Editprint: General Digital Image Forensics via Editing Fingerprint with Self-Augmentation Training

Supplementary Material

Contents

A. Related Works on Supervised Forensics

Digital image forensics constitutes a pivotal means for assessing the authenticity and originality of digital images. Most of the forensics developed in earlier days are based on handcrafted features [7, 10, 17, 24, 35, 36, 48, 54, 57, 65, 74, 77, 81, 86, 94] (and references therein). With the continuous advancement of various deep learning techniques, learning-based forensics has become much more powerful and promising. As the current studies are also learning-based, we, in this section, mainly focus on reviewing learning-based forensics and discussing their differences from our work.

Distinct forensic tasks usually rely on specific features [65]. For tasks such as SID, the required forensic features often depend on high-level semantic information [55, 62, 88, 89, 92, 93], while low-level vision information is widely utilized in traceability recognition tasks like CSI and SNP [39, 49, 52, 56, 90, 95–97]. Specifically, for learning forensic features related to facial semantics,

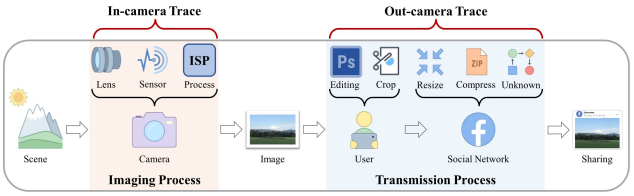

Figure 7. The potential operations that could introduce processing history clues during the imaging and transmission of digital images.

Luo *et al*. [55] devise instance and local similarity-aware objective losses alongside a progressive learning controller. Chandrasegaran *et al*. [14] propose a detector based on color structural saliency differences derived from semantic correlation statistics. On the other hand, for traceability recognition tasks, Mandelli *et al*. [56] introduce a rapid and efficient paired correlation network to extract source characteristics of image capture devices, while Liao *et al*. [52] present a method based on blind signal separation to discern the signal features of different editing operations. The success of the above forensics depends heavily on task-oriented annotated data and forensic strategies. However, the task-oriented factors potentially limit the generalization of the features learned by the forensic methods [21, 65].

B. Editing Pool Specification

Tab. 5 presents the in-camera and out-camera operations adopted in the editing pool. Note that here we purposefully omit the in-camera JPEG compression, as it can be combined with the out-camera compression part to be discussed in $\mathcal{O}_{out}$. Within $\mathcal{O}_{in}$, the demosaicing can be implemented by using one of the existing algorithms AHD, DCB, DHT, PPG from the LibRaw [4]. The white balance can be set to use either the as-shot, automatically calculated, or user-defined balance values [25]. For simplicity, we assume that the user-defined balance values are 1's. In addition, the tone mapping contains options for whether to use automatic increase of brightness and pixel value scaling. An additional null variant is included for white balance and tone mapping, since they can be left unselected. Thus, there are a total of $|\mathcal{C}_{in}| = 4 \times 4 \times 3 = 48$ options for in-camera operations.

As for the out-camera operations, we specify quality factor (QF) values from 50 to 98 with a step of 1 for the JPEG and WEBP compressions. For scaling, we employ four interpolation methods: linear, nearest, area, and cubic. For blurring, we adopt a Gaussian kernel of sizes ranging from 3 to 9 with step 2. Then, for noise addition, we consider the

Table 5. Operations and parameters in the editing pool.

Category	Operation	Parameter
In-Camera	Demosaicing	{AHD, DCB, DHT, PPG}
	White Balance	{Shot, Auto, User}
	Tone Mapping	{Scale, Bright}
Out-Camera	JPEG Compression	QF $\in [50, 98]$ (step = 1)
	WEBP Compression	QF $\in [50, 98]$ (step = 1)
	Gaussian Noise	Variance $\in [3, 9]$ (step = 1)
	Poisson Noise	Intensity $\in [0.1, 0.5]$ (step = 0.05) & Color Shift $\in [0.01, 0.05]$ (step = 0.005)
	Blurring	Kernel Size $\in [3, 9]$ (step = 2)
	Scaling	{Linear, Nearest, Area, Cubic}

commonly used Gaussian noise and Poisson noise [53]. The mean is set to 0 and variance ranges from 3 to 9 with a step of 1 for Gaussian noise, while the intensity levels from 0.1 to 0.5 with a step of 0.05, and color shift levels from 0.01 to 0.05 with a step of 0.005 for Poisson noise. Overall, for out-camera operations, compression, scaling, blurring, and noise addition include 98, 88, 4, and 4 variants, respectively, leading to $|\mathcal{C}_{out}| = \sum_{i=1}^{m} 194^i$

C. Usage of Editprint

After the training phase, the well-trained $E_{\boldsymbol{\theta}}$ can be used in two test scenarios, open-set verification and closed-set classification, following prior works [21, 56, 58, 90].

C.1. Open-set Verification

This scenario aims to determine whether two paired images have undergone the same imaging process or not. Let the test dataset be defined as $\mathcal{D}_{\text{test}} = \{(\mathbf{X}_i, y_i)\}_i$, where $\mathbf{X}_i$ denotes the i-th image and $y_i \in \{1, \ldots, Y\}$ represents its class label. To construct evaluation pairs, we randomly sample two distinct images $\mathbf{X}_{i_1}$ and $\mathbf{X}_{i_2}$ from $\mathcal{D}_{\text{test}}$ (where $i_1 \neq i_2$) to form a pair $P_i = (\mathbf{X}_{i_1}, \mathbf{X}_{i_2})$. The pair is labeled as *positive* ($L_i = 1$) if $y_{i_1} = y_{i_2}$, indicating shared class membership, or *negative* ($L_i = 0$) otherwise. This sampling is repeated until N_{pos} positive pairs and N_{neg} negative pairs are generated. For each pair P_i, we first extract their Editprints by the trained extractor as $\mathbf{E}_{i_1} = E_{\boldsymbol{\theta}}(\mathbf{X}_{i_1})$ and $\mathbf{E}_{i_2} = E_{\boldsymbol{\theta}}(\mathbf{X}_{i_2})$. The likelihood of the pair P_i being a positive pair is quantified via cosine similarity $s_i = \cos(\mathbf{E}_{i_1}, \mathbf{E}_{i_2})$. Ideally, positive pairs should yield higher similarity scores than negative pairs. Finally, the open-set performance is evaluated using the Area Under the ROC Curve (AUC). By comparing the similarity scores $\{s_i\}_{i=1}^{2N}$ against their binary labels $\{L_i\}_{i=1}^{2N}$, the AUC measures separability across all decision thresholds. An AUC of 0.5 corresponds to random guessing, while an AUC of 1.0 indicates perfect discrimination.

C.2. Closed-set Classification

This scenario aims to identify the source imaging process of a test image from a finite reference pool containing Y distinct classes. Given the test dataset $\mathcal{D}_{\text{test}} = \{(\mathbf{X}_i, y_i)\}_i$, we first randomly split the images in a class-wise manner into a reference set and a query set. Assume each class contains N_{ref} images in the reference set and N_{query} images in the query set. Then, we employ $E_{\boldsymbol{\theta}}$ to extract Editprints from the reference images and average the Editprints of the same type to construct a reference pool $\{\bar{\mathbf{E}}_k\}_{k=1}^{Y}$. When a query image $\mathbf{X}_q$ is given, we predict its class $\hat{y}_q$ by identifying the element in the reference pool with the maximum cosine similarity to its Editprint $\mathbf{E}_q = E_{\boldsymbol{\theta}}(\mathbf{X}_q)$, namely,

$$\hat{y}_q = \underset{k \in \{1,\ldots,Y\}}{\arg\max} \cos(\mathbf{E}_q, \bar{\mathbf{E}}_k). \tag{16}$$

The closed-set performance is then evaluated using classification metrics including precision, recall, and F1 score, calculated by comparing predicted classes $\{\hat{y}_q\}$ against ground-truth labels $\{y_q\}$ across all query samples.

Figure 8. Preview of training data (raw images) in $\mathcal{D}_{\text{tr}}$ through visual rendering for better clarity.

D. Implementation Details

Editprint is implemented by PyTorch framework. The visualization of the 10 raw images forming $\mathcal{D}_{\text{tr}}$ is shown in Fig. 8, where the average file size is 9.1 MB per image, with resolutions ranging from 2336×3504 to 4368×2912. These images are excluded from any test sets used in the CSI/SID/SNP tasks. The image encoder $E_{\boldsymbol{\theta}}$ employs EfficientNet [78] initialized with weights pre-trained on ImageNet, while the text encoder $F_{\boldsymbol{\phi}}$ utilizes a Text Transformer [66] architecture loaded with GPT-2 pre-trained weights. We employ the Adam optimizer [37] with default parameters. The initial learning rate is set to 1e-4 and is halved if the validation loss fails to decrease for 1000 iterations until convergence is attained. During the training phase, the mini-batch size $B = 24$ and the self-constructed pairs $M = 4$. All input images are randomly cropped into 512×512 patches to balance computational constraints and model performance, as larger patches reduce batch size and could destabilize training, while smaller ones shrink the extractor's receptive field and could impair the global modeling of operation traces. The dimension D of both the Editprint $\mathbf{E}$ and text features $\mathbf{T}$ is 1024.

Table 6. Comparison of closed-set classification performance for the SNP task using the precision (PRC), recall (RCL), and F1 metrics.

Methods	VISION [72]			FODB [30]			SDR [71]			Mean		
	PRC	RCL	F1	PRC	RCL	F1	PRC	RCL	F1	PRC	RCL	F1
Self-supervised												
ForSim [58]	.7340	.7333	.7319	.6857	.6983	.6866	.1187	.1170	.1177	.5128	.5162	.5121
NoiPri [21]	.6946	.6700	.6431	.7344	.7267	.7029	.0251	.0286	.0186	.4847	.4751	.4549
NoiPri++ [28]	.8736	.8700	.8698	.7804	.7800	.7796	.5921	.5670	.5534	.7487	.7390	.7343
ExifMeta [99]	.9221	.9167	.9170	.8959	.8717	.8723	.8533	.8560	.8539	.8904	.8815	.8811
Editprint	**.9604**	**.9567**	**.9572**	**.9163**	**.9133**	**.9137**	**.8980**	**.9000**	**.8980**	**.9249**	**.9233**	**.9230**
Supervised												
Seq2Seq [96]	**.9669**	**.9667**	**.9667**	.9258	.9250	.9251	.8956	.8960	.8957	.9294	.9292	.9292
MultiClue [82]	.9635	.9633	.9633	**.9277**	**.9267**	**.9268**	**.9019**	**.9030**	**.9023**	**.9310**	**.9310**	**.9308**

E. Task SID

E.1. Testing Details

It is well-known that images synthesized by deep learning (*e.g.*, GANs [27] and DMs [32, 45]) often exhibit distinctive artifacts [84], such as the checkerboard patterns generated by upsampling [61]. On the other hand, natural (real) images captured by cameras lack the aforementioned artifacts. Hence, this subsection evaluates whether Editprint can distinguish between synthesized images and real images. Specifically, we select 5 representative GANs (StarGAN [19], CRN [18], SITD [15], IMLE [46], and SAN [22]) from the ForenSynths dataset [84], and 5 DMs (DALL·E [68], GLIDE [59], Stable Diffusion (SD) [69], Midjourney (MJ) [5], and Adobe Firefly [2]) from the Synthbuster dataset [11]. In addition to general competitors, we include task-specific supervised methods CNNDet [84], HiFi [29], and UniDet [62] that are trained on labeled datasets (*e.g.*, ForenSynths [84], HiFi [29]) for comparative analysis.

F. Task SNP

F.1. Testing Details

Nowadays, a voluminous amount of multimedia content is shared across social networks on a daily basis. Verifying the transmission path of digital images on social platforms constitutes a pivotal aspect of digital forensics [13, 82, 96]. Given that social platforms often employ different processing mechanisms for uploaded images, distinct traces tend to linger. Hence, this subsection endeavors to trace the provenance of images on social networks using Editprint. We choose three representative datasets: VISION [72], FODB [30], and SDR [71], which include 3, 6, and 10 distinct online social networks (OSNs), respectively. To be specific, VISION includes no transmitted images (the absence of transmission can also be regarded as a form of OSN) as well as those transmitted via Facebook and WhatsApp. The FODB further involves Instagram, Twitter, and Telegram. Additionally, the SDR incorporates four commonly used Chinese social networks, namely, QQ, WeChat, Weibo, and DingDing, beyond the aforementioned 6 OSNs. In addition to general competitors, we include task-specific supervised methods Seq2Seq [96] and MultiClue [82] that are trained on labeled dataset SMUD [64] (55k+ images from Facebook, Twitter, and Flickr) for comparative analysis.

F.2. Closed-set Scenario

Fig. 9 depicts the confusion matrices, while detailed precision, recall, and F1 scores are presented in Tab. 6. It is evident that ForSim and NoiPri struggle to be effectively applied in closed-set classification, yielding average F1 scores of 0.5121 and 0.4549 across these datasets, respectively. On the other hand, NoiPri++ and ExifMeta exhibit slightly improved performance, achieving F1 scores of 0.7343 and 0.8811, respectively. In contrast, our proposed Editprint outperforms all other self-supervised methods, attaining the highest F1 score of 0.9230. Similar to the open-set scenario, the supervised Seq2Seq and MultiClue achieve only a marginal advantage of 0.78%, further demonstrating the strong competitiveness of the proposed Editprint. Furthermore, the absence of transmission processing for all uploaded images on the Twitter platform makes some original images indistinguishable from those transmitted through Twitter. This is particularly obvious in the last four confusion matrices of the final row in Fig. 9.

G. Task CSI

G.1. Testing Details

Following the convention [21, 28, 58, 99], we also conduct an evaluation of Editprint in the widely used task of CSI. Specifically, we employ four datasets, namely VISION [72], FODB [30], SDR [71], and Daxing [80], which respectively encompass 29, 25, 21, and 22 distinct camera models. While general competitors (ForSim, NoiPri, NoiPri++, and Ex-

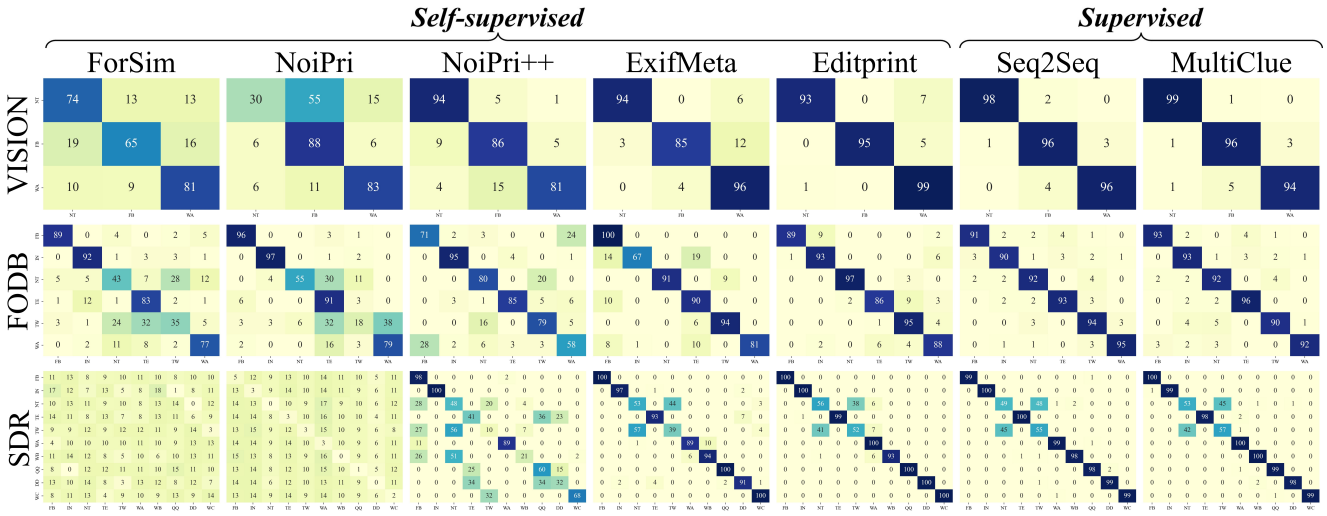

Figure 9. SNP task in closed-set classification via confusion matrices, with actual and predicted labels on the vertical and horizontal axes. NT, FB, WA, IN, TE, TW, WB, DD, WC represent No-Transmission, Facebook, WhatsApp, Instagram, Telegram, Twitter, WeiBo, DingDing, and WeChat, respectively.

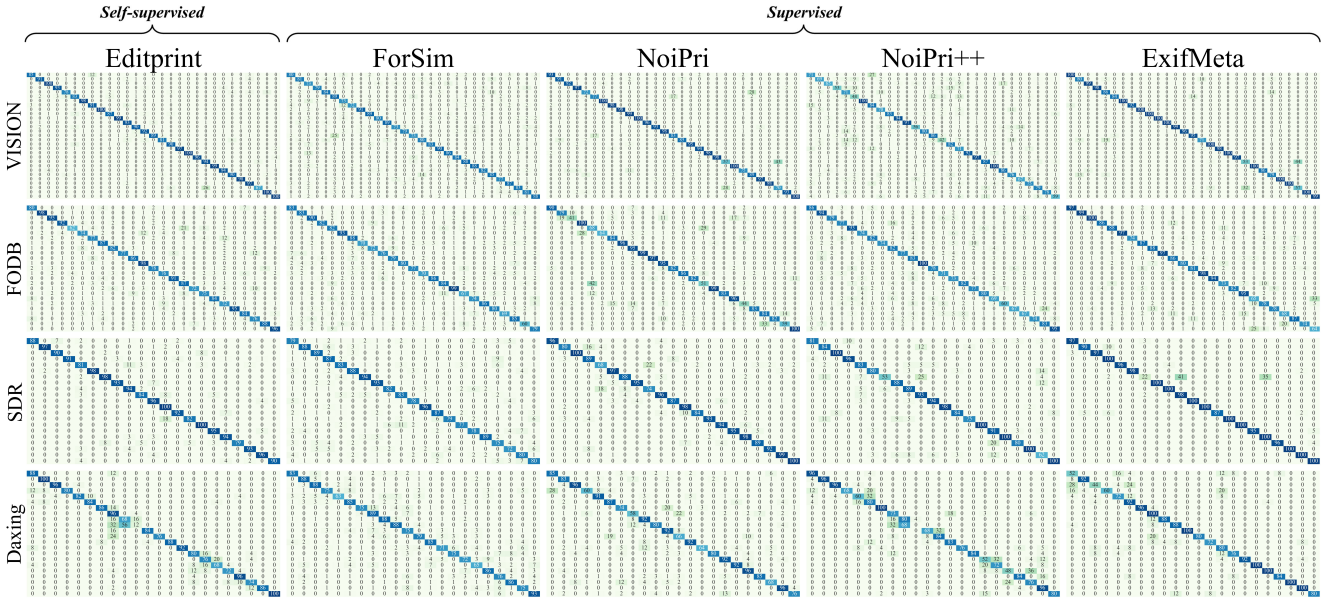

Figure 10. CSI task in closed-set classification via confusion matrices, with actual and predicted labels respectively on the vertical and horizontal axes.

ifMeta) are trained in a self-supervised manner, we categorize them as "***Supervised***" in the CSI task as they explicitly utilize camera model labels or EXIF metadata.

G.2. Closed-set Scenario

Fig. 10 and Tab. 7 present the confusion matrices and corresponding F1 scores for the closed-set classification. Intuitively, all competitors achieve satisfactory camera classification, with their predictions predominantly concentrated along the main diagonal of confusion matrices. However, certain methods exhibit limitations in cross-domain generalization on the challenging Daxing dataset. For instance, NoiPri++ attains an F1 score of only 0.7384. Our Editprint framework addresses this through the proposed self-augmentation training that mimics and refines camera trace discrimination, achieving a 0.8094 F1 on Daxing. Finally, across these four test sets, the self-supervised Editprint achieves an average F1-score of 0.8705, outperforming the

Table 7. Comparison of closed-set classification performance for the CSI task using the F1 metric.

Methods	Datasets				Mean
	VISION	FODB	SDR	Daxing	
Self-supervised					
Editprint	**.9208**	**.8321**	**.9197**	**.8094**	**.8705**
Supervised					
ForSim [58]	.8241	.7936	.8404	.7990	.8143
NoiPri [21]	.9085	.8118	.9098	.8182	.8621
NoiPri++ [28]	.7649	.7914	.8705	.7384	.7913
ExifMeta [99]	**.9196**	**.8468**	**.9326**	**.8412**	**.8851**

supervised algorithms ForSim, NoiPri, and NoiPri++. It only differs by 1.46% from the highest-performing ExifMeta with an F1-score of 0.8851.

H. Ablation Studies

We now analyze the contributions of each component in the Editprint framework, including the extractor backbone, editing pool, self-constructed branch, and adaptive transfer branch. Tab. 8 presents the open-set verification in SID, SNP, and CSI tasks by using AUC as the criterion.

H.1. Impact of Extractor Backbone

Different network architectures inherently exhibit distinct performance. To determine which architecture is best suited for extracting the Editprint, we first compare the performance of using ResNet [31], HRNet [75], ViT [38], and EfficientNet [78] as image encoders in Tab. 8a. For a fair comparison, the most lightweight variants of these architectures are adopted. As observed, ResNet performs much worse, whereas HRNet shows slightly better performance in the SNP and CSI tasks, and ViT performs well only on the SID task. In contrast, EfficientNet achieves superior performance across all tasks, likely due to its architecture designed based on dimension scaling, which enables more effective capture of low-level information. Finally, we select EfficientNet as the default image encoder for Editprint extraction.

Furthermore, we also compare different text encoder architectures, such as Text Transformer (TT) [66], ALBERT [40], and DistilBERT [70]. However, no significant improvements are observed. This may be because feature extraction of text is relatively more straightforward and uniform. Therefore, we simply choose TT as the text encoder.

H.2. Impact of Editing Pool

The editing pool definition is a prerequisite for effective Editprint training. Tab. 8b shows the impact of different editing pools, where $|\mathcal{C}_{in}|$ and $|\mathcal{C}_{out}|$ denote the numbers of in- and out-camera operations, respectively. First, #b1 and #b2 show the case where the editing pool contains only in- or out-camera operations. Although they can achieve certain performances, they are obviously limited by the lack of diversity of sampled editing chains. Specifically, #b1 fails on SNP due to the absence of post-processing operations, while #b2 underperforms on CSI without in-camera simulations. This validates that in-camera operations can simulate various imaging traces, thereby benefiting CSI tasks. Additionally, we explore the impact of reducing other operations, such as JPEG and ISO noising, and observe similar phenomena. This demonstrates that the editing pool needs to encompass a variety of basic operations to be effectively applied to a wider range of forensic tasks. By combining in- and out-camera operations (#b5), Editprint achieves significant performance improvements of 3.97% and 5.53% on SNP and CSI, respectively. In addition, we randomly halve in- and out-camera operations in #b3 compared to #b5, and the observed performance degradation verifies the importance of diversity in the editing pool. While we could potentially further increase the number of operation permutations by increasing m from 3 to 6, as #b5 to #b4, we find that additional improvements are not attainable. This may be because the existing categories already cover the majority of editing traces.

H.3. Impact of Self-constructed Branch

Having established an appropriate editing pool, we now discuss how to construct a training framework using it. First, we attempt to discard the self-constructed branch and rely solely on the adaptive transfer branch, as shown in #c1 of Tab. 8c. The performance degradation of over 30% indicates that the self-constructed branch is indispensable. Next, we compare the image-image contrastive learning (by using InfoNCE loss [63]) with the text-guided contrastive learning (based on our proposed SCPP) employed in the self-constructed branch. Clearly, the image-image paradigm fails to achieve comparable performance against the text-guided ones, *e.g.*, #c2 exhibits an AUC degradation of 6.62% in SID and 9.67% in SNP compared to #c3. This is primarily because the former lacks a global perspective and overlooks the similarities between different editing operations. Although the text-guided contrastive learning based on CLIP in #c3 can achieve certain success, it cannot constrain the similarity within image modality during training. Our proposed SCPP extends the original image-text pairs by explicitly constructing images with identical editing chains, enabling Editprint to capture more general features. However, for these additional constructed images, optimizing strategies such as averaging them before contrast (#c4) or performing numerous one-to-one contrasts (#c5) risk diverse trace erasure or failure to assess contribution variance. To address this, we introduce a fuzzy mapping-based optimization strategy (#c7) that dynamically adjusts contributions based on image-editing chain alignment. This achieves at least 1.90%, 1.88%, and 3.91% AUC improvements over the baseline strategies #c4

Table 8. Ablation studies of the extractor backbone, editing pool, self-constructed branch, and adaptive transfer branch. AUC is the criterion. The last row in each table represents our final selection.

(a) Impact of extractor backbone.

#	Encoder		Investigated Application		
	Image	Text	SID	SNP	CSI
#a1	ResNet	TT	.8679	.7689	.8451
#a2	HRNet	TT	.8903	.8210	.8708
#a3	ViT	TT	.9261	.7357	.8274
#a4	EffNet	ALBERT	.9618	.8850	.9197
#a5	EffNet	TT	.9625	.8853	.9199

(b) Impact of editing pool.

#	$\lvert\mathcal{C}_{in}\rvert$	$\lvert\mathcal{C}_{out}\rvert$	m	Investigated Application		
				SID	SNP	CSI
#b1	48	-	-	.9204	.8456	.8834
#b2	1	$\sum_{i=1}^{m} 194^i$	3	.9315	.8759	.8646
#b3	24	$\sum_{i=1}^{m} 97^i$	3	.9442	.8687	.9076
#b4	48	$\sum_{i=1}^{m} 194^i$	6	.9625	.8850	.9185
#b5	48	$\sum_{i=1}^{m} 194^i$	3	.9625	.8853	.9199

(c) Impact of self-constructed branch.

#	Variant	Text	Investigated Application		
			SID	SNP	CSI
#c1	-	-	.6446	.6829	.5820
#c2	InfoNCE	-	.8204	.7283	.7197
#c3	CLIP	Defined	.8996	.8250	.8521
#c4	SCPP (Mean)	Defined	.9418	.8665	.8930
#c5	SCPP (Each)	Defined	.9435	.8577	.8950
#c6	SCPP (FM)	Numeric	.8853	.7936	.8435
#c7	SCPP (FM)	Defined	.9625	.8853	.9199

(d) Impact of adaptive transfer branch.

#	Label	Adaptation	Investigated Application		
			SID	SNP	CSI
#d1	-	-	.9129	.8455	.8725
#d2	One-hot	-	.9232	.8664	.8843
#d3	Soft	-	.9355	.8691	.9072
#d4	Soft	CDD	.9491	.8710	.8974
#d5	Soft	DINE	.9503	.8745	.9108
#d6	Soft	$\mathcal{L}_{\mathsf{CL}}$ (MMD)	.9531	.8714	.9025
#d7	Soft	$\mathcal{L}_{\mathsf{CL}}$ (Wass.)	.9625	.8853	.9199

and #c5 in SID, SNP, and CSI tasks, respectively.

In text-guided contrastive learning, the definition of text labels can also impact the final performance. #c6 simply assigns numeric indexes to different editing operations, while #c7 employs specific operation names. For instance, given chains with the first operation being DM-AHD and the second one with Blur-3 or WEBP-80, #c6 might use the labels `"123"` and `"456"`, while #c7 would describe them as `"DM-AHD Blur-3"` and `"DM-AHD WEBP-80"`. Clearly, the purely numeric labels in the former cannot capture the implicit relationships present in the latter. Therefore, #c7 significantly outperforms #c6 across the three tasks, illustrating the importance of utilizing language modality.

H.4. Impact of Adaptive Transfer Branch

This subsection analyzes whether the adaptive transfer branch can contribute to the training of Editprint, whose results are presented in Tab. 8d. While using the self-constructed branch alone already yields good results, the introduction of the adaptive transfer branch can further enhance the performance, *e.g.*, #d2 can improve an average of 1.67% compared to #d1. Considering that using one-hot labels may not effectively describe different operations, #d3 improves upon #d2 by using soft labels and successfully gains an average improvement of 1.22%. However, the contribution brought by the adaptive transfer branch may be limited by training instability caused by category-level discrepancy between different batches. We thus consider introducing adaptive constraints supervision $\mathcal{L}_{\mathsf{CL}}$ to stabilize training at #d7. In addition, we also implement existing domain adaptation methods CDD [34] and DINE [51] for comparison in #d4 and #d5 respectively. Taking the CSI task as an example, although CDD grounded in intra-modal contrast constraints can achieve a 0.8974 AUC, it remains constrained by the absence of leveraging target domain label information. Similarly, due to the lack of applying prior knowledge within the text encoder [76], DINE is prone to suboptimal optimization, thereby may not bring substantial enhancements. Our adaptive transfer branch, by utilizing the soft labels distilled by the text encoder and coupling with category-level adaptive constraints, can ultimately yield a 0.9199 AUC for the CSI task. When optimizing $\mathcal{L}_{\mathsf{CL}}$, we adopt a Wasserstein distance-based adaptive constraint in Eq. (10) to regularize cross-batch feature distributions, which unlike nonparametric Maximum Mean Discrepancy (MMD), provides geometrically meaningful gradients through full-distribution shape modeling, enabling stable feature alignment while preserving classifier consistency, as evidenced by a 1.74% AUC drop in MMD-replacement #d6 ablation study.

H.5. Impact of Training Dataset

Before concluding the ablation studies, we discuss the impact of different training datasets. It is well known that using more training data leads to better results for deep learning based methods. To this end, we try to expand the training dataset $\mathcal{D}_{\mathrm{tr}}$ and present the results when the training dataset respectively contains 5, 10, 1000, and 5000 images, in Fig. 11. It can be seen that when the $|\mathcal{D}_{\mathrm{tr}}|$ is expanded to 1000 or even 5000 images, there is no additional performance gain. In addition, the performance stability can be ensured when $|\mathcal{D}_{\mathrm{tr}}| = 10$, whose standard deviation is within 0.001. This is primarily because our objective is to learn how to distinguish editing operations that are unrelated to image

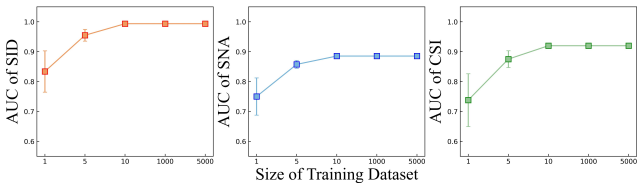

Figure 11. Impact of the size of training dataset $|\mathcal{D}_{\mathrm{tr}}|$ over applications SID, SNP, and CSI by using AUC metric.

content. Furthermore, an interesting question arises:

Can we train Editprint with even less data?

To explore this, we adjust the training dataset with just one image. In this scenario, we observe significant performance degradation and instability. Firstly, having only one image severely restricts the distribution of signal information, leading to severe overfitting of the model. Secondly, if the image contains large flat or saturated regions, it may result in very similar traces for different operations, making them theoretically indistinguishable. Fortunately, increasing the amount of data easily mitigates these issues. In the end, Editprint requires very little data to be trained effectively, and we simply set $|\mathcal{D}_{\mathrm{tr}}| = 10$.

Compared to building the training dataset based on FiveK [12], we evaluate the performance when utilizing different raw data sources, such as from SIDD [6] or IRR [73]. No significant performance difference is observed, indicating that the Editprint training framework relies more on the editing pool than on the training dataset. This also reflects the high practicality of Editprint's training framework.