

GH-Feat: Learning Versatile Generative Hierarchical Features from GANs

Yinghao Xu, Yujun Shen, Jiapeng Zhu, Ceyuan Yang, and Bolei Zhou, *Member, IEEE*

Abstract—Recent years witness the tremendous success of generative adversarial networks (GANs) in synthesizing photo-realistic images. GAN generator learns to compose realistic images and reproduce the real data distribution. Through that, a hierarchical visual feature with multi-level semantics spontaneously emerges. In this work we investigate that such a generative feature learned from image synthesis exhibits great potentials in solving a wide range of computer vision tasks, including both generative ones and more importantly discriminative ones. We first train an encoder by considering the pre-trained StyleGAN generator as a learned loss function. The visual features produced by our encoder, termed as *Generative Hierarchical Features (GH-Feat)*, highly align with the layer-wise GAN representations, and hence describe the input image adequately from the reconstruction perspective. Extensive experiments support the versatile transferability of GH-Feat across a range of applications, such as image editing, image processing, image harmonization, face verification, landmark detection, layout prediction, image retrieval, *etc.* We further show that, through a proper spatial expansion, our developed GH-Feat can also facilitate fine-grained semantic segmentation using only a few annotations. Both qualitative and quantitative results demonstrate the appealing performance of GH-Feat. Code and models are available at <https://genforce.github.io/ghfeat/>.

Index Terms—Generative adversarial network, generative representation, feature learning, image editing.

1 INTRODUCTION

REPRESENTATION learning plays an essential role in the rise of deep learning. The learned representations are able to express the variation factors of the complex visual world. Accordingly, the performance of a deep learning algorithm largely depends on the features extracted from the input data. As pointed out by Bengio *et al.* [1], a good representation is expected to have the following properties. First, it should capture multiple configurations of the input. Second, it should organize the explanatory factors of the input data as a hierarchy, where more abstract concepts are at a higher level. Third, it should have strong transferability, not only from datasets to datasets but also from tasks to tasks.

Deep neural networks supervisedly trained for image classification on large-scale datasets (e.g., ImageNet [2] and Places [3]) have resulted in expressive and discriminative visual features [4]. However, the developed features are heavily dependent on the training objective. For example, the deep features learned for the object recognition task may give more attention to the shapes and parts of the objects while remain invariant to rotation [5], [6], and the deep features from a scene classification model may focus more on detecting the categorical objects (e.g., bed for bedroom and sofa for living room) [7]. Thus the discriminative features learned from solving high-level image classification tasks might not be necessarily good for other mid-level and low-level tasks, limiting their transferability [8], [9]. Besides,

it remains unknown how the discriminative features can be used in generative applications like image editing.

In contrast to discriminative models, generative models such as generative adversarial networks (GANs) [10] offer an alternative way for representation learning. A GAN is typically formulated to match the synthetic data distribution to the observed data distribution. Through adversarial training, the generator in GAN is required to capture the multi-level variation factors underlying the input data to the most extent, otherwise, the discrepancy between the real and synthesized data would be spotted by the discriminator. Recent studies have confirmed that the StyleGAN family [11], [12] spontaneously encodes rich semantics in a hierarchical manner [13], [14]. But the transferability of the per-layer representation learned by GANs is not fully verified in the literature. Some attempts have been made to apply the generative representations (*i.e.*, the representations emerging from solving generative tasks) to the high-level image classification task [15], [16], [17], yet leaving mid-level and low-level tasks less explored.

In this work, we make a thorough investigation into the utility of GAN representations and demonstrate their wide applications to downstream tasks, including both generative ones and more importantly discriminative ones. To appropriately obtain the GAN-derived features from a given image, we train a hierarchical encoder using the pre-trained StyleGAN generator as a learned loss function. Through that way, our encoder works together with the generator to reconstruct the image, implicitly required to extract the variation factors that describe the input. We call the output visual features as *Generative Hierarchical Features (GH-Feat)*. We also observe in the encoder training that, only exploiting the supervision at the image level (*i.e.*, the pixel reconstruction loss) may cause the overfitting of pixel

- Y. Xu and C. Yang are with the Department of Information Engineering, the Chinese University of Hong Kong, Hong Kong SAR.
- Y. Shen is with Ant Research, China.
- J. Zhu is with the School of Computer Science and Engineering, the Hong Kong University of Science and Technology, Hong Kong SAR.
- B. Zhou is with Computer Science Department, University of California, Los Angeles, USA.

values, severely limiting the transferability of the extracted representations. To mitigate such a negative effect, we introduce a training regularizer from the statistical perspective. It alleviates the problem of distribution mismatching between the learned GH-Feat and the native GAN representations.

After the encoder is well prepared, we evaluate the resulting GH-Feat on a broad range of downstream applications. On one hand, we verify that the representations learned by GANs naturally support various generative tasks. Concretely, to edit or process target images, we can simply modulate their corresponding features, and reuse the generator (*i.e.*, the same as the one used in encoder training) as a renderer to decode features back to images. The extensive experiments show that our approach achieves global and local image manipulation, transferring styles between two images, image colorization, image inpainting, image super-resolution, *etc.* Besides, our GH-Feat also allows fusing objects from an image into another as the application of image harmonization. On the other hand, we are interested in the discriminative capability of the generative features. For this purpose, we treat the features extracted by our encoder as the base representations, on top of which we learn different linear task heads for a range of high-level and middle-level tasks. Experiments on multiple datasets validate the effectiveness of GH-Feat on large-scale image classification, face verification, facial landmark detection, room layout prediction, transferring learning, image retrieval, *etc.* Furthermore, to enable dense prediction tasks, we manage to expand GH-Feat along spatial dimensions with an adequate modification of our encoder. The improved spatial-aware visual features suggest compelling performance on fine-grained semantic segmentation using only a few annotations.

The preliminary result of this work is published at [18] as oral presentation. We include the following new contents as the extension to the conference paper. (1) We find the limitation of only using image-level supervision for encoder training, and provide an effective solution by introducing a distribution-level regularizer. Analyses and improvements are illustrated in Sec. 4.2.2. (2) We include two more image editing tasks, *i.e.* style transfer (Sec. 4.3.4) and semantic manipulation (Sec. 4.3.5), to validate that our GH-Feat can describe images moderately, aligning with human perception. (3) We confirm in Sec. 4.3.6 that GH-Feat also facilitates conventional image processing tasks, including image colorization, image inpainting, and image super-resolution. (4) We include image retrieval as an additional discriminative task to verify the hierarchical property of GH-Feat, whose details are explained in Sec. 4.4.5. (5) We propose a spatial expansion to our GH-Feat via learning a spatial-aware encoder, and show the great potential of the improved representations in data-efficient fine-grained semantic segmentation in Sec. 4.5.

2 RELATED WORK

Visual Features. Visual Feature plays a fundamental role in the computer vision field. Traditional methods used manually designed features [19], [20], [21] for pattern matching and object detection. These features are significantly improved by deep models [22], [23], [24], which

automatically learn the feature extraction from large-scale datasets. However, the features supervisedly learned for a particular task could be biased to the training task and hence become difficult to transfer to other tasks, especially when the target task is too far away from the base task [8], [9]. Unsupervised representation learning is widely explored to learn a more general and transferable feature [25], [26], [27], [28], [29], [30], [31], [32], [33], [34]. However, most of existing unsupervised feature learning methods focus on evaluating their features on the tasks of image recognition, yet seldom evaluate them on other mid-level or low-level tasks, let alone generative tasks. Shocher *et al.* [35] discover the potential of discriminative features in image generation, but the transferability of such features are not fully verified. **Generative Adversarial Networks.** GANs [10] are able to produce photo-realistic images via learning the underlying data distribution. The recent advance of GANs [36], [37], [38] has significantly improved the synthesis quality. StyleGAN [11] proposes a style-based generator with multi-level style codes and achieves the start-of-the-art generation performance. However, little work explores the representation learned by GANs as well as how to apply such representation for other applications. Some recent work interprets the semantics encoded in the internal representation of GANs and applies them for image editing [13], [14], [39], [40], [41], [42]. But it remains much less explored whether the learned GAN representations are transferable to discriminative tasks.

Adversarial Representation Learning. The main reason of hindering GANs from being applied to discriminative tasks comes from the lack of inference ability. To fill this gap, prior work introduces an additional encoder to the GAN structure [15], [16]. Donahue and Simonyan [17] and Pidhorskyi *et al.* [43] extend this idea to the state-of-the-art BigGAN [38] and StyleGAN [11] models respectively. In this paper, we also study the representation learning using GANs, with following **improvements** compared to existing methods. First, we propose to treat the well-trained StyleGAN generator as a *learned loss function*. Second, instead of mapping the images to the initial GAN latent space, like most algorithms [15], [16], [17], [43] have done, we design a novel encoder to produce *hierarchical* features that well align with the layer-wise representation learned by StyleGAN. Third, besides the image classification task that is mainly targeted at by prior work [15], [16], [17], [43], we validate the *transferability* of our proposed GH-Feat on a range of generative and discriminative tasks, demonstrating its generalization ability.

3 METHODOLOGY

This section introduces the encoder used to extract hierarchical visual features from the input images. This encoder is trained in an unsupervised manner using a well-prepared StyleGAN generator. Sec. 3.1 describes how we abstract the multi-level representation from StyleGAN. Sec. 3.2 presents the structure of the novel hierarchical encoder. Sec. 3.4 describes the idea of using pre-trained StyleGAN generator as a learned loss function for representation learning. Sec. 3.3 introduces the training regularizer to prevent the encoder from overfitting pixel values.

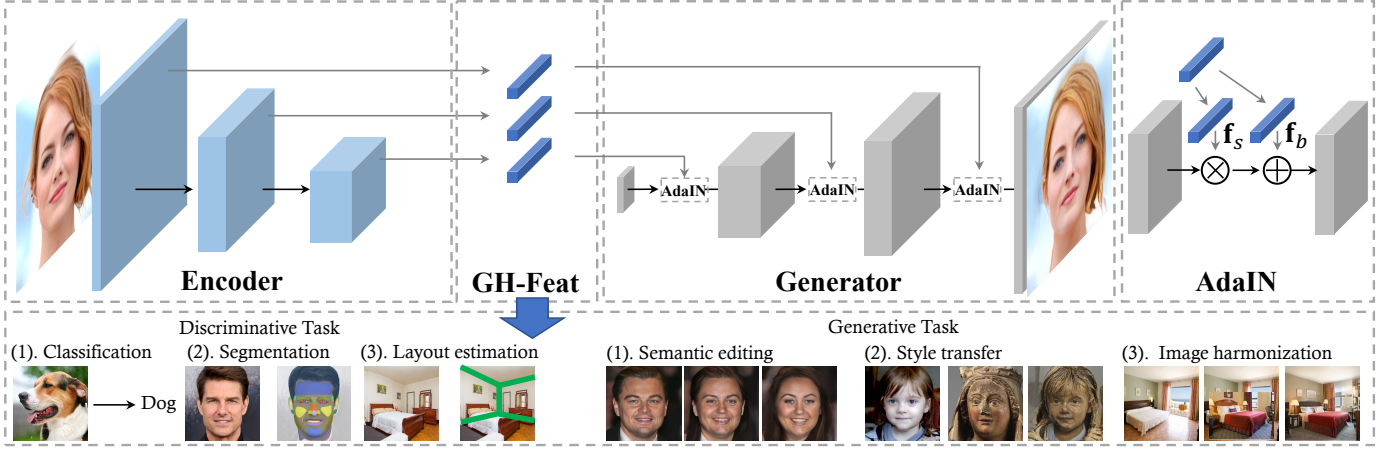


Fig. 1. **Framework** of the GH-Feat. This feature hierarchy highly aligns with the layer-wise representation (i.e., style codes of per-layer AdaIN) learned by the StyleGAN generator. Parameters in blue blocks are learnable while others are frozen.

TABLE 1

Encoder structure, which is based on ResNet-50 [24]. Fully-connected (FC) layers are employed to map the feature maps produced by the Spatial Alignment Module (SAM) to our proposed Generative Hierarchical Features (GH-Feat). GH-Feat exactly align with the multi-scale style codes used in StyleGAN [11]. The numbers in brackets indicate the dimension of features at each level.

Stage	Encoder Pathway	Output Size	SAM & Pool	FC Dimension	GH-Feat	Style Code in StyleGAN
input	—	3×256^2				
conv ₁	$7 \times 7, 64$ stride 2, 2	64×128^2				
pool ₁	$3 \times 3, \max$ stride 2, 2	64×64^2				
res ₂	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	256×64^2				
res ₃	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	512×32^2	512×4^2	8192×1792	Level 1-2 Level 3-4 Level 5-6	Layer 14-13 ($128d \times 2$) Layer 12-11 ($256d \times 2$) Layer 10-9 ($512d \times 2$)
res ₄	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	1024×16^2	512×4^2	8192×4096	Level 7-8 Level 9-10	Layer 8-7 ($1024d \times 2$) Layer 6-5 ($1024d \times 2$)
res ₅	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	2048×8^2	512×4^2	8192×4096	Level 11-12 Level 13-14	Layer 4-3 ($1024d \times 2$) Layer 2-1 ($1024d \times 2$)

3.1 Layer-wise Representation from StyleGAN

The generator $G(\cdot)$ of GANs typically takes a latent code $\mathbf{z} \in \mathcal{Z}$ as the input and is trained to synthesize a photo-realistic image $\mathbf{x} = G(\mathbf{z})$. The recent state-of-the-art StyleGAN [11] proposes to first map \mathbf{z} to a disentangled space \mathcal{W} with $\mathbf{w} = f(\mathbf{z})$. Here, $f(\cdot)$ denotes the mapping implemented by multi-layer perceptron (MLP). The \mathbf{w} code is then projected to layer-wise style codes $\{\mathbf{y}^{(\ell)}\}_{\ell=1}^L \triangleq \{(\mathbf{y}_s^{(\ell)}, \mathbf{y}_b^{(\ell)})\}_{\ell=1}^L$ with affine transformations, where L is the number of convolutional layers. $\mathbf{y}_s^{(\ell)}$ and $\mathbf{y}_b^{(\ell)}$ correspond to the channel-wise scale and weight parameters in Adaptive Instance Normalization (AdaIN) [44]. The space constructed by these layer-wise style parameters is named as \mathcal{Y} space. These style codes are used to modulate the output feature maps of each convolutional layer with

$$\text{AdaIN}(\mathbf{x}_i^{(\ell)}, \mathbf{y}^{(\ell)}) = \mathbf{y}_{s,i}^{(\ell)} \frac{\mathbf{x}_i^{(\ell)} - \mu(\mathbf{x}_i^{(\ell)})}{\sigma(\mathbf{x}_i^{(\ell)})} + \mathbf{y}_{b,i}^{(\ell)}, \quad (1)$$

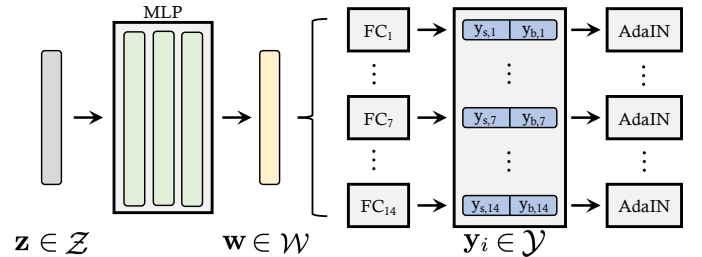


Fig. 2. Multiple latent spaces of StyleGAN. FC refers to the affine layer between \mathcal{W} space and \mathcal{Y} space.

where $\mathbf{x}_i^{(\ell)}$ indicates the i -th channel of the output feature map from the ℓ -th layer. $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and variance respectively. Fig. 2 illustrates the \mathcal{Z} , \mathcal{W} and \mathcal{Y} space of StyleGAN.

Here, we treat the layer-wise style codes of \mathcal{Y} space, $\{\mathbf{y}^{(\ell)}\}_{\ell=1}^L$, as the generative visual features that we would like to extract from the input image. There are two

major advantages. First, the synthesized image can be completely determined by these style codes without any other variations, making them suitable to express the information contained in the input data from the generative perspective. Second, these style codes are organized as a hierarchy where codes at different layers correspond to semantics at different levels [11], [14]. To the best of our knowledge, this is the first work that adopts the style codes for the per-layer AdaIN module as the learned representations of StyleGAN. Wu *et al.* [45] also shows \mathcal{Y} space can be leveraged for disentangled control of image editing, while our work explores the potential of generative representations in facilitating both generative and more importantly discriminative downstream tasks.

3.2 Hierarchical Encoder

Based on the layer-wise representation described in Sec. 3.1, we propose a novel encoder $E(\cdot)$ with a hierarchical structure to extract multi-level visual features from a given image. As shown in Fig. 1, the encoder is designed to best align with the StyleGAN generator. In particular, the Generative Hierarchical Features (GH-Feat) produced by the encoder, $\{\mathbf{f}^{(\ell)}\}_{\ell=1}^L \triangleq \{(\mathbf{f}_s^{(\ell)}, \mathbf{f}_b^{(\ell)})\}_{\ell=1}^L$, are fed into the per-layer AdaIN module of the generator by replacing the style code $\mathbf{y}^{(L-\ell+1)}$ in Eq. (1).

We adopt ResNet [24] architecture as the encoder backbone and add an extra residual block to get an additional feature map with lower resolution. In fact, there are totally six stages in our encoder, where the first one is a convolutional layer (followed by a pooling layer) and each of the others consists of several residual blocks. Besides, we introduce a feature pyramid network [46] to learn the features from multiple levels. The output feature maps from the last three stages, $\{R_4, R_5, R_6\}$, are used to produce GH-Feat. Taking a 14-layer StyleGAN generator as an instance, R_4 aligns with layer 9-14, R_5 with 5-8, while R_6 with 1-4. Here, to bridge the feature map with each style code, we first downsample it to 4×4 resolution and then map it to a vector of the target dimension using a fully-connect (FC) layer. In addition, we introduce a lightweight Spatial Alignment Module (SAM) [47], [48] into the encoder structure to better capture the spatial information from the input image. SAM works in a simple yet efficient way:

$$R_i = W_i \text{down}(R_i) + W_6 R_6 \quad i \in \{4, 5\},$$

where W_4 , W_5 , and W_6 (all are implemented with an 1×1 convolutional layer) are used to project the feature maps R_4 , R_5 , and R_6 to have the same number of feature channels respectively. R_4 and R_5 are downsampled to the same resolution of R_6 before fusion.

Encoder Structure. Tab. 1 provides the detailed architecture of our hierarchical encoder by taking a 14-layer StyleGAN [11] generator as an instance. Recall that the design of GH-Feat treats the layer-wise style codes used in the StyleGAN model (*i.e.*, the code fed into the AdaIN module [44]) as generative features. Accordingly, GH-Feat consists of 14 levels that exactly align with the multi-scale style codes yet in a reverse order, as shown in the last two columns of Tab. 1.

3.3 Statistical Training Regularizer

As discussed in Sec. 3.1, our approach aims at learning the style representations encoded in \mathbf{y} , which are transformed from the \mathbf{w} code using pre-layer linear projection. \mathcal{Y} space is less constrained than \mathcal{W} space and hence may suffer from the problem of overfitting pixel values, which further leads to poor transferability of the learned features. To solve such a problem, we infer $\{\mathbf{y}_{avg}^{(\ell)}\}_{\ell=1}^L$ from the averaged latent code (*i.e.*, a statistics from the training stage), \mathbf{w}_{avg} , and propose to only learn the residual code at each layer. Thus, we have $E(\mathbf{x}) = \{\Delta \mathbf{y}^{(\ell)}\}_{\ell=1}^L$, which induces the final features as $\{\mathbf{y}_{avg}^{(\ell)} + \Delta \mathbf{y}^{(\ell)}\}_{\ell=1}^L$. We then penalize the l_2 norm of each residual code to prevent it from shifting too far from the native distribution, resulting in a training regularizer

$$\mathcal{L}_{reg} = \sum_{\ell=1}^L \|\Delta \mathbf{y}^{(\ell)}\|_2^2. \quad (2)$$

e4e [49] also regularizes the inversion space when training the encoder yet from a different aspect against GH-Feat. In particular, the regularization in e4e [49] targets minimizing the latent code variation across layers (*i.e.*, they expect the inverted codes regarding different layers to be close to each other) to reconstruct the input image from coarse to fine. Differently, the regularization in our work bonds the latent code close to the distribution center (*i.e.*, the statistical average) to prevent the model from overfitting pixel values. In this way, our approach could better represent an image from the semantic level, further facilitating downstream tasks.

3.4 StyleGAN Generator as Learned Loss

We consider the pre-trained StyleGAN generator as a leaned loss function. Specifically, we employ a StyleGAN generator to supervise the encoder training with the objective of image reconstruction. We also introduce a discriminator to compete with the encoder, following the formulation of GANs [10], to ensure the reconstruction quality. To summarize, the encoder $E(\cdot)$ and the discriminator $D(\cdot)$ are jointly trained with

$$\min_{\Theta_E} \mathcal{L}_E = \|\mathbf{x} - G(E(\mathbf{x}))\|_2 - \lambda_1 \mathbb{E}_{\mathbf{x}}[D(G(E(\mathbf{x})))] + \lambda_2 \|F(\mathbf{x}) - F(G(E(\mathbf{x})))\|_2 + \lambda_3 \mathcal{L}_{reg}, \quad (3)$$

$$\min_{\Theta_D} \mathcal{L}_D = \mathbb{E}_{\mathbf{x}}[D(G(E(\mathbf{x})))] - \mathbb{E}_{\mathbf{x}}[D(\mathbf{x})] + \lambda_4 \mathbb{E}_{\mathbf{x}}[\|\nabla_{\mathbf{x}} D(\mathbf{x})\|_2^2], \quad (4)$$

where $\|\cdot\|_2$ denotes the l_2 norm and $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are loss weights to balance different loss terms. The last term in Eq. (3) represents the perceptual loss [50] and $F(\cdot)$ denotes the conv4_3 output from a pre-trained VGG [23] model.

4 EXPERIMENTS

We evaluate Generative Hierarchical Features (GH-Feat) on a wide range of downstream applications. Sec. 4.1 introduces the experimental settings, such as implementation details, datasets, and tasks. Sec. 4.2 presents the analysis of our approach including ablation study and the importance of the regularizer. Sec. 4.3 and Sec. 4.4 evaluate the applicability of GH-Feat on generative and

discriminative tasks respectively. Sec. 4.5 shows the results of the introduced spatial expansion.

4.1 Experimental Settings

Implementation Details. The loss weights are set as $\lambda_1 = 0.1$, $\lambda_2 = 5e^{-5}$, $\lambda_3 = 5e^{-4}$ and $\lambda_4 = 5$. We use Adam [51] optimizer, with $\beta_1 = 0$ and $\beta_2 = 0.99$, to train both the encoder and the discriminator. The learning rate is initially set as $1e^{-4}$ and exponentially decayed with the factor of 0.8.

Datasets and Models. We conduct experiments on four StyleGAN [11] models, pre-trained on MNIST [52], FF-HQ [11], LSUN bedrooms [53], and ImageNet [2] respectively. The MNIST model is with 32×32 resolution and the remaining models are with 256×256 resolution.

Generative Tasks. (1) *Image editing*. It focuses on manipulating the image content or style, e.g., global editing, local editing. (2) *Image harmonization*. This task harmonizes a discontinuous image to produce a realistic output. (3) *Style transfer*. This task focuses on transferring the style of the reference image to the source image. (4) *Semantic manipulation*. It targets at modifying the semantic meaning of an object while preserving other characteristics. (5) *Image colorization*. It focuses on colorizing the grayscale image. (6) *Image inpainting*. This task reconstructs missing regions in an image. (7) *Image super-resolution*. It aims at improving the resolution of the image.

Discriminative Tasks. (1) *MNIST digit recognition*. It is a long-standing image classification task. We report the Top-1 accuracy on the test set following [52]. (2) *Face verification*. It aims at distinguishing whether the given pair of faces come from the same identity. We validate on the LFW dataset [54] following the standard protocol [54]. (3) *ImageNet classification*. This is a large-scale image classification dataset [2], consisting of over 1M training samples across 1,000 classes and 50K validation samples. We use Top-1 accuracy as the evaluation metric following existing work [15], [17]. (4) *Pose estimation*. This task targets at estimating the yaw pose of the input face. 70K real faces on FF-HQ [11] are split into 60K training samples and 10K test samples. The ℓ_1 regression error is used as the evaluation metric. (5) *Landmark detection*. This task learns a set of semantic points with visual meaning. We use FF-HQ [11] dataset and follow the standard MSE metric [55] to report performances in inter-ocular distance (IOD). (6) *Layout prediction*. We extract the corner points of the layout line and convert the task to a landmark regression task. The annotations of the collected 90K bedroom images (70K for training and 20K for validation) are obtained with [56]. Following [57], we report the corner distance as the metric. (7) *Face luminance regression*. It focuses on regressing the luminance of face images. We use it as a low-level task on the FF-HQ [11] dataset. (8) *Image retrieval*. It aims at retrieving the images with specific attributes. (9) *Data-efficient image segmentation*. This task focuses on predicting the class of each spatial pixel with limited annotated data.

4.2 Analysis on GH-Feat

4.2.1 Ablation Study

We make ablation studies on the training of encoder from two perspectives. (1) We choose the layer-wise style codes

TABLE 2
Ablation studies on the feature space and the SAM module.

Space	SAM	Reg	MSE↓	SSIM↑	FID↓
\mathcal{W}	✓		0.0601	0.540	22.24
\mathcal{Y}			0.0502	0.550	19.06
\mathcal{Y}	✓		0.0464	0.558	18.48
\mathcal{Y}	✓	✓	0.0494	0.551	16.84



Fig. 3. Qualitative comparison on image reconstruction between training the generator from scratch together with the encoder, and our GH-Feat that treats the well-learned StyleGAN generator as a loss function.

y over the w codes as the representation from StyleGAN. (2) We introduce Spatial Alignment Module (SAM) into the encoder to better handle the spatial information. (3) We involve a regularizer in the training of encoder.

Since the encoder is trained with the objective of image reconstruction, we use mean square error (MSE), SSIM [58], and FID [59] to evaluate the encoder performance. Tab. 2 shows the results where we can tell that our encoder benefits from the effective SAM module and that choosing an adequate representation space (i.e., the comparison between the first row and the last row) results in a better reconstruction. Introducing the regularizer alleviates the pixel value overfitting and improves the reconstruction quality at the distribution level. More discussion on the differences between \mathcal{W} space and \mathcal{Y} space can be found in Sec. 4.4.1.

Random Generator. Recall that, during the training of the encoder, we propose to treat the well-trained StyleGAN generator as a learned loss function. In this part, we explore what will happen if we train the generator from scratch together with the encoder. Fig. 3 and Tab. 3 show the qualitative and quantitative results respectively, which demonstrate the strong performance of GH-Feat. It suggests that besides higher efficiency, reusing the knowledge from a well-trained generator can also bring better performance.

4.2.2 Importance of Regularizer

Although GH-Feat has achieved good results in image reconstruction, it cannot perform very well on image editing. Compared with the \mathcal{W} space that previous attempts adopt as the inversion space, the \mathcal{Y} space used by GH-Feat ignores the linear transformation between w code and y code, resulting in its flexibility. Hence, it is easier to overfit a given image through a simple combination of generative features. This leads to a mismatch between the

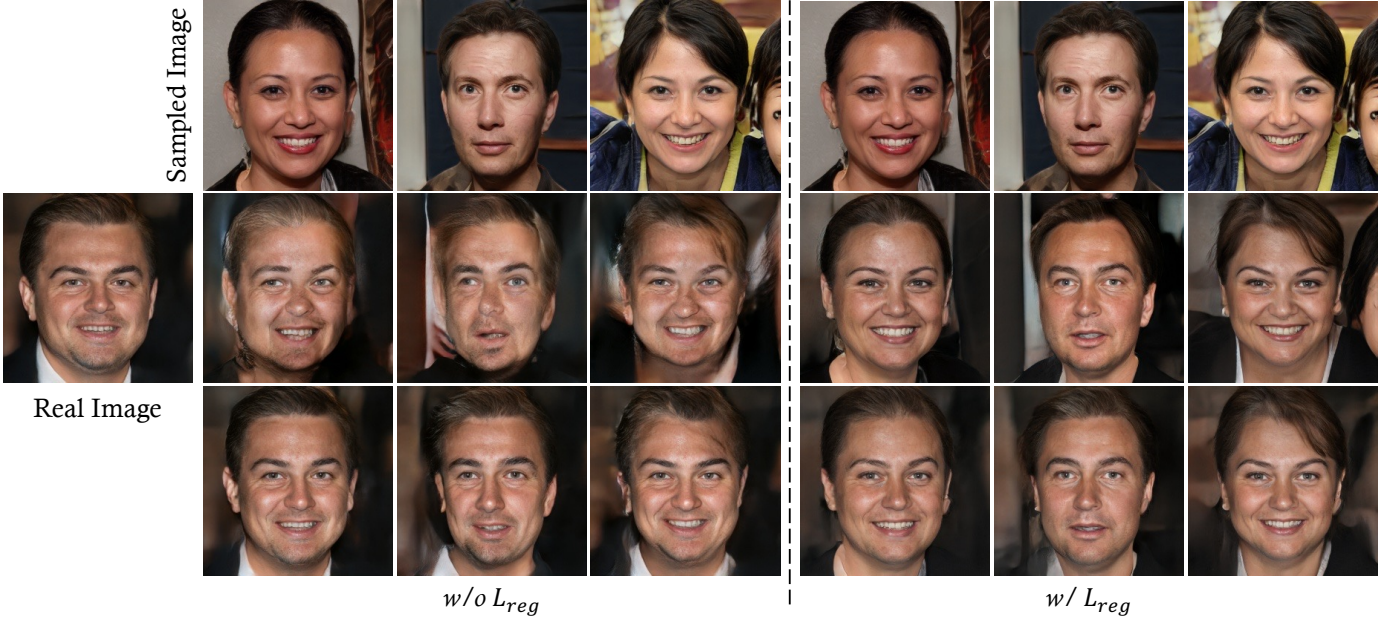


Fig. 4. Qualitative comparison on the style mixing task between using training regularizer or not. The first row are the sampled images. The second row shows the results mixed with the codes predicted by our encoder. The third row presents the mixing results with original latent code.

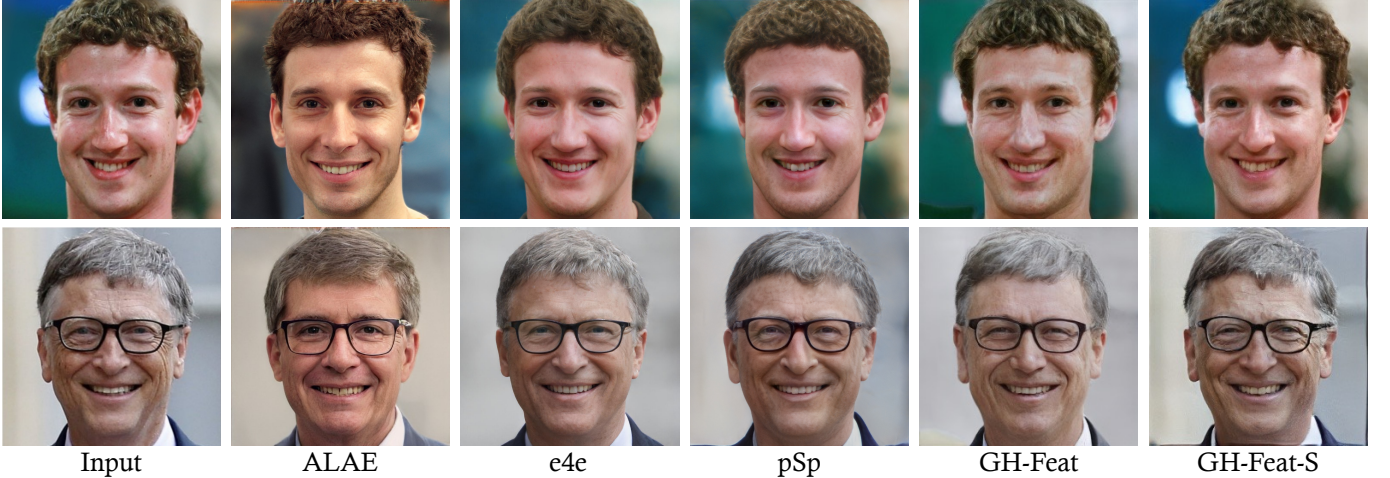


Fig. 5. Qualitative comparison on reconstructing real images. GH-Feat-S denotes the spatial GH-Feat. Our GH-Feat and GH-Feat-S, which are built on StyleGAN, could get comparable and better performance as pSp [60] and e4e [49], which employ a more powerful StyleGAN2 generator.

TABLE 3

Quantitative comparison on image reconstruction between training the generator from scratch together with the encoder, and our GH-Feat that treats the well-learned StyleGAN generator as a loss function. GH-Feat-R denotes GH-Feat trained with regularizer.

	MSE↓	SSIM↑	FID↓
Training $G(\cdot)$ from Scratch	0.429	0.301	46.20
GH-Feat (Ours)	0.0464	0.558	18.48
GH-Feat-R	0.0494	0.551	16.84

TABLE 4

Cosine similarity of the encoder output and the native latent code.

	$w/o \mathcal{L}_{reg}$	w/ \mathcal{L}_{reg}
FED	0.444	0.879

learned visual features and the latent space distribution of the generator.

We choose the task of global editing as a benchmark to explore the mismatch problem. Specifically, we extract

the generative features of the real image first and then randomly replace them with the randomly sampled features in the latent space at layers 0-4 to achieve global editing. The results are shown in the 2nd row of Fig. 4. Besides, we also extract the visual feature of the sampled images and do the same operation to achieve the editing result in the third line of Fig. 4. Obviously, the mixed results by the two sets of features both extracted by the encoder are better, suggesting the domain shift between the visual features and the latent space of the generator. Based on this, we apply the constraints proposed in Sec. 3 to the encoder training. The right part of Fig. 4 shows the editing results with the \mathcal{L}_{reg} . The global editing results with sampled and extracted features are very similar, and both are much better than the result without \mathcal{L}_{reg} . It demonstrates that the generative features learned with \mathcal{L}_{reg} are more in line with its distribution of latent space.

To quantitatively measure the similarity between two domains, we use cosine distance between generative feature

and native code. Specifically, we sample 10k fake images and extract the corresponding GH-Feat by our encoder, and then cosine similarity is calculated for the two distributions. As shown in Tab. 4, minimizing the variation of the generative features can improve the similarity from 0.444 to 0.879, suggesting the effectiveness of this regularization.

4.3 Evaluation on Generative Tasks

Thanks to using the StyleGAN as a learned loss function, a huge advantage of GH-Feat over existing unsupervised feature learning approaches [29], [30], [31], [32], [34], which mainly focus on the image classification task, is its generative capability. In this section, we conduct a number of generative experiments to verify this point.

4.3.1 Image Reconstruction

Image reconstruction is an important evaluation on whether the learned features can best represent the input image. MSE and SSIM [58] are used as quantitative metrics to evaluate the reconstruction performance. Tab. 5 and Fig. 5 show the quantitative and qualitative comparison between our GH-Feat and other GAN inversion methods on FF-HQ faces [11] and LSUN bedrooms [53]. The very recent work ALAE [43] also employs StyleGAN for representation learning. We have following differences from ALAE: (1) We use the \mathcal{V} space instead of the \mathcal{W} space of StyleGAN as the representation space. (2) We learn *hierarchical* features that highly align with the per-layer style codes in StyleGAN. (3) Our encoder can be *efficiently* trained with a well-learned generator by treating StyleGAN as a loss function. We can tell that GH-Feat better reconstructs the input by preserving more information, resulting a more expressiveness representation.

Besides pSp [60], e4e [49] and Restyle [62], we include the results of PTI [61] as well as the improved version of our GH-Feat (*i.e.*, spatial expansion introduced in Sec. 4.5). We also include the inference time to help evaluate the model efficiency. We have three observations from the table below. (1) Our GH-Feat, which is built on StyleGAN, could get comparable performance as pSp [60] and e4e [49], which employ a more powerful StyleGAN2 generator. We surmise that such an advantage originates from the replacement from \mathcal{W} space to \mathcal{V} space. (2) Restyle [62] (which requires iterative refinement) and PTI [61] (which requires tuning of the weights of the generator) provide good reconstruction results but suffer from slow inference speed. (3) Our improved version, *i.e.*, Spatial GH-Feat, substantially improves the inversion quality without sacrificing the model efficiency, and achieves the best performance among all encoder-based methods without generator tuning.

4.3.2 Image Editing

In this part, we evaluate GH-Feat on a number of image editing tasks. Different from the features learned from discriminative tasks [24], [31], our GH-Feat naturally supports sampling and enables creating new data.

Style Mixing. To achieve style mixing, we use the encoder to extract visual features from both the content image and the style image and swap these two features at some particular level. The swapped features are then visualized by the

generator, as shown in Fig. 6. We can observe the compelling hierarchical property of the learned GH-Feat. For example, by exchanging low-level features, only the image color tone and the skin color are changed. Meanwhile, mid-level features controls the expression, age, or even hair styles. Finally, high-level features correspond to the face shape and pose information (last two columns).

Global Editing. The style mixing results have suggested the potential of GH-Feat in multi-level image stylization. Sometime, however, we may not have a target style image to use as the reference. Thanks to the design of the latent space in GANs [10], the generative representation naturally supports sampling, resulting in a strong creativity. In other words, based on GH-Feat, we can arbitrarily sample meaningful visual features and use them for image editing. Fig. 7 presents some high-fidelity editing results at multiple levels. This benefits from the matching between the learned GH-Feat and the internal representation of StyleGAN.

Local Editing. Besides global editing, our GH-Feat also facilitates editing the target image locally by deeply cooperating with the generator. In particular, instead of directly swapping features, we can exchange a certain region of the spatial feature map at some certain level. In this way, only a local patch in the output image will be modified while other parts remain untouched. As shown in Fig. 8, we can successfully manipulate the input face with different eyes, noses, and mouths.

4.3.3 Image Harmonization

Our hierarchical encoder is robust such that it can extract reasonable visual features even from discontinuous image content. We copy the patches from other images onto the original image and feed the stitched image into our proposed encoder for feature extraction. The extracted features are then visualized via the pre-trained generator, as in Fig. 9. On the bedroom, we can see that the copied bed, window and ceiling light well blend into the “background”. We also surprisingly find that when copying a window into the source image, the view from the original window and that from the new window highly align with each other (*e.g.*, vegetation or ocean). On face image, besides eye, nose and mouth, GH-Feat also blends the glasses with the background very well, benefiting from the robust generative visual features.

4.3.4 Style Transfer

Our GH-Feat can not only edit the image attributes by replacing the randomly sampling feature at a particular level but also can facilitate the editing with the given conditional input. Here, we take style transfer as an example, aiming to transfer the style of the given image to the source image. We first extract the generative features of the content image I_c and style image I_s , and then style-mixing is performed by replacing the visual features of I_c with the corresponding ones of I_s at the layer 8-16. We leverage the disentanglement of the generative features across different layers to perform style transfer. As shown in Fig. 10, our encoder can successfully transfer the style of the given image to the source images, suggesting the effectiveness of the generative features. It is worth noting that although the texture of the given style images

TABLE 5

Quantitative comparison on reconstructing images from FF-HQ faces [11] and LSUN bedrooms [53]. GH-Feat-S denotes the spatial GH-Feat. **bold** ones rank the best among the methods w/o generator tuning and underlined ones are the second.

Method	Face			Bedroom	
	MSE↓	SSIM↑	TIME↓	MSE↓	SSIM↑
<i>w/ generator tuning</i>					
PTI [61]	0.009	0.74	58.02	-	-
<i>w/o generator tuning</i>					
ALAE [43]	0.182	0.40	0.023	0.275	0.32
pSp [60]	0.034	0.56	0.063	-	-
e4e [49]	0.052	0.50	0.063	-	-
Restyle [62]	0.030	0.66	0.304	-	-
GH-Feat	0.046	0.56	0.035	0.068	0.52
GH-Feat-S	0.029	0.67	0.038	0.057	0.581

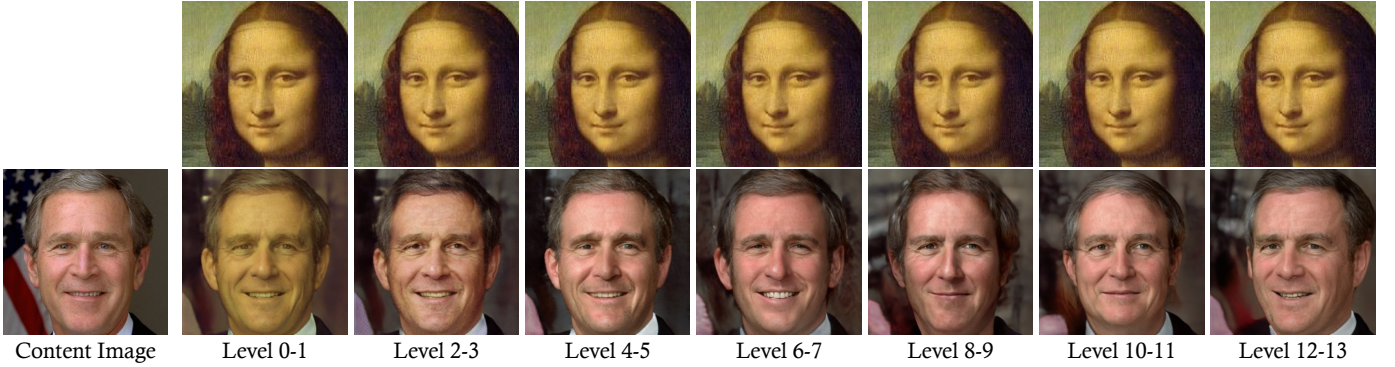


Fig. 6. **Style mixing** results by exchanging the GH-Feat extracted from the content image and the style image (first row) at different levels. Higher level corresponds to the high-level semantics like the face shape and pose information.



Fig. 7. **Global image editing** achieved by GH-Feat. On the left is the input image, while the others are generated by randomly sampling the visual feature at some particular level.



Fig. 8. **Local image editing** achieved by GH-Feat. On the left is the input image, while the others are generated by randomly sampling the visual feature and replacing the spatial feature map (for different regions) at some particular level. Zoom in for details.

rarely appears in the training dataset, our encoder can still reconstruct it and extract reasonable visual features with good disentangle properties. It also supports the robustness and generalization of the visual features extracted by our hierarchical encoder.

4.3.5 Semantic Manipulation

Here we explore the semantic editability of the generative features. We utilize off-the-shelf semantic directions from InterFaceGAN [13] to edit the inversion results. Fig. 11 presents the results of the manipulated faces. Obviously, the learned generative features can preserve most other details when manipulating a particular facial attribute. These

editing results demonstrate that generative features can not only reconstruct the given image in high quality, but also facilitate it with good semantic manipulation properties.

4.3.6 Image Processing

In this section, we demonstrate that our method facilitates various image processing tasks such as image colorization, image inpainting, and image super-resolution by utilizing the prior knowledge learned by GANs. Generally, these tasks can be formulated as follows:

$$s^* = \arg \min_{s \in S} L(G(s), x). \quad (5)$$

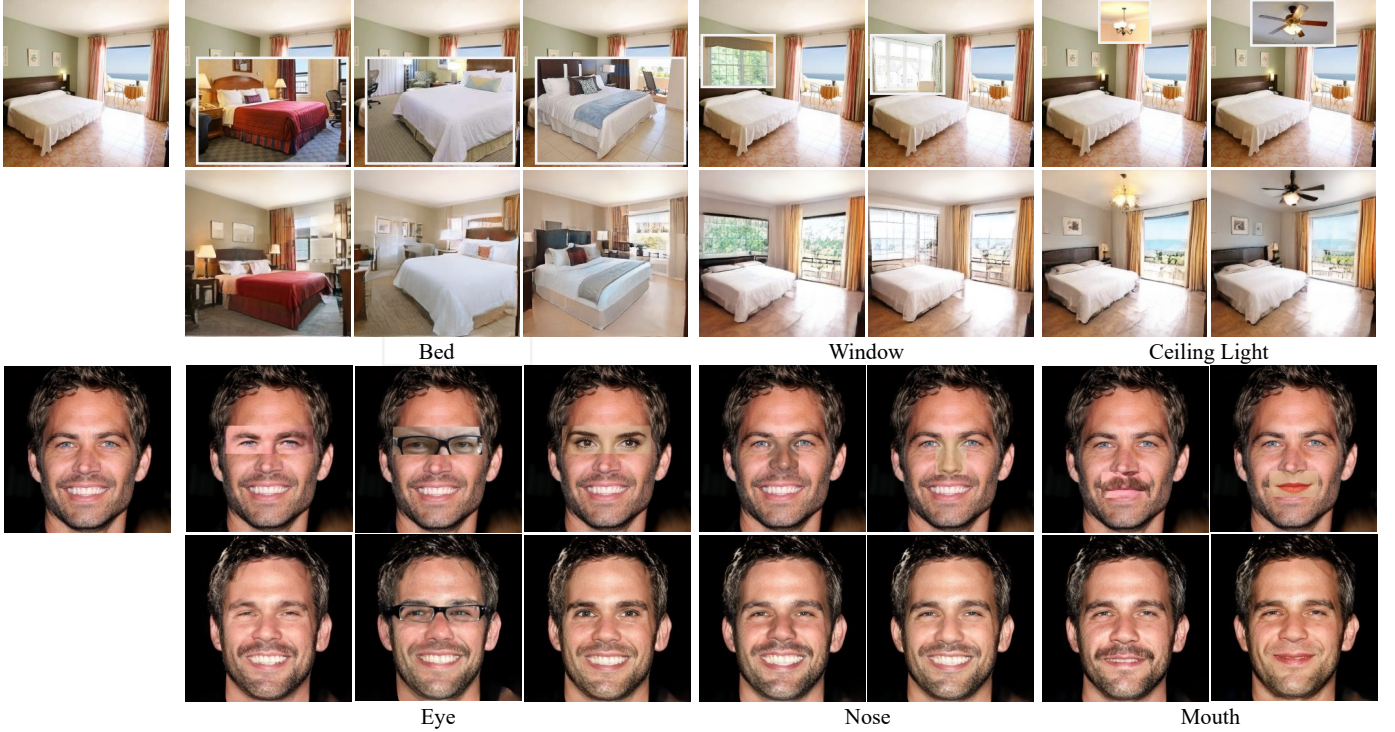


Fig. 9. **Image harmonization** on bedroom and face with GH-Feat. The top left corner of the first and third rows are the original images. Pasting a target image patch onto the original image then feeding it as the input (first and third row), our hierarchical encoder is able to smooth the image content and produce a photo-realistic image (second and fourth row).

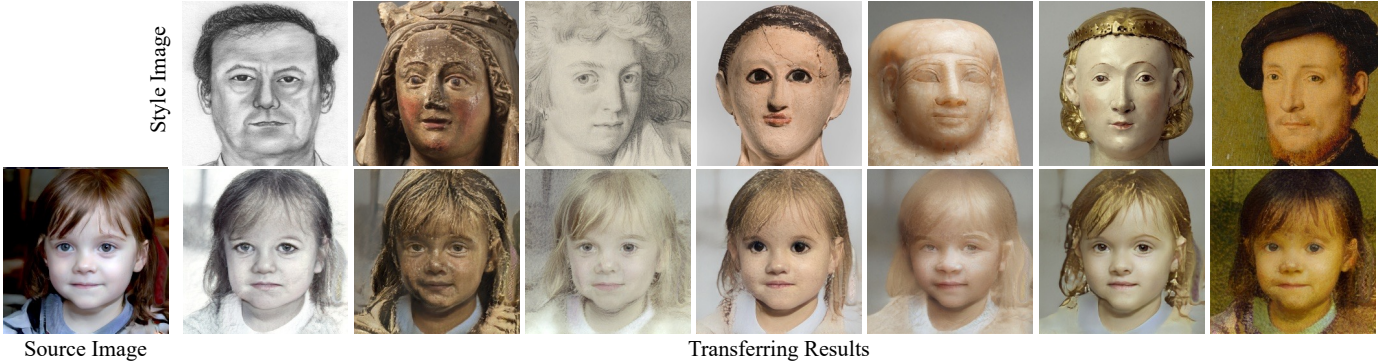


Fig. 10. **Style transfer** results with GH-Feat. GH-Feat can extract and then transfer the style of the reference image to the given image.

where s is the style code initialized by our encoder, L is the l2 loss function, and x is the reference image (e.g., gray-scale image for image colorization, corrupted image for the inpainting, and low-resolution image for super-resolution).

Image colorization tries to restore the original color of a gray-scale image. The results from our method are listed in Fig. 12a. Image inpainting aims at filling the missing pixels of the input images. As shown in Fig. 12b, when some pixels value of the input image is missing, our method still successfully recovers them. The last one is super-resolution, which manages to generate a high-resolution image of the low-resolution one. Fig. 12c shows the super-resolution result scale 16 times using our method.

4.4 Evaluation on Discriminative Tasks

In this part, we verify that even the proposed GH-Feat is learned from generative models, it can be applicable to a wide range of discriminative tasks with competitive performances. Here, we do not fine-tune the encoder for

any certain task. In particular, we choose multi-level downstream applications, including image classification, face verification, pose estimation, layout prediction, landmark detection, and luminance regression. For each task, we use our encoder to extract visual features from both the training and the test set. A linear regression model (*i.e.*, a fully-connected layer) is learned on the training set with ground-truth and then evaluated on the test set. Besides, we include image retrieval as an addition discriminative task to verify the hierarchical property of GH-Feat, whose details are explained in Sec. 4.4.5.

4.4.1 Discriminative and Hierarchical Property

Recall that GH-Feat is a multi-scale representation learned by using StyleGAN as a loss function. As a results, it consists of features from multiple levels, each of which correspond to a certain layer in the StyleGAN generator. Here, we would to explore how this feature hierarchy is organized as well as how they can facilitate multi-level

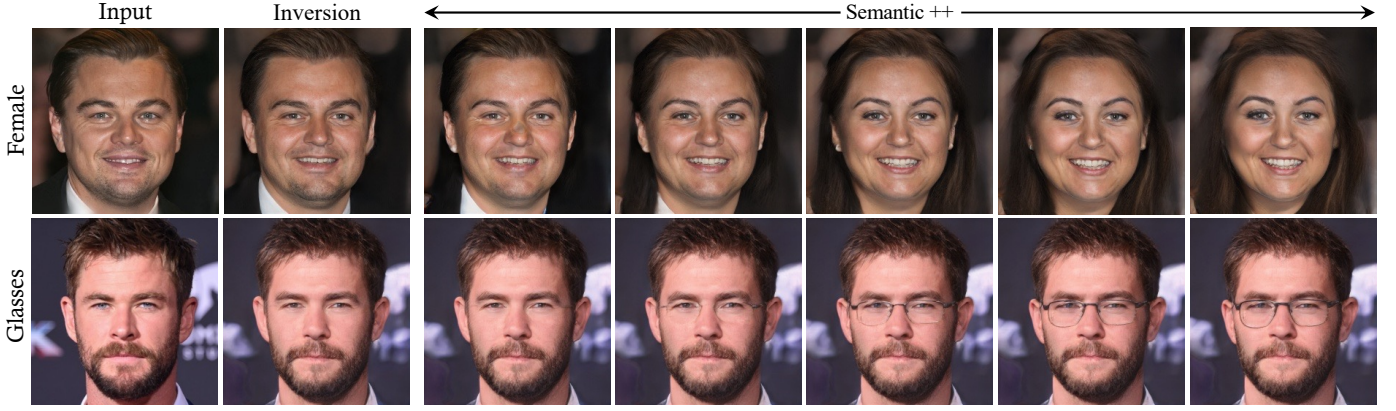
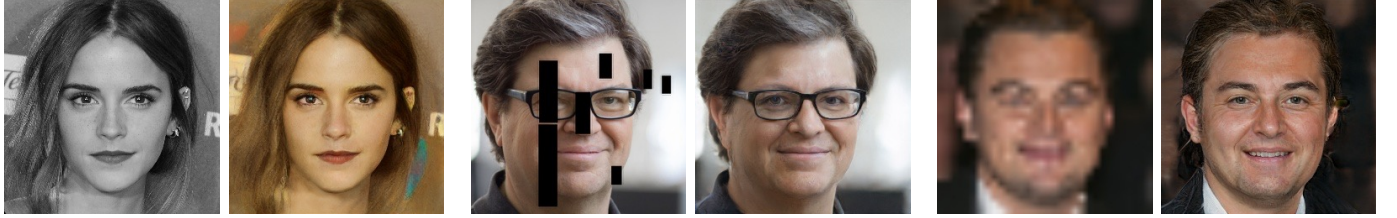


Fig. 11. **Semantic Manipulation** results with GH-Feat. We utilize the off-the-shelf semantic directions from InterFaceGAN [13] to edit the gender and glasses of the given images.



(a) Colorization

(b) Inpainting

(c) Super-resolution

Fig. 12. **Image processing** with GH-Feat. GH-Feat facilitates many image processing applications using the hierarchical encoder.

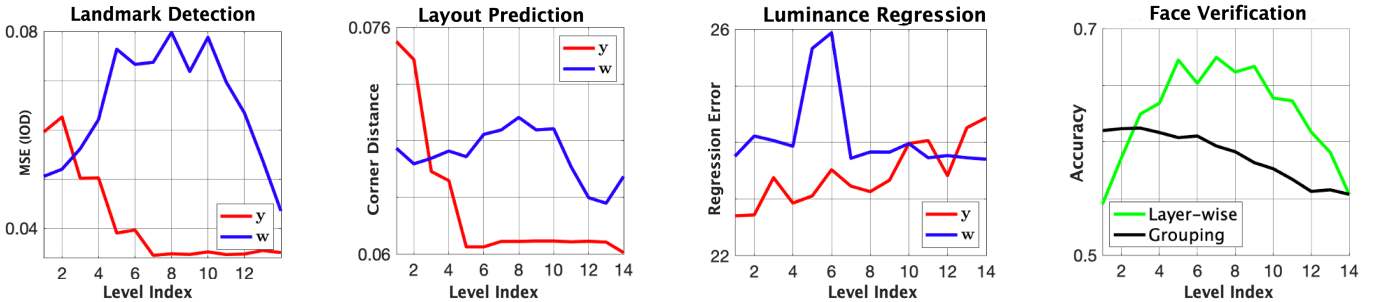


Fig. 13. Performances on different discriminative tasks using GH-Feat. Left three columns enclose the comparisons between using different spaces of StyleGAN as the representation space, where \mathcal{Y} space (in red color) shows stronger discriminative and hierarchical property than \mathcal{W} space (in blue color). This is discussed in Sec. 4.4.1. The last column compares the two different strategies used in the face verification task, which is explained in Sec. 4.4.2.

discriminative tasks, including face pose estimation, indoor scene layout prediction, and luminance¹ regression from face images. In particular, we evaluate GH-Feat on each task level by level. As a comparison, we also train encoders by treating the w code, instead of the style code y , as the representation. From Fig. 13, we have three observations: (1) GH-Feat is discriminative. (2) Features at lower level are more suitable for low-level tasks (e.g., luminance regression) and those at higher level better aid high-level tasks (e.g., pose estimation). (3) \mathcal{Y} space demonstrates a more obvious hierarchical property than \mathcal{W} space. The comparison on hierarchical property between using regularizer or not is included at **Supplementary Material**.

4.4.2 Digit Recognition & Face Verification

Image classification is widely used to evaluate the performance of learned representations [17], [29], [30], [31], [32]. In this section, we first compare our proposed GH-Feat with

1. We convert images from RGB space to YUV space and use the mean value from Y space as the luminance.

other alternatives on a toy dataset, i.e., MNIST [52]. Then, we use a more challenging task, i.e., face verification, to evaluate the discriminative property of GH-Feat.

MNIST Digit Recognition. We first show a toy example on MNIST following prior work [15], [43]. We make a little modification to ResNet-18 like [63] which is widely used in literatures to handle samples from MNIST [52] in lower resolution. The Top-1 accuracy is reported in Tab. 6a. Our GH-Feat outperforms ALAE [43] and BiGAN [15] with 1.45% and 1.92%, suggesting a stronger discriminative power. Here, ResNet-18 [24] is employed as the backbone structure for both MoCo [31] and GH-Feat.

LFW Face Verification. We directly use the proposed encoder, which is trained on FF-HQ [11], to extract GH-Feat from face images in LFW [54] and tries three different strategies on exploiting GH-Feat for face verification: (1) using a single level feature; (2) grouping multi-level features (starting from the highest level) together; (3) voting by choosing the largest face similarity across all levels. Fig. 13 (last column) shows the results from the first two strategies.



Fig. 14. **Image reconstruction** results on LFW [54]. For each pair of images, left is the low-resolution input while right is reconstructed by GH-Feat. All samples are with the same identity.



Fig. 15. **Qualitative comparison** between BigBiGAN [17] and GH-Feat on reconstructing images from ImageNet [2].

TABLE 6

Quantitative comparison between our proposed GH-Feat and other alternatives on MNIST [52] and LFW [54]. GH-Feat-R denotes GH-Feat trained with regularizer.

(a) Digit recognition on MNIST.		(b) Face verification on LFW.	
Methods	Acc.	Methods	Acc.
AE(ℓ_1) [64]	97.43	VAE [65]	49.3
AE(ℓ_2) [64]	97.37	MoCo-R50 [31]	48.9
BiGAN [15]	97.14	ALAE [43]	55.7
ALAE [43]	97.61	GH-Feat (Grouping)	60.1
MoCo-R18 [31]	95.89	GH-Feat (Layer-wise)	67.5
GH-Feat (Ours)	99.06	GH-Feat (Voting)	69.7
GH-Feat-R	98.78	GH-Feat-R (Voting)	69.1

Obviously, GH-Feat from the 5-th to the 9-th levels best preserve the identity information. Tab. 6b compares GH-Feat with other unsupervised feature learning methods, including VAE [65], MoCo [31], and ALAE [43]. All these competitors are also trained on FF-HQ dataset [11] with optimally chosen hyper-parameters. ResNet-50 [24] is employed as the backbone for MoCo and GH-Feat. Our method with voting strategy achieves 69.7% accuracy, surpassing other competitors by a large margin. We also visualize some reconstructed LFW faces in Fig. 14, where our GH-Feat well handles the domain gap (e.g., image resolution) and preserves the identity information.

4.4.3 Large-Scale Image Classification

We further evaluate GH-Feat on the high-level image classification task using ImageNet [2]. Before the training of encoder, we first train a StyleGAN model, with 256×256 resolution, on the ImageNet training collection. After that, we learn the hierarchical encoder by using the pre-trained generator as the supervision. No labels are involved in

the above training process.² For the image classification problem, we train a linear model on top of the features extracted from the training set with the softmax loss. Then, this linear model is evaluated on the validation set.³ Tab. 7 shows the comparison between GH-Feat and other unsupervised representation learning approaches [15], [17], [31], [32], [67], [68], where we beat most of the competitors. The state-of-the-art MoCo [31] gives the most compelling performance. But different from the representations learned with contrastive learning, GH-Feat has huge advantages in generative tasks, as already discussed in Sec. 4.3. Among adversarial representation learning approaches, BigBiGAN [17] achieves the best performance, benefiting from the incredible large-scale training. However, GH-Feat presents a stronger ability for image reconstruction. BigBiGAN is learned by discriminating the data-latent joint distribution, while our GH-Feat targets image reconstruction by treating a well-trained GAN generator as a learned loss function. Consequently, as shown in Fig. 15, BigBiGAN can only recover the input images from the category level, instead, our approach can recover the inputs with much more details. The reconstruction error in Tab. 8 conveys the same conclusion. This is also the reason why GH-Feat could facilitate various low-level and middle-level discriminative tasks beyond image classification. More details about ImageNet training can be found in **Supplementary Material**.

4.4.4 Transfer Learning

In this part, we explore how GH-Feat can be transferred from one dataset to another.

2. Our encoder can be trained very efficiently, usually $3 \times$ faster than the GAN training.

3. During testing, we adopt the fully convolutional form as in [66] and average the scores at multiple scales.

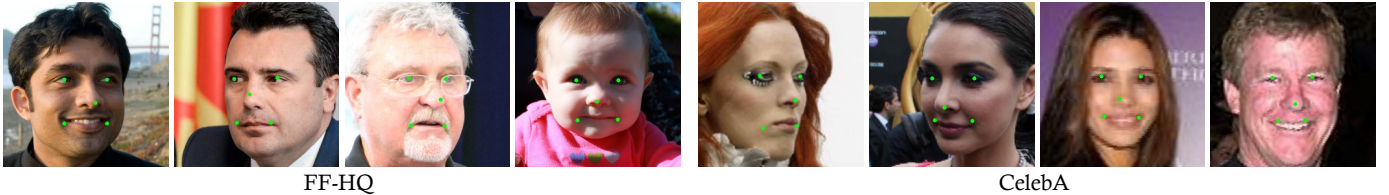


Fig. 16. **Landmark detection** results. GH-Feat is trained on FF-HQ [11] dataset but can successfully handle the hard cases (large pose and low image quality) in MAFL dataset [55], a subset of CelebA [74].

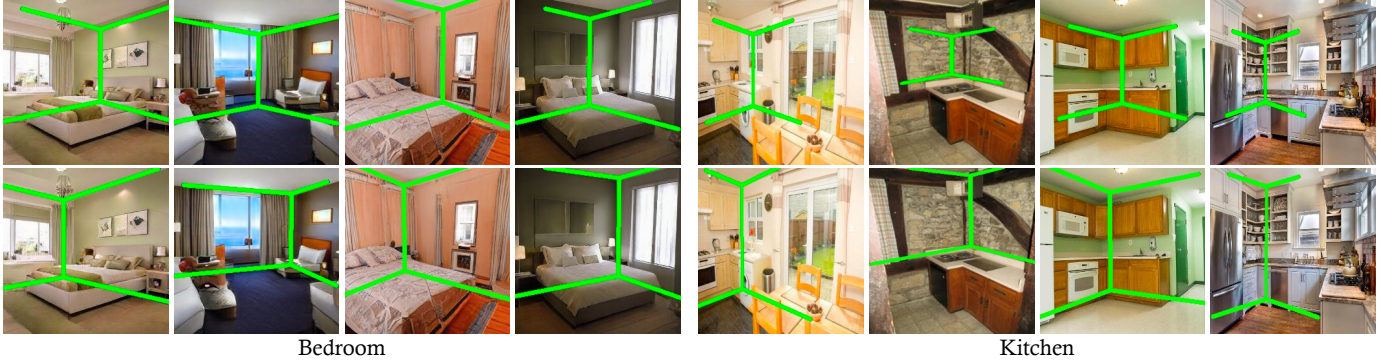


Fig. 17. **Layout prediction** results using feature learned by MoCo [31] (top row) and our GH-Feat (bottom row). Both methods are trained on LSUN bedrooms [53] and then transferred to LSUN kitchens.

TABLE 7

Quantitative comparison on the ImageNet [2] classification task.

Method	Architecture	Top-1 Acc.
Motion Seg (MS) [69], [70]	ResNet-101	27.6
Exemplar (Ex) [70], [71]	ResNet-101	31.5
Relative Po (RP) [70], [72]	ResNet-101	36.2
Colorization (Col) [70], [73]	ResNet-101	39.6
<i>Contrastive Learning</i>		
InstDisc [67]	ResNet-50	42.5
CPC [32]	ResNet-101	48.7
MoCo [31]	ResNet-50	60.6
<i>Generative Modeling</i>		
BiGAN [15]	AlexNet	31.0
SS-GAN [68]	ResNet-19	38.3
BigBiGAN [17]	ResNet-50	55.4
GH-Feat (Ours)	ResNet-50	51.1

TABLE 8

Qualitative comparison between BigBiGAN [17] and GH-Feat on reconstructing images from ImageNet [2].

	MSE↓	SSIM↑	FID↓
BigBiGAN [17]	0.363	0.236	33.42
GH-Feat (Ours)	0.078	0.431	22.70

Landmark Detection. We train a linear regression model using GH-Feat on FF-HQ [11] and test it on MAFL [55], which is a subset of CelebA [74]. This two datasets have a large domain gap, *e.g.*, faces in MAFL have larger poses yet lower image quality. As shown in Fig. 16, GH-Feat shows a strong transferability across these two datasets. We compare our approach with some supervised and unsupervised alternatives [31], [55], [75], [76]. CLIP [77] trained with 400,000,000 image-text paired samples is also included to serve as a strong baseline to compare with GH-Feat. For a fair comparison, we try the multi-scale representations from MoCo [31] and CLIP [77] (*i.e.*, Res2, Res3, Res4, and Res5 feature maps) and report the best results. Tab. 9 demonstrates the strong generalization ability of GH-Feat.

TABLE 9

Landmark detection results on MAFL [55]. GH-Feat-R denotes GH-Feat trained with regularizer.

Method	Supervision	MSE↓
TCDCN [55]	✓	7.95
MTCNN [75]	✓	5.39
Cond. ImGen [76]		4.95
ALAE [43].		10.13
MoCo-R50 [31]		9.07
CLIP-R50		4.98
GH-Feat (Ours)		5.12
GH-Feat-R		4.92

In particular, it achieves on-par or better performance than the methods that are particular designed for this task [55], [75], [76]. Also, it outperforms MoCo [31] on this mid-level discriminative task. As the Tab. 9 below suggests, GH-Feat achieves comparable performance as CLIP-R50 with significantly better data efficiency. Such a comparison is not 100% eye-to-eye because our approach is particularly trained on human faces while CLIP could cover a much larger data domain. But it still demonstrates, to some extent, that adequately leveraging the pre-trained GAN generator as a learned loss function yields a discriminative and transferable visual representation.

Layout Prediction. We train the layout predictor on LSUN [53] bedrooms and test it on kitchens to validate how GH-Feat can be transferred from one scene category to another. Feature learned by MoCo [31] on the bedroom dataset is used for comparison. We can tell from Fig. 17 that GH-Feat shows better predictions than MoCo, especially on the target set (*i.e.*, kitchens), suggesting a stronger transferability. Like landmark detection, we also conduct experiments with the 4-level representations from MoCo [31] and select the best.

4.4.5 Image Retrieval

In this section, we verify the hierarchical property of the proposed GH-Feat with image retrieval. Concretely, given a

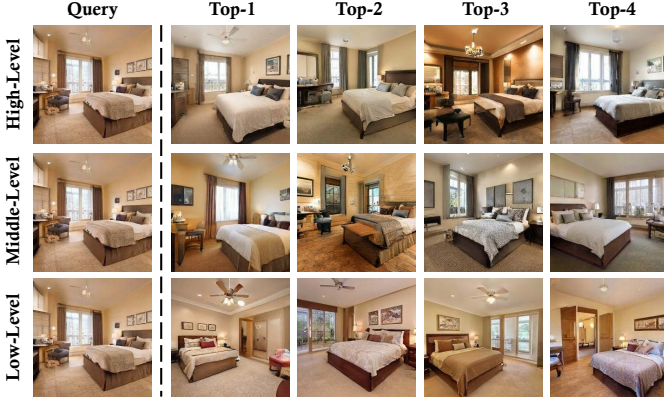


Fig. 18. Retrieval results on LSUN bedroom [53].

query image, we use encoder to extract its GH-Feat. Then, we use different levels of GH-Feat to perform retrieval from 10K real images. Note that GH-Feat from these 10K images are prepared in advance and ℓ_1 distance is used as the metric for retrieval. Fig. 18 shows the retrieval results on LSUN bedroom [53]. We can tell that when we use higher level (first row) features for retrieval, all retrieved results are with the same layout as the query image, but they may have different lighting conditions. Meanwhile, when using lower level (bottom row) features for retrieval, the retrieved results are with similar lighting condition as the query image.

4.5 Spatial Expansion

4.5.1 Spatial GH-Feat

Spatial-Aware Style Codes. Even though the layer-wise style codes can describe the global semantics of synthesized images, the fine-grained semantics cannot be expressed precisely because the style codes are too coarse to maintain spatial semantics. To facilitate the style codes with semantic segmentation, we equip the layer-wise style codes with spatial dimension. It is noteworthy that the introduced spatial dimension make the layer-wise representation more flexible for various of vision tasks.

Spatial-Aware Encoder. For the vision tasks requiring the spatial-aware representation of the input image, a spatial-aware encoder is also needed to produce the spatial-aware style codes. We inherit the backbone and FPN to fuse the semantics encoded at different level. The last three stages feature maps $\{R_4, R_5, R_6\}$, are used to produce spatial-aware GH-Feat. We also use the same instantiation for the layer equipment. But differently, we use an 1×1 convolution layer to embed the feature maps $\{R_4, R_5, R_6\}$ and an upsampler to match the spatial size of the corresponding convolution feature map. It can be formulated as:

$$GH_j = \text{up}(W_j R_{a[j]}, h_{C_j} / h_{R_{a[j]}}) \quad j \in \{1, N\},$$

where GH_j is the learned spatial-aware representation, C_j is the convolutional feature map, $a[j]$ denotes the corresponding index of the output feature map from FPN, and $h_{C_j}, h_{R_{a[j]}}$ denotes the spatial dimension of feature map C_j and $R_{a[j]}$.

Ablation. The proposed spatial generative feature is adopted to provide spatial information, and thus it is critical to the quality of the reconstructed image. As shown in Tab. 10, the spatial generative feature can improve the

TABLE 10
Quantitative comparison on image reconstruction between GH-Feat and spatial GH-Feat. GH-Feat-R denotes GH-Feat trained with regularizer.

Method	Face		Bedroom	
	MSE↓	SSIM↑	MSE↓	SSIM↑
GH-Feat	0.046	0.56	0.068	0.52
GH-Feat-R	0.049	0.55	0.070	0.50
Spatial GH-Feat	0.029	0.67	0.057	0.58

reconstruction performance, and the qualitative results in Fig. 21 present that the spatial GH-Feat is able to reconstruct the background and the out-of-the-distribution objects *i.e.* hands and hats well. It supports the effectiveness of the spatial-aware generative features.

4.5.2 Data-Efficient Semantic Segmentation

Compared with classification, image segmentation needs more precise prediction along the spatial dimension. However, the generative features without spatial dimension cannot facilitate this task because they cannot be aware of the semantics for each pixel. To enable this task, we use the spatial-aware encoder to obtain a set of generative features with spatial dimension, and a segmentation head *i.e.* the Style Interpreter in [78] is followed to obtain the segmentation results. Because of the generalization of the spatial visual features, we only need a few samples to achieve a good segmentation head. In our experiment, we used 20 annotated samples for the training. We visualize predictions learned from our visual features in Fig. 19. Obviously, the spatial-aware generative features provide precise information for dense pixels, facilitating image segmentation with a few annotations.

We include several extreme cases in Fig. 20 to verify the robustness of the segmentation results achieved by GH-Feat. Concretely, we include samples under extreme poses, as well as samples containing out-of-distribution objects (*i.e.*, the objects without annotations during the training of the segmentation branch). We have three observations: (1) Even there are few samples under extreme poses during training, our approach could still produce promising segmentation results on such challenging cases at the inference stage. (2) The model could well recognize the eyeglass frames yet perform poorly on eyeglass lens. We guess this is caused by the overlap between lens and eyes. (3) Hats (recognized as hair), earrings and microphones (recognized as background) could be regarded as failure cases, because our segmentation branch is learned with simple annotations (*e.g.*, eyes, nose, cheek, *etc.*). A more competitive performance could be expected given richer segmentation labels.

5 CONCLUSION

In this work, we consider the well-trained GAN generator as a learned loss function for learning multi-scale features. Unlike previous work, we treat layer-wise style codes in \mathcal{Y} space as generative visual features rather than \mathcal{W} space, resulting in better hierarchical properties. A distribution-level regularizer is introduced to overcome the limitation of only using image-level supervision for encoder training. The resulting Generative Hierarchical Features are shown

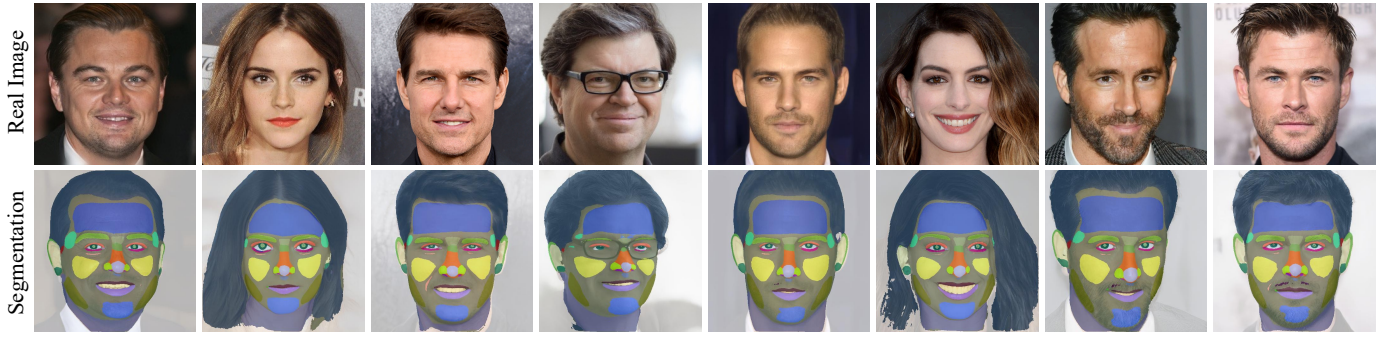


Fig. 19. **Data-efficient Image Segmentation** with Spatial GH-Feat. We use the spatial-aware encoder to obtain a set of generative features with spatial dimension and a segmentation head trained with limited annotated data to obtain segmentation results.

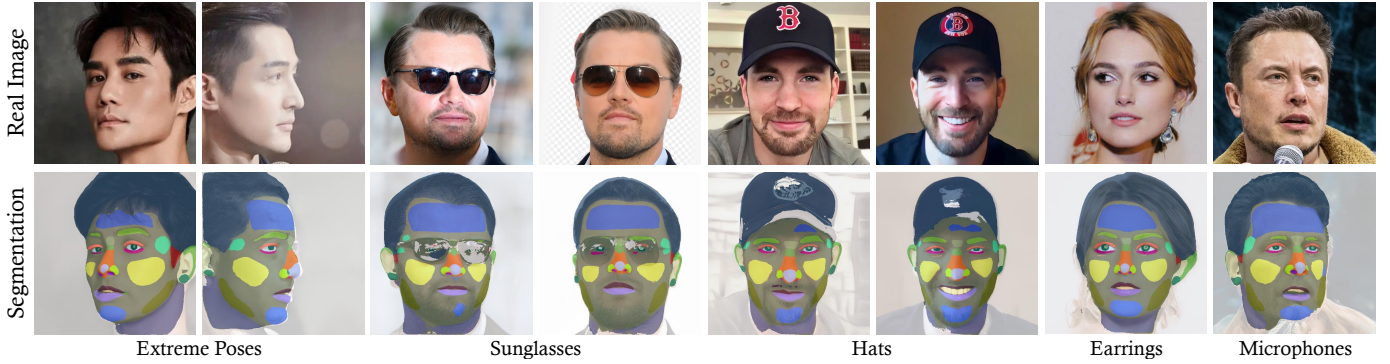


Fig. 20. Extreme cases of data-efficient image segmentation with Spatial GH-Feat. These extreme samples (*i.e.* extreme pose, hat, sunglasses, earrings as well as microphones) show the robustness of the segmentation head only trained with fewer annotated samples.

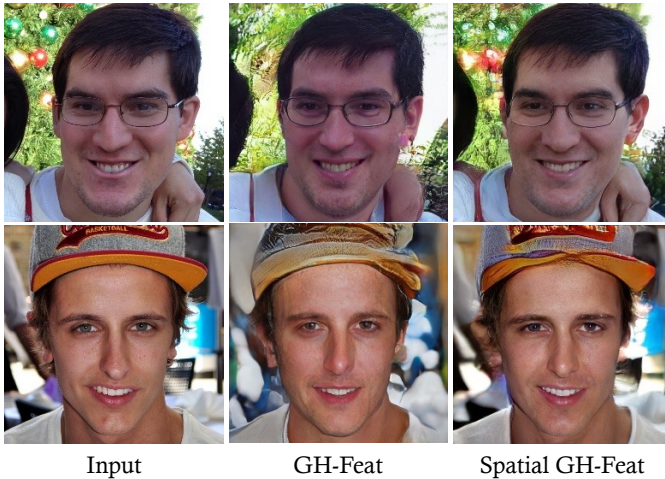


Fig. 21. Qualitative comparison between GH-Feat and spatial GH-Feat.

to be generalizable to a wide range of vision tasks. Since GH-Feat only leverages the semantics learned in GANs, the features may lack the good properties of the discriminative model features. In the future, we hope to learn deep representations by unifying discriminative and generative models that can complement each other.

ACKNOWLEDGMENTS

This work is supported in part by the Early Career Scheme (ECS) through the Research Grants Council (RGC) of Hong Kong under Grant No.24206219, Grant No.14204521, CUHK FoE RSFS Grant, and Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Fund.

REFERENCES

- [1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013. **1**
- [2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2009. **1, 5, 11, 12, 16**
- [3] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017. **1**
- [4] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2014. **1**
- [5] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. **1**
- [6] D. Matthew and R. Fergus, "Visualizing and understanding convolutional neural networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014. **1**
- [7] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," in *Int. Conf. Learn. Represent.*, 2015. **1**
- [8] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Adv. Neural Inform. Process. Syst.*, 2014. **1, 2**
- [9] N. Zhao, Z. Wu, R. W. Lau, and S. Lin, "What makes instance discrimination good for transfer learning?" *arXiv preprint arXiv:2006.06606*, 2020. **1, 2**
- [10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Adv. Neural Inform. Process. Syst.*, 2014. **1, 2, 4, 7**
- [11] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. **1, 2, 3, 4, 5, 7, 8, 10, 11, 12**
- [12] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 8110–8119. **1**
- [13] Y. Shen, C. Yang, X. Tang, and B. Zhou, "Interfacegan: Interpreting the disentangled face representation learned by gans," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. **1, 2, 8, 10**

- [14] C. Yang, Y. Shen, and B. Zhou, "Semantic hierarchy emerges in deep generative representations for scene synthesis," *Int. J. Comput. Vis.*, 2020. **1, 2, 4**
- [15] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *Int. Conf. Learn. Represent.*, 2017. **1, 2, 5, 10, 11, 12**
- [16] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," in *Int. Conf. Learn. Represent.*, 2017. **1, 2**
- [17] J. Donahue and K. Simonyan, "Large scale adversarial representation learning," in *Adv. Neural Inform. Process. Syst.*, 2019. **1, 2, 5, 10, 11, 12, 16**
- [18] Y. Xu, Y. Shen, J. Zhu, C. Yang, and B. Zhou, "Generative hierarchical features from synthesizing images," in *CVPR*, 2021. **2**
- [19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, 2004. **2**
- [20] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Eur. Conf. Comput. Vis.*, 2006. **2**
- [21] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2005. **2**
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Adv. Neural Inform. Process. Syst.*, 2012. **2**
- [23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Int. Conf. Learn. Represent.*, 2015. **2, 4**
- [24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. **2, 3, 4, 7, 10, 11, 16**
- [25] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *Int. Conf. Comput. Vis.*, 2017. **2**
- [26] R. Zhang, P. Isola, and A. A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. **2**
- [27] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. **2**
- [28] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Int. Conf. Learn. Represent.*, 2018. **2**
- [29] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *Int. Conf. Learn. Represent.*, 2019. **2, 7, 10**
- [30] C. Zhuang, A. L. Zhai, and D. Yamins, "Local aggregation for unsupervised learning of visual embeddings," in *Int. Conf. Comput. Vis.*, 2019. **2, 7, 10**
- [31] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. **2, 7, 10, 11, 12**
- [32] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018. **2, 7, 10, 11, 12**
- [33] O. J. Hénaff, A. Srinivas, J. De Fauw, A. Razavi, C. Doersch, S. Eslami, and A. v. d. Oord, "Data-efficient image recognition with contrastive predictive coding," *arXiv preprint arXiv:1905.09272*, 2019. **2**
- [34] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *arXiv preprint arXiv:1906.05849*, 2019. **2, 7**
- [35] A. Shocher, Y. Gandelsman, I. Mosseri, M. Yarom, M. Irani, W. T. Freeman, and T. Dekel, "Semantic pyramid for image generation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. **2**
- [36] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Int. Conf. Learn. Represent.*, 2016. **2**
- [37] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *Int. Conf. Learn. Represent.*, 2018. **2**
- [38] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *Int. Conf. Learn. Represent.*, 2019. **2**
- [39] A. Jahanian, L. Chai, and P. Isola, "On the "steerability" of generative adversarial networks," in *Int. Conf. Learn. Represent.*, 2020. **2**
- [40] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," in *Int. Conf. Learn. Represent.*, 2019. **2**
- [41] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code gan prior," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. **2**
- [42] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *Eur. Conf. Comput. Vis.*, 2020. **2**
- [43] S. Pidhorskyi, D. Adjeroh, and G. Doretto, "Adversarial latent autoencoders," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. **2, 7, 8, 10, 11, 12**
- [44] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Int. Conf. Comput. Vis.*, 2017. **3, 4**
- [45] Z. Wu, D. Lischinski, and E. Shechtman, "Stylespace analysis: Disentangled controls for stylegan image generation," in *CVPR*, 2021. **4**
- [46] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. **4**
- [47] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. **4**
- [48] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. **4**
- [49] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for StyleGAN image manipulation," *ACM Trans. Graph.*, 2021. **4, 6, 7, 8**
- [50] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Eur. Conf. Comput. Vis.*, 2016. **4**
- [51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Int. Conf. Learn. Represent.*, 2015. **5**
- [52] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998. **5, 10, 11**
- [53] 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. **5, 7, 8, 12, 13**
- [54] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Technical Report 07-49, University of Massachusetts, Amherst, Tech. Rep., 2007. **5, 10, 11**
- [55] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *Eur. Conf. Comput. Vis.*, 2014. **5, 12**
- [56] W. Zhang, W. Zhang, and J. Gu, "Edge-semantic learning strategy for layout estimation in indoor environment," *Transactions On Cybernetics*, 2019. **5**
- [57] C. Zou, A. Colburn, Q. Shan, and D. Hoiem, "Layoutnet: Reconstructing the 3d room layout from a single rgb image," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. **5**
- [58] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, 2004. **5, 7**
- [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 *Adv. Neural Inform. Process. Syst.*, 2017. **5**
- [60] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. **6, 7, 8**
- [61] D. Roich, R. Mokady, A. H. Bermano, and D. Cohen-Or, "Pivotal tuning for latent-based editing of real images," *ACM Trans. Graph.*, 2021. **7, 8**
- [62] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "ReStyle: A residual-based StyleGAN encoder via iterative refinement," in *Int. Conf. Comput. Vis.*, 2021. **7, 8**
- [63] K. Liu, "Pytorch cifar10," <https://github.com/kuangliu/pytorch-cifar.git>, 2019. **10**
- [64] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, 2006. **11**
- [65] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Int. Conf. Learn. Represent.*, 2014. **11**
- [66] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with

- convolutions,” in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015. 11
- [67] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, “Unsupervised feature learning via non-parametric instance discrimination,” in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 11, 12
- [68] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby, “Self-supervised gans via auxiliary rotation loss,” in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 11, 12
- [69] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, “Learning features by watching objects move,” in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 12
- [70] C. Doersch and A. Zisserman, “Multi-task self-supervised visual learning,” in *Int. Conf. Comput. Vis.*, 2017. 12
- [71] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, “Discriminative unsupervised feature learning with convolutional neural networks,” in *Adv. Neural Inform. Process. Syst.*, 2014. 12
- [72] C. Doersch, A. Gupta, and A. A. Efros, “Unsupervised visual representation learning by context prediction,” in *Int. Conf. Comput. Vis.*, 2015. 12
- [73] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *Eur. Conf. Comput. Vis.*, 2016. 12
- [74] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Int. Conf. Comput. Vis.*, 2015. 12
- [75] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Processing Letters*, 2016. 12
- [76] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, “Unsupervised learning of object landmarks through conditional image generation,” in *Adv. Neural Inform. Process. Syst.*, 2018. 12
- [77] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark et al., “Learning transferable visual models from natural language supervision,” in *ICML*, 2021. 12
- [78] Y. Zhang, H. Ling, J. Gao, K. Yin, J.-F. Lafleche, A. Barriuso, A. Torralba, and S. Fidler, “Datasetgan: Efficient labeled data factory with minimal human effort,” in *CVPR*, 2021. 13
- [79] T. Karras, S. Laine, and T. Aila, “Stylegan - official tensorflow implementation,” <https://github.com/NVlabs/stylegan>, 2019. 16

A1. HIERARCHICAL PROPERTY

We also re-evaluate the layer-wise representation on different discriminative tasks. As shown in Fig. 22, the training regularizer improves the hierarchical property of the original GH-Feat. Since the training regularizer prevents the model from overfitting pixel values, the layer-wise representation is closer to the distribution center and achieve better hierarchical properties on the discriminative tasks.

A2. EXPERIMENTS ON IMAGENET

Training Details. During the training of the StyleGAN model on the ImageNet dataset [2], we resize all images in the training set such that the short side of each image is 256, and then centrally crop them to 256×256 resolution. All training settings follow the StyleGAN official implementation [79], including the progressive strategy, optimizer, learning rate, *etc.* The generator and the discriminator are alternatively optimized until the discriminator have seen 250M real images. After that, the generator is fixed and treated as a well-learned loss function to guide the training of the encoder. During the training of the hierarchical encoder, images in the training collection are pre-processed in the same way as mentioned above. After the encoder is ready, we treat it as a feature extractor. We use the output feature map at the “res₅” stage, apply adaptively average pooling to obtain 2×2 spatial feature and vectorize it. A linear classifier, *i.e.*, with one fully-connected layer, takes these extracted features as the inputs to learn the image classification task. SGD optimizer, together with batch size 2048, is used. The learning rate is initially set as 1 and decayed to 0.1 and 0.01 at the 60-th and the 80-th epoch respectively. During the training of the final classifier, ResNet-style data augmentation [24] is applied.

The FID score on ImageNet is 40.92. Fig 23 shows the uncurated samples of the pretrained ImageNet samples. Although the synthesized samples are not very realistic, they can still help downstream tasks like ImageNet classification.

Discussion. We have already shown in the main submission that GH-Feat achieves comparable accuracy to existing alternatives. Especially, among all methods based on generative modeling, GH-Feat obtains second performance only to BigBiGAN [17], which requires incredible large-scale training. However, as discussed in the main submission, our GH-Feat facilitates a wide range of tasks besides image classification. Taking image reconstruction as an example, our approach can well recover the input image, significantly outperforming BigBiGAN [17].

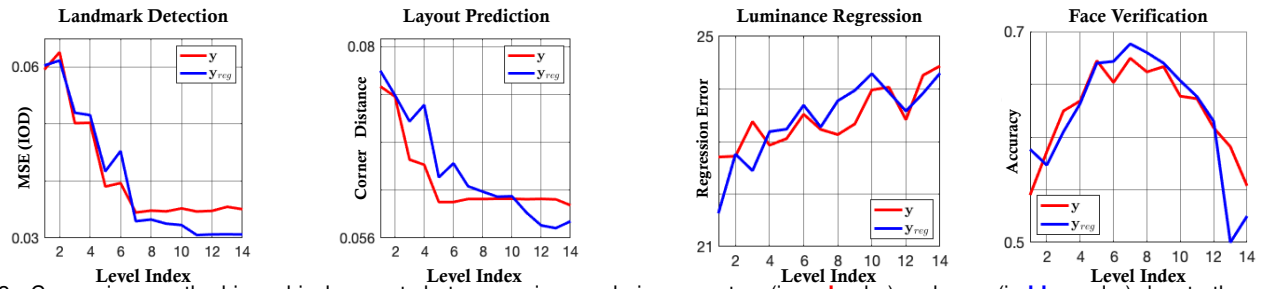


Fig. 22. Comparison on the hierarchical property between using regularizer or not. y (in red color) and y_{reg} (in blue color) denote the original GH-Feat and GH-Feat with regularizer, respectively.

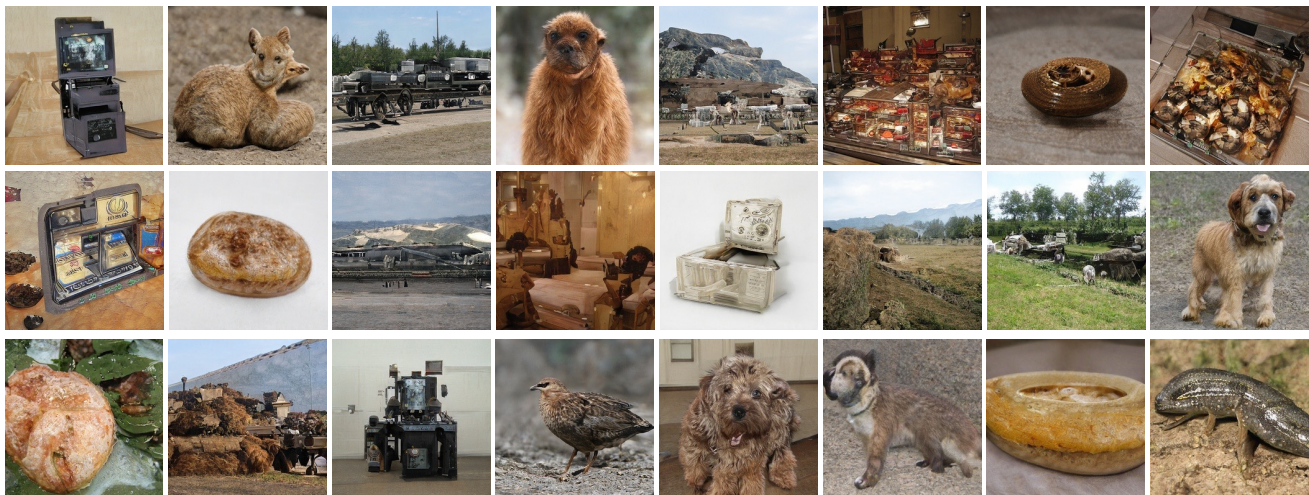


Fig. 23. **Uncrated generated samples** of StyleGAN model on ImageNet.