Delving StyleGAN Inversion for Image Editing: A Foundation Latent Space Viewpoint

Hongyu Liu\textsuperscript{1} Yibing Song\textsuperscript{2*} Qifeng Chen\textsuperscript{1*}
\textsuperscript{1}Hong Kong University of Science and Technology \textsuperscript{2}AI Institute, Fudan University

\texttt{hliudq@cse.ust.hk} \texttt{yibingsong.cv@gmail.com} \texttt{cqf@ust.hk}

\begin{figure}[h]
\centering
\begin{tabular}{cccccc}
Input & Inversion & Beard & Smile↓ & Lip & Eye openness↓ \\
\end{tabular}
\caption{The inversion and editing results of our model in the real images. We show from the left to right of each row: an input image, inversion results, and our editing results. We edit images by modifying the attributes in the embedding space following \cite{21, 54}. The ↓ means a decreased magnitude of the manipulation attribute.}
\end{figure}

\section*{Abstract}

GAN inversion and editing via StyleGAN maps an input image into the embedding spaces (\(W\), \(W^+\), and \(F\)) to simultaneously maintain image fidelity and meaningful manipulation. From latent space \(W\) to extended latent space \(W^+\) to feature space \(F\) in StyleGAN, the editability of GAN inversion decreases while its reconstruction quality increases. Recent GAN inversion methods typically explore \(W^+\) and \(F\) rather than \(W\) to improve reconstruction fidelity while maintaining editability. As \(W^+\) and \(F\) are derived from \(W\) that is essentially the foundation latent space of StyleGAN, these GAN inversion methods focusing on \(W^+\) and \(F\) spaces could be improved by stepping back to \(W\). In this work, we propose to first obtain the proper latent code in foundation latent space \(W\). We introduce contrastive learning to align \(W\) and the image space for proper latent code discovery. Then, we leverage a cross-attention encoder to transform the obtained latent code in \(W\) into \(W^+\) and \(F\), accordingly. Our experiments show that our exploration of the foundation latent space \(W\) improves the representation ability of latent codes in \(W^+\) and features in \(F\), which yields state-of-the-art reconstruction fidelity and editability results on the standard benchmarks. Project page: \url{https://kumapowerliu.github.io/CLCAE}.

1. Introduction

StyleGAN [29–31] achieves numerous successes in image generation. Its semantically disentangled latent space enables attribute-based image editing where image content is modified based on the semantic attributes. GAN inversion \cite{62} projects an input image into the latent space, which benefits a series of real image editing methods \cite{4, 36, 49, 65, 72}. The crucial part of GAN inversion is to find the inversion space to avoid distortion while enabling editability. Prevalent inversion spaces include the latent space \(W^+\) \cite{1} and the feature space \(F\) \cite{28}. \(W^+\) is shown to balance distortion and editability \cite{56, 71}. It attracts many editing methods \cite{1, 2, 5, 20, 25, 53} to map real images into this latent space. On the other hand, \(F\) contains spatial im-
age representation and receives extensive studies from the image embedding [28, 48, 59, 63] or StyleGAN’s parameters [6,14] perspectives.

The latent space \( W^+ \) and feature space \( F \) receive wide investigations. In contrast, Karras et al. [31] put into exploring \( W \) and the results are unsatisfying. This may be because that manipulation in \( W \) will easily bring content distortions during reconstruction [56], even though \( W \) is effective for editability. Nevertheless, we observe that \( W^+ \) and \( F \) are indeed developed from \( W \), which is the foundation latent space in StyleGAN. In order to improve image editability while maintaining reconstruction fidelity (i.e., \( W^+ \) and \( F \)), exploring \( W \) is necessary. Our motivation is similar to the following quotation:

“\( \text{You can’t build a great building on a weak foundation. You must have a solid foundation if you’re going to have a strong superstructure.} \)”

—Gordon B. Hinckley

In this paper, we propose a two-step design to improve the representation ability of the latent code in \( W^+ \) and \( F \). First, we obtain the proper latent code in \( W \). Then, we use the latent code in \( W \) to guide the latent code in \( W^+ \) and \( F \). In the first step, we propose a contrastive learning paradigm to align the \( W \) and image space. This paradigm is derived from CLIP [51] where we switch the text branch with \( W \). Specifically, we construct the paired data that consists of one image \( I \) and its latent code \( w \in W \) with pre-trained StyleGAN. During contrastive learning, we train two encoders to obtain two feature representations of \( I \) and \( w \), respectively. These two features are aligned after the training process. During GAN inversion, we fix this contrastive learning module and regard it as a loss function. This loss function is set to make the one real image and its latent code \( w \) sufficiently close. This design improves existing studies [31] on \( W \) that their loss functions are set on the image space (i.e., similarity measurement between an input image and its reconstructed image) rather than the unified image and latent space. The supervision on the image space only does not enforce well alignment between the input image and its latent code in \( W \).

After discovering the proper latent code in \( W \), we leverage a cross-attention encoder to transform \( w \) into \( w^+ \in W^+ \) and \( f \in F \). When computing \( w^+ \), we set \( w \) as the query and \( w^+ \) as the value and key. Then, we calculate the cross-attention map to reconstruct \( w^+ \). This cross-attention map enforces the value \( w^+ \) close to the query \( w \), which enables the editability of \( w^+ \) to be similar to that of \( w \). Besides, \( w^+ \) is effective in preserving the reconstruction ability. When computing \( f \), we set the \( w \) as the value and key, while setting \( f \) as the query. So \( w \) will guide \( f \) for feature refinement. Finally, we use \( w^+ \) and \( f \) in StyleGAN to generate the reconstruction result.

We named our method CLCAE (i.e., StyleGAN inversion with Contrastive Learning and Cross-Attention Encoder). We show that our CLCAE can achieve state-of-the-art performance in both reconstruction quality and editing capacity on benchmark datasets containing human portraits and cars. Fig. 1 shows some results. This indicates the robustness of our CLCAE. Our contributions are summarized as follows:

- We propose a novel contrastive learning approach to align the image space and foundation latent space \( W \) of StyleGAN. This alignment ensures that we can obtain proper latent code \( w \) during GAN inversion.
- We propose a cross-attention encoder to transform latent codes in \( W \) into \( W^+ \) and \( F \). The representation of latent code in \( W^+ \) and feature in \( F \) are improved to benefit reconstruction fidelity and editability.
- Experiments indicate that our CLCAE achieves state-of-the-art fidelity and editability results both qualitatively and quantitatively.

2. Related Work

2.1. GAN Inversion

GAN inversion [70] is the task to find a latent code in a latent space of pretrained-GAN’s domain for the real image. As mentioned in the GAN inversion survey [62], the inversion methods can be divided into three groups: optimization-based, encoder-based, and hybrid. The optimization-based methods [1, 2, 7, 11, 11, 19, 64, 71] try to directly optimize the latent code or the parameters of GAN [53] to minimize the distance between the reconstruction image. The encoder-based methods [5, 10, 20, 25, 28, 33, 44, 50, 52, 56] learn a mapper to transfer the image to the latent code. The hybrid methods [69,70] combine these two methods.

**StyleGAN Inversion.** Our work belongs to the StyleGAN inversion framework. Typically, there are three embedding spaces (i.e., \( W \) [30], \( W^+ \) [1], and \( F \) [28]) and they are the trade-off design between the distortion and editability. The \( W \) is the foundation latent space of StyleGAN, several works [56,71] have shown inverting the image into this space produces a high degree of editability but unsatisfied reconstruction quality. Differently, the \( W^+ \) is developed from \( W \) to reduce distortions while suffering less editing flexibility. On the other hand, the \( F \) space consists of specific features in StyleGAN, and these features are generated by the latent input code of foundation latent space \( W \) in the StyleGAN training domain. The \( F \) space contains the highest reconstruction ability, but it suffers the worst editability. Different from these designs that directly explore \( W^+ \) and \( F \), we step back to explore \( W \) and use it to guide \( W^+ \) and \( F \) to improve fidelity and editability.
2.2. Latent Space Editing

Exploring latent space’s semantic directions improves editing flexibility. Typically, there are two groups of methods to find meaningful semantic directions for latent space based editing: supervised and unsupervised methods. The supervised methods [3, 13, 17, 54] need attribute classifiers or labeled data for specific attributes. InterfaceGAN [54] use annotated images to train a binary Support Vector Machine [45] for each label and interprets the normal vectors of the obtained hyperplanes as manipulation direction. The unsupervised methods [21, 55, 58, 65] do not need the labels. GanSpace [21] find directions use Principal Component Analysis (PCA). Moreover, some methods [24, 49, 60, 72] use the CLIP loss [51] to achieve amazing text guiding image manipulation. And some methods use the GAN-based pipeline to edit or inpaint the images [37–41]. In this paper, we follow the [56] and use the InterfaceGAN and GanSpace to find the semantic direction and evaluate the manipulation performance.

2.3. Contrastive Learning

Contrastive learning [8, 15, 16, 18, 22, 47] has shown effective in self-supervised learning. When processing multi-modality data (i.e., text and images), CLIP [51] provides a novel paradigm to align text and image features via contrastive learning pretraining. This cross-modality feature alignment motivates generation methods [32, 49, 60, 61, 72] to edit images with text attributes. In this paper, we are inspired by the CLIP and align the foundation latent space \( \mathcal{W} \) and the image space with contrastive learning. Then, we set the contrastive learning framework as a loss function to help us find the suitable latent code in \( \mathcal{W} \) for the real image during GAN inversion.

3. Method

Fig. 3 shows an overview of the proposed method. Our CNN encoder is from pSp [52] that is the prevalent encoder in GAN inversion. Given an input image \( I \), we obtain latent code \( w \) in foundation latent space \( \mathcal{W} \in \mathbb{R}^{512} \). This space is aligned to the image space via contrastive learning. Then we set the latent code \( w \) as a query to obtain the latent code \( w^+ \in \mathcal{W}^+ \in \mathbb{R}^{256 \times 512} \) space via \( \mathcal{W}^+ \) cross-attention block. The size of \( N \) is related to the size of the generated image (i.e., \( N = 18 \) when the size of the generated image is \( 1024 \times 1024 \)). Meanwhile, we select the top feature in the encoder as \( f \in \mathcal{F} \in \mathbb{R}^{256 \times 256 \times C} \) space and use \( w \) to refine \( f \) with \( \mathcal{F} \) cross-attention block. Finally, we send \( w^+ \) and \( f \) to the pretrained StyleGAN pipeline to produce the reconstruction results.

![Figure 2. The process of contrastive learning pre-training. The encoders and projection heads extract the embedding of the image and latent code. Then we make the paired embeddings similar to align the image and latent code distribution. After alignment, we fix the parameters in the contrastive learning module to enable the latent code to fit the image during inversion.](image)

3.1. Aligning Images and Latent Codes

We use contrastive learning from CLIP to align image \( I \) and its latent code \( w \). After pre-training, we fix this module and use it as a loss function to measure the image and latent code similarity. This loss is set to train the CNN encoder in Fig. 3 as to align one image \( I \) and its latent code \( w \).

The contrastive learning module is shown in Fig. 2. We synthesize 100K \((I)\) and latent code \((w)\) pairs with pre-trained StyleGAN. The \( I \) and \( w \) are fed into the module where there are feature extractors (i.e., CNN for \( I \) and transformer for \( w \)) and projection heads. Specifically, our minibatch contains \( S \) image and latent code pairs \((I \in \mathbb{R}^{256 \times 256 \times 3}, w \in \mathbb{R}^{512}\)) in \( \mathcal{W} \). We denote their embeddings after projection heads (i.e., hidden state) as \( h_f(I) \in \mathbb{R}^{512} \) and \( h_w(w) \in \mathbb{R}^{512} \), respectively. For the \( i \)-th pair from one minibatch (i.e., \( i \in [1, 2, \ldots, S] \)), its embeddings are \( h_f(I_i) \) and \( h_w(w_i) \). The contrastive loss [46, 68] can be written as

\[
\mathcal{L}_i^{(I \rightarrow w)} = -\log \frac{\exp \left[ \langle h_f(I_i), h_w(w_i) \rangle / t \right]}{\sum_{k=1}^{S} \exp \left[ \langle h_f(I_i), h_w(w_k) \rangle / t \right]},
\]

\[
\mathcal{L}_i^{(w \rightarrow I)} = -\log \frac{\exp \left[ \langle h_w(w_i), h_f(I_i) \rangle / t \right]}{\sum_{k=1}^{S} \exp \left[ \langle h_w(w_i), h_f(I_k) \rangle / t \right]},
\]

where \( \langle \cdot \rangle \) denotes the cosine similarity, and \( t \in \mathbb{R}^{+} \) is a learnable temperature parameter. The alignment loss in the contrastive learning module can be written as

\[
\mathcal{L}_{\text{align}} = \frac{1}{S} \sum_{i=1}^{S} \left( \lambda \mathcal{L}_i^{(I \rightarrow w)} + (1 - \lambda) \mathcal{L}_i^{(w \rightarrow I)} \right),
\]

where \( \lambda = 0.5 \). We use the CNN in pSp [52] as the image encoder, and StyleTransformer [25] as the latent code
encoder. Then in the GAN inversion process, we fix the parameters in the contrastive learning module and compute $L_{align}$ to enable the latent code to fit the image. Aligning images to their latent codes directly via supervision $L_{align}$ enforces our foundation latent space $W^+$ close to the image space to avoid reconstruction distortions.

### 3.2. Cross-Attention Encoder

Once we have pre-trained the contrastive learning module, we make it frozen to provide the image and latent code matching loss. This loss function is utilized for training the CNN encoder in our CLCAE framework shown in Fig. 3. Our CNN encoder is a pyramid structure for hierarchical feature generations (i.e., $T_1, T_2, T_3$). We use $T_1$ to generate latent code $w$ via a map2style block. Both the CNN encoder and the map2style block are from pSp [52]. After obtaining $w$, we can use $I$ and $w$ to produce an alignment loss via Eq. 3. This loss will further update the CNN encoder for image and latent code alignment. Also, we use $w$ to discover $w^+$ and feature $f$ with the cross-attention blocks.

#### 3.2.1 $W^+$ Cross-Attention Block

As shown in Fig. 3, we set the output of $W^+$ cross-attention block as the residual of $w$ to predict $w^+$. Specifically, we can get the coarse residual $\Delta w^+ \in \mathbb{R}^{N \times 512}$ with the CNN’s features and map2style blocks first. Then we send each vector $\Delta w^+_i \in \mathbb{R}^{512}$ in $\Delta w^+$ and $w \in \mathbb{R}^{512}$ to the $W^+$ cross-attention block to predict the better $\Delta w^+_i$, where $i = 1, ..., N$. In the cross-attention block, we set the $w$ as query($Q$) and $\Delta w^+_i$ as value($V$) and key($K$) to calculate the attention map. This attention map can extract the potential relation between the $w$ and $\Delta w^+_i$, and it can make the $w^+$ close to the $w$. Specifically, the $Q$, $K$, and $V$ are all projected from $\Delta w^+_i$ and $w$ with learnable projection heads, and we add the output of cross-attention with $w$ to get final latent code $w^+_i$ in $W^+$, the whole process can be written as

$$Q = wW^+_Q, K = \Delta w^+_iW^+_K, V = \Delta w^+_iW^+_V,$$

$$\text{Attention}(Q, K, V) = \text{Softmax} \left( \frac{QK^T}{\sqrt{d}} \right) V,$$

$$w^+_i = w + \text{Attention}(Q, K, V),$$

where $W^+_Q$, $W^+_K$, $W^+_V \in \mathbb{R}^{512 \times 512}$ and the feature dimension $d$ is 512. We use the multi-head mechanism [57] in our cross-attention. The cross-attention can make the $w^+$ close to the $w$ to preserve the great editability. Meanwhile, the reconstruction performance can still be preserved, since we get the refined $w$ via the $L_{align}$.

#### 3.2.2 $F$ Cross-Attention Block

The rich and correct spatial information can improve the representation ability of $f$ as mentioned in [48]. We use the $T_3 \in \mathbb{R}^{512 \times 64 \times 64}$ as our basic feature to predict $f$ as
shown in Fig. 3, since the $T_3$ has the richest spatial information in the pyramid CNN. Then we calculate cross attention between the $w$ and $T_3$ and output a residual to refine the $T_3$. In contrast to the $W^+$ cross-attention block, we set the $w$ as value($V$) and key($K$) and $T_3$ as query($Q$), this is because we want to explore the spatial information of $w$ to support the $T_3$. Finally, we use a CNN to reduce the spatial size of the cross-attention block’s output to get the final prediction $f$, the shape of $f$ matches the feature of the selected convolution layer in $F$ space. We choose the 5th convolution layer following the FS [63]. The whole process can be written as:

$$Q = T_3W^X, K = wW^X, V = wW^Y,$$

$$\text{Attention}(Q, K, V) = \text{Softmax} \left( \frac{QK^T}{\sqrt{d}} \right),$$

$$f = \text{CNN} \left[ \text{Attention}(Q, K, V) + T_3 \right],$$

where $W^X, W^Y \in R^{512 \times 512}$ and the feature dimension $d$ is 512. Finally, we send the $w^+$ to the pretrained StyleGAN ($G$) via AdaIN [26] and replace the selected feature in $G$ with $f$ to get the final reconstruction result $G(w^+, f)$.

**Image Editing.** During the editing process, we need to get the modified $\hat{w}^+$ and $\hat{f}$. For the $\hat{w}^+$, we obtain it with the classic latent space editing methods [21, 54]. For the $\hat{f}$, we follow the FS [63] to generate the reconstruction result $G(w^+)$ and the edited image $G(w^+)$ respectively. Then we extract the feature of the 5th convolution layer of $G(w^+)$ and $G(w^+)$ respectively. Finally, we calculate the difference between these two features and add it to the $f$ to predict the $\hat{f}$. The whole process to get the $\hat{f}$ is:

$$\hat{f} = f + G^5(\hat{w}) - G^5(w),$$

where the $G^5(\hat{w})$ and $G^5(w)$ is the feature of 5-th convolution layer. With the modified $\hat{w}^+$ and $\hat{f}$, we can get the editing results $G(\hat{w}^+, \hat{f})$.

### 3.3. Loss Functions

To train our encoder, we use the common ID and reconstruction losses to optimize the three reconstruction results $I_1^{\text{rec}} = G(w)$, $I_2^{\text{rec}} = G(w^+)$ and $I_3^{\text{rec}} = G(w^+, f)$ simultaneously. Meanwhile, we use the feature regularization to make the $f$ close to the original feature in $G$ similar to the FS [63].

**Reconstruction losses.** We utilize the pixel-wise $\mathcal{L}_2$ loss and $\mathcal{L}_{\text{LPIPS}}$ [67] to measure the pixel-level and perceptual-level similarity between the input image and reconstruction image as

$$\mathcal{L}_{\text{rec}} = \sum_{i=1}^{3} \left( \lambda_{\text{LPIPS}} \mathcal{L}_{\text{LPIPS}} (I, I^{\text{rec}}_i) + \lambda_2 \mathcal{L}_2 (I, I^{\text{rec}}_i) \right),$$

(7)

where the $\mathcal{L}_{\text{LPIPS}}$ and $\mathcal{L}_2$ are are weights balancing each loss. We set the $\mathcal{L}_{\text{LPIPS}} = 0.2$ and $\mathcal{L}_2 = 1$ during training.

**ID loss.** We follow the e4e [56] to use the identity loss to preserve the identity of the reconstructed image as

$$\mathcal{L}_{\text{id}} = \sum_{i=1}^{3} \left( 1 - \langle R(I), R(I^{\text{rec}}_i) \rangle \right).$$

(8)

For the human portrait dataset, the $R$ is a pretrained ArcFace facial recognition network [27]. For the cars dataset, the $R$ is a ResNet-50 [23] network trained with MOCOv2 [9].

**Feature regularization.** To edit the $f$ with Eq. 6, we need to ensure $f$ is similar to the original feature of $G$. So we adopt a regularization for the $f$ as

$$\mathcal{L}_{\text{fog}} = \| f - G^5(w^+) \|_2^2,$$

(9)

**Total losses.** In addition to the above losses, we add the $\mathcal{L}_{\text{align}}$ to help us find the proper $w$. In summary, the total loss function is defined as:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{ID}} \mathcal{L}_{\text{ID}} + \lambda_{\text{fog}} \mathcal{L}_{\text{fog}} + \lambda_{\text{align}} \mathcal{L}_{\text{align}},$$

(10)

where $\lambda_{\text{rec}}, \lambda_{\text{ID}}, \lambda_{\text{fog}}$ and $\lambda_{\text{align}}$ are the weights that adjust the contribution of each loss term. And we set the $\lambda_{\text{rec}} = 1, \lambda_{\text{ID}} = 0.1, \lambda_{\text{fog}} = 0.01$ and $\lambda_{\text{align}} = 1$ respectively by default.

### 4. Experiments

In this section, we first illustrate our implementation details. Then we compare our method with existing methods qualitatively and quantitatively. Finally, an ablation study validates the effectiveness of our contributions. More results are provided in the supplementary files. We will release our implementations to the public.

#### 4.1. Implementation Details

During the contrastive learning process, we follow the CLIP [51] and use the Adam optimizer [34] to train the image and latent code encoders. We synthesize the image-latent code pair dataset with the pre-trained StyleGAN2 in cars and human portrait domains. We set the batch size to 256 for training. During the StyleGAN inversion process, we train and evaluate our method on cars and human portrait datasets. For the human portrait, we use the FFHQ [30] dataset for training and the CelebA-HQ test set [43] for evaluation. For cars, we use the Stanford Cars [35] dataset.
for training and testing. We set the resolution of the input image as 256 × 256. We follow the pSp [52] and use the Ranger optimizer to train our encoder for GAN inversion, the Ranger optimizer is a combination of Rectified Adam [42] with the Lookahead technique [66]. We set the batch size to 32 during training. We use 8 Nvidia Tesla V100 GPUs to train our model.

4.2. Qualitative Evaluation

Our CLCAE improves the representation ability of the latent code in $W^+$ and feature in $F$ spaces. We evaluate qualitatively how our latent codes $w^+$ and $f$ improve the output result. To clearly compare these two latent codes, we split the evaluation methods into two groups. The first group consists of methods only using latent code $w^+$, we denote this group as ‘group $W^+$’. The second group consists of methods using both $w^+$ and $f$, we denote this group as ‘group $F$’. When comparing to the group $W^+$, we use our results CLCAE$_{w^+}$ computed via $G(w^+)$ for fair comparisons. When comparing to the group $F$, we use our results computed via $G(w^+, f)$. During image editing, we use InterfaceGAN [54] and GanSpace [21] to find the semantic direction and manipulate the face and car images, respectively.

$W^+$ space. Fig. 4 shows the visual results where our CLCAE$_{w^+}$ is compared to e4e [56], pSp [52], restyle$_{pSp}$ [5], restyle$_{e4e}$ [5] and StyleTransformer (ST) [25]. Both our CLCAE$_{w^+}$ and e4e show better inversion performance in the human portrait. This phenomenon is caused by the over-fitting of those methods in (b)~(e), since the $W^+$ space pays more attention to the quality of the reconstruction. The CLCAE$_{w^+}$ and e4e can produce $w^+$ close to the $w$, which improves the robustness of these two methods. Moreover, our CLCAE$_{w^+}$ is more capable of avoiding distortions while maintaining editability than other methods, including e4e (see the second row). This is because our $w^+$ is based on the solid $w$ that does not damage the reconstruction performance of $w^+$. For the domain of cars, we observe that pSp and restyle$_{pSp}$ are limited to represent editing ability (see the (b) and (e) of the viewpoint row). On the other hand, e4e and ST are able to edit images, but their reconstruction performance are unsatisfying. In contrast to these methods, our CLCAE$_{w^+}$ maintains high fidelity and flexible editability at the same time.

$F$ space. Fig. 5 shows our comparisons to PTI [53], Hyper [6], HFGI [59], and FS [63] in the $F$ space. The results of PTI, Hyper, HFGI, and FS contain noticeable distortion in the face (e.g., the eyes in the red box regions in (a)~(d)) and the car (e.g., the background in (a)~(c) and the red box regions in car images). Although FS [63] reconstructs the background of the car image well, it loses editing flexibility (e.g., see (d) of 4 rows). This is because the FS method relies too much on $F$ space, which limits the editability. In contrast, our results are in high fidelity as well as a wide range of editability with powerful $f$ and $w^+$.
4.3. Quantitative Evaluation

**Inversion.** We perform a quantitative comparison in the CelebA-HQ dataset to evaluate the inversion performance. We apply the commonly-used metric: PSNR, SSIM, LPIPS [67] and ID [27]. Table 1 shows these evaluation results. The PTI in $\mathcal{F}$ group and Restyle$_{pSp}$ in $\mathcal{W}^+$ group have better performance than our method in ID and LPIPS metric, respectively. But these two method takes a lot of time for the optimization operation or the iterative process. With the simple and effective cross-attention encoder and the proper foundation latent code, our method can achieve good performance in less time.

**Editing.** There is hardly a straight quantitative measurement to evaluate editing performance. We use the InterFaceGAN [54] to find the manipulation direction and edit the image, then we calculate the ID distance [27] between the original image and the manipulation one. For a fair comparison, during the ID distance evaluation, we use the
our method to predict better while \(\downarrow\) indicates lower is better while \(\uparrow\) indicates higher is better.

### 4.4. Ablation Study

**Effect of contrastive learning.** We compare the optimization method [31] to calculate the distance between the latent code \(w\) and the image domain. Meanwhile, we present the results generated by \(w\) without \(L_{\text{align}}\) in (b) to prove our contrastive learning validity further. The associated numerical results are shown in Table 2.

**Effect of the \(\mathcal{W}^+\) Cross-Attention.** To validate the effectiveness of \(\mathcal{W}^+\) cross-attention block, we remove it and use the coarse residual as \(w^+\) directly to do a comparison experiment. As shown in Fig. 6, the experiment results in (d) have distortion (see the eyes regions of the first row and the hair regions of the second row). And the cross-attention block in (e) can improve performance. This is because the cross-attention block utilizes the solid latent code \(w\) to support our method to predict better \(w^+\). The numerical analysis results are shown in Table 2.

**Effect of the \(\mathcal{F}\) Cross-Attention.** We analyze the effect of \(\mathcal{F}\) cross-attention block by comparing the results produced with it and without it. We can see the visual comparison in Fig. 6. The results in (f) show that our method has artifacts in the hair and eye regions of the face without the \(\mathcal{F}\) cross-attention block. And our method with \(\mathcal{F}\) cross-attention block demonstrates better detail (see the hair and eyes in (g)). This phenomenon can prove that the \(\mathcal{F}\) cross-attention block can extract the valid information in \(w\) and refine the \(f\), which also tells us the importance of a good foundation. The numerical evaluation in Table 2 also indicates that \(\mathcal{F}\) cross-attention block improves the quality of reconstructed content.

### 5. Conclusion and Future Work

we propose a novel GAN inversion method CLCAE that revisits the StyleGAN inversion and editing from the foundation space \(\mathcal{W}\) viewpoint. CLCAE adopts a contrastive learning pre-training to align the image space and latent code space first. And we formulate the pre-training process as a loss function \(L_{\text{align}}\) to optimize latent code \(w\) in \(\mathcal{W}\) space during inversion. Finally, CLCAE sets the \(w\) as the foundation to obtain the proper \(w^+\) and \(f\) with proposed cross-attention blocks. Experiments on human portrait and car datasets prove that our method can simultaneously produce powerful \(w\), \(w^+\), and \(f\). In the future, we will try to expand this contrastive pre-training process to other domains (e.g., Imagenet dataset [12]) and do some basic downstream tasks such as classification and segmentation. This attempt could bring a new perspective to contrastive learning.

<table>
<thead>
<tr>
<th>Group</th>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>LPIPS</th>
<th>Time</th>
<th>ID</th>
<th>User Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inversion</td>
<td>PSNR</td>
<td>19.08</td>
<td>0.53</td>
<td>0.20</td>
<td>0.029s</td>
<td>0.44</td>
<td>70%</td>
</tr>
<tr>
<td>PSNR</td>
<td>20.39</td>
<td>0.56</td>
<td>0.16</td>
<td>0.028s</td>
<td>0.52</td>
<td>60%</td>
<td>-</td>
</tr>
<tr>
<td>TS</td>
<td>25.90</td>
<td>0.57</td>
<td>0.19</td>
<td>0.026s</td>
<td>0.53</td>
<td>62%</td>
<td>-</td>
</tr>
<tr>
<td>restyle</td>
<td>19.45</td>
<td>0.50</td>
<td>0.50</td>
<td>1.154s</td>
<td>0.47</td>
<td>84%</td>
<td>-</td>
</tr>
<tr>
<td>Att</td>
<td>21.20</td>
<td>0.65</td>
<td>0.679</td>
<td>1.158s</td>
<td>0.64</td>
<td>73%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE</td>
<td>21.23</td>
<td>0.65</td>
<td>0.62</td>
<td>0.071s</td>
<td>0.62</td>
<td>74%</td>
<td>-</td>
</tr>
<tr>
<td>Optimization [31]</td>
<td>21.23</td>
<td>0.65</td>
<td>0.62</td>
<td>0.071s</td>
<td>0.62</td>
<td>74%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/o (L_{\text{align}})</td>
<td>20.61</td>
<td>0.57</td>
<td>0.20</td>
<td>0.071s</td>
<td>0.54</td>
<td>60%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/o (L_{\text{align}}) (w^+) Att</td>
<td>19.36</td>
<td>0.50</td>
<td>0.09</td>
<td>0.071s</td>
<td>0.12</td>
<td>96%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/ (L_{\text{align}}) Att</td>
<td>21.23</td>
<td>0.65</td>
<td>0.06</td>
<td>0.071s</td>
<td>0.10</td>
<td>96%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/ (L_{\text{align}}) (w^+) Att</td>
<td>21.23</td>
<td>0.65</td>
<td>0.06</td>
<td>0.071s</td>
<td>0.10</td>
<td>96%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/ (L_{\text{align}}) Att (w^+)</td>
<td>24.50</td>
<td>0.74</td>
<td>0.09</td>
<td>0.074s</td>
<td>0.12</td>
<td>96%</td>
<td>-</td>
</tr>
<tr>
<td>CLCAE w/ (L_{\text{align}}) Att (w^+)</td>
<td>24.50</td>
<td>0.74</td>
<td>0.09</td>
<td>0.074s</td>
<td>0.12</td>
<td>96%</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1. Quantitative comparisons of state-of-the-art methods on the CelebA-HQ dataset. We conduct a user study to measure the editing performance. The number denotes the preference rate of our method against the competing methods. Chance is 50%. \(\downarrow\) indicates lower is better while \(\uparrow\) indicates higher is better.

Table 2. Quantitative ablation study on the CelebA-HQ dataset. \(\downarrow\) indicates lower is better while \(\uparrow\) indicates higher is better.
References


[56] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation, 2021.


[71] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Improved stylegan embedding: Where are the good latents?, 2020.