NaturalInversion: Data-Free Image Synthesis Improving Real-World Consistency

Yujin Kim$^{1,2}$, Dogyun Park$^{1,2}$, Dohee Kim$^2$, Suhyun Kim$^{2,*}$

$^1$Korea University, Republic of Korea
$^2$Korea Institute of Science and Technology, Republic of Korea
{lakeye1220, koparkrea, kimdohe1070, dr.suhyun.kim}@gmail.com

Abstract

We introduce NaturalInversion, a novel model inversion-based method to synthesize images that agrees well with the original data distribution without using real data. In NaturalInversion, we propose: (1) a Feature Transfer Pyramid which uses enhanced image prior of the original data by combining the multi-scale feature maps extracted from the pre-trained classifier, (2) a one-to-one approach generative model where only one batch of images are synthesized by one generator to bring the non-linearity to optimization and to ease the overall optimizing process, (3) learnable Adaptive Channel Scaling parameters which are end-to-end trained to scale the output image channel to utilize the original image prior further. With our NaturalInversion, we synthesize images from classifiers trained on CIFAR-10/100 and show that our images are more consistent with original data distribution than prior works by visualization and additional analysis. Furthermore, our synthesized images outperform prior works on various applications such as knowledge distillation and pruning, demonstrating the effectiveness of our proposed method. Code is available at https://github.com/kdst-team/NaturalInversion.git

Introduction

Convolutional Neural Networks (CNNs) have achieved a great success in various computer vision tasks such as image classification, image generation, etc (Simonyan and Zisserman 2014; Goodfellow et al. 2014). The emergence of large datasets (Krizhevsky and Hinton 2009; Deng et al. 2009) has led to a progressive improvement in various applications (Zhai et al. 2021; Chu et al. 2020). The CNNs learn meaningful feature spaces with rich information from low-level features to high-level semantic content (Zeiler and Fergus 2014). Hence, several works have been proposed to transfer knowledge from pre-trained networks into a lightweight model to inference in various conditions. Specially, Knowledge Distillation (Hinton, Vinyals, and Dean 2015) transfers the knowledge from the pre-trained teacher network to student networks. Network pruning (Liu et al. 2017) reduces redundant neural connections of a pre-trained network and fine-tunes the remaining weights to recover the accuracy. However, these methods require original data to train or fine-tune the compressed networks, limiting their use in privacy-sensitive or data-limited scenarios.

To overcome these scenarios, several approaches have synthesized the data from a pre-trained network without original datasets or images prior (Mahendran and Vedaldi 2015). Recently, Yin et al. (2020) has proposed a novel method to optimize the raw input space $x$ to synthesize images, using a regularizer that follows the statistics in the batch-normalization (BN) layers of a pre-trained classifier. However, we argue that optimizing the high-dimensional raw input space is challenging for two reasons: 1) limited image prior since the statistics in BN layer are averaged, ignoring the specific information for individual images, 2) optimization without non-linearity. These problems lead to sub-optimized images which are heavily inconsistent with original data distribution: low fidelity and low diversity of synthesized images. Therefore, these images cause performance degradation in applications compared to performance trained with original dataset.

Therefore, for the richer image prior, we focus on features of pre-trained CNNs, since they encode the low to high level image prior on multi-scale feature maps (Islam, Jia, and Bruce 2020). The sparsity in feature maps eliminates the irrelevant variability of the input data while preserving the important information (Xu et al. 2021). Furthermore, feature maps of each layer encourage to capture multi-scale characteristics of the target class and real data distribution. Thus, utilizing the feature maps of the pre-trained classifier for generation strengthens the real data characteristics on the synthesized images (Shocher et al. 2020; Wang et al. 2021).

Inspired by the above methods, we propose the sub-generator: Feature Transfer Pyramid (FTP) that uses multi-scale feature maps from the pre-trained network to generated images. Thus, FTP enhances the characteristics of real data by sequentially combining the multi-scale feature maps from a pre-trained classifier. By adding the final combined multi-scale feature maps to synthesized image, it strengthens the original data characteristics on the synthesized images, making images more consistent to original data distribution. Secondly, we use the CNN-based generator to bring a non-linearity in optimization, which makes generator possess higher ability to represent the complex characteristics. We further propose the one-to-one approach, which utilizes...
proposed methods can be summarized as follows:

- We propose a Feature Transfer Pyramid, which enhances the synthesized images with the multi-scale feature maps extracted from a pre-trained classifier.
- We introduce the one-to-one approach based generator to ease the optimizing process and bring a non-linear optimization.
- We further implicitly learn the image prior of original data channel scale by end-to-end training the Adaptive Channel Scaling parameters.

Related Work

Model Inversion. Inverting a pre-trained model helps to interpret and understand the deep representation that the model stored. Several works have reconstructed what the model learned by optimizing input space from noise to image with regularizers. DeepDream (Mordvintsev, Olah, and Tyka 2015) visualizes the pattern that the model watch by selecting one or more layers of the model and optimizing the input space to maximize the class probability using the arbitrary target label \( \hat{y} \). Follow-up studies such as DAFL (Chen et al. 2019) simultaneously perform inversion and knowledge distillation, training a generator to minimize the cross-entropy loss and maximize the representation of the pre-trained teacher network. DFAD (Fang et al. 2019) trains the generator that maximizes disagreement between the predictions of the pre-trained teacher and the student model, while a student network reduces the discrepancy with a teacher to generate more confusing samples. However, DAFL and DFAD concentrate on synthesizing the images for a specific task, such as knowledge distillation. Thus, these images are far from the original data distribution, leading to performance degradation on various vision applications. To alleviate this issue, Yin et al. (2020); Haroush et al. (2020) use a regularizer for channel-wise feature distribution matching with stored BN statistics to improve consistency with original data distribution, achieving the improvement in other applications. However, due to the problems with optimizing the input space with limited image prior, their images still lack of fidelity and diversity.

Usage of Feature Maps. The feature maps from pre-trained CNNs can provide insight into which characteristic in feature map does the classifier intensively detects for. The low-level feature maps generated by the shallower layers of CNNs encode basic representations such as edges and corners of real images. The deeper features include more complex semantic representations such as complicated geo-
metric shapes in their feature maps (Yosinski et al. 2015). For confirming the encoded information of CNNs, many works visualize feature maps of a pre-trained model to interpret feature representation in intermediate layers (Zeiler and Fergus 2014; Olah, Mordvintsev, and Schubert 2017). Inspired by previous works, several studies have captured realistic images using the properties of feature maps (Kalischek, Wegner, and Schindler 2021; Heitz et al. 2021; Xu et al. 2021). Many style transfer algorithms ( Gatys, Ecker, and Bethge 2016; Lin et al. 2021; Kalischek, Wegner, and Schindler 2021) extract the multi-scale feature maps on multiple layers by forwarding the style image to the network. Then they render the style from an image to a content image while preserving the semantic information of the content image. Gatys, Ecker, and Bethge (2015) utilizes the correlations between feature maps in several layers to capture spatial-based texture representations. GAN-based Semantic Generation Pyramid (Shocher et al. 2020) proposes to feed the target image to a pre-trained classifier and combine both the feature map and same level generator block output to replicate the classifier features. We draw inspiration from several tasks of using multi-scale feature maps to obtain different scale representations of pre-trained CNNs. Our proposed method gradually enhances the real data characteristics encapsulated in a pre-trained teacher network to capture the distribution of the original dataset.

**Method**

We aim to synthesize more realistic images which agrees well with the original data distribution. In this section, we provide a brief background and introduce our NaturalInversion methods.

**Preliminary**

Model inversion approaches update the input space to get an image \( \hat{x} \) given the pre-trained teacher network \( T \). Given trainable input space \( \hat{x} \) with randomly initialized noise and arbitrary target label \( y \), the input space is updated by minimizing the below objective function.

\[
\min_{\hat{x}} \mathcal{L}(\hat{x}, y) + \mathcal{R}(\hat{x})
\]

We can divide objective function into two part, loss function \( \mathcal{L} \) and regularizer \( \mathcal{R} \) that captures natural image prior.

**Inception Loss.** Inceptionism-style images synthesis approach (Mordvintsev, Olah, and Tyka 2015) uses inception loss to maximizes the probability of the expected target class produced by the pre-trained teacher model. Given an arbitrary target label \( \hat{y} \), it encourages minimizing the cross-entropy loss to find out the class probability distribution of original data, as below:

\[
\mathcal{L}_{CE} = CE(T(\hat{x}), y) = - \sum_{i} \hat{y}_i \log y_i
\]

**Feature Distribution Regularizer.** For more image prior, DI (Yin et al. 2020) synthesizes the images that follow the original dataset distribution stored in the pre-trained teacher network. Given running mean \( \mu \), and running variance \( \sigma^2 \) of each BN layer (Ioffe and Szegedy 2015), they minimize the difference between the channel-wise mean \( \mu(\hat{x}) \) and variance \( \sigma^2(\hat{x}) \) of synthesized images and running statistics of every BN layer.

\[
\mathcal{R}_{BN} = \sum_{i=1}^{l}(\|\mu_i(\hat{x}) - \mu_i\|_2 + \|\sigma_i^2(\hat{x}) - \sigma_i^2\|_2) \tag{3}
\]

where \( \mu_i(\hat{x}) \) and \( \sigma_i^2(\hat{x}) \) are the mean and variance of output feature maps from each network layer \( i \), and \( \mu_i \) and \( \sigma_i^2 \) denote the running mean and variance of pre-trained network.

**Image Prior Regularizer.** DeepDream(Mordvintsev, Olah, and Tyka 2015) uses image prior regularizers forcing synthetic images to be stably optimized following pre-defined prior:

\[
\mathcal{R}_{prior} = \lambda_{TV} \mathcal{R}_{TV}(\hat{x}) + \lambda_{l2} \mathcal{R}_{l2}(\hat{x}) \tag{4}
\]

\( \mathcal{R}_{TV} \) penalizes the sparsity in synthesized images with scaling factor \( \lambda_{TV} \). \( l_2 \) normalization, \( \mathcal{R}_{l2}(\hat{x}) = \|\hat{x}\|_2 \), encourages the synthesized images to have a small norm with scaling factor \( \lambda_{l2} \).

**Inversion Using Generative Model**

Our goal is to encourage the generator to implicitly approximate the original dataset distribution \( p_{real}(x) \). We use the conventional conditional GAN (Mirza and Osindero 2014) concept: the objective of the generator is to map \( z \) from \( p_z(z) \) to original data space \( X \) from \( p_{real}(x|y) \). We sample the latent vector \( z \) from the normal distribution \( \mathcal{N}(0, 1) \) concatenated with \( y \) encoded as one-hot vector. However, if the latent vector changes every training epoch, the latent vectors are largely ignored or have minor effects on the variations of the images in our framework. To address the mode-collapse problem, we propose a “one-to-one” approach, which utilizes one generator for synthesizing one batch of samples during the current batch synthesis step \( B \). Then, after sufficiently optimizing \( \theta_G \) for a latent vector \( z \), we re-sample \( z' \) and re-initialize the generator weight from \( \theta_G \) to \( \theta_G' \). As a result, the continuously re-initialized generator specifically captures a unique sample \( x \) corresponding to each \( z \) by optimizing the \( \theta_G \). Therefore, our generator can synthesize images from various modes with respect to the different \( B \), leading to diverse images. The generator architecture are shown in the appendix.

**Feature Transfer Pyramid**

Our goal is to utilize the richer image prior from the pre-trained classifier. Thus, we propose Feature Transfer Pyramid (FTP) as a sub-generator, a novel hierarchical schema to gradually capture the distribution of the original dataset from different scale feature maps. As shown in Fig.1, FTP operates in conjunction with a pre-trained classifier. More specifically, given the generator output \( G(z|y) \), we feed it into the pre-trained classifier and extract the feature maps \( F = \{f_1, f_2, f_3, ..., f_N \} \) with different layers. The \( f_1 \) denotes the low-level feature maps, and \( f_N \) is higher-level feature map. We extract the feature maps at the downsampling point.
Adaptive Channel Scaling

The pre-trained classifier is trained with dataset which is scaled to [0,1] and normalized with its mean and variation. Thus, the pre-trained classifier is biased to channel scale range of original data, which can be used as image prior. To implicitly utilize the optimal scale range of original data, we propose the Adaptive Channel Scaling parameters $\alpha$, which are learnable.

$$\alpha \in \mathbb{R}^{B \times C \times 1 \times 1}$$

(6)

Where the $B$ denotes the batch size, and the value of $C$ is 3 known as RGB channel. We iteratively multiply $\alpha$ to final images during inversion steps to adjust the output channel distribution of synthesized images. Finally, we can get the final synthesized images $\hat{I}$ using below equation:

$$\hat{I}_i = \alpha_i \odot (m_L \oplus G(z|y))$$

(7)

because different scale feature maps have different representations of a pre-trained classifier. The low-level feature maps include the simple and primary characteristics, and the higher-level feature map increasingly represents complex semantic representation. Fig.1(c) shows the building block of FTP. Now, we can get the outputs of each block, the Feature enhancement map $M = \{m_1, m_2, \ldots m_L\}$, as:

$$m_l = \begin{cases} W_1^l(\Phi(f_N)), & \text{if } l = 1 \\ W_1^l(\Phi(W_3^l(m_{l-1} \oplus f_{N-1}))), & \text{otherwise} \end{cases}$$

(5)

where $\Phi(\cdot)$ denotes the upsample layer, $W_1^l$ is the $l_{th}$ $1 \times 1$ convolution layer, and $W_3^l$ denotes the $l_{th}$ $3 \times 3$ convolution layer of FTP. First, $f_{N-1}$, the last feature map, is upsampled to fit the first FTP output size and fed to FTP as input. Then, the output of FTP block is summed with the $f_{N-1}$ which is a prior stage feature map from a classifier. Each output of FTP block is gradually added to feature maps at the corresponding stage of the classifier and undergoes convolution operations except the first block. Finally, the characteristic of original data is enhanced in the synthesized images by adding the $m_L$ to $G(z|y)$ element-wisely. FTP gradually captures the different spatial characteristics in the target class, and it leads to synthesized images which resembles the real data. We build up the layers of FTP when the spatial resolution change. More detailed structure of FTP are in the appendix.

Adaptive Channel Scaling

In this section, we evaluate the performance of NaturalInversion on CIFAR-10 and CIFAR-100 (Krizhevsky and Hinton 2009). Our experiments contain two parts, (1) analysis of our method: we verify that our method can synthesize more natural images (2) applications: we ensure the effectiveness of NaturalInversion on various applications in data-free conditions. Our experiments settings can be found in Alg. 1.
We further analyze our methods with the generative model evaluation metric to assess how similar our images are to the original dataset: (a) single-value metric - Inception Score (IS) (Salimans et al. 2016) and Frechet Inception Distance (FID) better whereas a lower score of FID is better.

Algorithm 1: NaturalInversion Algorithm

| Require: A pre-trained teacher network $T$ |
| Output: Inversion Images $I$ |
| 1: for number of batches $B$ do |
| 2: initialize Generator $G(\cdot; \theta_G)$, FTP $P(\cdot; \theta_F)$, $\alpha$ |
| 3: $z \leftarrow \mathcal{N}(0, 1)$ |
| 4: for inversion epoch $E$ do |
| 5: $\hat{x} \leftarrow G(z|y)$ $\triangleright$ generator output |
| 6: $m_L \leftarrow P(\hat{x})$ $\triangleright$ FTP output |
| 7: $\hat{x}' \leftarrow m_L + \hat{x}$ $\triangleright$ add last FTP output to $\hat{x}$ |
| 8: $\hat{I} \leftarrow \alpha \times \hat{x}'$ $\triangleright$ synthesize the final image |
| 9: $L_{inv} \leftarrow T(\hat{I})$ by Eq.8 $\triangleright$ compute loss function |
| 10: $\theta_G \leftarrow \theta_G - \eta_G \nabla_{\theta_G} L_{inv}$ |
| 11: $\theta_P \leftarrow \theta_P - \eta_P \nabla_{\theta_P} L_{inv}$ |
| 12: $\alpha \leftarrow \alpha - \eta_\alpha \nabla_\alpha L_{inv}$ |
| end for |
| return Mini batch Images $\hat{I}$ |

Analysis of NaturalInversion

We perform several studies to verify that synthesized images using our methods capture the original dataset distribution. As part of our experiments, we compare the (a) visualization, (b) t-SNE, (c) comparison of generative model evaluation metric result and (d) from-scratch training experiments.

Visualization. We show that our synthesized images of CIFAR-10 are more visually realistic than prior works in Fig. 2. This is because FTP further captures the characteristics of the target class: the shape or color of individual samples are more clearly reflected in synthesized images. Furthermore, we show that our “one-to-one” approach synthesizes more diverse images by optimizing particular generator for particular subset of original data as shown in Fig. 3. In conclusion, our method are more consistent with original data distribution with high fidelity and diversity than prior works.

Feature Visualization. To verify that our method encourages capturing original data distribution, we visualize the feature space from each residual block output and embedding layer of ResNet-34 using t-SNE (Van der Maaten and Hinton 2008). For visualizing feature space, we use the ResNet-34 trained by CIFAR-10 and compare the low-level feature space among 10k original CIFAR-10, DI, and our method. Entire t-SNE results are in the appendix. As shown in Fig. 4, the feature representations of DI have low diversity and are heavily different from the original distribution. In contrast, the feature representation of our method is more similar to original data than DI, which verifies that our method fairly estimates the internal feature representation of real data.

Generative Model Evaluation Metric. We further analyze our methods with the generative model evaluation metric to assess how similar our images are to the original dataset: (a) single-value metric - Inception Score (IS) (Salimans et al. 2016) and Frechet Inception Distance (FID) using our methods capture the original dataset distribution.

As part of our experiments, we compare the (a) visualization, (b) t-SNE, (c) comparison of generative model evaluation metric result and (d) from-scratch training experiments.

Figure 4: t-SNE of original CIFAR-10, DI, and ours. We extract the feature embedding from 1st and 2nd residual blocks. The green denotes the randomly sampling 10k original CIFAR-10, The orange is NaturalInversion 10k images and DI 10k images denote purple color.

| Dataset       | Method | IS  | FID  | Precision | Recall  |
|---------------|--------|-----|------|-----------|---------|
| CIFAR-10      | WGAN-GP | 7.86 | 29.3 | 0.7040    | 0.4353 |
|               | DI     | 2.67 | 197.33 | 0.5996    | 0.0028 |
|               | Ours   | 5.15 | 76.04 | 0.6797    | 0.2792 |
| CIFAR-100     | WGAN-GP | 6.08 | 37.30 | 0.7607    | 0.4023 |
|               | DI     | 4.08 | 151.52 | 0.6851    | 0.0319 |
|               | Ours   | 6.40 | 62.90 | 0.6845    | 0.2894 |

Recall (P&R) (Sajjadi et al. 2018). We specifically verify the qualitative fidelity and diversity of images by the two-value metrics. We synthesize 50k CIFAR-10/100 images from ResNet-34 and use the logit and embedding layer of ImageNet pre-trained Inception-v3 (Szegedy et al. 2016) for calculating the IS and remaining metrics respectively. To show how well we approximated the original data distribution, we compare the synthesized images with the images by DI and GAN-based model, WGAN-GP, which utilizes the original data. As shown in Table 1, NaturalInversion outperforms DI across CIFAR-10/100 in terms of IS, FID, P&R. In addition, NaturalInversion is even close to WGAN-GP baselines without original data. Through these evaluation metrics, we confirm that our method estimates the original distribution well without the real data.

From Scratch Training. To demonstrate how well our method captures the original distribution, we train the model from scratch with images by our method without any real data and compare the training accuracy of model with other methods: DAFL and DI. First, we synthesize 256k images using DAFL, DI, and our methods from ResNet-34, VGG-16 trained by CIFAR-100 dataset. Then, we train the randomly initialized networks from scratch using synthesized images. We set the mini-batch size as 128 and train the model for 200 epochs using an SGD optimizer with a 0.05 learning rate.
Table 2: From scratch experiments of CIFAR-100. We train the various models from scratch using synthesized images without any information about the training set or assistance from other pre-trained networks.

| Inversion Model | Train Model | DAFL | DI | Ours |
|-----------------|-------------|------|----|------|
| ResNet-34       | ResNet-18   | 32.40| 49.61| **74.55** |
|                 | VGG-11      | 15.85| 39.46| **67.51** |
|                 | MobileNetV2 | 33.49| 34.86| **64.34** |
| VGG-16          | ResNet-18   | 1.96 | 46.90| **72.38** |
|                 | VGG-11      | 1.90 | 41.94| **67.95** |
|                 | MobileNetV2 | 2.11 | 31.32| **62.86** |

Table 3: Effectiveness of DeepInversion and each component of our method. Our method outperforms DI in IS/FID scores and K.D with a small number of images (50k).

| O2O | FTP | ACS | K.D acc | IS | FID | Precision | Recall |
|-----|-----|-----|---------|----|-----|-----------|--------|
|✓   | ✓   | ✓   | 19.96%  | 4.29| 87.09| 0.7483    | 0.0018 |
|✓   | ✓   | ✓   | 38.61%  | 5.41| 63.36| **0.7829**| 0.1529 |
|✓   | ✓   | ✓   | 59.42%  | 6.31| 65.35| 0.7859    | 0.2845 |
|✓   | ✓   | ✓   | **60.57%**| 6.44| 65.94| 0.6692    | 0.2960 |

Bottom-Up Usage of Feature Map. To verify the effectiveness of our Feature Transfer Pyramid, we gradually increase feature maps combined to FTP. We observe how close the distribution of synthesized images is to the original distribution according to the number of feature maps. We continuously select high-level and low-level feature maps, then build up the FTP blocks. First, we synthesize 10k images from the ResNet-34 trained CIFAR-10 on each case. Then, we calculate FID scores using the embedding layer of the Inception-v3. As shown in Table 4, we demonstrate that the more we gradually stack FTP block, the lower the FID score. Since each block of FTP enhances the corresponding characteristics of the original target class, we synthesize the images closer to the original data.

Training Loss Curve. To test that our generative model with one-to-one approach helps to converge to a specific optimal point faster, we test training loss convergence of three cases: (a) optimization without non-linearity (b) non-linear optimization without one-to-one approach (c) our method. Fig.5 shows the training loss curve of $L_{CE}$ and $R_{BN}$ that play a key role in generating images in NaturalInversion. We simultaneously plot the training loss curve during 2k CIFAR-10 image inversion epoch. Because of the difficulty of optimizing the input space without non-linearity, DI converges slower than our method, and the final training loss is greater than ours for both $L_{CE}$ and $R_{BN}$. Moreover, using the generator to induce non-linearity helps the convergence by greater ability to represent the complex characteristics of the original distribution. In the end, we maximize the improvement in converge speed and training loss when using one-to-one generator, leading to eased optimizing process.

Application 1: Knowledge Distillation (KD) In this section, we ensure that we can transfer information from a teacher network to a student network without origi-
Accuracy (%)

**Table 5:** The results of data-free knowledge distillation on CIFAR-10/100 with synthesized images from various inversion methods. The bold type is the best value of the same architecture experiment, and the underbar type is the second-best value. T. and S. means the baseline accuracy of teachers and students trained on the original training set. *: our re-implementations.

**Application 2: Pruning**

We investigate that our method can improve the pruned model accuracy without real data. For the pruning, we utilized L1-norm pruning criteria that prunes a certain percentage of filters with smaller L1-norm (Liu et al. 2017). We carry out experiments on different pruning ratios from 50% to 90%, and all experiments are performed with ResNet-34. Our pruning setup is the same with Liu et al. (2018), and we locally prune the least important channels in each layer by the same pruning ratio. After pruning ResNet-34 on CIFAR-10/100, we fine-tune the pruned model using synthesized images from the baseline model for 20 epochs by SGD with 0.001 learning rate. As shown in Fig.6, DAFL has poor performance with 58.34% at a pruning ratio 90% on CIFAR-10 because it synthesizes images for a specific purpose, knowledge distillation. DI has low performance on CIFAR-100 because the more complexity the image, the more difficult it is to optimize without non-linear characteristics. In contrast, we achieve the best accuracy recovery of a pruned model up to 11.21% improvement on 90% pruning ratio due to the high fidelity and diversity images. This result ensure that our approach reflects the statistics of original data.

**Conclusion**

In this paper, we improve the quality of the synthesized image compared to conventional inversion methods with our proposed approach: NaturalInversion. First, we enhanced the characteristics of target class via **Feature Transfer Pyramid** by using multi-scale feature maps from classifier. Second, we used one-to-one generator for alleviating the mode collapse problem and bring the non-linearity. Lastly, we proposed **Adaptive Channel Scaling** parameters to implicitly learn the optimal channel scale range which has been learned by the classifier. Through extensive experiments, we demonstrated the effectiveness of NaturalInversion. Our methods not only capture the original data distribution but also are generalized, less biased to inversion model. We hope this work helps for the further progress in synthesizing realistic images in data-free conditions.
Acknowledgements

This work was supported by the National Research Council of Science & Technology (NST) grant by the Korea government (MSIT) [CRC-20-02-KIST], and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2021-0-00456, Development of Ultra-high Speech Quality Technology for Remote Multi-speaker Conference System).

References

Chen, H.; Wang, Y.; Xu, C.; Yang, Z.; Liu, C.; Shi, B.; Xu, C.; Xu, C.; and Tian, Q. 2019. Data-free learning of student networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 3514–3522.

Chu, X.; Zheng, A.; Zhang, X.; and Sun, J. 2020. Detection in crowded scenes: One proposal, multiple predictions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 12214–12223.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, 248–255. Ieee.

Fang, G.; Song, J.; Shen, C.; Wang, X.; Chen, D.; and Song, M. 2019. Data-free adversarial distillation. arXiv preprint arXiv:1912.11006.

Gatys, L.; Ecker, A. S.; and Bethge, M. 2015. Texture synthesis using convolutional neural networks. Advances in neural information processing systems, 28: 262–270.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2414–2423.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. Advances in neural information processing systems, 27.

Haroush, M.; Hubara, I.; Hoffer, E.; and Soudry, D. 2020. The knowledge within: Methods for data-free model compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8494–8502.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, 770–778.

Heitz, E.; Vanhoey, K.; Chambon, T.; and Belcour, L. 2021. A Sliced Wasserstein Loss for Neural Texture Synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9412–9420.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two-time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems, 30.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, 448–456. PMLR.

Islam, M. A.; Jia, S.; and Bruce, N. D. 2020. How much position information do convolutional neural networks encode? arXiv preprint arXiv:2001.08248.

Kalischek, N.; Wegner, J. D.; and Schindler, K. 2021. In the light of feature distributions: moment matching for Neural Style Transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9382–9391.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario.

Lin, T.; Ma, Z.; Li, F.; He, D.; Li, X.; Ding, E.; Wang, N.; Li, J.; and Gao, X. 2021. Drafting and Revision: Laplacian Pyramid Network for Fast High-Quality Artistic Style Transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5141–5150.

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision, 2736–2744.

Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; and Darrell, T. 2018. Rethinking the value of network pruning. arXiv preprint arXiv:1810.05270.

Mahendran, A.; and Vedaldi, A. 2015. Understanding deep image representations by inverting them. In Proceedings of the IEEE conference on computer vision and pattern recognition, 5188–5196.

Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

Mordvintsev, A.; Olah, C.; and Tyka, M. 2015. Deepdream—a code example for visualizing neural networks. Google Research, 2(5).

Olah, C.; Mordvintsev, A.; and Schubert, L. 2017. Feature visualization. Distill, 2(11): e7.

Sajjadi, M. S.; Bachem, O.; Lucic, M.; Bousquet, O.; and Gelly, S. 2018. Assessing generative models via precision and recall. arXiv preprint arXiv:1806.00035.

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X. 2016. Improved techniques for training gans. Advances in neural information processing systems, 29: 2234–2242.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 4510–4520.

Shocher, A.; Gandelsman, Y.; Mosseri, I.; Yarom, M.; Irani, M.; Freeman, W. T.; and Dekel, T. 2020. Semantic pyramid for image generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7457–7466.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2818–2826.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. Journal of machine learning research, 9(11).

Wang, P.; Li, Y.; Singh, K. K.; Lu, J.; and Vasconcelos, N. 2021. IMAGINE: Image Synthesis by Image-Guided Model Inversion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 3681–3690.

Xu, Y.; Shen, Y.; Zhu, J.; Yang, C.; and Zhou, B. 2021. Generative hierarchical features from synthesizing images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4432–4442.

Yin, H.; Molchanov, P.; Alvarez, J. M.; Li, Z.; Mallya, A.; Hoiem, D.; Jha, N. K.; and Kautz, J. 2020. Dreaming to distill: Data-free knowledge transfer via deep inversion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8715–8724.

Yosinski, J.; Clune, J.; Nguyen, A.; Fuchs, T.; and Lipson, H. 2015. Understanding neural networks through deep visualization. arXiv preprint arXiv:1506.06579.

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In European conference on computer vision, 818–833. Springer.

Zhai, X.; Kolesnikov, A.; Houlsby, N.; and Beyer, L. 2021. Scaling vision transformers. arXiv preprint arXiv:2106.04560.
Appendix

Generator Architecture for CIFAR-10/100

We use the same architecture on CIFAR-10/100 image generation, and generator architecture can be shown in Table.6. We use the latent space $z$ from conditional distribution given arbitrary label $\hat{y}$, so the $C$ means the number of classes. (For CIFAR-10 image synthesis, $C=10$, and $C=100$ for CIFAR-100). The $s,p$, and $c$ are the stride, padding of the convolution filter, and the number of channels. Finally, the generator synthesizes the CIFAR-10/100 images with a resolution of $32 \times 32$.

| Layer   | Detail                      | Output Size       |
|---------|-----------------------------|-------------------|
| input   | $z \in \mathbb{R}^{1024} \sim \mathcal{N}(0, 1)$ | $(1024 + C) \times 1$ |
| layer1  | Linear layer                | $128 \times 8 \times 8$ |
| Block 1 | LeakyReLU (0.2) | $128 \times 16 \times 16$ |
|         | Upsample $\times 2$         |                   |
| Block 2 | $3 \times 3$ Conv ($s=1, p=1$), LeakyReLU (0.2), Upsample $\times 2$ | $128 \times 32 \times 32$ |
| Block 3 | BN layer (c=64), LeakyReLU (0.2) | $64 \times 32 \times 32$ |
| output  | $3 \times 3$ Conv ($s=1, p=1$) | $3 \times 32 \times 32$ |

Table 6: Generator architecture of NaturalInversion for CIFAR-10/100 image synthesis.

Feature Transfer Pyramid Architecture

Our Feature Transfer Pyramid (FTP) operates in conjunction with a pre-trained classifier $T$. We usually use the ResNet-34 and VGG-11/16 as a pre-trained classifier. In these two classifiers, the downsampling point of the feature map is the same, so we set the feature extraction stage the same. Therefore, we consist of a total of four blocks of FTP, and details can be found in Table.7.

| Layer   | Detail                      | $n_t$ Size       |
|---------|-----------------------------|------------------|
| Block 1 | input $f_3$ ($512 \times 4 \times 4$), $1 \times 1$ Conv ($512 \rightarrow 256$) | $256 \times 8 \times 8$ |
| Block 2 | input $f_3$ ($256 \times 8 \times 8$), element-wise sum $f_3 + m_1$, $3 \times 3$ Conv, Upsample $\times 2$, $1 \times 1$ Conv ($256 \rightarrow 128$) | $128 \times 16 \times 16$ |
| Block 3 | input $f_2$ ($128 \times 16 \times 16$), element-wise sum $f_2 + m_2$, $3 \times 3$ Conv, Upsample $\times 2$, $1 \times 1$ Conv ($128 \rightarrow 64$) | $64 \times 32 \times 32$ |
| Block 4 | input $f_1$ ($64 \times 32 \times 32$), element-wise sum $f_1 + m_3$, $3 \times 3$ Conv, $1 \times 1$ Conv ($64 \rightarrow 3$), Tanh activation | $3 \times 32 \times 32$ |
| output  | input $G(z|y)$ ($3 \times 32 \times 32$), element-wise sum $G(z|y) + m_4$, $1 \times 1$ Conv ($3 \rightarrow 3$), Tanh activation | $3 \times 32 \times 32$ |

Table 7: Feature Transfer Pyramid architecture for CIFAR-10/100 image synthesis. Each $f_*$ means the feature map extracted from classifier and $m_t$ is each FTP block output.

256 and we synthesize the CIFAR-10/100 images for 2k, 4k inversion epochs, respectively.

Frechet Distance Evaluation

We show the “generality” of NaturalInversion by comparing Frechet distance on various models with prior works in Table.8. In general, Frechet Inception Distance (FID) is widely used to assess the image quality in generative models. Inception-v3 model trained on the ImageNet is commonly used as an embedding network. This study uses different type embedding networks trained on ImageNet and tests CIFAR-10/100 samples to evaluate that our method synthesizes high-quality images regardless of model type or pre-trained dataset domain. We reuse the images used in the generative model evaluation metric experiments in Section 4.1. When utilizing the model trained with ImageNet, NaturalInversion performs better with a more considerable difference than DI, and only 13.67% difference compared to WGAN-GP (baseline) trained with original data on CIFAR-10 experiments. We also choose the embedding networks trained with the CIFAR-10, the same domain we generate, so the overall FD score is low on embedding networks trained with CIFAR-10. Since NaturalInversion and DI synthesize images from ResNet-34 trained with CIRAR-10, FD score is better than WGAN-GP when using ResNet-34 as an embedding network. We achieve 3.22% better than WGAN-GP when using VGG-11. CIFAR-100 experiments also show similar trends in CIFAR-10 experiments. In conclusion, NaturalInversion approximates the original dataset distribution well regardless of the pre-trained classifier for inversion, and we ensure that our method synthesizes generalized images.
from scratch training

We expand the from scratch experiments in Section 4.1 to measure the CIFAR-10/100 top-1 test accuracy of the model trained with synthesized images in Table.9. Also, we calculate the training/test loss by forwarding the CIFAR-10/100 training/test sets to ensure that NaturalInversion smooths the optimization problem with non-linearity properties. For experiments, we train the randomly initialized networks from scratch using synthesized images for this study. Then forward original CIFAR-10/100 datasets and calculate the training/test accuracies and loss. Table.9 shows the result of training/test accuracies and loss, and we can demonstrate three properties: (a) Our method can capture original training data distribution because the CIFAR-10/100 training accuracy of the model is much higher than others. (b) NaturalInversion converges safely and quickly at optimal points, so the training loss is lower than DAFL and DI. (c) Our method can synthesize generalized images that are less biased to the inverted model, as seen from the test accuracy results. From the results, we can observe that NaturalInversion outperforms prior works on synthesizing more natural images.

Table 8: Frechet distance across the various backbone networks. Frechet distance result of CIFAR-10 images(top) and CIFAR-100 images(bottom)

| Feature Visualization on Various Embedding |
|-------------------------------------------|

We further expand the t-SNE result in feature visualization experiment in Section 4.1. For verifying the feature representation, we choose the feature embedding from each residual block output and embedding layer of ResNet-34. Fig.7 is a visualization of the feature representation of DI and NaturalInversion. Since DI optimizes the input space without non-linearity, it is difficult to distinguish the complex feature of the CIFAR-10 target classes. Therefore, the feature space shows a different distribution from the original distribution. In contrast, our NaturalInversion brings a non-linearity to optimization and causes ease of optimization step with a one-to-one approach generator. Therefore it leads to classify the characteristics of the original target class easily. As a result, our method can represent the original dataset more precisely.
| Inversion Model | Train Model | Train Acc. (Loss) | Test Acc. (Loss) |
|-----------------|-------------|------------------|------------------|
|                 |             | DAFL  | DI  | Ours | DAFL | DI  | Ours |
| **ResNet-34**   | ResNet-18   | 31.45 | 73.46 | **90.84** | 0.003 | 31.59 | 72.05 | **87.98** | 0.004 |
|                 | VGG-11      | 21.04 | 67.18 | **86.76** | 0.005 | 21.52 | 65.30 | **85.74** | 0.005 |
|                 | MobileNetV2 | 42.50 | 69.57 | **86.52** | 0.004 | 42.29 | 66.38 | **84.83** | 0.005 |
| **VGG-11**      | ResNet-18   | 46.98 | 80.21 | **88.13** | 0.004 | 46.08 | 78.16 | **85.53** | 0.005 |
|                 | VGG-11      | 32.38 | 78.94 | **86.76** | 0.004 | 32.52 | 76.31 | **83.92** | 0.006 |
|                 | MobileNetV2 | 44.17 | 73.82 | **84.03** | 0.005 | 43.52 | 72.02 | **81.40** | 0.006 |

Table 9: The original train/test accuracy and loss of model trained with synthesized CIFAR-10/100 images from scratch. The bold type is the best accuracy of the same architecture experiment, and the underbar type is the lowest loss.