Pattern Recognition Letters 180 (2024) 75–81

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A siamese-based verification system for open-set architecture attribution of synthetic images

Lydia Abady *, Jun Wang, Benedetta Tondi, Mauro Barni

Department of Information Engineering and Mathematics, University of Siena, Via Roma 56, 53100 Siena, Italy

ARTICLE INFO

Editor: Maria De Marsico

Keywords:
Synthetic image manipulation
Deep learning for forensics
Source attribution
Open-set classification/recognition
Siamese networks

ABSTRACT

Despite the wide variety of methods developed for synthetic image attribution, most of them can only attribute images generated by models or architectures included in the training set and do not work with *unknown* architectures, hindering their applicability in real-world scenarios. In this paper, we propose a verification framework that relies on a Siamese Network to address the problem of open-set attribution of synthetic images to the architecture that generated them. We consider two different settings. In the first setting, the system determines whether two images have been produced by the same generative architecture or not. In the second setting, the system verifies a claim about the architecture used to generate a synthetic image, utilizing one or multiple reference images generated by the claimed architecture. The main strength of the proposed system is its ability to operate in both closed and open-set scenarios so that the input images, either the query and reference images, can belong to the architectures considered during training or not. Experimental evaluations encompassing various generative architectures such as GANs, diffusion models, and transformers, focusing on synthetic face image generation, confirm the excellent performance of our method in both closed and open-set settings, as well as its strong generalization capabilities.

1. Introduction

Synthetic manipulation and generation of images have become ubiquitous and are being increasingly used in a wide variety of applications. Contents generated by Artificial Intelligence (AI) and deepfake technology have garnered widespread attention because they are often used with malicious intent, thus representing a serious threat to public trust. In response to this, several methods have been developed for the detection of synthetic images, performing real vs fake classification. However, in many cases, only knowing that the image is fake is not enough and more information is required on the synthetic nature of the image. In particular, in some cases, it is necessary to know the specific model or the type of architecture used to produce the fake image (synthetic image attribution). Several methods have been proposed for model-level attribution via multi-class classifiers by relying on the artifacts or signatures (fingerprints) left by the models in the images they generate [1–3]. With model-level attribution, models that are fine-tuned or retrained with a different configuration, for instance by using a different initialization or training data, are considered different models, as they are characterized by different fingerprints. This can be a limitation in many real-world applications, where model-level granularity is not needed or is too difficult to achieve. As an answer, recent approaches have started addressing the attribution task under a more general setting, attributing the synthetic images to the architecture that was used to generate them, instead of the specific model [4,5].

A common drawback of most model-level and architecture-level attribution methods [4,5] is that they cannot work in an open-set scenario wherein the test images are generated by a model/architecture that has not been considered during training. This seriously limits the applicability of these methods in real-world applications, where the images seen at operation time may be produced by models/architectures that have not been seen during training, with the consequence that the predictions made by the methods are not trustable. Some approaches address this issue by performing classification with a rejection class, revealing unknown models/architectures, and refraining from identifying the model/architecture, in this case, [6–8].

In this paper, we adopt a different approach, treating the synthetic architecture attribution task as a verification task. In particular, we propose a method to decide whether two synthetic input images have been produced by the same generative architecture or not.[1] We also consider a slightly different setting, where the system is asked to verify a claim about the architecture used to generate a given image, by relying on multiple reference images produced by the same

* Corresponding author.
E-mail addresses: lydia.abady@unisi.it (L. Abady), j.wang@student.unisi.it (J. Wang), benedetta.tondi@unisi.it (B. Tondi), barni@dii.unisi.it (M. Barni).

[1] Focusing on attribution of synthetic images, our system do not consider pristine data as input.

https://doi.org/10.1016/j.patrec.2024.03.002
Received 3 July 2023; Received in revised form 22 December 2023; Accepted 4 March 2024
Available online 5 March 2024

architecture. The system is based on a Siamese Network architecture with an EfficientNet-B4 backbone, trained in two phases: the first one focusing on the feature extraction part to learn the embeddings and the second on the final decision layers. We carried out a thorough experimental campaign considering several generative architectures, including Generative Adversarial Networks (GANs), diffusion models, and transformers, by focusing on synthetic face image generation. The results showed that the proposed system performs very well in both closed and open-set settings, with a significant advantage with respect to systems based on the introduction of a rejection class, which are not able to provide any information about out-of-set architectures, other than recognizing that they do not belong to the set used for training.

The contributions of this paper can be summarized as follows:

- We propose a new verification framework for open-set architecture attribution of synthetic images. Two verification scenarios are considered: in the first one, the system has to decide whether two synthetic input images are generated by the same architecture or not; in the second one, an synthetic image and a claim on the architecture generating it are given, and the system has to decide whether to support the claim or not.
- By focusing on the face image generation domain, we run extensive experiments that prove the good performance of the proposed verification method with several types of generative architectures when different combinations of architectures are considered in both closed and open set scenarios.
- We perform several generalization tests proving that the system can verify the architecture also when unknown models for the various architectures are considered to produce the test images, e.g. models trained with different pristine data, different training procedures, and different configurations of parameters.
- We exploit the verification architecture to build a system for classification with a rejection option, showing that the proposed system outperforms state-of-the-art methods for open-set architecture attribution with a rejection class.

The rest of the paper is organized as follows: in Section 2, we briefly review the state-of-the-art of synthetic image generation and attribution. Then, in Section 3, we describe the proposed framework and architecture. In Section 4, we describe the datasets and the methodology, including the training procedure and the verification protocol. The results of the experiments we carried out to validate the effectiveness of the proposed verification system are reported in Section 5.

2. Related works

To build our verification system for synthetic image attribution, we resort to a Siamese Network-based architecture. It is proper to stress that the use of Siamese Networks is not novel in the forensic literature, and several approaches have been proposed that relies on contrastive learning, and Siamese Networks in particular. For instance, Mayer et al. [9], proposed using a Siamese Network to predict whether pairs of image patches come from the same camera model. Notably, in [10], a Siamese Network was employed to extract a camera model fingerprint, called noiseprint, from image patches, that can be used for image forgery localization. In [11] a method is proposed that utilizes Siamese Networks to reveal inter-eye symmetries and inconsistencies for GAN face detection. Additionally, [12], utilized a Siamese network to reveal whether patches from different images have consistent metadata, facilitating the localization of spliced image content. To the best of our knowledge, this paper is the first attempt to exploit Siamese Networks for synthetic image attribution.

Below, we will briefly present the state of the art of synthetic image generation and image attribution.

2.1. Image generation

After the early attempts that could only generate low-resolution images, nowadays generative models can produce very high-quality, high-resolution images. The first approaches generating low-resolution images were all based on GAN technology, e.g., BEGAN or BigGAN [13,14]. An impressive advance was marked by the emergence of the ProGAN architecture [15], capable of producing 1024×1024 resolution images by using a progressive learning strategy. The quality of the generated high-resolution images was further improved by the StyleGAN series, and in particular, StyleGAN2 [16,17] and StyleGAN3 [18]. More recently, taming transformers [19], a.k.a. VQGANs, combined the power of transformer architectures with the convolutional approach of generative models for image synthesis. They utilize encoder–decoder architectures and transformer-based modules to generate high-quality images with coherent image structures. A drawback of GANs is that they suffer from mode collapse, according to which generators tend to produce a small variety of data that is not as diverse as real-world data. Notable achievements in this direction have been made by diffusion models. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate samples from noisy images (denoising). The pioneering work is the Denoising Diffusion Probabilistic Models (DDPM) [20], which first demonstrated the model's ability to generate high-quality samples with high levels of detail. Later on, methods applying the diffusion process in the latent space have been proposed (see, for instance, [21,22]).

2.2. Synthetic image attribution

The problem of attributing the image to the synthetic model that generated it has been addressed through both active and passive approaches. Active methods involve injecting specific information, e.g., an artificial fingerprint [3], into the generated images during the generation process, which can later be used to identify the source model. On the other hand, passive methods rely on the presence in the generated images of intrinsic artifacts (namely model fingerprints), that are peculiar to the specific model and that can be used to attribute the images to the source model. Marra et al. [1] revealed that each GAN leaves its specific fingerprint in the images it generates. The averaged noise residual image can be taken as a GAN fingerprint. Yue et al. [23] replace the hand-crafted fingerprint formulation in [1] with a learning-based one, decoupling the GAN fingerprint into a model fingerprint and an image fingerprint. Frank et al. [24] and Joslin et al. [25] perform model attribution considering features in the transformed domain. In addition to model-level attribution techniques, researchers have started proposing approaches that address the attribution problem at the architecture-level [4,5,26], whose goal is to attribute the synthetic images to the source architecture, regardless of how the generative models have been trained, fine-tuned or retrained with a different dataset of pristine images or with different configurations.

All the above model-level and architecture-level methods work in a closed-set setting, that is when they are asked to analyze images produced by models included in the training set, in the model-level attribution case, or by (possibly unknown) models from known architectures, in the architecture-level attribution case, and fail in the open-set scenario. The problem of open-set classification has been addressed in several forensic tasks, like camera model identification [27,28]. With regard to the forensic analysis of synthetically generated contents, while the problem of generalization to unknown manipulation methods has been considered recently by several works dealing with synthetic image detection, e.g., [29,30], only a few methods have been proposed dealing with synthetic image attribution in open-set scenario, at model-level [6,7] and at architecture-level [8]. These methods rely on the introduction of a rejection option, whereby images generated by models or architectures that were not seen during training are rejected to avoid making a wrong prediction. Such methods, then, must be

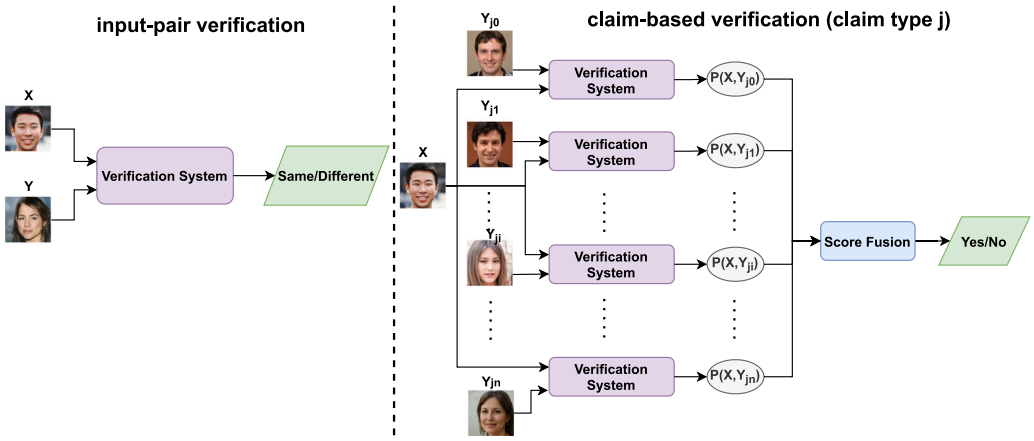

Fig. 1. Verification scenarios considered in the paper.

retrained with samples coming from the new models/architectures if the analysts want to be able to attribute the samples to these architectures/models. The method in [6], for instance, permits the attribution to any new class, assuming that a few labeled images of such a class are available. These images are exploited to derive the distribution of the new class in the latent space.

As we already stated, in this paper, we adopt a different perspective and treat the open-set architecture attribution problem as a verification task, whereby images produced by out-of-set architectures are handled naturally, without introducing a rejection option.

3. Proposed verification system

The proposed verification system for synthetic image attribution is illustrated in Fig. 1. The following verification scenarios are considered:

- Given two input images X and Y, verify whether they are produced by the same generative architecture or not (input pair verification).
- Given an input image X and a claim on the generating architecture, verify whether X has been produced by the claimed architecture or not (claimed-based verification).

In the first scenario, the system is fed with the input pair (X, Y). The label m associated with the input pair is equal to 0 if X and Y have been generated by the same architecture, 1 otherwise. By indicating with $\hat{m}$ the output of the system, and with $p(X, Y)$ the probability score, we have $\hat{m} = 0$ if $p(X, Y) < 0.5$, $\hat{m} = 1$ otherwise. In the second scenario, the verification works by considering one or multiple reference images Y_j generated from the claimed architecture (of Type-j) and evaluating the system with the resulting pairs. In the multiple-reference case, given a dataset of references D_j, all the pairs (X, Y_{ji}), $Y_{ji} \in D_j$, are tested and the final decision (Yes/No) is taken by fusing the outputs according to some fusion strategy. In our experiments, we considered both majority voting and score-level fusion. The latter gave the best performance. In particular, the best results were achieved by considering the minimum probability score. The proposed verification framework naturally works in an open-set scenario, where one of the two inputs or both inputs come from an architecture that has not been used for training (with reference to the second verification scenario, either the input X, or the claim, or both, may come from an unknown architecture).

3.1. Siamese network-based architecture

The system we are proposing to address the tasks described in Fig. 1 relies on a Siamese Network (SN) architecture, see Fig. 2. It consists of two parts, the feature extraction part, and the decision-making part. Feature extraction is performed by an SN that is based on EfficientNet-B4 [31] as a backbone for each of the two branches with shared weights. The input image size for each branch is 380 × 380. The output of each branch is flattened and then fed as input to a dense layer with size input neurons. The feature embedding, then, consists of 512 elements. The features are input to a normalization layer, then the point-wise absolute distance between the two output vectors is computed. The distance vector enters the decision-making network, consisting of three consecutive dense layers of sizes 256, 64, and 1 respectively. The final probability scores are obtained by inputting the output of the last dense layer into a sigmoid activation layer. In our experiments, we also tried other backbone networks to implement the Siamese branches, based on ResNet [32] and SWIN transformers [33]. While we got perfect results with all these networks in the closed-set setting, the EfficientNet backbone is the one giving the best result in the open-set setting.

4. Methodology

4.1. Datasets

To train and test the verification system, we considered a total of 10 architectures, reported below, including GANs, diffusion models, and transformers. Specifically, we considered the following generative architectures: BigGAN [14], BEGAN [13], ProGAN [15], StyleGAN2 [16], StarGANv2 [34], StyleGAN3 [18], DDPM [20], Latent Diffusion [22], LSGM [21] and taming transformers [19].

The generative models utilized for the experiments correspond to models pre-trained on the FFHQ dataset [35] and the CelebA datasets [36], with different configuration parameters, made available in the various repositories. In particular, for StyleGAN3, we considered the two best configurations, namely *t* and *r* [18], trained on different real datasets and with different resolutions. For StyleGAN2, the best performing configuration, namely configuration *f*, is used for training [16].

Starting from this pool of architectures, three different splittings of in-set and out-of-set architectures were considered, with 5 in-set and 5 out-of-set architectures each, named config1, config2, and config3. The details of the splittings are reported in Table 1. We observe that in the first and second configurations, a mixture of GANs, diffusion architectures, and transformers were considered as in-set, while in the third configuration, only GANs are included as in-set architectures. The in-set architectures are used to train the Siamese verification network, while the out-of-set architectures are only considered for testing. For each in-set architecture, we considered 48,000 images, split into training, validation, and testing sets according to the following numbers 45000:2500:500. For each out-of-set architecture, 500 images were considered for testing. Fig. 3 shows an example of generated images for every selected architecture.

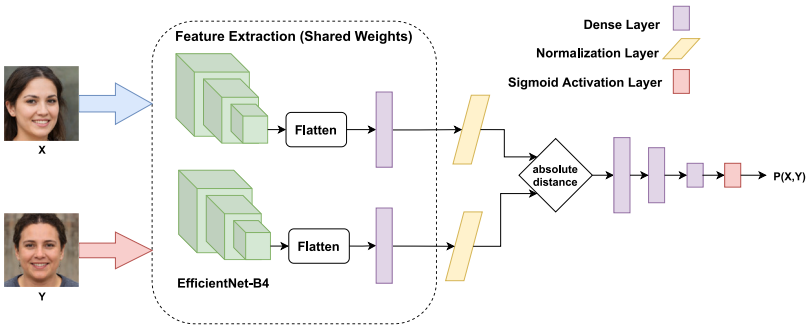

Fig. 2. High-Level Architecture for the verification task.

Fig. 3. Examples of synthetic images from the 10 architectures.

Table 1
Dataset splitting information. Architectures split (in-set and out-of-set) considered in our experiments.

	Domain	Config1	Config2	Config3
In-set	FFHQ	Latent diffusion, Taming transformers, StyleGAN2-f	StyleGAN2-f, Latent diffusion	StyleGAN2-f, StyleGAN3
	CelebA	Latent diffusion, DDPM, BEGAN	BigGAN, ProGAN, Latent diffusion, LSGM	ProGAN, BEGAN, BigGAN
Out-of-set	FFHQ	StyleGAN3, StarGAN2	StyleGAN3, Taming transformers	Latent diffusion, Taming transformers
	CelebA	LSGM, ProGAN, BigGAN	Taming transformers, BEGAN, DDPM, StarGAN2	Latent diffusion, Taming transformers, LSGM, DDPM, StarGAN2

Table 2
Closed-set verification results (Acc).

	Config1	Config2	Config3
Accuracy	1	1	1

As we said, to produce the images, we used pre-trained models released by the authors in the online repositories.

4.2. Siamese network training

We trained three different SN-based verification models, one for each configuration of in-set and out-of-set architectures, namely config1, config2, and config3. The SN-based architecture is trained on a dataset of paired inputs, corresponding to images produced by the same or different architectures, hereafter referred to as positive and negative pairs. For every configuration, the dataset is built from the in-set training dataset as follows: each image is coupled with another image from the same architecture to build a positive pair, and another image is selected randomly from a different architecture to build a negative pair. In this way, the SN is trained on a balanced dataset. Specifically, the training dataset is made up of 45000 × 5 (no. of images per arch × no. of in-set arch) negative pairs and the same number of positive pairs, for a total of 450.000 pair.

In all cases, training was carried out in two distinct phases: the feature extraction phase and the decision phase. In the first phase, the two SN branches are trained for 100 epochs, starting from an EfficientNet-B4 model pre-trained on ImageNet, with Adam optimizer and learning rate equal to 0.0001, using the early stopping condition. The network is trained using contrastive loss [37], defined as

$$\mathcal{L} = (1 - m) \cdot d_E^2 + m \cdot [\max(0, h - d_E)]^2, \quad (1)$$

where d_E is the Euclidean distance between the output of the branches of the SN (embeddings) and h is a margin hyperparameter that enforces a minimum distance between the two embeddings. We set h to 1 in the experiments. The contrastive loss enforces the embeddings of the images in the latent space to be far away whenever the images come from different architectures and close to each other when they belong to the same architecture. Augmentation is performed during training. In particular, we considered JPEG compression, random color transformations (brightness, contrast, saturation, and hue), and random flip. The JPEG quality factors are randomly selected within the range [70–100]. For saturation, a random factor between 0.5 and 1 is considered, while hue is adjusted using a random factor between −0.2 and 0.2. Similarly, brightness undergoes modification with a random factor between −0.2 and 0.2, and contrast enhancement is applied with a random factor between 0.2 and 0.5. Each type of augmentation is applied to the image with a probability of 0.3. Therefore, different images undergo a different number and different types of augmentations. Once the embeddings have been obtained, in the second phase, the weights of the feature extraction network are frozen and the three dense layers

Table 3
Verification results (AUC and pd@0.05) in closed and open-set. The cells with green backgrounds indicate in-set architectures, while the white backgrounds indicate out-of-set architectures in the various configurations.

Generating Architecture	Config1		Config2		Config3	
	AUC	pd@0.05	AUC	pd@0.05	AUC	pd@0.05
Latent Diffusion	1	1	1	1	0.91	0.76
DDPM	0.94	0.74	0.85	0.68	0.91	0.8
Taming transformers	1	1	0.88	0.72	0.84	0.66
StyleGAN2	1	1	1	1	1	1
BEGAN	1	0.99	0.95	0.79	1	1
StyleGAN3	0.9	0.81	0.9	0.81	0.97	0.92
LSGM	0.84	0.68	1	0.98	0.7	0.34
StarGAN v2	0.88	0.72	0.84	0.68	0.89	0.8
BIGGAN	0.95	0.79	0.9	0.74	0.92	0.79
PROGAN	0.85	0.68	1	1	1	1

following the normalization and the absolute distance layer, responsible for the decision, are trained. The binary cross-entropy (BCE) loss is used for training these layers (decision-making network). The layers are trained for 20 epochs with Adam optimizer and a learning rate equal to 0.0001, with an early stopping condition. The code is made publicly available for reproducibility at the link https://github.com/lydialy8/openset_attribution_synthetic_images.git.

4.3. Testing procedure

We evaluated the proposed method by considering two testing scenarios for verification: one-vs-one and one-vs-many. In the one-vs-one case, each input image in the test set is paired with images generated by the 10 architectures (5 in-set, 5 out-of-set), chosen at random from the test set, thus getting a total of 5000 × 10 (10% positive pairs and 90% negative pairs). Then the SN-based model is evaluated on those pairs. The one-vs-one tests measure the performance of the system in the input-pair verification scenario depicted in Fig. 1, and in the claim-based verification scenario, when only one (random) reference is used to verify the claim.

The one-vs-many test setting measures the performance of the system in the claim-based verification scenario when multiple references are available. In our experiments, we considered 100 reference images. Given a test input image and a claim on the architecture (10 possible claims are considered to correspond to all the architectures) - say Type j, we paired each input image with $D_j = 100$ reference images from the claimed architecture. The reference images are randomly selected from the test set. The final decision is taken by considering either the mean or the minimum probability score (the latter resulting in the best results). Formally, we consider, respectively, the statistic $(1/|D_j|)\sum_{i\in D_j} p(X, Y_{ji})$, and $\min_{i\in D_j} p(X, Y_{ji})$.

4.4. Comparison with classification approaches

Given that the verification framework proposed in this paper to address the synthetic attribution task is a novel one, no baseline and state-of-the-art methods can be considered for comparison. In order to show the good capabilities of our system in learning good embeddings for the attribution task, we exploited the SN-based verification model inside a classification framework and run a comparison with existing methods for the classification of synthetic attribution in an open-set scenario. In particular, for every configuration of in-set/out-of-set architectures in Table 1, a classifier with rejection is obtained as follows: (i) we chose one representative image for every in-set architecture. Specifically, the cluster centroid of the validation sub-dataset (corresponding to the architecture) is considered; (ii) the input image is paired with the 5 representative images obtained at step 1, and the SN-based architecture is tested with these pairs; (iii) the pair associated with the minimum score is considered. Formally, let Z_j, $j = 1, \ldots, 5$ denote the 5 centroids. Given an image X, the final classification score associated with X is $\min_{j=1,\ldots,5} p(X, Z_j)$ and the decision is made for the closed-set architecture i^* that achieving the minimum. Rejection is performed by exploiting the so-called Maximum Softmax Probability (MSP) [38]. According to this approach, low confidence in the predicted class reflects the uncertainty of the network prediction, providing evidence that the input sample belongs to an out-of-set class. Then, given a threshold t, the output of the classifier j^* is accepted if $p(X, Z_{j^*}) < t$ (lower scores correspond to higher confidences for the 'Same'/'Yes' class in our case), rejected otherwise (unknown input declared).

5. Experimental results

In this section, we report the performance of the proposed system in the closed and open-set cases and the results of the generalization tests, when unknown models are considered for the same in-set architectures. Finally, we report the comparison results, obtained by considering the classification with rejection system described in Section 4.4.

5.1. Verification results

The results in the one-vs-one setting are reported and discussed below. In Table 2, we report the Accuracy (Acc) of the verification task in the closed-set scenario, when X and Y are produced by in-set architectures, for the 3 configurations. These results show that in the closed-set scenario perfect verification (Acc = 1) can always be achieved by our system. The verification performance in the closed and open-set settings are reported in Table 3 for each architecture, that is for Y belonging to each of the 10 architectures. The average Area Under Curve (AUC) of the ROC curve and the probability of correct detection for a false alarm rate equal to 5% (pd@0.05), are reported for each architecture. The average is computed for the negative pairs over both in-set and out-of-set architectures (9 architectures in total). We observe that the results corresponding to in-set architectures refer to a mixture of closed and open-set scenarios, given that Y may either belong to an in-set or out-of-set architecture (with probability 50%). Said differently, at least one input of the pair comes from the in-set architectures in this case. Instead, the results reported corresponding to out-of-set architectures refer to the open set scenario, where at least one input of the pair, or both inputs (with probability 50%) are produced by out-of-set architectures. By looking at this table, we see that when at least one of the two inputs comes from a known architecture, the verification is perfect or almost perfect. In particular, focusing on config1, we see that the AUC is 1 in 4 out of 5 cases (in which the pd@0.05 is also perfect) and 0.94 in the other case. Similar results are observed in the other configurations. The verification performance decreases, still remaining pretty good, in cases where at least one or both inputs come from unknown architectures (results associated with out-of-set architectures). Overall, similar behavior and results are obtained in the three configurations.

In Table 4, we report the average results for all configurations. The total AUC is averaged over all the possible pairs of inputs, hence

Table 4
Total, open-set and closed-set AUCs.

	Config1	Config2	Config3
Total AUC	0.95	0.93	0.93
Open-set AUC	0.92	0.81	0.85
Closed-set AUC	1	1	1

Table 5
Total, open-set, and closed-set AUCs in the one-vs-many setting.

	Config1		Config2		Config3	
	Mean	Min	Mean	Min	Mean	Min
Total AUC	0.95	0.96	0.94	0.94	0.94	0.94
Open-set AUC	0.92	0.94	0.78	0.84	0.87	0.85
Closed-set AUC	1	1	1	1	1	1

considering all the pairs' combinations (in-set vs in-set, in-set vs out-of-set, out-of-set vs in-set, and out-of-set vs out-of-set). The open-set AUC instead is computed by considering only the out-of-set vs out-of-set pairs (fully open set), while the closed-set AUC is computed over the in-set vs in-set pairs. The results show that config1 shows better results in the open-set scenario. We observe that in this configuration, the out-of-set set contains (mostly) GAN architectures and a diffusion-type architecture (LSGM), that are also present in the in-set. This is not the case in the other configurations where, for instance, transformers in config2 and both diffusion models and transformers in config3 are only considered as out-of-set, without any of these types of architectures included in the in-set set.

In Table 5, we report the average results of the tests one-vs-many, for all the configurations. In all the cases, a slight improvement is observed when the minimum score is considered, compared to the case of one reference only, while the mean score case only improves in a few cases. These results show that using multiple random references for the verification improves the results only slightly. A possible reason is that all the feature vectors for a given architecture tend to cluster close to each other, yielding a similar verification result.

5.2. Generalization tests

We also ran some generalization tests in order to prove the capability of the proposed system to perform attribution when *unknown* models are considered for the various in-set architectures during testing. The purpose of these tests is to show that the system works as supposed to and indeed attributes images to the architecture and not to the single models. In all generalization tests, positive pairs were formed by pairing images from the unknown models with random images from the known models from the same in-set architecture, while negative pairs were created by pairing images from unknown models with random images from different in-set architectures.

In particular, we considered new models from the in-set architectures that were trained: (i) on a different dataset; (ii) by using a different training methodology; and (iii) using different training configurations. For case (i), we tested a system trained in config1, considering as unknown model a taming transformer model trained on the CelebA dataset (the models considered during training for the taming transformer architectures are trained on the FFHQ dataset). Therefore, in this case, positive pairs are formed by generating images from taming transformers trained respectively on FFHQ and CelebA. For the negative pairs, images from the taming transformer model trained CelebA (namely, the unknown) are coupled with images generated from known models for different architectures. In case (ii), we tested the systems trained in all configurations considering as unknown model the StyleGAN2-ada [17] (the systems are trained with images from StyleGAN2-f), which employs an adaptive discriminator augmentation mechanism for training stability in limited data regimes. Finally, for case (iii), we evaluated the system trained in config3 using, as unknown model, a StyleGAN model obtained through retraining on unaligned (*-u-*) high-resolution faces (FFHQ-U) with resolution 1024 × 1024.

Table 6
Results with models trained with different datasets, parameters, and training procedure (in the last line, the number in the models' names refers to the image resolution).

Architecture	Type of mismatch	Train	Test	AUC	ACC
Taming Transf Config1	Dataset	FFHQ	CelebA	1	1
StyleGAN2 Config1	Training Methodology	StyleGAN2-f	StyleGAN2-ada	1	1
StyleGAN2 Config2	Training Methodology	StyleGAN2-f	StyleGAN2-ada	1	1
StyleGAN2 Config3	Training Methodology	StyleGAN2-f	StyleGAN2-ada	1	1
StyleGAN3 Config3	Configuration	StyleGAN3-(t-1024/t-u256/r)	StyleGAN3-t-u1024	1	1

Table 7
Comparison of closed-set (Acc) and open-set (AUC) performance of the classifier based on our SN-based model with state-of-the-art classifiers.

		ResVit[8]	PCSSR[39]	RCSSR[39]	Ours
Config1	**Accuracy**	0.99	0.99	0.99	**1**
	AUC	0.79	**0.84**	0.83	0.82
Config2	**Accuracy**	0.99	0.99	0.99	**1**
	AUC	0.76	0.74	0.75	**0.82**
Config3	**Accuracy**	0.99	0.99	0.99	**1**
	AUC	0.68	0.66	0.64	**0.83**

The results reported in Table 6 demonstrate that the system generalizes well in all scenarios, achieving an AUC and Accuracy equal to 1.

5.3. Comparison results

In this section, we report the results of the experiments that we run considering a classifier built by starting from the proposed verifier, as detailed in Section 4.4. This system is compared with state-of-the-art classification methods, namely the PCSSR and RCSSR variants of the method in [39], recently proposed in the general literature of machine learning for open-set classification, and the ResVit method [8] for the classification of synthetic manipulation and attribution in open-set settings. All these methods perform classification with a rejection option, thus rejecting unknown samples. As to the performance metrics, following [8], we use the Accuracy to measure the closed-set performance, and the AUC to measure the rejection performance. The results reported in Table 7 show that the proposed classifier is the one obtaining the best average performance in all the three configurations of in-set and out-of-set architectures, with a perfect Acc and an AUC gain which is about 8% on the average over PCSSR and RCSSR, and 11% over [8]. These results show the superior capability of our method based on similarity learning of getting characteristic embeddings for the various architectures. Once again, we stress that we considered this framework only for comparison purposes. Indeed, the capabilities of the verification system that we proposed in this paper in the open-set scenario are not limited to sample rejection, given that our system can provide the same functionality in both closed and open-set scenarios.

6. Conclusion

We have proposed a novel verification framework to address the problem of synthetic architecture attribution in open set conditions. The experiments we ran demonstrated good performance of our system

in both closed and open-set settings when different mixtures of generative architectures of synthetic face images are considered as in-set and out-of-set. Generalization performance is also good, with unknown models correctly attributed to the source architectures. We also showed that when the SN-based verification model is used to build a classifier with a rejection class, the results, we got are superior to those achieved by state-of-the-art methods. Future work will focus on improving the results obtained for the one-vs-many case, by optimizing the choice of the images used as references. Moreover, we will consider synthetic images with different semantic content (beyond the face domain) and investigate the generalization capabilities of the proposed system, when the architectures are trained on other domains to get models producing images belonging to different categories.

CRediT authorship contribution statement

Lydia Abady: Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft. **Jun Wang:** Data curation, Software, Validation, Writing – original draft. **Benedetta Tondi:** Supervision, Writing – review & editing. **Mauro Barni:** Funding acquisition, Resources, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work has been partially supported by the China Scholarship Council (CSC), file No. 202008370186, the Italian Ministry of University and Research, PREMIER project, under contract PRIN 2017 2017Z595XS-001, FOSTERER project under contract PRIN 2022 202289RHHP, and the Defense Advanced Research Projects Agency (DARPA), USA and the Air Force Research Laboratory (AFRL), USA under agreement number FA8750-20-2-1004. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

[1] F. Marra, D. Gragnaniello, L. Verdoliva, G. Poggi, Do gans leave artificial fingerprints? in: 2019 IEEE Conference on Multimedia Information Processing and Retrieval, MIPR, IEEE, 2019, pp. 506–511.
[2] N. Yu, V. Skripniuk, D. Chen, L.S. Davis, M. Fritz, Responsible disclosure of generative models using scalable fingerprinting, in: International Conference on Learning Representations, 2020.
[3] N. Yu, V. Skripniuk, S. Abdelnabi, M. Fritz, Artificial fingerprinting for generative models: Rooting deepfake attribution in training data, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 14448–14457.
[4] T. Yang, Z. Huang, J. Cao, L. Li, X. Li, Deepfake network architecture attribution, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, (4) 2022, pp. 4662–4670.
[5] T. Bui, N. Yu, J. Collomosse, RepMix: Representation mixing for robust attribution of synthesized images, in: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIV, Springer, 2022, pp. 146–163.
[6] S. Girish, S. Suri, S.S. Rambhatla, A. Shrivastava, Towards discovery and attribution of open-world gan generated images, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 14094–14103.
[7] T. Yang, D. Wang, F. Tang, X. Zhao, J. Cao, S. Tang, Progressive open space expansion for open-set model attribution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 15856–15865.
[8] J. Wang, O. Alamayreh, B. Tondi, M. Barni, Open set classification of GAN-based image manipulations via a vit-based hybrid architecture, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 953–962.
[9] O. Mayer, M.C. Stamm, Learned forensic source similarity for unknown camera models, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2018, pp. 2012–2016.
[10] D. Cozzolino, L. Verdoliva, Noiseprint: A CNN-based camera model fingerprint, IEEE Trans. Inf. Forensics Secur. 15 (2018) 144–159.
[11] J. Wang, B. Tondi, M. Barni, An eyes-based siamese neural network for the detection of gan-generated face images, Front. Signal Process. 2 (2022) 918725.
[12] M. Huh, A. Liu, A. Owens, A.A. Efros, Fighting fake news: Image splice detection via learned self-consistency, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018.
[13] D. Berthelot, T. Schumm, L. Metz, BEGAN: boundary equilibrium generative adversarial networks, 2017, CoRR, arXiv:1703.10717.
[14] A. Brock, J. Donahue, K. Simonyan, Large scale GAN training for high fidelity natural image synthesis, in: ICLR, OpenReview.net, 2019.
[15] T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of GANs for improved quality, stability, and variation, in: ICLR, OpenReview.net, 2018.
[16] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, T. Aila, Analyzing and improving the image quality of stylegan, in: CVPR, Computer Vision Foundation / IEEE, 2020, pp. 8107–8116.
[17] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, T. Aila, Training generative adversarial networks with limited data, 2020, CoRR, arXiv:2006.06676, URL https://arxiv.org/abs/2006.06676.
[18] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, T. Aila, Alias-free generative adversarial networks, in: NeurIPS, 2021, pp. 852–863.
[19] P. Esser, R. Rombach, B. Ommer, Taming transformers for high-resolution image synthesis, in: CVPR, Computer Vision Foundation / IEEE, 2021, pp. 12873–12883.
[20] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: NeurIPS, 2020.
[21] A. Vahdat, K. Kreis, J. Kautz, Score-based generative modeling in latent space, in: NeurIPS, 2021, pp. 11287–11302.
[22] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models, in: CVPR, IEEE, 2022, pp. 10674–10685.
[23] N. Yu, L.S. Davis, M. Fritz, Attributing fake images to gans: Learning and analyzing gan fingerprints, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 7556–7566.
[24] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, T. Holz, Leveraging frequency analysis for deep fake image recognition, in: International Conference on Machine Learning, PMLR, 2020, pp. 3247–3258.
[25] M. Joslin, S. Hao, Attributing and detecting fake images generated by known GANs, in: 2020 IEEE Security and Privacy Workshops, SPW, IEEE, 2020, pp. 8–14.
[26] T. Yang, J. Cao, Q. Sheng, L. Li, J. Ji, X. Li, S. Tang, Learning to disentangle gan fingerprint for fake image attribution, 2021, arXiv preprint arXiv:2106.08749.
[27] F. de O. Costa, E. Silva, M. Eckmann, W.J. Scheirer, A. Rocha, Open set source camera attribution and device linking, Pattern Recognit. Lett. 39 (2014) 92–101, Advances in Pattern Recognition and Computer Vision.
[28] M. Goljan, M. Chen, J. Fridrich, Identifying common source digital camera from image pairs, in: 2007 IEEE International Conference on Image Processing, vol. 6, IEEE, 2007, pp. VI–125.
[29] X. Yang, S. Liu, Y. Dong, H. Su, L. Zhang, J. Zhu, Towards generalizable detection of face forgery via self-guided model-agnostic learning, Pattern Recognit. Lett. 160 (2022) 98–104.
[30] A.V. Nadimpalli, A. Rattani, On improving cross-dataset generalization of deepfake detectors, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 91–99.
[31] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning, vol. 97, PMLR, 2019, pp. 6105–6114.
[32] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: CVPR, IEEE Computer Society, 2016, pp. 770–778.
[33] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: ICCV, IEEE, 2021, pp. 9992–10002.
[34] Y. Choi, Y. Uh, J. Yoo, J. Ha, Stargan v2: Diverse image synthesis for multiple domains, in: CVPR, Computer Vision Foundation / IEEE, 2020, pp. 8185–8194.
[35] T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, Computer Vision Foundation / IEEE, 2019, pp. 4401–4410.
[36] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: Proceedings of International Conference on Computer Vision, ICCV, 2015.
[37] R. Hadsell, S. Chopra, Y. LeCun, Dimensionality reduction by learning an invariant mapping, in: CVPR (2), IEEE Computer Society, 2006, pp. 1735–1742.
[38] D. Hendrycks, K. Gimpel, A baseline for detecting misclassified and out-of-distribution examples in neural networks, in: International Conference on Learning Representations, 2016.
[39] H. Huang, Y. Wang, Q. Hu, M.-M. Cheng, Class-specific semantic reconstruction for open set recognition, IEEE Trans. Pattern Anal. Mach. Intell. (2022).