EFFICIENT FINE-TUNING OF NEURAL NETWORKS FOR ARTIFACT REMOVAL IN DEEP LEARNING FOR INVERSE IMAGING PROBLEMS

Alice Lucas[1]*, Santiago Lopez-Tapia*[2]*, Rafael Molina*[2]*, Aggelos K. Katsaggelos*[1]

[1]Dept. of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA

[2]Computer Science and Artificial Intelligence Department, Universidad de Granada, Spain

ABSTRACT

While Deep Neural Networks trained for solving inverse imaging problems (such as super-resolution, denoising, or inpainting tasks) regularly achieve new state-of-the-art restoration performance, this increase in performance is often accompanied with undesired artifacts generated in their solution. These artifacts are usually specific to the type of neural network architecture, training, or test input image used for the inverse imaging problem at hand. In this paper, we propose a fast, efficient post-processing method for reducing these artifacts. Given a test input image and its known image formation model, we fine-tune the parameters of the trained network and iteratively update them using a data consistency loss. We show that in addition to being efficient and applicable to large variety of problems, our post-processing through fine-tuning approach enhances the solution originally provided by the neural network by maintaining its restoration quality while reducing the observed artifacts, as measured qualitatively and quantitatively.

Index Terms— Deep Neural Networks, Image and Video Processing, Inversion, Fine-tuning, Artifacts, Data Consistency

1. INTRODUCTION

In the past decade, the application of Deep Neural Networks (DNNs) to inverse imaging problems has gained a considerable amount of popularity [1]. This approach requires the training of a neural network $\mathrm{f}_\theta(\cdot)$ with parameters θ to learn the mapping between any observed test image y to its restored version x by completing a forward pass: $\hat{\mathrm{x}} = \mathrm{f}_\theta(\mathrm{y})$. In the non-blind case, the observed image y is assumed to come from a known image formation model with degradation operator A, which we formulate here as $\mathrm{y} = \mathrm{Ax}$. The parameters θ are learned through a lengthy training stage which requires the use of a large dataset with input-output (y, x) pairs. The training data is commonly obtained by applying the degradation operator A to the clean images to obtain the corresponding degraded images used for training. With this straight-forward framework combined with the fast-growing nature of Deep Learning, new state-of-the-art results for image restoration tasks are regularly achieved.

However, what the Deep Learning community fails to expose in their publications are the failure cases of using DNNs for inverse imaging problems which may result in unnatural-looking images with unpleasant artifacts. Indeed, under certain conditions, these networks may produce images of unsatisfactory quality that is not up to the established standards. To illustrate this, we summarize below two case scenarios in which such a situation may arise.

Case scenario #1: Disagreements between the training and testing image formation model in super-resolution. It is well known that a DNN trained for a specific image recovery problem will not generalize well to a test data point whose image formation model differs from the model established when synthesizing the training dataset. In the case of super-resolution (SR), for example, providing a test image downsampled by 3 to a neural network trained for performing the SR problem of scale factor 4 will result in the introduction of mild artifacts in the restored image. We show an example of such artifacts in Figure 1b. While we observe a sharp increase in resolution when compared with the bicubic interpolation solution shown in Figure 1a, the result in Figure 1b carries an artificial quality, expressed by an un-natural looking super-resolved texture.

Case scenario #2: Artifacts resulting from the training of Generative Adversarial Networks. The application of the Generative Adversarial Network (GAN) [2] formulation to inverse imaging problems has resulted in solutions of previously unseen restoration quality ([3, 4, 5]). However, as a result of their challenging training dynamics, GANs frequently generate undesired artifacts in their produced images. An example of these artifacts is presented in Figure 2b, which shows a super-resolved frame obtained by the GAN-based model presented in [5] and [6], trained for the video SR problem for scale factor 4. It is clear from this figure that a small amount of noise was introduced by the GAN into the super-resolved frame, which the authors of [5] refer to as a "dot-pattern" artifact.

The standard approach to resolve the two case scenarios described above consists of returning to a training procedure and adapting the training set-up in an attempt to minimize, or completely remove, these artifacts. For example a solution to case scenario #1 is to modify the training dataset such that multiple degradation operators of different scale factors are used to fabricate the training data. As a result, the neural network becomes robust to several degradation operators A and is thus less prone to generate artifacts. To resolve case scenario #2, researchers may want to experiment with the use of improved loss functions during training, such as the use of WGANs [7], LSGANs [8], and cycle-consistent GANs [9], with the objective of regularizing the produced artifacts.

While these approaches are perfectly sound course of actions, they possess the significant drawback of being excessively time-consuming. Modifying the training procedures of neural networks may take weeks, sometimes months, of research to converge to a satisfying product. In many settings, such as industry, this approach would not be an effective use of time. This approach also requires access to large computing power and training dataset, which a user may not necessarily have access to at test time.

In this work, we provide an efficient approach to post-process an image by fine-tuning the parameters θ of the neural network $\mathrm{f}_\theta(\cdot)$

This work was supported in part by the Sony 2016 Research Award Program Research Project and by the National Science Foundation under grant DGE-1450006. The work of SLT and RM was supported by the the Spanish Ministry of Economy and Competitiveness through project DPI2016-77869-C2-2-R and the Visiting Scholar program at the University of Granada. SLT received financial support through the Spanish FPU program.

without the use of a training dataset. The supervision signal used for fine-tuning the parameters corresponds to the constraint that our observed test image y should be related to the output of the DNN through the known image formation model: $\mathrm{Af}_\theta(\mathrm{y}) = \mathrm{y}$. With this approach, noise patterns and artifacts that are not in agreement with the observed data will be penalized accordingly during fine-tuning. Thus as the parameters θ of the DNN are updated to satisfy the data consistency constraint, they are displaced in parameter space towards generating a solution with less artifacts.

2. RELATED WORKS

Guiding the neural network's solution to satisfy the data consistency constraint $\mathrm{Af}_\theta(y) = \mathrm{y}$ for improved results has become an increasingly popular trick in the literature of deep learning for inverse imaging problems. The constraint is in most cases implemented as a non-trainable projection layer in the architecture of a neural network (see for example, [10, 11, 12]). These layers naturally map the output image of the DNNs onto the set of data consistent solutions.

A recent approach in the image processing literature has been to fully depart from the definition of DNNs as learned mappings from degraded to restored images, and instead use un-trained DNNs as regularizers when inverting an observed image y using an analytical framework. A pioneer of this approach is the Deep Image Prior (DIP) [13] method, which defines the minimization problem $\hat{\theta} = \arg\min_\theta \; ||\mathrm{Ah}_\theta(\mathrm{z}) - \mathrm{y}||^2$ using a randomly initialized neural network $\mathrm{h}_\theta(\cdot)$, a fixed random input vector z, and the degradation operator A. Through the use of an iterative gradient descent scheme, the algorithm eventually converges to a satisfactory restored image: $\hat{\mathrm{x}} = \mathrm{h}_\theta(\mathrm{y})$. The supervision used during "training" is solely given by the data consistency constraint. The authors of DIP claim that the architecture of $\mathrm{h}_\theta(\cdot)$ is a strong enough regularizer to produce pleasing images without necessitating further regularization. Indeed, the obtained solutions of [13] are surprisingly well regularized and similar to natural images.

In this work, we consider the case in which we are provided with a deep neural network $\mathrm{f}_\theta(\cdot)$, trained for a specific image restoration task on a large dataset of input-output pairs. We focus on the case in which the network provides an overall suitable solution $\hat{\mathrm{x}}$, with the exception of generating mild artifacts in its output. In Section 3, we describe in detail our formulation of fine-tuning the parameters of the neural network as an inversion problem at test time, without requiring a training data set. We apply our method on the two case scenarios defined in Section 1 and show that we can successfully attenuate the artifacts while keeping the restoration quality originally produced by $\mathrm{f}_\theta(\cdot)$. We end this paper with a discussion in Section 4, which details the benefits and limitations of our proposed method. To the best of our knowledge, we are the first to establish a framework for fine-tuning neural networks to enhance output images at test time.

3. METHODS

We assume access to a network $\mathrm{f}_\theta(\cdot)$ trained on a large dataset (X, Y) with an objective function $\mathcal{C}$. As a result of training, we obtain the parameters θ by minimizing $\hat{\theta} = \arg\min_\theta \; \mathbb{E}_{X,Y}\, \mathcal{C}(\mathrm{x}, \mathrm{y}, \theta)$ where $\mathrm{x} \sim X$ and $\mathrm{y} \sim Y$ are sampled from the training dataset. We wish to preserve the original restoration quality provided by $\mathrm{f}_\theta(\cdot)$ whilst removing the generated artifacts. To achieve that, we propose to fine-tune the parameters of our neural networks by iteratively updating them in the direction which minimizes the generated artifacts. We eliminate the requirement of re-training the neural network and instead formulate a supervision signal at test time which only requires the observed image y, its degradation operator A, the trained θ weights and the network architecture $\mathrm{f}_\theta(\cdot)$. Because artifacts seen in $\mathrm{A}\hat{\mathrm{x}}$ are not observed in the y image, a supervision signal suitable for our task penalizes contents of $\mathrm{A}\hat{\mathrm{x}}$ and y that are dissimilar from each other:

$$\hat{\psi} = \arg\min_\psi \; ||\mathrm{Af}_\psi(\mathrm{y}) - \mathrm{y}||_2^2. \tag{1}$$

We solve the problem posed in Equation 1 using gradient descent by efficiently computing the derivatives of f_ψ with automatic differentiation in the Pytorch [14]. Our first iteration sets the initial parameters to those obtained from the earlier training stage, i.e.: $\psi_0 = \theta$. Given the fine-tuned final parameters ψ, our final post-processed image is obtained through the mapping: $\mathrm{x} = \mathrm{f}_\psi(\mathrm{y})$.

We note here that our formulation in Equation 1 corresponds to the well established inversion problem posed by all image recovery tasks, which we apply here to the context of fine-tuning and output enhancement of an already trained network. While this supervision signal is very similar to the one used by the Deep Image Prior approach [13], our motivations for using it are distinct. We formulate the inversion problem as in Equation 1 not to restore an image from scratch as in DIP, but to enhance the solution supplied by a trained neural network. Our work thus combines the learning power of pre-trained DNNs with the dependability of analytical approaches for image restoration.

In the next section, we apply our fine-tuning in Equation 1 in the context of the two case scenarios previously reported. We obtain our pre-trained neural networks $f_\theta(\cdot)$ from [5] which were both trained for the task of video super-resolution (VSR) for scale factor 4. In this paper, we refer to these two models as the VSRMSE model, trained with the Mean-Squared-Error loss $\mathcal{C} = ||\mathrm{f}_\theta(\mathrm{y}) - \mathrm{x}||_2^2$, and the VSRGAN model trained by introducing a discriminator network g_ϕ with trainable parameters ϕ and using the cost function $\mathcal{C} = \log(1 - \mathrm{g}_\phi(\mathrm{f}_\theta(\mathrm{y}))$. Both the VSRMSE and VSRGAN architectures correspond to a Convolutional Neural Networks (CNNs) with 15 residual blocks (See Figure 1 in [5]). The input to both networks is a low-resolution video sequence $\mathrm{y} = \{\mathrm{y_{t-2}}, \mathrm{y_{t-1}}, \mathrm{y_t}, \mathrm{y_{t+1}}, \mathrm{y_{t+2}}\}$ and the output is the corresponding center high-resolution frame $\mathrm{x} = \mathrm{x_t}$.

3.1. Disagreements between the training and testing image formation in super-resolution.

The VSRMSE [5] is pre-trained to perform super-resolution for scale factor 4. It is thus reasonable to expect that at test time, inputting a low-resolution video sequence that was downsampled by an SR factor other than 4 will result in VSRMSE failing to produce clean super-resolved frames. Figure 1b shows the result of downsampling the test video frames by 3 instead of 4 and super-resolving these with the trained VSRMSE. A comparison between the bicubic image in Figure 1a and the SR result $\hat{\mathrm{x}}$ in Figure 1b reveals that while the VSRMSE was not trained for the VSR task of factor 3, it is remarkably successful at increasing its resolution. Unfortunately, this increase in resolution is also accompanied with textural artifacts, particularly around the branches of the trees. Our main objective is to reduce the generated artifacts observed in Figure 1b while keeping the obtained increase in resolution provided by VSRMSE. To this end, we apply the formulation introduced in Equation 1, where y is our video sequence downsampled by scale factor 3, A is the bicubic down-

(a) 16.90/0.6630 **(b)** 23.51/0.7230 **(c) 25.60/0.7497** **(d)** 21.72/0.5162

Fig. 1: Super-resolving a sequence downsampled by factor 3. (a) Bicubic interpolation; (b) VSRMSE output ([5]) with disagreeing downsampling factors; (c) Our fine-tuned VSRMSE output; and (d) DIP [13].

sampling operator for scale factor 3 and the f_θ network is the trained VSRMSE network for scale factor 4. Provided with our new parameters ψ as a result of the fine-tuning method, we generate a new solution $\hat{\mathrm{x}} = f_\psi(\mathrm{y})$ and display it in Figure 1d. From this figure we observe that our fine-tuning method successfully attenuates the ringing artifacts while maintaining the initially acquired super-resolved quality of Figure 1b.

To demonstrate that we truly benefit from fine-tuning the pre-trained parameters θ as opposed to optimizing with initial random parameters as in the Deep Image Prior [13] work, we run the DIP algorithm for super-resolving the same observed sequence y by adapting their code available online at `https://dmitryulyanov.github.io/deep_image_prior` to the SR task for scale factor 3. We show the results of the DIP optimization in Figure 1c. While the DIP approach successfully sharpens the low-resolution observation, it also generates strong granular noise in its solution. Because the weights of our VSRMSE network act as the sole regularizers to our inversion problem, it should not be surprising that using learned parameters as opposed to random parameters produces cleaner super-resolved frames. A quantitative comparison presented in the caption of Figure 2 reveals that the fine-tuned $\mathrm{f}_\psi(\mathrm{y})$ results in an increase in PSNR of 2.09 dB from the original $\mathrm{f}_\theta(\mathrm{y})$ and an increase in 3.88 dB from the DIP solution $\mathrm{h}_\theta(\mathrm{z})$.

3.2. Removing artifacts in GANs.

We now set $\mathrm{f}_\theta(\cdot)$ to correspond to the VSRGAN model proposed by [5]. We show an example of a frame super-resolved by the VSRGAN model in Figure 2b. While producing a frame $\hat{\mathrm{x}}$ of significantly sharper quality than its MSE-trained counterpart VSRMSE (Figure 2d), it also inevitably introduces artifacts in the super-resolved frame. Indeed, a closer look at Figure 2b reveals the presence of a "dot-like" pattern in the super-resolved frame. These artifacts usually arise from complicated training dynamics due to the adversarial loss in VSRGAN. More specifically, we hypothesize that during training, the VSRGAN (erroneously) learns that the artifacts seen in Figure 2b are suitable high-frequency signals to generate when simulating high-resolution frames. Our objective thus is to remove these artifacts whilst keeping the super-resolved quality obtained from our GAN through appropriate fine-tuning of its parameters θ. Before applying the method of Equation 1, we note that the edges in the super-resolved frame are less affected by the artifacts, as opposed to the particular noisy flat regions in the frame in Figure 2b. The objective of our fine-tuning is thus to reduce the artifacts in the flat region whilst preserving the sharpness of the edges. With this in mind, we modify the problem defined in Equation 1 to use a mask that effectively places more penalty on the parameters that are responsible for influencing the flat areas of the frame. We compute the mask by applying the well-knwon Sobel operator $\mathrm{S}(\cdot)$ on the super-resolved frame $\hat{\mathrm{x}} = \mathrm{f}_\theta(\mathrm{y})$ to obtain a high-resolution edge map of the frame, and then define $\mathrm{M}(\hat{\mathrm{x}}) = 1 - \mathrm{S}(\hat{\mathrm{x}})$ as our masking operator in our objective function. Thus when applied to the problem of GAN artifact removal, our inversion problem becomes:

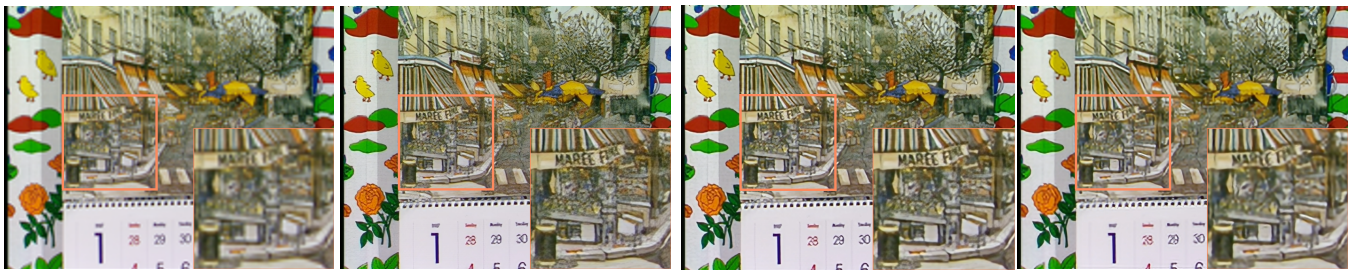
$$\hat{\psi} = \arg\min_{\psi} ||\mathrm{M}(\hat{\mathrm{x}}) \odot (\mathrm{Af}_\psi(\mathrm{y})_{bic} - \mathrm{y}_{bic})||_2^2, \qquad (2)$$

where $\odot$ denotes the element-wise multiplication operator, and $\mathrm{Af}_\theta(\mathrm{y})_{bic}$ and y_{bic} refer to the bicubically interpolated "predicted" low-resolution and ground-truth low-resolution frames, respectively. Working with these bicubically interpolated variables as opposed to their low-resolution counterparts is necessary in order to apply the masking $\mathrm{M}(\hat{\mathrm{x}})$ originally computed in high-resolution space. We show the results of our post-processing $\mathrm{x} = \mathrm{f}_\psi(\mathrm{y})$ in Figure 2c. Comparing Figure 2c with Figure 2b reveals that the dot-pattern artifacts were reduced by a significant extent. One may argue that the result of the fine-tuning led to a slightly smoother solution than prior to optimization, however, the perceptual quality seen in Figure 2c is still significantly higher than the VSRMSE solution in Figure 2d. In other words, fine-tuning the VSRGAN model results in a sharper solution than VSRMSE, and a cleaner solution than VSRGAN, hence combining the best of both models. The quantitative metrics for this experiments are consistent with these observations: the post-processed results with our fine-tuned network obtain the largest PSNR and SSIM metrics, surpassing the metrics computed from both the trained VSRGAN and VSRMSE.

4. DISCUSSION

The experimental results of the previous section showed that applying the data consistency constraints of Equations 1 and 2 on our trained models resulted in a reduction of the artifacts produced by VSRMSE and VSRGAN when tested on the case scenarios defined in Section 1.

Here, we emphasize that our proposed method is not limited to the problem of VSR or to the two case scenarios detailed in this paper. In fact, this method is easily extendable to numerous situations in which a user may want to reduce mild artifacts produced by a neural network trained for an inverse imaging problem at test time. Our method is, a priori, agnostic to the type of artifacts, or noise, that we wish to correct for. Figure 4a, for example, shows an image generated during a failed training experiment of VSRGAN, during which the generated frames darkened after each epoch. The result of applying our post-processing algorithm on such an image is shown in Figure 4b, where clearly the data consistency term successfully corrects for the discrepancy in brightness and increases the test PSNR by 1.25 dB.



(a) 17.67/0.5411 **(b)** 21.62/0.6892 **(c) 22.04/0.7183** **(d)** 21.61/0.6711

Fig. 2: Correcting the GAN artifacts. (a) Bicubic interpolation; (b) VSRGAN output ([5]); (c) Our fine-tuned VSRGAN output; and (d) VSRMSE output ([5]).

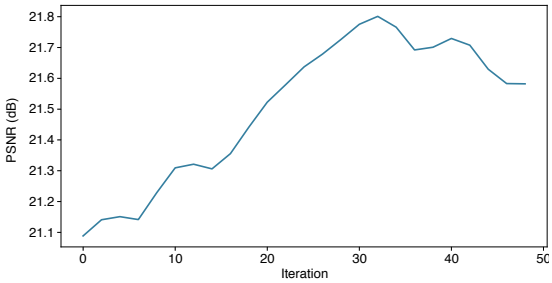


Fig. 3: An example of how PSNR changes as a function of fine-tuning iterations.

(a) 19.10/0.7315 **(b) 20.35/0.7779**

Fig. 4: Correcting failed training of VSRGAN for scale factor 3. (a) output of flawed VSRGAN; (b) fine-tuned VSRGAN.

As the iterations of our algorithm progress, we observe two simultaneous effects: the first is an iterative reduction of the artifacts in the frame, and the second is a slight smoothing of the frame at each iteration. The smoothing effect can be reduced by the user by setting a desired "trade-off" between the minimization of the artifacts and the conservation of the sharpness originally produced. The user may either make use of his/her subjective judgement to stop the algorithm when satisfied with the solution, or may compute the PSNR metric at each iteration, as shown in Figure 3, and settle on a specific iteration based on the metric. We note here that the post-processed images shown in Section 3 were extracted from choosing the iteration at which the PSNR peaks. Another user certainly could have chosen to maintain some of the artifacts and further restrict the subtle smoothing effect of the post-processing by stopping at the earlier iterations of the optimization.

A clear benefit of our proposed method, which may not be apparent at first, is its exceptionally fast execution. In case scenario #1, for example, fine-tuning our network for 30 iterations is completed in less than 4 seconds when implemented on a GTX 1080 GPU. On the other hand, the image restored by the DIP [13] method in Figure 1c required a total of 25 minutes of computation on the same GPU card. Our fine-tuning procedure is thus particularly suitable for real-time applications, which is certainly an advantageous quality in the industry domain.

One may argue that a limitation of our method is that certain conditions need to be fulfilled for the fine-tuning to succeed. First, Equations 1 and 2 require access to a trained network $f_\theta(\cdot)$, which may not always be accessible to all. A second requirement is that the degradation operator A used at test time should perform a task which is akin to the one used during training. Case scenario #1, for example, explored the case of super-resolution, in which the downsampling scale factor of the test input was different from the scale factor used to train the network. In this case, one may argue that fine-tuning parameters θ is an appropriate procedure as the trained parameters for the SR problem of scale factor 3 should not significantly differ from those of scale factor 4. However, if the degradation model A changes drastically from training to testing, we do not expect to obtain suitable post-processed results. For example, a neural network trained for the image denoising task will certainly not test well, even if fine-tuned for the super-resolution problem at test time, as these two tasks differ in nature.

5. CONCLUSION

In this work, we proposed an algorithm to attenuate artifacts in images generated by DNNs for imaging problems, by fine-tuning the parameters of our neural networks to satisfy a data consistency term given a new test data point. We found that our approach resulted in images of higher quality, as measured both qualitatively and quantitatively. We show that our methods introduced in this paper are efficient, applicable to a wide range of cases, and provide full control to the user to select the image that satisfy the given goals.

Post-processing the solutions of DNNs using fine-tuning on a test input image is a topic that has so far not been explored in the literature, which naturally provides room for further research and improvement. An example of such improvements may be obtained by combining the abundant research in analytical methods for inverse imaging problems with efficient fine-tuning in pre-trained parameter space. Investigating futher in this direction will provide deep learning scientists with more leeway to address the failure cases of deep learning, through the use of reliable, well-established analytical approaches offered by the image processing community.

6. REFERENCES

[1] A. Lucas, M. Iliadis, R. Molina, and A.K. Katsaggelos, "Using deep neural networks for inverse problems in imaging: beyond analytical methods," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 20–36, 2018.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[3] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2017.

[4] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C.C. Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *European Conference on Computer Vision*. Springer, 2018, pp. 63–79.

[5] A. Lucas, S. Lopez-Tapia, R. Molina, and A.K. Katsaggelos, "Generative adversarial networks and perceptual losses for video super-resolution," in *IEEE Transactions on Image Processing*, 2019.

[6] A. Lucas, S. Lopez-Tapia, R. Molina, and A.K. Katsaggelos, "Generative adversarial networks and perceptual losses for video super-resolution," in *IEEE International Conference on Image Processing (ICIP)*, 2018.

[7] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[8] X. Mao, Q. Li, H. Xie, R.Y.K. Lau, Z. Wang, and S.P. Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2794–2802.

[9] J-Y. Zhu, T. Park, P. Isola, and A.A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint*, 2017.

[10] C.K Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, "Amortised map inference for image super-resolution," *arXiv preprint arXiv:1610.04490*, 2016.

[11] J. Schlemper, J. Caballero, J.V. Hajnal, A.N. Price, and D. Rueckert, "A deep cascade of convolutional neural networks for dynamic mr image reconstruction," *IEEE transactions on Medical Imaging*, vol. 37, no. 2, pp. 491–503, 2018.

[12] K. Hammernik, T. Klatzer, E. Kobler, M.P. Recht, D.K. Sodickson, T. Pock, and F. Knoll, "Learning a variational network for reconstruction of accelerated mri data," *Magnetic resonance in medicine*, vol. 79, no. 6, pp. 3055–3071, 2018.

[13] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[14] Ad. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.