Super-resolving SEM images using limited training data

A Project Report

submitted by

NIKAM ASHUTOSH SHASHIKANT

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR AND MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2021

THESIS CERTIFICATE

This is to certify that the thesis titled **Super-resolving SEM images using limited training data**, submitted by **Nikam Ashutosh Shashikant**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. A.N. Rajagopalan
Research Guide
Professor
Dept. of Electrical Engineering
IIT Madras, 600 036

Place: Chennai

Date: June 2020

ACKNOWLEDGEMENTS

Throughout the work for this thesis, I have received a great deal of support and assistance.

I would like to firstly thank my advisor, **Professor A. N. Rajagopalan**, who trusted me with this challenging research problem and patiently guided me throughout the course of this work. His expertise and insightful feedback has brought this work to a higher level.

I would like to thank **KLA team**, for giving me an opportunity to work on their problem statement, and for all their time and feedbacks.

I would like to thank **Samsung Pravartak Award** for the scholarship grant.

I would like to thank my PhD guide, **Mahesh Mohan**, for all the insightful discussions and constant support in every little aspect related to this work. I would like to thank **Saurabh Goswami**, MS student at IPCV lab, for his time and guidance when I hit a brick wall.

I would like to thank **Priyatham Kattakinda**, former M.Tech. student at IPCV Lab, and **Saurabh** again, for providing me with the source code for their previous solutions. The current work is largely built upon their shoulders.

I would also like to acknowledge **Kranthi Kumar Rachavarapu** and **Praveen Kandula**, PhD students at the lab, for their technical help. It helped increase the pace of this work by multiple folds.

Finally, I would like to thank my parents, who have always been with me sharing both my happiness and troubles. I also want to thank my friends for all their kindness and frequent forgiveness. I would like to thank the people out there, working relentlessly to bring us out of these tough times.

ABSTRACT

In recent years, the task of single image super resolution (SISR) has seen a significant progress with the advent of data-driven deep learning based methods. Most of the state-of-the-art models employ supervised-training method where LR images are synthetically generated using simple bicubic downsampling, and hence do not generalize well to real-world data.

Very recently several attempts using unsupervised training have been made to solve this problem. They try to model real-world degradations and produce more data for supervised training. These methods assume availability of large amounts of degraded data, which may not be true in certain cases, where they will fail to faithfully estimate the real-world distributions.

To solve this problem, we propose a empirically-motivated data augmentation pipeline to generate training data. Our experiments with a KLA-SR dataset validate the effectiveness of our proposed method, producing near-realistic SR images. We also discuss the challenges involved in training SR models on datasets with limited variation, namely the unstable GAN training and noise-induced artefacts. We investigated these problems througly, and provide simple guidelines to avoid them. Our proposed solutions has been evaluated on all the KLA-SR datasets, and solves all the major challenges faced by previous methods.

Original images used for our experiments are the property of KLA Corp. and hence, censored with alternate fake [source] images.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The task of single image super-resolution (SISR) involves reconstruction of high resolution (HR) image from a low resolution (LR) image. It has broad applications in tasks like medical imaging and surveillance, and for image enhancement tools. In our case, the need for super-resolution is primarily motivated by need of accelarated yet robust defect detection on semiconductor substrate using Scanning Electron Microscope (SEM) images.

In recent years, deep learning based methods, specifically Convolutional Neural Networks (CNNs) have achieved remarkable results on the SR task, sparking huge interest among the computer vision community. These methods are based on superised learning approach under which models are trained on large number of LR-HR pairs and learns the mapping between them. But LR images used for training are generated using simple bicubic downsampling, both during the training and testing phase. Such a data-creation mechanism does not capture the real-world degradations failing to generalize for real applications.

In order to solve this limitation, unsupervised learning methods have been proposed very recently. They basically try to model the real-world degradations, for example, the sensor noise and artefacts present in real-images. Most of these methods adopt either cycle consistency loss to style transfer the degradations from real LR images to clean LR image, and create more training data. But these methods require enough samples of real LR images to capture the distribution. Another set of popular approach is based on the idea of blind SR in which LR degradations are assumed to be fixed but unknown. By far, the blind SR solutions assume only a simple blur and noise degradation.

We consider a related version of blind SR problem, in which the degradations are mild structural distortions. Specifically, we are presented a training dataset with clean LR-HR images, and we learn a end-to-end SR model which can generalize the mapping from training LR-HR pairs to LR images containing slight distortions, namely *defects* in the images. Our problem statement is slighly different that blind SR because the

SR model is not supposed to *remove* or *repair* the defects; rather it needs to enhance them because the goal of super-resolving these images is to improve the defect *detection* performance. We call it the *blind-defects SR* problem. The problem of generalizing SR mapping to defects is interesting because defects are majorly manufacturing anomalies and hence, are rare considering the modern six-sigma standards. Creating a dataset with defective LR-HR pairs or simply with defective LR images, large enough to train deep-learning models is a cumbersome task.

(a) N1 (b) proposed method (c) GT

Figure 1.1: SR results obtained by (a) N1 (b) The proposed method (c) ground truth HR

(a) N1 (b) GT

Figure 1.2: The lost defects issue (a) SR result from N2 (b) ground truth HR

We experimented with 4 different SEM image datasets, and identified two major challenges in solving the blind-defects SR problem:

- Popular deep learining based SR models do not always generalize the mapping learnt from clean LR-HR pairs to defects. In such cases, defects are heavily distorted in the generated images, as shown in Fig. 1.1. More alarmingly, for small defects, the structural alterations due to the defects are ignored, and defects are absent in the generated outputs, as shown in figure Fig 1.2. We call this the *lost defects* problem. If defects are lost in LR to SR conversion, our main aim of boosting defect detection is defeated.

- Semiconductor substrate datasets contain very less variation across different samples, when compared to natural images datasets. This leads to GAN training issues when using GAN-based SR methods.

We sill discuss these issues in more details in Chapter 2. The 4 SEM datasets will be discussed in Section 4.

To solve the lost defects issue, we propose a **empirically-motivated data augmentation approach**. This method is based on the ability of a modern SR networks to *identify* the defects regions and learning a totally *separate* LR-HR mapping for these identified regions, all in a automatic end-to-end manner. To solve the GAN training problem, we suggest **simple hyperparameter free combination**. This solution is not dependent on dataset in any manner, hence can be easily incorporated to improve the GAN training stability in other domains. Our overall solution draws heaviy from a previous solution, namely *N1*. Hence, We call the proposed method as *Enhanced N1* or *EN1*. We also investigate *artefacts* issue faced by another previous method, namely *N2*. We evaluate our methods on all 4 SEM datasets. The proposed method is compared to N1 and N2, to show the show the qualitative improvements in results. An example is shown in Fig. 1.1.

In the following sections, we will look at the previous work in super-resolution and blind-defect SR.

1.1 Related work in Super-resolution

Recently, significant progress has be achieved in performing SISR based on Convolutional Neural Networks (CNNs). The first one is proposed by [Dong *et al.* (2015)], which was a shallow neural network. Initial works [Dong *et al.* (2015), Kim *et al.* (2016), Tai *et al.* (2017)] used a bicubically interpolated LR image as input to CNN. In contrast, Shi *et al.* (2016) take a actual LR image as input and perform the upsampling as part of the SR network. Recently, many succesfult architectures [Lim *et al.* (2017), Zhang *et al.* (2018), Haris *et al.* (2018), Dai *et al.* (2019)] have been proposed for SISR task which achieve state-of-the-art performance on PSNR metrics. Ledig *et al.* (2017) proposed GAN based framework to improve the perceptual quality of the SR results. Many successful works [Wang *et al.* (2018), Zhang *et al.* (2019), Rad *et al.* (2019)]

follow a similar GAN-based design now.

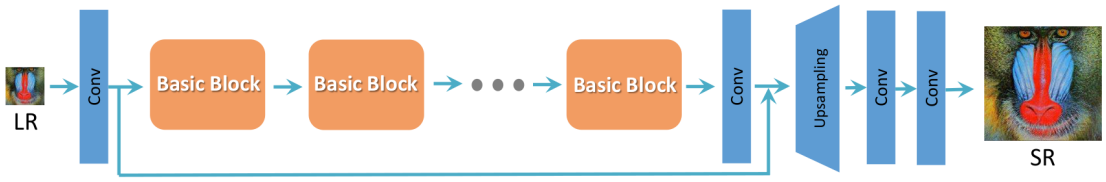
As discussed earlier, the aforementioned methods use supervised training on LR-HR pairs in which LR images is generated by bicubic downsampling. Bulat *et al.* (2018) introduced *realSR* dataset containing true LR-HR images. However, creating real LR-HR dataset sufficient to train a neural network, and repeating it for every domain-specific SR task, is a cumbersome task. Hence, such datasets are limited.

Lugmayr *et al.* (2019) introduced unsupervised learning into the domain of SISR. They style-transfer the degradations in real-world images (e.g. sensor noise artifacts) by using cycle consistency loss[Zhu *et al.* (2017)]. This synthetic generated images are used as LR images for supervised training on a neural network.Yuan *et al.* (2018) use cycle GAN to achieve the opposite, i.e. produce noise-free LR image, followed by separate SR step. Chen *et al.* (2020) use similar framework of unsupervised learning of degradation followed by supervised SR learning.

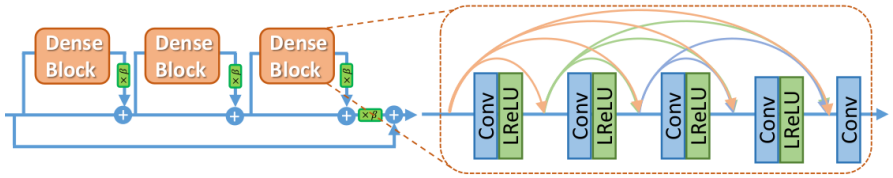
Another philosophy to deal with real-world degradtion is assuming the degradation to be blind-SR problem, meaning the degradation is fixed but unknown. Recent works [Zhou and Susstrunk (2019), Bell-Kligler *et al.* (2019)] assume blur degradation use deep learning models to estimate blur kernels. Shocher *et al.* (2018) train a CNN on single LR images.

1.2 Network N1

This network, introduced in 2019, uses a ESRGAN[Wang *et al.* (2018)] generator. ESRGAN introduced two important features to the SRResNet[Ledig *et al.* (2017)] generator: 1) removal of all BN layers 2) replace the basic building block of original SRResNet with a Residual-in-Residual Dense Block (RRDBs). They argue that BN layers can introduce unpleasant artifacts at the time of testing, when using a GAN framework. RRDB blocks boost performance by allowing more layers and connections. N1 uses 8 RRDBs generator with 128 channels. Fig. 1.3 (b) shows the RRDB structure.



(a)



(b)

Figure 1.3: (a) N1 generator architecture (b) RRDB structure [source : Wang *et al.* (2018)]

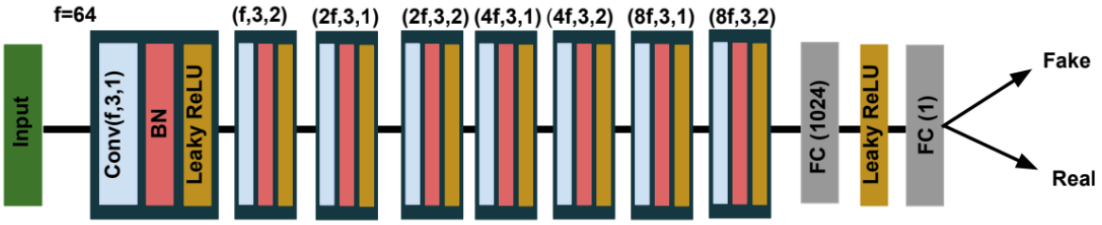


Figure 1.4: ESRGAN discriminator architecture. Conv (x, y, z) stands for 2D conv layer with channels x, kernel size y and stride z. N1 uses same distrimi-nator with f=32, [source : Vasu *et al.* (2018)]

Apart from this, its features are very similar to SRGAN[Ledig *et al.* (2017)]. It uses a SRGAN discriminator (used in ESRGAN as well) with 8 convolutional layers, starting with 32 channels in 1st conv layer. The overall generator loss function is given as follows:

$$L_G = L_{VGG} + \lambda\, L_{adv} \tag{1.1}$$

Where L_{VGG}, used as perceptual loss, is defined as euclidean distance between features representation of ground truth HR image and SR image produced by generator. We will explain the VGG loss design more in Section 3.4. L_{adv} is adversarial loss. N1 uses the Standard non-saturating GAN loss for L_{adv}. We will exlain about adversarial loss in more detail in Section 3.4. N1 uses VGG19-31(after activation) features, which are much shallower when compared to the levels VGG19-22 and VGG19-54 as prescribed

by the papers (VGG features naming convention is as described in [Ledig *et al.* (2017)]). No pointwise losses are used in generator loss. It also uses a early stopping regularizer for discriminator. The discriminator learning rate is annealed much faster compared to generator, effectively stopping its training after few epochs. This is done to stabilize the GAN training.

This model, when proposed as solution to dataset 4, used $\lambda = 0.01$ and no L1 or L2 loss pretraining. For dataset 4, N1 produced very impressive results. Compared to its predecessor, the results were much sharper and detailed. It had one drawback : the results were missing the background noise pattern and hence, looked overly clean and unrealistic. As we will see in Section 2.2, this is related to presence of early stopping regularizer.

1.3 Network N2

This network, introduced in 2020, is modified SRGAN[Ledig *et al.* (2017)] based network. It used a SRGAN generator with BN layers replaced by Instance Normalization[Ulyanov *et al.* (2016)], and pixshuffle layers [Shi *et al.* (2016)] with convolutional layers using computationally-efficient bilinear upsampling. The generator was 8 Residual blocks SRResNet with only 16 channels.

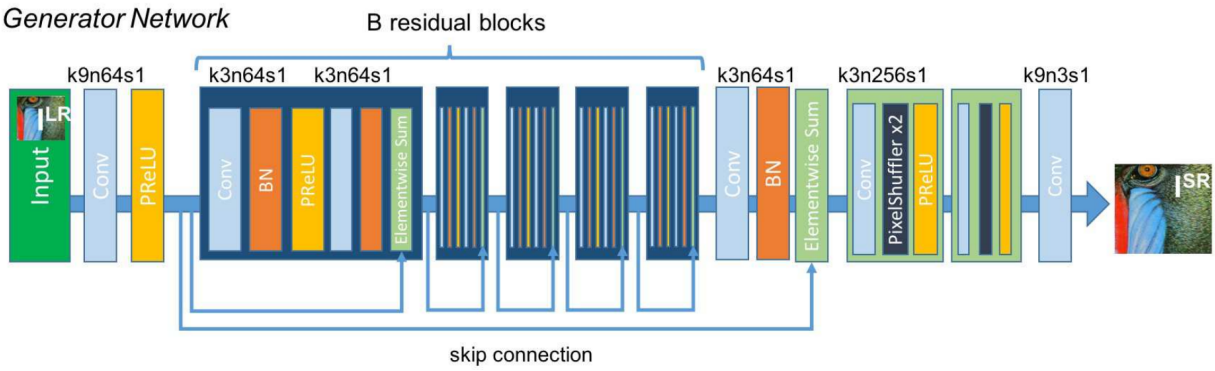


Figure 1.5: SRResNet architecture. kxnysz stands for Conv layer with kernel size x, channels y and stride z. N2 uses similar generator with B=8 and 16 channels instead of 64 [souce : Ledig *et al.* (2017)]

The dicriminator is a fully convolutional network with expanding and shrinking feature maps blocks. Both the blocks have 2 convolutional layers each. The generator

loss used has the same expression as Equation 1.1, but it uses a VGG19-32 features for VGG loss and WGAN-GP[Gulrajani *et al.* (2017)] loss for adversarial loss. WGAN-GP loss has been shown to stabilize GAN training, and hence no separate regularizer is required for N2.

This network was proposed for dataset 2, for which $\lambda = 0.1$ was suggested and pre-training with MSE loss was also prescribed. Due to very small size training data (only 20 samples) provided for dataset 2, N2 uses a small 16-channels generator, which is much smaller to that used in practice. For dataset 2, the network produced good results both qualitiatively (realistic looking) and quantitaively (defect detection accuracy). It was also computationally fast requiring only 65 mins to train completely. It had one major drawback though : The results contained artefacts due to heavy noise present in input images. We will see in Section 2.1. that this is related to using the small-size generator.

CHAPTER 2

CHALLENGES WITH LIMITED-DATA SR

As discussed in chapter 1, most state-of-the-art SR models, using supervised or unsupervised nethods, use large amount of data. In our case, we do not have the luxury of large image data, especially the ones with *defects*. In this chapter, we will look at what issues manifest when we train SR models with limited data.

We will first look at defect distortion issue arising from the overfitting of large generators on limited data, and why naive reduction in size of the generator is not a good solution. In section 2.2, we will look at GAN training issue arising from the limited variation in the data. We will use network N1 for experiments and inferences in this chapter, for both its simplicity and impressive performance in the past. Since most of the popular SR methods use similar architectures, we believe our observations and inferences should generalize to them.

2.1 Defects distortions

Our aim under the blind defects SR is to generalize the LR-HR mapping from our limited dataset to the LR images with defects. Upon training N1 with dataset 2, we observe a peculiar behaviour. Fig. 2.1 shows a N1 result for a test samples from dataset-2 (see section 4.1 for deetails of this dataset). Surprisingly, we get a background (everything other than defects) to be very clean and more visually appealing compared even to ground truth HR, but the defects are significatly distorted, to the extent that they convey almost no information about the real defects structures (Fig.2.1 (c)).

(a) LR (b) HR (c) GT

Figure 2.1: SR results obtained on dataset 2 by N1. The yellow bounding box shows the defect region in each image

Fig. 2.2 shows some more results for defects, obtained by N1. Fig. 2.2 (a) shows a tripetal defect, formed from boundaries of 3 neighboring balls dissolving into one other. The N1 results do not capture this information. The produced results contains 3 distinct balls as if there is no defect at all. This is what we call the *lost defects* problem. The SR results are completely losing the information of small defects. When we use these SR images for defect detection, these small defects cannot be captured at all. So, by using SR naively, we are hurting the defect detection performance instead of improving it.

Figure 2.2: SR performance by N1 on dataset 2 defects

Through rigorous sensitivity analysis experiments to study of the effects of various factors on defect distortion, we conclude that the **overfitting** is the primary reason. Reducing the size of the generator network, allowed it to generalize the LR-HR mapping to defect regions, improving defect structure drasticly and even fixing the lost defect problem. This conclusion advocates the use of small generator networks to avoid overfitting. Following this philosophy, we designed a network similar to N1, but using much smaller networks with less number of parameters. We call this network $\text{N1}_{\text{reduced}}$. Appendix A gives details related to $\text{N1}_{\text{reduced}}$ architecture.

(a) 1 RRDB (b) 4 RRDB (c) N1 (8 RRDB)

Figure 2.3: SR results obtained on dataset 2 by $\text{N1}_{\text{reduced}}$

Fig. 2.3 shows SR results produced by 1 RRDB and 4 RRDB deep $\text{N1}_{\text{reduced}}$. By network size reductions, We observe improvement in defect sharpness and structure. However, there is significant fall in background quality. Smaller networks produce backgrounds with distorted balls. Fig. 2.4 compares the defects produced using 1 RRDB and 4 RRDBs $\text{N1}_{\text{reduced}}$. As we move from 1 RRDB to 4 RRDBs, we see the defects are already being distorted due to overfitting, although we do not have lost defects problem. The distortions similar to Fig. 2.4 also exist for 2 and 3 RRDBs $\text{N1}_{\text{reduced}}$. This shows that increasing the size even from 1 RRDB to 2 RRDB, the generalization of LR-HR mapping to defects falls apart. We get generalization only for 1 RRDB. But 1 RRDB network capacity is too small to produce good quality backgrounds. Moreover, we still do not get ideal defects structures even after exploiting 1 RRDB generator's generalization.

There is another problem to reducing network size. Fig. 2.5 shows the SR results produced by N1 (8 RRDBs), and those from 1 RRDB $\text{N1}_{\text{reduced}}$, on dataset-1 (see section

Figure 2.4: SR performance by $N1_{reduced}$ on dataset 2 defects

4.2 for details of this dataset). We see dark undesirable artifacts in the 1 RRDB N1 results. Similar artifacts are observed in SR results obatined by N2 (Section 1.3) on dataset-1. The only common feature $\text{N1}_{\text{reduced}}$ and N2 is their small size. Also, we do not get such artefacts by increasing $\text{N1}_{\text{reduced}}$ depth to 3 RRDBs. Figure 2.5 also shows that the artifacts are highly correlated to noise present in input LR image. This observations prove that **small generator size is susceptible to input noise**. The reason is that the network capacity is too small to learn a complex denoising and SR mapping from LR to HR domain. It is difficult to produce results which are perfect in all aspects, using small network size.

(a) LR (b) 1 RRDB $\text{N2}_{\text{reduced}}$ (c) GT

Figure 2.5: SR results obtained on dataset 1 by $\text{N1}_{\text{reduced}}$

Although defect distortion is caused by overfitting, we see that reduction in network size is not a good solution in the face of limited data. In section 3.1, we motivate and propose a dataset-augmentation approach which allows us to use large generator, and retain background quality which improving defect structure.

2.2 GAN training instability

To understand the problem, we need to understand the GAN loss function and its optimal solution. The basic idea of Generative Adversarial Network[Goodfellow *et al.* (2014)] (GAN) is to conduct a game between two neural networks. The Generator network G, takes some input z (generally noise) and transforms it into $G(z)$. Ideally, the generated outputs $G(z)$ distribution should resemble a target distribution p_r. The second network, namely the discriminator D, is optimized towards correctly identifying the

generator output $G(z)$ from real samples of target distribution p_r. This target for each network is framed as a differentiable cost function. L_G and L_D are the cost functions for generator and discriminator respectively. There expressions are given below:

$$L_D = -\mathbb{E}_{x\sim p_r} log[D(x)] - \mathbb{E}_{x\sim p_z} log[1 - D(G(x))] \tag{2.1}$$

$$L_G = \mathbb{E}_{x\sim p_z} log[1 - D(G(x))] \tag{2.2}$$

These loss functions result in a zero-sum game between the two players, and Goodfellow *et al.* (2014) have shown that there exists a unique solution to the problem. They show that the overall GAN framework tries to optimize the Jensen-Shannon divergence (JSD) between the target distribution p_r and generator ouptut distribution p_g

$$JSD(p_r \mid\mid p_g) = \frac{1}{2} KL\left(p_r \mid\mid \frac{p_r + p_g}{2}\right) + \frac{1}{2} KL\left(p_g \mid\mid \frac{p_r + p_g}{2}\right) \tag{2.3}$$

Where KL is the Kullback-Leibler divergence. Using the above expression, the optimal discriminator function D^* is

$$D^*(x) = \frac{p_r(x)}{p_r(x) + p_g(x)} \tag{2.4}$$

However, when discriminator function D becomes close to D^*, the gradients of L_G vanish.

$$\lim_{D\to D^*} \nabla_\theta \mathbb{E}_{x\sim p_r}[1 - D(G_\theta(x))] = 0 \tag{2.5}$$

This fact is terrible (and counter-intuitive), because it means that as discriminator becomes accurate, generator gets less useful information from the discriminator. For solving this vanishing gradient problem, the popularly used alternative to equation 2.2, referred to as non-saturating (NS) loss L_{NS} is given below:

$$L_{NS} = -\mathbb{E}_{x\sim p_z} log[D(G(x))] \tag{2.6}$$

Arjovsky and Bottou (2017) show that this modification alleviates the problem. When discriminator approches optimality, distribution of gradients of L_{NS} become zero

mean and infinite variance. Not only is the feedback from discriminator absent (zero mean) but make the training highly unstable (large variance).

Notice that the unstability conclusions are pivoted on the fact that discriminator *is* or *close to* its optimal solution. In unsupervised learning tasks, this assumption is very easily realised because generator tasks is much difficult compared to discriminator. From the literature using supervised learning approach, GAN instability does not appear to be a very alarming problem in practice, because we have other sources of feedback to the generator, namely pointwise loss and/or perceptual loss in the SR case, apart from adversarial loss. Our experiments with SEM images of semiconductor substrate show that, in certain cases, even supervised learning methods can be faced with unstable GAN training. Fig. 2.6 shows the SR results produced by N1, without its early stopping regularizer for the discriminator, with high values of λ. We have verified through experiments that this behaviour is due to GAN training only.

(a) (b) (c)

Figure 2.6: unstable training results obtained by N1 on dataset 2 (a) $\lambda = 0.1$ (b) $\lambda = 3.3$ (c) $\lambda = 0.5$

A possible reason for manifestation of GAN instability in our case, is the fact that semiconductor substrate dataset have very small variation across samples, and their distribution lies in a low dimensional manifold, on which discriminator can easily overfit on. With natural images with lots of variation, it is difficult for discriminator to capture all the variations and reach optimality very fast.

In supervised SR settings like ours, GAN instability is more vexing. Through experiments, we have observed that, with smaller values of λ which are generally used, the SR results and corresponding loss values look acceptable, but results have high vari-

ance. Fig. 2.7 shows the SR results for various lambda values. From these results, it is diffcult to recognize the presence of training instability. We recommend trying larger λ values in general, to detect whether the training is unstable.

(a) (b) (c) (d)

Figure 2.7: SR results obtained by N1 for small lambda values (a) λ = 0 (b) λ = 5e-3 (c) λ = 1e-2 (d) λ = 2e-2. All the results look acceptable, which makes it difficult to spot the presense of GAN instability

The previously proposed solutions employ very different methods to deal with GAN instability issue. The original N1 network uses a early stopping regularizer for discriminator which freezes the discriminator weights after some epochs, while generator continues to learn. This stops the discriminator before becoming too accurate and therefore, avoids unstable gradients. We observe empirically that stopping the discriminator training too early, results in output being very similar to *not using disriminator at all*. To really exploit the adversarial loss, we need to train discriminator for sufficient number of epochs. But tuning the discriminator epoch limit is very non-intuitive and time-consuming.

N2 uses WGAN-GP[Gulrajani *et al.* (2017)] loss for stable training. WGAN-GP uses wassertian distance[Arjovsky *et al.* (2017)] belonging to IPM family, and has been shown to be very effective at improving training stability both in theory and in practice. In our opinion, this solution is equivalent to *using a sledgehammer to crack a nut*. [Gulrajani *et al.* (2017)] show that WGAN and WGAN-GP use non-intuitive hyperparameters and their performance is very sensitive to tuning. Moreover, they require many more epochs to converge, and do not always produce results which are superior to Non-saturating GAN loss [Gulrajani *et al.* (2017), Lucic *et al.* (2017)].

One more important point to mention here is using smaller and simpler networks also alleviates GAN instability issue, as suggested by Karras *et al.* (2017). We observed

that, unlike N1, $N1_{reduced}$ does not face GAN instability, But we have already discussed in section 2.1 that reduction in network size is not a good solution in general.

CHAPTER 3

PROPOSED METHOD

In previous chapter, we discussed the major challenges that manifest when performing SR in presence of limited data. In this chapter, we provide solutions to these challenges and propose a complete solution for the task of blind defects SR.

First, We will look at a data augmentation scheme which solves the overfitting problem and avoids defect distortion issue (Section 3.1). Then we look at 3 simple changes suggested for GAN training stability (Section 3.2), and and simple extension of proposed GAN framework to work with augmentated dataset (Section 3.3). Following this, we will discuss the loss functions (Section 3.4) in detail. The results presented in this chapter are obtained by applying our proposed changes to network N1, but the flexibilty of these solutions allows them to be applied on top of any supervised SR method.

3.1 Data Augmentation

We discussed in Section 2.1 that defect distortion is due to overfitting and reducing the size of network does alleviate overfitting but produces sub-standard results due to limited capacity. We need a method to reduce overfitting without reducing network capacity. The only way to do this is to extend the dataset. But we need to be thoughful about the kind of augmentations we use. Simple geometric augmentation techniques (e.g. rotation, flipping) are not enough because it is likely that large networks continue to overfit on such extended datasets, and these transformations do not capture the actual degradation that appears in test samples: the defects.

Many unsupervised SR methods create a sythetic training dataset with degradation copied from real degraded images to clean samples, using deep learning techniques. But we do not have real defects to use such deep learning transfer methods. Hence we propose a empirically-motivated image degradation technique. In next subsection, we discuss the properties of deep-learning models underlying the proposed method.

3.1.1 Motivation

Fig. 3.1 shows a non-defective LR-HR pair each for dataset-1, 2 and 3 (see chapter 4 for dataset details). Dataset-2 is the dataset for which we see significant defect distortion when performing blind defects SR.

(a) dataset 1 (b) dataset 2 (c) dataset 3

Figure 3.1: LR-HR pairs from training set of dataset 1, 2 and 3

As an attempt to introduce synthetic defects to the training dataset of dataset-2, lets add random cropped patches LR patches from dataset-1 and 3 to random locations in dataset-2 LR images, and corresponding HR patches to corresponding locations in dataset-2 HR images. Fig. 3.2 (a) shows 2 pairs of LR-HR images obtained by such a method. Fig. 3.2 (b) shows an LR-HR pair for a real defect from testing dataset. From Fig. 3.2, it is easy to notice that our synthetically-added defects are far from real defects. Producing defects similar to real defects is a difficult task, if not impossible. Our idea is to instead just break the uniform ball mesh pattern present in non-defective images (see Fig 3.1 (b)).

Figure 3.2: dataset 2 LR-HR pairs with (a) sythetic defects (b) real defects

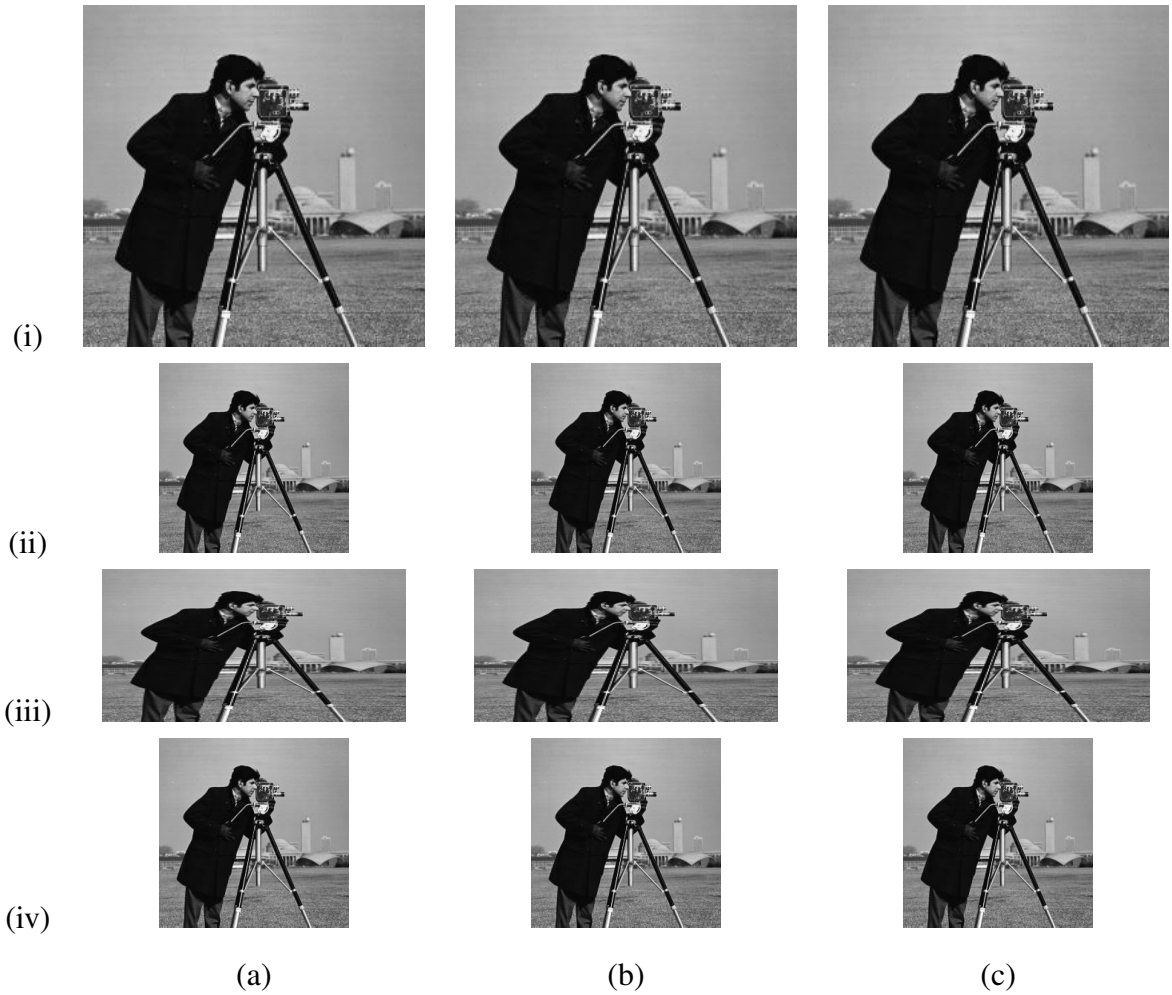


Figure 3.3: SR Results obtained by N1 on dataset 1 (a) using original clean data (b) synthetic defects data (c) ground truth HR

Fig. 3.3 shows the results produced by training N1 on dataset with and without synthetic-defects. As we can observe, the produced defects are very different in the both the results, but the background results are surprisingly similar. There are 2 more interesting points to note about produced defects:

- Although while creating synthetic-defect samples, we added the patches cropped from other datasets at random locations, we see that defects in SR results appear only at the place where real defects are present, and not anywhere in the background. This shows that, inspite of its simplicity, the defect-synthesizing method mentioned above guides the networks to accurately **detect structural degradations, namely defects**.
- The produced results look very similar to synthetic-defects cropped from dataset 1. Figure 3.4 shows how produced defects look like patches cropped from dataset 2 image. However, the background results are not affected by the use of synthetic-defects.

(a) (b)

Figure 3.4: (a) SR defects produced by N1 using synthetic defects data (b) dataset-1 HR. The annotated portion in (a) looks similar to the annoted portion with corresponding color in (b)

This above obversations remain true even if we make alterations to generator loss function. This implies that training on synthetic-defects dataset as created above trains a neural network to **decouple the SR task for defects from SR task on backgrounds, irrespective of the loss function**. It can accurately classify the LR input into defect and background regions, and performs SR on each region differently. Figure 3.5 illustrates the same idea pictorically. More importantly, the properties of SR function learnt for defect region depends highly on datasets we used for adding synthetic defect patches (dataset 2 in the example above), while the SR function for background is not affected by synthetic defects and its properties are determined by background portions of train-

ing images. So, it is possible to **control the SR performance for defects independent of SR performance of background, by changing datasets used for synthetic defects**.

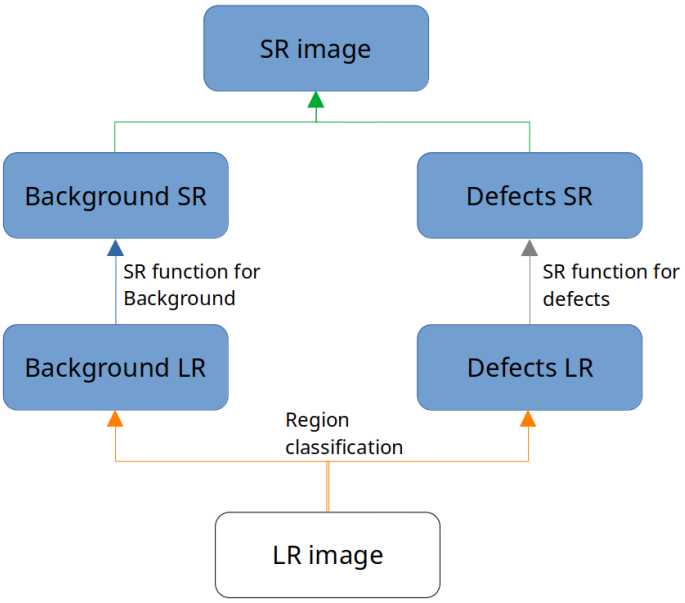


Figure 3.5: The pipeline learnt by network *implicitly* when trained with synthetic defects dataset

3.1.2 Proposed augmentation method

Based on these observations, we propose to use DIV2K dataset[Agustsson and Timofte (2017)] LR-HR pairs for LR and HR defects patches respectively. Fig. 3.6 (a) shows some samples of DIV2K dataset. DIV2K is a public dataset with diverse 2k resolution high quality images, and is used as training data for general SR tasks. Fig. 3.6 (b) shows some synthetic-defect samples generated using DIV2K dataset. Since our images are grayscale, we convert DIV2K images to grayscale. The motivation is to use DIV2k dataset for defects SR, is to encourage our network to **retain maximum information about defect structure from LR images**, in the SR output. Figure 3.7 shows the results obtained by training N1 on synthetic DIV2K-based defects. The defects are structurally accurate but do not look realistic. In section 3.3, we will see that we can get around this using GAN-based stylization of generated defects.

Figure 3.6: (a) DIV2K samples (b) dataset-2 LR-HR pairs with synthetic defects using DIV2K images

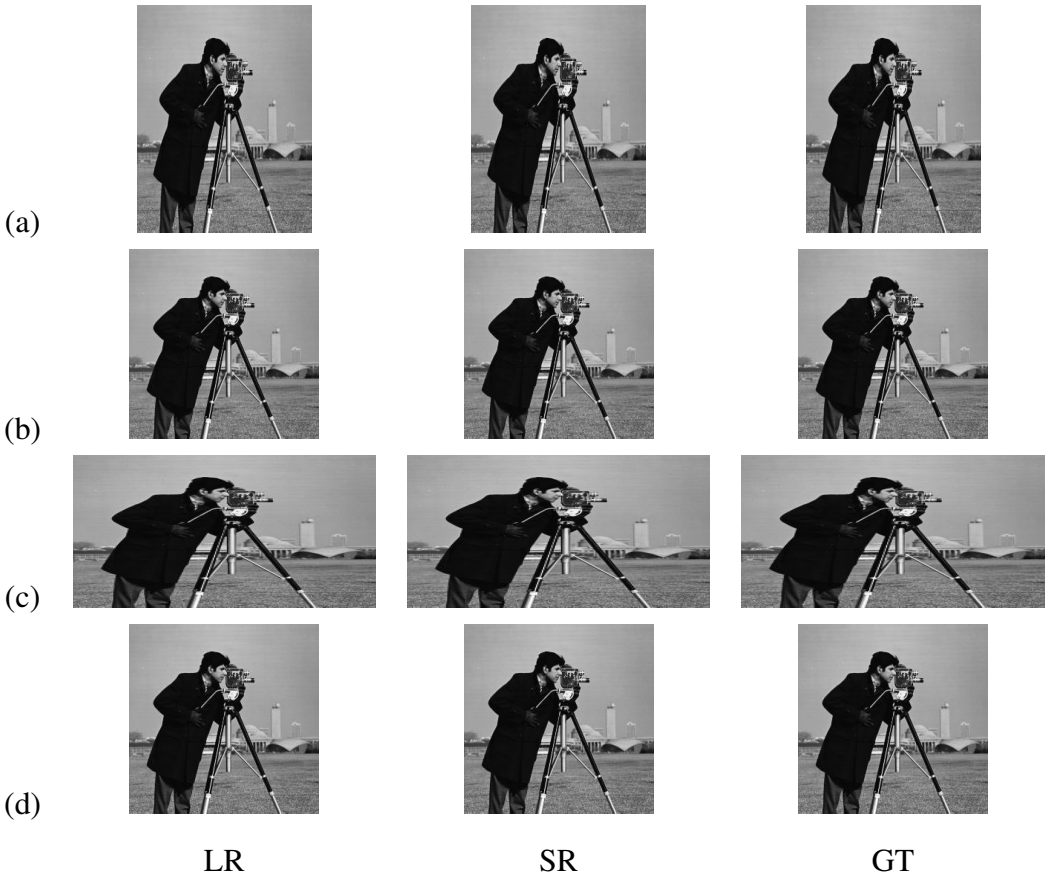


Figure 3.7: SR performance by N1 on dataset 2 defects, using DIV2K-based synthetic defects

The psuedo code for the proposed augmentation method is given algorithm 1. We take $\alpha \times \alpha$ crops from each image in the original training dataset, add β randomly

Algorithm 1: our proposed augmentation method. All experiments in the paper use $\alpha = 64$, $\beta = 128$

Input: Training dataset A with N clean images with dimensions $W \times H$,
F = dataset to crop defect patches from,
α = dimension of output images, β = number of defective images per original cropped image
Output: Training dataset R with $N.\lfloor \frac{W}{\alpha} \rfloor.\lfloor \frac{H}{\alpha} \rfloor.\beta$ defective images with dimensions $\alpha \times \alpha$

R = { }
for $x \leftarrow \alpha \times \alpha$ *cropped*(A) **do**
  for $i = 0, ..., \beta$ **do**
    $r \leftarrow$ random image from F
    $\tilde{r} \leftarrow$ random shape crop(r)
    $x_d \leftarrow$ add defect (x, $\tilde{r}$)
    $\tilde{x}_d \leftarrow$ rotate x_d by $\theta_i = \frac{\pi}{\beta}.i$
    $R \leftarrow R \cup \{\tilde{x}_d\}$

shaped patches from DIV2K dataset to each cropped image, generating β defective copies. Then, We rotate each of the β copies by different angles to generate final dataset. Rotation is crucial because apart from the synthetic defects, these copies are similar to each other.

3.2 Improving GAN stability

In Section 2.2, we discussed that SRGAN training can become unstable when using datasets with limited variation. There are various solutions to solve this problem based on regularizing the discriminator[Salimans *et al.* (2016), Arjovsky *et al.* (2017), Arjovsky and Bottou (2017), Gulrajani *et al.* (2017), Sønderby *et al.* (2016), Roth *et al.* (2017)], but they are too complicated and restrictive for a SR task. In this section, we shall discuss 3 simple modifications to GAN framework of N1 which are flexible and hyperparameter-free.

3.2.1 Relativistic GAN

Equations 2.1 and 2.6 give the Standard GAN loss function expressions, as proposed by and Goodfellow *et al.* (2014) used by N1. With Standard GAN loss, discriminator tries to maximise the probability of real samples being real and fake samples being

fake. With the relavistic GAN[Jolicoeur-Martineau (2018)] (RGAN) loss, discriminator instead maximizes the probability of real samples being more realistic than fake samples and fake samples being more fake than real samples. To achieve this we need to following modifications to the discriminator and loss function.

$$D_{SGAN}(x) = \sigma[C(x)] \rightarrow D_{RGAN}(x_r, x_f) = \sigma[C(x_r) - C(x_f)] \tag{3.1}$$

$$L_G^{RSGAN} = -\mathbb{E}_{(x_r,z)\sim(p_r,p_z)} log[D_{RGAN}(G(z), x_r)] \tag{3.2}$$

$$L_D^{RSGAN} = -\mathbb{E}_{(x_r,z)\sim(p_r,p_z)} log[D_{RGAN}(x_r, G(z))] \tag{3.3}$$

where x_r and x_f are real and fake samples respectively, fed to the discriminator, and C is pre-sigmoidal functional expression for discriminator. [Jolicoeur-Martineau (2018)] show that such a modification makes the gradients more similar to IPM-based GANs[Mroueh *et al.* (2017)], thereby, improving stability. Using equations 3.2 and 3.3 is computationally expensive. Hence, we use a more practical version of RGAN called relavistic averaging GAN (RAGAN). The RAGAN discriminator maximizes the relative probabilities on average. The discriminator and loss function for RAGAN are as follows:

$$D_{RAGAN}(x) = \begin{cases} \sigma[C(x) - \mathbb{E}_{z\sim p_z} C(G(z))] & x\ is\ real \\ \sigma[C(x) - \mathbb{E}_{x_r\sim p_r} C(x_r)] & x\ is\ fake \end{cases} \tag{3.4}$$

$$L_D^{RAGAN} = -\mathbb{E}_{x\sim p_r} log[D_{RAGAN}(x)] - \mathbb{E}_{x\sim p_z} log[1 - D_{RAGAN}(G(x))] \tag{3.5}$$

$$L_G^{RAGAN} = -\mathbb{E}_{x\sim p_z} log[D_{RAGAN}(G(x))] - \mathbb{E}_{x\sim p_r} log[1 - D_{RAGAN}(x)] \tag{3.6}$$

The expectataion in equation 3.4 can be approximated by averaging over the mini-batch. ESRGAN[Wang *et al.* (2018)] also advocates the use of RGAN but the motivation was to improve the finer details in the output. Our motivation is based on training stability.

3.2.2 Fully Convolutional Discriminator

Next change is replacing the N1 discriminator with a fully convolutional discriminator. N1 uses SRGAN[Ledig *et al.* (2017)] discriminator which has 2 dense-connected layers before final ouput layer. DCGAN[Radford *et al.* (2015)] have shown that using a fully concolutional discriminator stabalizes the GAN training. Recent state-of-the-art SR methods[Ji *et al.* (2020), Jo *et al.* (2020)] are also using fully convolutional discriminators, with the motivation to improve details and reduce unwanted artefacts. We use DCGAN discriminator suggested for 128 $\times$ 128 dimensional input [Jolicoeur-Martineau (2018)]. The architecture is given below.

$x \in$ SR/HR images
Conv2d 4x4, stride 2, pad 1, no bias, 1->64
LeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 64->128
BN andLeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 128->256
BN andLeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 256->512
BN andLeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 512->1024
BN andLeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 1024->1

3.2.3 Pretraining

In section 2.2, we mentioned that GAN training is more stable in supervised settings like SR, because generator depends more on pixelwise and perceptual loss instead of GAN loss. Pretraining is taking this idea of exploiting the supervised setting to the fullest. If we do not use pretraining (as proposed by N1), we have the discriminator training with generator from epoch 1. As in any generative task, generator starts by generating absurd results which slowly become better as training progresses. But, the absurd results from generator makes it easier for discriminator to identify it accurately

as fake sample, which leads to unstable updates. By pretraining the generator before it goes against discriminator, **makes the discriminators task more difficult, reducing the chances of becoming too accurate**.

Although the exact loss functions used for pretraining should not have too much impact, we advocate using pretraining with L1 loss, because ESRGAN[Wang *et al.* (2018)] also uses the same for its pretraining. Previous GAN methods encourage pretraining from the the viewpoint of avoiding local minima, but our experiments have shown that a simple pretraining phase has major impact on stability.

The 3 modifications suggested above: RAGAN, fully convolutional discriminator, pretraining, are very flexible in the sense that each of them can be changed independently without losing the benefits of others. Even more involved GAN loss [Arjovsky *et al.* (2017), Gulrajani *et al.* (2017), Berthelot *et al.* (2017)] functions can be used on the top of RAGAN, without lossing its benefits.

3.3 PatchGAN

Fig. 3.7 (section 3.1) shows the results obtained by N1 on dataset 2 defects when trained on DIV2K-based synthetic defects data. The defects look unrealistic because the SR function learnt from DIV2K defects merely *super-resolved* the defects. But inspecting the LR-HR pairs of dataset 2 shows that the function is more complicated than super-resolution. It also **colorizes the LR images and adds other details like shadows and a fine-grainy pattern throughout**. The DIV2K dataset does not capture these transformations, but neither do we have any other dataset which captures these transformation.

So, we need to *stylize* the defects using the dataset-2 samples, that are available to us in the training data. We suggest a **GAN-based style transfer** to the generated defects using PatchGAN [Isola *et al.* (2017)]. The PatchGAN discriminator classifies $P \times P$ patches of the input image, instead of looking at the entire image. The patch size P is chosen to be much smaller than the image dimensions, which means the receptive field $P \times P$ of the discriminator is much smaller compared to the image. This encourages GAN to only stylize the image and leave the low-frequency structural control to other losses (Perceptual and pixelwise losses). Fig. 3.8 shows the working of the PatchGAN

for our case. Notice that we feed only clean samples from training dataset to PatchGAN as real samples. We never feed synthetically-generated defective samples to PatchGAN.

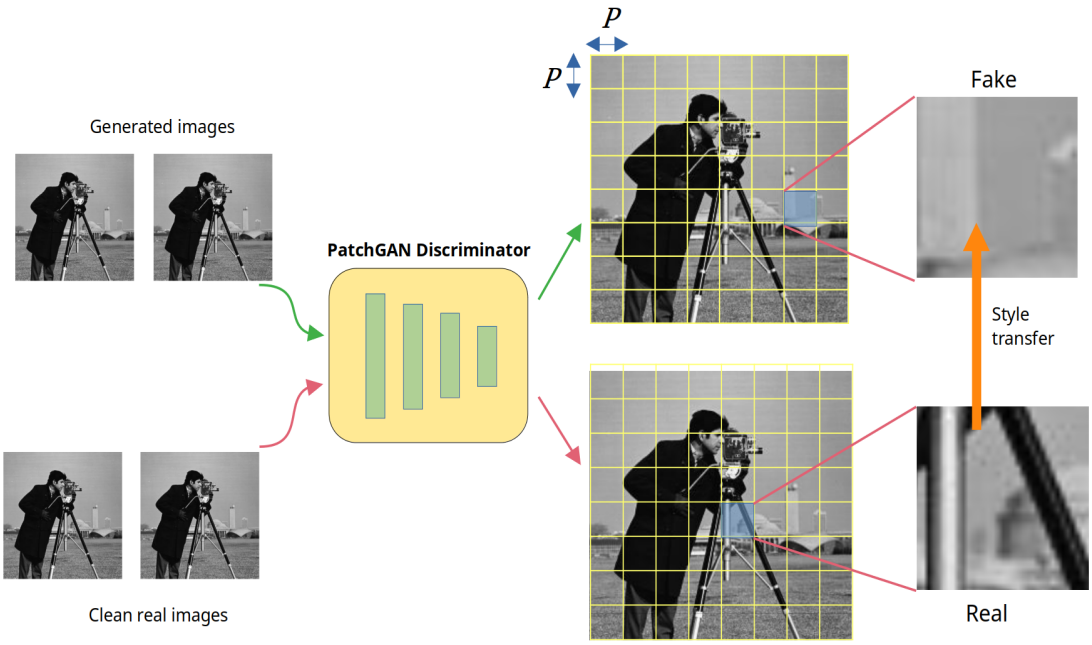


Figure 3.8: Working of PatchGAN. The discriminator looks at $P \times P$ sizes patches at a time, which results in style-transfer at patch-level

Based on the guidelines by [Ku (2019)], We propose the following discriminator architechture which has receptive field $P = 34$ pixels.

$x \in$ SR/HR images
Conv2d 4x4, stride 2, pad 1, no bias, 1->64
LeakyReLU 0.2
Conv2d 4x4, stride 2, pad 1, no bias, 64->128
BN andLeakyReLU 0.2
Conv2d 4x4, stride 1, pad 1, no bias, 128->256
BN andLeakyReLU 0.2
Conv2d 4x4, stride 1, pad 1, no bias, 256->1

Also, note that using PatchGAN to reduce the receptive field is independent of suggestions in Section 3.2. Section 3.2 proposed method to stabilize GAN training and PatchGAN is used purely for stylizing the generated defects. PatchGAN is also a fully convolutional discriminator (infact, it can be thought of as tuned DCGAN discriminator), and RAGAN and pretraining can be used with PatchGAN as well. The architecture

given in Section 3.2.2 is for cases when we do not use synthetic defects (see section 4.2.1).

3.4 Loss functions

The proposed loss functions is a slight modification of those used for N1 (equation 1.1). The overall generator loss function is formulated as:

$$L_G = (1 - \delta).L_{VGG} + \delta L_{L1} + \lambda L_{adv} \tag{3.7}$$

In this section, we discuss each of the component in detail.

3.4.1 VGG Loss

Consider $\phi_{i,j}$ the feature maps obtained by the j-th convolution (after activation) before the i-th maxpooling layer of pretrained VGG-19[Simonyan and Zisserman (2014)] network. Then, the VGG loss is defined as euclidean distance between feature maps of SR image $G(I_{LR})$ produced from LR image $I_{LR}$and ground truth HR image I_{HR}.

$$L_{VGG} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} [\phi_{i,j}(I_{HR})_{x,y} - \phi_{i,j}(G(I_{LR}))_{x,y}]^2 \tag{3.8}$$

where $W_{i,j}$ and $H_{i,j}$ are the dimensions of $\phi_{i,j}$ feature maps. The perceptual property of VGG loss has been pointed in [Johnson *et al.* (2016)]. The VGG19 network can be thought of as a filter of high-frequency details retaining only high level features of the input image. Thus, reducing VGG loss requires the generated image to match the ground truth image perceptually, and not exactly as required by pointwise losses. This gives more freedom to style losses (e.g. adversarial loss) to control the finer details independently. We use VGG19-34 (after activation) features when training on synthetic defects dataset, while VGG19-31 (after activation) for original datasets. We will study at impact of VGG loss in Section 4.3.

3.4.2 Pointwise Loss

The pointwise loss can be expressed as:

$$L_{L1} = \frac{1}{WH} \sum_{x=1}^{W} \sum_{y=1}^{H} |(I_{HR})_{x,y} - G(I_{LR})_{x,y}| \tag{3.9}$$

where W and H are the dimensions of HR/SR images. Equation 3.9 calculates the L1 distance between generated and groud truth image. Similarly to L_{L1} , we can have L_{L2} calculating euclidean distance. Empirically, we found out that both the norms have very similar behaviour, atleast as far as SR is concerned. So, we use only L1 distance in the final generator loss function, because of its superiority in avoiding artefacts as compared to L2 shown by Zhao *et al.* (2016). Apart from using VGG loss, using pointwise losses is crucial because VGG loss has some drawbacks (will be discussed in Section 4) which can be overcome by using combination of VGG and L1 loss.

3.4.3 Adversarial Loss

SRGAN[Ledig *et al.* (2017)] show that using only pointwise losses produces over-smooth results, because of convergence to the mean value of all possible HR images instead of generating a single valid image with good perceptual quality. They suggest a adversarial loss to improve the low-frequency details. For adversarial loss, We use the RAGAN loss function formulations discussed in 3.2.1 for both generator and discriminator, with all expectations approximated by empirical averages over mini-batches.

CHAPTER 4

EXPERIMENTS

We train our models on 4 different dataset provided by *KLA Corp*. These datasets contain SEM images of semiconductor substrate separated into training and testing set. Following section outline the training process for different datasets. For all our experiments, We use Nvidia GeForce RTX 2080 Ti GPU.

4.1 Dataset-2

4.1.1 Training details

It contains a total of 20 non-defective training images and 64 test images with moderate structural defects. LR image resolution is 240 x 240 and HR image resolution is 960 x 960 (4x SR) in both training and testing set. The original datasets are non-aligned and we use normalized correlation maximization to align them. Figure 4.1 shows LR-HR pair sample from training and testing set each.

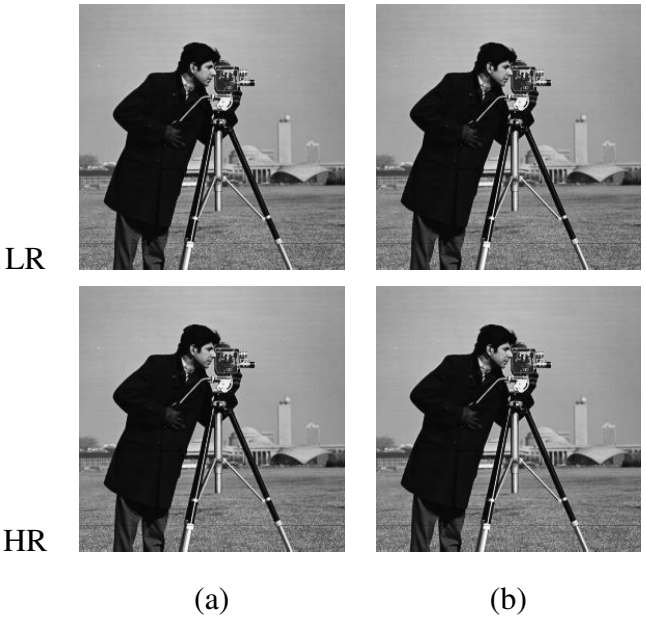


Figure 4.1: LR-HR pair for dataset-2 (a) training set (b) testing set

This dataset is more challenging compared to other datasets (discussed in Section 4.2) because of its simple topology (repetitive patterns) which can be represented in a low-dimensional manifold. Due to this, N1 exhibits significant defect distortion in SR results (Fig. 1.1). We propose synthetic-defects based augmentation specifically for this dataset. Additionally, training samples in this dataset contain higher levels of noise compared to test samples, which makes the producing test-images-like results more challenging.

For dataset-2, We use exact same generator as that of N1 (discussed in section 1.2) because of its complex dense-connected modules which allows us to learn the required LR-HR tranformation accurately. The PatchGAN architecture (section 3.3) is used for our discriminator to stylize defects. We feed(generate) grayscale images (number of channels = 1) to(from) generator.

Training dataset is created using Algorithm 1 (subsection 3.1.2). The original 240 x 240 training images form the input set A. We crop 64 x 64 size non-overlapping patches in a tiled-fashion ($\alpha = 64$) from original LR images. Similarly, we crop 64 x 64 size non-overlapping patches for DIV2K images (grayscale) and use as our defects dataset B, and use $\beta = 128$.

The training is preformed in two phases. As dicussed in subsection 3.2.3, we train the generator with L1 loss for 50 epochs. Using pre-trained generator as initialization, we perform GAN-based training using loss function in equation 3.1. with $\delta = 0.9$ and $\lambda = 0.04$ for 200 epochs. For both pre-training and GAN-based training, the learning rate is set to $\eta = 2 \times 10^{-4}$ and decayed by a factor of 0.85 every 10 epochs. Both training phases use a batch size of 16. We use Adam and SGD optimizer for generator and discriminator training respectively. The training time is around 60 hours on our system. Inference takes 0.8 secs per test image.

4.1.2 Qualitative Results

We compare our proposed solution to previously suggested methods N1 and N2 on the test dataset provided. Since the test LR-HR pairs are highly unaligned, we cannot use any quantitative metrics (e.g. PSNR, SSIM) to evaluate the results.

It can be observed from Fig. 4.2 and 4.3 that our proposed EN1 obtains much better

results in terms of both defects and background, compared to other two methods. The background is clean (not distorted) yet looks real, with all the finer details like shadows and colorization. PatchGAN works effectively in similarly stylizing our defects, without have any observable impact on their structure (compare to Fig. 3.7). Fi.4.3 (a) and (d) defects results are especially close to the ground truth HR. N1, as discusssed previously overfits and produces distorted defects. The background produced by N1 looks very clean but looks very unrealistic compared to HR. This is because of GAN training being unstable and not contributing much. N2, on the other hand, produces fairly realistic results (due to WGAN-GP) but produces lot of undesirable noise-induced artefacts due to overly small network size.

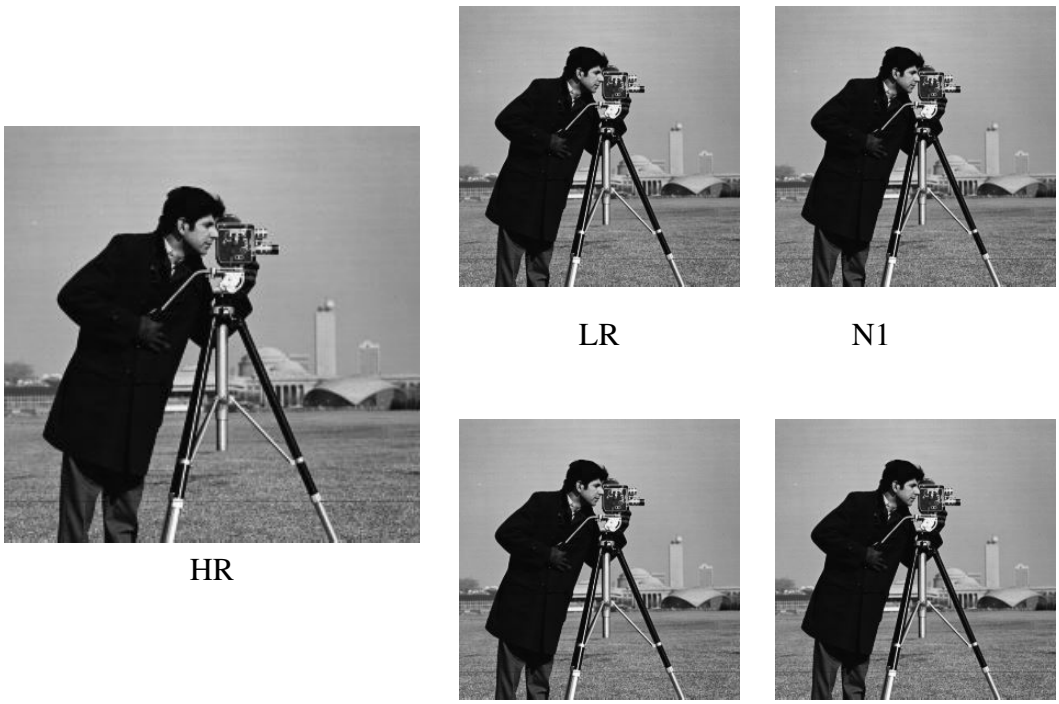


Figure 4.2: SR result of various methods on dataset-2 background

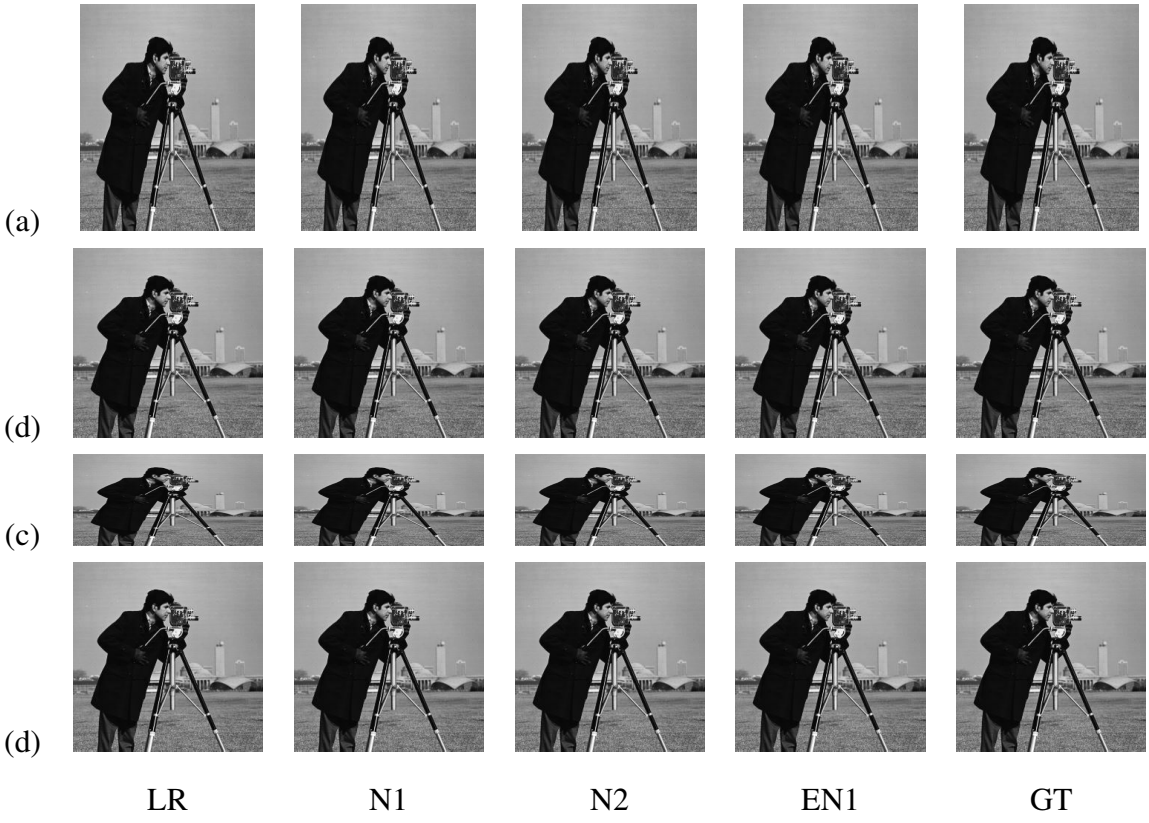


Figure 4.3: SR result of various methods on defects in dataset-2 test set

4.2 Other datasets

4.2.1 Training details

Apart from dataset-2, we have 3 other SEM datasets, namely dataset-1, 3 and 4. Table 4.1 gives their details, while Figure 4.4 show a LR-HR sample for each.

dataset	LR dimensions	HR dimensions	training set size	testing set size
1	256 x 256	1024 x 1024 (4x SR)	28	328
3	512 x 512	1024 x 1024 (2x SR)	15	606
4	128 x 128	512 x 512 (4x SR)	843	56

Table 4.1: Dataset 1,3 and 4 details

Figure 4.4: LR-HR pair for (a) dataset-1 (b) dataset-3 (c) dataset-4

These datasets are not as challenging as dataset-2, because the basic structure of the images is complex enough that training SR methods on original images generalizes well to the test set. So, we do not recommend using synthetic data augmentation for these datasets. Instead, only simple rotation augmentation is sufficient to produce realistic results. Also, unlike dataset-2, test and train images have similar noise characteristics.

Using large networks like N1, we still observe unstable GAN training even on these datasets. Hence, we propose changes described in Section 3.2. The DCGAN discriminator from subsection 3.2.2 should be used because we are not using synthetic datasets.

We crop 64 x 64 size non-overlapping patches in a tiled-fashion ($\alpha = 64$) from original LR images. We rotate these by 16 different angles to generate final training dataset. For dataset 1 and 3, we use $\delta = 0.7$, while we use only VGG loss for dataset-4 because it has *brightness inconsistency* between LR-HR images. There are samples in dataset-4 for which at some portion of LR image where it is brightly lit, HR image is dark, and for some other samples, the visa versa. We use $\lambda = 0.06$ for all 3 datasets. All other details remain as a those mentioned in subsection 4.1.1.

4.2.2 Qualitative Results

We compare our proposed solution to previously suggested methods N1 and N2 on the test dataset provided. Although LR-HR test pairs are aligned for these dataset, we still avoid comparing based on quantitative metrics for uniformity and simplicity. The real quantitative test for our SR results is to defect-detection performance (as on the date of submission of this work, this is yet to be measured).

Fig. 4.5, 4.6 and 4.7 show results for dataset 1, 4 and 3 respectively. Our methods improves the realistic look of the outputs. Due to small size, N2 produces lots of noise-induced artefacts for dataset-2, while N1 can filter most of the artefcts, but some striking ones are still visible in outputs. EN1, due to its better GAN loss, can filter these artefacts even more effectively. Since the dataset-3 LR-HR mapping is very simple (only requires super-resolution, unlike dataset-2 which requires super-resolution, colorization and shadow-effects), all methods perform satisfactorily on it, but EN1 results are slightly better. For dataset-4, we see most improvement in fine details and brightness levels. But brightness cannot be made very accurate because of LR-HR brightness inconsistency in the training data.

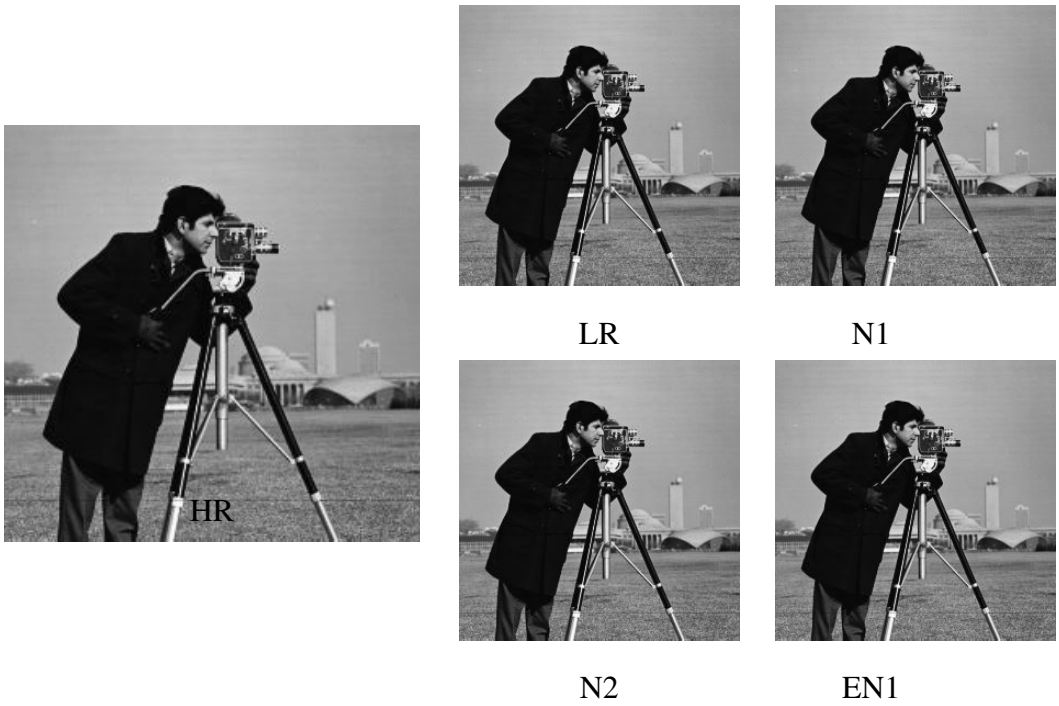


Figure 4.5: SR results obtained by various methods on dataset-1. The highlighted region shows noise-artefacts in N1 and N2 results. EN1 does not produce such artefacts.

LR

N1

EN1

HR

LR

N1

EN1

HR

Figure 4.6: SR results obtained by various methods on dataset-4

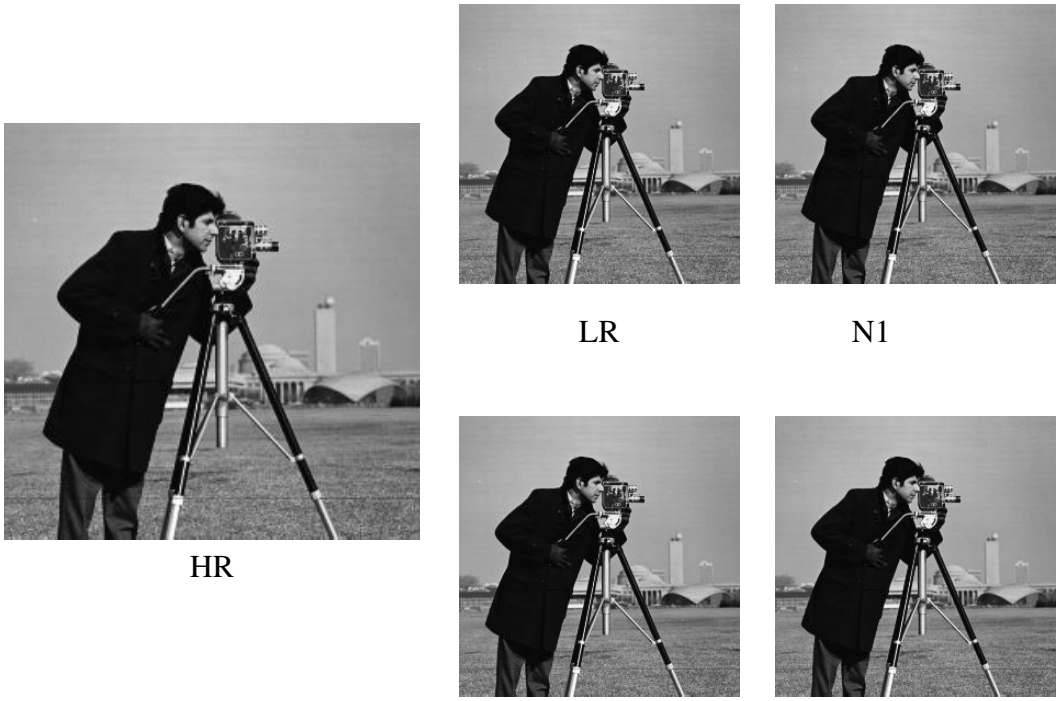


Figure 4.7: SR results obtained by various methods on dataset-3

4.3 VGG Loss Study

We conducted a study to better understand the effects of using VGG Loss as opposed to pixelwise losses. For this, we use our proposed EN1 network on dataset-2 and remove the adversarial loss (set λ = 0) to study the effects independent of adversarial loss. We have experimentally verified that L1 and L2 loss have very similar behaviour even in presence of VGG loss. So, we present the results only for VGG and L1 loss combination, but same inferences hold true for VGG and L2 loss combination. Fig. 4.8 show the SR results obtained by δ = [0.0, 0.3, 0.7, 0.9. 1.0].

Brightness δ stands for the percentage of VGG loss in VGG-L1 combination. With $\delta = 0$, the brightness is not accurate. But adding a small amount of L1 loss, at $\delta = 0.9$, we see huge improvement in the same. This shows VGG loss cannot produce accurate brightness on its own, but needs another loss to take care of it. Similar behaviour is observed by Wang *et al.* (2018). Such a behaviour probably stems from how pixel values in different regions of the image are scaled differently in feature maps used for

VGG loss. We have observed this behaviour to cause contrast variance in presence of adversarial loss. So small amounts of L1/L2 loss should always be used in combination to VGG loss.

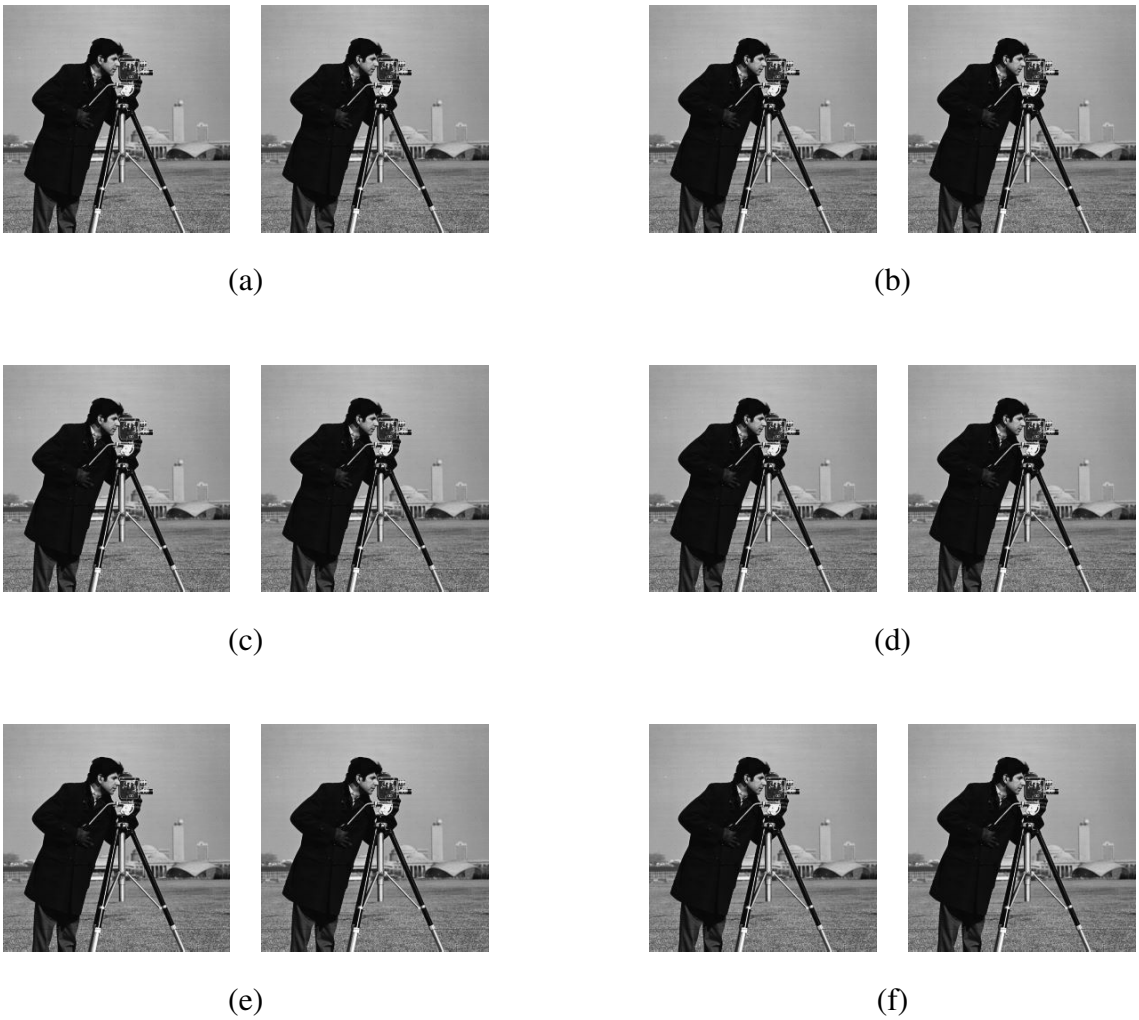


Figure 4.8: EN1 (GAN ablated) results on dataset-2 using (a) $\delta = 0.0$ (b) $\delta = 0.3$ (c) $\delta = 0.7$ (d) $\delta = 0.9$ (e) $\delta = 1.0$ (f) HR

Over-clean results With increasing δ, we also see SR outputs looks overly clean, and finer detais are missing. It probably result of VGG loss filtering high-frequency details in the image. This produces unrealistic looking results with higher level of VGG loss. But, this issue can be solved with adversarial loss adding the finer details.

Checkboard pattern Using high percentage of VGG loss also adds checkboard pattern, which is also observed for VGG-based texture loss[Waleed Gondal *et al.* (2018)]. Adversarial loss also solves this problem to some extent, but this issue can still result in

repetitive patterns in presense of adversarial loss [Jo *et al.* (2020)].

Although, the VGG loss has many drawbacks it is essential to use a perceptual loss which give finer level control to adversarial loss. Without the perceptual property of VGG loss, we cannot achieve the stylization of defects. Apart from GAN-friendliness, over-cleanliness property of VGG loss can produce visually more appealing results, but it is completely a subjective matter.

CHAPTER 5

CONCLUSION

In this thesis, we explore the various challenges involved in super-resolving SEM images in face of limited data. We show that directly using state-of-the-art methods causes defects distortion due to overfitting, and instead of reducing network size, we should augment the dataset to include the degradations. We saw that GAN training can become unstable in supervised settings with limited data, but can be stabilized using simple modifications. We also discussed various properties of VGG loss as a perceptual loss for SR.

We suggest following directions for future work in blind-defects SR:

- For our use-case (KLA-SR), we did not require styling our synthetic-defects dataset to match the characterics of LR background. In a general case, style transfer techniques should be adopted for LR synthetic-defects so that they blend in with the LR background.
- Use of latest style transfer tenchinques instead of simple adversarial loss should be adopted to improve the output-defects stylization.
- In this thesis, We have considered a *hard* version of blind-defects SR problem where we do not use real defects at all. A *soft* version can be considered where limited number of real defects samples can be used to create synthetic-defects using deep learning methods, similar to unsupervised SR methods.
- For styling output-defectss, perceptual loss plays an important role. In section 4.3, we discussed the drawbacks of VGG loss. Searching a better perceptual loss which overcomes these drawbacks, can improve the blind-defects SR.
- Developing a quantitative metric which works with unaligned images, might help to direct future work.

APPENDIX A

Network $N1_{reduced}$

In section 2.1, we discuss network size reduction as a way to reduce overfitting on training dataset. Along this line, we modified the Network N1 (section 1.2) to reduce its number of weight parameters. Apart from this, we use this tuned various other parameters to achieve best results. We call this tuned network as $N1_{reduced}$. All the features of this network are listed below:

1. The generator is still an ESRGAN[Wang *et al.* (2018)] generator, with B RRDB blocks and 64 channels. The value of B is chosen to be smaller that 8.
2. A reduced SRGAN[Ledig *et al.* (2017)] discriminator is used, with only 2 convololutional layers and starting with 16 channels.
3. VGG Loss is removed from the generator loss function, and is formulated as follows, instead:
$$L_G = L_{L1} + 0.06L_{adv}^{RAGAN} \tag{A.1}$$
where L_{L1} is the L1 loss discussed in section 3.4.2 and L_{adv}^{RAGAN} is the RAGAN loss discussed in section 3.2.1.
4. The training dataset is augmented by using simple stretching and sheering sransformations to help the network learn LR-HR structure correlation.
5. No pretraining and discriminator regularizer is required, because of the small network sizes.

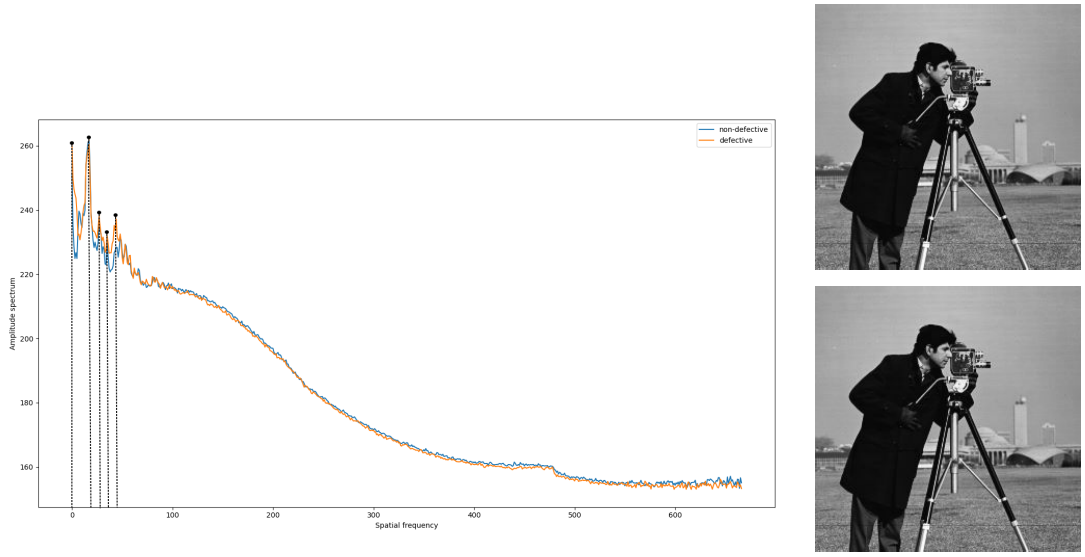
APPENDIX B

Frequency-based View of Synthetic defects

In section 3.1, we motivated our proposed data sugmentation method using empirical results. In this chapter, we shall discuss an alternate view of the same solution based on frequency, and extend the solution to solve blind-defects SR problem for general datasets. Please note that we do not need the extended solution for KLA-SR dataset 2 (See section 4.1).

B.1 Frequency view of Defects distortion problem

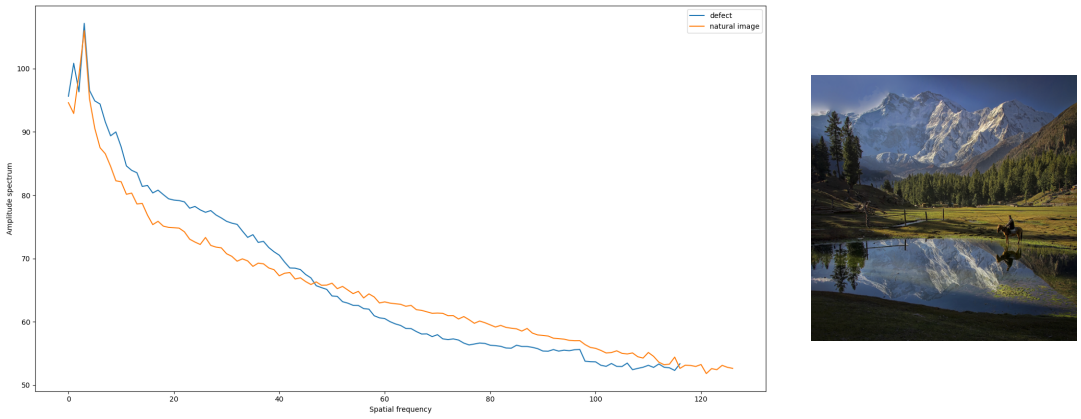
In section 2.1, we mentioned that the reason for defects distortion (and lost defects in certain cases) is overfitting. Now let us look at frequency-domain view of this problem. Fig. B.1(a) shows amplitude spectrum of non-defective and defective HR images from Dataset-2, shown in Fig. B.1(b). The x-axis of the plot represents spatial frequency and the amplitude spectra are obtained by averaging the 2D spectra azimuthally, and is plotted as dBs.



(a) (b)

Figure B.1: (a) Amplitude spectrum of Dataset-2 defective and non-defective HR samples (b) Samples used for spectrum plots.

Due to the repeating patterns, we see several discrete frequency peaks in low frequency bands of both the spectra. Fig. B.1 (a) highlights them with dashed lines. One noticable difference between the two spetra is how the power in low-pass band is increased due to presence of a defect. This is expected because defects do not have any repeating pattern in particular, and high frquency content is similar to that of background. Fig. B.2 shows amplitude spectrum of defect cropped, and compares it to amplitude spectra of natural image patch taken from a DIV2K[Agustsson and Timofte (2017)] image used. We can observe that the effect brought by a defect to the image has very similar frequency charactericstics to that of natural image, especially in the very low frequency band. This explains why DIV2K is a good choice for defects dataset.



(a) (b)

Figure B.2: (a) Amplitude spectrum of cropped defects and natural image (b) Natural image used.

One more thing to note in Fig. B.1 and Fig. B.2, which was already apparent, is that high frequency content of defect and background are same. To summarize, **defects has low frequency characteristics of natural image and high frequency characteristics of background**. With this insight, we can extend the solution to take advantage of these defect charateristics. Although, we showed the results for only HR, similar properties hold for defects in LR space.

B.2 Systematic degradation in LR space

In section 3.1, we discussed that we can degrade the both LR and HR images by adding random-shaped patches from natural image space. But as discussed in previous section,

defects only have low frequency characteristics of natural images. For our synthetic LR defects to mimic original LR defects completely, the synthetic defects need to match the high frequency characterstics of background. For Dataset-2, we did not require this process because it seems that LR background fairly matches natural images in high frequency characteristics, but this may not be the case in general.

[Fritsche *et al.* (2019)] suggest handling different frequency bands using separate specialised loss functions. The advantage is that optimization of each loss component becomes easier because of a focused target for each component, and no interference between different loss functions. Following the same philosophy, we can use following loss function to transfering high frequency characteristics of LR defects before our proposed data sugmentation method (section 3.1)

$$L_G = \delta\ L_{perc} + (1-\delta)\ L_{cont} + \lambda_1\ L_{tex} + \lambda_2\ L_{col} \tag{B.1}$$

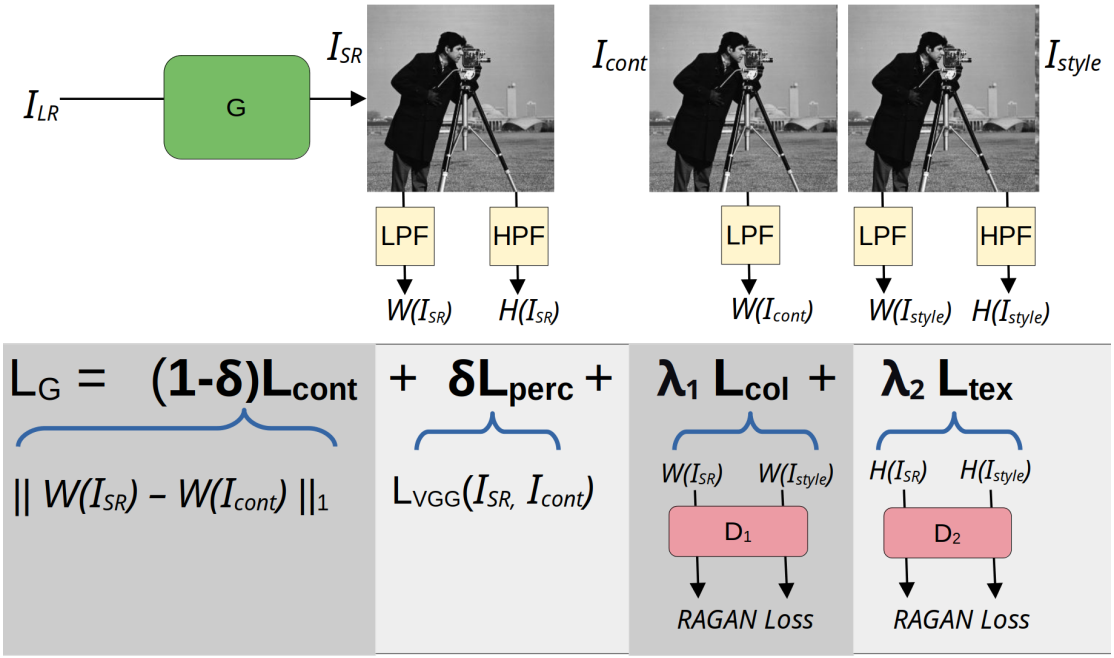


Figure B.3: Procedure for low-frequency degradation

Fig. B.3 illustrates the training framework. It is very similar to ESRGAN framework, with only difference being that there is no change in dimensions between input and output of the generator G, and use of two disriminators, one optiizing color loss and . In fact, we use the same ESRGAN[Wang *et al.* (2018)] generator (with upsampling layers removed) (section 1.2) and PatchGAN architectures for G and discriminators D_1,

D_2 (section 3.3). In next subsections we elaborate on the each loss function component.

B.2.1 Content Loss

The content loss can be calculated as:

$$L_{cont} = L_1 \left[W(I_{nat}), W(I_{deg})\right] \tag{B.2}$$

where L_1 is the L1 loss function as defined in equation 3.9, I_{nat} is the natural image whose high frequency components are to be altered, $I_{deg} = G(I_{nat})$ is the generated output. W is a low-pass filter. As suggested by Fritsche *et al.* (2019), we use 5×5 moving average filter for this. Unlike L_1 loss, L_{cont} matches only the low frequency contents on the I_{nat} and I_{deg}, thereby not interfering with L_{tex} which is responsible for high frequency details.

B.2.2 Texture Loss

The loss function expression is similar to that used previously (equation 3.4, 3.5 and 3.6), with only difference being instead of passing $G(I_{nat})$ and I_{style}, we pass high-pass filtered versions, $H(G(I_{nat}))$ and $H(I_{style})$ to the discriminator D_2. The high-pass filter is simply calculated as $H(I) = I - W(I)$. Since, the main motivation to use GAN is for learning high frequency details, passing exactly the same makes optimization easier.

B.2.3 Color Loss

We add another GAN-based loss function which takes only the low-pass versions of I_{nat} and I_{style}. Why do we do this? This is used to make sure that color statistics of generated images matches that of background image. It is possible that color statistics of SEM image are skewed compared to natural images. Using a low receptive fields patchGAN on low-frequency content encourages generated image color distribution to match the background[Isola *et al.* (2017)]. We use a separate discriminators to learn color and high frequency distribution, instead of using a single GAN to handle both (by passing the unfiltered I_{nat} and I_{style} to it), because separating the target makes

optimization for each GAN easier, and gives increased control through hyperparameter tuning.

B.2.4 Perceptual loss

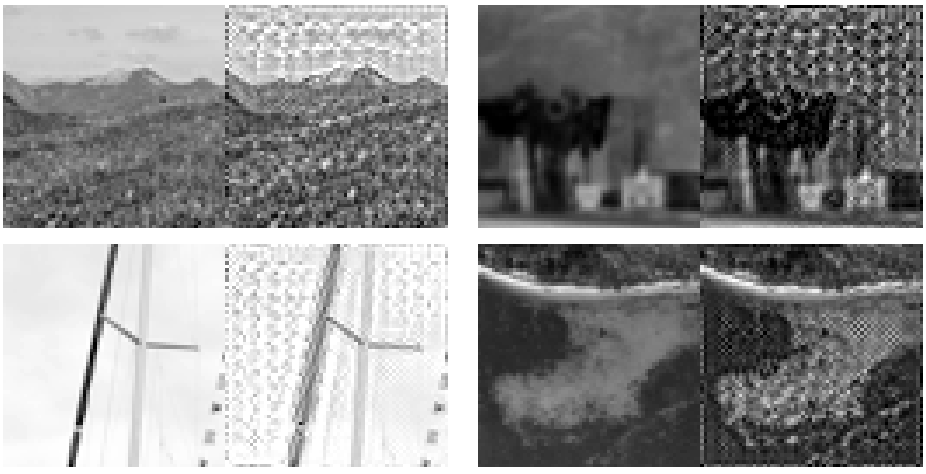
The perceptual loss is calulated as :

$$L_{perc} = L_{VGG}\left[I_{nat}, I_{deg}\right] \tag{B.3}$$

Where L_{VGG} is the VGG19-based perceptual loss from equation 3.8. This loss function also works on matching high level features like L_{cont}, but is used to make sure that various loss functions fit together to produce a consistent image.

B.2.5 Qualitative results

We try our degradation method on Dataset-2 just as an illustration of the idea. As mentioned earlier, for Dataset-2, directly using natural images suffices, but we perform the experiments for demonstration of the suggested method. For the experiments, we use 8 RRDB ESRGAN generator with 64 channels, and both generator use PatchGANs with 9×9 receptive field. Hyparameters values are $\delta = 0.1$, $\lambda_1 = \lambda = 0.06$ .From Figure B.4, We can see high frequency and color characteristics being transformed to those of dataset-2 LR. Now, these modified samples can be used as defects dataset.



natural transformed natural transformed

Figure B.4: High frequency characterictics transfer results on dataset-2 LR

B.3 Frequency-Separation based super-resolution

We can use the process discussed in section B.2 to transfer high-frequency characteristics from HR images to natural images before cropping patches from it. A better approach is to rather perform frequency separation based super-resolution and transfer the high-frequency characterstics to HR domain defects while performing super-resolution. This reduces overall training time of the complete process, and avoids discriminator overfitting on static generated noise. Fig. B.5 illustrates the SR framework with frequency separation. For experiments, we use 9×9 moving average filter as low-pass filter and same G and D architecture for generator and both discriminator as mentioned in section 4.1.2. We used hyperparameters $\delta = 0.9, \lambda_1 = \lambda_2 = 0.04$. Experiments till the date of submission of this work do not produce any significant improvement over results shown in Section 4.1.2, but again the motivation is to have more control for better generalization.

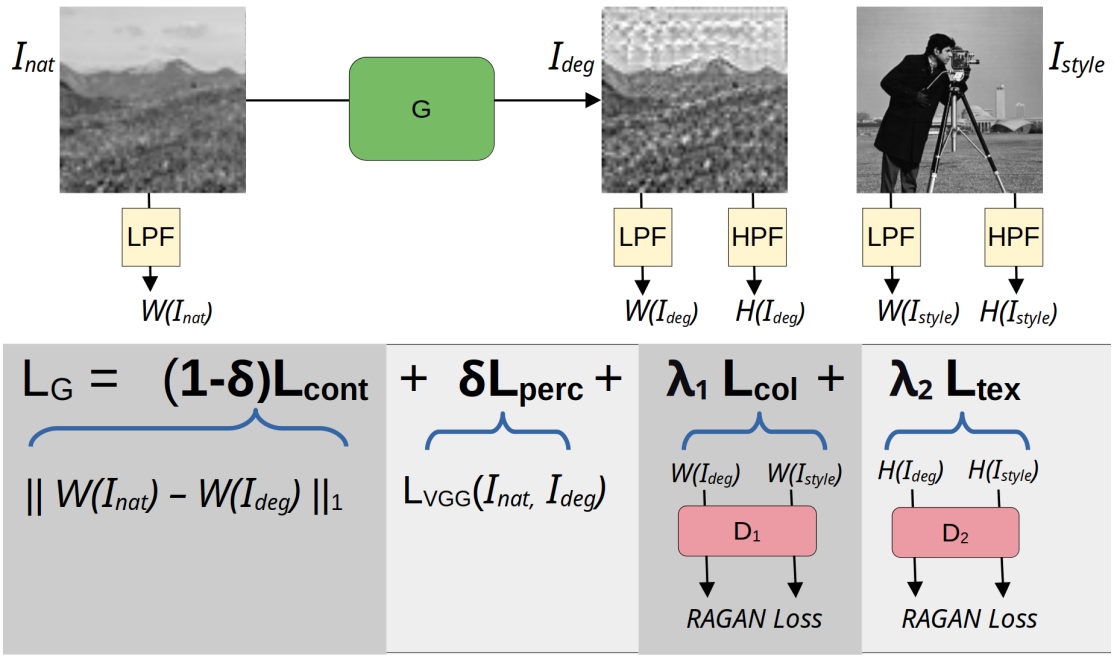


Figure B.5: Procedure for SR with frequency-separation

REFERENCES

1. **Agustsson, E.** and **R. Timofte**, Ntire 2017 challenge on single image super-resolution: Dataset and study. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017.

2. **Arjovsky, M.** and **L. Bottou** (2017). Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*.

3. **Arjovsky, M.**, **S. Chintala**, and **L. Bottou** (2017). Wasserstein gan.

4. **Bell-Kligler, S.**, **A. Shocher**, and **M. Irani** (2019). Blind super-resolution kernel estimation using an internal-gan. *arXiv preprint arXiv:1909.06581*.

5. **Berthelot, D.**, **T. Schumm**, and **L. Metz** (2017). Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*.

6. **Bulat, A.**, **J. Yang**, and **G. Tzimiropoulos**, To learn image super-resolution, use a gan to learn how to do image degradation first. *In Proceedings of the European conference on computer vision (ECCV)*. 2018.

7. **Chen, S.**, **Z. Han**, **E. Dai**, **X. Jia**, **Z. Liu**, **L. Xing**, **X. Zou**, **C. Xu**, **J. Liu**, and **Q. Tian**, Unsupervised image super-resolution with an indirect supervised path. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020.

8. **Dai, T.**, **J. Cai**, **Y. Zhang**, **S.-T. Xia**, and **L. Zhang**, Second-order attention network for single image super-resolution. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

9. **Dong, C.**, **C. C. Loy**, **K. He**, and **X. Tang** (2015). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, **38**(2), 295–307.

10. **Fritsche, M.**, **S. Gu**, and **R. Timofte**, Frequency separation for real-world super-resolution. *In 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2019.

11. **Goodfellow, I. J.**, **J. Pouget-Abadie**, **M. Mirza**, **B. Xu**, **D. Warde-Farley**, **S. Ozair**, **A. Courville**, and **Y. Bengio** (2014). Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.

12. **Gulrajani, I.**, **F. Ahmed**, **M. Arjovsky**, **V. Dumoulin**, and **A. Courville** (2017). Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*.

13. **Haris, M.**, **G. Shakhnarovich**, and **N. Ukita**, Deep back-projection networks for super-resolution. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

14. **Isola, P.**, **J.-Y. Zhu**, **T. Zhou**, and **A. A. Efros**, Image-to-image translation with conditional adversarial networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

15. **Ji, X.**, **Y. Cao**, **Y. Tai**, **C. Wang**, **J. Li**, and **F. Huang**, Real-world super-resolution via kernel estimation and noise injection. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020.

16. **Jo, Y.**, **S. Yang**, and **S. J. Kim**, Investigating loss functions for extreme super-resolution. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020.

17. **Johnson, J.**, **A. Alahi**, and **L. Fei-Fei**, Perceptual losses for real-time style transfer and super-resolution. *In European conference on computer vision*. Springer, 2016.

18. **Jolicoeur-Martineau, A.** (2018). The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*.

19. **Karras, T.**, **T. Aila**, **S. Laine**, and **J. Lehtinen** (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

20. **Kim, J.**, **J. K. Lee**, and **K. M. Lee**, Accurate image super-resolution using very deep convolutional networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

21. **Ku, E.** (2019). Image-to-image / pix2pix. `https://medium.com/@EricKuy/image-to-image-pix2pix-e690098231fd`.

22. **Ledig, C.**, **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. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

23. **Lim, B.**, **S. Son**, **H. Kim**, **S. Nah**, and **K. Mu Lee**, Enhanced deep residual networks for single image super-resolution. *In Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2017.

24. **Lucic, M.**, **K. Kurach**, **M. Michalski**, **S. Gelly**, and **O. Bousquet** (2017). Are gans created equal? a large-scale study. *arXiv preprint arXiv:1711.10337*.

25. **Lugmayr, A.**, **M. Danelljan**, and **R. Timofte**, Unsupervised learning for real-world super-resolution. *In 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2019.

26. **Mroueh, Y.**, **C.-L. Li**, **T. Sercu**, **A. Raj**, and **Y. Cheng** (2017). Sobolev gan. *arXiv preprint arXiv:1711.04894*.

27. **Rad, M. S.**, **B. Bozorgtabar**, **U.-V. Marti**, **M. Basler**, **H. K. Ekenel**, and **J.-P. Thiran**, Srobb: Targeted perceptual loss for single image super-resolution. *In Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

28. **Radford, A.**, **L. Metz**, and **S. Chintala** (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

29. **Roth, K.**, **A. Lucchi**, **S. Nowozin**, and **T. Hofmann** (2017). Stabilizing training of generative adversarial networks through regularization. *arXiv preprint arXiv:1705.09367*.

30. **Salimans, T.**, **I. Goodfellow**, **W. Zaremba**, **V. Cheung**, **A. Radford**, and **X. Chen** (2016). Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*.

31. **Shi, W.**, **J. Caballero**, **F. Huszár**, **J. Totz**, **A. P. Aitken**, **R. Bishop**, **D. Rueckert**, and **Z. Wang**, Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

32. **Shocher, A.**, **N. Cohen**, and **M. Irani**, "zero-shot" super-resolution using deep internal learning. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

33. **Simonyan, K.** and **A. Zisserman** (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

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

35. **Tai, Y.**, **J. Yang**, and **X. Liu**, Image super-resolution via deep recursive residual network. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

36. **Ulyanov, D.**, **A. Vedaldi**, and **V. Lempitsky** (2016). Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

37. **Vasu, S.**, **N. Thekke Madam**, and **A. Rajagopalan**, Analyzing perception-distortion tradeoff using enhanced perceptual super-resolution network. *In Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018.

38. **Waleed Gondal, M.**, **B. Scholkopf**, and **M. Hirsch**, The unreasonable effectiveness of texture transfer for single image super-resolution. *In Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018.

39. **Wang, X.**, **K. Yu**, **S. Wu**, **J. Gu**, **Y. Liu**, **C. Dong**, **Y. Qiao**, and **C. Change Loy**, Esrgan: Enhanced super-resolution generative adversarial networks. *In Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018.

40. **Yuan, Y.**, **S. Liu**, **J. Zhang**, **Y. Zhang**, **C. Dong**, and **L. Lin**, Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018.

41. **Zhang, W.**, **Y. Liu**, **C. Dong**, and **Y. Qiao**, Ranksrgan: Generative adversarial networks with ranker for image super-resolution. *In Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

42. **Zhang, Y.**, **K. Li**, **K. Li**, **L. Wang**, **B. Zhong**, and **Y. Fu**, Image super-resolution using very deep residual channel attention networks. *In Proceedings of the European conference on computer vision (ECCV)*. 2018.

43. **Zhao, H.**, **O. Gallo**, **I. Frosio**, and **J. Kautz** (2016). Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, **3**(1), 47–57.

44. **Zhou, R.** and **S. Susstrunk**, Kernel modeling super-resolution on real low-resolution images. *In Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

45. **Zhu, J.-Y.**, **T. Park**, **P. Isola**, and **A. A. Efros**, Unpaired image-to-image translation using cycle-consistent adversarial networks. *In Proceedings of the IEEE international conference on computer vision*. 2017.