FakeLocator: Robust Localization of GAN-Based Face Manipulations

Yihao Huang, Felix Juefei-Xu, *Member, IEEE*, Qing Guo, *Member, IEEE*, Yang Liu, *Senior Member, IEEE*, and Geguang Pu

***Abstract*— Full face synthesis and partial face manipulation by virtue of the generative adversarial networks (GANs) and its variants have raised wide public concerns. In the multi-media forensics area, detecting and ultimately locating the image forgery has become an imperative task. In this work, we investigate the architecture of existing GAN-based face manipulation methods and observe that the imperfection of upsampling methods therewithin could be served as an important asset for GAN-synthesized fake image detection and forgery localization. Based on this basic observation, we have proposed a novel approach, termed *FakeLocator*, to obtain high localization accuracy, at full resolution, on manipulated facial images. To the best of our knowledge, this is the very first attempt to solve the GAN-based fake localization problem with a gray-scale fakeness map that preserves more information of fake regions. To improve the universality of *FakeLocator* across multifarious facial attributes, we introduce an attention mechanism to guide the training of the model. To improve the universality of *FakeLocator* across different DeepFake methods, we propose partial data augmentation and single sample clustering on the training images. Experimental results on popular FaceForensics++, DFFD datasets and seven different state-of-the-art GAN-based face generation methods have shown the effectiveness of our method. Compared with the baselines, our method performs better on various metrics. Moreover, the proposed method is robust against various real-world facial image degradations such as JPEG compression, low-resolution, noise, and blur.**

***Index Terms*— DeepFake, face manipulation, DeepFake detection and localization.**

I. Introduction

EVERY day we receive newsletters from the media channels such as television, social media, newspaper, *etc*. Limited by the presentation of these media, compared with text descriptions, the information with images and videos is prone to be accepted by humans and thus becomes more trustworthy to us. However, with the development of digital manipulation technologies, even videos can be synthesized at a small price. In recent years, a lot of synthetic videos represented by celebrities [2] were produced by a series of techniques that can produce fake images, audios, and videos, collectively called DeepFakes [3], [4]. DeepFake has been widely used in politics and pornography [5], [6]. Fake images can be created easily for that many free tools are available to us. Public concerns about fraud and credibility problems have been raised by such misinformation. Hence, it is urgent to study and investigate effective and robust methods in face forgery detection and forensics, for achieving a safe and responsible multi-media environment.

The facial images synthesized by GAN-based methods are more authentic than other methods. Due to the potential security and privacy issues of synthesized facial images, researchers have a great interest in detecting images generated by GAN-based methods or defending fake generation by tagging the images [7]. Recently, many studies have worked on classifying images with various methods [4], [8]–[26]. In these methods, most of them use deep neural networks (DNNs) to deal with the DeepFake detection problem while a few explore other ways. Among the methods using DNNs, the networks used by them can be directly classified into convolutional neural networks (CNNs) and recurrent neural networks (RNNs). In our investigation, [17] and [20] use RNN and take images as the input of the network. Similarly, [9], [10], [13]–[16], [21]–[24] also take images as input while using CNNs as the backbone networks. In methods that use CNNs, there are three categories based on the input used by them. References [8], [12] consider that DeepFake methods will leave fingerprints in the fake images and they take the fingerprint and images as the input in detection. References [25], [26] take the spectrum of images as the input of CNN. References [11] uses the activated neuron feature of the given images as the input of DNN to detect

Manuscript received 23 February 2021; revised 1 October 2021; accepted 10 November 2021. Date of publication 7 January 2022; date of current version 27 July 2022. This work was supported in part by the National Key Research and Development Program under Grant 2020AAA0107800; in part by the Shanghai Collaborative Innovation Center of Trusted Industry Internet Software; in part by NSFC under Project 61632005 and Project 61532019; in part by the National Research Foundation, Singapore, under its AI Singapore Program, under Grant AISG2-RP-2020-019; in part by the National Research Foundation, Prime Ministers Office, Singapore, under its National Cybersecurity Research and Development Program, under Grant NRF2018NCR-NCR005-0001; in part by NRF Investigatorship under Grant NRFI06-2020-0001; in part by the National Research Foundation through its National Satellite of Excellence in Trustworthy Software Systems (NSOE-TSS) Project under the National Cybersecurity Research and Development (NCR) under Grant NRF2018NCR-NSOE003-0001. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Andrew Beng Jin Teoh. *(Corresponding author: Geguang Pu.)*

Yihao Huang is with the School of Software Engineering, East China Normal University, Shanghai 200050, China.

Felix Juefei-Xu is with Alibaba Group, Sunnyvale, CA 94085 USA.

Qing Guo is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798.

Yang Liu is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, and also with the School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 314423, China.

Geguang Pu is with the School of Software Engineering, East China Normal University, Shanghai 200050, China, and also with Shanghai Industrial Control Safety Innovation Technology Company Ltd., Shanghai 200331, China (e-mail: ggpu@sei.ecnu.edu.cn).

Digital Object Identifier 10.1109/TIFS.2022.3141262

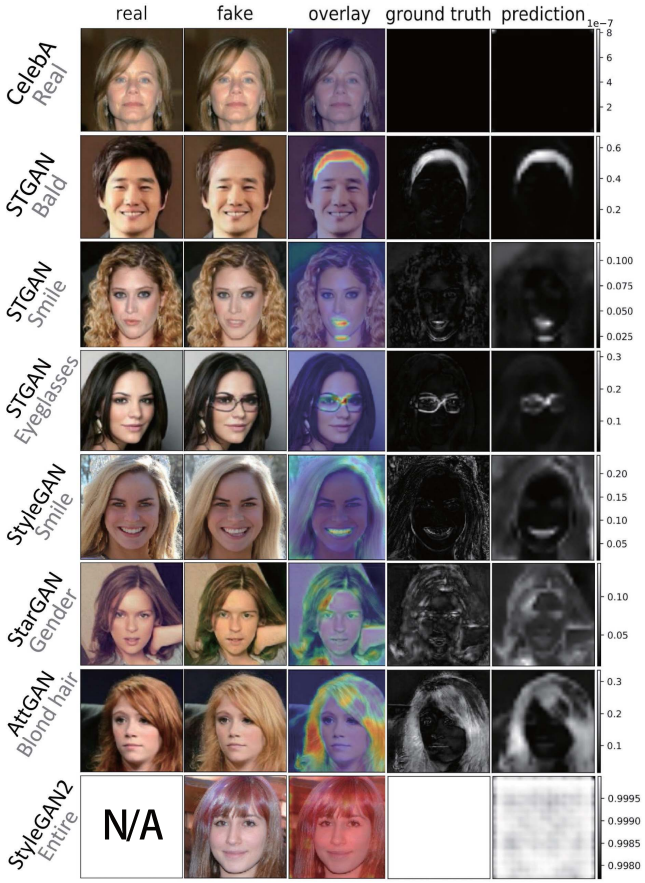


Fig. 1. **Fake region localization results.** These are the results of different GANs and properties. In the left comments, CelebA [1] is a real image database and others are GAN-based face generation methods. The gray text represents the facial property. **Fake** image is produced by manipulating corresponding **real** image through GAN-based face generation method. **Ground truth** is calculated by **fake** image and **real** image. **Fake** image and **ground truth** are the input and expected output of our method. **Overlay** is combined of **prediction** and **fake** image. The **prediction** has colorbar which shows the value range of pixels. For the first row which uses real image of CelebA as input, for unity of the figure, we also regard it as **fake** image.

DeepFake images. In addition to these DNN-based detection methods, [18] uses the visual artifacts as the clue. References [19] models facial expressions and movements that typify an individual's speaking pattern.

However, none of them considers locating fake regions of the fake images where modifications of facial properties commonly occur. Localization is more significant and valuable in the research field of multimedia forensics. In multi-media forensics, a good localization method would better satisfy the following requirements. (1) The localization map is of high resolution with fine-grained fake regions represented (high-resolution). Because it is important to get a good visualization in real forensics scenarios. (2) The method is robust to degradations (robustness), which is important for locators to be deployed in the wild. Because the concept of method robustness is too wide and we can not guarantee to verify all the aspects of method robustness. Thus the "robustness" mentioned in the following content mainly represents the degradation robustness. (3) The method is universal enough to tackle unknown facial properties (cross-attribute universality) and unknown GAN methods (cross-method universality). Because the concept of method universality is too wide and we can not guarantee to verify all the aspects of method universality. Thus the "universality" mentioned in the following content mainly represents the cross-attribute and cross-method universality.

The generators in GAN-based face generation methods are typical encoder-decoder architecture with an upsampling design in its decoder. The upsampling design is used to magnify the feature maps produced by the encoder to be a colorful image. However, the upsampling design may introduce special features into the synthesized images. According to our investigation, there are only three kinds of upsampling methods. The textures produced by all these upsampling methods contain special features, which we refer to as **fake texture**. We observe that the **fake texture** can not only be used for fake detection but also used for fake localization. Thus we propose a universal pipeline that is one of the first attempts to solve the fake localization problem. As an improvement, we also introduce attention mechanism into the architecture to learn **fake texture** better and generalize our approach to unseen facial properties.

FaceForensics++ [2], DFFD [27], UADFV [28], and Celeb-DF [29] are widely used in DeepFake research. Thus we conduct experiments on them and compare with other state-of-the-art (SOTA) fake detection methods and fake localization methods. Furthermore, we build our own dataset with available GANs to generate high-quality forgery images for evaluating the effectiveness and robustness of our proposed method. To the best of our knowledge, only [27] and [30] have been working on the same topic as ours. Therefore, we select their methods as the baselines in our experiment. For [27], in the literature, the authors insert an attention map module into a classifier such as Xception [31] to obtain the location of fake regions. However, the resolution of the attention map is constrained by their design and can only output a tiny map. For example, an image of size 299×299 obtains an attention map of size 19×19. So the attention map can not exactly pinpoint the fake regions. For [30], we compare with them on several GAN-based face generation methods. All the localization methods mentioned above do not satisfy high-resolution, universality, and robustness simultaneously. In addition, the fakeness maps of all these localization methods are not suitable for localization task, which will be introduced and improved in our method.

The main contributions are summarized as follows.

- We have a new observation that the artifact (shown in Fig. 2) inducted by GAN-based face generation methods could be used in DeepFake forensics including detection and localization. The pipeline proposed by us can locate the manipulated facial regions effectively at full resolution.
- To improve the cross-attribute universality of the model, we introduce the attention mechanism into our framework by using face parsing. To improve the cross-method universality of the model, we propose partial data augmentation and single sample clustering to enhance the training data. The fake textures are captured by the gray-scale fakeness map proposed by us.
- Experiments are conducted on two popular DeepFake datasets (*ie.*, FaceForensics++ and DFFD) and seven

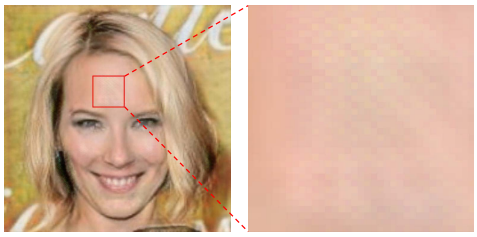


Fig. 2. Artifact introduced by GAN-based face generation methods. Here we demonstrate the typical one (*ie.*, checkerboard pattern). The fake image is produced by StarGAN [32].

SOTA GAN-based face generation methods. Experimental results show that our approach outperforms prior works [27] and [30] in locating fake regions. Furthermore, our method is also robust against various real-world facial image degradations.

II. Related Work

A. GAN-Based Face Generation

GAN has drawn attention from both academia and industry since it was first proposed in 2014 [33]. The GAN-based face generation methods can be classified into two categories: entire face synthesis and partial face manipulation methods. Here we will introduce seven state-of-the-art GAN-based face generation methods. IcGAN [34], AttGAN [35], StarGAN [32], and STGAN [36] are partial face manipulation methods, PGGAN [37] and StyleGAN2 [38] are entire face synthesis methods. StyleGAN [39] is not only a partial face manipulation method but also an entire face synthesis method.

IcGAN [34] introduces the encoder that allows the network to reconstruct and modify real face images with arbitrary attributes. PGGAN [37] proposes progressively growing on both the generator and the discriminator to obtain large high-resolution images. StarGAN [32] simply uses a single model to perform image-to-image translations for multiple facial properties. AttGAN [35] applies an attribute classification constraint to the generated images to guarantee the correct change of desired attributes. STGAN [36] simultaneously improves attribute manipulation accuracy as well as perception quality on the basis of AttGAN. StyleGAN [39] proposes a new generator to learn unsupervised separation of high-level attributes and stochastic variation in the generated images. Recently, StyleGAN2 [38] fixes the imperfection of StyleGAN to improve image quality.

These seven SOTA GAN-based methods typically represent GAN-based entire face synthesis and partial face manipulation methods. Thus we verify the effectiveness of our method on these seven GAN-based methods. In the following sections, **seven GAN-based face generation methods** are referred to as the GANs introduced here, unless particularly addressed.

B. Manipulated Face Localization

Although there are a lot of DeepFake detection methods, only several works have been proposed on the manipulated face localization problem. Songsri-in and Zafeiriou [40] propose an architecture to predict face forensic localization.

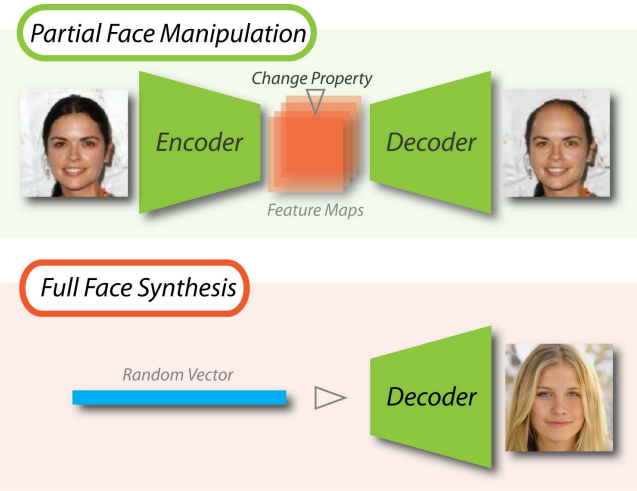


Fig. 3. The architecture of GAN-based face generation methods. The top subplot shows how partial face manipulation is carried out. The bottom subplot shows how the entire face synthesis is carried out.

Nguyen *et al.* [41] use a multi-task learning approach to simultaneously detect manipulated videos and locate the manipulated regions. Li *et al.* [42] propose face X-ray to locate the fake regions. However, the method fails when the image is entirely synthetic. Furthermore, the datasets used by them are videos that focus on face swap. The fake regions are very large and easy to locate. We mainly focus on locating modified facial properties. Our task is much harder than theirs due to the small fake regions.

Only Joel *et al.* [26] and Chai *et al.* [30] have been working on the same topic as us. Dang *et al.* [27] present the first and only technique that applies the attention mechanism to address the problem. The attention map is advantageous to be added into arbitrary networks and helps to improve detection accuracy. However, the attention map is too small to point out the fake regions in the fine-grained level. Chai *et al.* [30] propose a patch-based classifier with limited receptive fields to visualize fake regions of the images. However, their method puts too much emphasis on local artifacts and ignores global ones.

To sum up, all the localization methods above do not satisfy high-resolution, universality, and robustness simultaneously. Furthermore, the fakeness maps used by these methods may lose information about fake regions.

III. Imperfection of GAN-Based Methods

A. General Architecture of GAN-Based Methods

There are mainly two ways to generate facial images: entire face synthesis and partial face manipulation. We call these two methods **face generation methods**. Their typical architecture is based on the encoder-decoder framework shown in Fig. 3.

B. Imperfection of Upsampling

Upsampling is a technique that can improve image resolution. There are three kinds of upsampling methods: unpooling, transposed convolution, and interpolation, as shown in Fig. 4.

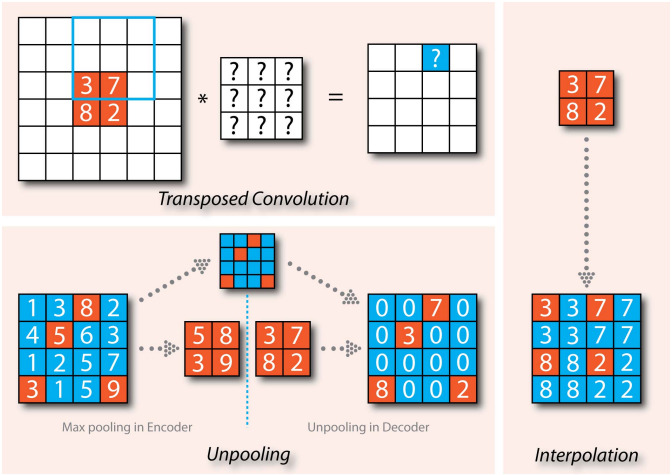


Fig. 4. Upsampling methods. Transposed convolution is similar to convolution. It results in the checkerboard texture of the output. In interpolation, the inserted pixels are calculated by the existing pixels. Here we show the nearest neighbor interpolation. Unpooling simply uses zero to fill the inserted pixels, which also produces fake textures.

For GAN-based methods, the most commonly used interpolations are nearest neighbor interpolation, bilinear interpolation, and bicubic interpolation. In Fig. 4 we just show the nearest neighbor interpolation as an example.

By analyzing the official implementation of **seven GAN-based face generation methods**, we find that only transposed convolution and interpolation have been used. IcGAN uses interpolation and others use transposed convolution.

Although only two methods are used, all of these three methods have been proved to induce fake texture. Google Brain [43] has proved that the transposed convolution results in the checkerboard texture of the output image. [25] has shown that the transposed convolution and nearest neighbor interpolation have fake texture. Though not mentioned explicitly, the process of their proof also points out that unpooling produces fake textures. For the remaining bilinear and bicubic interpolations, they bring periodicity into the second derivative signal of images [44]. This means that the interpolated images exhibit fake textures which can be detected by convolution kernel, for instance, Laplace operator.

IV. Methodology

The fake texture produced by upsampling methods is totally different from the real texture in the real image. In this section, we firstly present the framework for face manipulation forensics by leveraging the imperfection of upsampling design in GANs.

Then, we propose the gray-scale fakeness map to visualize the manipulated regions in fake images. The gray-scale fakeness map is more informative than that of other localization methods. To ensure the effectiveness of the gray-scale fakeness map, we adopt a suitable loss function in Sec. IV-E.2.

Finally, we introduce attention mechanism to generalize our approach in tackling unseen facial properties. We also propose partial data augmentation and single sample clustering to improve the performance on unseen GAN-based fake image generation methods. The pipeline proposed by us produces a full resolution fakeness map. We also verify the robustness and universality of our method in Sec. V.

Please note that since it is almost inevitable for GAN-based face generation methods to use upsampling methods and our method is designed for the imperfection of the upsampling methods, thus our method can be used for localization tasks on most of the GAN-based face generation methods. Furthermore, within a better encoder-decoder network, our method will achieve better performance on fake localization problems.

A. Problem Formulation

For fake localization tasks, the main target is to output a fakeness prediction map. We define the problem as below. The input is a fake image $\mathbf{X}'$ while the output is a fakeness prediction map $\mathbf{M}_{\text{Pred}}$. The solution is a function $\mathcal{F}(\cdot)$.

$$\mathbf{M}_{\text{Pred}} = \mathcal{F}(\mathbf{X}'). \tag{1}$$

Furthermore, we think that the method should satisfy three properties. First, the output fakeness prediction maps should have the same resolution as the fake counterparts. Second, as we mainly locate the fake regions of the face, the cross-attribute universality of the method on different facial properties is necessary. That means, if we use images that change hair color to train a model, the model should have the ability to locate the fake regions of other facial properties manipulated. Meanwhile, as different GAN methods generate various fake textures, the cross-method universality of the method is necessary as the supplementary to cross-attribute universality. Third, the method should be robust to many degradations that may be used by attackers and evaders.

To output a full resolution fakeness prediction map, the method should provide fake images and their ground truth maps for training. The generation method of the fakeness prediction map should fully reflect the location and strength of the fake region. This requires a good design of fakeness maps. In Sec. IV-E, we introduce the implementation of our design and the drawbacks of other methods.

The cross-attribute universality of different facial properties is a difficult problem. Existing GAN-based face generation methods inevitably modify the pixels in other regions that are outside the regions of modified facial properties. Therefore, they use some methods (*eg*., skip connection) to repair the fake images, which makes the region outside modified properties not totally fake. Only the regions of modified properties are full of fake textures. Thus the textures learned by the model in the training procedure lose the universality. To solve this problem, we propose a face-aware attentional encoder-decoder network to improve the cross-attribute universality of the model.

The cross-method universality of various GAN methods is also a difficult problem. To improve the performance of the model across different GAN methods, we propose partial data augmentation and single sample clustering in Sec. IV-D. In particular, we only add data augmentation to the real images of the training dataset while keeping fake images unchanged. The effectiveness of this method is shown in Table VII and Table VIII of Sec. V.

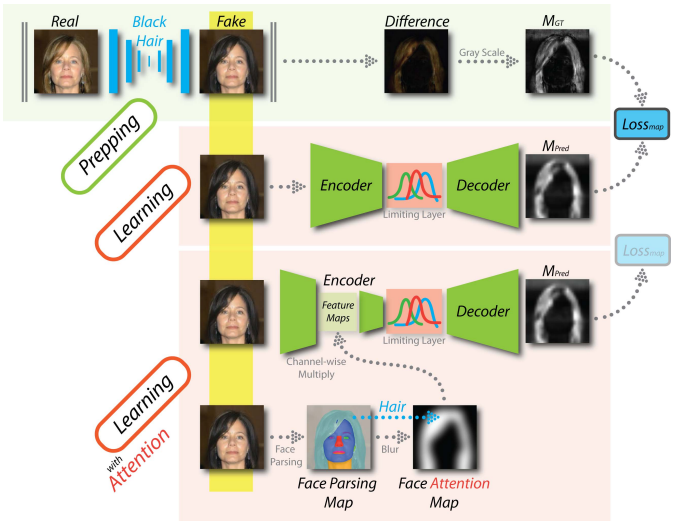


Fig. 5. The framework of our proposed *FakeLocator* method. In the prepping procedure, we use pairs of real images and fake images to produce gray-scale fakeness ground truth maps. In the learning procedure, the network is an encoder-decoder architecture. The inputs are real images or fake images while the outputs are gray-scale fakeness prediction maps. We use the gray-scale fakeness prediction maps and the gray-scale fakeness ground truth maps corresponding to the inputs to calculate the loss. To improve the cross-attribute universality of the model, we introduce the attention mechanism into the model by using the face parsing module. We channel-wise multiply the face attention map with the feature maps of the encoder.

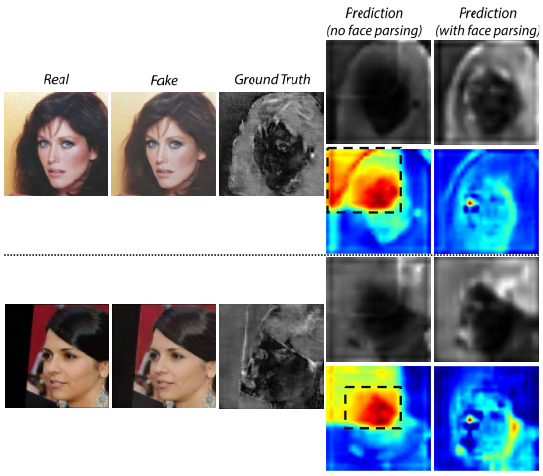


Fig. 6. The images from left to right: real image, fake image, ground truth fakeness map, (top: prediction fakeness map, bottom: attention map) of the model without face parsing module trained by the property *Blond hair*, (top: prediction fakeness map, bottom: attention map) of the model with face parsing module trained by the property *Blond hair*.

In the real world, images may be degraded by various operations such as compression, low-resolution, *etc*. We fully verify the robustness of the model to see whether the proposed method can survive these degradations in Sec. V.

B. General Encoder-Decoder Network for Fake Localization

Fig. 5 shows the framework of our methodology. The backbone network is an encoder-decoder architecture. In our method, any encoder-decoder network can be used as the backbone network. The input is a fake image $\mathbf{X}'$ while the output is a fakeness prediction map $\mathbf{M}_{\text{Pred}}$.

$$\mathbf{M}_{\text{Pred}} = \mathcal{F}_{\text{dec}}(\mathcal{F}_{\text{enc}}(\mathbf{X}')). \tag{2}$$

$\mathcal{F}_{\text{dec}}$ and $\mathcal{F}_{\text{enc}}$ are the decoder and encoder of *FakeLocator* respectively. $\mathbf{M}_{\text{GT}}$ represents the fakeness ground truth map. In our method, the pixel values in the fakeness prediction map are real numbers. So we should fine-tune the network and limit the pixel values in the fakeness prediction map. We add a limiting layer between the encoder and the decoder so that the pixel values could be limited to the range within [0,1]. The subsequent decoder can further process these intermediate features without worrying about correcting different scales of the features. This renders the learning of the decoder much more efficient. This limiting layer can also be added after the decoder. The loss $\mathcal{L}_{\text{map}}$ is simply calculated by the following formula and we will detail the $\mathcal{L}$ in Sec. IV-E.2.

$$\mathcal{L}_{\text{map}} = \mathcal{L}(\mathbf{M}_{\text{Pred}}, \mathbf{M}_{\text{GT}}). \tag{3}$$

We will introduce the network architectures of $\mathcal{F}_{\text{dec}}$ and $\mathcal{F}_{\text{enc}}$ in Sec. V-A. Furthermore, to allow fake classification, we add a binary classifier as a new branch after the encoder and formulate this process with the input image $\mathbf{X}'$ as

$$y_{\text{pred}} = \mathcal{C}(\mathcal{F}_{\text{enc}}(\mathbf{X}')), \tag{4}$$

where $\mathcal{C}$ denotes the classification branch and $y_{\text{pred}} \in \{\text{real}, \text{fake}\}$ is the predicted category of the $\mathbf{X}'$. With this classification module, we can calculate the fake classification accuracy that is an important metric to validate the discriminative power of the encoder and makes our method comparable to state-of-the-art DeepFake detection methods. We will detail the architecture of the classification branch in Sec. V-A.

Although the above framework has the ability to locate fake regions, the cross-attribute universality still needs improvement. As shown in Fig. 6, we use two groups of images to demonstrate the issue in cross-attribute detection. In both groups, the fake images in Fig. 6 are generated by manipulating real images with the property *Black hair*. The model used to locate fake regions is trained by the property *Blond hair*. In the column "prediction (no face parsing)", the top image is the prediction result of the previous model while the bottom image is the attention of the model on the fake image. The prediction result is not good enough. We can find that the model lacks the cross-attribute universality for that it puts too much attention on the face area (*ie.*, regions enclosed by the dotted line).

C. Face-Aware Attentional Encoder-Decoder Network

The previous method obtains a good result in detecting and locating the fake images of the specific facial properties. That is, the model trained with images that have facial property p changed has a good performance on the corresponding fake images while performing poorly on detecting and localizing images with facial property p' ($p' \neq p$) changed.

Since the decoder of each trained GAN-based fake image generation model is fixed, the fake texture generated by the same decoder should be similar. Under this situation, we think

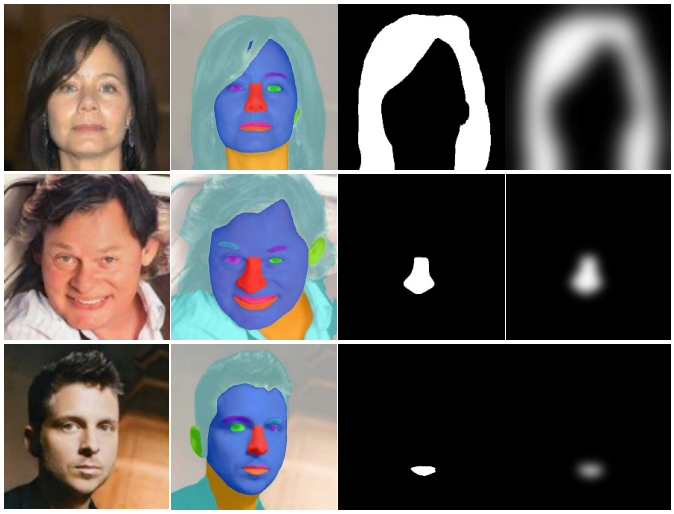


Fig. 7. From left to right: fake image, face parsing map, region map of the modified property, face attention map. The emphasized properties from top to bottom are *Hair*, *Nose*, and *Mouth*.

that the problem is caused by the following reason. Although the GAN-based face generation methods can change facial properties very well, they still inevitably modify the pixels in other regions of the image. Therefore, they use some methods (*eg*., skip connection) to repair the fake images by referring to the real ones, which destroys the fake texture of the region outside modified properties and makes the region not totally fake. Only the regions of modified properties are full of fake textures. Thus, we should provide additional information to make the network pay more attention to the fake texture in the regions of modified properties. Thus we introduce attention mechanism into our method by using a face parsing module to learn the fake textures better. The experiment in Table X demonstrates the effectiveness of the face parsing module.

As shown in Fig. 5, we use face parsing to mark the area corresponding to the modified property and insert the face attention map into the encoder. The attention mechanism urges the model to put more emphasis on the regions of modified property, which improves the cross-attribute universality of the model. In the training step, the face attention map is calculated by the face parsing module (FPM). In the testing step, the face attention map is a white map that does not provide any region information and reserves the original information of feature maps in the encoder. If we set the modified facial property as p, then the face parsing map of fake image $\mathbf{X}'$ is $\mathrm{FPM}(\mathbf{X}', p)$, where function $\mathrm{FPM}(\cdot)$ represent the face parsing operation of the facial image. The face attention map can be written as $\mathrm{Blur}(\mathrm{FPM}(\mathbf{X}', p))$, where $\mathrm{Blur}(\cdot)$ means the blur operation. The formula of the encoder which takes face parsing module into account shall be rewritten as $(\mathcal{F}_{\mathrm{enc}}(\mathbf{X}', \mathrm{Blur}(\mathrm{FPM}(\mathbf{X}', p))))$. Eq. (2) shall be rewritten as

$$\mathbf{M}_{\mathrm{Pred}} = \mathcal{F}_{\mathrm{dec}}(\mathcal{F}_{\mathrm{enc}}(\mathbf{X}', \mathrm{Blur}(\mathrm{FPM}(\mathbf{X}', p))))). \tag{5}$$

As shown in Fig. 7, we demonstrate the face attention map corresponding to the modified properties. The images in turn are the fake images, the face parsing maps, the region maps of the modified properties, and our proposed face attention maps. To produce the face attention map, there are three steps. First, use a face parsing method to generate a face parsing map with the input image. Second, choose the region of the modified property. Third, use the blur method to expand the white region. The reason why we add the third step is that the region outside modified properties also has reference significance. However, their weight should not be higher than the regions of modified properties. The blur method is suitable for this transformation.

As shown in the last column of Fig. 6, the model with face parsing module trained by the property *Blond hair* shows better results on locating fake regions. We can find that the model does not put too much emphasis on the face area. Instead, it distributes enough attention to other areas, which makes it easier to locate the fake textures in the hair area.

D. Partial Data Augmentation

To evaluate the cross-method performance of our method, we compare on three DeepFake types: entire face synthesis, attribute manipulation, identity swap. As shown in Table I, the models in the first column are trained by the corresponding datasets (*ie.*, FaceForensics++, StarGAN, StyleGAN, PGGAN). The datasets in the first row are the test datasets. In each cell, the left value is the accuracy result. We can find that the cross-method accuracy between attribute manipulation and entire face synthesis are high while the cross-method accuracy between them and identity swap is pretty low.

Intuitively, we can take data augmentation into consideration. However, simply using data augmentation is not effective. We perform data augmentation on FaceForensics++ model and test it on StarGAN, StyleGAN, and PGGAN. The accuracy is 0.454, 0.449 and 0.411, respectively, which is only a little higher than without augmentation.

Thus we propose partial data augmentation to further improve the cross-method ability of the model. We think that using data augmentation on both real and fake images may confuse their distribution. Thus we only do data augmentation on real images to tell the model the maximum possible distribution of real images, which instructs it to treat any image that is not real as fake. As shown in Table I, the second value of each cell is the accuracy result of partial data augmentation. We can find that the accuracy is higher than without data augmentation and performing data augmentation on both real and fake images. However, the accuracy result between identity swap and the other two DeepFake types is still not high enough.

We further research the features before classifier (*ie.*, the feature output of the encoder). To be specific, we use t-SNE [45] to reduce the dimension of features and visualize them. As shown in the first row of Fig. 8, we use t-SNE to demonstrate the feature of the FaceForensics++ model on PGGAN, StarGAN, StyleGAN. We can find that the features of real images are entangled with fake images and it is hard to separate them. In the second row of Fig. 8, we use t-SNE to demonstrate the feature of the PGGAN model on StarGAN, StyleGAN, FaceForensics++. We can find that the FaceForensics++ features of real images are

TABLE I

THE ACCURACY BETWEEN THREE DIFFERENT DEEPFAKE TYPES. THE FIRST COLUMN SHOWS THE MODELS WHILE THE SECOND ROW SHOWS THE TEST DATASETS. IN EACH CELL, THE LEFT VALUE IS THE ACCURACY OF THE MODEL WITHOUT DATA AUGMENTATION AND THE SECOND VALUE IS THE ACCURACY OF THE MODEL WITH PARTIAL DATA AUGMENTATION. THE THIRD VALUE (IF EXISTS) IS THE ACCURACY OF THE MODEL WITH PARTIAL DATA AUGMENTATION AND SINGLE SAMPLE CLUSTERING

Model \ Dataset	identity swap	attribute manipulation	entire face synthesis	
	FaceForensics++	StarGAN	StyleGAN	PGGAN
FaceForensics++	-	0.490/0.988 (0.999)	0.363/0.500 (0.854)	0.369/0.500 (0.975)
StarGAN	0.425/0.449 (0.817)	-	1.000/0.999	1.000/0.999
StyleGAN	0.452/0.452 (0.817)	0.989/0.993	-	1.000/0.999
PGGAN	0.448/0.451 (0.676)	0.999/0.990	1.000/0.999	-

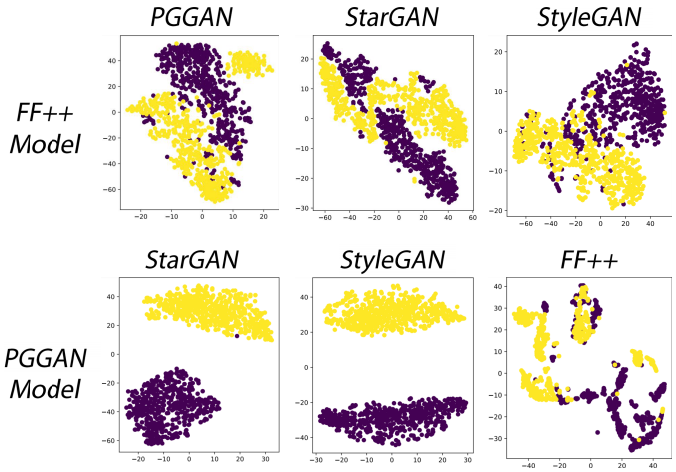


Fig. 8. The t-SNE visualization of features before classifier. The models are on the left and the test datasets are above the right three subfigures.

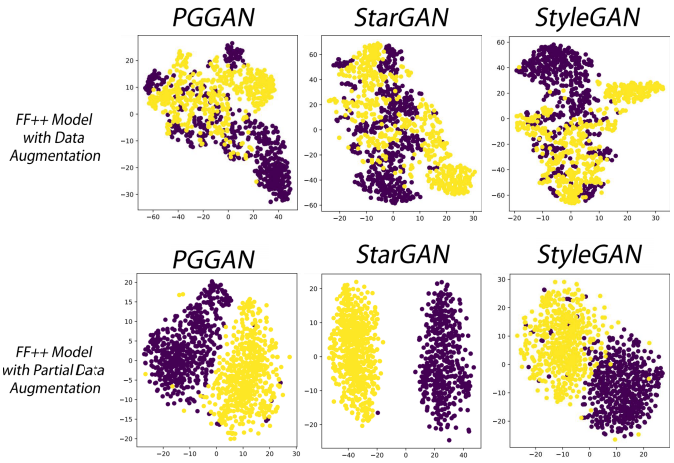


Fig. 9. The t-SNE visualization of features before classifier. The models are on the left and the test datasets are above the right three subfigures.

entangled with fake images while the real and fake images features of StarGAN and StyleGAN are clearly distinct. The visualization is consistent with the experiment result that the cross-method accuracy between attribute manipulation type and entire face synthesis type are high while the cross-method accuracy between them and identity swap is pretty low.

Furthermore, the FaceForensics++ model with data augmentation has a similar conclusion as that without data augmentation. As shown in the first row of Fig. 9, the features of real images are entangled with fake images. Compared with that, in the second row of Fig. 9, the model is the FaceForensics++ model with partial data augmentation. We can find that the real and fake images features of PGGAN, StarGAN, and StyleGAN are clearly distinct, even a simple linear classifier can distinguish them, which violates the low accuracy in Table I. It is obvious that the classifier of the FaceForensics++ partial data augmentation model is not suitable for distinguishing PGGAN, StarGAN, and StyleGAN. That is to say, the universality of the classifier is not enough for cross-method detection.

To deal with this problem, we propose to use clustering with a single sample, which does not include the classifier and only uses the features before the classifier. To be specific, we use k-means [46] to cluster the dimension reduction results from t-SNE into two clusters without labels. Then we randomly choose a single sample that has a ground truth label to confirm which cluster is features of real images and which cluster is features of fake images (*ie.*, we just need to access one single sample with its label from the unknown target GAN). To mitigate the error of random selection, we run the method ten times and calculate the average accuracy.

As shown in the second row of Table I, we can find that the accuracy (*ie.*, the third value in each cell) of the FaceForensics++ partial data augmentation model on StarGAN, StyleGAN, and PGGAN significantly increase, some even doubled. As shown in the second column of Table I, we can find that the accuracy (*ie.*, the third value in each cell) of StarGAN, StyleGAN, and PGGAN partial data augmentation model on the FaceForensics++ dataset also significantly increase.

E. Implementation Details and Analysis of FakeLocator

1) Gray-Scale vs. *Binary Fakeness Map:* In the existing GAN-based face generation methods, even the best of them changes most of the pixels in the image when manipulating a property of the face. If we set the values of manipulated pixels as 1 while unmodified pixels as 0, then the fakeness prediction map does not catch the emphasis of the change in the figure.

To solve the problem, other localization methods [27], [40], [42] use the **binary fakeness map**. They produce difference maps between real images and fake images and use a threshold to generate binary fakeness maps from the difference maps.

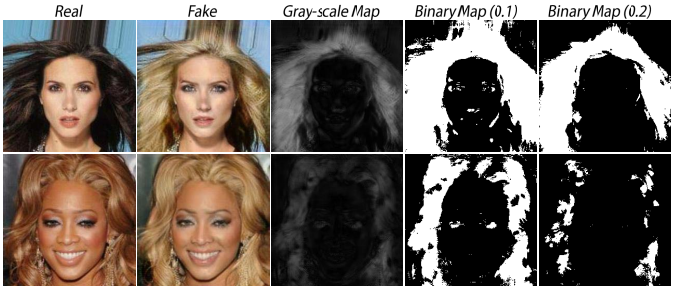


Fig. 10. The images from left to right: real images, fake images generated by modifying with StarGAN of *Blond hair* property, the gray-scale fakeness map, the binary fakeness map generated by threshold 0.1, the binary fakeness map generated by threshold 0.2.

The pixel values larger than the threshold become 1 while the others become 0. However, the setting of a threshold is a big flop. Firstly, all the information about fake regions less than the threshold is omitted. Secondly, the threshold is usually a fixed value defined by experience, which may not work in all cases. Finally, the values of manipulated pixels in the fake image are all less than the threshold if the fake image is slightly manipulated. The binary fakeness map thus turns out to be entirely black and far from the truth.

Here we demonstrate the defect of the binary fakeness map clearly. As shown in Fig. 10, we show the influence of different thresholds on the binary fakeness map. The modified facial property is *Blond hair*. In the first row, according to the same difference map (*ie.*, the gray-scale fakeness map used by us), the binary map with threshold 0.1 not only points out the difference in hair but also points out the difference on eye and mouth, which is imprecise. It can be observed that with a 0.2 threshold we can get a basically satisfied binary fakeness map. However, the gray-scale fakeness map has already reflected the difference between the variation of hair area and non-modified area. There is no need to use a threshold. Moreover, as shown in the second row, the binary map of threshold 0.1 is better than that of threshold 0.2, which is different from the situation of the first row. Just for two images with the same properties modified, we need different thresholds. How to choose a suitable threshold for other different facial properties is a more serious problem. To summarize, using the binary map to represent the fakeness map in the DeepFake localization task is too subjective, which is not as flexible as the gray-scale map.

Therefore, we discard the threshold when producing our fakeness map. The training samples consist of two parts: input image and **gray-scale fakeness ground truth map**. The input images fall into two categories, real images and fake images. For a training sample that regards the real image as the input image, the corresponding fakeness map is a gray-scale ground truth map with all the pixel values equal to 0. This means that the current input image has no fake texture. On the other hand, if the input image is a fake image, the gray-scale fakeness ground truth map is obtained by the following procedure.

As shown in Fig. 11, these images in turn are real image $\mathbf{X}$, fake image $\mathbf{X}'$, gray-scale fakeness ground truth map $\mathbf{M}_{\mathrm{GT\_G}}$ and binary fakeness ground truth map $\mathbf{M}_{\mathrm{GT\_B}}$. $\mathbf{X}$, $\mathbf{X}' \in \mathbb{R}^{H\times W\times 3}$ and $\mathbf{M}_{\mathrm{GT\_G}}$, $\mathbf{M}_{\mathrm{GT\_B}} \in \mathbb{R}^{H\times W\times 1}$, where H, W are height and width of the these images. The fake image is produced by adding property p to the real image.

Fig. 11. From left to right: real image, fake image, gray-scale fakeness ground truth map, and binary fakeness ground truth map. The modified properties from top to bottom are *Bald*, *Blond hair*, and *Smile*.

$$\mathbf{X}' = \mathcal{G}_{\mathrm{dec},S}(\mathcal{G}_{\mathrm{enc},S}(\mathbf{X}), p). \tag{6}$$

$\mathcal{G}_{\mathrm{dec},S}$ and $\mathcal{G}_{\mathrm{enc},S}$ are the decoder and encoder of the method S, $S \in \{\text{all the partial face manipulation GANs}\}$. To achieve the gray-scale fakeness ground truth map, there are three steps as follows. (1) calculate the pixel difference between the real image and the fake image. (2) take the absolute value of the result and turn it into a gray-scale map. (3) divide each pixel by 255. Then each pixel value falls in the range of [0,1]. Eq. (7) shows the formula, in which $\mathbf{X}_{i,j,k}$, $\mathbf{X}'_{i,j,k}$ $(1 \le i \le H, 1 \le j \le W, 1 \le k \le 3, 0 \le \mathbf{X}_{i,j,k}, \mathbf{X}'_{i,j,k} \le 255)$ are the value of a channel of a pixel of $\mathbf{X}$ and $\mathbf{X}'$ respectively. $\mathbf{M}_{\mathrm{GT\_G_{i,j}}}$, $\mathbf{M}_{\mathrm{GT\_B_{i,j}}}$ $(1 \le i \le H, 1 \le j \le W, 0 \le \mathbf{M}_{\mathrm{GT\_G_{i,j}}} \le 1, \mathbf{M}_{\mathrm{GT\_B_{i,j}}} \in \{0, 1\})$ are the value of a pixel of $\mathbf{M}_{\mathrm{GT\_G}}$ and $\mathbf{M}_{\mathrm{GT\_B}}$, respectively. Gray$(\cdot)$ is a function that converts an RGB pixel to a gray-scale pixel.

$$\mathbf{M}_{\mathrm{GT\_G_{i,j}}} = \mathrm{Gray}\Big(|\mathbf{X}_{i,j,k} - \mathbf{X}'_{i,j,k}|\Big)/255. \tag{7}$$

For the entire face synthesis, the fakeness ground truth map is a gray-scale map with all the pixel values equal to 1. The gray-scale fakeness ground truth map clearly depicts the difference between real and fake images and highlights the regions with large differences.

2) Loss Function for Localization: For the gray-scale fakeness map, we have tried four loss functions. Two of them are $\mathcal{L}_1$ loss and $\mathcal{L}_2$ loss, which is commonly used in regression problems. The other two are Focal loss [47] and Dice loss [48], which is commonly used in traditional segmentation problems. In the experiment, we find that $\mathcal{L}_1$ loss and $\mathcal{L}_2$ loss are better choices than the others. This means that for gray-scale fakeness maps, regression losses are better than traditional segmentation losses. The comparison is shown in Table IV. The formulas of $\mathcal{L}_1$ and $\mathcal{L}_2$ are shown in Eq. (8) and (9).

$\mathbf{M}_{\text{Pred\_G}_{i,j}}$ and $\mathbf{M}_{\text{GT\_G}_{i,j}}$ $(1 \le i \le H, 1 \le j \le W)$ represent the pixel in the gray-scale fakeness prediction map and of the gray-scale fakeness ground truth map respectively, where H, W are the height and width of the maps.

$$\mathcal{L}_1 = \frac{1}{n} \sum |\mathbf{M}_{\text{Pred\_G}_{i,j}} - \mathbf{M}_{\text{GT\_G}_{i,j}}|. \tag{8}$$

$$\mathcal{L}_2 = \frac{1}{n} \sum |\mathbf{M}_{\text{Pred\_G}_{i,j}} - \mathbf{M}_{\text{GT\_G}_{i,j}}|^2. \tag{9}$$

The formula of Focal loss is shown below in Eq. (10) and (11). Assume $p \in [0,1]$ is the model's estimated probability for the class with label $y = 1$.

$$p_t = \begin{cases} p & y = 1 \\ 1 - p & \text{otherwise} \end{cases} \tag{10}$$

$$\mathcal{L}_{\text{Focal}}(p_t) = -\alpha_t (1 - pt)^{\gamma} \log(p_t). \tag{11}$$

α_t is an equilibrium factor. The focusing parameter γ smoothly adjusts the rate at which easy examples are down-weighted.

The formulation of Dice loss is shown below in Eq.(12). Function PS$(\cdot)$ represents the point set of any fakeness map. PS$(\mathbf{M}_{\text{GT}})$ and PS$(\mathbf{M}_{\text{Pred}})$ represent the point set of fakeness ground truth map and fakeness prediction map respectively.

$$\mathcal{L}_{\text{Dice}} = \frac{2|\text{PS}(\mathbf{M}_{\text{GT}}) \cap \text{PS}(\mathbf{M}_{\text{Pred}})|}{|\text{PS}(\mathbf{M}_{\text{GT}})| + |\text{PS}(\mathbf{M}_{\text{Pred}})|}. \tag{12}$$

For training the classification network (*ie.*, $\mathcal{C}(\cdot)$) in Eq. (4), we add the cross-entropy loss and jointly train it with the encoder-decoder architecture. We also discuss the influence of this additional module in Table IV.

V. Experimental Result

A. Experimental Setup

1) Databases: Our experiment benchmarks the real face databases CelebFaces Attributes (CelebA) [1] and Flickr-Faces-HQ (FFHQ) [39]. CelebA contains 202,599 face images of celebrities, each annotated with 40 binary attributes. FFHQ contains 70,000 high-quality images. The database includes vastly more variation than CelebA in terms of age, ethnicity, and image background.

The real images are from CelebA and FFHQ databases. The fake images are produced by seven GAN-based face generation methods and their corresponding databases and properties, as shown in Table II. The images are all in PNG format. We use *Entire* to represent the property of the images produced by entire face synthesis methods. We also take FaceForensics++, UADFV, Celeb-DF, and DFFD as the DeepFake dataset.

2) Settings: All the experiments were run on an Ubuntu 16.04 system with an Intel(R) Xeon(R) CPU E5-2699 with 196 GB of RAM, with an NVIDIA Tesla V100 GPU of 32G RAM. In the experiment, we train for 10 epochs with an Adam [49] optimizer whose $\alpha = 0.0001$, $\beta_1 = 0.99$, $\beta_2 = 0.999$, and weight decay $= 10^{-7}$. The minibatch size of every training and validation dataset is 8.

TABLE II
PROPERTIES OF GANS. WE CHOOSE THE FACIAL PROPERTIES WHICH WELL MODIFIED BY THESE GANS

STGAN (CelebA)	AttGAN (CelebA)
Bald, Bangs, Black hair, Blond hair, Smile, Brown hair, Eyeglasses, Male, Mustache	Bald, Bangs, Black hair, Blond hair, Brown hair, Eyeglasses, Male, Smile
StarGAN (CelebA)	IcGAN (CelebA)
Black hair, Blond hair, Brown hair, Gender, Age	Bald, Bangs, Eyeglasses, Smile
StyleGAN (FFHQ)	PGGAN (FFHQ),StyleGAN (FFHQ), StyleGAN2 (FFHQ)
Smile, Age, Gender	Entire

3) Network Architecture: Any encoder-decoder network is feasible in our framework. In the available networks, we choose the Deeplabv3 architecture as our backbone network. Deeplabv3 has a good performance in segmentation and PyTorch [50] provides a pre-trained Deeplabv3-ResNet101 model. Deeplabv3-ResNet101 is constructed by a Deeplabv3 model with a ResNet-101 backbone. The pre-trained model has been trained on a subset of COCO train2017, on the 20 categories that are present in the Pascal VOC dataset. We fine-tune the network and use the Sigmoid function as the limiting layer. The input size supported by Deeplabv3-ResNet101 is 224×224, thus we resized all the input images to this size. Other encoder-decoder networks are also available. In terms of the classification network in Eq. (4), it contains four convolutional layers (kernel size is 3), a max-pooling layer (the kernel size is 2, the stride is 2, padding is 0), three fully connected layers and a Sigmoid layer for classification. It is just added for facilitating the calculation of the accuracy of detecting whether an image is real or fake. As a result, we can also compare our method with state-of-the-art DeepFake detection methods.

4) Metrics: We evaluate our method from two aspects, *ie.*, fake localization and classification. For the fake classification, we employ the classification accuracy (ACC) on all images, accuracy on the real images (real ACC), and accuracy on the fake images (fake ACC) as the metrics. We also report the equal error rate (EER) and area under the curve (AUC) for a comprehensive comparison. Note that, we report the classification results since the accuracy results are the foundation of localization. Only with high accuracy results, is the localization result meaningful. For the evaluation of fake localization (*ie.*, the quality of fakeness prediction maps), we use cosine similarity (COSS), peak signal-to-noise ratio (PSNR) and structural similarity (SSIM), intersection over union (IoU), Dice coefficient (Dice), pixel-wise binary classification accuracy (PBCA) and inverse intersection non-containment (IINC) as the metrics. COSS is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. PSNR is the most commonly used measurement for the reconstruction quality of lossy compression. SSIM is used for measuring the similarity between two images. IoU is the most popular evaluation metric for segmentation tasks. Dice

TABLE III

COMPARISON WITH DETECTION METHODS. WE COMPARE OUR METHOD WITH CNNDETECTION [13] AND DCTA [26] ON UADFV [28], AND CELEB-DF [29]. HERE SHOWS THE ACCURACY RESULT OF EACH METHOD

	UADFV			Celeb-DF		
	Ours	CNNDetection	DCTA	Ours	CNNDetection	DCTA
ACC	1.0	1.0	0.8560	0.9304	0.9992	0.5800
real ACC	1.0	1.0	-	0.8560	0.9993	-
fake ACC	1.0	1.0	-	0.9977	0.9991	-

is a spatial overlap index and a reproducibility validation metric. PBCA treats each pixel as an independent sample to measure classification accuracy. IINC is a more robust metric for comparing the prediction maps proposed by [27]. To enable the calculation of IoU, Dice, PBCA and IINC, we transform the gray-scale fakeness maps to binary fakeness maps. The pixels less than 0.1 become 0 while the others become 1. The threshold value 0.1 is referred to the choice of [27], which has been verified to be a good choice. ACC, AUC, COSS, PSNR, SSIM, IoU, Dice, PBCA metrics are better if a higher value is provided while a lower value is better for EER and IINC. The value ranges of ACC, EER, AUC, COSS, SSIM, IoU, Dice, PBCA and IINC are all in [0,1]. When compared with baseline methods, we use the metrics. For ablation study of our method, we select COSS, PSNR, SSIM as the representative localization metric for that the IoU, Dice, PBCA, and IINC are likely to change with different thresholds chosen by us.

B. Results and Analysis

Experiments evaluate the effectiveness, robustness, and universality of our method.

1) Comparison With Detection Methods: Before evaluating the localization performance of our method, we add extra experiments to verify the effectiveness of the detection performance of our method. After all, the ability of localization is only meaningful if the detection is good. We take state-of-the-art methods (CNNDetection [13] and DCTA [26]) as the baselines. We use the popular datasets UADFV [28] and Celeb-DF [29] as the DeepFake dataset. For UADFV, all three methods are trained with 28,000 images and tested on 2,000 images. For Celeb-DF, all three methods are trained with 280,000 images and tested on 20,000 images. As shown in Table III, we can find that our method has a comparable result with CNNDetection and does better than DCTA.

2) Comparison With Baseline: There are two methods selected by us as the **Baseline**: [27] and [30]. As the method of [27] has an underlying theoretical flaw (low-resolution fakeness map) in localization, we just simply use STGAN with specific property to show the huge gap between their performance and ours. In Table IV, we show a comparison of our method and the **Baseline** [27]. The table also shows the comparison of results with different loss function strategies. Here we choose *Bald* of STGAN as the property of fake images. In the first column, Baseline represents the result of [27]. We enlarge its binary fakeness prediction maps to the size of 224 $\times$ 224 before calculating the metrics. The label *no cl* means the model that does not have a classifier. It is obvious that, compared to **Baseline**, the gray-scale fakeness prediction maps predicted by our method not only have a higher resolution, but also a better performance (several times higher). Moreover, the performance is distinctly higher on all the other properties and GAN-based face generation methods.

For **Baseline** [30], we use DeepFake dataset DFFD, which is collected by [27] to show the comparison. DFFD contains images from FaceForensics++, faceapp, StarGAN, PGGAN, and StyleGAN. Both our method and [30] are trained with 12,618/ 39,914/ 43,698/ 39,998 images of faceapp/ PGGAN/ StarGAN/ StyleGAN as the training dataset. The test dataset for them are all 9,000 images. For identity swap videos in FaceForensics++, we use 180,000 images as the training dataset and 18,000 images as the test dataset. As shown in Table V, we can find that our method has better performance on almost all the metrics than [30] among all the four datasets.

3) Comparison of Loss Functions: Using the same loss function, Deeplabv3 with/without classifier achieves a similar value on metrics. Thus the classifier used to calculate ACC does not affect the fakeness prediction map. The highest value of PSNR and SSIM is achieved by L_1 loss. We can conclude that the loss functions outstanding in regression problems are better for the gray-scale fakeness map. In the following experiments, we choose L_1 loss as the default loss function. Although Dice loss has better performance on Dice, IoU and PBCA, we don't take it as the default loss function for that the values of these metrics are strongly related to the threshold, which is unstable.

4) Generality of Our Approach With Different GANs and Facial Properties: In the experiment, we find that our method is effective for all the GANs and their specified facial properties. Hence there is a further question. Is the model trained on one GAN is effective for other GANs?

As shown in Table VI, we demonstrate some of the test results. Here are two different test pairs. The model trained from STGAN (Bald) tests properties of AttGAN and the model trained from STGAN (Smile) tests properties of StyleGAN. The green columns are the first column of each test pair. It records the value of the model testing on the test dataset of itself, which is used as the reference substance. For each test pair, the three columns right beside the green column are the performance of the model on other datasets.

We can find that the model trained by one GAN and with specified property is more effective on other GANs with the same dataset and property than the GANs with different datasets or properties. In our observation from extensive experiments, the model trained by one GAN and specified property also has bad performance on the same GAN with other properties.

5) Improve The Universality of The Method: A simple idea is to train a model with many single-face-property fake images. Although there is no doubt that this method will improve the universality, we just want to see how well it is. We select STGAN, StyleGAN, AttGAN, StarGAN, IcGAN, PGGAN, StyleGAN2 and all their properties. Each category $C_{i,j}(i \in \text{GANs},\ j \in \text{Properties of GANs}(i))$ supports 2,000 fake images into the training dataset. There are a total of

TABLE IV

LOSS FUNCTION COMPARISON. HERE WE COMPARE THE PERFORMANCE OF DIFFERENT LOSS STRATEGIES. WE CHOOSE *Bald* OF STGAN AS THE PROPERTY OF FAKE IMAGES. IN THE FIRST COLUMN, BASELINE REPRESENTS THE RESULT OF [27]. THE LABEL *no cl* MEANS THE MODEL THAT DOES NOT HAVE A CLASSIFIER

	ACC	real ACC	fake ACC	AUC	EER	COSS	PSNR	SSIM	Dice	IoU	PBCA	IINC
Baseline [27]	0.9975	0.995	1.000	0.9975	0.0050	0.6230	6.214	0.2178	0.1415	0.1432	0.3211	0.4301
L_1 Loss (*no cl*)	\	\	\	\	\	0.9271	22.54	0.7533	0.1975	0.5359	0.8985	0.1898
L_1 Loss	0.9945	0.998	0.991	0.9998	0.0077	0.8813	22.89	0.7876	0.1899	0.5517	0.9272	0.1996
L_2 Loss	0.9935	0.999	0.988	0.9997	0.0072	0.8916	22.36	0.7452	0.1891	0.5371	0.9078	0.2010
Dice Loss	0.9950	1.000	0.990	0.9989	0.0080	0.7645	13.04	0.3581	0.2904	0.5573	0.9339	0.1977
Focal Loss	0.9920	0.996	0.988	0.9995	0.0095	0.4048	20.74	0.3287	0.0708	0.1369	0.8872	0.3357

TABLE V

COMPARISON WITH LOCALIZATION METHOD. HERE WE COMPARE THE LOCALIZATION AND DETECTION PERFORMANCE OF OUR METHOD WITH [27] AND [30]. IN THE FIRST COLUMN ARE THE DIFFERENT DEEPFAKE METHODS IN DATASET DFFD AND DATASET FACEFORENSICS++

		ACC	real ACC	fake ACC	AUC	EER	COSS	PSNR	SSIM	Dice	IoU	PBCA	IINC
DFFD (faceapp)	Ours	0.9998	1	0.9997	1	0.2355	0.7250	23.11	0.6550	0.0974	0.2384	0.8092	0.4036
	Baseline [27]	1	1	1	1	0	0.5538	6.874	0.1848	0.0937	0.0448	0.2964	0.4499
	Baseline [30]	0.9983	0.9995	0.9971	0.9999	0.0020	0.5985	0.7197	0.0570	0.0779	0.0996	0.1017	0.4510
DFFD (StarGAN)	Ours	0.9997	1	0.9995	1	0.0086	0.8373	19.39	0.6433	0.1822	0.4953	0.6749	0.2551
	Baseline [27]	1	1	1	1	0	0.5308	16.23	0.5541	0.1137	0.0345	0.5733	0.5141
	Baseline [30]	1	1	1	1	0	0.7462	1.019	0.1453	0.2033	0.3887	0.3887	0.3056
DFFD (PGGAN)	Ours	1	1	1	1	0	1	142.8	0.9999	0.9999	1	1	0
	Baseline [27]	0.9997	0.9995	1	0.9997	0.0004	0.9231	4.656	0.5358	0.6148	0.9809	0.9809	0.0095
	Baseline [30]	1	1	1	1	0	0.9984	28.86	0.9745	0.9957	0.9993	0.9993	0.0003
DFFD (StyleGAN)	Ours	0.9998	1	0.9997	0.9999	0.0024	0.9998	127.3	0.9997	0.9997	0.9998	0.9998	0.0001
	Baseline [27]	1	1	1	1	0	0.7927	3.286	0.3186	0.5472	0.8052	0.8052	0.0973
	Baseline [30]	0.9994	1	0.9988	0.9999	0.0008	0.9868	19.01	0.8779	0.9710	0.9921	0.9921	0.0039
FaceForensics++	Ours	0.9398	0.9275	0.9521	0.9846	0.0576	0.7529	26.94	0.7691	0.0811	0.3097	0.8687	0.3143
	Baseline [27]	0.8714	0.8042	0.9386	0.8714	0.1725	5.539	0.1877	0.5739	0.0863	0.1174	0.3391	0.4194
	Baseline [30]	0.4909	0.4667	0.5150	0.4659	0.5332	0.5208	6.717	0.1017	0.0689	0.1056	0.1230	0.4432

TABLE VI

UNIVERSAL TEST. HERE ARE TWO DIFFERENT TEST PAIRS. THE MODEL TRAINED BY STGAN (BALD) TESTS PROPERTIES OF ATTGAN AND THE MODEL TRAINED BY STGAN (SMILE) TESTS PROPERTIES OF STYLEGAN

	STGAN	AttGAN			STGAN	StyleGAN		
	Bald	Bald	Black hair	Eyeglasses	Smile	Smile	Age	Gender
ACC	0.994	0.973	0.513	0.519	1.000	0.588	0.593	0.588
PSNR	22.83	21.46	22.84	23.52	35.49	20.85	18.59	16.50
SSIM	0.7823	0.6688	0.3966	0.4168	0.9218	0.5571	0.4958	0.4301
COSS	0.8887	0.8141	0.6932	0.6950	0.8844	0.6176	0.3141	0.6470

TABLE VII

ABLATION STUDY OF THE FUNCTIONALITY OF PARTIAL AUGMENTATION ON THE UNIVERSAL TEST. THE TABLE MAINLY SHOWS THE DETECTION PERFORMANCE OF THE MODEL TRAINED FROM ATTGAN (BALD) TESTING ON STGAN

	AttGAN	STGAN	STGAN	STGAN	STGAN
	no aug	no aug	real(1.0)fake(1.0)	real(1.0)fake(0.0)	real(0.0)fake(1.0)
ACC	0.998	0.7315	0.641	0.9535	0.521
real ACC	0.998	0.9980	1.000	0.9210	1.000
fake ACC	0.998	0.4650	0.282	0.9860	0.042

64,000 fake images and 64,000 real images randomly selected in CelebA and FFHQ. The model performs well on the test dataset, which is shown in Table IX. The gray-scale fakeness prediction maps corresponding to these metric values are referred to in Fig. 1. In addition, even though the model uses a lot of GANs and properties for training, in the testing dataset, it only labels the location of fake regions of the modified facial properties. This is a nice phenomenon demonstrating that the model has learned to recognize the fake textures and distinguish them. This method not only improves the universality but also improves the performance. The metric value of STGAN (Bald) is significantly higher than that in Table IV.

6) Improve The Cross-Method Universality of The Method: Here we also introduce the effect of our method on the same DeepFake category (*ie.*, attribute manipulation) with different GAN types. As shown in Table VII, we train the model on AttGAN (Bald) with/without data augmentation and test the performance of the model on STGAN (Bald). The green column records the value of the model testing on the test dataset of AttGAN (Bald), which is used as the reference substance. In the second row, "no aug" means that the model is trained from a dataset without data augmentation. The format "real(p_{real})fake(p_{fake})" represents the model trained from an augmented dataset. The p_{real} and p_{fake} represent the probability of a real/fake image being augmented in the training procedure. For example, "real(1.0)" means that 100% of the real images are processed by augmentation. The four columns right beside the green column are the performance of the models on detecting STGAN (Bald). *Eg.*, the third

TABLE VIII

THE EFFECT OF PARTIAL DATA AUGMENTATION ON IMPROVING CROSS-METHOD UNIVERSALITY. IN THE FIRST ROW ARE THE FACIAL PROPERTIES USED IN THE DATASET OF THE BELOW COLUMNS. THERE ARE SIX PAIRS OF EXPERIMENTS. FOR EACH OF THEM, THE FIRST COLUMN RECORDS THE VALUE OF THE MODEL TESTING ON THE TEST DATASET OF ITSELF. THE SECOND/THIRD COLUMN RECORDS THE VALUE OF THE MODEL TESTING ON THE OTHER TEST DATASET WITHOUT/WITH PARTIAL DATA AUGMENTATION, WHERE "REAL(p_{real})FAKE(p_{fake})" REPRESENTS THE PROBABILITY OF A REAL/FAKE IMAGE BEING AUGMENTED IN THE TRAINING PROCEDURE

	Bald			Blond hair			Brown hair		
	AttGAN no aug	STGAN no aug	STGAN real(1.0)fake(0.0)	AttGAN no aug	STGAN no aug	STGAN real(1.0)fake(0.0)	AttGAN no aug	STGAN no aug	STGAN real(1.0)fake(0.0)
ACC	0.998	0.7315	0.9535	1.0	0.5	0.5015	1.0	0.5	0.5015
real ACC	0.998	0.9980	0.9210	1.0	1.0	1.0000	1.0	1.0	0.9980
fake ACC	0.998	0.4650	0.9860	1.0	0.0	0.0030	1.0	0.0	0.0050
	STGAN no aug	AttGAN no aug	AttGAN real(1.0)fake(0.0)	STGAN no aug	AttGAN no aug	AttGAN real(1.0)fake(0.0)	STGAN no aug	AttGAN no aug	AttGAN real(1.0)fake(0.0)
ACC	0.993	0.9595	0.9875	0.9975	0.796	0.850	0.9995	0.725	0.8145
real ACC	0.998	0.9980	0.9970	1.0000	1.000	0.775	0.9990	0.999	0.9760
fake ACC	0.988	0.9210	0.9780	0.9950	0.592	0.925	1.0000	0.451	0.6530

TABLE IX

PERFORMANCE OF THE UNIVERSAL MODEL. WE TRAIN A MODEL WITH MANY SINGLE-FACE-PROPERTY FAKE IMAGES. WE SELECT STGAN, STYLEGAN, ATTGAN, STARGAN, ICGAN, PGGAN, STYLEGAN2, AND ALL THEIR PROPERTIES. EACH CATEGORY $C_{i,j}$($i \in$ GANs, $j \in$ Properties of GANs(i)) SUPPORTS 2,000 FAKE IMAGES INTO THE TRAINING DATASET. THERE ARE A TOTAL OF 64,000 FAKE IMAGES AND 64,000 REAL IMAGES RANDOMLY SELECTED IN CELEBA AND FFHQ

	STGAN						StyleGAN		
	Smile	Bangs	Bald	Mustache	Eyeglasses	Black hair	Age	Smile	Entire
ACC	0.9985	0.9985	**0.9985**	0.9935	0.9970	0.9985	0.9985	0.9990	0.9210
PSNR	34.72	33.07	**27.51**	36.59	33.23	31.05	20.71	22.68	55.08
SSIM	0.9086	0.9033	**0.8572**	0.8829	0.8663	0.9029	0.6159	0.6625	0.8268
COSS	0.8634	0.8850	**0.9014**	0.9152	0.9093	0.8817	0.7479	0.7365	0.9174
	StarGAN			AttGAN			IcGAN	PGGAN	StyleGAN2
	Age	Gender	Black hair	Bald	Smile	Eyeglasses	Bald	Entire	Entire
ACC	0.9990	0.9990	0.9990	0.9985	0.9985	0.9985	0.9995	0.9775	0.8550
PSNR	26.38	26.71	25.25	25.31	29.49	28.19	17.57	48.70	72.00
SSIM	0.7929	0.7952	0.7911	0.7839	0.8736	0.8207	0.6002	0.8945	0.9262
COSS	0.8494	0.8416	0.8780	0.8883	0.8277	0.8313	0.8352	0.9972	0.9678

column shows the result of a model trained on AttGAN (Bald) real(1.0)fake(1.0) testing on STGAN (Bald). From the table, we can find that if we augment all the real images and do nothing with fake images, the performance of the model on cross-method universality increases significantly. Both augmenting real and fake images or merely augmenting fake images reduces the performance of our method on cross-method universality. To fully demonstrate the effectiveness, as shown in Table VIII, we test the model with partial data augmentation on both AttGAN and STGAN with three different facial properties. All the accuracies increase when using models trained with partial data augmentation.

7) Improve The Cross-Attribute Universality of The Method on Unseen Facial Properties: If the facial properties are known by us, the method mentioned above is useful. However, sometimes the fake images are modified with unseen facial properties. Here we introduce a method that improves the cross-attribute universality of the model on unseen facial properties.

As introduced in Fig. 5, through adding face attention information, the model can learn fake texture better. In the testing procedure, we use a white map that does not provide any location information as the face attention map. The result shows that the model does learn fake texture better. As an example, we train two models on STGAN (Blond hair). One has face attention information while the other does not. In Fig. 12, we demonstrate the localization performance of the model testing on STGAN (Black hair). **FP** means face parsing. The model with face parsing achieves a better result. It can capture the main fake textures while the model without face parsing is in a mess.

Because this is a cross-attribute test, so we use in-region similarity to compare the results with ground truth. In-region means the region of the modified property. In Fig. 12, it represents the red region of the image in the second row.

In Table X, we show the testing performance on STGAN (Black hair), STGAN (Brown hair), and STGAN (Smile) of the model trained by STGAN (Blond hair). The similarity metrics between the model with face parsing or the model without face parsing are similar. However, the detection accuracy of the model with face parsing is much better than that without face parsing. In Fig. 12, we can also find that model with parsing

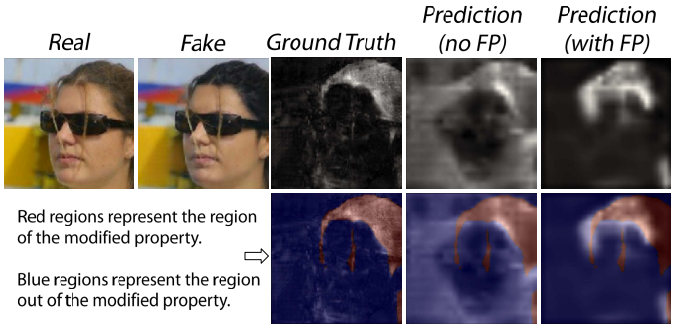


Fig. 12. Result of the models (model with face parsing and model without face parsing) testing on STGAN (Black hair). The models are trained on STGAN (Blond hair). The first row in turn shows the real image, fake image, ground truth, prediction without face parsing, prediction with face parsing. The second row shows the regions where we used to calculate similarity metrics.

TABLE X
COMPARISON OF THE MODEL WITH AND WITHOUT FACE PARSING

	Black hair		Brown hair		Smile	
	no FP	with FP	no FP	with FP	no FP	with FP
ACC	0.237	0.670	0.588	0.859	0.448	0.752
PSNR	28.270	25.296	30.611	21.926	41.836	40.723
SSIM	0.783	0.825	0.854	0.802	0.989	0.990
COSS	0.855	0.817	0.876	0.863	0.803	0.808

can catch fake regions more accurately. To the model without face parsing, if we change facial property *Hair* with different colors, the cross-attribute universality of it will reduce. To the model with face parsing, it learns fake texture better with the introduction of the attention mechanism, which leads to a better cross-attribute universality. It even gets a good detection accuracy on facial property *Smile*, which is very different from facial property *Hair*.

8) Robustness of The Model Against Degradations: It is very necessary to test the robustness of the localization model. In the real world, images may be degraded by various operations such as compression, low-resolution, *etc*. Moreover, we also need to ensure that the model can locate fake textures in any place. This means that the model is location-independent.

To test the robustness of our model, we use two different test scenarios. In the first scenario, we crop each fake image uniformly into four pieces and splice them randomly, which is called *disorganization test*. The purpose of this test is to verify whether the model is strongly correlated to the location, namely, it just remembers the location instead of recognizing the fake texture. Fig. 13 shows the percentage of the metric values compared to that before the disorganization test. The model performance is just a little worse than the formal situation. Fig. 14 shows the prediction result of the *disorganization test*.

In the second scenario, we also apply four different real-world facial image degradations (**JPEG Compression, Blur, Noise, and Low-resolution**) on 1,000 fake images. The gray-scale fakeness ground truth map is processed by the real image and the degraded fake image. In Fig. 15, the vertical

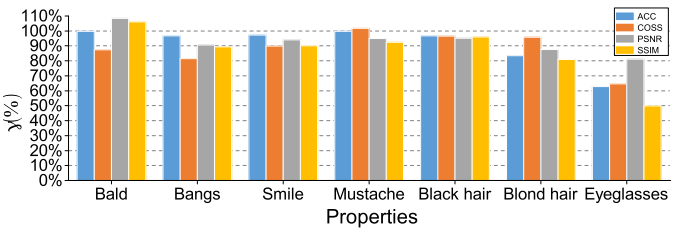
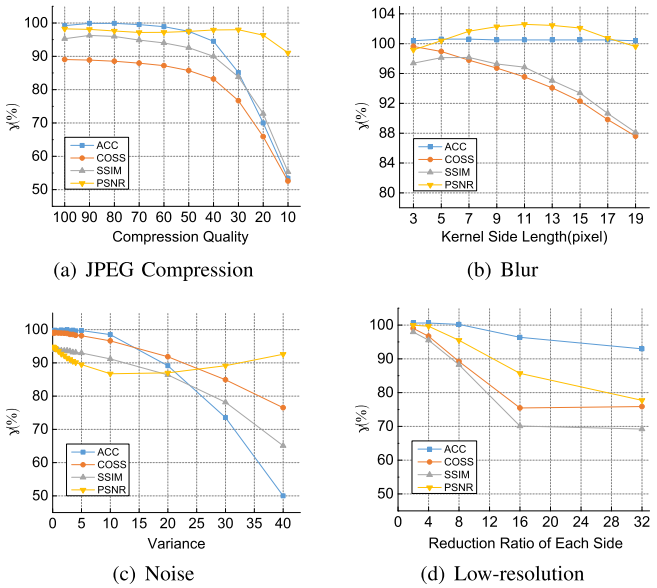


Fig. 13. Comparison with no disorganization. γ means the percentage of the metric values compared with no disorganization.

Fig. 14. **Disorganization test result.** These are the property *Bald* and *Blond hair* of STGAN. We clip the image into four pieces and mess up their order. This operation is for verifying whether our model is location-independent. The result shows that our model is robust to this test.



(a) JPEG Compression (b) Blur

(c) Noise (d) Low-resolution

Fig. 15. The anti-degradation capability of the model. γ means the percentage of the metric values compared with no degradation.

axis of all the four subfigures represents the percentage of the metric values compared to that before degradation. **JPEG Compression** means that converting an image from PNG format to JPEG format. The horizontal axis of the image represents the compression quality during conversion. **Blur** and **Noise** mean that applying Gaussian blur and Gaussian noise to fake images respectively. The horizontal axis respectively represents the filter size of the Gaussian blur and the variance of the Gaussian noise. **Low-resolution** means that resizing the fake image to a low resolution, then restoring it to the original

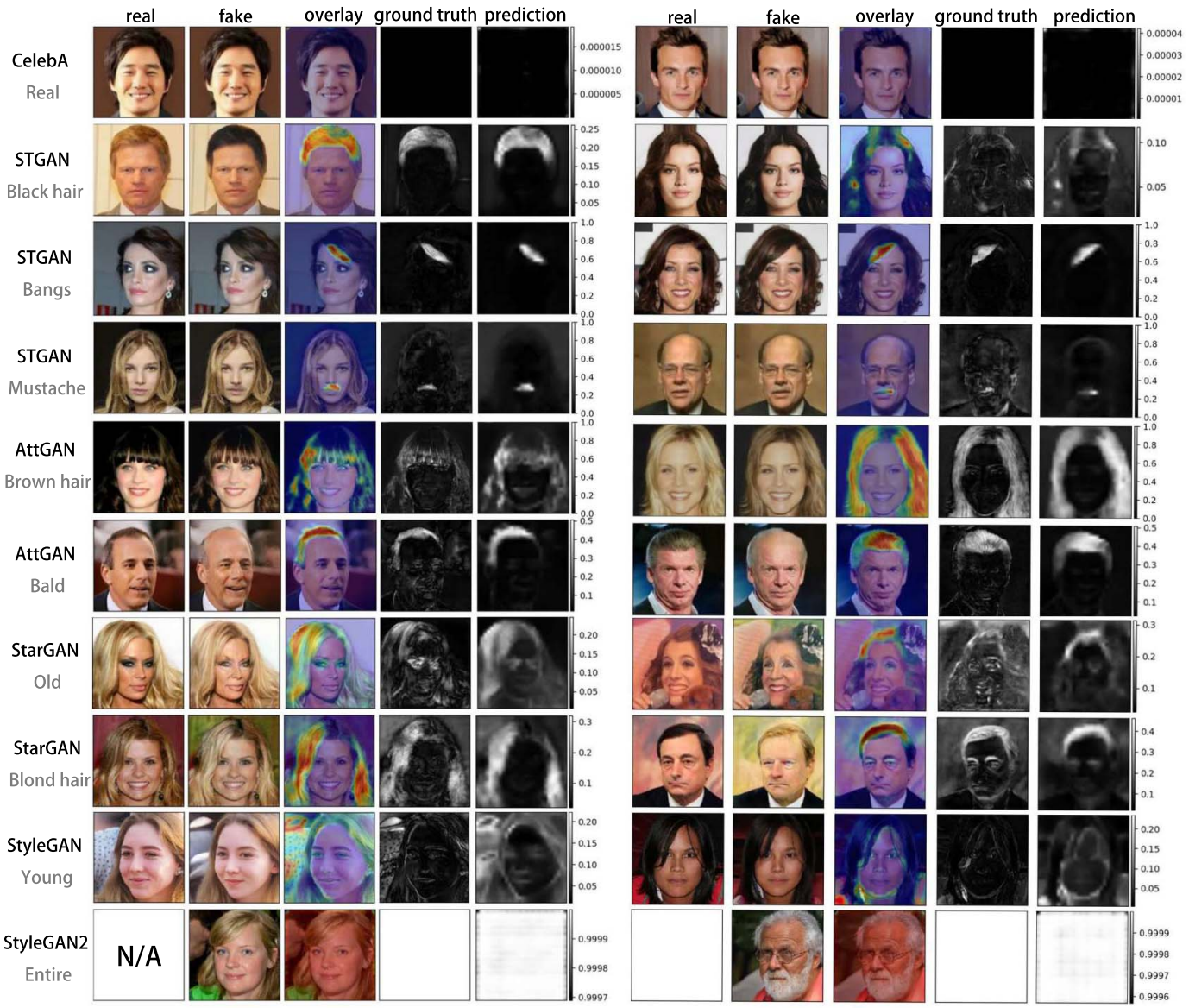


Fig. 16. Fake region localization results. These are the results of different GANs and properties. In the left comments, CelebA [1] is a real image database and others are GAN-based face generation methods. The gray text represents the facial property. **Fake** image is produced by manipulating corresponding **real** image through GAN-based face generation method. **Ground truth** is calculated by **fake** image and **real** image. **Fake** image and **ground truth** are the input and expected output of our method. **Overlay** is combined of **prediction** and **fake** image. The **prediction** has colorbar which shows the value range of pixels. For the first row which uses real image of CelebA as input, for unity of the figure, we also regard it as **fake** image.

resolution with linear interpolation. The quality of the fake image reduces in the resizing procedure. The horizontal axis represents the reduction ratio of each side of the image.

We can find that all the metric values gradually reduce in **Low-resolution**. In the other three degradations, **PSNR** achieves only minor changes, while **ACC**, **COSS**, and **SSIM** decrease when the interference is extremely high. Overall, our methodology performs well against various degradations.

VI. Conclusion

In this paper, we utilize the imperfection of the upsampling procedure in all the GAN-based partial face manipulation methods and entire face synthesis methods. This imperfection can be used for fake detection and fake localization. Thus we propose a universal pipeline to solve the fake localization problem. Through using a gray-scale fakeness map, we achieve the SOTA localization accuracy. As an improvement of the cross-attribute universality of the model, the attention mechanism is inserted into the encoder-decoder architecture by using face parsing information. We also propose partial data augmentation and single sample clustering to improve the cross-method universality of our model. The model is robust to real-world degradations such as blur, compression, *etc*.

Beyond DeepFake detection and localization, we conjecture that the *FakeLocator* is capable of detecting and localizing non-additive noise adversarial attacks such as [51], [52] where the attacked images do not exhibit visual noise patterns and are usually much harder to detect accurately. Recently, some works also have tried to improve the fake images by reconstruction methods [53], [54], which propose a new challenge

to our method. In future work, we think visualizing the fake texture in each image and classify them according to different GANs and upsampling methods is a good idea.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of NVIDIA AI Tech Center (NVAITC) to their research.

REFERENCES

[1] Z. Liu, P. Luo, X. Wang, and X. Tang, "Large-scale celebfaces attributes (celeba) dataset," *Retrieved*, vol. 15, p. 2018, Aug. 2018.

[2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1–11.

[3] deepfakes. (2017). *Faceswap*. [Online]. Available: https://github.com/deepfakes/faceswap

[4] F. Juefei-Xu, R. Wang, Y. Huang, Q. Guo, L. Ma, and Y. Liu, "Countering malicious deepfakes: Survey, battleground, and horizon," 2021, *arXiv:2103.00218*.

[5] D. Thomas. (2020). *Deepfakes: A Threat to Democracy or Just a Bit of Fun?*. [Online]. Available: https://www.bbc.com/news/business-51204954

[6] D. Lee. (2018). *Deepfakes Porn Has Serious Consequences*. [Online]. Available: https://www.bbc.com/news/technology-42912529

[7] R. Wang, F. Juefei-Xu, M. Luo, Y. Liu, and L. Wang, "FakeTagger: Robust safeguards against deepfake dissemination via provenance tracking," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 3546–3555.

[8] N. Yu, L. Davis, and M. Fritz, "Attributing fake images to GANs: Learning and analyzing GAN fingerprints," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7556–7566.

[9] L. Dang, S. Hassan, S. Im, J. Lee, S. Lee, and H. Moon, "Deep learning based computer generated face identification using convolutional neural network," *Appl. Sci.*, vol. 8, no. 12, p. 2610, 2018.

[10] H. Mo, B. Chen, and W. Luo, "Fake faces identification via convolutional neural network," in *Proc. 6th ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2018, pp. 43–47.

[11] R. Wang *et al.*, "Fakespotter: A simple yet robust baseline for spotting Ai-synthesized fake faces," in *Proc. 29th Int. Joint Conf. Artif. Intell. (IJCAI)*, C. Bessiere, Ed., Jul. 2020, pp. 3444–3451.

[12] F. Marra, D. Gragnaniello, L. Verdoliva, and G. Poggi, "Do GANs leave artificial fingerprints?" in *Proc. IEEE Conf. Multimedia Inf. Process. Retr. (MIPR)*, Mar. 2019, pp. 506–511.

[13] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "CNN-generated images are surprisingly easy to spot…for now," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, vol. 7, Jun. 2020, pp. 8695–8704.

[14] F. Marra, C. Saltori, G. Boato, and L. Verdoliva, "Incremental learning for the detection and classification of GAN-generated images," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2019, pp. 1–6.

[15] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Two-stream neural networks for tampered face detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1831–1839.

[16] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A compact facial video forgery detection network," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2018, pp. 1–7.

[17] D. Guera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–6.

[18] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW)*, Jan. 2019, pp. 83–92.

[19] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes," in *Proc. CVPR Workshops*, 2019, pp. 38–45.

[20] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," in *Proc. Interfaces (GUI)*, 2019, vol. 3, no. 1, pp. 80–87.

[21] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2307–2311.

[22] S. J. Sohrawardi *et al.*, "Poster: Towards robust open-world detection of deepfakes," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 2613–2615.

[23] A. Khodabakhsh, R. Ramachandra, K. Raja, P. Wasnik, and C. Busch, "Fake face detection methods: Can they be generalized?" in *Proc. Int. Conf. Biometrics Special Interest Group (BIOSIG)*, Sep. 2018, pp. 1–6.

[24] X. Xuan *et al.*, "On the generalization of GAN image forensics," in *Proc. Chin. Conf. Biometric Recognit.* Cham, Switzerland: Springer, 2019, pp. 134–141.

[25] X. Zhang, S. Karaman, and S.-F. Chang, "Detecting and simulating artifacts in GAN fake images," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2019, pp. 1–6.

[26] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz, "Leveraging frequency analysis for deep fake image recognition," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3247–3258.

[27] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5781–5790.

[28] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing AI generated fake face videos by detecting eye blinking," 2018, *arXiv:1806.02877*.

[29] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for deepfake forensics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3207–3216.

[30] L. Chai, D. Bau, S.-N. Lim, and P. Isola, "What makes fake images detectable? Understanding properties that generalize," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 103–120.

[31] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[32] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8789–8797.

[33] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[34] G. Perarnau *et al.*, "Invertible conditional GANs for image editing," 2016, *arXiv:1611.06355*.

[35] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "AttGAN: Facial attribute editing by only changing what you want," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5464–5478, Nov. 2019.

[36] M. Liu *et al.*, "STGAN: A unified selective transfer network for arbitrary image attribute editing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3673–3682.

[37] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=Hk99zCeAb

[38] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8110–8119.

[39] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.

[40] K. Songsri-in and S. Zafeiriou, "Complement face forensic detection and localization with FacialLandmarks," 2019, *arXiv:1910.05455*.

[41] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," in *Proc. IEEE 10th Int. Conf. Biometrics Theory, Appl. Syst. (BTAS)*, Sep. 2019, pp. 1–8.

[42] L. Li *et al.*, "Face X-ray for more general face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5001–5010.

[43] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016. [Online]. Available: http://distill.pub/2016/deconv-checkerboard

[44] A. C. Gallagher, "Detection of linear and cubic interpolation in JPEG compressed images," in *Proc. 2nd Can. Conf. Comput. Robot Vis. (CRV)*, vol. 5, 2005, pp. 65–72.

[45] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

[46] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means clustering algorithm," *J. Roy. Stat. Soc. C (Appl. Statist.)*, vol. 28, no. 1, pp. 100–108, 1979.

[47] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[48] C. H. Sudre *et al.*, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Cham, Switzerland: Springer, 2017, pp. 240–248.

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

[50] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[51] R. Wang *et al.*, "Amora: Black-box adversarial morphing attack," 2019, *arXiv:1912.03829*.

[52] Q. Guo *et al.*, "Watch out! Motion is blurring the vision of your deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33. Red Hook, NY, USA: Curran, 2020, pp. 975–985.

[53] Y. Huang *et al.*, "Dodging deepfake detection via implicit spatial-domain notch filtering," 2020, *arXiv:2009.09213*.

[54] Y. Huang *et al.*, "FakePolisher: Making deepfakes more detection-evasive by shallow reconstruction," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 1217–1226.

Yihao Huang received the B.S. degree from the Software Engineering Institute, East China Normal University, China, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include computer vision and AI security.

Felix Juefei-Xu (Member, IEEE) received the B.S. degree in electronic engineering from Shanghai Jiao Tong University (SJTU), Shanghai, China, and the M.S. degree in electrical and computer engineering, the M.S. degree in machine learning, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, USA. He is currently a Research Scientist with Alibaba Group, Sunnyvale, CA, USA, with research focus on a fuller understanding of deep learning, where he is actively exploring new methods in deep learning that are statistically efficient and adversarially robust. He also has broader interests in pattern recognition, computer vision, machine learning, optimization, statistics, compressive sensing, and image processing. He was a recipient of multiple best/distinguished paper awards, including IJCB'11, BTAS'15–16, ASE'18, and ACCV'18.

Qing Guo (Member, IEEE) received the B.S. degree in electronic and information engineering from the North China Institute of Aerospace Engineering in 2011, the M.E. degree in computer application technology from the College of Computer and Information Technology, China Three Gorges University, in 2014, and the Ph.D. degree in computer application technology from the School of Computer Science and Technology, Tianjin University, China. He was a Research Fellow with Nanyang Technological University, Singapore, from December 2019 to September 2020. He is currently a Wallenberg-NTU Presidential Postdoctoral Fellow with Nanyang Technological University. His research interests include computer vision, AI security, and image processing.

Yang Liu (Senior Member, IEEE) received the Bachelor of Computing degree (Hons.) and the Ph.D. degree from the National University of Singapore (NUS) in 2005 and 2010, respectively.

He started his post-doctoral work at NUS and MIT. In 2012, he joined Nanyang Technological University (NTU), where he is currently a Full Professor and the Director of the Cybersecurity Laboratory. By now, he has more than 400 publications in top tier conferences and journals. He specializes in software engineering, cybersecurity, and artificial intelligence. His research has bridged the gap between the theory and practical usage of program analysis, data analysis, and AI to evaluate the design and implementation of software for high assurance and security. He has received a number of prestigious awards, including the MSRA Fellowship, the TRF Fellowship, a Nanyang Assistant Professor, the Tan Chin Tuan Fellowship, the Nanyang Research Award 2019, the ACM Distinguished Speaker, the NRF Investigatorship, and 15 best paper awards and one most influence system award in top software engineering conferences like ASE, FSE, and ICSE.

Geguang Pu received the B.S. degree in mathematics from Wuhan University in 2000 and the Ph.D. degree in mathematics from Peking University in 2005. He is currently a Professor with the Software Engineering Institute, East China Normal University. He has published over 100 publications on the topics of software engineering and system verification, including ICSE, FSE, ASE, and CAV. His research interests include program testing and reliable AI systems. He served as a PC member for more than 20 international conference committees.