *arXiv preprint arXiv:2103.10596*, 2021. 1

[33] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 5

[34] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 11

[35] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022. 7

[36] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 7

[37] Safa C. Medin, Bernhard Egger, Anoop Cherian, Ye Wang, Joshua B. Tenenbaum, Xiaoming Liu, and Tim K. Marks. MOST-GAN: 3D morphable StyleGAN for disentangled face image manipulation. In *AAAI*, 2022. 1

[38] Lakshmanan Nataraj, Tajuddin Manhar Mohammed, BS Manjunath, Shivkumar Chandrasekaran, Arjuna Flenner, Jawadul H Bappy, and Amit K Roy-Chowdhury. Detecting GAN generated fake images using co-occurrence matrices. *Electronic Imaging*, 2019:532–1, 2019. 3, 7

[39] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In *AMDO*, 2018. 7, 11

[40] Huy H Nguyen, Fuming Fang, Junichi Yamagishi, and Isao Echizen. Multi-task learning for detecting and segmenting manipulated facial images and videos. In *BTAS*, 2019. 1, 2, 3

[41] Ori Nizan and Ayellet Tal. Breaking the cycle - colleagues are all you need. In *CVPR*, 2020. 7, 11

[42] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. GauGAN: semantic image synthesis with spatially adaptive normalization. In *ACM*, 2019. 1, 7, 11

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 7

[44] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 7, 11

[45] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *CVPR*, 2020. 7, 11

[46] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *ECCV*, 2018. 7, 11

[47] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 11

[48] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1

[49] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *CVPR*, 2019. 1, 3

[50] Nataniel Ruiz, Sarah Adel Bargal, and Stan Sclaroff. Disrupting deepfakes: Adversarial attacks against conditional image translation networks and facial manipulation systems. In *ECCV*, 2020. 1, 2

[51] Eran Segalis and Eran Galili. OGAN: Disrupting deepfakes with an adversarial attack that survives training. *arXiv preprint arXiv:2006.12247*, 2020. 2

[52] Kaede Shiohara and Toshihiko Yamasaki. Detecting deepfakes with self-blended images. In *CVPR*, 2022. 1, 3

[53] Kritaphat Songsri-in and Stefanos Zafeiriou. Complement face forensic detection and localization with facial landmarks. *arXiv preprint arXiv:1910.05455*, 2019. 1, 2, 3

[54] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017. 1, 7, 11

[55] Radim Tyleček and Radim Šára. Spatial pattern templates for recognition of objects with regular structure. In *GCPR*, 2013. 11

[56] Run Wang, Felix Juefei-Xu, Meng Luo, Yang Liu, and Lina Wang. FakeTagger: Robust safeguards against deepfake dissemination via provenance tracking. In *ACMM*, 2021. 1, 2

[57] Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. Sketch your own gan. In *ICCV*, 2021. 1

[58] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. CNN-generated images are surprisingly easy to spot... for now. In *CVPR*, 2020. 3, 7

[59] Xiaogang Wang and Xiaoou Tang. Face photo-sketch synthesis and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31:1955–1967, 2008. 11

[60] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-ESRGAN: Training real-world blind super-resolution with pure synthetic data. In *CVPR*, 2021. 7, 11

[61] Xi Wu, Zhen Xie, YuTao Gao, and Yu Xiao. SSTNET: Detecting manipulated faces through spatial, steganalysis and temporal features. In *ICASSP*, 2020. 1

[62] Ying Xu, Kiran Raja, and Marius Pedersen. Supervised contrastive learning for generalizable and explainable deepfakes detection. In *WACV*, 2022. 1, 3

[63] Chin-Yuan Yeh, Hsi-Wen Chen, Shang-Lun Tsai, and Sheng-De Wang. Disrupting image-translation-based deepfake algorithms with adversarial attacks. In *WACVW*, 2020. 1, 2

[64] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. DualGAN: Unsupervised dual learning for image-to-image translation. In *CVPR*, 2017. 7, 11

[65] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in GAN fake images. In *WIFS*, 2019. 3, 7, 11

[66] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 7, 11

[67] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NeurIPS*, 2017. 7, 11

[68] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. SEAN: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020. 7, 11

MaLP: Manipulation Localization Using a Proactive Scheme – Supplementary material –

1. Implementation Details

Experimental Setup and Hyperparameters We train MaLP for $150,000$ iterations with a batch size of 4. For all of the networks, we use Adam optimizer except for the transformer which uses AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay $0.5e^{-5}$ and eps $1e^{-8}$. The learning rate is $1e^{-5}$ for all networks. The constraint weights are set as: $\lambda_1 = 100, \lambda_2 = 5, \lambda_3 = 4, \lambda_4 = 25, \lambda_5 = 25, \lambda_6 = 25, \lambda_7 = 50, \lambda_8 = 15, \lambda_9 = 20, \lambda_{10} = 50$. We use a template set size of 1 and template strength as 30% unless mentioned. All experiments are conducted on one NVIDIA K80 GPU.

Network Architecture. We show the network architecture of various components of MaLP in Fig. 1. The shared network consists of 1 stem convolutional layer and 4 convolution blocks. Each convolution block consists of convolutional and batch normalization layers followed by ReLU activation. The output of the shared network is given to $\mathcal{E}_E$ and $\mathcal{E}_C$, both having the same architecture with 3 convolution blocks and 1 stem convolutional layer. We use the transformer $\mathcal{E}_T$ in the second branch of the framework where the ViT [13] architecture is adopted. The transformer consists of 6 encoder blocks, and a dropout of 0.1 is used. The features of the transformer are reshaped to the shape of the fakeness map *i.e.* $1 \times 128 \times 128$. Finally, we use a classifier $\mathcal{C}$ on the predicted fakeness maps to perform real *vs.* fake binary classification. The classifier has 8 convolution blocks, 1 stem convolutional layer, and 3 fully connected layers. We apply the ReLU activation between the layers.

GMs and dataset license information. We use a variety of face and generic GMs to show the effectiveness of MaLP. The information for all the GMs along with their training datasets, is shown in Tab. 1. For many GMs used by [1], We use the test images released by [1]. for the remaining GMs, we would release the test images for fair comparison of generalization benchmark by the future works. We also show more visualization samples of the predicted fakeness maps by MaLP in Fig. 2- 5. All the fakeness maps are shown in "pink" cmap for better representation. We also indicate the cosine similarity between the predicted and ground truth fakeness maps. We observe that the fakeness maps for encrypted images have minimal bright regions. However, for fake images, MaLP is able to localize the modified regions well, considering the modified attributes/GMs are unseen in training.

Table 1. List of GMs along with their training datasets

Dataset	GMs
CelebA [34]	STGAN [30], AttGAN [21], StarGAN [9], GANimation [46], CouncilGAN [41], ESRGAN [60], GDWCT [7]
CelebA-HQ [25]	SEAN [68], StarGAN-v2 [10], ALAE [45], DRGAN [54], ColorGAN [39],
Facades [55]	CycleGAN [66], BicycleGAN [67], Pix2Pix [24]
COCO [3]	GauGAN [42]
Horse2Zebra [66]	AutoGAN [65]
Summer2Winter [66]	DRIT [28]
GTA2CITY [47]	UNIT [31]
Edges2Shoes [24]	MUNIT [22]
Paris Street-view [41]	Cont_Enc [44]
Sketch-Photo [59]	DualGAN [64]

The face datasets include CelebA [34] and CelebA-HQ [25], both of which don't have any associated Institutional Review Board (IRB) approval. The authors for both datasets mention the availability of the dataset for non-commercial research purposes, which we strictly adhere to. For generic images datasets, we use Facades [55], COCO [3], Horse2Zebra [66], Summer2Winter [66], GTA2CITY [47], Edges2Shoes [24], Paris street-view [44] and Sketch-Photo [59] datasets. All the mentioned generic image datasets can be used for non-commercial research purposes, as mentioned by the authors, and we use the datasets for the same purposes.

Image Editing Degradations. We apply several image editing degradations to the test set to verify the robustness of MaLP. The details of these operations are listed below:

1. JPEG compression: We compress the image with the compression quality of 50%.
2. Blur: We apply the Gaussian blur with a filter size of 7×7.
3. Noise: We apply a Gaussian noise with zero mean and unit variance.
4. Low-resolution: We resize the image to half the original resolution and restore it back to the original resolution using linear interpolation.

Potential Societal Impact The problem of manipulation localization is crucial from the perspective of media forensics. Localizing the fake regions not only helps in the detection of these fake media but, in the future, can also help recover the original image that the GM has manipulated.

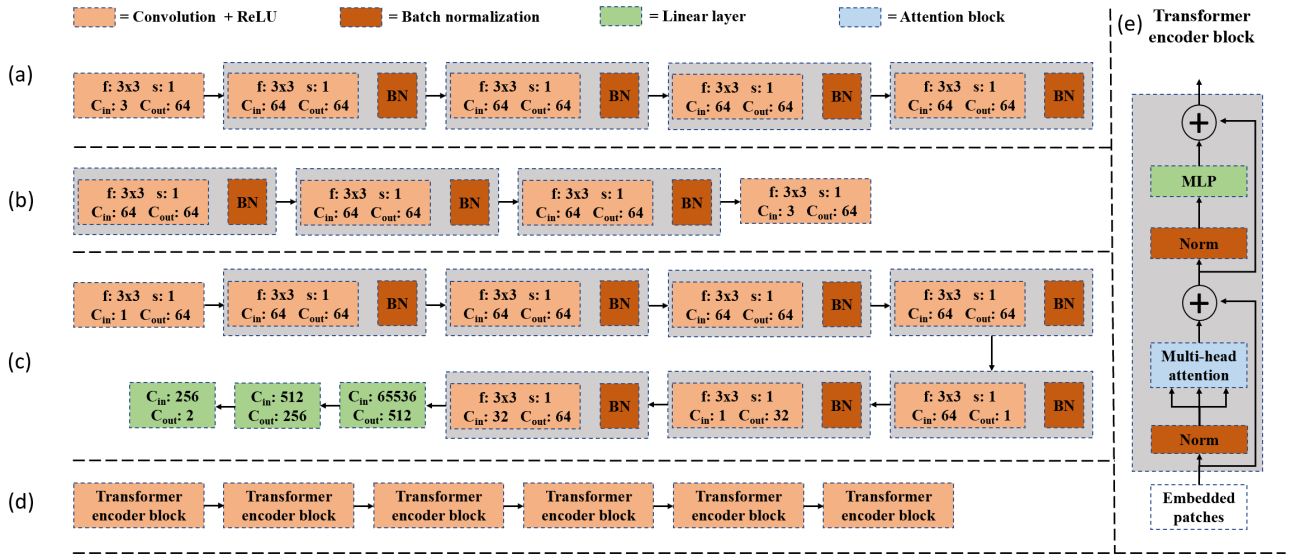


Figure 1. Network architecture for different components of MaLP. (a) Shared network, (b) Encoder $\mathcal{E}_E$ and CNN network $\mathcal{E}_C$, (c) Classifier $\mathcal{C}$, (d) Transformer $\mathcal{E}_T$, and (e) Transformer encoder block.

Table 2. Ablation for localization loss.

Loss	CS ↑	PSNR ↑	SSIM ↑
CS	0.9356	22.16	0.7114
CS + L_2	0.9230	18.98	0.6614
CS + SSIM + L_2	0.9211	19.12	0.6816
CS + SSIM + L_1	0..8777	14.01	0.3712
CS + SSIM	**0.9394**	**23.020**	**0.7312**

We also show that MaLP can be used as a discriminator to improve the quality of GMs. While this is an interesting application of MaLP, it can be a possibility that the GMs become more robust to our framework, decreasing the localization performance if the training of the GM is done from scratch.

2. Additional Experiments

Localization Loss. We show the importance of manipulation loss (defined in Eq. 8) in Sec. 4.6. We perform an ablation to formulate the loss of fakeness maps for manipulated images. As shown in Tab. 2, we try experimenting with various loss functions *i.e.* cosine similarity (CS), L_1, L_2 and structural similarity index measure (SSIM). Using just the CS loss results in better performance compared to combining it with L_1 or L_2 loss. We observe a huge deterioration in performance when using L_1 loss. This can be explained as PSNR and SSIM are directly related to mean squared error which is optimized by either an L_2 or SSIM loss. Finally, adopting an SSIM loss with CS loss results in a better performance as both of them are more related to the metrics, making it easier for MaLP to converge.

Comparison with Baseline. Due to the limited GPU memory, we conduct proactive training with one GM only because the GM needs to be loaded to the memory and used on the fly. On the other hand, passive methods can be trained on multiple GMs because the image generation processes are conducted offline. As shown in Tab. 3, [23] trains on images manipulated by 7 different GMs, unlike MaLP, which is trained on images manipulated by only 1 GM. We show the performance on three GMs, which are seen for [23], but unseen for MaLP. MaLP performs better even though these GMs' images are not seen in training. Therefore, even though the training of MaLP is limited by 1 GM, it can achieve better generalization to other GMs proving the effectiveness of proactive schemes.

Table 3. Comparison with [23] using multiple GMs in training. MaLP is able to outperform [23] by training images manipulated by only STGAN.

Method	Training GMs	Cosine similarity ↑		
		AttGAN	StarGAN	StyleGAN
Hunag *et al.* [23]	STGAN + ICGAN + PGGAN + StyleGAN + StyleGAN2 + StarGAN + AttGAN	0.6940	0.8494	0.7479
MaLP	STGAN	**0.8557**	**0.8718**	**0.8255**

Table 4. Performance of MaLP across different attribute modifications seen in training.

Method	Cosine similarity ↑					
	Bald	Bangs	Black Hair	Eyeglasses	Mustache	Smile
[23]	0.9014	0.8850	0.8817	0.9093	0.9152	0.8634
MaLP	**0.9478**	**0.9329**	**0.9367**	**0.9549**	**0.9470**	**0.9489**

Multiple Attribute Modifications. Instead of training on bald attribute modification by STGAN, we train and test MaLP on multiple attribute modifications. These include bald, bangs, black hair, eyeglasses, mustache, and smile manipulation. We show the results in Tab. 4. MaLP per-

Table 5. Ablation study for transformer architecture.

Optimizer	Depth	Dropout	Cosine similarity↑	Accuracy↑
Adam	6	0.1	0.8839	0.9514
AdamW	1	0.0	0.8825	0.9647
AdamW	1	0.0	0.8826	0.9680
AdamW	3	0.0	0.8830	0.9705
AdamW	6	0.1	**0.8848**	**0.9856**

forms better for all the attribute modifications compared to the passive method [23]. We also observe an increase in cosine similarity compared to when MaLP is trained on only bald attribute modification. This is expected, as the more types of modifications MaLP sees in training, the better it learns to localize.

Transformer Architecture Ablation. We ablate various parameters of the transformer to select the best architecture for manipulation localization. We experiment with parameters that include optimizer, depth *i.e*. number of blocks, and dropout. We only use the transformer branch and switch off the CNN branch during training. The results are shown in Tab. 5. We observe that the localization performance is almost the same when using the transformer to predict fakeness maps. However, the detection accuracy has a significant impact. Having dropout does increase the performance for detection and localization. Further, using the weighted Adam optimizer is more beneficial than using the vanilla Adam optimizer. Therefore, we adopt the architecture of the transformer with 6 blocks and optimize it with a weighted Adam optimizer. Finally, we also include the dropout to achieve the best performance for localization and detection.

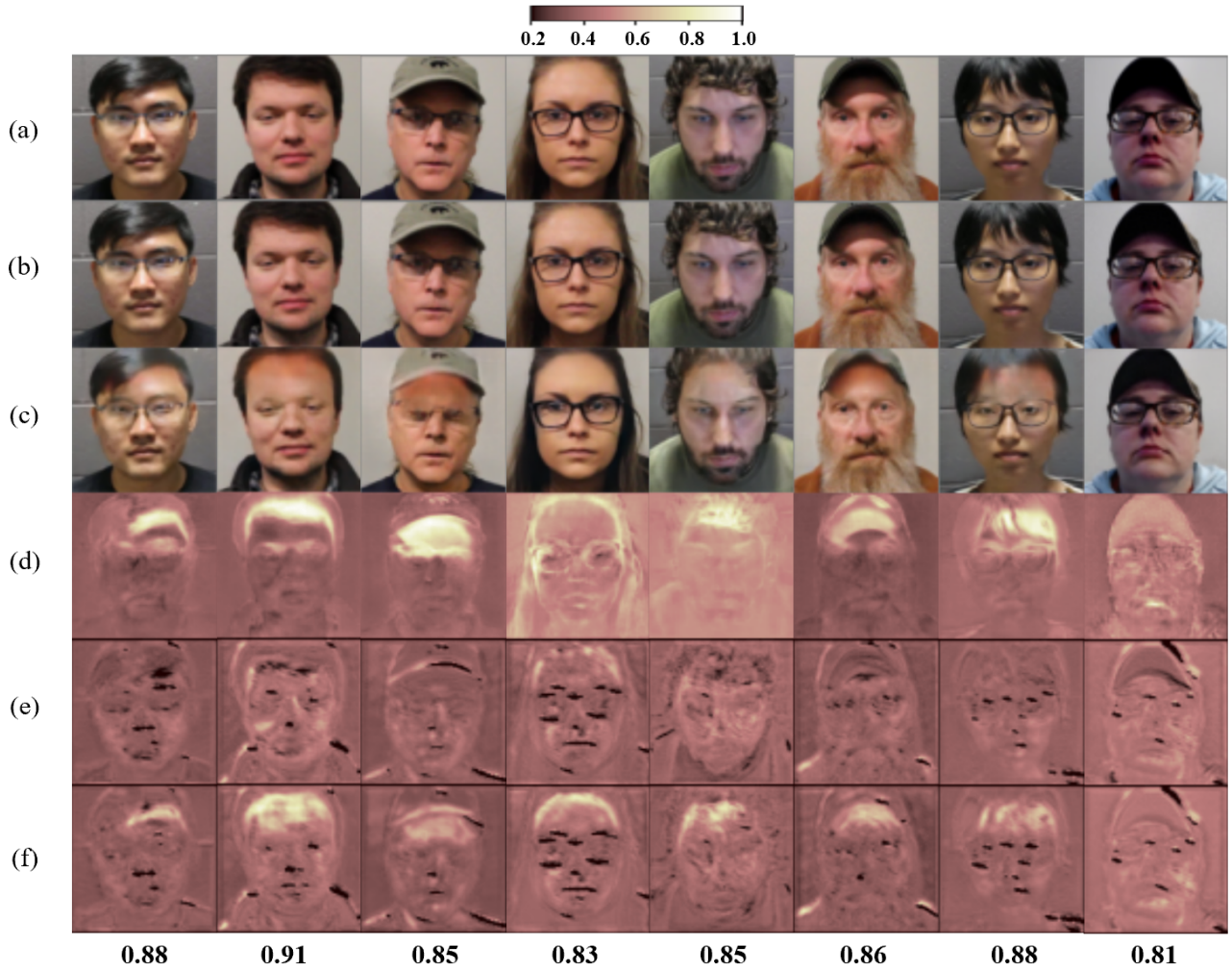


Figure 2. Visualization of fakeness maps for different attribute modifications by STGAN. (a) Real image, (b) encrypted image, (c) manipulated image, (d) ground-truth $\boldsymbol{M}_{GT}$, (e) predicted fakeness map for encrypted images, and (f) predicted fakeness map for manipulated images. We also show the cosine similarity between the predicted and ground-truth fakeness map below (f). All face images come from SiWM-v2 data [19].

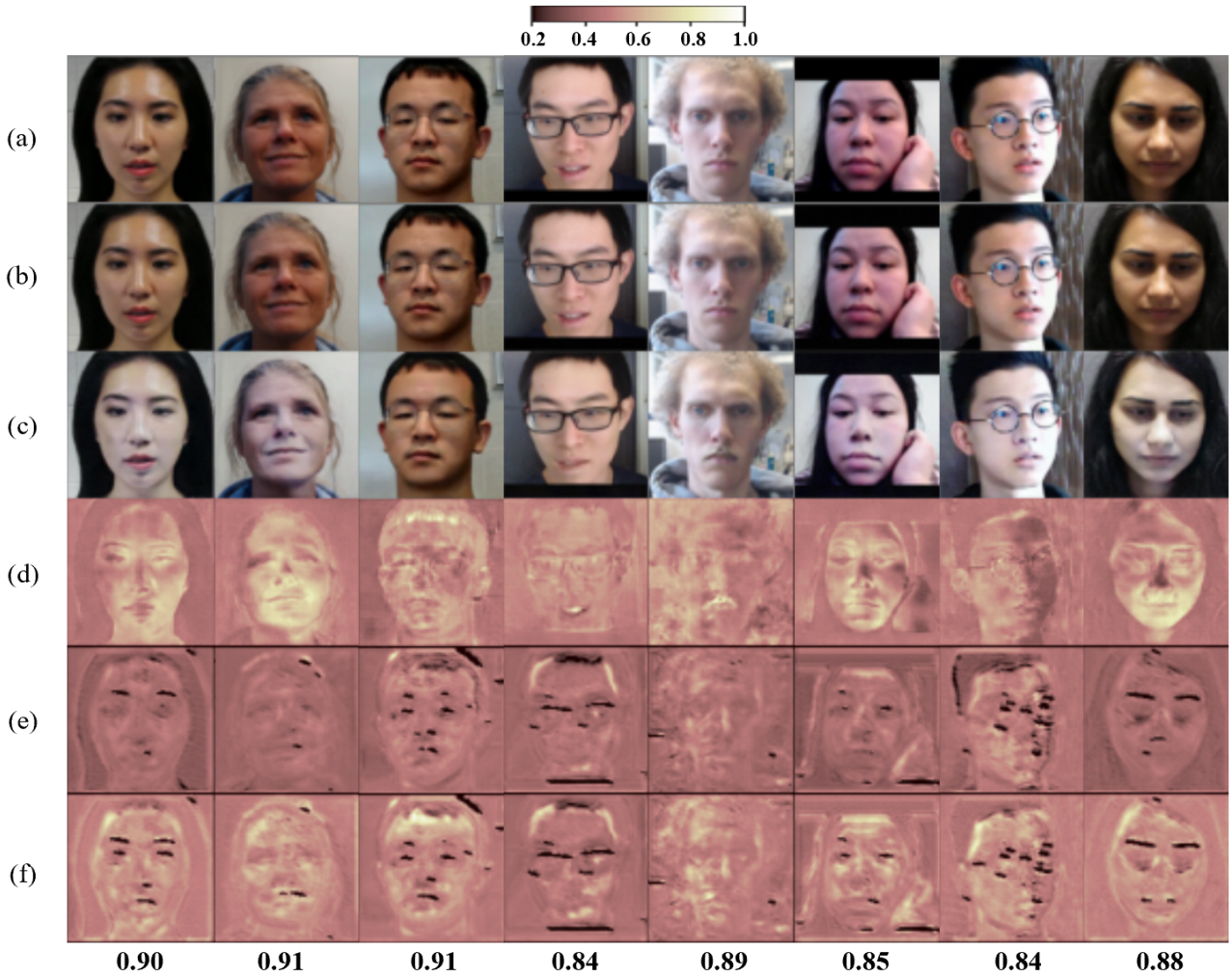


Figure 3. Visualization of fakeness maps for different attribute modifications by STGAN. (a) Real image, (b) encrypted image, (c) manipulated image, (d) ground-truth $\boldsymbol{M}_{GT}$, (e) predicted fakeness map for encrypted images, and (f) predicted fakeness map for manipulated images. We also show the cosine similarity between the predicted and ground-truth fakeness map below (f). All face images come from SiWM-v2 data [19].

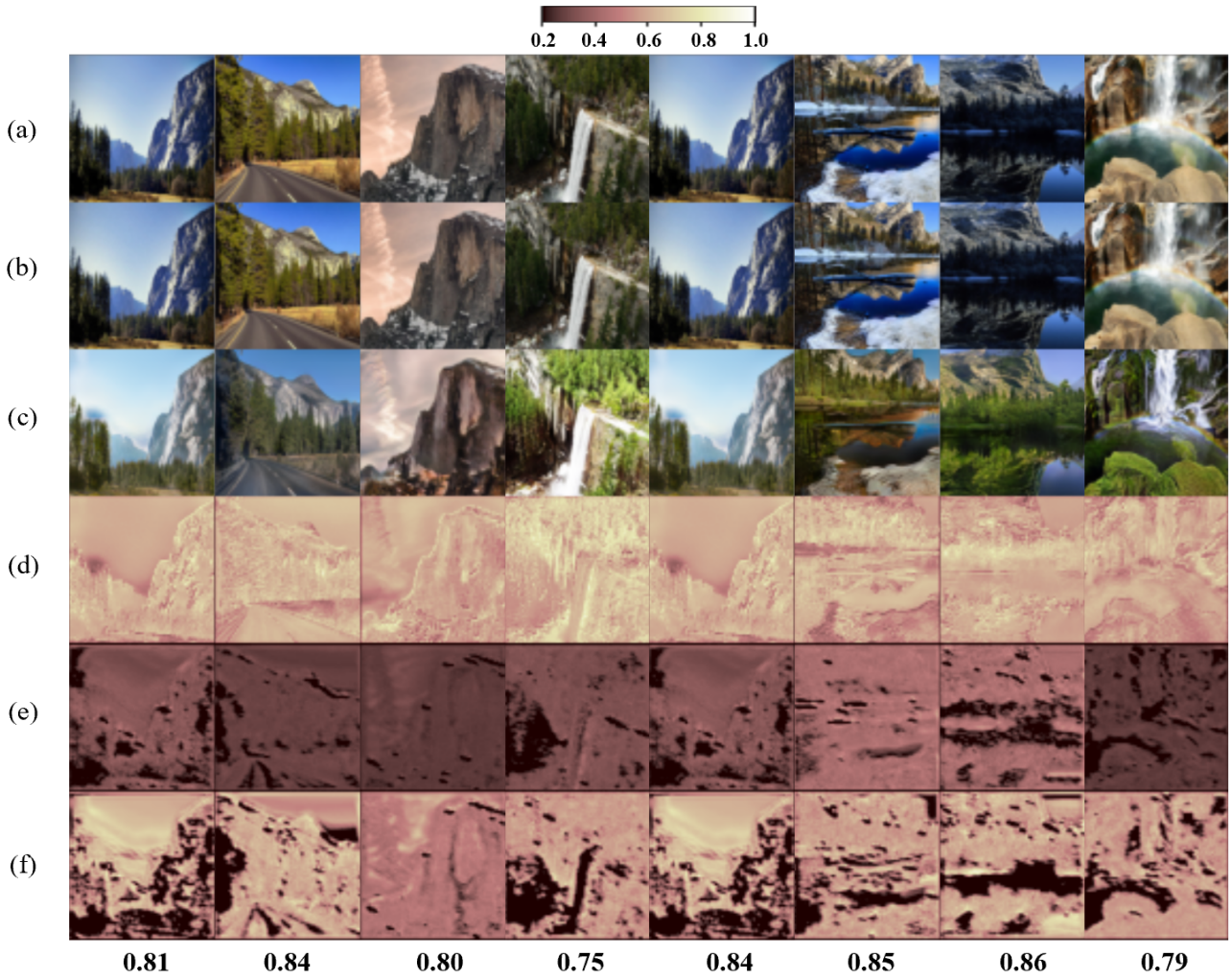


Figure 4. Visualization of fakeness maps for manipulation by DRIT. (a) Real image, (b) encrypted image, (c) manipulated image, (d) ground-truth $\boldsymbol{M}_{GT}$, (e) predicted fakeness map for encrypted images, and (f) predicted fakeness map for manipulated images. We also show the cosine similarity between the predicted and ground-truth fakeness map below (f).

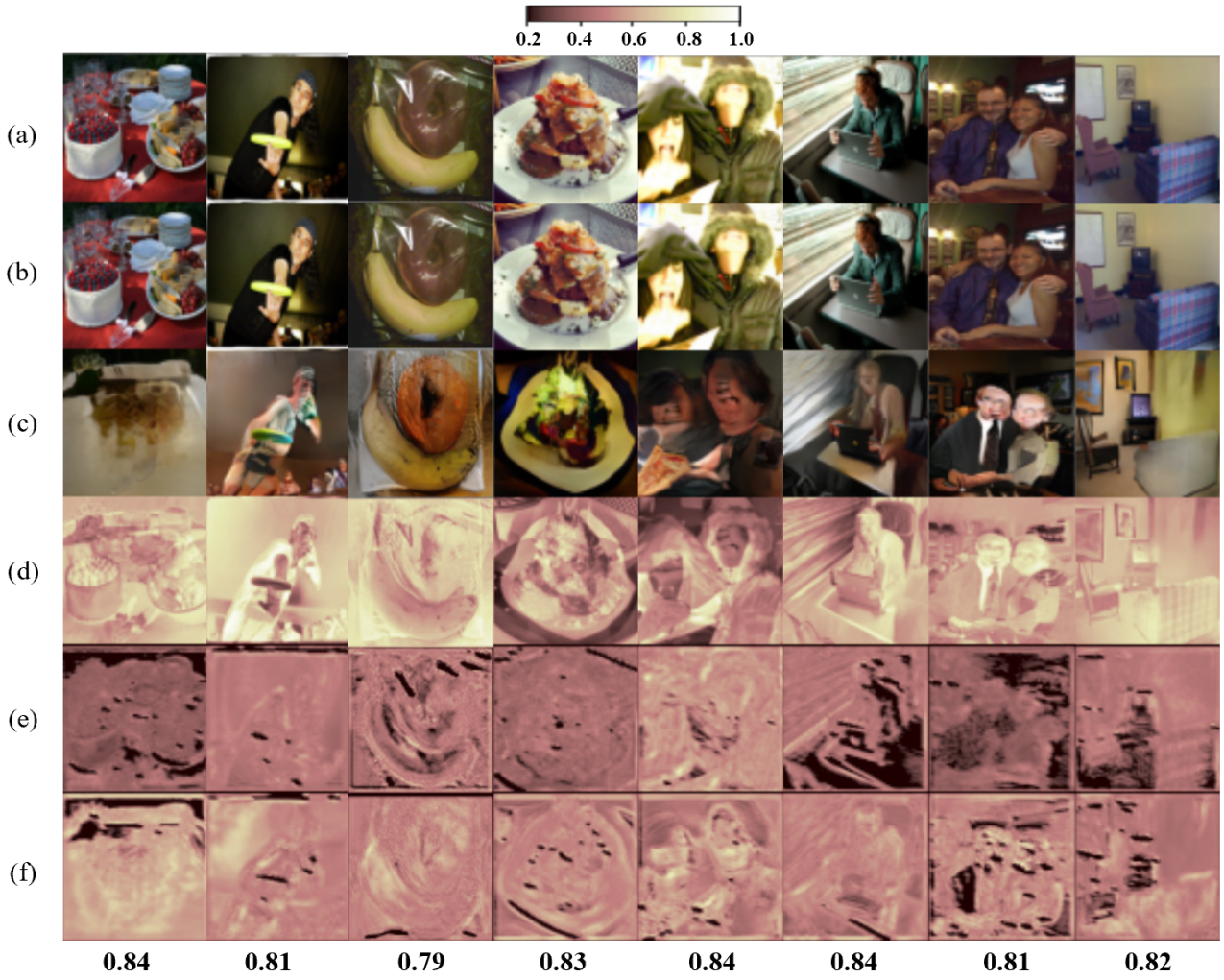


Figure 5. Visualization of fakeness maps for manipulation by GauGAN. (a) Real image, (b) encrypted image, (c) manipulated image, (d) ground-truth $\boldsymbol{M}_{GT}$, (e) predicted fakeness map for encrypted images, and (f) predicted fakeness map for manipulated images. We also show the cosine similarity between the predicted and ground-truth fakeness map below (f).