

E2Style: Improve the Efficiency and Effectiveness of StyleGAN Inversion

Tianyi Wei*, Dongdong Chen*, Wenbo Zhou[†], Jing Liao, Weiming Zhang, Lu Yuan, Gang Hua, *Fellow, IEEE*, Nenghai Yu

Abstract—This paper studies the problem of StyleGAN inversion, which plays an essential role in enabling the pretrained StyleGAN to be used for real image editing tasks. The goal of StyleGAN inversion is to find the exact latent code of the given image in the latent space of StyleGAN. This problem has a high demand for quality and efficiency. Existing optimization-based methods can produce high-quality results, but the optimization often takes a long time. On the contrary, forward-based methods are usually faster but the quality of their results is inferior. In this paper, we present a new feed-forward network “E2Style” for StyleGAN inversion, with significant improvement in terms of efficiency and effectiveness. In our inversion network, we introduce: 1) a shallower backbone with multiple efficient heads across scales; 2) multi-layer identity loss and multi-layer face parsing loss to the loss function; and 3) multi-stage refinement. Combining these designs together forms an effective and efficient method that exploits all benefits of optimization-based and forward-based methods. Quantitative and qualitative results show that our E2Style performs better than existing forward-based methods and comparably to state-of-the-art optimization-based methods while maintaining the high efficiency as well as forward-based methods. Moreover, a number of real image editing applications demonstrate the efficacy of our E2Style. Our code is available at <https://github.com/wty-ustc/e2style>

Index Terms—StyleGAN inversion, Effectiveness, Efficiency

I. INTRODUCTION

GAN inversion aims to invert a real image back into the latent space of a pretrained GAN model, such as StyleGAN [1], [2], for the image to be faithfully reconstructed from the inverted code by the generator. It not only provides an alternative flexible image editing framework but also helps reveal the mechanism underneath deep generative models. As an emerging technique to bridge the pretrained GAN model and real image editing tasks [3], [4], [5], GAN inversion has a high demand for quality and efficiency.

Recently, numerous GAN inversion methods [4], [5], [6], [7], [8] have been proposed and shown strong competence in performing meaningful manipulations of human faces in the latent space. They can be mainly categorized as optimization-based [4], [5], [9] and forward-based [10], [6], [7]. The

Tianyi Wei, Wenbo Zhou, Weiming Zhang, Nenghai Yu are with University of Science and Technology of China, Hefei, Anhui 230026, China. E-mail: {bestwty@mail., welbeckz@, zhangwm@, ynh@}ustc.edu.cn

Dongdong Chen and Lu Yuan are with Microsoft Cloud AI, Redmond, Washington 98052, USA. Email: cddlyf@gmail.com, luyuan@microsoft.com

Jing Liao is with the Department of Computer Science, City University of Hong Kong. Email: jingliao@cityu.edu.hk

Gang Hua is with Wormpex AI Research LLC, WA 98004, US. E-mail: ganghua@gmail.com

* Tianyi Wei and Dongdong Chen are co-first authors, † Wenbo Zhou is the corresponding author.

optimization-based approach optimizes the latent code directly for a given single image by back-propagation. It is capable of producing high-quality inversion but the optimization process is too time-consuming, thus greatly limiting its real-time applications. The forward-based method uses an encoder network to learn the mapping from the image space to the latent space, where only one feed-forward pass is required in the inference, providing higher efficiency for real-time applications. However, it suffers from lower reconstruction quality, and its network structure is usually large and complex. Besides, the hybrid approach [11], [12], [13], [8] that incorporates the optimization on top of the forward network mitigates the problem of quality, but the time cost is greatly increasing.

In this paper, we focus on StyleGAN inversion and propose a new feed-forward network “E2Style”, which significantly improves existing approaches in terms of both Efficiency and Effectiveness.

For the network structure design, we have two insights. First, it is critical to explicitly decouple the information of different layers. In StyleGAN, the latent code of different layers corresponds to different semantic levels, and the later part of the latent code corresponds to the lower level feature information. Existing frameworks such as IDGI [11] usually only use the final feature compressed by the encoder to regress the latent code, which forces various semantic information to couple together in the final feature. Unlike them, our encoder network considers a *hierarchical* structure for the latent code prediction. In this way, feature vectors extracted from various spatial levels of the encoder can correspond to different semantic levels of details from the pretrained StyleGAN generator, achieving semantic correspondence and reducing the learning difficulty. Meanwhile, we find it is not that the deeper the encoder, the better the inversion will be. And a *shallower* encoder is sufficient for the latent code prediction.

Second, the regression should be performed with as little information loss as possible. The pSp [10] downsamples the features from the encoder through several convolutional layers until the resolution is 1×1 , and then regresses the latent code through the full-connected layer, which causes a large degree of information loss. Instead, we adopt a shared *efficient prediction head* at each level, which only consists of a global average pooling layer with varied sizes to levels and a full-connected layer. For features corresponding to the low-level information we use a larger size, which preserves more information and therefore brings performance gains. This efficient prediction head is more lightweight and efficient

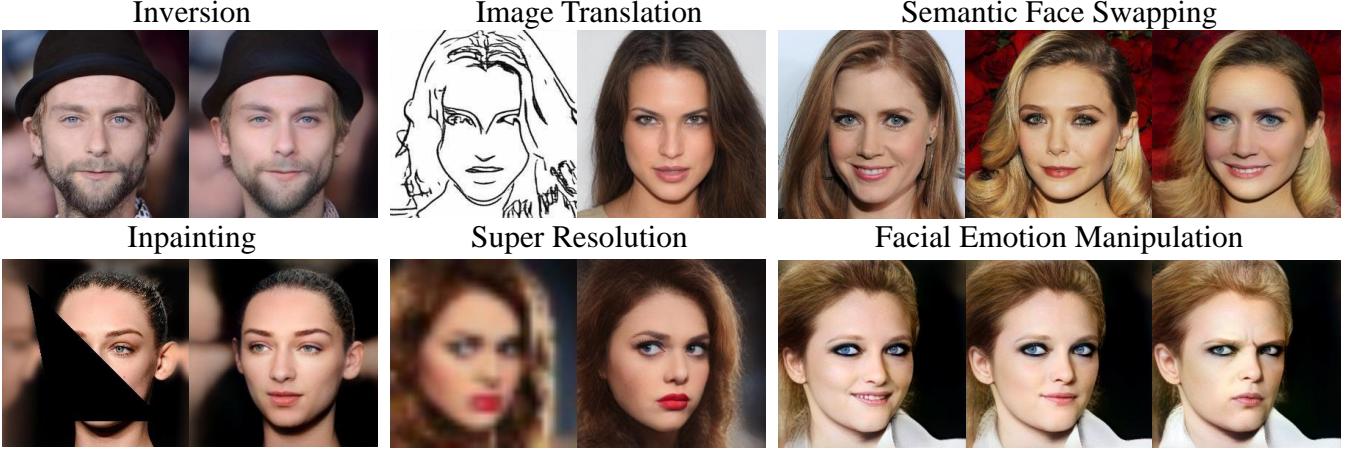


Fig. 1. E2Style can be applied to a large number of real image editing tasks, such as image restoration, image manipulation, image translation, etc.

than the complex and independent heads used in pSp [10] that consists of a series of convolutional layers. With these two network architecture improvements, our network becomes smaller but outperforms IDGI [11] and pSp [10].

Besides the architecture, we introduce two new losses into the objective function for better effectiveness. One is *multi-layer identity loss*, which provides stronger semantic alignment supervision compared to single-layer identity loss used in pSp [10] and thus greatly improves the identity consistency between the reconstructed image and the input real image. The other is *multi-layer face parsing loss*, which helps capture local facial details (e.g., eyes, mouth) for a more fine-grained reconstruction.

In order to further reduce the quality gap between the single stage prediction of latent code and the ideal prediction, we propose a *multi-stage refinement* learning approach to progressively predict the residual of the latent code through multiple passes of the encoders to achieve better inversion quality. Specifically, we introduce a recursive forward refinement phase in our framework, which learns the residual of the latent code output from the first phase to find a more optimal solution that exists around that latent code. E2Style shares the similar spirit to IDGI's prediction-before-optimization method [11], but we improve the initial inverted latent code by introducing a refinement network. This improves the performance without a significant increase in time consumption.

Qualitative and quantitative experiments show that E2Style substantially outperforms existing forward-based methods and even achieves performance comparable to the state-of-the-art optimization-based methods. A number of applications including secure deep hiding, image manipulation, image restoration, and image translation evidence the generalization of E2Style for real-time image editing tasks, illustrated in Figure 1.

To summarize, our contributions are summarized as follows:

- We design an efficient network structure for StyleGAN inversion, which has fewer network parameters and better performance than existing forward-based methods.
- To further improve the inversion effectiveness, we utilize multi-layer identity loss and multi-layer face parsing loss as well as introduce a recursive forward refinement phase.

- We propose a new application of StyleGAN inversion: secure deep hiding. And various real image editing applications demonstrate the power of our approach.

II. RELATED WORK

A. Generative Adversarial Networks (GANs).

Since GANs were first proposed by Goodfellow *et al.* [14] in 2014, it has evolved considerably in terms of training strategies [2], [15], loss functions [16], [17], [18], regularizations [19], [20], [21], and network structures [22], [23]. Today's popular GANs have demonstrated amazing capabilities in many computer vision tasks [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], especially image synthesis [34], [35], [36]. BigGAN [20] can successfully generate high-fidelity, diverse samples for complex and large-scale datasets like ImageNet [37] by only feeding the class condition. ProGAN [38] and StyleGAN [1], [2] can synthesize high-resolution images up to 1024×1024 with a progressive upsample network. To control the GAN synthesis process, some works explore the semantic editing of latent codes by finding semantic directions on their latent space in a supervised [3] or unsupervised [39], [40] manner. Besides, built upon these pretrained GAN models, a lot of real image editing tasks can also be conducted in the latent space, such as super resolution, face attribute manipulation. To enable these tasks, GAN inversion acts as the intermediate bridge. Specifically, the target real image will be firstly inverted into the latent space via such inversion techniques, then be edited in the latent space. In this paper, we follow existing methods [41], [10], [4], [5], [7], [11] and choose StyleGAN as the inversion target, but the proposed feed-forward GAN inversion network and the accompanying design principles can be easily adapted to other GAN models.

B. GAN Inversion.

Existing GAN inversion methods can be subdivided into optimization-based [4], [5], [42], [9], forward-based [10], [6], [43], [7] and hybrid [11], [12], [13], [8], [44] approaches.

The optimization-based approach directly optimizes the latent code to minimize the reconstruction loss, so that the

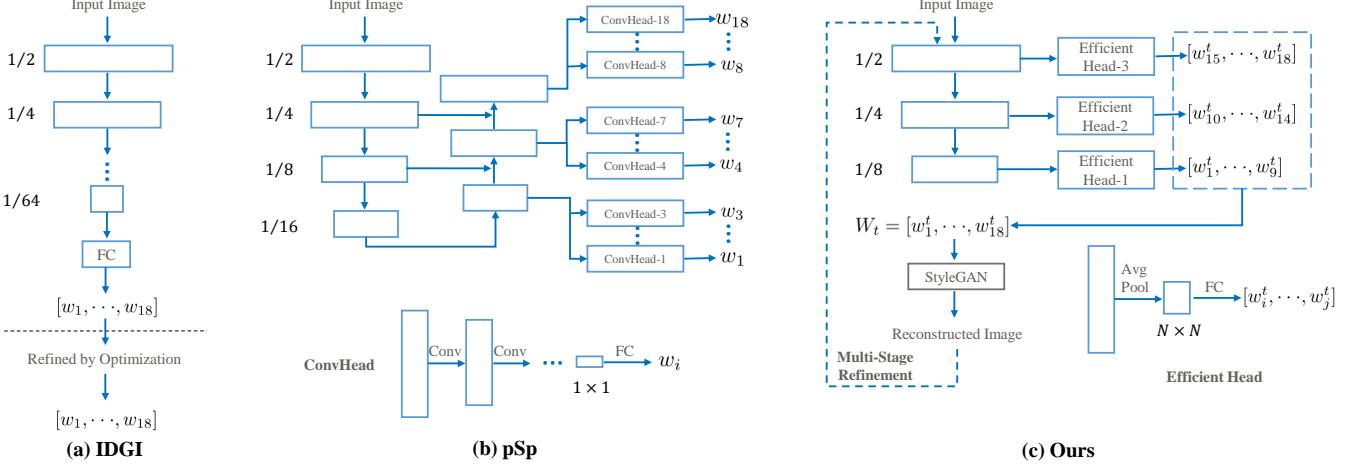


Fig. 2. Network structure comparison with other methods. IDGI [11] uses a deep backbone network and pSp [10] employs 18 high-overhead ConvHeads to predict the single layer latent code separately. Compared with them, we design a shallower backbone and a more efficient prediction head. We also introduce the purely feed-forward-based multi-stage refinement, which improves the performance without bringing a significant increase in time consumption.

synthetic image generated by the inverted latent code is as similar as possible to the target real image. For example, Pan *et al.* [9] optimize the latent code while fine-tuning the parameters of the generator. Image2StyleGAN++ [5] takes alternative optimizations of StyleGAN’s latent code and noise space to further improve the inversion performance. Gu *et al.* [42] argue that the mapping of a single image to the low-dimensional latent code is necessarily lossy and therefore uses multiple latent codes to reconstruct a single image, thus alleviating this situation. Although such optimization-based methods can achieve good reconstruction results, the optimization process is too time-consuming and requires even thousands of iterations and ten minutes for some examples, especially when the initialization point is too far from the ideal embedding, thus greatly limiting its real-time application.

To speed up, the forward-based methods directly use an encoder network to learn the mapping from image space to the latent space. For example, Nitzan *et al.* [6] use two encoders to encode identity information and attribute information separately. Guan *et al.* [7] propose a novel collaborative learning framework to train the encoder in an unsupervised manner. Concurrent with our work, ReStyle [45] extends the encoder-based inversion method by introducing an iterative refinement mechanism. This idea is similar to our multi-stage refinement, but we use a separate encoder at each stage and outperform ReStyle in terms of all evaluation metrics. The pSp [10] designs a feature pyramid network with independent inversion heads for each latent code and introduces the identity loss as an additional constraint. Different from these methods, we introduce a simple feed-forward network with improved efficiency and quality. Moreover, the newly proposed multi-layer identity loss, local parsing loss, and multi-stage refinement should also generalize to these methods.

Furthermore, the hybrid approach combines forward-based and optimization-based methods. Zhu *et al.* [11] use an encoder to generate an initial latent code, and the following optimizer uses it as the starting point to further refine the latent code. Yang *et al.* [44] follow the prediction-then-optimization

approach of [11] except that a detached dual-channel domain encoder is proposed. Similarly, DNI [8] uses pSp [10] as the domain-guided encoder to predict the initial latent code, and then a noise optimization mechanism is implemented to capture high-frequency details and further improve the reconstruction quality. Our multi-stage refinement is inspired by these methods but improves the inversion quality with purely feed-forward-based refinements, thus our speed is much faster than such hybrid approaches.

III. PROPOSED METHOD

In this section, we will first briefly introduce some representative feed-forward networks designed for GAN inversion. Then we will elaborate on the details of our “E2Style” for better efficiency and effectiveness from three aspects: network design, improved loss functions, and multi-stage refinement.

A. Network Structure Design

In Figure 2, we show two representative network structures designed in IDGI [11] and pSp [10] for StyleGAN inversion. All these methods choose the $\mathcal{W}+$ space [4], [5], [10], [7], [11] as the target latent space to invert, which is demonstrated to be more suitable for GAN inversion than the original \mathcal{W} space. It is defined by the cascade of 18 different 512-dimensional vectors $[w_1, \dots, w_{18}]$, $w_i \in \mathcal{W}$.

For the feed-forward network designed in IDGI, it uses a deep backbone network consisting of a series of residual blocks to encode the input image into high-level semantic feature maps (1/64 of the original input resolution) and then uses an extra fully connected (FC) layer to regress the latent vectors based on the last encoded features. However, a too deep network backbone may result in a mismatch between the feature used for inversion and the semantic level of StyleGAN. Similarly, pSp also designs a deep backbone network but leverages the feature pyramid idea [46] to fuse the high-level feature with the low-level feature, then split each latent code into different feature pyramid levels. And for each latent

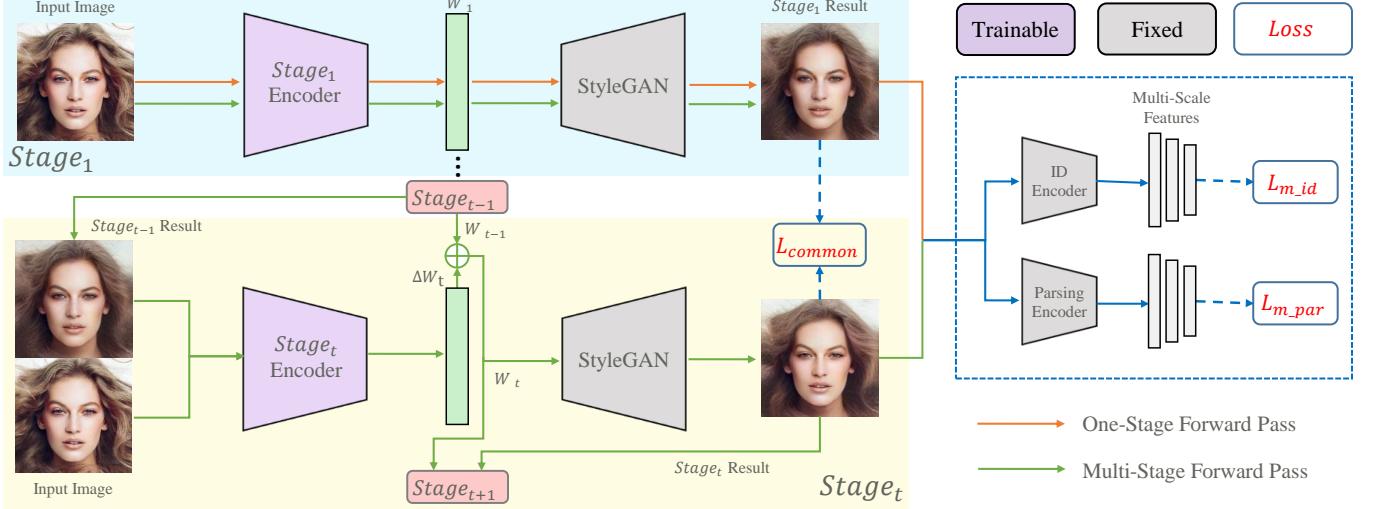


Fig. 3. The overview of our feed-forward network **E2Style** for StyleGAN inversion. Data flow is marked by solid lines and losses by dashed ones. Only the encoder is trainable. Except for the first stage encoder, all other encoders take the original image and the output image of the previous stage as input and output the residual of the latent code that was predicted in the previous stage.

code w_i regression, an independent convolution based head (ConvHead) is used. Depending on the input feature resolution, the ConvHead consists of different numbers of convolutional layers that progressively downsample the features into 1×1 resolution, followed by one FC layer. Therefore, there are a total of 18 ConvHeads in pSp, thus introducing a lot of overheads. Besides, downsampling the features into 1×1 will lose much useful information for predicting the latent code, thus resulting in unsatisfactory reconstruction quality.

For more efficient network design, we ask three questions: “*Is there any guideline for the efficient backbone design?*”, “*How to split the latent code into different feature levels?*”, and “*Can we use cheaper regression heads?*”.

Shallower Backbone. To answer the first question, we propose a simple “semantic level alignment” principle, *i.e.*, *the maximum semantic level of the backbone network should match that of the target GAN model*. pSp [10] downsamples the resolution of the feature to $1/16$ of the input, and IDGI [11] downsamples the feature to $1/64$ of the input image. However, does the network really need to be such deep? In fact, there exists a semantic level best matching point. If the backbone network is too shallow, its encoded feature will not be able to predict the high-level latent codes well. On the other hand, too deep backbone network will not only introduce much overhead but also possibly bring even worse inversion results.

To find the best match point, we follow the recognition network design convention [47], [48] and roughly regard the features of the same downsampled resolution as the same semantic level. Guided by this principle, we conduct a simple ablation on the backbone of pSp by using a single FC head like IDGI, and find the features corresponding to $1/8$ of the input resolution are sufficient to match the maximum semantic level of StyleGAN’s latent codes and can even get better inversion results than the $1/16$ version. This demonstrates that *a deeper backbone network does not mean better inversion results*. In contrast, the shallower encoder can not only bring

the performance improvement but also reduce the model complexity. Moreover, we empirically find the feature pyramid does not help, and we guess it is because the features of deeper layers in the feature pyramid structure of pSp are derived from the features of early layers, and therefore the additional introduction of deeper-layer features in predicting low-dimensional latent codes does not bring extra information gain. Detailed results will be given in the ablation part.

Hierarchical Structure. Given the above backbone network, we get three different semantic levels of features (*i.e.*, $1/8, 1/4, 1/2$). Inspired by pSp, we then split the features from different semantic levels to predict different parts of the latent codes. Considering the feature dimension of higher semantic levels are also higher, assigning more latent codes to higher semantic-level features will have a large model. To reduce the model size while maintaining a good inversion performance, we consider the above assignment problem as a constrained optimization problem. Denote the latent code number assigned to each semantic level to be n_1, n_2, n_3 , it can be formally formulated as:

$$\begin{aligned} n_1^*, n_2^*, n_3^* = & \arg \max_{n_1, n_2, n_3} Q(n_1, n_2, n_3) + \lambda P(n_1, n_2, n_3), \\ \text{s.t., } & n_1 + n_2 + n_3 = 18, \end{aligned} \quad (1)$$

where Q, P denotes the inversion quality and model size function with respect to n_1, n_2, n_3 . Since it is hard to have an explicit function of Q due to its dependency on the training process, we simply adopt a simple binary search algorithm to find the rough optimal values. Specifically, we first regard n_2, n_3 as a whole and find the smallest n_1 with acceptable performance reduction (SSIM decreases by no more than 2% of the original), then we continue to find the smallest value for n_2 in a similar way. After the above search process, we finally adopted a three-layer hierarchical structure to predict the front, middle and tail layers latent code, with the corresponding number of layers n_1, n_2, n_3 being 9, 5, and 4 respectively.

Efficient Prediction Head. As mentioned above, there are 18 complex ConvHeads in pSp, each of which compresses the input feature into 1×1 resolution by a series of convolutional layers with stride of 2, and predict the single-layer latent code. We find such a complex ConvHead is not only unnecessary but also introduces heavy overhead. To keep the regression head lightweight while keeping important information for regression, we design a very simple and efficient head, which just consists of an average pooling layer and a fully connected layer. For the deep (1/8), medium(1/4), and shallow (1/2) features from the hierarchical structure, they will have such a simple head respectively. Besides, to balance performance and overhead, the average pooling layer in the three heads downsamples the input features to the resolution of 7×7 , 5×5 , and 3×3 respectively, which effectively aggregates the information to predict the latent code while reducing the number of model parameters.

B. Loss Functions

In order to train a better GAN inversion network, besides the commonly used losses, we improve the existing GAN inversion loss by introducing two new losses, *i.e.*, multi-layer identity loss and multi-layer face parsing loss.

Common Losses. The common losses consist of two types of constraints from the pixel level and the feature level, respectively. First, we use ℓ_2 loss to provide pixel-level supervision:

$$\mathcal{L}_2 = \|\mathbf{x} - G(E(\mathbf{x}))\|_2, \quad (2)$$

where \mathbf{x} represents the input image, $E(\cdot)$ represents the GAN inversion network, $G(\cdot)$ represents the StyleGAN, and $G(E(\mathbf{x}))$ represents the reconstructed image of the input image. However, only using the ℓ_2 loss will result in blurring reconstruction results, so we choose to use LPIPS [49] as our feature-level loss, which is demonstrated [7] to yield clearer reconstruction results compared to the perceptual loss [50].

$$\mathcal{L}_{LPIPS} = \|F(\mathbf{x}) - F(G(E(\mathbf{x})))\|_2, \quad (3)$$

where $F(\cdot)$ represents the AlexNet [51] feature extractor.

Multi-Layer Identity Loss. As the key face image attribute, keeping the original identity information is extremely important for GAN inversion. Especially for human perception, whether the identity consistency between the inverted image and the original image can be effectively retained is a decisive factor for users to evaluate the quality of face inversion. Therefore, we leverage the multi-layer features from one pre-trained face recognition network (ArcFace [52]) to impose semantic constraints on identity information. According to the resolution size, we choose 5 different levels of features as supervision to better supervise the semantic alignment of the identity information between the reconstructed image and the input image:

$$\mathcal{L}_{m_id} = \sum_{i=1}^5 (1 - \cos(R_i(\mathbf{x}), R_i(G(E(\mathbf{x}))))), \quad (4)$$

where \cos means the cosine similarity and $R_i(\mathbf{x})$ denotes the feature corresponding to the i -th semantic level from the face recognition network R of the input image \mathbf{x} .

Multi-Layer Face Parsing Loss. Because features from the face recognition network often focus on capturing the global identity characteristics, relying on multi-layer identity loss alone cannot achieve accurate reconstructions of local details (*e.g.*, glasses). To provide better local supervision, we introduce multiple layers of features from one pre-trained facial parsing network P [53] to provide a more localized knowledge:

$$\mathcal{L}_{m_par} = \sum_{i=1}^5 (1 - \cos(P_i(\mathbf{x}), P_i(G(E(\mathbf{x}))))), \quad (5)$$

Similar to the well-known perceptual loss, even though the multi-layer identity loss and face parsing loss rely on extra pre-trained networks, we think it is valuable to introduce them into the GAN inversion to push the inversion quality to a new limit. More importantly, as they only appear in the training stage but not inference stage, they are indeed the free lunch for downstream applications. The generalization ability of the proposed loss to non-face domains will be verified in the experiment section.

To summarize, the overall loss function is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_2 + \lambda_2 \mathcal{L}_{LPIPS} + \lambda_3 \mathcal{L}_{m_id} + \lambda_4 \mathcal{L}_{m_par}, \quad (6)$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are set to 1, 0.8, 0.5, 1 respectively by default.

C. Multi-Stage Refinement

As shown in [11], GAN inversion is a challenging problem and often difficult to achieve satisfactory inversion results with a single pass. Therefore, they first use a feed-forward network to get an initial inversion result, then refine it by using the optimization-based method as a post-processing step. Despite better inversion results, such optimization-based refinement is very time-consuming. But it motivates us to introduce feed-forward-based recursive refinement. In this paper, thanks to our better inversion network design, we proposed the multi-stage refinement by purely using the feed-forward inversion networks, thus making the whole process still run in a feed-forward way. The overall framework diagram is shown in Figure 2 (c) and the whole process is formalized as follows:

$$W_t = \begin{cases} E_t(\mathbf{x}) & t = 1 \\ E_t(\mathbf{x}, G(W_{t-1})) + W_{t-1} & t > 1 \end{cases} \quad (7)$$

where E_t represents the network of stage t and $W_t = [w_1^t, \dots, w_{18}^t]$ is the inverted latent code. All the refinement stages ($t > 1$) adopt the same GAN inversion network structure as E_1 but take the concatenation of the original input image and the previous stage inverted image as input. Besides, the learning objective of the refinement stages is changed to predicting the residuals, so as to find a better latent code around the previous stage prediction result.

To sum up, E2Style shares a similar spirit to IDGI's prediction-then-optimization approach. However, E2Style polishes the initial latent code by adding the multi-stage refinement, which requires only one forward pass to output the final result and does not bring a significant increase in time consumption while improving the inversion performance.

IV. EXPERIMENTS

In this section, we first compare E2Style quantitatively and qualitatively with the existing state-of-the-art approaches on the face domain. And the corresponding detailed ablation analysis is then provided to justify the effect of different improvement aspects. Finally, we demonstrate the generality of E2Style on non-face domains.

A. Comparisons on the Face Domain

Implementation Details. For the backbone structure, we follow the SE-ResNet50 [54] backbone but only keep the stages before the 1/16 downsample stage. And we use StyleGAN2 pre-trained on the FFHQ dataset [1] as the target GAN model to inverse. In order to improve the generalization ability of the inversion network, the horizontal flip is used during training. Regarding the training strategy, we first train the first-stage network alone. After its convergence, we freeze it and then train the refinement stage network in a similar way. For all the network training, the base learning rate is set to 0.0001. By default, all network is trained for 25 epochs. Following pSp [10], the Ranger optimizer is used, which is a combination of Rectified Adam [55] with the Lookahead technique [56].

Datasets and Metrics. To evaluate the cross-dataset inversion and editing performance of existing methods, both the baselines and E2Style use the FFHQ dataset [1] with 70,000 faces as the training set, while the qualitative and quantitative comparisons are performed on the CelebA-HQ dataset [38]. For inversion, since the optimization-based method I2S [4] is too time-consuming, we randomly selected 2,000 images from the CelebA-HQ dataset and resized them to the resolution of 256×256 for the evaluation of all methods. For editing, the entire CelebA-HQ dataset is used for quantitative and qualitative comparisons. For the quantitative evaluation of the inversion, five metrics are adopted: Peak Signal-to-Noise Ratio (PSNR) [57], Structural SIMilarity (SSIM) [58], IDentity Similarity (IDS), runtime (using a single NVIDIA GEFORCE RTX 3090 GPU), and model size. For quantitative evaluation of editing quality, we use FID [59] and IDentity Similarity (IDS) as metrics and conduct a user study. For SSIM, PSNR, and IDS, higher indicates better. The method we used here to calculate the identity similarity is Curricularface [60], instead of ArcFace [52] which we used for training.

Inversion Quality. We compare E2Style with the current state-of-the-art GAN inversion approaches, including forward-based methods: pSp [10], IDGI-Encoder [11], ReStyle [45], e4e [61], optimization-based method: I2S [4], and hybrid method: IDGI [11], DNI [8]. For ReStyle, the iterative refinement mechanism is executed 5 times as they recommend. For IDGI, we followed the procedure in their paper: the output of the network was used as the initialization point and the optimization is iterated 100 times. I2S used the mean latent code as initialization and then iterated 1,000 times for each input image. For DNI, the latent code is initialized by the domain-guided encoder, and then the noise optimization process is iterated 100 times.

As shown in Table I, our single-stage inversion network already outperforms all existing single-stage forward-based

TABLE I

QUANTITATIVE COMPARISON OF INVERSION QUALITY ON THE CELEBA-HQ DATASET, IDS MEANS IDENTITY SIMILARITY, * MEANS WITHOUT RESIDUAL LEARNING, C MEANS REPLACING THE AVERAGE POOLING LAYER WITH CONTINUOUS CONVOLUTIONAL LAYERS. OUR E2STYLE ACHIEVES COMPARABLE RESULTS TO OPTIMIZATION-BASED METHODS AND BETTER RESULTS THAN FEED-FORWARD-BASED METHODS WHILE MAINTAINING HIGH EFFICIENCY.

Methods	SSIM	PSNR	IDS	RunTime(s)	Param(M)
pSp	0.58	20.7	0.57	0.09	262
IDGI-Encoder	0.51	18.9	0.19	0.02	165
ReStyle	0.62	21.6	0.67	0.36	201
e4e	0.55	19.5	0.51	0.09	262
E2Style-1Stage	0.63	21.3	0.69	0.04	85
E2Style-1Stage-C	0.62	21.2	0.68	0.04	87
E2Style-2Stage	0.66	22.5	0.74	0.09	170
E2Style-3Stage	0.67	23.0	0.75	0.13	255
E2Style-2Stage*	0.63	21.3	0.69	0.09	170
IDGI	0.62	22.3	0.37	6.60	165
DNI	0.65	21.0	0.55	13.1	525
I2S	0.67	23.9	0.60	54.4	-

TABLE II

QUANTITATIVE COMPARISON OF EDITING QUALITY ON THE CELEBA-HQ DATASET, IDS MEANS IDENTITY SIMILARITY. OUR E2STYLE ACHIEVES THE BEST IDENTITY CONSISTENCY AND HIGHEST PREFERENCE RATES.

Methods	FID	IDS	Preference Rate
pSp	54.4	0.33	12.0%
ReStyle	48.8	0.40	14.8%
e4e	55.5	0.31	18.8%
E2Style	49.4	0.49	54.5%

methods with the smaller model size. With multi-stage refinement, our 2-stage network outperforms the multi-stage forward-based method ReStyle, the hybrid method IDGI, DNI. Our 3-stage network has comparable performance with the optimization-based method I2S in terms of SSIM metric. Moreover, our identity consistency and speed both have obvious advantages over I2S. We further provide some qualitative comparison examples in Figure 4, whose quality rank is consistent with the quantitative results.

Editing Quality. We now compare the editing quality of E2Style with baselines. InterFaceGAN [3] is used to obtain editing directions, and then these are used to edit latent codes obtained by different inversion methods. As shown in Figure 5, compared to pSp [10], e4e [61], Restyle [45], our results are more desirable. For example, when editing the expression, our method better preserves the identity consistency of the image. In terms of age manipulation, our method manipulates other facial semantics less while changing the age of the target face.

In Table II, we provide quantitative comparisons. FID and IDentity Similarity are calculated between input images and edited images. E2Style achieves the best identity consistency while completing the editing. To perceptually measure the performance of different methods for editing, we ask 200 volunteers to conduct a user study with 20 randomly picked result groups, and ask them to select the one which completes the specified editing while retaining the other irrelevant attributes to a large extent. The preference rates are shown in Table II and demonstrate the superiority of our method.

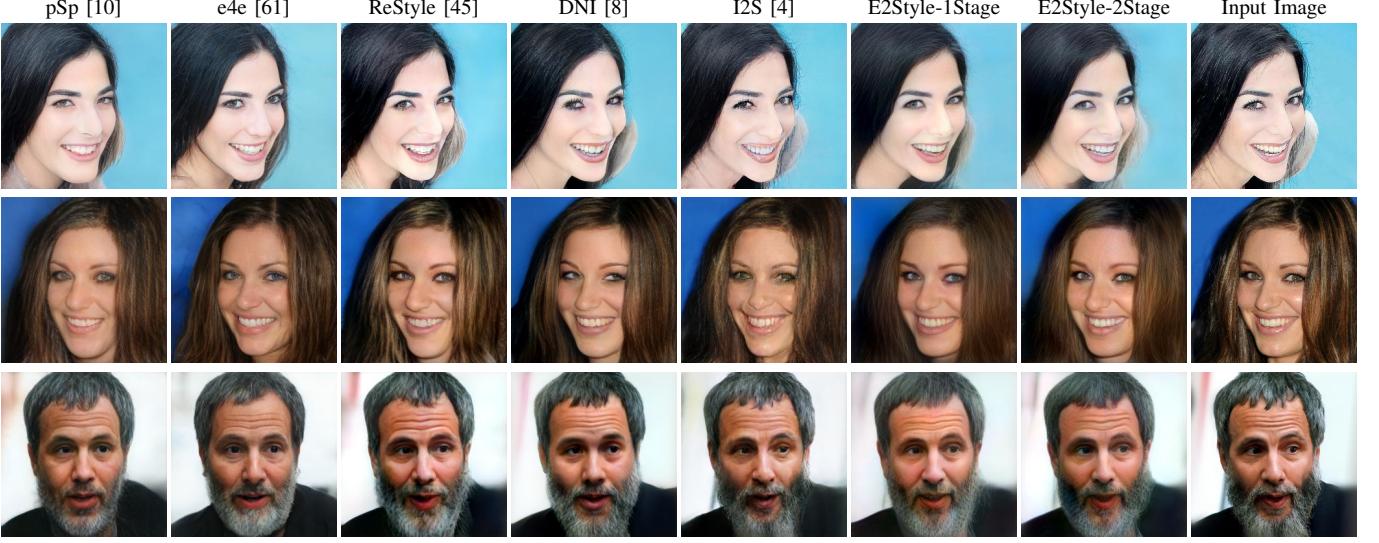


Fig. 4. Visual comparison of the GAN inversion quality on the CelebA-HQ dataset. Our E2Style performs better than existing forward-based methods and comparably to state-of-the-art optimization-based methods while maintaining high efficiency as well as forward-based methods.

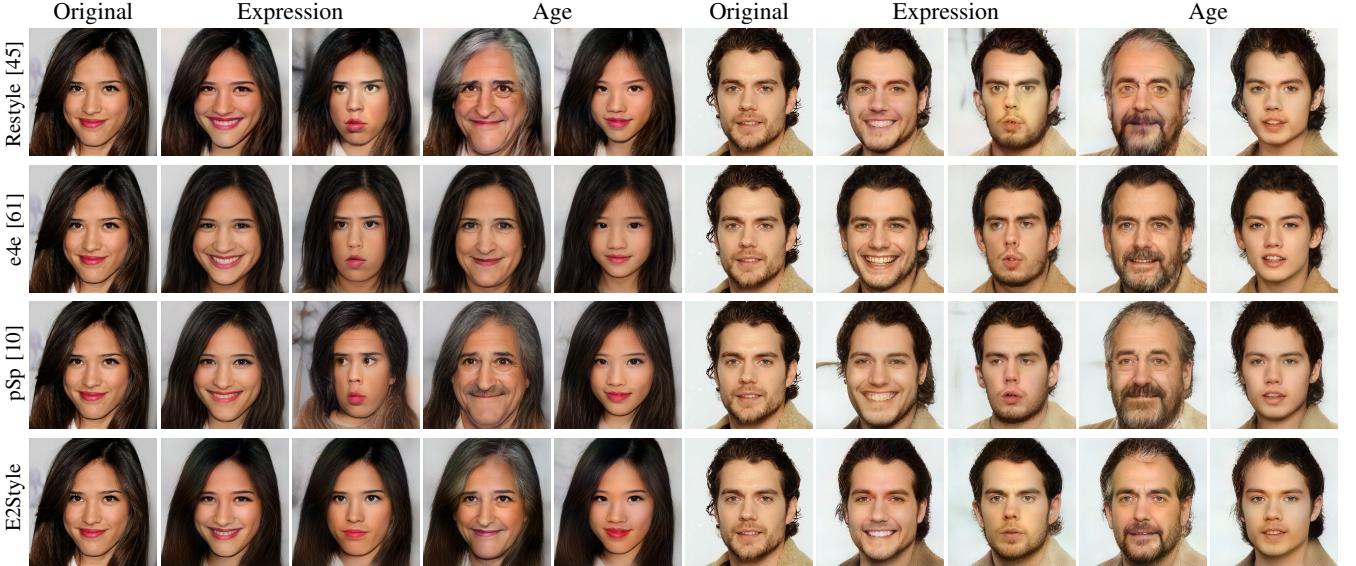


Fig. 5. Qualitative comparison of our E2Style with pSp [10], e4e [61], and Restyle [45] on editing. As can be seen, our approach shows the best identity consistency while completing the editing.

B. Ablation Study

Effectiveness of Network Structure Design. In the top part of Table III, we first validate the effectiveness of the proposed network structure design. (a), (b), and (c) indicate that we used the single efficient prediction head to directly predict the latent code after downsampling the input image to 1/4, 1/16, and 1/8 of the original resolution, respectively. For all these three settings we used common (ℓ_2 , LPIPS) and the single-layer identity loss function to train the network. It clearly shows that deeper networks are not better, and the larger network capacity does not lead to a performance gain. For the inversion task, there is a point where the semantic level of StyleGAN’s latent code is best matched, which may explain the unsatisfactory performance of the IDGI encoder.

It is intuitive to design the inversion network to have the same stage number as StyleGAN and strictly match their semantic levels based on the feature resolution, but our experimental results above show deeper inversion networks even have worse performance. It also indicates that fully relying on the feature resolution for semantic alignment is not a good choice. Moreover, that deep inversion network will be super large and infeasible for real-time applications. Considering these points, we design a relatively shallow inversion network and adopt a three-layer hierarchical structure to predict the latent code as (d). For (e), we took the same scheme as pSp [10], *i.e.*, we used 18 Convheads to predict the latent code of each layer. Compared with (d), (e) brings a larger number of parameters but worse performance. We guess this may be

TABLE III

QUANTITATIVE ABLATION STUDY. SIZE REPRESENTS THE MAXIMUM FACTOR TO DOWNSAMPLE THE INPUT IMAGE. FOR THE PREDICTION HEAD, 1E, 3E, AND 18C REPRESENT THE SINGLE EFFICIENT HEAD, THE THREE-LEVEL EFFICIENT HEADS, AND 18 INDEPENDENT CONVHEADS, RESPECTIVELY. REGARDING TRAINING LOSS, C, SI, MI, AND MP STAND FOR COMMON, SINGLE-LAYER IDENTITY, MULTI-LAYER IDENTITY, AND MULTI-LAYER PARSING LOSS, RESPECTIVELY.

	Size	Head	Loss	SSIM	PSNR	IDS	Param(M)
(a)	$\downarrow 4$	1E	C+SI	0.57	20.3	0.62	59
(b)	$\downarrow 16$	1E	C+SI	0.57	20.5	0.66	262
(c)	$\downarrow 8$	1E	C+SI	0.59	20.9	0.67	133
(d)	$\downarrow 8$	3E	C+SI	0.59	21.0	0.67	85
(e)	$\downarrow 8$	18C	C+SI	0.55	20.0	0.61	233
(f)	$\downarrow 8$	3E	C	0.59	21.0	0.27	85
(g)	$\downarrow 8$	3E	C+MI	0.60	20.9	0.69	85
(h)	$\downarrow 8$	3E	C+MI+MP	0.63	21.3	0.69	85

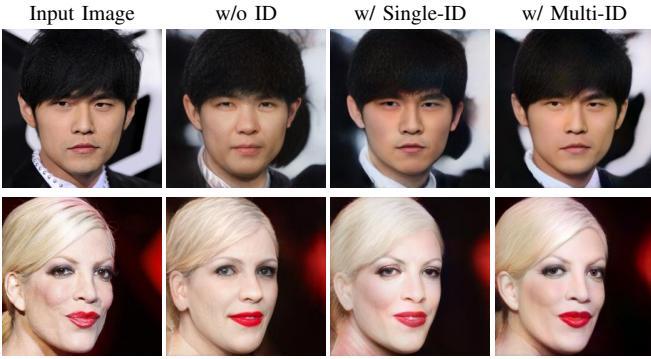


Fig. 6. The effect of multi-layer identity loss. With the multi-layer identity loss, the identity consistency between the reconstructed image and the input image is significantly improved.

because continuous downsampling to the feature resolution of 1×1 will lead to excessive information missing, while using average pooling can effectively aggregate useful information with a shorter path.

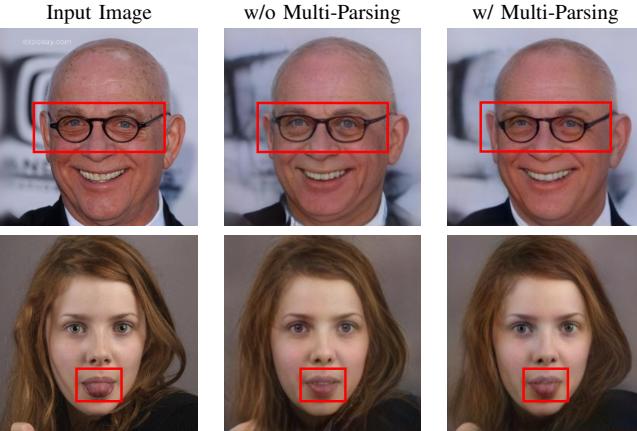


Fig. 7. The effect of multi-layer face parsing loss. With the multi-layer face parsing loss, the network is able to pay attention to the local face regions, thus more accurately reconstructing the local details, e.g., the glasses and tongue of the two cases.

Next, we give the quantitative ablation study about feature pyramid network (FPN). Compared with baseline (d), we add

the same feature pyramid structure as pSp [10] to (d), however, quantitative results (exactly the same SSIM, PSNR, and IDS) prove that this structure does not lead to performance gains. But model parameters are increased from 85M to 98M and the running time is increased from 0.04 seconds to 0.05 seconds. We guess it is because the features of deeper layers in the feature pyramid structure of pSp are derived from the features of early layers, and therefore the additional introduction of deeper-layer features in predicting low-dimensional latent codes does not bring extra information gain.

Finally, we provide an experiment E2Style-1Stage-C which replaces the average pooling layer in E2Style-1Stage with continuous convolutional layers to downsample the features to the same size as E2Style-1Stage. The quantitative results in Table I show that more parameters do not result in performance improvement. Therefore, we choose the more efficient and parameter-free average pooling to downsample the features.

Importance of Multi-Layer Identity and Parsing Losses. The bottom part of Table III shows our quantitative results for removing identity loss, introducing multi-layer identity loss, and introducing both multi-layer identity loss and multi-layer face parsing loss on top of (d), respectively. It well demonstrates the effectiveness of multi-layer identity and face parsing loss. We further provide some qualitative visual comparisons in Figure 6 and Figure 7. With the multi-layer identity loss, the identity consistency between the reconstructed image and the input image is significantly improved. With the multi-layer face parsing loss, the network is able to pay attention to the local face regions, thus more accurately reconstructing the local details, e.g., the glasses and tongue of the two cases.

Significance of the Refinement Stage. As shown in Table I, our 2-stage inversion network significantly improves all the metrics compared to the 1-stage counterpart, but the gain starts to saturate by adding more stages. Considering both speed and performance, we adopt the two-stage inversion network as the default setting. In the last row of Table I, we further change the second stage learning objective to directly predict the latent code instead of the residuals of the first stage latent code. The experimental results show that learning the residuals in the refinement stages is extremely important. In contrast, if we continue to learn the absolute latent code, the refinement stage is still difficult to bring better performance, which may be because the latent code residuals (smaller value range) have smaller variance than the absolute latent codes, thus making the inversion network easier to regress.

C. Comparisons on Non-Face Domains

We will verify the versatility of the proposed improvements on the network design and loss function to non-face domains.

Effectiveness of Our Encoder. For the non-face domains, we use the Stanford Cars [62], LSUN [63] Cat, and LSUN [63] Horse datasets to evaluate the performance of the different methods, with the number of training sets being 8144, 10000, and 10000, respectively, and the number of test sets all being 2,000. To compare the performance of different

TABLE IV

QUANTITATIVE COMPARISON OF DIFFERENT ENCODER STRUCTURES ON THE NON-FACE DOMAINS. NOTE THAT pSp, IDGI AND E2STYLE-1STAGE ALL USE THE SAME TRAINING LOSSES AND STRATEGIES, THE ONLY DIFFERENCE BETWEEN THEM IS THE NETWORK STRUCTURE. IT IS CLEAR THAT OUR NETWORK STRUCTURE ACHIEVES THE BEST PERFORMANCE WITH THE LOWEST NUMBER OF NETWORK PARAMETERS.

Methods	Cats				Horses				Cars			
	SSIM	PSNR	Time(s)	Param(M)	SSIM	PSNR	Time(s)	Param(M)	SSIM	PSNR	Time(s)	Param(M)
pSp	0.39	18.1	0.06	206	0.33	17.1	0.06	206	0.40	16.2	0.06	234
IDGI	0.40	18.4	0.02	165	0.34	17.7	0.02	165	0.41	16.7	0.03	201
E2Style-1Stage	0.41	18.5	0.02	85	0.37	17.8	0.02	85	0.43	16.8	0.03	95

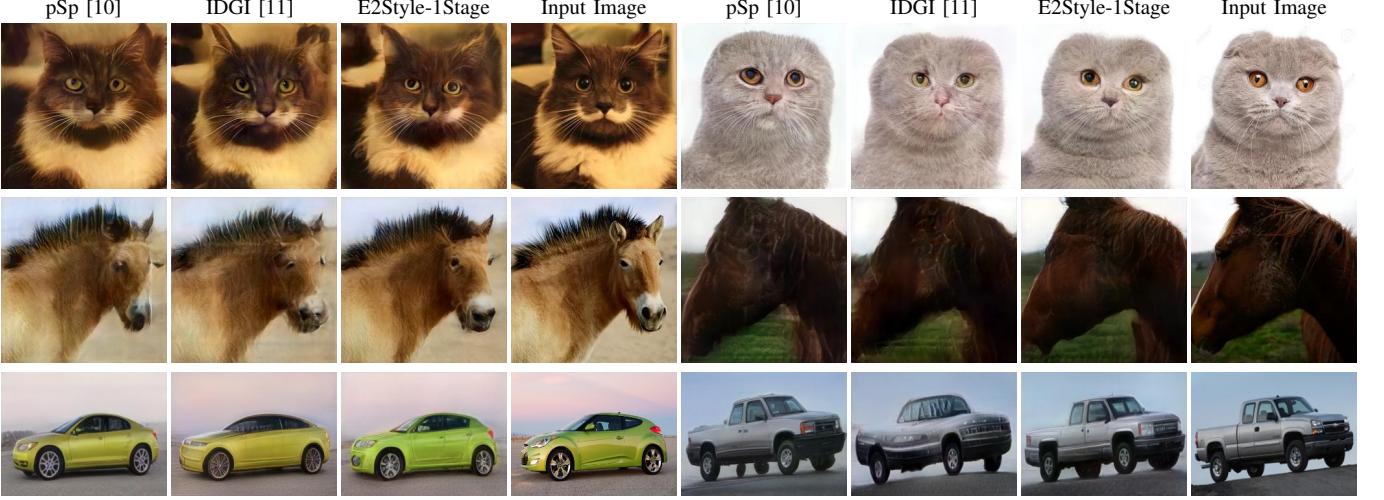


Fig. 8. Visual comparison of the GAN inversion quality on the non-face domains. Thanks to our improvements in network structure, our encoder generates higher quality reconstruction results with the lowest network parameters compared to pSp, IDGI.

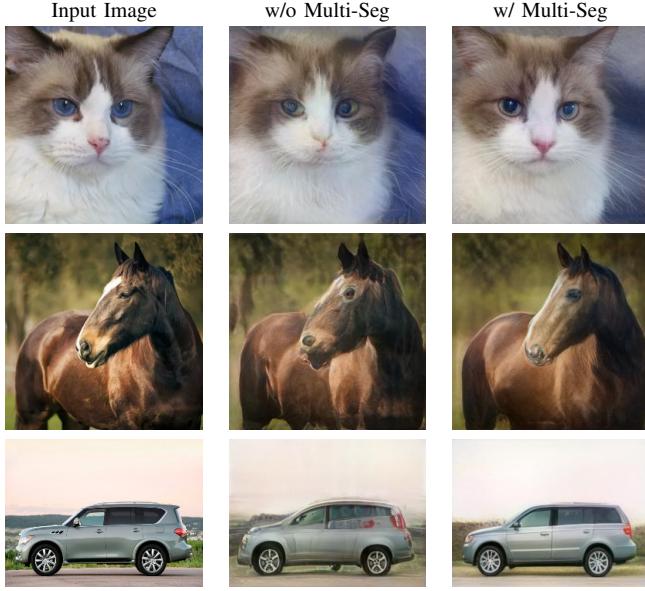


Fig. 9. The effect of multi-layer segmentation loss. It enables the network to pay more attention to salient objects that are of more interest to humans, resulting in a large improvement in perceptual quality.

encoder network structures, we retrain pSp [10], IDGI [11], and our encoder on these datasets. These methods all use common (ℓ_2 , LPIPS) losses as training constraints and the training strategies for these methods are completely identical,

which are consistent with the strategy we use for the face domain. Note that for each non-face domain we use the official StyleGAN2 as the generator, which is trained by images from that domain.

Quantitative and qualitative comparisons are shown in Table IV and Figure 8. Obviously, doing inversion tasks on these non-face domains is more challenging due to the more complex and diverse datasets. However, this does not hinder us from comparing the performance of different network structure designs. Thanks to our improvements in network structure, our encoder has the best performance and the lowest network parameters of all forward-based methods.

Extensibility of Multi-Layer Face Parsing Loss. The essence of the multi-layer face parsing loss is showing that local supervision is useful for GAN inversion. By replacing the multi-layer face parsing loss into multi-layer segmentation loss with a pretrained segmentation model [66], we find the perceptual quality of inversion results on these non-face datasets can also be significantly improved, although there is a slight decrease in PSNR, SSIM. As shown in Figure 9, the introduction of multi-layer segmentation loss enables the network to pay more attention to salient objects that are of more interest to humans, resulting in a large improvement in perceptual quality.

V. APPLICATIONS

GAN inversion provides the community with new ideas and perspectives for solving image editing problems. With our

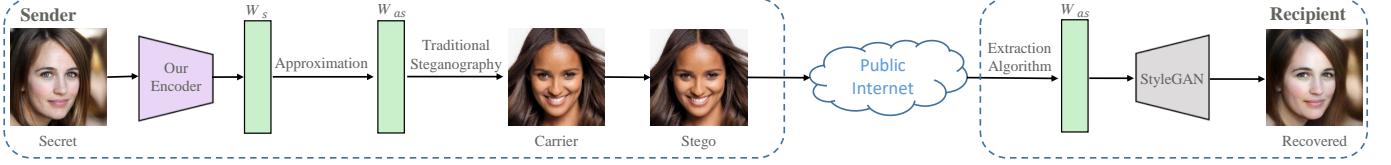


Fig. 10. The illustration of our proposed secure deep hiding. By combining the proposed GAN inversion method with traditional steganography, we can easily implement the application of securely hiding an image with 1024×1024 resolution to another image.

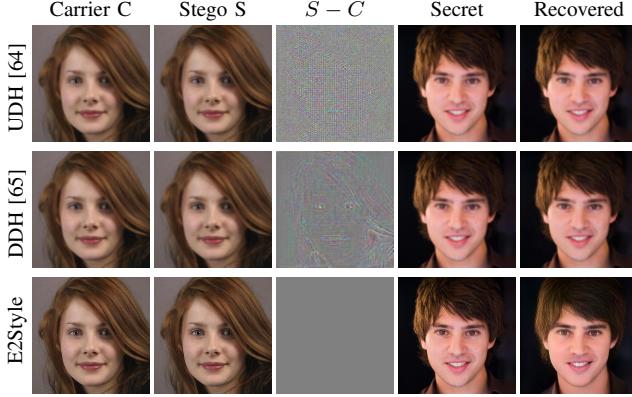


Fig. 11. Qualitative comparison of deep hiding. Stego is obtained by hiding Secret into the Carrier. Please zoom in to get a better view of the residuals between the stego image S and the original carrier image C .

fast and accurate GAN inversion technique as the foundation, many downstream image processing tasks will be benefited greatly. In this section, we show some interesting and practical applications to demonstrate the potentials of our E2Style, including: secure deep hiding, image manipulation, image restoration, and image translation. Except image translation tasks, all applications use FFHQ [1] as the training dataset and use CelebA-HQ [38] as the test dataset.

A. Secure Deep Hiding

Algorithm 1: Secure Deep Hiding

- Input:** secret image I_s ; carrier image I_c ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder $E(\cdot)$; traditional steganography embedding and extraction algorithms $Emb(\cdot, \cdot)$, $Ext(\cdot)$.
Output: reconstructed secret image I_r .
- 1 $W_s = E(I_s)$;
 - 2 $W_{as} = \lfloor W_s \rfloor$;
 - 3 $I_{cs} = Emb(I_c, W_{as})$;
 - 4 The sender sends I_{cs} , then the recipient receives I_{cs} ;
 - 5 $W_{as} = Ext(I_{cs})$;
 - 6 $I_r = G(W_{as})$.

In this application, the goal is to hide one secret image into one carrier image in an imperceptible way. In contrast to traditional steganography [67], [68], [69], which embeds secret messages with small capacity, deep hiding aims at large capacity hidden messages, such as hiding a secret image with

1024×1024 resolution into a carrier image. Existing deep hiding schemes [65], [64], [70] either directly concatenate two images together and feed them into the network, or feed only the secret image into the network for the carrier-agnostic purpose, which have weak imperceptibility and undetectability, therefore the security cannot be guaranteed. Empowered by our excellent inversion quality, we can combine the proposed method with traditional steganography to propose a novel application: secure deep hiding, which overcomes the aforementioned drawbacks. Our proposed secure deep hiding framework is shown in Figure 10. Alg. 1 shows the details of the algorithm. Specifically, the whole hiding process is divided into four steps:

- 1) The sender selects the secret image to be sent and then uses the proposed GAN inversion method to get its latent code $W_s \in \mathbb{R}^{18 \times 512}$. In order to reduce the amount of secret message data while not causing a significant visual impact on the reconstruction result, the sender keeps the integer part of W_s to obtain $W_{as} \in \mathbb{Z}^{18 \times 512}$.
- 2) The sender selects a carrier image, hides W_{as} in the carrier image using the traditional steganography method [71] to get the stego image I_{cs} , and then sends the stego image out.
- 3) After receiving the stego image, the recipient obtains W_{as} accurately using the corresponding extraction algorithm [71].
- 4) The recipient feeds W_{as} to the StyleGAN which has the same network weight as the one inverted by the sender to reconstruct the secret image.

In this process, the pre-trained StyleGAN indeed acts as the encryption and decryption codebook.

Figure 11 shows the visual comparison of our method with universal deep hiding (UDH) [64] and dependent deep hiding (DDH) [65]. With the benefit of the proposed high-quality GAN inversion method, our deep hiding scheme achieves comparable reconstruction quality with UDH and DDH. But as shown in the third column of Figure 11, UDH and DDH either expose the secret image or the carrier image, while our scheme has better imperceptibility and undetectability. More importantly, the traditional steganography we use provides theoretical security, whereas UDH and DDH do not have it because they are both based on deep learning networks. To summarize, by combining the proposed GAN inversion method with traditional steganography, we can easily implement the application of securely hiding an image with 1024×1024 resolution to another image, which is referred to as *Secure Deep Hiding*.

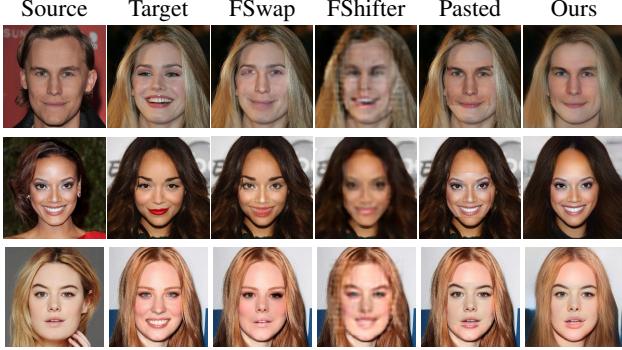


Fig. 12. Comparison of our approach with FSwap [72] and FShifter [73] on semantic face swapping.

B. Image Manipulation

Algorithm 2: Semantic Face Swapping

Input: source image I_s ; target image I_t ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder $E(\cdot)$.

Output: identity-swapped face image I_{is} .

- 1 Crop out the center area of I_s and paste it onto I_t to get the pasted image I_p ;
 - 2 $I_{is} = G(E(I_p))$.
-

Semantic Face Swapping. Semantic face swapping is intended to replace the identity of the target image with that of the source image, however, it does not guarantee strict alignment of facial attributes such as expression, lighting, head pose, etc. With the pretrained StyleGAN2 as a strong face reconstruction prior, we find our method can also be used to create more natural face swapping results easily. Alg. 2 shows the details of the algorithm. Specifically, we crop out the central area of the source face image and paste it directly into the same position of the target face image to create a pasted image. To eliminate the obvious artifacts present in the pasted image, we use the proposed GAN inversion technique to reconstruct the pasted image. As shown in Figure 12, the reconstructed image perfectly blends the pasted face with its surrounding area. Compared to traditional geometric transformation based FaceSwap [72] and learning-based FaceShifter [73], our results have higher identity consistency with the source image and better image quality.

Algorithm 3: Style Mixing

Input: style image I_s ; content image I_c ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder $E(\cdot)$; style mixing operation $Mix(\cdot, \cdot)$.

Output: style mixing result image I_m .

- 1 $I_m = G(Mix(E(I_s), E(I_c)))$.
-

Style Mixing. Style mixing aims to transfer the low-level appearance characteristics from the style image to the content image. Alg. 3 shows the details of the algorithm. In detail, after obtaining the respective latent codes of the style image and the content image, we replace the last 11 layers of the



Fig. 13. Style mixing results. All the content images are coated with the same color lipstick as the style image.

latent codes corresponding to the content image with those of the style image. A specific example is shown in Figure 13, where all the content images are coated with the same color lipstick as the style image.

Algorithm 4: Face Interpolation

Input: real image I_A ; real image I_B ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder $E(\cdot)$; blending parameter λ .

Output: face interpolation result image I_i .

- 1 **for** $\lambda = 0$ to 1 **do**
 - 2 $I_i = G(\lambda E(I_B) + (1 - \lambda)E(I_A))$;
 - 3 **end**
-

Face Interpolation. Given two real-world images A and B, Face interpolation is intended to complete the gradual morphing from input A to input B. Alg. 4 shows the details of the algorithm. we use the proposed GAN inversion method to obtain their respective latent codes $W_A, W_B \in \mathcal{W}^+$. Then, we combine the two latent codes by linear weighting to generate the intermediate latent code $W_I = \lambda W_B + (1 - \lambda)W_A$. Finally, the image corresponding to the intermediate latent code is generated. By gradually increasing the blending parameter λ from 0 to 1, we can achieve the gradual morphing effect from input A to input B, as shown in Figure 14.

C. Image Restoration

Algorithm 5: Image Restoration

Input: impaired image I_i ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder for the specified restoration task $E_r(\cdot)$.

Output: restored image I_r .

- 1 $I_r = G(E_r(I_i))$.
-

By introducing corresponding data augmentations to the input images during the training phase, our E2Style can be easily extended to perform a number of image restoration tasks, such as colorization, inpainting, super resolution. Alg. 5 shows the details of the algorithm in the inference phase. Here we qualitatively compare our E2Style with pSp [10], e4e [61],

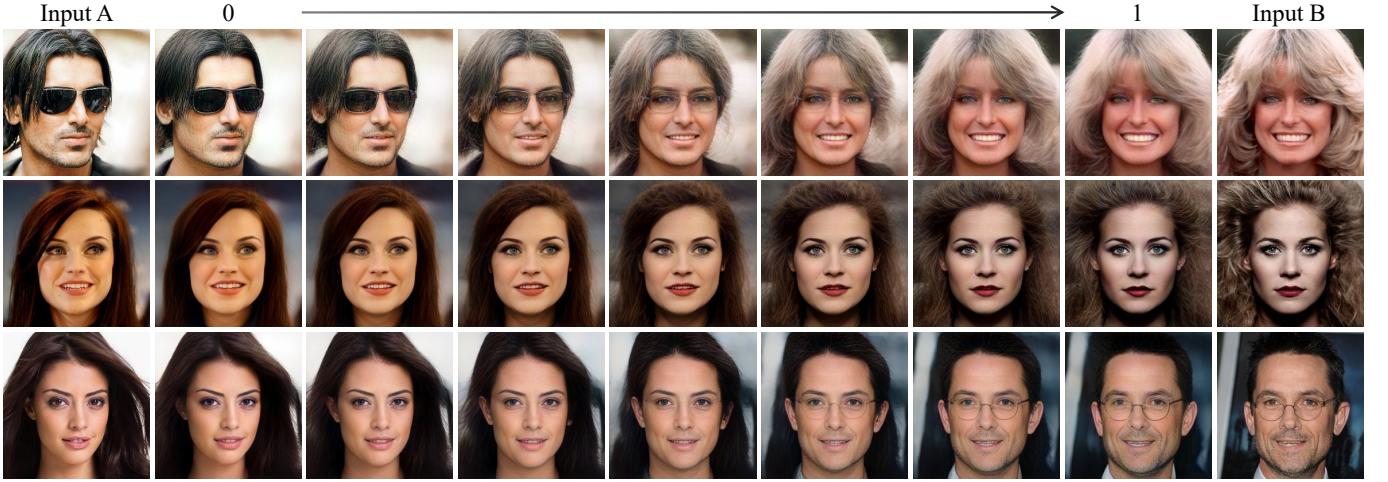


Fig. 14. Face interpolation results. By gradually increasing the blending parameter λ from 0 to 1, we achieve the morphing effect from input A to B.

and Restyle [45] on these tasks, both of which use the same data augmentations. Figure 15 shows the visual comparison of our E2Style with other methods on these three tasks.

Colorization. Image colorization aims to convert the single-channel gray-scale image into a semantically reasonable three-channel color image. For the colorization task, we use the color image as the target and the corresponding grayscale image as input when training the network, so the goal of the encoder is no longer to reconstruct the input image accurately but to find the most appropriate latent code which can colorize the input grayscale image reasonably. Compared with other methods, E2Style accomplishes reasonable coloring for the grayscale image while ensuring consistency in identity and local details (*e.g.*, glasses).

Inpainting. Given an incomplete image, the task of inpainting is to recover the missing pixels so that the recovered part is compatible with the known pixels. To perform inpainting using the GAN inversion framework, we employ a degraded transformation to the input image, in which a region is randomly selected and the pixel value of that region is set to 0 during the training process. As illustrated in Figure 15, E2Style achieves reasonable filling of unknown regions while keeping existing pixel values unchanged as much as possible.

Super Resolution. The goal of super resolution is to reconstruct the corresponding high-resolution image from the observed low-resolution image. Following the data augmentation method of pSp, we randomly downsample the input image by $\times 1$, $\times 2$, $\times 4$, $\times 8$, and $\times 16$ and use the original resolution image as the target during training. Compared to other methods (Figure 15), our results are more realistic and with better facial details, *e.g.*, the eyes in the shown example are more faithfully aligned with the low-resolution image, and the glasses of another example are faithfully reconstructed.

D. Image Translation

By replacing the input with the corresponding sketch or segmentation label map of the image during the network

Algorithm 6: Image Translation

Input: sketch or segmentation map I_s ; a pretrained StyleGAN2 generator $G(\cdot)$; our encoder for the specified image translation task $E_t(\cdot)$.
Output: translated image I_t .

$$1 \quad I_t = G(E_t(I_s)).$$

training phase, our framework can also perform image translation tasks. Given that there is no relevant dataset for the FFHQ, we perform this task on the CelebA-HQ, of which 3,000 images are reserved for testing purposes. Specifically, we follow [10] to generate the sketch dataset of CelebA-HQ, while the segmentation label maps are from the CelebAMask-HQ dataset [74]. Regarding the training loss, it is the same as the GAN inversion task except that the multi-layer identity loss is removed. Alg. 6 shows the details of the algorithm in the inference phase. The visual comparison results are shown in Figure 15. Compared to other methods, it is clear that our results are more faithfully aligned to the respective semantics of the input, *e.g.* the glasses in the sketch-to-image example.

VI. CONCLUSION

In this paper, we propose E2Style, which improves GAN inversion in terms of efficiency and effectiveness. Such improvements come from three aspects: 1) designing a more efficient GAN inversion network with a shallow backbone, hierarchical latent code regression, and efficient prediction heads; 2) introducing multi-layer identity loss and multi-layer parsing loss; and 3) purely feed-forward-based multi-stage refinement. Extensive evaluation and applications demonstrate that our E2Style performs much better than existing feed-forward-based methods and comparably to state-of-the-art optimization-based methods with higher efficiency. In the future, more emphasis should be put on creating a universal approach to encode all kinds of images into the latent space of a StyleGAN model pre-trained in the corresponding domain.

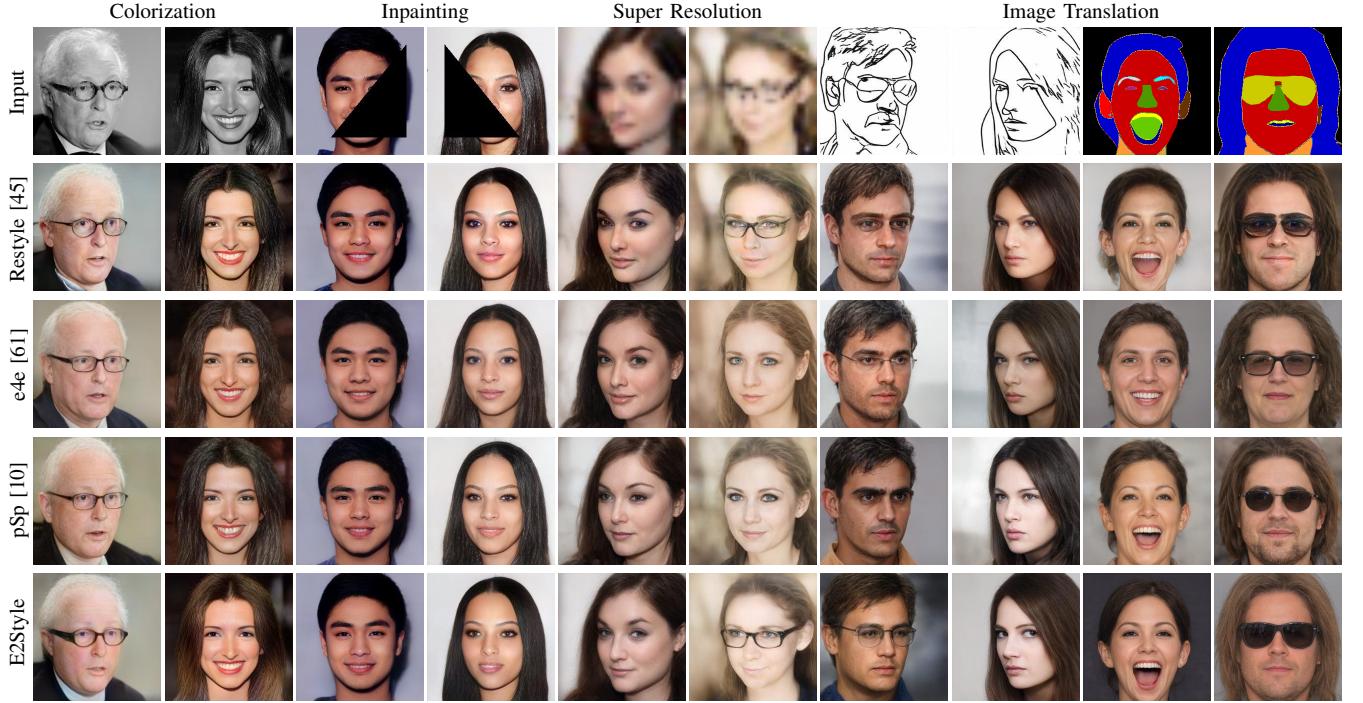


Fig. 15. Comparison of our E2Style with pSp [10], e4e [61], and Restyle [45] on image restoration including colorization, inpainting, super resolution and image translation tasks.

REFERENCES

- [1] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [2] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8107–8116, 2020.
- [3] Y. Shen, J. Gu, X. Tang, and B. Zhou, “Interpreting the latent space of gans for semantic face editing,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9240–9249, 2020.
- [4] R. Abdal, Y. Qin, and P. Wonka, “Image2stylegan: How to embed images into the stylegan latent space?” *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4431–4440, 2019.
- [5] ———, “Image2stylegan++: How to edit the embedded images?” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8293–8302, 2020.
- [6] Y. Nitzan, A. Bermano, Y. Li, and D. Cohen-Or, “Face identity disentanglement via latent space mapping,” *ACM Transactions on Graphics (TOG)*, vol. 39, pp. 1 – 14, 2020.
- [7] S. Guan, Y. Tai, B. Ni, F. Zhu, F. Huang, and X. Yang, “Collaborative learning for faster stylegan embedding,” *ArXiv*, vol. abs/2007.01758, 2020.
- [8] N. Yang, Z. Zheng, M. Zhou, X. Guo, L. Qi, and T. Wang, “A domain-guided noise-optimization-based inversion method for facial image manipulation,” *IEEE Transactions on Image Processing*, vol. 30, pp. 6198–6211, 2021.
- [9] X. Pan, X. Zhan, B. Dai, D. Lin, C. C. Loy, and P. Luo, “Exploiting deep generative prior for versatile image restoration and manipulation,” *ArXiv*, vol. abs/2003.13659, 2020.
- [10] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, “Encoding in style: A stylegan encoder for image-to-image translation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 2287–2296.
- [11] J. Zhu, Y. Shen, D. li Zhao, and B. Zhou, “In-domain gan inversion for real image editing,” in *ECCV*, 2020.
- [12] D. Bau, J.-Y. Zhu, J. Wulff, W. Peebles, H. Strobelt, B. Zhou, and A. Torralba, “Seeing what a gan cannot generate,” *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4501–4510, 2019.
- [13] D. Bau, H. Strobelt, W. Peebles, J. Wulff, B. Zhou, J.-Y. Zhu, and A. Torralba, “Semantic photo manipulation with a generative image prior,” *ACM Transactions on Graphics (TOG)*, vol. 38, pp. 1 – 11, 2019.
- [14] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, “Generative adversarial nets,” in *NIPS*, 2014.
- [15] N.-T. Tran, V.-H. Tran, N.-B. Nguyen, T.-K. Nguyen, and N.-M. Cheung, “On data augmentation for gan training,” *IEEE Transactions on Image Processing*, vol. 30, pp. 1882–1897, 2021.
- [16] A. F. Ansari, J. Scarlett, and H. Soh, “A characteristic function approach to deep implicit generative modeling,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7476–7484, 2020.
- [17] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *ICML*, 2017.
- [18] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, “Least squares generative adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.
- [19] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” *ArXiv*, vol. abs/1802.05957, 2018.
- [20] A. Brock, J. Donahue, and K. Simonyan, “Large scale gan training for high fidelity natural image synthesis,” *ArXiv*, vol. abs/1809.11096, 2019.
- [21] Y. Qin, N. Mitra, and P. Wonka, “How does lipschitz regularization influence gan training?” in *European Conference on Computer Vision*. Springer, 2020, pp. 310–326.
- [22] E. Schönfeld, B. Schiele, and A. Khoreva, “A u-net based discriminator for generative adversarial networks,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8204–8213, 2020.
- [23] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” in *NIPS*, 2017.
- [24] Z. Wan, B. Zhang, D. Chen, P. Zhang, D. Chen, J. Liao, and F. Wen, “Bringing old photos back to life,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2747–2757.
- [25] W. R. Tan, C. S. Chan, H. E. Aguirre, and K. Tanaka, “Improved artgan for conditional synthesis of natural image and artwork,” *IEEE Transactions on Image Processing*, vol. 28, pp. 394–409, 2019.

- [26] C.-C. Hsu, C.-W. Lin, W.-T. Su, and G. Cheung, "Sigan: Siamese generative adversarial network for identity-preserving face hallucination," *IEEE Transactions on Image Processing*, vol. 28, pp. 6225–6236, 2019.
- [27] H. Xu, J. Ma, and X. Zhang, "Mef-gan: Multi-exposure image fusion via generative adversarial networks," *IEEE Transactions on Image Processing*, vol. 29, pp. 7203–7216, 2020.
- [28] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-gan: Adversarial gated networks for multi-collection style transfer," *IEEE Transactions on Image Processing*, vol. 28, pp. 546–560, 2019.
- [29] X. Gao, Y. jie Tian, and Z. Qi, "Rpd-gan: Learning to draw realistic paintings with generative adversarial network," *IEEE Transactions on Image Processing*, vol. 29, pp. 8706–8720, 2020.
- [30] A. Lucas, S. Lx00F3pez-Tapia, R. Molina, and A. Katsaggelos, "Generative adversarial networks and perceptual losses for video super-resolution," *IEEE Transactions on Image Processing*, vol. 28, pp. 3312–3327, 2019.
- [31] K. Liu, Z. Ye, H. Guo, D. Cao, L. Chen, and F.-Y. Wang, "Fiss gan: A generative adversarial network for foggy image semantic segmentation," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 8, pp. 1428–1439, 2021.
- [32] K. Zhang, Y. Su, X. Guo, L. Qi, and Z. Zhao, "Mu-gan: Facial attribute editing based on multi-attention mechanism," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 9, pp. 1614–1626, 2020.
- [33] J. Li, Y. Tao, and T. Cai, "Predicting lung cancers using epidemiological data: A generative-discriminative framework," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 5, pp. 1067–1078, 2021.
- [34] Z. Tan, M. Chai, D. Chen, J. Liao, Q. Chu, L. Yuan, S. Tulyakov, and N. Yu, "Michigan: multi-input-conditioned hair image generation for portrait editing," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 95–1, 2020.
- [35] Z. Tan, D. Chen, Q. Chu, M. Chai, J. Liao, M. He, L. Yuan, G. Hua, and N. Yu, "Efficient semantic image synthesis via class-adaptive normalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- [36] Z. Tan, M. Chai, D. Chen, J. Liao, Q. Chu, B. Liu, G. Hua, and N. Yu, "Diverse semantic image synthesis via probability distribution modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.
- [37] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.
- [38] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *ArXiv*, vol. abs/1710.10196, 2018.
- [39] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," *ArXiv*, vol. abs/2007.06600, 2020.
- [40] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable gan controls," *ArXiv*, vol. abs/2004.02546, 2020.
- [41] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, "Cross-domain and disentangled face manipulation with 3d guidance," *IEEE Transactions on Visualization and Computer Graphics*, 2021.
- [42] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code gan prior," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3009–3018, 2020.
- [43] C. Bartz, J. Bethge, H. Yang, and C. Meinel, "One model to reconstruct them all: A novel way to use the stochastic noise in stylegan," *ArXiv*, vol. abs/2010.11113, 2020.
- [44] N. Yang, M. Zhou, B. Xia, X. Guo, and L. Qi, "Inversion based on a detached dual-channel domain method for stylegan2 embedding," *IEEE Signal Processing Letters*, vol. 28, pp. 553–557, 2021.
- [45] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "Restyle: A residual-based stylegan encoder via iterative refinement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.
- [46] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.
- [47] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
- [48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [49] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.
- [50] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*, 2016.
- [51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84 – 90, 2012.
- [52] J. Deng, J. Guo, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4685–4694, 2019.
- [53] Z. Liu, https://github.com/switchablenorms/CelebAMask-HQ/tree/master/face_parsing, accessed: Mar. 2021. [Online].
- [54] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 2011–2023, 2020.
- [55] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *ArXiv*, vol. abs/1908.03265, 2020.
- [56] M. R. Zhang, J. Lucas, G. E. Hinton, and J. Ba, "Lookahead optimizer: k steps forward, 1 step back," in *NeurIPS*, 2019.
- [57] A. Horé and D. Ziou, "Image quality metrics: Psnr vs. ssim," *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369, 2010.
- [58] Z. Wang, A. Bovik, H. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 2004.
- [59] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *NIPS*, 2017.
- [60] Y. Huang, Y. Wang, Y. Tai, X. Liu, P. Shen, S. xin Li, J. Li, and F. Huang, "Curricularface: Adaptive curriculum learning loss for deep face recognition," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5900–5909, 2020.
- [61] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for stylegan image manipulation," *ACM Transactions on Graphics (TOG)*, vol. 40, pp. 1 – 14, 2021.
- [62] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," *2013 IEEE International Conference on Computer Vision Workshops*, pp. 554–561, 2013.
- [63] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.
- [64] C. Zhang, P. Benz, A. Karjauv, G. Sun, and I.-S. Kweon, "Udh: Universal deep hiding for steganography, watermarking, and light field messaging," in *NeurIPS*, 2020.
- [65] S. Baluja, "Hiding images in plain sight: Deep steganography," in *NIPS*, 2017.
- [66] V. Nekrasov, C. Shen, and I. Reid, "Light-weight refinenet for real-time semantic segmentation," in *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6*, 2018.
- [67] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 234–239, 2012.
- [68] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, pp. 1–13, 2014.
- [69] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 221–234, 2016.
- [70] J. Zhang, D. Chen, J. Liao, H. Fang, W. Zhang, W. Zhou, H. Cui, and N. Yu, "Model watermarking for image processing networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 805–12 812.
- [71] T. Filler, J. Judas, and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Transactions on Information Forensics and Security*, vol. 6, pp. 920–935, 2011.
- [72] FaceSwap, <https://github.com/wuhukai/FaceSwap>, accessed: Feb. 2021. [Online].
- [73] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen, "Faceshifter: Towards high fidelity and occlusion aware face swapping," *ArXiv*, vol. abs/1912.13457, 2019.
- [74] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.