Improving Self-supervised Learning in Computer Vision with Generative Models

Zhen Liu Yutong Yan

Abstract

We investigate the possibility of leveraging generative models to improve the performance of self-supervised training. With the observation that convolutional structures in both encoders and decoders can capture priors in natural images, we speculate that a pretrained generative models on the same or a similar dataset can improve both the accuracy of self-supervisedly trained models and the convergence of self-supervised training. We experimented with two major approaches: 1) use images generated by pretrained generative models and 2) use feature maps from the discriminator/energy-based model of pretrained generative model. We empirically found that while it can be hard for image augmentations to work in general, BYOL can benefit from augmentations with either forms. The codes can be found in https://github.com/yutongyan/aug_data_gen_ssl.

1. Introduction

To fully leverage the huge amount of data collected from various sources requires unsupervised representation learning (Hinton et al., 1999). Recent papers prove that self-supervised learning, which exploits supervised signals through pretext tasks (Jing and Tian, 2020), is a promising method to bridge the gap between unsupervised and supervised performance in computer vision (Grill et al., 2020; Chen et al., 2020b;a; Caron et al., 2020).

Different from other unsupervised methods that explicitly require a generator/decoder like VAE (Kingma and Welling, 2013) and GAN (Radford et al., 2016), self-supervised learning typically only needs to train encoder CNNs due to the nature of pretext tasks. However, there are evidences that image generators may contain useful priors on natural images. One notable results from the paper Deep Image Prior (Ulyanov et al., 2018) show that even randomly initialized CNNs (to be more precise, convolutional and deconvolutional nets) contain useful structural priors and can be used for semantic reconstruction. More specifically, consider the downsteam task of image inpainting where the goal is to reconstruct x_{gt} from an image x_0 which is corrupted by masking a square region M out. If we randomly initialize a decoder network $x = f(z)$ and optimize the latent code z with some loss (e.g. L2 loss) $L(M \cdot x + x_{gt}, x_0)$, as long as the network is capable enough we may get an image $\hat{x} = M \cdot x + x_{gt}$ that is close to x_0. However, if we stop the optimization process early, $\hat{x}$ can look very similar to x_{gt}, assuming that the corrupted region is highly correlated with the unmasked region. This phenomenon suggests "Deep Image Prior" does exist and priors of natural image distributions can be captured even with random decoders.

As self-supervised learning is to distinguish positive and negative examples explicitly or implicitly, the structural knowledge learned from reconstruction is never used. We hypothesize that a pretrained generative model contains such knowledge and can be used to improve the performance of self-supervised learning and experiment with possible ways of leveraging pretrained generative models in self-supervised learning on images.

Our findings are summarized below:

- Directly enlarge datasets with images can be hard due to the poor quality of generated images from class-unconditional generative models.
- Auxiliary views with output feature maps of energy-based model (somewhat equivalently, discriminators in GANs) is more efficient than those with generated images.
- BYOL-like methods where only positive examples are used can benefit from additional views provided by generative models, especially when the batch size is small.

2. Related Works

Data Augmentation. Different data augmentation methods are extensively to improve model performance in both supervised and unsupervised learning (Oord et al., 2018; Zhang et al., 2016; Noroozi and Favaro, 2016; Asano

et al., 2019). Traditional augmentation techniques in computer vision include but are not limited to image flipping, grayscaling and random scaling and cropping (He et al., 2015), while there exist some more recent ones like variants of Mixup (Zhang et al., 2017; Verma et al., 2019), which augment the dataset by doing linear interpolation in pixel or manifold space. It is also possible to do data augmentation with generative models (Sandfort et al., 2019).

Prior of CNNs. Convolutional neural net (CNN) is known to capture the priors in nature images and generalize well in many downstream tasks. One example for such prior is CNN's capability of measuring the perceptual distance between images (Zhang et al., 2018). Similarly, (Ulyanov et al., 2018) shows that the structure of a generator network is sufficient to capture a great deal of low-level image statistics without any learning, bridging the gap between learning-based methods and learning-free methods based on handcrafted image priors.

Combing self-supervised learning and generative learning. To our knowledge, there is no previous work that tries to use generative models for self-supervised learning. However, some papers have done the reverse direction, i.e. to improve the performance of generative models with self-supervised learning. For instance, in addition to the standard GAN loss (predicting real/fake images), (Chen et al., 2019) adds an auxiliary loss on predicting the rotation of images.

3. Method

3.1. Basic self-supervised methods

For completeness, we present two baseline methods we use in our project: BYOL and SimCLR.

Given a set of images $\mathcal{D}$, an image $x \sim \mathcal{D}$ sampled uniformly from $\mathcal{D}$ and two distributions of image augmentations $\mathcal{T}$ and $\mathcal{T}'$, two augmented views $v := t(x)$ and $v' := t'(x)$ from x by applying respectively image augmentations $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$. For the first augmented view, a representation $y_\theta := f_\theta(v)$ and a projection $z_\theta := g_\theta(y)$, parametrized by weights θ. The target representation and projection are $y'_\xi := f_\xi(v')$ and $z'_\xi := g_\xi(y')$, parametrized by weights ξ. The mean squared error between the normalized predictions and target projections is defined as:

$$\mathcal{L}_{\theta,\xi} := 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{||q_\theta(z_\theta)||_2 \cdot ||z'_\xi||_2}, \tag{1}$$

where $q_\theta(z_\theta)$ is a prediction of z'_ξ. The total loss of BYOL has an additional part $\tilde{\mathcal{L}}_{\theta,\xi}$, which is computed by separately feeding v' as first view and v' as second view, a symmetry of $\mathcal{L}_{\theta,\xi}$,

$$\mathcal{L}^{\text{BYOL}}_{\theta,\xi} = \mathcal{L}_{\theta,\xi} + \tilde{\mathcal{L}}_{\theta,\xi}, \tag{2}$$

where we only optimize with respect to θ and ξ is updated using exponential average of θ.

In SimCLR, given an image x, similarly, two views v and v' are computed in the same way as in BYOL. A base encoder network $k_\theta(\cdot)$ and a projection head $m_\theta(\text{cot})$ are used to extract representation vectors $h := k_\theta(v)$, $h' := k_\theta(v')$ and map representations to the contrastive loss space, $z := m_\theta(h)$, $z' := m_\theta(h)$, respectively. The loss is defined as:

$$\mathcal{L}^{\text{SimCLR}}_\theta = -\log \frac{\exp(\text{sim}(z, z')/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[z \neq z_k]} \exp(\text{sim}(z, z_k)/\tau)}, \tag{3}$$

where $\mathbb{1}_{[z \neq z_k]} \in \{0, 1\}$ is an indicator function evaluating to 1 if $z \neq z_k$, τ denotes a temperature parameter, $\text{sim}(u, v) = \frac{u^\top v}{||u||||v||}$ denotes the dot product betweeen l_2 normalized u and v.

3.2. Augmentation with images

Our first idea is to augment self-supervised learning with images from generative models. More speficially, we hypothesize that a well-trained generative model can be used to efficiently sample images that are semantically similar but visually similar, and those generated images can be better augmentations than simple operations like rotation and grayscaling commonly used in self-supervised learning.

3.2.1. Image generation methods

In theory we can use any generative model that allows conditional inference. The simplest method is to perturb the latent embedding of an image in latent space and generate an augmented image using a trained variational encoder (VAE) (Kingma and Welling, 2013). However, it is hard (and we empirically observed) for class-unconditional models to create meaningful augmented images (in the sense that images should be visually dissimilar to some degree) by random perturbations. Therefore, for the VAE approach, we instead do linear interpolation in latent space between ground truth images to create new examples:

$$\tilde{x} = g(\theta f(x_1) + (1 - \theta) f(x_2)) \tag{4}$$

, where f, g are the encoder and the decoder respectively. We pick $\theta = 0.5$ so that it maintains maximal distance from both ground truth images in latent space. This is similar to the approach used in Mixup augmentation (Zhang et al., 2017) except that it does interpolation in pixel space.

For better augmentation quality, we instead train an energy-based model and sample from it using Langevin dynamics

(Du and Mordatch, 2020; Dai et al., 2019). The benefit for doing MCMC sampling on energy-based model is that the transition from random noises to well-generated images can be explicitly controlled by the the number of steps and step size without looking into the latent embeddings of images - if two images are generated from the same image with a few number of steps, they tend to look more similar.

In this paper we follow the image generation method with Langevin dynamics in (Du and Mordatch, 2020). Given a datapoint x, let $E_\theta(x) \in \mathbb{R}$ be the energy function, parameterized by weights θ. The energy function defines a probabiliuty distribution via the Boltzmann distribution $p_\theta(x) = \frac{\exp(-f_\theta(x))}{Z(\theta)}$, where $Z(\theta)$ is the partition function. This EBM is trained with standard contrastive loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_{pos} \sim p_D}[f_\theta(x_{pos})] - \mathbb{E}_{x_{neg} \sim p_\theta}[f_\theta(x_{neg})] \quad (5)$$

where $x_{neg} = \tilde{x}^{(T)}$ is sampled using Langevin dynamics:

$$\tilde{x}^{(t)} = \tilde{x}^{(t-1)} - \frac{\lambda}{2}\nabla_x f_\theta(\tilde{x}^{t-1}) + \omega^k, \quad \omega^k \sim \mathcal{N}(0, \lambda), \quad (6)$$

The start of the Markov chain $x^{(0)}$ can either be random noise or some simple generative model like a flow model (Dai et al., 2019). During training, however, we cannot simulate a long Markov chain and do backprop through every single step of it. Therefore, during training we follow (Du and Mordatch, 2020) and obtain $x^{(0)}$ from a replay buffer (with a small probability to re-initialize samples) so that a longer Markov chain is implicitly implemented. During inference, since no backprop is involved, we may simply run the T-step Markov chain iteratively to obtain a generated image. An example of the MCMC chain on EBM is shown in Figure 1.

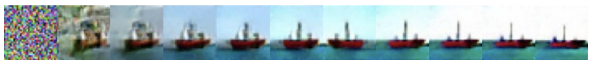

Figure 1: Example of MCMC chain with Langevin dynamics on EBM. The image on the left end is generated by uniform random noise on pixel space. As we run more Langevin steps, the generated image gradually transform to the images to the right end of the sequence.

To obtain a semantically similar image, the theoretically ideal solution is to run an inverse Langevin dynamics (probably noise-free one) to get a 'noisy' image and run forward Langevin dynamics. However, we empirically observed that there are too many artifacts (as those can be commonly seen in adversarial examples) using this approach. Therefore, we instead generate this noisy image by linear interpolating between the source image and a random image (sampled from uniform distribution on $[0, 1]$ in pixel space) and then run forward Langevin dynamics on this image.

An alternative to get the noisy image is to mask out a region in an image and do image inpainting with EBM. However, we found out that the image quality is more sensetive to the choice of hyperparameters and can be significantly worse in many cases. Therefore we do not include the results from this method but leave it to future work.

3.2.2. Direct dataset augmentation

The simplest approach to leverage the generated images is to directly append them to the dataset. Such method will be best suited for methods like BYOL because false-negative image pairs (if the images are too visually similar) will decrease the performance as shown in (Cai et al., 2020).

3.2.3. Auxiliary view from generated images

Instead of directly enlarging the dataset, we may treat the generated images as an auxiliary view of the source images. Specifically, given an image x, we first generate an auxiliary sample $\tilde{x}$, using the method described in Section 3.2.1. Then, we apply the same transformation as the second view $\tilde{v} := t'(\tilde{x})$, and output $\tilde{z}_\xi$ and $\tilde{z}$, respectively in BYOL and SimCLR, using the same model as for the second view. The new loss is then defined as:

$$\mathcal{L} = \mathcal{L}^{\text{con}}(v, v') + \lambda \mathcal{L}^{\text{con}}(v, \tilde{v}), \quad (7)$$

where $\mathcal{L}^{\text{con}}$ can be chosen as $\mathcal{L}^{\text{BYOL}}_{\theta,\xi}$ or $\mathcal{L}^{\text{SimCLR}}_{\theta}$, and λ denotes the parameter to control the auxiliary contrastive loss $\mathcal{L}^{\text{con}}(v, \tilde{v})$. We abuse the notatio n as taking the two views as input to highlight the difference between the first and second term in $\mathcal{L}$.

3.3. Augmentation with feature maps

In this section, similarily as the auxiliary view method, we will also add an auxiliary contrastive loss, in addition to the contrastive loss used by the baseline methods. However, instead of using the same model as the second view, here we use a pretrained unsupervised encoder, with fixed weights. Given an image x, the pretrained unsupervised encoder outputs the representation vector $\tilde{z} := m(x)$. The new loss for BYOL is then defined as:

$$\mathcal{L} = \mathcal{L}^{\text{BYOL}}(q_\theta(z_\theta), z'_\xi) + \lambda \mathcal{L}^{\text{BYOL}}(q_{\theta'}(z_\theta), \tilde{z}), \quad (8)$$

where λ denotes the parameter to control the auxiliary contrastive loss $\mathcal{L}^{\text{BYOL}}(z, \tilde{z})$. Notice that we use a different

prediction head $q_{\theta'}$) for the auxiliary view. Similarly, the new loss for SimCLR is defined as:

$$\mathcal{L} = \mathcal{L}^{\text{SimCLR}}(q(z), q(z')) + \lambda \mathcal{L}^{\text{SimCLR}}(q'(z), q'(\tilde{z})), \quad (9)$$

where λ denotes the parameter to control the auxiliary contrastive loss $\mathcal{L}^{\text{SimCLR}}(q'(z), q'(\tilde{z}))$, where q' is a different prediction head used for the auxiliary view. Note that we again abuse the notation as the loss taking input the projection vector, instead of the view as in Section 3.2.3, to highlight the difference between the original loss and the newly added auxiliary loss.

4. Experiments

For the EBM model, we use a model trained on CIFAR-10 with the same architecture (shown in Fig 2) and hyperparameters as in (Du and Mordatch, 2020). During image generation, we use the following as the default set of hyperparameters:

- Step size: 10.0
- Number of MCMC steps: 100
- Standard deviation scale of Langevin dynamics noise (with base being 0.005): 1.0
- Coefficient for linear interpolation (between natural image and noise): 0.6

For the coefficient of the auxiliary loss, we use $\beta = 0.01$ as the default coefficient. We use batch size 512 for SimClR and batch size 256 for BYOL, due to memory constraint on our available computational resources. For the rest of the hyperparameters and settings, We follow the settings in (Ermolov et al., 2020): we set the embedding size to 64, learning rate to 3×10^{-3} and train models for 1000 epochs. We use a simple two layer MLP for the projection heads.

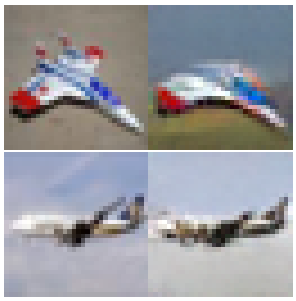

Figure 3: Examples of data augmentation done via EBM. Images on the left are the original ones in CIFAR-10 dataset while the ones on the right are the corresponding augmented images.

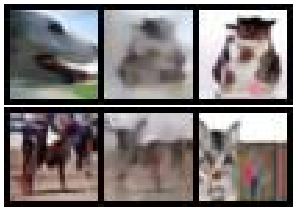

Figure 4: Examples of data augmentation done via VAE. Images on the left and right are the original ones in CIFAR-10 dataset while the ones in the middle are the corresponding interpolated images.

3x3 conv2d, 128
ResBlock down 128
ResBlock 128
ResBlock 128
ResBlock down 256
ResBlock 256
ResBlock 256
ResBlock down 256
ResBlock 256
ResBlock 256
Global Sum Pooling
dense $\rightarrow$ 1

Figure 2: Architecture for the EBM model.

4.1. Direct dataset augmentation

For direct dataset augmentation with EBM, we generate 8 augmented random images for each ground truth image in the dataset. For VAE, we randomly sample 60000 pairs of images (non-repetitive) in the dataset and create images

with linear interpolation (with coefficient being 0.5) in latent space. The results are shown in Table 1. The baseline behaves much better than the augmented ones.

Method	Baseline	VAE-interpolation	EBM
SimCLR	**91.81**	88.60	87.53
BYOL (bs=256)	**90.20**	89.13	85.09

Table 1: Direct dataset augmentation (%).

4.2. View augmentation with images

The results of ablation study are shown in the following tables. While the augmentation fails on SimCLR, its performance is comparable with the baseline in BYOL. We do not find significant dependency on the choice of hyperparameters.

	$N = 25$	$N = 100$	$N = 225$	$N = 400$	Baseline
SimCLR	88.78	89.61	89.04	89.68	**91.81**
BYOL (bs=256)	89.68	**90.57**	90.14	89.12	90.20

Table 2: Number of Langevin dynamics steps, CIFAR-10 (%).

	$\sigma = 0.5$	$\sigma = 1.0$	$\sigma = 2.0$	Baseline
SimCLR	89.60	89.04	88.90	**91.81**
BYOL (bs=256)	89.96	**90.57**	90.01	90.20

Table 3: Standard deviation scale, CIFAR-10 (%).

	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	Baseline
SimCLR EBM	89.39	89.04	89.43	**91.81**
BYOL EBM (bs=256)	90.55	**90.57**	90.53	90.20

Table 4: Coefficient for noise interpolation, CIFAR-10 (%).

	$N = 25$	$N = 100$	$N = 225$	$N = 400$	Baseline
SimCLR EBM	61.97	62.31	63.43	64.47	**69.52**
BYOL EBM (bs=256)	64.70	64.18	65.52	65.66	**69.41**

Table 5: Number of Langevin dynamics steps, CIFAR-100 (%).

	$\sigma = 0.5$	$\sigma = 1.0$	$\sigma = 2.0$	Baseline
SimCLR EBM	62.93	63.43	62.05	**69.52**
BYOL EBM (bs=256)	65.65	65.52	64.23	**69.41**

Table 6: Standard deviation scale, CIFAR-100 (%).

	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	Baseline
SimCLR EBM	63.70	63.43	63.57	**69.52**
BYOL EBM (bs=256)	65.57	65.52	64.74	**69.41**

Table 7: Coefficient for noise interpolation, CIFAR-100 (%).

4.3. View augmentation with feature maps

We take the output feature of the fully connected layer of the pretrained EBM before the final classifier as the auxiliary view. The accuracies are shown in Table 8 and the convergence plot of our best performing model is shown in Fig 5. While our model is marginally worse than the baseline in SimCLR, our best performing model behaves much better in BYOL with small batch size. It is also notable that the convergence of our model is faster in the first few epoches for BYOL. The results may suggest that BYOL-like methods (where only positive examples are supplied) is highly unstable and an additional view from a pretrained model can help in such case as some sort of directed noise.

	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1.0$	Baseline
SimCLR EBM	91.69	91.66	91.70	**91.81**
BYOL EBM (bs=256)	91.66	**92.13**	91.50	90.20

Table 8: Coefficient for auxiliary loss, CIFAR-10 (%).

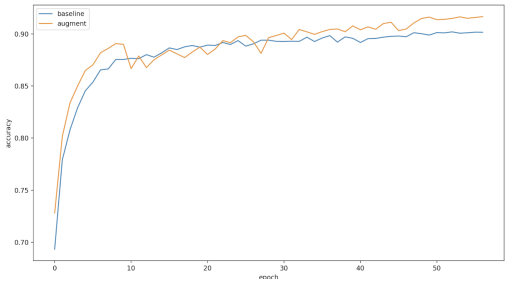

Figure 5: Convergence comparison between our best performance model (BYOL, bs=256) and the baseline on CIFAR-10.

5. Discussions

Why augmentations with images do not work? We notice that the generated images from class-unconditional generative models can contain many artifacts compared to the class-conditional counterparts, which implies that the artifacts are more likely a mixture of parts in different image categories. The introduction of the mixtures of parts can be detrimental for learning an accurate classifier. Also, the quality of generated images is hard to control. Wheter a better generative model helps remains to be studied in future works.

Why BYOL benefits from additional views? BYOL relies on the mean teacher (implemented using exponential mean averaging) (Shi et al., 2020). With deep models the dynamics can be highly unstable especially with small batch sizes. An additional view from a pretrained generative model can act as an anchor for these embeddings so that 1) the initialization becomes easier and 2) a directed noise is added to encourage the model to explore more generalizable features. We also observe that with larger batch size the gain from generative models vanishes, which deserves future investigation. It is also worth checking whether an ensemble of generative models produces better guidance for BYOL-like methods.

6. Conclusion

We explore some possibilities of leveraging generative models for self-supervised learning on images. Although the results are mixed, we observe interesting behaviors of augmentations for BYOL-like methods. We believe that further research on why these behaviors appear can be a promising direction. Besides, as we do not use a huge generative model (e.g. scale comparable to BigGAN (Brock et al., 2019)) in our studies due to constraints on computational resources, it remains to see if a good generative model is all we need for fixing the issues we encountered.

References

Asano, Y. M., Rupprecht, C., and Vedaldi, A. (2019). A critical analysis of self-supervision, or what we can learn from a single image. *arXiv preprint arXiv:1904.13132*. 1

Brock, A., Donahue, J., and Simonyan, K. (2019). Large scale gan training for high fidelity natural image synthesis. 6

Cai, T. T., Frankle, J., Schwab, D. J., and Morcos, A. S. (2020). Are all negatives created equal in contrastive instance discrimination? 3

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33. 1

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. (2020a). Big self-supervised models are strong semi-supervised learners. *Advances in Neural Information Processing Systems*, 33. 1

Chen, T., Zhai, X., Ritter, M., Lucic, M., and Houlsby, N. (2019). Self-supervised gans via auxiliary rotation loss. 2

Chen, X., Fan, H., Girshick, R., and He, K. (2020b). Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*. 1

Dai, B., Liu, Z., Dai, H., He, N., Gretton, A., Song, L., and Schuurmans, D. (2019). Exponential family estimation via adversarial dynamics embedding. 3

Du, Y. and Mordatch, I. (2020). Implicit generation and generalization in energy-based models. 3, 4

Ermolov, A., Siarohin, A., Sangineto, E., and Sebe, N. (2020). Whitening for self-supervised representation learning. 4

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33. 1

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. 2

Hinton, G. E., Sejnowski, T. J., Poggio, T. A., et al. (1999). *Unsupervised learning: foundations of neural computation*. MIT press. 1

Jing, L. and Tian, Y. (2020). Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. 1, 2

Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer. 1

Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*. 1

Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. 1

Sandfort, V., Yan, K., Pickhardt, P. J., and Summers, R. M. (2019). Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks. *Scientific reports*, 9(1):1–9. 2

Shi, H., Luo, D., Tang, S., Wang, J., and Zhuang, Y. (2020). Run away from your teacher: Understanding byol by a novel self-supervised approach. 5

Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2018). Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454. 1, 2

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., and Bengio, Y. (2019). Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR. 2

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*. 2

Zhang, R., Isola, P., and Efros, A. A. (2016). Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer. 1

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595. 2