CSC420 Project Report
Super Resolution

Team: Tree Wizards

Adam Adli, Linwen Huang, Jason Tang

Abstract

Super-resolution is an extremely difficult computer vision problem that has seen considerable breakthroughs in recent literature; some interesting solution have been developed through the use of deep learning on generative adversarial networks (GAN) to upsample images with high perceptual quality. SRResNet and SRGAN are two examples of state-of-the-art neural networks that produce high-fidelity image upsampling. Recent breakthroughs have been made in hyper-parameter optimization with literature discussing a new optimization method called Delta-STN. In this paper, we present our attempt at applying Delta-STN optimization to SRResNet while exploring the efficiency of transfer-learning on a dataset we prepared. Through this approach, we document considerations we made regarding dataset preparation, Delta-STN application, and neural network implementation.

Introduction and Literature Review

The Problem

The problem of super resolution (SR) aims to generate a high-resolution (HR) version image from one or more low resolution (LR) versions of the image. Using a single LR image is referred to as single image super resolution (SISR), and similarly, using multiple LR images is referred to as multi-image super resolution (MISR) [1].

Often the ability to generate an HR image is useful as it can provide more valuable details, which is important in a variety of fields such as medical imaging, satellite imaging, and image/video enhancement [1, 2]. This can also lead to reduced bandwidth required to transmit images and video, which make up a significant amount of the information transmitted throughout the internet. Traditional methods for upscaling have usually relied on methods such as interpolation. While these methods are simple to implement, they often produce subpar results.

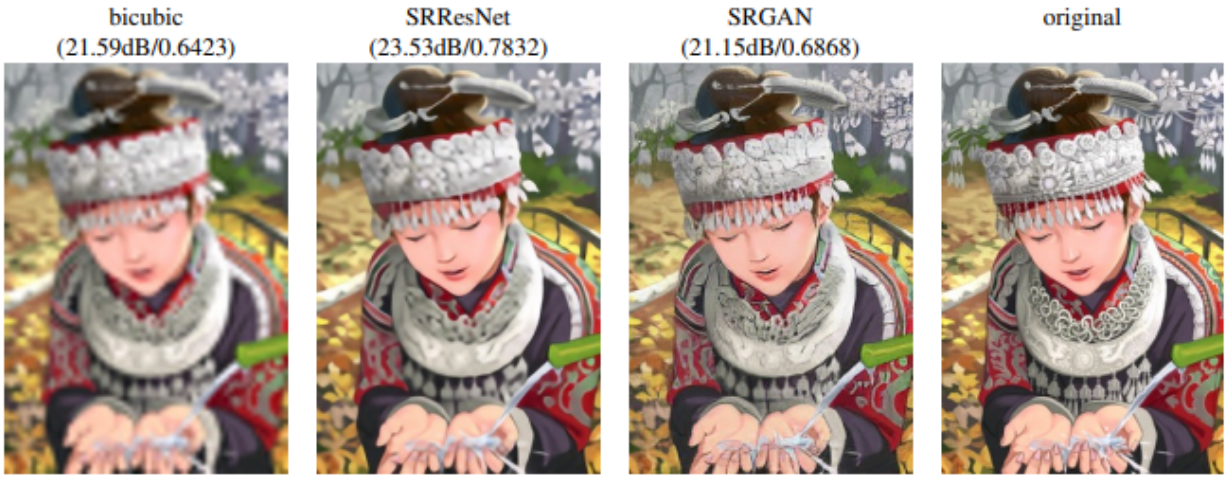


Figure 1: Examples of images processed by 3 different SR models: bicubic, SRResNet, SRGAN [2]

Relevant Research

There have been many approaches to the SR problem. Traditionally there are three main types of SISR algorithms: interpolation based, reconstruction-based and learning-based [1].Examples of interpolation-based algorithms include the bicubic interpolation and Lanczos resampling. Reconstruction based methods such as those by Dai et al., Sun et al., Yan et. Al. And Marquina et. Al have been used by restricting the solution space to generate HR images [1]. Learning-based methods have been used by Freeman et al. and Chang et al. These methods employ machine learning algorithms to analyze the statistical relationships between an LR image and a HR image using training examples [1]. However, these approaches have suffered from shortcomings, such as accuracy and generalizability issues in addition to lack of scalability. In recent years, the focus in SISR has been with deep learning-based methods which have shown significant improvements in results compared to older methods [1].

In the last few years, the application of deep learning convolutional neural networks (CNNs) has become prominent [1]. This has lead to the creation of SR models known as SRCNNs. The focus of our project however looks at another type of deep learning SR model, the SRGAN, created by Ledig et al. [2] SRGANs are super resolution models that use generative adversarial networks (GANs) are a type pf generative model that uses supervised learning. There are two networks associated SRGANs, the generator network and the discriminator network. During training a HR image is down sampled to a LR image. The image is then passed to the generator network which attempts to upsample the imae to create a SR image. The discriminator then attempts to distinguish between the SR image and the ground truth HR image. The discriminator then backpropagates the GAN loss to train the discriminator and the generator (to create SR images more reflective of the ground truth images) [3]. Another network is the SRResNet which is the same model as SRGAN but just without the GAN portion [3].

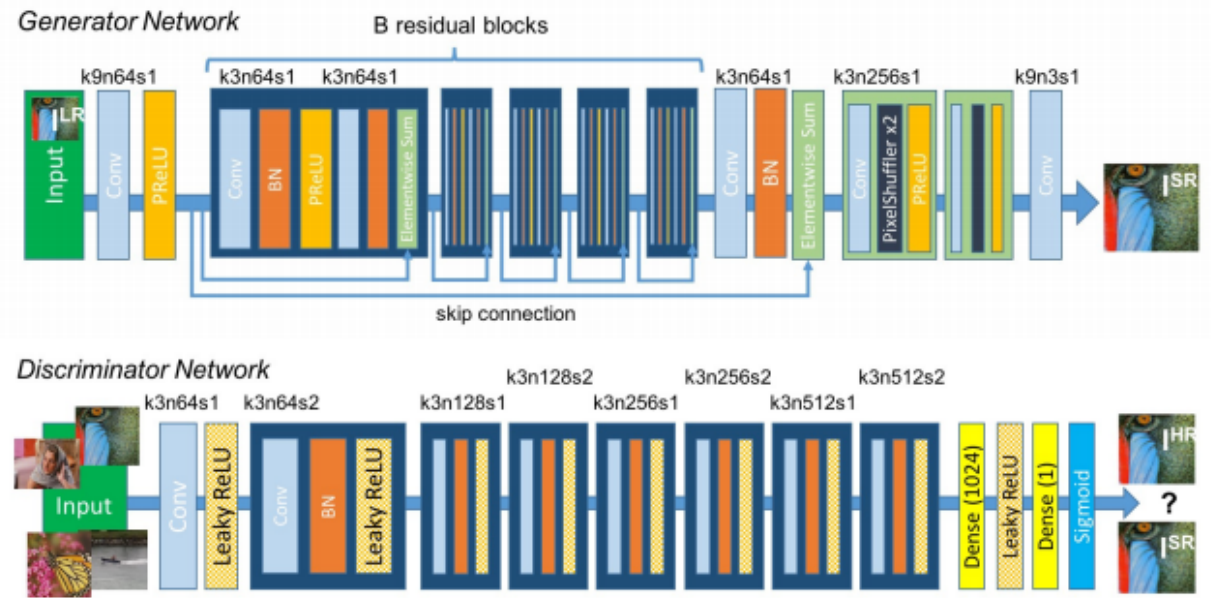


Figure 2: SRGAN architecture used by Ledig et al. [2]

The generator network uses residual blocks originated by ResNet, within each residual block is

two convolutional layers [3]. ResNet introduces skip connections aka shortcut connections which allows for deeper networks. This helps to alleviate the issues that arise when deep CNNs have too many layers [4]. The residual block uses small 3x3 kernels and 64 feature maps, followed by batch-normalization and Parametric ReLU [3]. In the discriminator network a Leaky Relu is used. The network contains 8 convolutional layers, and mimics the structure seen in the VGG network [3]. This results in 512 feature maps followed by 2 dense layers and a sigmoid activation function [3]. SRGAN also uses a perceptual loss function, which is the waited sum of the content loss and an adversarial loss component [3].

$$\underbrace{l^{SR}}_{\text{perceptual loss (for VGG based content losses)}} = \underbrace{\underbrace{l_{\mathrm{X}}^{SR}}_{\text{content loss}} + \underbrace{10^{-3} l_{Gen}^{SR}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

Figure 3: Perceptual loss function used in SRGAN [3].

Instead of using MSE loss for content loss SRGAN uses VGG loss, which is the euclidean distance between the feature representation of the SR generated image and the ground truth HR image [3]. The content loss uses perceptual similarity instead of pixel space similarity [3]. The adversarial loss is based on the probablities (the probablity the SR generated image is a ground truth HR image) of the discriminator over all training samples. As such the adversarial loss pushes our SR generated image closer to the ground truth HR image [3].

SRGAN outperforms traditional and CNN based SR models. It was also one of the first models that could infer photo realistic images with a 4x upscaling factor. As a result SRGAN shows state of the art results, and produces extremely high detailed textured images. SRGAN also scored extremely well in mean opinion score tests compared to other models [5]. SRResNet also outperformed models in PSNR and SSIM [5].

While SRGAN and SRGAN-based neural networks provide state-of-the-art SISR, it is still not a perfect solution. It can still be susceptible generalizability issues depending on training data set [5]. Additionally, SRGAN may be highly effective at high-frequency texture generation while introducing artifacts that may distort text [6]. Furthermore, the use of residual blocks come with a computational cost in exchange for higher accuracy in SRGAN models like SRResNet [6]. Making them even more difficult to develop in face of computational cost, GAN-based networks are known to be difficult to transfer-learn [7]. With these trade-offs in mind there is still heavy research underway in the domain of SISR and its use of general adversarial networks.

Delta-STN by Juhan Bae and Roger Grosse presents a novel approach for automating the tuning of regularization hyperparameters such as dropout, weight decay, and data augmentation values [8]. It improves upon conventional methods such as random search, grid search, and Bayesian Optimization by utilizing a hypernetwork to approximate the best response function. This can be especially useful in SISR since learning optimal hyperparameter values for data augmentation could better regularize the model and allow it to better generalize to unseen data.

Methodology, Results, and Experiments

Sources

The pretrained weights and a portion of our codebase is based on Donghee Son's unofficial implementation on Github [9]. Our main extensions are our methods for dataset processing and transfer learning the pretrained weights.

Dataset Preperation

A standard dataset used for super-resolution training and evaluation is the DIV2K dataset, which is the one our pretrained weights were trained upon. There are some considerations that must be made when preparing a super-resolution dataset. Namely, we want to ensure that the dataset is complex enough in detail to show the power of Super Resolution, but we also require that the dataset is small enough in size to train in a reasonable amount of time. We also found that using photos with JPEG compression would yield artifacts when input into super-resolution models as they are less trivial to use due to blocking artifacts; so we also limited our choices to uncompressed file formats [10].

To accomplish this, we chose a dataset of 800 Pokemon. Within it, there are some images that are image and with a low level of detail, and there are some of which are fairly complex and detailed. We then resized the original 475×475 images into 256×256 high resolution images, which we then scaled down by a factor of 4 to get 64×64 low resolution images. Both these operations were performed using Bicubic interpolation with the Pillow library. The smaller sizes allowed us to train both faster and with more stability due to the ability to use larger batch sizes.

The final action we performed on the data prior to actual training was data augmentation. Specifically, we used randomized affine and homography transformations to alter the perspective and view of the original image, as we learned in lecture. Performing this allowed us to essentially create new altered images for each training iteration, thereby reducing the possibility of our model overfitting to the small number training samples. Otherwise, it would be possible for our model, with its large parameter capacity, to memorize data points rather than actually learn how to perform super resolution, leading to degraded generalization performance on non-training data.

Experiments

Focusing on SRGAN and SRResNet neural network architectures, we took an exploratory approach to our project attempting to expand on existing work through different objectives:

1. prepare a dataset for super-resolution learning.
2. conduct transfer-learning using our prepared dataset using SRResNet.
3. experiment with Delta-STN.

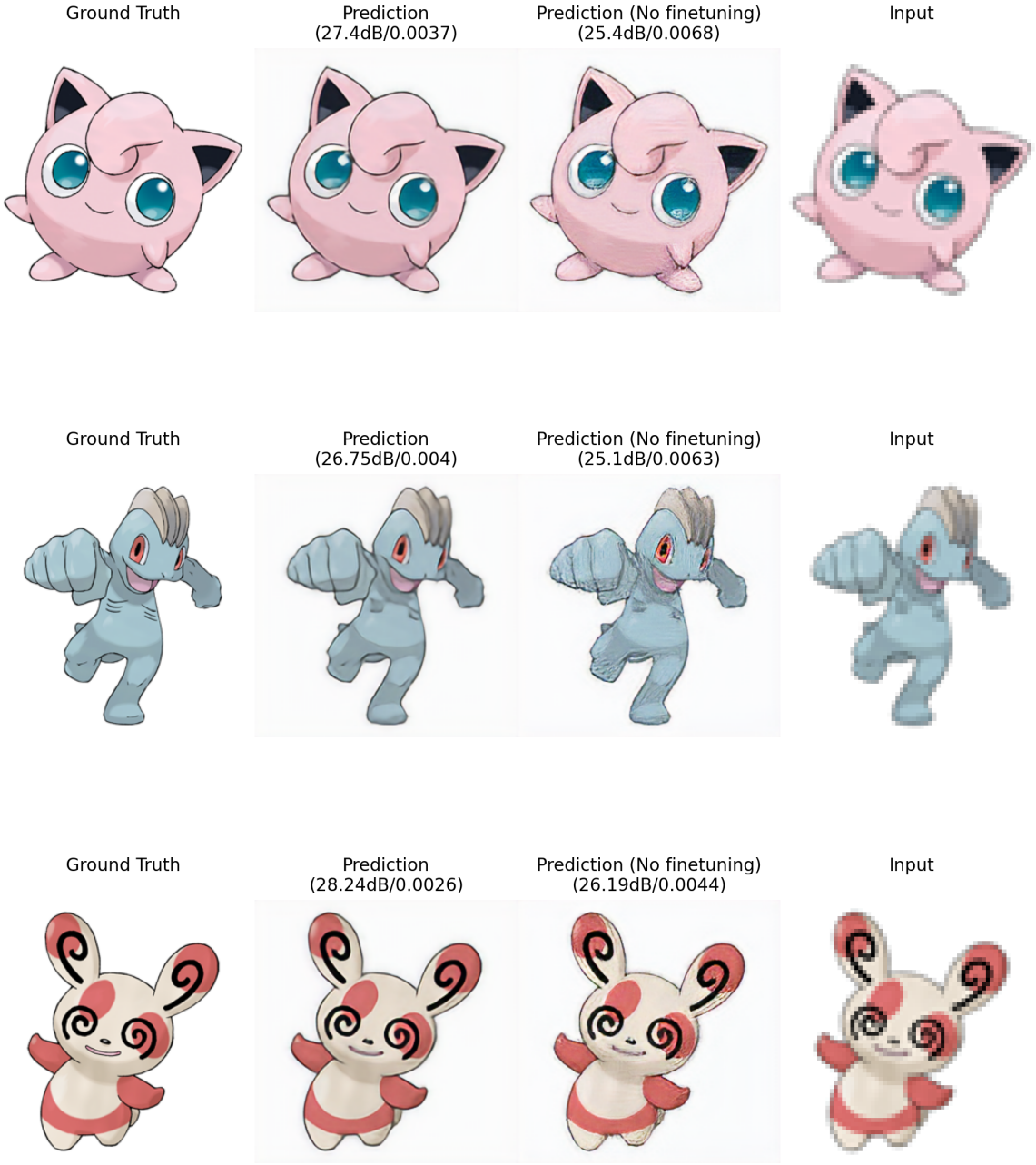
Although we made considerable progress on our objectives, we ran into some challenges that hindered our final results. Nonetheless, we gained some insights that we document.

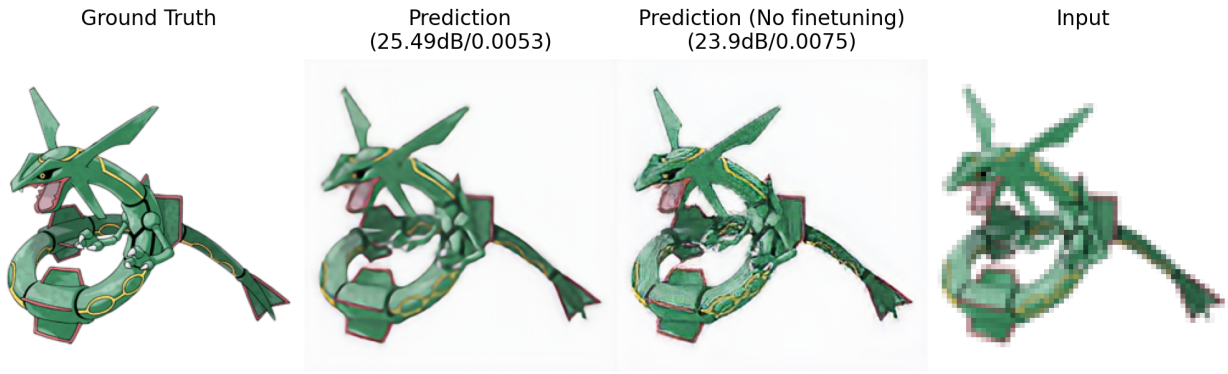
First, as described in the previous section, we were able to successfully prepare our dataset of Pokemon for use in transfer learning with the pretrained weights. We were then able to create a

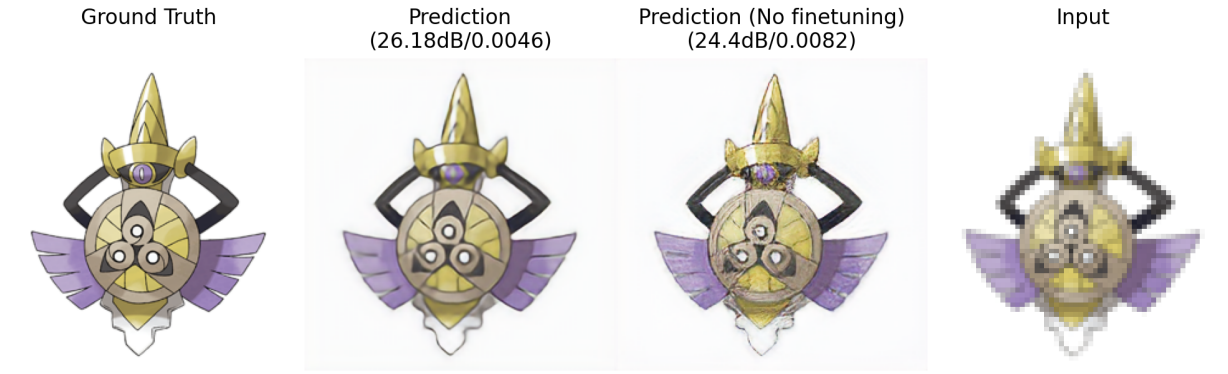
training loop in which we fine tuned these pretrained weights, resulting in a noticeable increase in performance, as we show in the following section. Lastly, we ran into time and compute challenges with incorporating the Delta-STN implementation into our SISR model so we leave it as an avenue for further research and exploration.

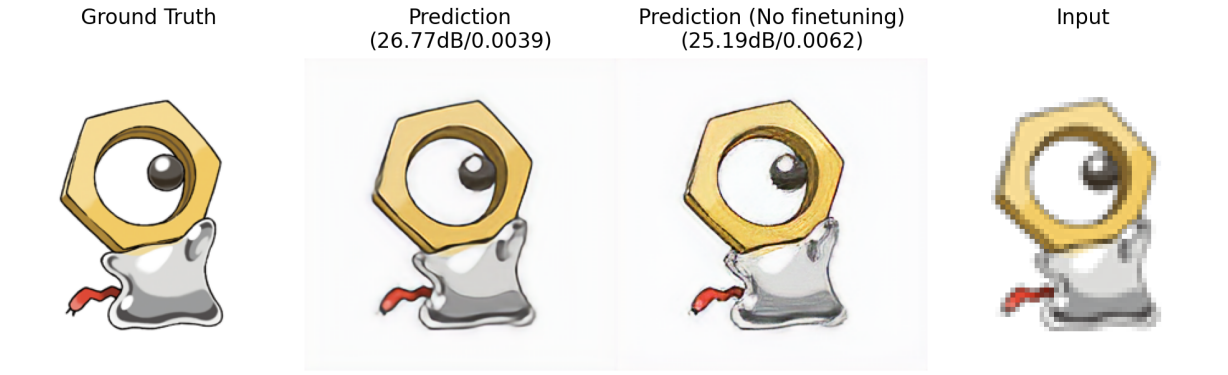
Results

Below are several selected example predictions and metric evaluations of images in the test set. The values in the brackets represent PSNR (Peak signal-to-noise ratio) and Perceptual Loss (MSE on the outputs of VGG19 layer 5-4).









Below are the averaged performance on various metrics for images in the validation set.

	MSE	PSNR	VGG22	VGG54
Without finetuning	0.002561	27.2486	0.015313	0.004093
With finetuning	0.002394	27.4498	0.012861	0.003504

Conclusion

Overall, we were able to slightly improve the scores achieved using MSE and PSNR, as well as improving both perceptual losses and MSE loss between output features at certain layers of the VGG19 model. In our results we also saw that our model was able to remove artifacts that were found in the non-finetuned model without sacrificing too much image sharpness. This increase in performance can be attributed to the various forms of regularization we introduced during the training of our model. For example, the affine transformations applied to augment our data, and the application of fine tuning models in parallel with an adversarial network.

Though we met many difficulties and obstacles in this project, we have learned a lot as a result. Specifically, we became more familiar with the different models and techniques in the field of downsampling and Super Resolution, as well as the various types of dataset preparation and data considerations to be made. Additionally, we became better acquainted with the PyTorch ML library, other implementation tools/frameworks, and the basic principles behind implementing and training neural networks.

Author Contributions

Adam Adli

- Research of models/pretrained weights
- Sourced pokemon datasource
- Set up code base for SRGAN/SRResNet
- Wrote Python scripts involved with using pretrained weights, data processing and modified source code for SRGAN/SRResNet
- Wrote up proposal and report

Linwen Huang

- Research of models/pretrained weights
- Sourced pokemon datasource
- Wrote python scripts used in generating LR images/data processing and modified source code for SRGAN/SRResNet
- Created presentation slide deck, wrote up proposal and report

Jason Tang

- Performed transfer learning on the pretrained weights
- Set up data augmentation pipeline
- Wrote class to evaluate multiple metrics on various datasets
- Attempted initial experiments with Delta-STN
- Wrote up proposal and report

References

[1] Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing Hao Xue, and Qingmin Liao. 2019. Deep Learning for Single Image Super-Resolution: A Brief Review. *IEEE Transactions on Multimedia*, 21, 12, (December 2019), 3106–3121. ISSN: 19410077. DOI: 10.1109/TMM.2019.2919431. arXiv: 1808.03344. https://arxiv.org/abs/1808.03344v3.

[2] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. Volume 2017-January. Institute of Electrical and Electronics Engineers Inc., (November 2017), 105–114. ISBN: 9781538604571. DOI: 10.1109/CVPR.2017.19. arXiv: 1609.04802. https://arxiv.org/abs/1609.04802v5.

[3] Jonathan Hui. 2018. GAN – Super Resolution GAN (SRGAN). (2018). Retrieved 12/19/2020 from https://jonathan-hui.medium.com/gan-super-resolution-gan-srgan-b471da7270ec.

[4] Sik-ho Tsang. 2018. Review: ResNet – Winner of ILSVRC 2015 (Image Classification, Localization, Detection). (2018). Retrieved 12/19/2020 from https://towardsdatascience.com/review-resnet-winner-of-ilsvrc-2015-image-classification-localization-detection-e39402bfa5d8.

[5] Sik-ho Tsang. 2020. Review: SRGAN & SRResNet – Photo-Realistic Super Resolution (GAN & Super Resolution). (2020). Retrieved 12/19/2020 from https://sh-tsang.medium.com/review-srgan-srresnet-photo-realistic-super-resolution-gan-super-resolution-96a6fa19490.

[6] Katarzyna Kańska. 2017. Using deep learning for Single Image Super Resolution. (2017). Retrieved 12/19/2020 from https://deepsense.ai/using-deep-learning-for-single-image-super-resolution/.

[7] Yaël Frégier and Jean-Baptiste Gouray. 2020. Mind2mind : transfer learning for gans. (2020). arXiv: 1906.11613 [cs.LG].

[8] Juhan Bae and Roger Grosse. 2020. Delta-STN: Efficient Bilevel Optimization for Neural Networks using Structured Response Jacobians. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*. Neural information processing systems foundation, Vancouver, Canada, (October 2020). arXiv: 2010.13514. http://arxiv.org/abs/2010.13514.

[9] Donghee Son and Seobin Park. [n. d.] dongheehand/SRGAN-PyTorch: SRGAN (Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network) implementation using PyTorch framework. (). Retrieved 12/19/2020 from https://github.com/dongheehand/SRGAN-PyTorch.

[10] Fengchi Xu, Zifei Yan, Gang Xiao, Kai Zhang, and Wangmeng Zuo. 2018. Jpeg image super-resolution via deep residual network. In *Intelligent Computing Methodologies*. De-Shuang Huang, M. Michael Gromiha, Kyungsook Han, and Abir Hussain, editors. Springer International Publishing, Cham, 472–483. ISBN: 978-3-319-95957-3.