Super-Resolution of SEM Images

A Project Report

submitted by

Subankar Chakraborty

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

July 2022

Thesis Certificate

This is to certify that the thesis titled **Super-Resolution of SEM Images**, submitted by **Subhankar Chakraborty**, 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.

Professor A N Rajagopalan
Research Guide
Professor
Department of Electrical Engineering
IIT Madras, 600036

Place: Chennai

Date: July 2022

Acknowledgements

Throughout the work for this project, I have received a great deal of support and assistance from many people. I would first like to thank my guide, **Prof. A N Rajagopalan** who trusted me with this challenging problem. His expertise, insights, and constructive feedback have been vital for the project's success. I would like to thank the **KLA team** for allowing me to work on their problem statement and for all their time and feedback. I would like to thank the **Samsung Pravartak Award** for their scholarship grant. I would also like to thank **Nikam Ashutosh Shashikant**, a former Dual Degree student at the IPCV Lab, for providing me with the source code for his previous solutions and guiding me in the initial stages of the project. The current work would have been much slower without his directions. Last but not least, I would like to thank my family and friends for their constant support and belief in me during these challenging times.

Abbreviations

SEM	Scanning Electron Microscope
SR	Super Resolution
SRd	Super Resolved
LR	Low Resolution
HR	High Resolution
SISR	Single Image Super Resolution
IR	Image Restoration

Abstract

SISR has seen a lot of progress in the recent past. However, most SISR methods are trained by generating the LR by bicubically downsampling the HR images, hampering their generalisability to real-world LR images. Blind SR is another domain which has received attention in the recent past. Blind SR methods use internal learning to train a network for each inference image. However, if there is a domain gap between the LR and HR images, they need to be supported by a style transfer method to match the look of the SRd images to those of the HR images. Reference-based SR is another area which has received attention recently. Those methods take in a reference image along with the LR image to try and recover the SR image. This report presents an upgradeable pipeline that uses a Blind SR method followed by a style transfer method. When the throughput is of paramount importance, we suggest ways to eliminate the dependence on Blind SR methods, which are slow. In that case, we rely purely on domain transfer. We consider a more challenging version of the previous problem where we do not assume the presence of paired data and rely on unsupervised methods. The proposed method(s) produce near realistic HR images, matching or even beating the performance of the previous method in multiple cases while being a lot more interpretable and less hacky.

Table of Contents

List of Figures

Chapter 1

Introduction

Minor defects can greatly impact the safety and function of semiconductor devices. Since component failure is often a direct result of numerous microscopic defects, their multi-scale observation and quantification are the only way to determine the root cause and accurately characterize these faults. Our electronic devices are getting smaller and more complex, giving failure-inducing defects more places to hide. To make matters worse, these failures often occur late in the fabrication process, resulting in high yield losses and significant delays in time-to-market for semiconductor manufacturers. As a result, researchers must identify failures as quickly and accurately as possible to resume production and reduce the likelihood of such issues returning.

An SEM is an electron microscope that produces images of a sample by scanning the surface with a focused beam of electrons. The electrons interact with atoms in the sample, producing various signals containing information about the sample's surface topography and composition. The electron beam is scanned in a raster scan pattern, and the beam's position is combined with the intensity of the detected signal to produce an image. SEMs are powerful, high-resolution tools that are widely used for a vast range of failure analyses. They provide the magnification and depth of field required to accurately analyze faults and identify failures.

With the progress in deep learning, object detection networks are often used to locate faults within SEM captures of the semiconductor wafers. Given the massive increase in the number of devices that need semiconductors, the fault detection process needs to be efficient and accurate.

One way to make fault identification faster is to capture as little data as possible. This is often achieved by capturing SEM images at lower and lower resolutions. However, since the defect detection networks are trained to work with data at higher resolutions, these LR images often need to be SRd before they are useful for the fault detection networks. This project deals with the SR part of the problem. KLA has been very kind in providing us with four datasets. The focus has been constrained to dataset-2 as it is the most challenging of the lot and still needs a satisfactory solution.

1.1 Dataset Description

We have been provided with multiple datasets from KLA. The goal is to be able to super-resolve the LR images so that they can be used with the defect detection networks. We are blind to the defect-detection networks. The focus has been constrained to dataset-2 and all results presented are on that dataset. The LR and HR images in that dataset do not look similar due to the different scan rates. Any SR model hence must be able to match this discrepancy apart from performing SR. A few samples from the LR and HR images of the training set have been shown in figures 1.1 and 1.2 respectively.

Figure 1.1: LR Train Images

Figure 1.2: HR Train Images

It is to be noted that the images are not to scale. In the real data, the HR images are $4\times$ the LR image in each dimension. The HR images are of the size 960×960, whereas the LR images are of the size 240×240. We have 20 such LR and HR images each. They are not necessarily aligned. If paired data is needed for an experiment, we must align them manually or write a script to align them automatically. Likewise, we have 64 test LR images of the same size as the train LR images. We also have 70 HR images

(of the same size as the train HR images) provided with the test set. But, these must be used only as a guide to see how good the model's performance is on the LR images and not used as ground truth data
In later experiments, when we talk about training networks with the LR images bicubically upsampled along with the HR images, we upsample both the LR and HR to be of the size 1024×1024. This is because many domain transfer networks expect the input image dimensions to be divisible by 4 and, in some cases, to be a power of 2 too.

1.2 Challenges

A few LR samples from the test set and their reference HR images have been shown in figures 1.3 and 1.4. It is to be noted that the HR images correspond to a similar LR image, and not the exact same LR image. Hence, there will not be structurally identical.

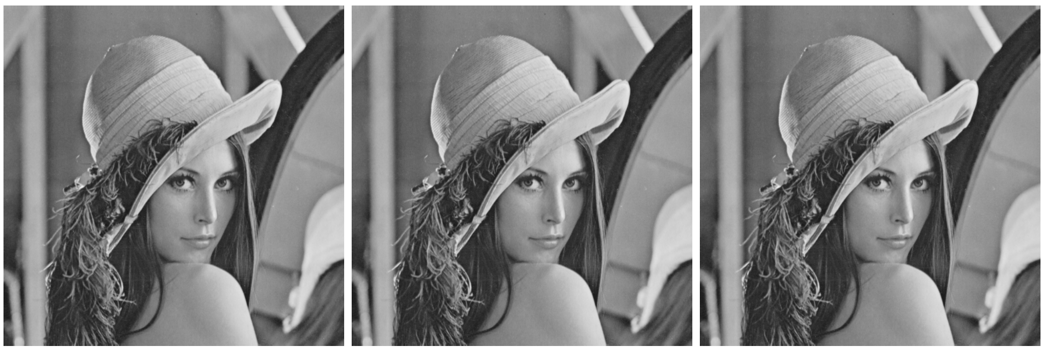


Figure 1.3: LR Test Images

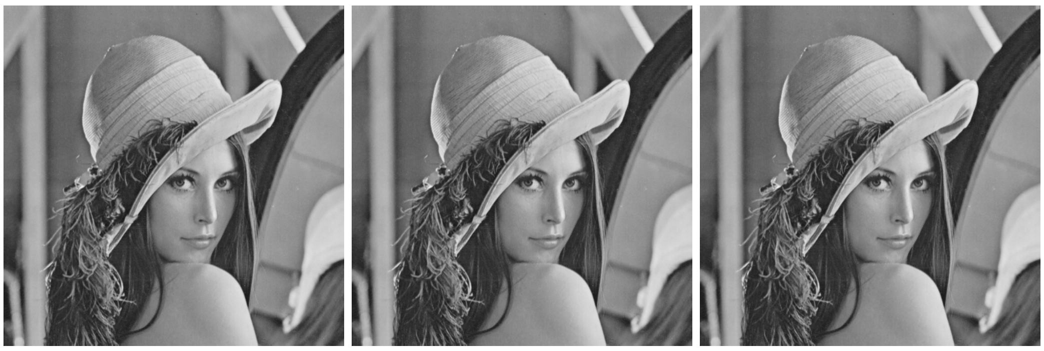


Figure 1.4: Reference HR Images for LR Test Images

As it would be apparent by now, the test set has deformities or defects which are simply not present in the train set. In this report we constrain ourselves to challenges from an image processing perspective (losses, architectures, pipelines etc) and do not delve into the optimization side of things; although we acknowledge that the latter too is quite significant when working with such limited data.

Another thing which is apparent is the fact that this is not a simple SR problem. There is a larger domain gap between the LR and HR images. Any proposed solution must overcome this domain gap too apart from performing SR. It must do that while maintaining the structure of the defects, as well as ensuring that the background looks natural too. The latter point is necessary to ensure that the defect detection network does not unnecessarily pick up false positives.

One obvious way to try and preserve structure is to get rid of the GAN loss. However, given the textured nature of the background in the HR images, it is clear the without a GAN loss, the output image will look unnaturally smooth. Hence, any solution must preserve defects while not getting rid of the GAN loss.

Apart from all this, the amount of data available is very limited. So, we will occasionally need to come up with data augmentation tricks too, even though we do not want to go into that line too much trying to keep the solution elegant.

Chapter 2

Overview of Methods

In this chapter we provide descriptions of the methods used for the project. Techniques have been adopted from blind SR, reference-based SR, image-restoration as well as paired and unpaired domain transfer. Data augmentation methods based on previous work on the same project has also been adopted. The exact configuration details along with observations can be found in the chapter 3.

2.1 Blind SR

Deep Learning has led to a dramatic leap in Super-Resolution (SR) performance in the past few years. However, being supervised, these SR methods are restricted to specific training data, where the acquisition of the low-resolution (LR) images from their high-resolution (HR) counterparts is predetermined (e.g., bicubic downscaling), without any distracting artifacts (e.g., sensor noise, image compression, non-ideal PSF, etc). Real LR images, however, rarely obey these restrictions, resulting in poor SR results by SotA (State of the Art) methods.
Zero-Shot Super-Resolution (ZSSR) [Shocher *et al.* (2018)] exploits the power of Deep Learning, but does not rely on prior training. It exploits the internal recurrence of information inside a single image, and trains a small image-specific CNN at test time, on examples extracted solely from the input image itself. As such, it can adapt itself to different settings per image. This allows it to perform SR of real old photos, noisy images, biological data, and other images where the acquisition process is unknown or non-ideal. This was the first unsupervised CNN-based SR method.
The image-specific CNN in ZSSR combines the predictive power and low entropy of internal image-specific information, with the generalization capabilities of Deep-Learning. Given a test image I, with no external examples available to train on, an Image-Specific CNN tailored to solve the SR task for this specific image is constructed. The CNN is trained on examples extracted from the test image itself. Such examples

are obtained by downscaling the LR image I, to generate a lower-resolution version of itself, $I \downarrow s$ (where s is the desired SR scale factor). A relatively light CNN is trained to reconstruct the test image I from its lower-resolution version $I \downarrow s$. The resulting trained CNN is then applied to the test image I, now using I as the LR input to the network, in order to construct the desired HR output $I \downarrow s$. Since the trained CNN is fully convolutional, it can be applied to images of different sizes. The overall pipeline is shown in figure 2.1

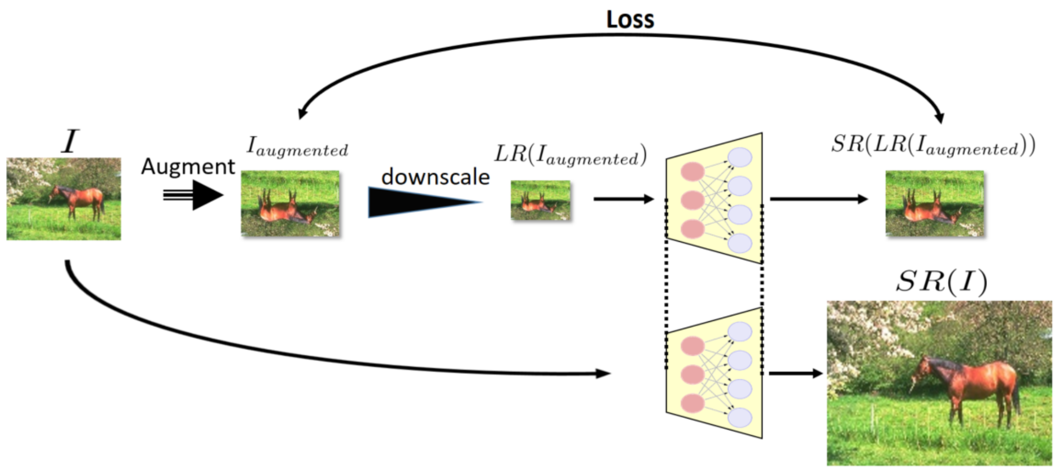


Figure 2.1: Description of the overall pipeline for ZSSR. The network is trained on the augmented dataset. After the training is done, the wanted HR image is simply obtained by passing I through the trained network

Since the "training set" consists of one instance only (the test image), data augmentation is employed on I to extract more LR-HR example-pairs to train on. The augmentation is done by downscaling the test image I to many smaller versions of itself $(I = I_0, I_1, ..., I_n)$. These play the role of the HR supervision and are called "HR fathers". Each of the HR fathers is then downscaled by the desired SR scale-factor s to obtain the "LR sons", which form the input training instances. The resulting training set consists of many image-specific LR-HR example pairs.

The network can then stochastically train over these pairs. The training set is further enriched by transforming each LR-HR pair using 4 rotations $(0°, 90°, 180°, 270°)$ and their mirror reflections in the vertical and horizontal directions. This adds 8 times more image-specific training examples.

For the sake of robustness, as well as to allow large SR scale factors s even from very small LR images, the SR is performed gradually. The algorithm is applied for several

intermediate scale-factors $(s_1, s_2, ..., s_m = s)$. At each intemediate scale s_i, the generated SR image HR_i and its downscaled/rotated versions are added to the gradually growing training-set, as new HR fathers. Those are downscaled (as well as the previous smaller 'HR examples') by the next gradual scale factor s_{i+1}, to generate the new LR-HR training example pairs. This is repeated until reaching the full desired resolution increase s.
In contrast to CNNs which train on large external datasets, the diversity of the LR-HR relations within a single image is significantly smaller, and hence, can be encoded by a much smaller and simpler image-specific network. A simple, fully convolutional network, with 8 hidden layers, each having 64 channel is the network used.

2.2 Image Translation and Restoration

[Isola *et al.* (2017)] investigate conditional adversarial networks as a general-purpose solution to image-to-image translation problems. These networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally would require very different loss formulations. This approach is effective at synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images, among other tasks. It also suggests that we no longer need to hand-engineer the mapping functions or the loss functions.
GANs [Goodfellow *et al.* (2014)] are generative models that learn a mapping from random noise vector z to output image y, $G : z \rightarrow y$. In contrast, conditional GANs [Mirza and Osindero (2014)] learn a mapping from observed image x and random noise vector z, to y, $G : \{x, z\} \rightarrow y$. The generator G is trained to produce outputs that cannot be distinguished from "real" images by an adversarially trained discriminator, D, which is trained to do as well as possible at detecting the generator's "fakes".
The objective of a conditional GAN can be expressed as

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[log(D(x, y)] + \mathbb{E}_{x,z}[log(1 - D(x, G(x, z)))] \tag{2.1}$$

where G tries to minimize this objective against an adversarial D that tries to maximize it, i.e. G^* = arg min$_G$ max$_D \mathcal{L}_{cGAN}(G, D)$.

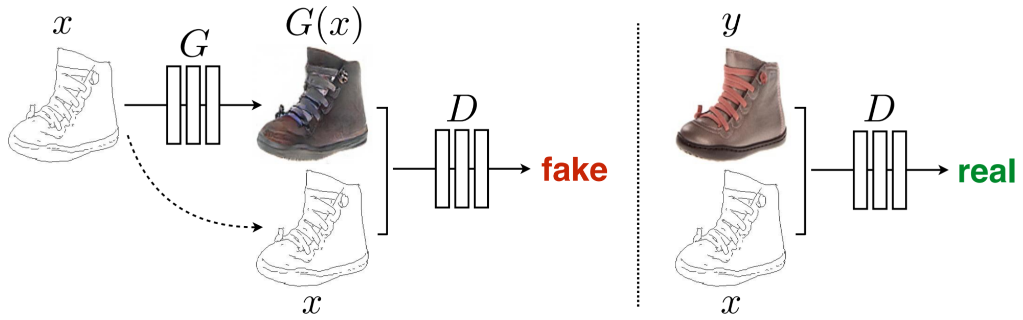


Figure 2.2: Training a conditional GAN to map edges → photo. The discriminator, D, learns to classify between fake (synthesized by the generator, G) and real {edge, photo} tuples. G learns to fool the discriminator.

Previous approaches [Pathak *et al.* (2016)] have found it beneficial to mix the GAN objective with a more traditional loss, such as L2 distance. The discriminator's job remains unchanged, but the generator is tasked to not only fool the discriminator but also to be near the ground truth output in an L2 sense. Often the L1 loss is preferred instead of the L2 loss as it causes less blurring. The L1 loss used is

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1] \tag{2.2}$$

The final objective is

$$\text{arg min}_G \ \text{max}_D \mathcal{L}_{cGAN}(G,D) + \lambda\mathcal{L}_{L1}(G) \tag{2.3}$$

Without z, the net could still learn a mapping from x to y, but would produce deterministic outputs, and therefore fail to match any distribution other than a delta function. Past conditional GANs have acknowledged this and provided Gaussian noise z as an input to the generator, in addition to x (e.g., [Wang and Gupta (2016)]). This strategy was not found effective – the generator simply learned to ignore the noise. Instead, for the final models, noise is provided only in the form of dropout, applied on several layers of our generator at both training and test time. Despite the dropout noise, only minor stochasticity is observed in the output of the networks.

A defining feature of image-to-image translation problems is that they map a high resolution input grid to a high resolution output grid. In addition, for the problems in consideration, the input and output differ in surface appearance, but both are renderings of the same underlying structure. Therefore, structure in the input is roughly aligned with structure in the output. For many image translation problems, there is a great deal of low-level information shared between the input and output, and it would be desirable to shuttle this information directly across the net. To give the generator a means to circumvent the bottleneck for information like this, skip connections are added, following the general shape of a U-Net [Ronneberger *et al.* (2015)]. Specifically, skip connections are added between each layer i and layer $n - i$, where n is the total number of layers. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$.

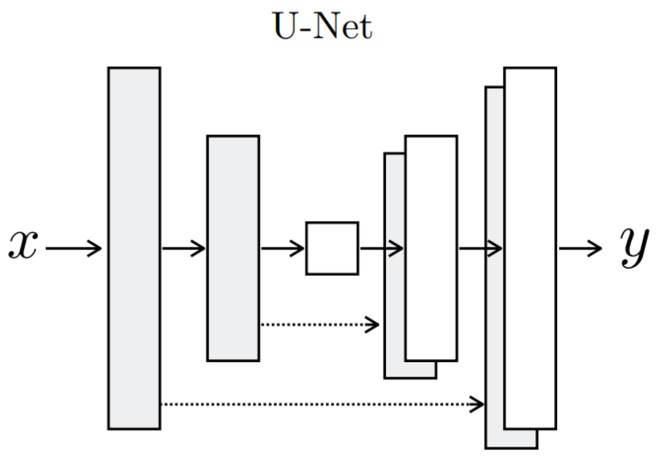


Figure 2.3: U-Net network with skip connections

Although losses like the L1 or L2 fail to encourage high-frequency crispness, in many cases they nonetheless accurately capture the low frequencies. This motivates restricting the GAN discriminator to only model high-frequency structure, relying on an L1 term to force low-frequency correctness. In order to model high-frequencies, it is sufficient to restrict attention to the structure in local image patches. Therefore, a discriminator architecture is designed – that only penalizes structure at the scale of patches (PatchGAN). This discriminator tries to classify if each $N \times N$ patch in an image is real or fake.

It is empirically found that N can be much smaller than the full size of the image and still produce high quality results. This is advantageous because a smaller PatchGAN has fewer parameters, runs faster, and can be applied to arbitrarily large images. Such a discriminator effectively models the image as a Markov random field, assuming independence between pixels separated by more than a patch diameter. The PatchGAN can hence be understood as a form of texture/style loss.

Recently, cross domain transfer has been applied for unsupervised image restoration tasks. However, directly applying existing frameworks would lead to domain-shift problems in translated images due to lack of effective supervision. [Du *et al.* (2020)] propose an unsupervised learning method that explicitly learns invariant presentation from noisy data and reconstructs clear observations. To do so discrete disentangling representation and adversarial domain adaption are introduced into general domain transfer framework, aided by extra self-supervised modules including background and semantic consistency constraints, learning robust representation under dual domain constraints, such as feature and image domains.

The goal is to learn abstract intermediate representations from noise inputs and reconstruct clear observations. In a certain way, unsupervised IR could be viewed as a specific domain transfer problem, i.e., from noise domain to clean domain. Therefore, the method is injected into the general domain transfer architecture. In supervised domain transfer, given samples (x, y) are drawn from a joint distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(x, y)$, where $\mathcal{X}$ and $\mathcal{Y}$ are two image domains. For unsupervised domain translation, samples (x, y) are drawn from the marginal distributions $\mathcal{P}_{\mathcal{X}}(x)$ and $\mathcal{P}_{\mathcal{Y}}(y)$. In order to infer the joint distribution from the marginal samples, a shared-latent space assumption is proposed that there exists a shared latent code z in a shared-latent space $\mathcal{Z}$, so that both images can be recovered from this code. Given samples (x, y) from the joint distribution, this process is represented as

$$z = E_{\mathcal{X}}(x) = E_{\mathcal{Y}}(y) \tag{2.4}$$

$$x = G_{\mathcal{X}}(z), y = G_{\mathcal{Y}}(z) \tag{2.5}$$

A key step is how to implement this shared-latent space assumption. To do so, an effective strategy is sharing high level representation by shared-weight encoder, which samples the features from the unified distribution. However, it is unsuitable for IR that latent representation only contains semantic meanings, which leads to domain shift in recovered images, e.g., blurred details and inconsistent backgrounds. Hence, the attempt is to learn more generalized representations containing richer texture and semantic features from inputs, i.e., invariant representations. To achieve it, adversarial domain adaption based discrete representation learning and self-supervised constraint modules are introduced into the method.

Discrete representation aims to compute the latent code z from inputs, where z contains texture and semantic information as much as possible. To do so, two auto-encoders are used to model $\{E_{\mathcal{X}}, G_{\mathcal{X}}\}$ and $\{E_{\mathcal{Y}}, G_{\mathcal{Y}}\}$ separately. Given any unpaired samples (x, y), where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ separately denote noise and clean sample from different domains, equation 2.4 is reformulated as $z_{\mathcal{X}} = E_{\mathcal{X}}(x)$ and $z_{\mathcal{Y}} = E_{\mathcal{Y}}(y)$. Further, IR could be represented as $F^{\mathcal{X}\to\mathcal{Y}}(x) = G_{\mathcal{Y}}(z_{\mathcal{X}})$. However, considering noise always adheres to high-frequency signals, directly reconstructing clean images is difficult due to varying noise levels and types, which requires powerful domain generator and discriminator. Therefore, disentangling representations are introduced into the architecture.

Disentangling Representation

For noise sample x, an extra noise encoder $E_{\mathcal{X}}^N$ is used to model varying noise levels and types. The self-reconstruction is formulated by $x = G_{\mathcal{X}}(z_{\mathcal{X}}, z_{\mathcal{X}}^N)$, where $z_{\mathcal{X}} = E_{\mathcal{X}}(x)$ and $z_{\mathcal{X}}^N = E_{\mathcal{X}}^N(x)$ Assuming the latent codes $z_{\mathcal{X}}$ and $z_{\mathcal{Y}}$ obey same distribution in shared-space that $\{z_{\mathcal{X}}, z_{\mathcal{Y}}\} \in \mathcal{Z}$, similar to image translation, unsupervised image restoration could be divided into two stages: forward translation and back reconstruction.

Forward Cross Translation

First the representations $\{z_{\mathcal{X}}, z_{\mathcal{Y}}\}$ are extracted from (x, y) along with the extra noise code $z_{\mathcal{X}}^N$. Restoration and degradation could be then represented as

$$\tilde{x}^{\mathcal{X}\to\mathcal{Y}} = G_{\mathcal{Y}}(z_{\mathcal{X}}) \tag{2.6}$$

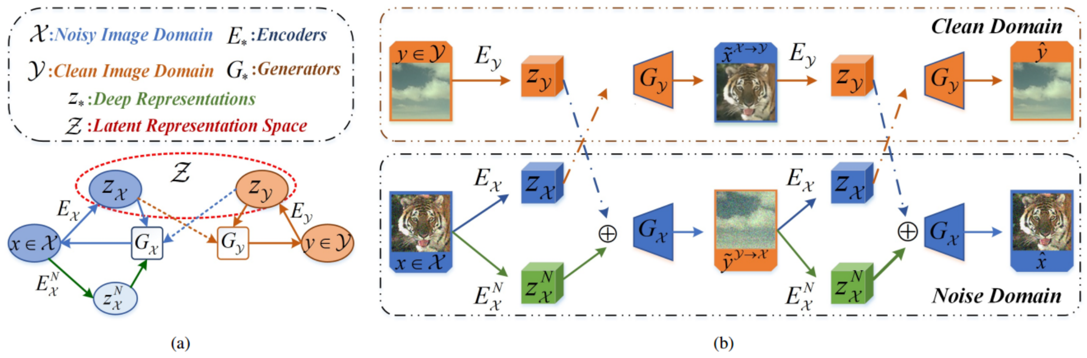
$$\tilde{y}^{\mathcal{Y}\to\mathcal{X}} = G_{\mathcal{X}}(z_{\mathcal{Y}} \oplus z_{\mathcal{X}}^N) \tag{2.7}$$

where $\tilde{x}^{\mathcal{X}\to\mathcal{Y}}$ represents the recovered clean sample, $\tilde{y}^{\mathcal{Y}\to\mathcal{X}}$ represents the degraded noise sample, $\oplus$ represents channel-wise concatenation operation. $G_{\mathcal{X}}$ and $G_{\mathcal{Y}}$ are viewed as specific domain generators.

Backward Cross Reconstruction

After performing the first translation, reconstruction could be achieved by swapping the inputs $\tilde{x}^{\mathcal{X}\to\mathcal{Y}}$ and $\tilde{y}^{\mathcal{Y}\to\mathcal{X}}$ such that

$$\hat{x} = G_{\mathcal{X}}(E_{\mathcal{Y}}(\tilde{x}^{\mathcal{X}\to\mathcal{Y}}) \oplus E_{\mathcal{X}}^N(\tilde{y}^{\mathcal{Y}\to\mathcal{X}})) \tag{2.8}$$



$\mathcal{X}$:Noisy Image Domain
E_*:Encoders
$\mathcal{Y}$:Clean Image Domain
G_*:Generators
z_*:Deep Representations
$\mathcal{Z}$:Latent Representation Space
(a)
(b)
Clean Domain
Noise Domain

Figure 2.4: The method aims to learn invariant representations from inputs and align them via adversarial domain adaption. The method is injected into general domain-transfer framework

$$\hat{y} = G_{\mathcal{Y}}(E_{\mathcal{X}}(\tilde{y}^{\mathcal{Y}\to\mathcal{X}})) \tag{2.9}$$

where $\hat{x}$ and $\hat{y}$ represent reconstructed inputs. To enforce this constraint, the cross-cycle consistency loss $\mathcal{L}^{CC}$ is added for $\mathcal{X}$ and $\mathcal{Y}$ domains. The overall architecture can be seen in figure 2.4

$$\mathcal{L}^{CC}_{\mathcal{X}}(G_{\mathcal{X}}, G_{\mathcal{Y}}, E_{\mathcal{X}}, E_{\mathcal{Y}}, E^N_{\mathcal{X}}) = \mathbb{E}_{\mathcal{X}}[\|G_{\mathcal{X}}(E_{\mathcal{Y}}(\tilde{x}^{\mathcal{X}\to\mathcal{Y}}) \oplus E^N_{\mathcal{X}}(\tilde{y}^{\mathcal{Y}\to\mathcal{X}})) - x\|_1] \tag{2.10}$$

$$\mathcal{L}^{CC}_{\mathcal{Y}}(G_{\mathcal{X}}, G_{\mathcal{Y}}, E_{\mathcal{X}}, E_{\mathcal{Y}}, E^N_{\mathcal{X}}) = \mathbb{E}_{\mathcal{Y}}[\|G_{\mathcal{Y}}(E_{\mathcal{X}}(\tilde{y}^{\mathcal{Y}\to\mathcal{X}})) - y\|_1] \tag{2.11}$$

Adversarial Domain Adaption

Another factor is how to embed latent representations $z_{\mathcal{X}}$ and $z_{\mathcal{Y}}$ into shared space. Inspired by unsupervised domain adaption, it is implemented by adversarial learning instead of shared-weight encoder. The goal is to facilitate representations from inputs obeying the similar distribution while preserving richer texture and semantic information of inputs. Therefore, a representation discriminator $D_{\mathcal{R}}$ is utilized in the architecture. This feature adversarial loss $\mathcal{L}^{\mathcal{R}}{}_{adv}$ is expressed as

$$\begin{aligned} L^{R}{}_{adv}(E_{\mathcal{X}}, E_{\mathcal{Y}}, D_{\mathcal{R}}) = \mathbb{E}_{\mathcal{X}}\left[\frac{1}{2}\log D_{\mathcal{R}}(z_{\mathcal{X}}) + \frac{1}{2}\log(1 - D_{\mathcal{R}}(z_{\mathcal{X}}))\right] \\ + \mathbb{E}_{\mathcal{Y}}\left[\frac{1}{2}\log D_{\mathcal{R}}(z_{\mathcal{Y}}) + \frac{1}{2}\log(1 - D_{\mathcal{R}}(z_{\mathcal{Y}}))\right] \end{aligned} \tag{2.12}$$

Due to lack of effective supervised signals for translated images, only relying on feature domain discriminant constraints would lead to domain shift problems inevitably in generated images. To speed convergence while learning more robust representations, self-supervised modules including *Background Consistency Module (BCM)* and *Semantic Consistency Module (SCM)* are introduced to provide more reasonable and reliable supervision.

BCM aims to preserve the background consistency between the translated images and inputs. A multi-scale Gaussian-Blur operator is used to obtain multi-scale features respectively. Therefore, a background consistency loss $\mathcal{L}_{BC}$ could be formulated as

$$\mathcal{L}_{BC} = \sum_{\sigma=5,9,15} \lambda_\sigma \left\| B_\sigma(\chi) - B_\sigma(\tilde{\chi}) \right\|_1 \tag{2.13}$$

Where B_σ denotes the Gaussian-Blur operator with blur kernel σ, λ_σ is the hyper-parameter to balance the errors at different Gaussian-Blur levels. χ and $\tilde{\chi}$ denote original input and the translated output, i.e., $\{x, \tilde{x}^{\mathcal{X}\to\mathcal{Y}}\}$ and $\{y, \tilde{y}^{\mathcal{Y}\to\mathcal{X}}\}$.

In addition, the features from the deeper layers of the pre-trained VGG-19 [Simonyan and Zisserman (2014)] model contain semantic meanings only, which are noiseless or with little noise. Therefore, different from the general feature loss, which aims to recover finer image texture details via similarities among shallow features, only deeper features are extracted as semantic representations from the corrupted and recovered images to keep consistency, referred to as semantic consistency loss $\mathcal{L}_{SC}$. It is formulated as

$$\mathcal{L}_{SC} = \left\| \phi_l(\chi) - \phi_l(\tilde{\chi}) \right\|_2^2 \tag{2.14}$$

where $\phi_l(.)$ represents features from the l^{th} layer of the VGG-19 model.

A few other losses are used along with the aforementioned losses.

Target Domain Adversarial Loss

Domain adversarial loss $\mathcal{L}_{adv}^{domain}$ is imposed where $D_\mathcal{X}$ and $D_\mathcal{Y}$ attempt to discriminate the realness of generated images from each domain. For the noise domain, the adversarial loss $\mathcal{L}^{\mathcal{X}}{}_{adv}$ is defined as

$$\begin{aligned} \mathcal{L}_{adv}^{\mathcal{X}} &= \mathbb{E}_{x\sim P_\mathcal{X}(x)}\left[\log D_\mathcal{X}(x)\right] + \\ &\mathbb{E}_{\substack{\mathcal{X}\\ y\sim P_\mathcal{Y}(y)\\ x\sim P_\mathcal{X}(x)}}\left[\log\left(1 - D_\mathcal{X}\left(G_\mathcal{X}\left(E_\mathcal{Y}(y), E_\mathcal{X}^N(x)\right)\right)\right)\right] \end{aligned} \tag{2.15}$$

Similarly for the clean image domain, the adversarial loss is defined as

$$\begin{aligned}\mathcal{L}_{adv}^{\mathcal{Y}} =&\mathbb{E}_{y\sim P_{\mathcal{Y}}(y)}\left[\log D_{\mathcal{Y}}(y)\right] + \\ &\mathbb{E}_{x\sim P_{\mathcal{X}}(x)}\left[\log\left(1 - D_{\mathcal{Y}}\left(G_{\mathcal{Y}}\left(E_{\mathcal{X}}(x)\right)\right)\right)\right]\end{aligned} \tag{2.16}$$

Self-Reconstruction Loss

In addition to the cross-cycle reconstruction, a self-reconstruction loss $\mathcal{L}^{Rec}$ is also applied to facilitate the training. This process is represented as $\hat{x} = G_{\mathcal{X}}\left(E_{\mathcal{X}}(x) \oplus E_{\mathcal{X}}^{N}(x)\right)$ and $\hat{y} = G_{\mathcal{Y}}(E_{\mathcal{Y}}(y))$.

KL Divergence Loss

In order to model the noise encoder branch, a KL divergence loss is added to regularize the distribution of the noise code $z_{\mathcal{X}}^{N} = E_{\mathcal{X}}^{N}(x)$ to be close to the normal distribution i.e., $p(z_{\mathcal{X}}^{N}) \sim \mathcal{N}(0,1)$, where $D_{KL} = -\int p(z)\log\left(\frac{p(z)}{q(z)}\right)dz$. The full objective function is summarized as

$$\begin{aligned}\min_{E_{\mathcal{X}},E_{\mathcal{X}}^{N},E_{\mathcal{Y}},G_{\mathcal{X}},G_{\mathcal{Y}}}\ \max_{D_{\mathcal{X}},D_{\mathcal{Y}},D_{\mathcal{R}}} &= \lambda_{\mathcal{R}}\mathcal{L}_{adv}^{\mathcal{R}} + \\ \lambda_{adv}\mathcal{L}_{adv}^{\text{domain}} + \lambda_{CC}\mathcal{L}^{CC} + \lambda_{rec}&\mathcal{L}^{Rec} + \\ \lambda_{bc}\mathcal{L}^{BC} + \lambda_{sc}\mathcal{L}^{SC} + \lambda_{KL}&\mathcal{L}^{KL}\end{aligned} \tag{2.17}$$

The network follows the architecture as the one used in [Liu *et al.* (2017)], the difference is that an extra noise encoder branch is introduced and the shared weight encoder is removed. Representation discriminator is a full convolutional network structure, which stacks four convolutional layers with two strides and a global average pooling layer.

The domain-specific nature of losses like pixel-level cycle-consistency or feature-level matching losses hinders translation across large domain gaps. [Zheng *et al.* (2021)] propose a novel spatially-correlative loss that is simple, efficient and yet effective for preserving scene structure consistency while supporting large appearance changes during unpaired image-to-image (I2I) translation. This loss exploits the spatial patterns of self-similarity as a means of defining scene structure. The spatially-correlative loss is geared towards only capturing spatial relationships within an image rather than domain appearance. A new self-supervised learning method to explicitly learn spatially-correlative maps for each specific translation task is also introduced. An illustration of this is shown in figure 2.5.

Given a collection of images $\mathcal{X} \in \mathbb{R}^{H\times W\times C}$ from a particular domain, the main goal

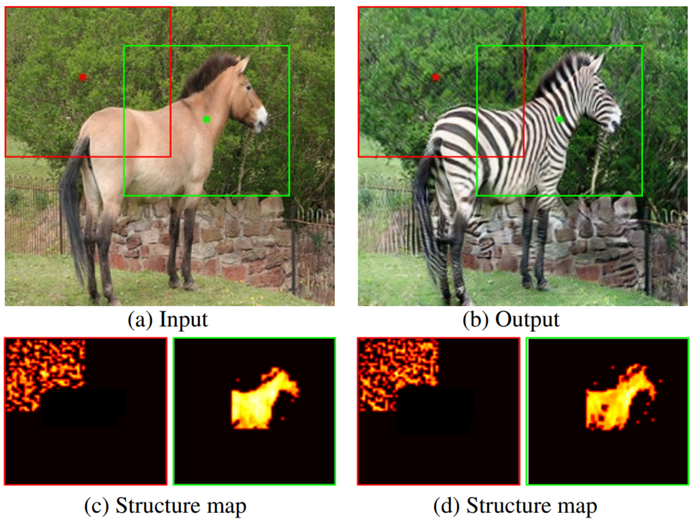


Figure 2.5: An illustration of the spatial correlation loss. Despite vast appearance differences between the horse and zebra, when the scene structures are identical, the spatial patterns of self-similarities are as well

is to learn a model Φ that receives the image $x \in \mathcal{X}$ as input and transfers it into the target domain $\mathcal{Y} \in \mathbb{R}^{H\times W\times C}$, in a manner that retains the original scene structure but converts the appearance appropriately. The focus is on designing a loss function that measures the structural similarity between the input image x and the translated image $\hat{y} = \Phi(x)$. However, unlike most existing approaches that directly attempt to evaluate the structural similarity between input and translated images at some deep feature level, the idea is to instead compute the self-similarity of deep features within each image, and then compare the self-similarity patterns between the images.

Two losses are investigated: *fixed self-similarity (FSeSim)* and *learned self-similarity (LSeSim)*. In the first instance, the self-similarity patterns of features extracted from a fixed pretrained network (e.g. VGG16 [Simonyan and Zisserman (2014)]) are directly compared. In the second instance, an additional structure representation model that learns to correctly compare the self-similarity patterns is introduced. The contrastive infoNCE loss [Van den Oord *et al.* (2018)] is used to learn such a network without label supervision.

Fixed Self-Similarity (FSeSim)

Given an image x in one domain and its corresponding translated image $\hat{y}$ in another, the features f_x and $f_{\hat{y}}$ are extracted using a simple network (e.g. VGG16). Instead of directly computing the feature distance $||f_x - f_{\hat{y}}||_p$, the self-similarity is computed in the form of a map. This is called the *spatially-correlative map*, formally:

$$S_{x_i} = (f_{x_i})^T (f_{x*}) \tag{2.18}$$

where $f_{x_i^T} \in \mathbb{R}^{1\times C}$ is a feature of a query point x_i with C channels, $f_{x*} \in \mathbb{R}^{C\times N_p}$

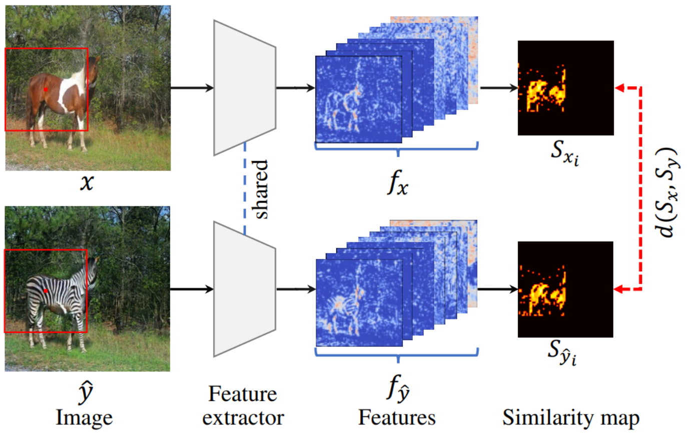


Figure 2.6: Image x and corresponding translated image $\hat{y}$ are first fed into the feature extractor. Then the local self-similarity for each query point is computed

contains corresponding features in a patch of N_p points, and $S_{x_i} \in \mathbb{R}^{1\times N_p}$ captures the feature spatial correlation between the query point and other points in the patch. An example is shown in figure 2.6, where the spatially-correlative map for the query patch is visualized as a heat map. Unlike the original features that would still encode domain-specific attributes such as color, lighting and texture, the self-similarity map only captures the spatially-correlative relationships.

Next, the structure of the whole image as is represented a collection of multiple spatially-correlative maps $S_x = [S_{x_1}; S_{x_2}; ...; S_{x_{N_s}}] \in \mathbb{R}^{N_s\times N_p}$, where N_s is the number of sampled patches. This is a semi-sparse representation, but is more computationally efficient. Then the multiple structure similarity maps between the input x and the translated image $\hat{y}$ are compared as

$$\mathcal{L}_s = d(S_x, S_{\hat{y}}) \tag{2.19}$$

where $S_{\hat{y}}$ is the collection of corresponding spatially-correlative maps in the target domain. Two forms for $d(\cdot)$ are considered: the L1 distance $\|S_x - S_{\hat{y}}\|$ and the cosine distance $\|1 - cos(S_x, S_{\hat{y}})\|$. The former term strongly encourages the spatial similarity to be consistent at all points in a patch, while the latter term supports pattern correlation without concern for differences in magnitude.

Learned Self-Similarity (LSeSim)

Although the FSeSim provides strong supervision for structure consistency, it does not explicitly learn a structure representation for a specific translation task. As opposed to existing feature-level losses [Johnson *et al.* (2016), Mechrez *et al.* (2018)] that only utilize the features from a fixed pre-trained network, the idea is to additionally learn a structure representation network for each task that expresses the learned self-similarity, or LSeSim.

In order to learn such a model without supervision, the self-supervised contrastive learning that associates similar features, while simultaneously dissociates different features is considered. Following [Park *et al.* (2020)], the contrastive loss is built at the patch level. Except, here the pairs for comparison are the spatially-correlative maps, rather than the original features.

To help generate pairs of similar patch features for self-supervised learning, aug-

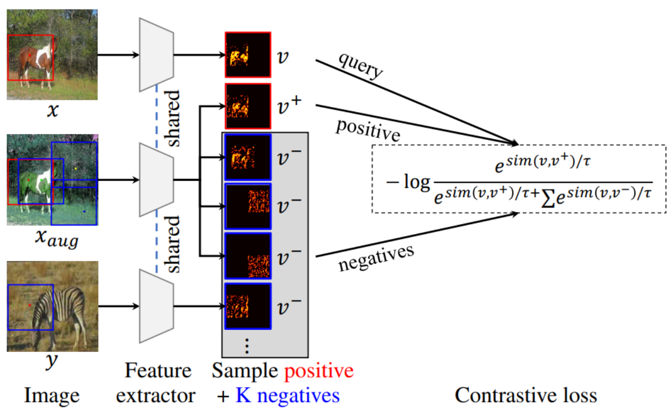


Figure 2.7: Three images are fed into the feature extractor, in which two images, x and x_{aug}, are homologous with the same structure but varied appearances, and y is another randomly sampled image. For each query patch in x, the "positive" sample is the corresponding patch in x_{aug}, and all other patches are considered as "negative" samples

mented images are created by applying structure-preserving transformations. Formally, let $\boldsymbol{v} = S_{x_i} \in \mathbb{R}^{1\times N_p}$ denotes the spatially-correlative map of the "query" patch. Let $\boldsymbol{v}^+ = S_{\hat{x}_i} \in \mathbb{R}^{1\times N_p}$ and $\boldsymbol{v}^- \in \mathbb{R}^{K\times N_p}$ be "positive" and "negative" patch samples, respectively. The query patch is positively paired with a patch in the same position i within an augmented image x_{aug} and and negatively paired to patches sampled from other positions in x_{aug} or patches from other images y. The number of negative patches used is $K = 255$. This is illustrated in figure 2.7.

The contrastive loss is given by

$$\mathcal{L}_c = -\log \frac{e^{\operatorname{sim}\left(\boldsymbol{v}, \boldsymbol{v}^+\right)/\tau}}{e^{\operatorname{sim}(\boldsymbol{v}, \boldsymbol{v}^+)/\tau} + \sum_{k=1}^{K} e^{\operatorname{sim}\left(\boldsymbol{v}, \boldsymbol{v}_k^-\right)/\tau}} \tag{2.20}$$

where $\operatorname{sim}(\boldsymbol{v}, \boldsymbol{v}^+) = \boldsymbol{v}^T \boldsymbol{v}^+ / \|\boldsymbol{v}\| \|\boldsymbol{v}^+\|$ is the is the cosine similarity between two spatially-correlative maps and τ is a temperature parameter. To minimize this loss, the network encourages the corresponding patches with the same structure to be close even if they have very different visual appearances, which fits in with the goal of image translation. This contrastive loss is only used for optimizing the structure representation network. The spatially-correlative loss for the generator is always the loss in equation 2.19.
Overall the networks are trained by jointly minimising the following losses.

$$\begin{aligned}
\mathcal{L}_D &= -\mathbb{E}_{y \sim p_d}[\log D(y)] - \mathbb{E}_{\hat{y} \sim p_g}[\log(1 - D(\hat{y}))] \\
\mathcal{L}_S &= \mathcal{L}_c \\
\mathcal{L}_G &= \mathbb{E}_{\hat{y} \sim p_g}[\log(1 - D(\hat{y}))] + \lambda d\left(S_x, S_{\hat{y}}\right)
\end{aligned} \tag{2.21}$$

where $\mathcal{L}_D$ is the adversarial loss for the discriminator D, $\hat{y}$ is the translated image, and $\mathcal{L}_S$ is the contrastive loss for the structure representation network f. $\mathcal{L}_G$ is the loss for the generation (translation) network G, which consists of the style loss term and the structure loss term. λ is a hyperparameter to trade off between style and content.
The CycleGAN [Zhu *et al.* (2017)] is used as the reference architecture, but only half of that pipeline is used and the cycle-consistency loss is replaced with the F/LSeSim loss. Specifically, the ResNet based generator is used along with the PatchGAN discriminator [Isola *et al.* (2017)].

2.3 Refence Based SR

Reference-based image super-resolution (RefSR) has shown promising success in recovering high-frequency details by utilizing an external reference image (Ref). In this task, texture details are transferred from the Ref image to the low-resolution (LR) image according to their point- or patch-wise correspondence. Therefore, high-quality correspondence matching is critical. It is also desired to be computationally efficient. Besides, existing RefSR methods tend to ignore the potential large disparity in distributions between the LR and Ref images, which hurts the effectiveness of the information utilization. In MASA-SR [Lu *et al.* (2021*b*)] two novel modules are designed to address these problems. The proposed Match and Extraction Module significantly reduces the computational cost by a coarse-to-fine correspondence matching scheme. The Spatial Adaptation Module learns the difference of distribution between the LR and Ref images, and remaps the distribution of Ref features to that of LR features in a spatially adaptive way. This makes the network robust to handle different reference images.

The framework mainly consists of three parts: the encoder, *Matching and Extraction Modules (MEM)*, and fusion modules that contain *Spatial Adaptation Modules (SAM)* and *Dual Residual Aggregation Modules (DRAM)*. A representation can be seen in figure 2.8. LR, Ref↓ and Ref denote the low resolution image, the $\times 4$ bicubic-downsampled reference image and the reference image, respectively. The encoder is trained along with other parts of the network from scratch.

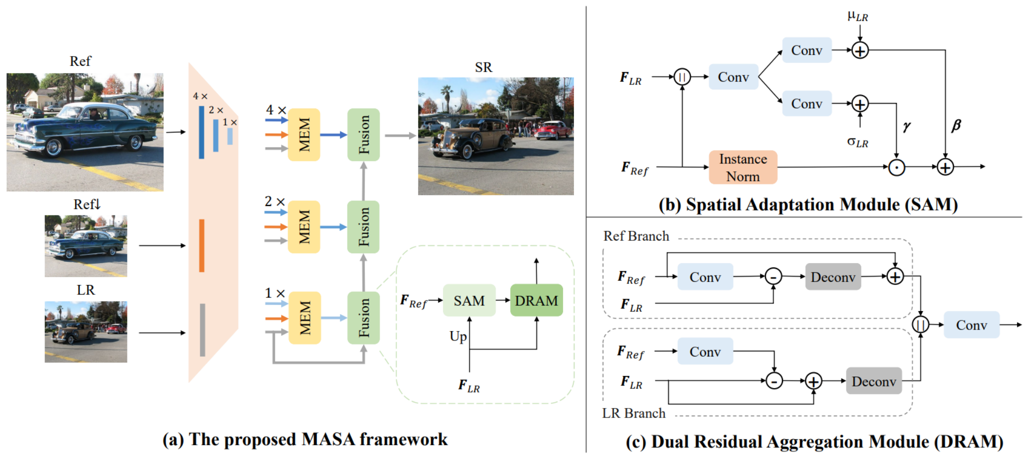


Figure 2.8: (a) Framework of MASA-SR. It consists of an encoder, several MEM, SAM, and DRAM. (b) Structure of the SAM. (c) Structure of the DRAM

The encoder consists of three building blocks – the second and third blocks halve the size of the feature maps with stride 2. After passing the Ref image into the encoder, three Ref features with different scales are obtained as $\boldsymbol{F}^{s}_{Ref}$ where $s = 1, 2, 4$. The LR image and the Ref↓ image only go through the first block of the encoder, producing $\boldsymbol{F}_{LR}$ and $\boldsymbol{F}_{Ref\downarrow}$. Afterwards, $\{\boldsymbol{F}_{LR}, \boldsymbol{F}_{Ref\downarrow}, \boldsymbol{F}^{s}_{Ref}\}$ are fed into MEM to perform coarse-to-fine correspondence matching and feature extraction as shown as in figure 2.9. Though there are three MEMs in figure 2.8(a), the matching steps are only performed once between $\boldsymbol{F}_{LR}$ and $\boldsymbol{F}_{Ref\downarrow}$. The feature extraction stage is performed three times, each for one Ref feature $\boldsymbol{F}^{s}_{Ref}$ of scale s. To generate the final SR output, the LR features and output

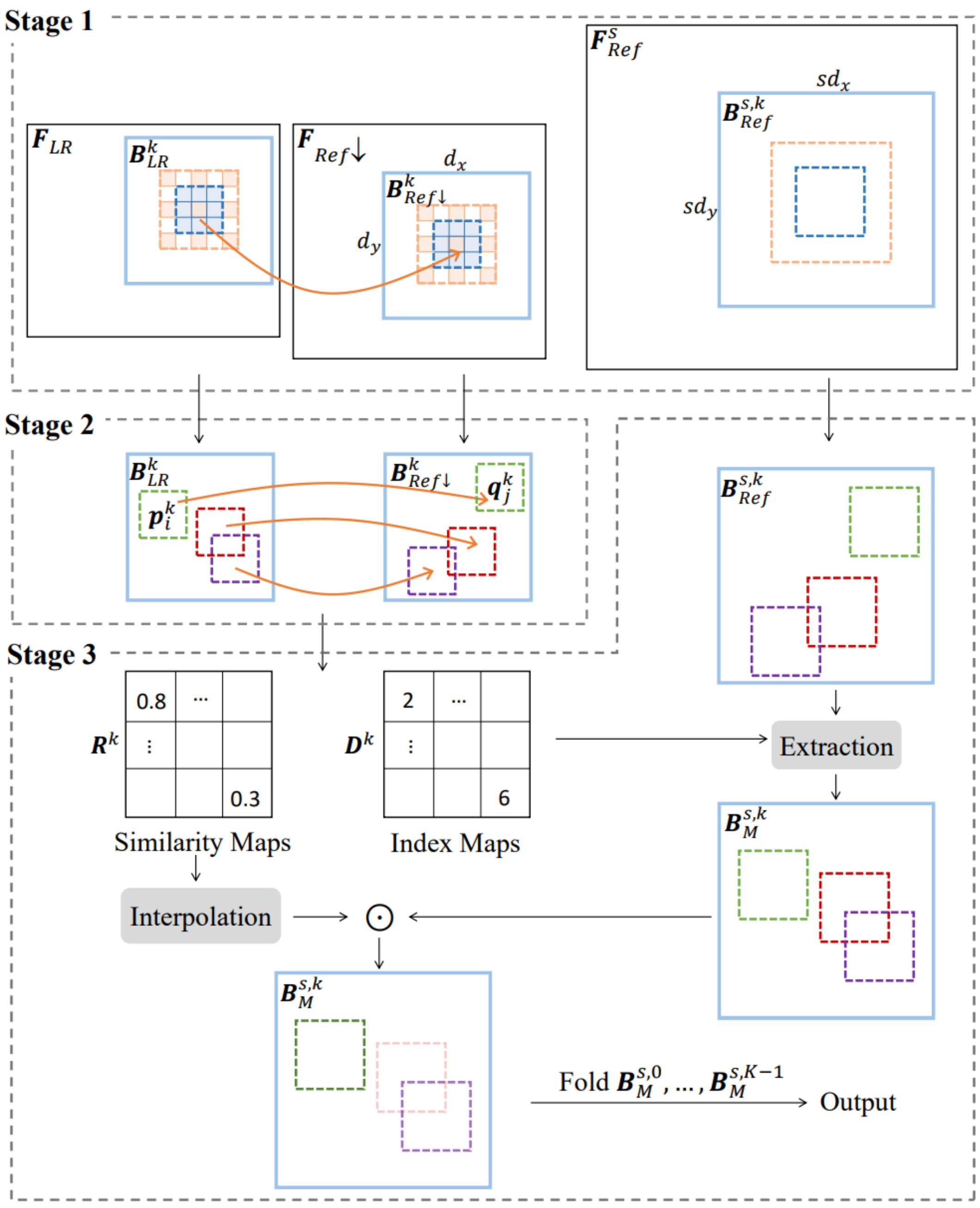


Figure 2.9: Pipeline of MEM. In **Stage 1**, each LR block finds the most relevant Ref↓ block. In **Stage 2**, dense patch matching is performed in each (LR block, Ref↓ block) pair. In **Stage 3**, Ref features are extracted according to the similarity and index maps produced in the second stage. All the blocks are denoted by solid squares in light blue and patches are denoted by dotted squares in different colors

features from MEM are fused through the fusion module, where the proposed SAM is used to align the statistics of Ref features to those of LR ones. The DRAM is used to enhance high-frequency details. The MEM, SAM, and DRAM details are mentioned in the following text.

Matching and Extraction Module (MEM)

It is known that in a local region of a natural image, neighboring pixels are likely to come from common objects and share similar color statistics. Previous research on natural image priors also indicates that neighboring patches in one image are likely to find their correspondence spatially coherent with each other. This motivates to use a coarse-to-fine matching scheme, i.e., coarse block matching and fine patch matching. Note that 'block' and 'patch' are two different concepts in the current context. The size of block is larger than patch (3×3 in experiments). As shown in figure 2.9, fine correspondences are found in the feature space only for blocks. Specifically, the LR feature is unfolded into non-overlapping blocks. Each LR block will find its most relevant Ref↓ block. By doing so, the computational cost of matching is reduced significantly compared with previous methods [Zhang *et al.* (2019), Yang *et al.* (2020)]. To achieve enough precision, further dense patch matching is performed within each (LR block, Ref↓ block) pair. In the last stage, useful Ref features are extracted according to the obtained correspondence information.

Stage 1: Coarse matching

In this stage, the LR feature $\boldsymbol{F}_{LR}$ is unfolded into K non-overlapping blocks: $\{\boldsymbol{B}_{LR}^0, ..., \boldsymbol{B}_{LR}^{K-1}\}$. For each LR block $\boldsymbol{B}_{LR}^k$,its most relevant Ref↓ block $\boldsymbol{B}_{Ref\downarrow}^k$ is found. The center patch of $\boldsymbol{B}_{LR}^k$ is taken to to compute the cosine similarity with each patch of $\boldsymbol{F}_{Ref\downarrow}$ as

$$r_{c,j}^k = \left\langle \frac{\boldsymbol{p}_c^k}{\|\boldsymbol{p}_c^k\|}, \frac{\boldsymbol{q}_j}{\|\boldsymbol{q}_j\|} \right\rangle \tag{2.22}$$

where $\boldsymbol{p}_c^k$ is the center patch of $\boldsymbol{B}_{LR}^k$, $\boldsymbol{q}_j$ is the $j-$th patch of $\boldsymbol{F}_{Ref\downarrow}$, and $r_{c,j}^k$ is their similarity score. According to the similarity scores the most similar patch for $\boldsymbol{p}_c^k$ in $\boldsymbol{F}_{Ref\downarrow}$ can be found. A block of size $d_x \times d_y$ is then cropped centered around this similar patch, denoted as $\boldsymbol{B}_{Ref\downarrow}^k$. According to the local coherence property, for all patches in $\boldsymbol{B}_{LR}^k$ their most similar patches are likely to reside in this $\boldsymbol{B}_{Ref\downarrow}^k$. . On the other hand, the corresponding $sd_x \times sd_y$ block from $\boldsymbol{F}_{Ref}^s$, denoted by $\boldsymbol{B}_{Ref}^{s,k}$ is cropped to be used in the feature extraction stage.

Note that the center patch may not be representative enough to cover full content of the

LR block if the size of the LR block is much larger than that of its center patch. This may lead to finding the irrelevant Ref↓ block. To address it, center patches with different dilation rates are used to compute the similarity. The details are shown in Stage 1 of figure 2.9, where the dotted blue patch denotes the case of $dilation = 1$ and the dotted orange patch denotes the case of $dilation = 2$. Then the similarity score is computed as the sum of results of different dilations. After this stage, for each LR block, its most relevant Ref↓ block and the corresponding Ref block are obtained, forming triples of $(\boldsymbol{B}^k_{LR}, \boldsymbol{B}^k_{Ref\downarrow}, \boldsymbol{B}^{s,k}_{Ref})$. The search space of $\boldsymbol{B}^k_{LR}$ is limited to $\boldsymbol{B}^k_{Ref\downarrow}$ in the fine matching stage.

Stage 2: Fine matching

In this stage, dense patch matching is performed between each LR block and its corresponding Ref↓ block independently. A set of index maps $\{\boldsymbol{D}^0, ..., \boldsymbol{D}^{K-1}\}$ and similarity maps $\{\boldsymbol{R}^0, ..., \boldsymbol{R}^{K-1}\}$ are obtained. More precisely, taking the $k-$th pair $(\boldsymbol{B}^k_{LR}, \boldsymbol{B}^k_{Ref\downarrow})$ for example, the similarity score is computed between each patch of $\boldsymbol{B}^k_{LR}$ and each patch of $\boldsymbol{B}^k_{Ref\downarrow}$ as

$$r^k_{i,j} = \left\langle \frac{\boldsymbol{p}^k_i}{\|\boldsymbol{p}^k_i\|}, \frac{\boldsymbol{q}^k_j}{\|\boldsymbol{q}^k_j\|} \right\rangle \tag{2.23}$$

where $\boldsymbol{p}^k_i$ is the $i-$the patch of $\boldsymbol{B}^k_{LR}$, $\boldsymbol{q}^k_j$ is the $j-$th patch of $\boldsymbol{B}^k_{Ref\downarrow}$, and $r^k_{i,j}$ is their similarity score. Then the $i-$th element of $\boldsymbol{D}^k$ is calculated as

$$\boldsymbol{D}^k_i = \arg\max_j r^k_{i,j} \tag{2.24}$$

The $i-$th element of $\boldsymbol{R}^k$ is the highest similarity score related to the $i-$th patch of $\boldsymbol{B}^k_{LR}$ as

$$\boldsymbol{R}^k_i = \max_j r^k_{i,j} \tag{2.25}$$

Stage 3: Feature extraction

In this stage, patches are first extracted from $\boldsymbol{B}^{s,k}_{Ref}$ according to the index map $\boldsymbol{D}^k$, and a new feature map $\boldsymbol{B}^{s,k}_M$ is formed. Specifically, the $\boldsymbol{D}^k_i-$th patch of $\boldsymbol{B}^{s,k}_{Ref}$ is cropped as the $i-$th patch of $\boldsymbol{B}^{s,k}_M$. Moreover, since Ref features with higher similarity scores are more useful, $\boldsymbol{B}^{s,k}_M$ is multiplied with the corresponding similarity score map $\boldsymbol{R}^k$ to get the weighted feature block as

$$\boldsymbol{B}_M^{s,k} := \boldsymbol{B}_M^{s,k} \odot \left(\boldsymbol{R}^k\right)\uparrow \tag{2.26}$$

where $()\uparrow$ and $\odot$ denote bilinear interpolation and element-wise multiplication respectively.

The final output of MEM is obtained by folding $\{\boldsymbol{B}_M^{s,0}, ..., \boldsymbol{B}_M^{s,K-1}\}$ together, which is the reverse operation of the unfolding operation in Stage 1.

Spatial Adaptation Module (SAM) In many situations, the LR and the Ref images may have similar content and texture. But color and luminance distributions diverge. Thus the distribution of extracted Ref features may not be consistent with that of the LR features. Therefore, simply concatenating the Ref and LR features together and feeding them into the following convolution layers is not optimal. The Spatial Adaptation Module (SAM) is proposed to remap the distribution of the extracted Ref features to that of the LR features.

The structure of SAM is illustrated in figure 2.8(b). The LR feature and extracted Ref feature are first concatenated before feeding into convolution layers to produce two parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ which are the same size as the LR feature. Then instance normalization [Ulyanov *et al.* (2016)] is applied to the Ref feature as

$$\boldsymbol{F}_{Ref}^{c} \longleftarrow \frac{\boldsymbol{F}_{Ref}^{c} - \boldsymbol{\mu}_{Ref}^{c}}{\boldsymbol{\sigma}_{Ref}^{c}} \tag{2.27}$$

where $\boldsymbol{\mu}_{Ref}^{c}$ and $\boldsymbol{\sigma}_{Ref}^{c}$ are the mean and standard deviation of $\boldsymbol{F}_{Ref}$ in channel c as

$$\boldsymbol{\mu}_{Ref}^{c} = \frac{1}{HW}\sum_{y,x}\boldsymbol{F}_{Ref}^{c,y,x} \tag{2.28}$$

$$\boldsymbol{\sigma}_{Ref}^{c} = \sqrt{\frac{1}{HW}\sum_{y,x}\left(\boldsymbol{F}_{Ref}^{c,y,x} - \boldsymbol{\mu}_{Ref}^{c}\right)^2} \tag{2.29}$$

H and W are the height and width of $\boldsymbol{F}_{Ref}$. $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are then updated with the mean and standard deviation of the LR feature as

$$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \boldsymbol{\mu}_{LR} \tag{2.30}$$

$$\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} + \boldsymbol{\sigma}_{LR} \tag{2.31}$$

Finally, $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are multiplied and added to the normalized Ref feature in an element-wise manner as

$$\boldsymbol{F}_{Ref} \leftarrow \boldsymbol{F}_{Ref} * \boldsymbol{\gamma} + \boldsymbol{\beta} \tag{2.32}$$

Since the difference between the Ref features and LR features varies with respect to the spatial location, while the statistics $\boldsymbol{mu}_{LR}$, $\boldsymbol{\sigma}_{LR}$, $\boldsymbol{mu}_{Ref}$ and $\boldsymbol{\sigma}_{Ref}$ are of size $C \times 1 \times 1$, learnable convolutions are used to predict two spatial-wise adaptation parameter $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. The convolutions in SAM takes as the input both Ref and LR features to learn their difference. Besides, after obtaining $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ from the convolutions, they are added with the mean and standard deviation of the LR feature.

Dual Residual Aggregation Module (DRAM)

After spatial adaptation, the transferred Ref features are fused with the LR features using our the Dual Residual Aggregation Module (DRAM) as shown in figure 2.8 (c). DRAM consists of two branches, i.e., the LR branch and the Ref branch.

The Ref branch aims to refine the high-frequency details of the Ref features. It first downsamples the Ref feature $\boldsymbol{F}_{Ref}$ by a convolution layer with stride 2, and the residual $\boldsymbol{R}es_{Ref}$ between the downsampled Ref feature and the LR feature $\boldsymbol{F}_{LR}$ is then upsampled by a transposed convolution layer as

$$\begin{cases} \boldsymbol{R}_{esRef} = Conv\left(\boldsymbol{F}_{Ref}\right) - \boldsymbol{F}_{LR} \\ \boldsymbol{F}'_{Ref} = \boldsymbol{F}_{Ref} + Deconv\left(\boldsymbol{R}es_{Ref}\right) \end{cases} \tag{2.33}$$

Similarly, the high-frequency details of the LR features are refined as

$$\begin{cases} \boldsymbol{R}es_{LR} = \boldsymbol{F}_{LR} - Conv\left(\boldsymbol{F}_{Ref}\right) \\ \boldsymbol{F}'_{LR} = Deconv\left(\boldsymbol{F}_{LR} + \boldsymbol{R}es_{LR}\right) \end{cases} \tag{2.34}$$

At last, the outputs of two branches are concatenated and passed through another convolution layer with stride 1. In this way, the details in the LR and Ref features are enhanced and aggregated, leading to more representative features.

The loss functions used are described next. The reconstruction loss $\mathcal{L}_{rec}$ is defined as

$$\mathcal{L}_{rec} = \|\boldsymbol{I}_{HR} - \boldsymbol{I}_{SR}\|_1 \tag{2.35}$$

where $\boldsymbol{I}_{HR}$ and $\boldsymbol{I}_{SR}$ denote the ground truth image and the network output. The Perceptual loss $\mathcal{L}_{per}$ is expressed as

$$\mathcal{L}_{per} = \|\phi_i(\boldsymbol{I}_{HR}) - \phi_i(\boldsymbol{I}_{SR})\|_2 \tag{2.36}$$

where ϕ_i denotes the $i-$th layer of VGG-19 (in this case the layer used was $conv5\_4$. The adversarial loss $\mathcal{L}_{adv}$ is adopted from the Relativistic GANs [Jolicoeur-Martineau (2018)].

$$\begin{aligned}
\mathcal{L}_D = &- \mathbb{E}_{\boldsymbol{I}_{HR}}\left[\log\left(D\left(\boldsymbol{I}_{HR}, \boldsymbol{I}_{SR}\right)\right)\right] \\
&- \mathbb{E}_{\boldsymbol{I}_{SR}}\left[\log\left(1 - D\left(\boldsymbol{I}_{SR}, \boldsymbol{I}_{HR}\right)\right)\right] \\
\mathcal{L}_G = &- \mathbb{E}_{\boldsymbol{I}_{HR}}\left[\log\left(1 - D\left(\boldsymbol{I}_{HR}, \boldsymbol{I}_{SR}\right)\right)\right] \\
&- \mathbb{E}_{\boldsymbol{I}_{SR}}\left[\log\left(D\left(\boldsymbol{I}_{SR}, \boldsymbol{I}_{HR}\right)\right)\right]
\end{aligned} \tag{2.37}$$

The overall objective $\mathcal{L}$ is

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{per}\mathcal{L}_{per} + \lambda_{adv}\mathcal{L}_{adv} \tag{2.38}$$

The encoder consists of 3 building blocks, each composed of 1 convolutional layer and 4 ResBlocks [He *et al.* (2016)]. The fusion module consists of 1 spatial adaptation module, 1 dual residual aggregation module, several convolutional layers and ResBlocks. The numbers of ResBlocks in $1\times$, $2\times$ and $4\times$ fusion modules are 12, 8, and 4. The number of all intermediate channels is 64. The activation function is ReLU. No batch normalization (BN) layer is used.
In MEM, the LR block size is set to 8×8. The Ref↓ block size is set to $\frac{12H_{Ref\downarrow}}{H_{LR}} \times \frac{12W_{Ref\downarrow}}{W_{LR}}$ where $H_{LR}, W_{LR}, H_{Ref}\downarrow, W_{Ref}\downarrow$ are the height and width of the LR image and the Ref↓ image respectively. The patch size is set to 3×3. The discriminator structure is the same as that adopted in [Wang *et al.* (2018)].

Chapter 3

Experiments

In this chapter we describe the different experiments conducted. The results are also presented in this chapter. Apart from the configuration and results, we also present the line of thought which led to that experiment in case that might come in handy for future works.

3.1 ZSSR

A recurring issue with the given SR problem is the distortion of defects. This is a consequence of the GAN loss as well as overfitting on the training data. Hence, we first start experimenting with Blind SR methods to see what we can get. The idea is that since such methods rely only on internal learning, which learns from the test image itself, the defect distortion will be minimal. We use ZSSR [Shocher *et al.* (2018)] as the Blind SR model. We us a fully convolutional network, with 8 hidden layers, each having 64 channels. We use ReLU activations on each layer. The network input is interpolated to the output size. We only learn the residual between the interpolated LR and its HR parent. We use L1 loss with ADAM optimizer. We start with a learning rate of 0.001. We periodically take a linear fit of the reconstruction error and if the standard deviation is greater by a factor than the slope of the linear fit we divide the learning rate by 10. We stop when we get to a learning rate of 10^{-6}.
To accelerate the training stage and make the runtime independent of the size of the test image I, at each iteration we take a random crop of fixed size from a randomly-selected father-son (refer chapter 2) example pair. The crop is typically 128×128 (unless the sampled image-pair is smaller). The probability of sampling a LR-HR example pair at each training iteration is set to be non-uniform and proportional to the size of the HR-father. The closer the size-ratio (between the HR-father and the test image I) is to 1, the higher its probability to be sampled. This reflects the higher reliability of non-synthesized HR examples over synthesized ones.

Our problem is a case of $4\times$ SR. Although [Shocher *et al.* (2018)] recommends using a gradual increase in resolution, we do not see a performance difference between gradually going from $\{1\times, 2\times, 4\times\}$ and directly performing a $4\times$ SR. We use the latter for the better efficiency.

A few results are shown in figures 3.1-3.7. The leftmost image is the LR image. It has

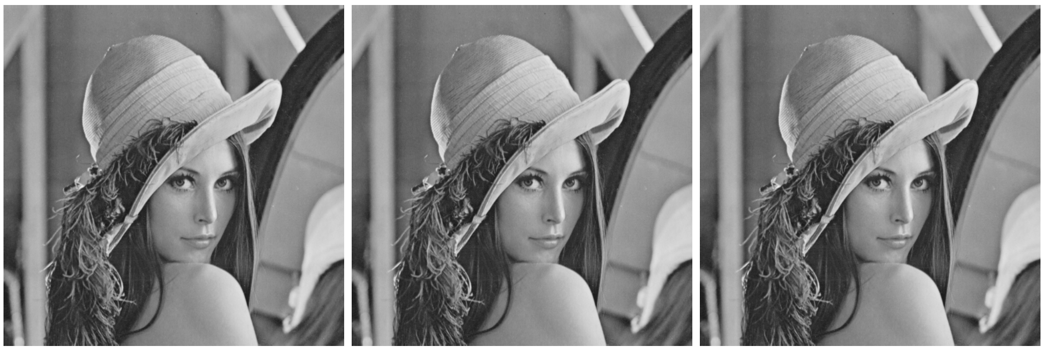


Figure 3.1: LR, SR, and HR Image

been bicubically upscaled to increase the visibility. The image on the center is the SRd image. The right most image is an example HR image. We call it as an example HR image as we do not have the ground truth HR images. We just have a reference image containing the same type of defect as in the LR image to give us an idea of how good is our SRd image. Unless otherwise mentioned, all results will be in this format. The (upscaled) LR image on the left, the SRd image in the center, and the example HR image on the rise. Also, we will be dropping the term "example HR" in favor of just "HR" in the following sections of this report for the sake of brevity. It should be understood that from this part of the report onward, unless otherwise mentioned, HR refers to the example HR image. Also, figures 3.1-3.7 cover all types of defects which are of interest to us. All of them have been presented to have a general idea of what each defect looks like. From the next set of results, if a method performs poorly on a few types of defects, we present its outputs only on those types of defects, instead of presenting on all seven. This will prevent the report from being unnecessarily long and filled with bad results.

There are two observations to be made. The defect structure is preserved completely as we expected. However, the overall look of the image is nowhere close to the HR image. This is not a fault of ZSSR itself and more due to the fact that the problem is not one of just SR but also involves a large domain shift. Internal learning alone cannot

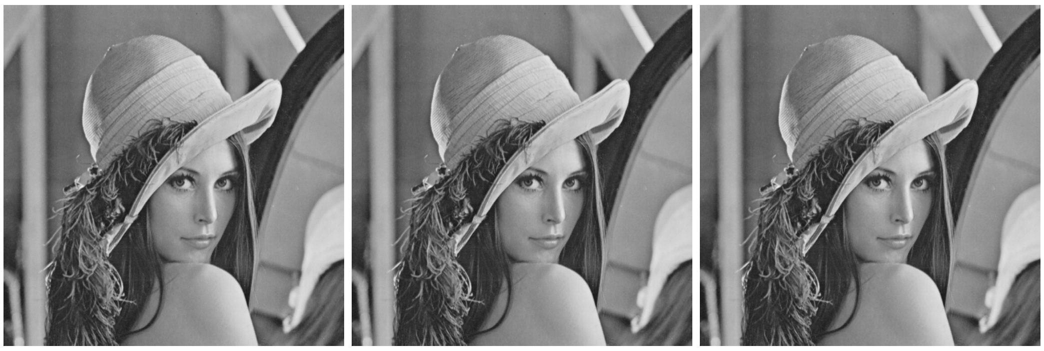


Figure 3.2: LR, SR, and HR Image

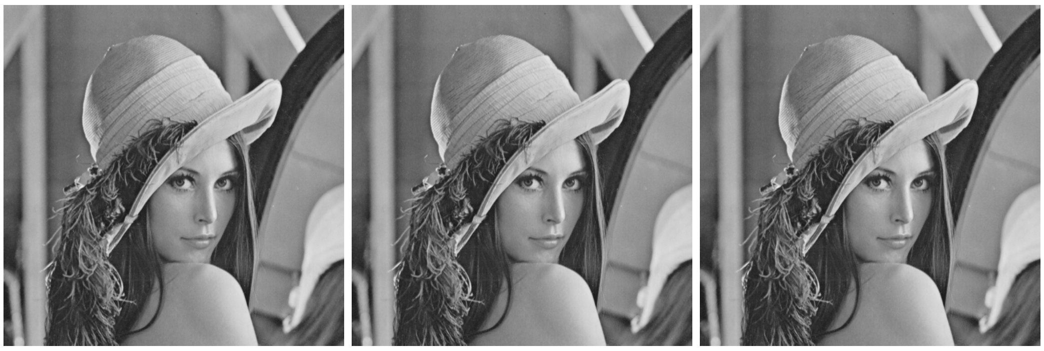


Figure 3.3: LR, SR, and HR Image

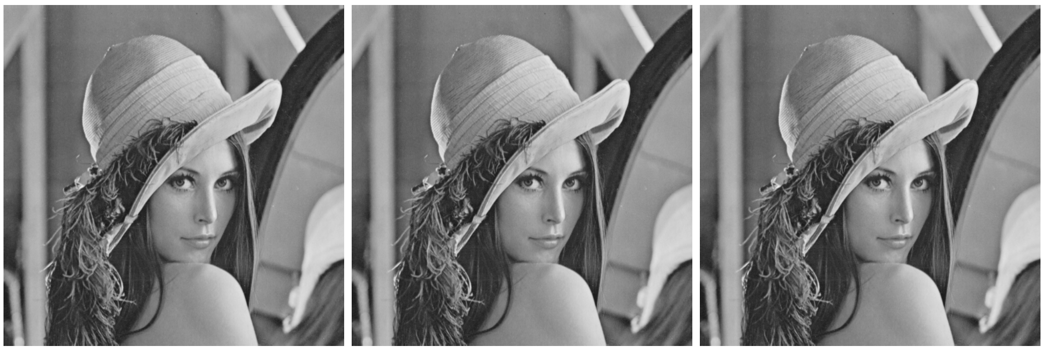


Figure 3.4: LR, SR, and HR Image

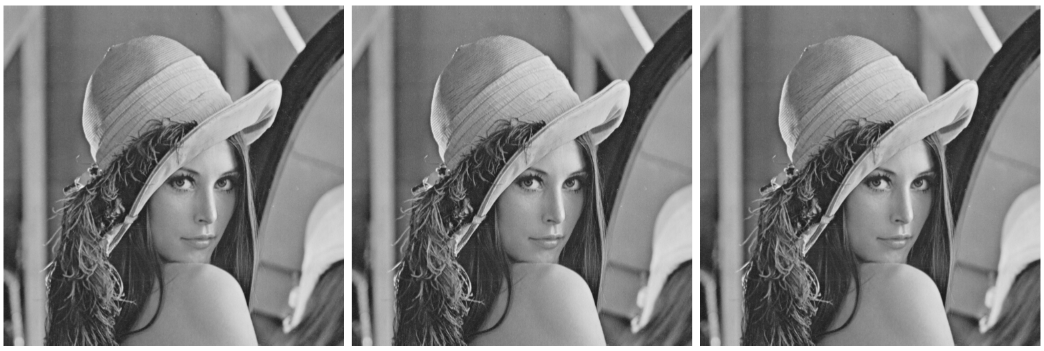


Figure 3.5: LR, SR, and HR Image

Figure 3.6: LR, SR, and HR Image

Figure 3.7: LR, SR, and HR Image

take care of domain shift as that information just does not exist in the test image.

3.2 ZSSR + pix2pix

The idea here is that while ZSSR manages to successfully perform SR without affecting the defects, there is a large domain gap which it fails to match. We can try to overcome that domain gap by performing domain transfer on the SRd images. We use pix2pix[Isola *et al.* (2017)] to perform the style transfer. The images in the two domains are the LR train images SRd using ZSSR and the corresponding HR images. The discriminator used is a PatchGAN discriminator. The loss used was a combination of the L1 loss and the GAN loss, as mentioned in chapter 2.

3.2.1 ResNet Generator

We first start with a ResNet[He *et al.* (2016)] based generator. The generator is a cascade of two downsampling layers, followed by 9 ResNet blocks, followed by two upsampling layers. The network was trained on crops of size 256×256 for 400 epochs. The Adam optimizer was used with a learning rate of 0.0002, and momentum parameters $\beta_1 = 0.5, \beta_2 = 0.999$. A few results can be seen in figures 3.8 and 3.9. In this case the SR image refers to pix2pix run on ZSSR SRd images.

It is clear that although the background looks very close to the HR images now, the

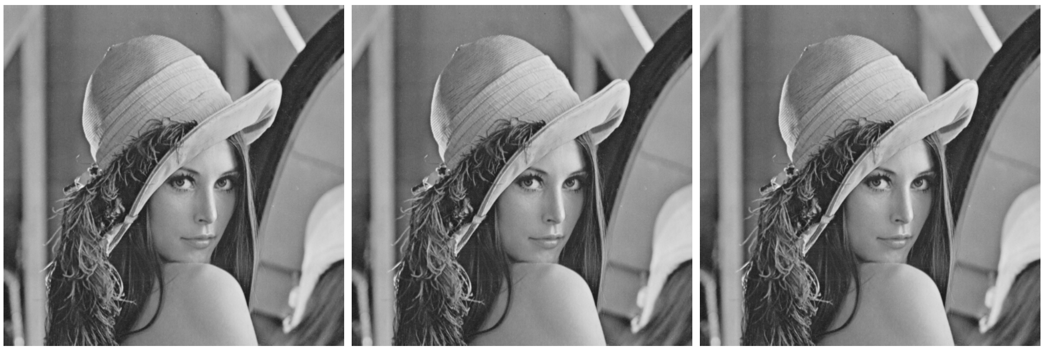


Figure 3.8: LR, SR, and HR Image

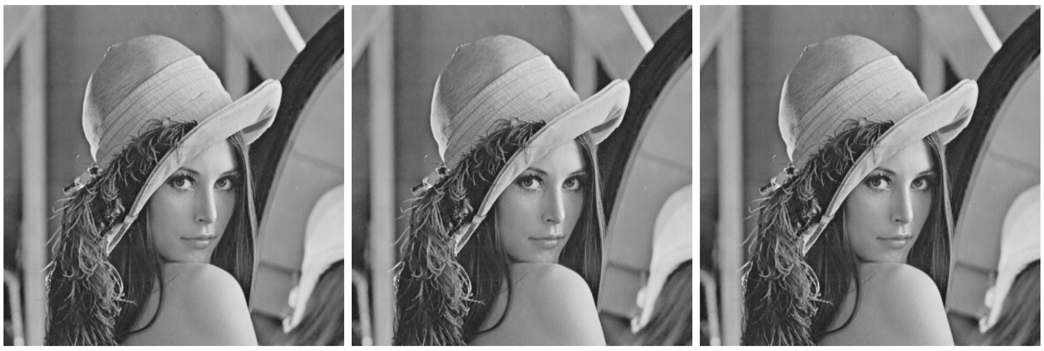


Figure 3.9: LR, SR, and HR Image

defects are completely distorted. This bring us back to the old problem of defects being distorted at test time. We try to overcome this using data augmentation.

It is to be noted that now we are dealing with the ZSSR images along with the HR images for the training. We do not really need the LR images for now for the domain transfer application. For data augmentation, we use the the DIV2K dataset [Agustsson and Timofte (2017)]. We convert the HR images from the dataset to black and white. We then randomly select a div2k image and take n_1 random crops from it. The sizes of the crops is chosen randomly from a set of predefined sizes which roughly cover the sizes of all the defects. These n_1 crops are placed in random locations on both the HR image as well as the ZSSR SRd image. We do this n_2 times for each training image. The idea is that the network should know that if it comes across something which is not a background, it should leave it as it is. n_1 decides how many random patches appear on an image, and n_2 decides the size of the augmented dataset. A few augmented images are shown in figures 3.10 - 3.13. For a given value of n_1 and n_2, for each original image, there are n_2 augmented images each with n_1 crops from the DIV2K dataset. Apart from this, there is the original image with no augmentation.

Figure 3.10: Augmented ZSSR SR and HR for $n_1 = 1$

Figure 3.11: Augmented ZSSR SR and HR for $n_1 = 1$

Figure 3.12: Augmented ZSSR SR and HR for $n_1 = 10$

We start for the case of $n_1 = 1$ and $n_2 = 1$. The training configuration is maintained the same as the previous section. Some results can be seen in figures 3.14 - 3.16.

Figure 3.13: Augmented ZSSR SR and HR for $n_1 = 10$

Figure 3.14: LR, SR, and HR Image

Figure 3.15: LR, SR, and HR Image

The results are a lot better now with a much larger part of the defect structure preserved while retaining a good background. Clearly the data augmentation is helping. This motivates us to increase the augmentation. We next try with $n_1 = 10$ and $n_2 = 1$ with everything else kept the same. The results are shown in figures 3.17 - 3.18.

Contrary to expectations, the results now are worse than the case with $n_1 = 1$. We also tried the case with $n_1 = 10$ and $n_2 = 10$ whose results are shown in figures 3.19 -

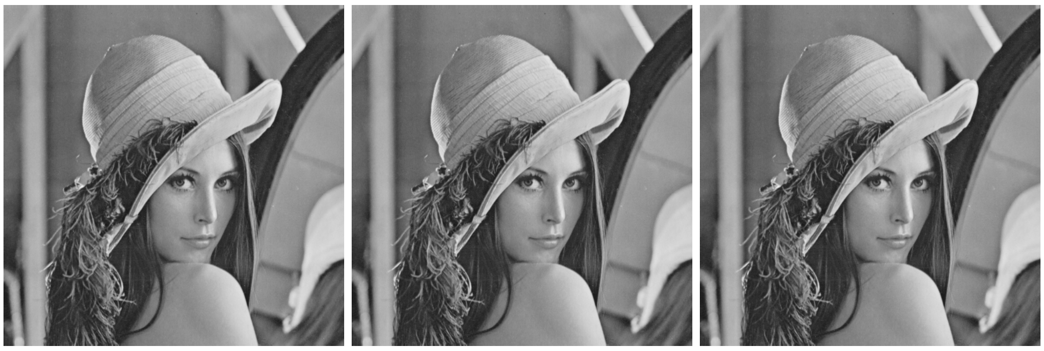


Figure 3.16: LR, SR, and HR Image

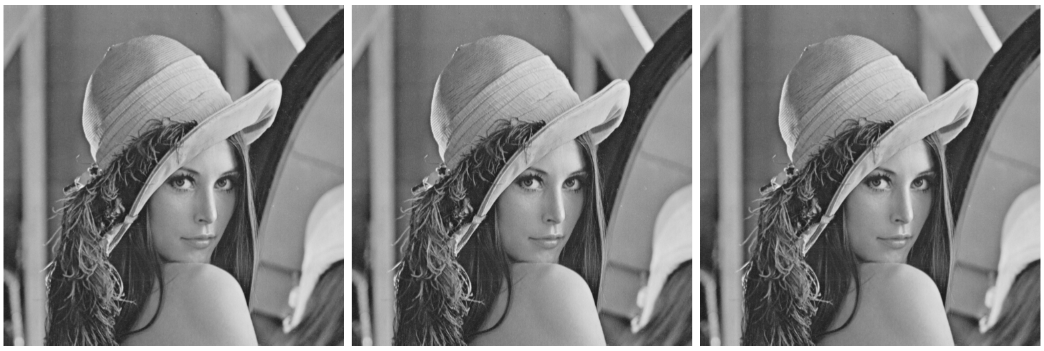


Figure 3.17: LR, SR, and HR Image

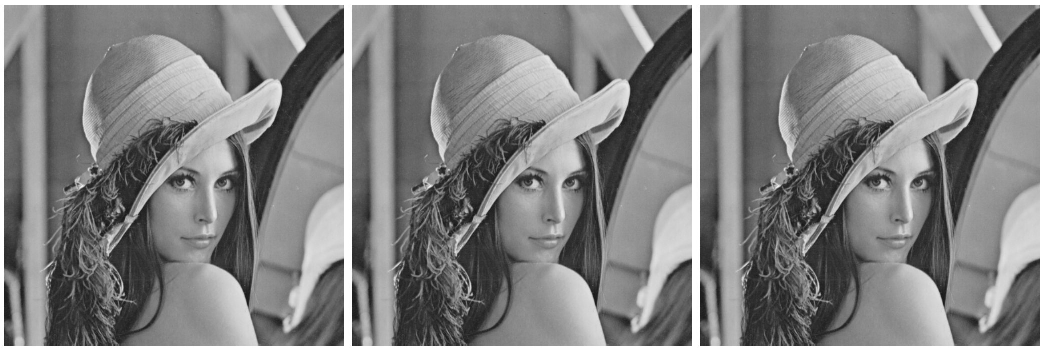


Figure 3.18: LR, SR, and HR Image

3.20.

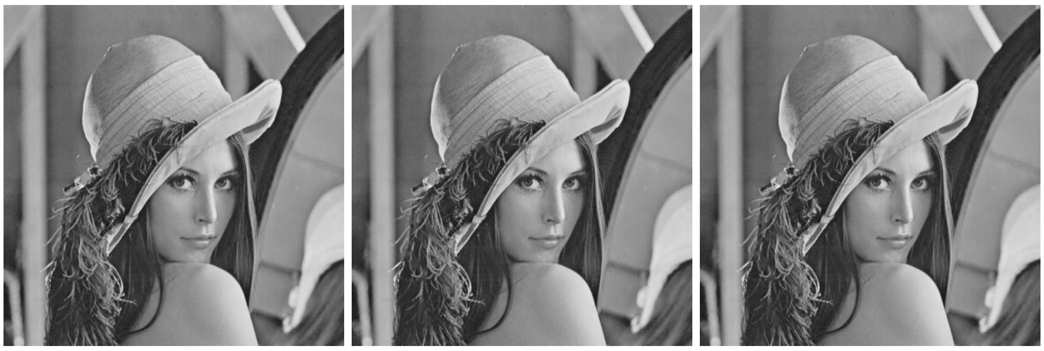


Figure 3.19: LR, SR, and HR Image

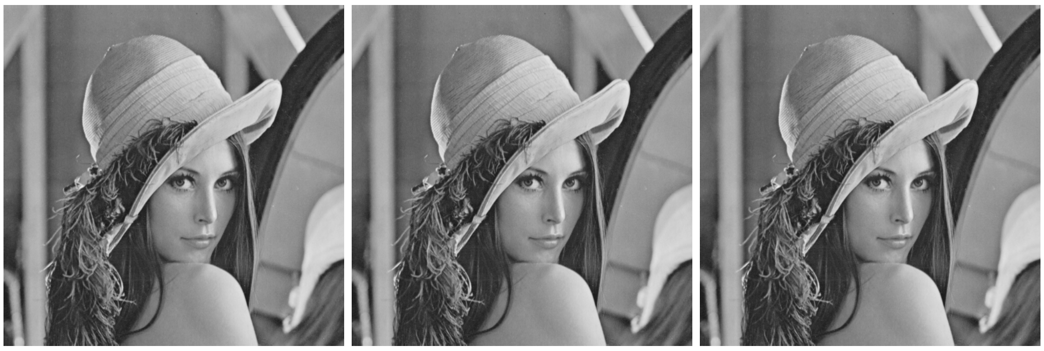


Figure 3.20: LR, SR, and HR Image

Strangely, the results are even more off this time. Although counter-intuitive, this is what happened on repeated experimentation. We tried removing the GAN loss to see if that helped preserve structure. Although it did help preserve structure, it defeats the purpose of the experiment as the model trained with L1 loss only failed to do the domain transfer accurately. The results also look over-smoothed. This motivates us to look elsewhere.

3.2.2 U-Net Generator

Since the network must be such that the domain transfer is performed mostly on the background, and anything which is not the background must be left as it is, we try with a U-Net [Ronneberger *et al.* (2015)] generator instead of the ResNet based generator as we feel the skip connections can help with letting the defects pass through as they are without distorting them. We start with U-Net 256 with a crop size of 256 and augmented data with $n_1 = 10$ and $n_2 = 1$. The results are present in figures 3.21 - 3.23. We did try the experiment without any data augmentation as well as for the case $n_1 = 1$ and $n_2 = 1$ but that led to complete distortion of the defects and hence they have not been presented.

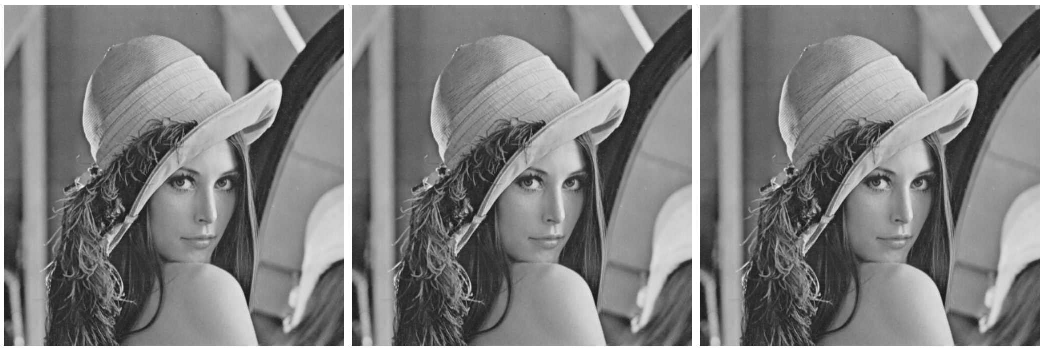


Figure 3.21: LR, SR, and HR Image

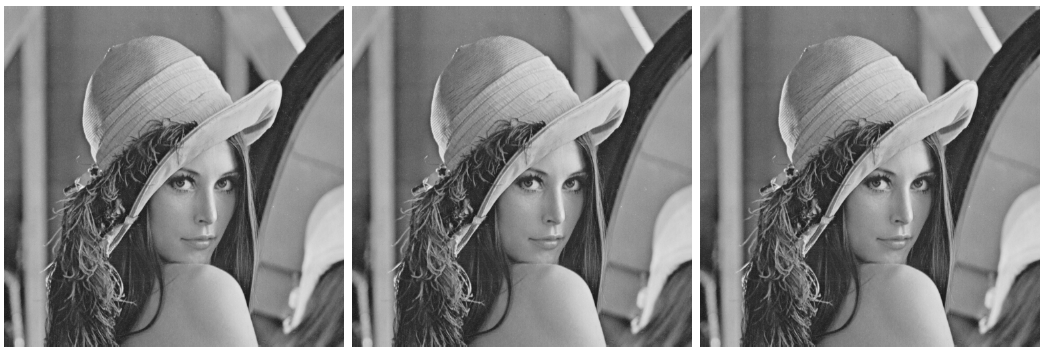


Figure 3.22: LR, SR, and HR Image

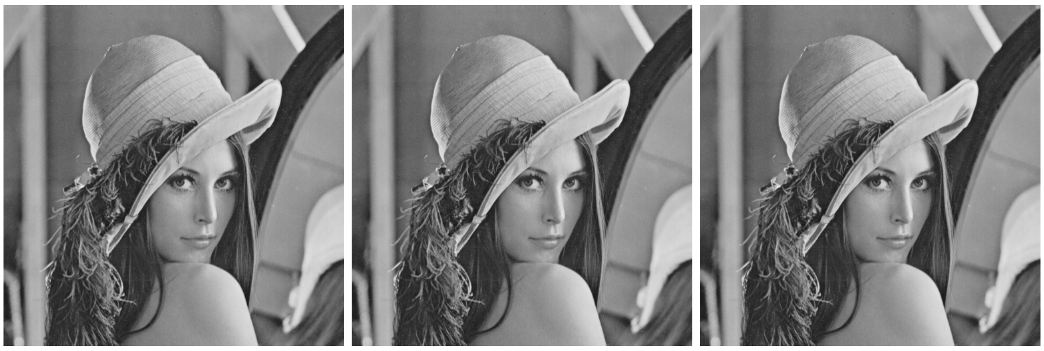


Figure 3.23: LR, SR, and HR Image

The results are now much better than what we had for the ResNet based generator. The defect structure is maintained a lot better while keeping the background looking close to the HR image too. However, there are some artefacts present in the output images (bottom left corner of figure 3.21). Apart from that, some of the defects span a larger area than they should. We experiment with different numbers of U-Net blocks to see what effect does it have. We first try with U-Net 128 instead of U-Net 256 keeping everything else the same. The results can be seen in figures 3.24 - 3.26.

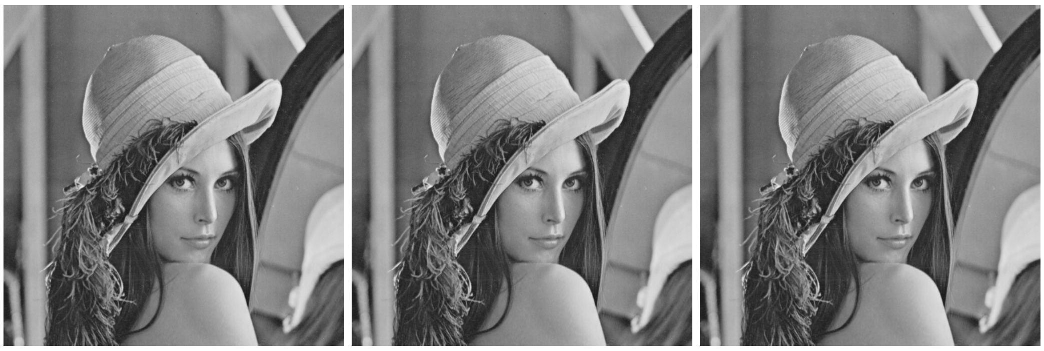


Figure 3.24: LR, SR, and HR Image

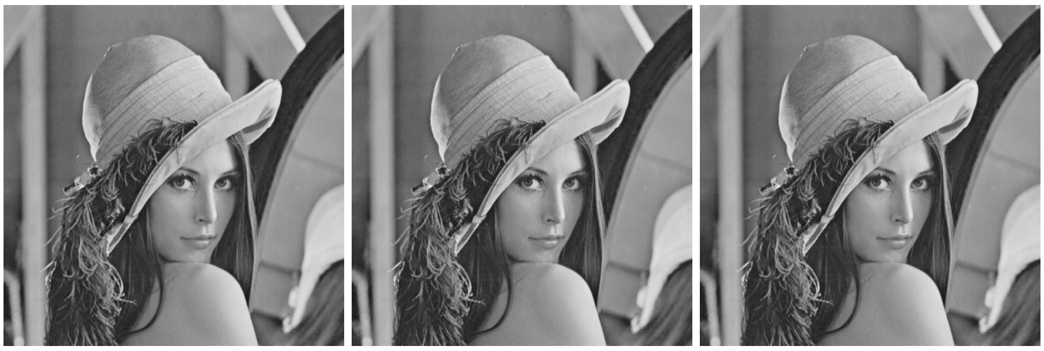


Figure 3.25: LR, SR, and HR Image

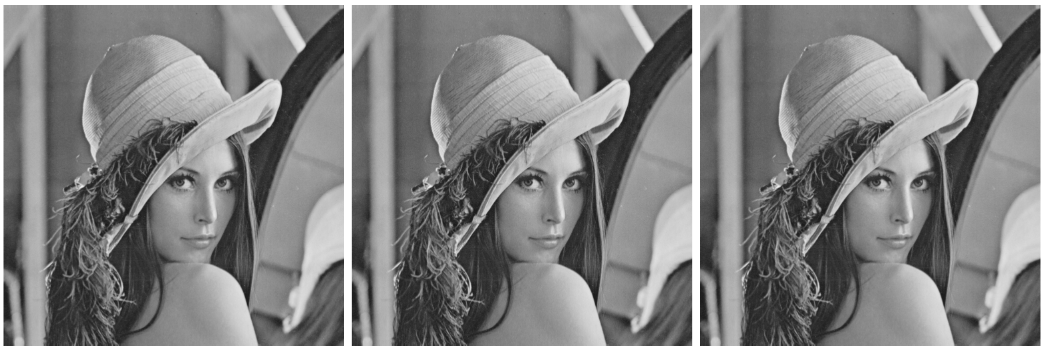


Figure 3.26: LR, SR, and HR Image

The artefacts issue is gone but now the problem of some of the defects spanning a larger area than they should is still there. What if we vary the crop size along with the number of U-Net blocks? To see why this is relevant, we demonstrate an extreme edge case with an U-Net 1024. For this case, we have no choice but to have a crop size of at least 1024 because of the repeated downsampling taking place in U-Net blocks. The results are shown in figures 3.27 and 3.28 respectively.

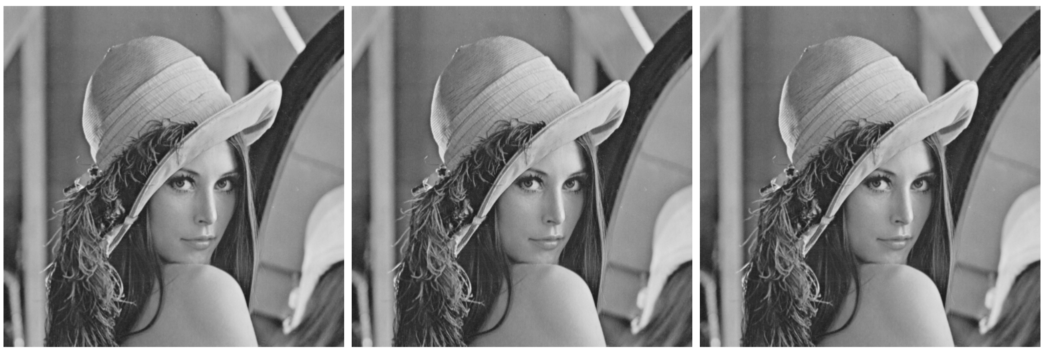


Figure 3.27: LR, SR, and HR Image

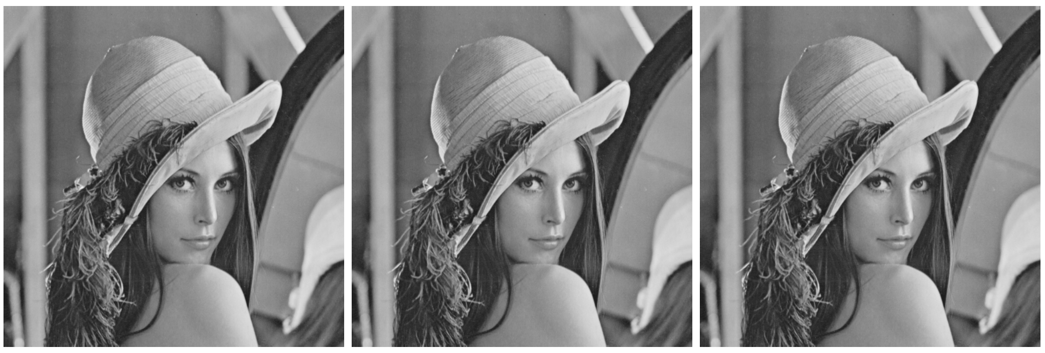


Figure 3.28: LR, SR, and HR Image

Although the U-Net 1024 block has at least as much modelling capacity as the U-Net 256 block, it struggles with both the background as well as the defects. We hypothesize that this is due to the fact that a larget U-Net block looks at a much larger window of data for context, and such a large window does not necessarily always benefit. Hence, training U-Net 128 with a different crop size might help as it might localise the defects better. We try with a crop size of 128 now instead of 256 keeping everything else constant. The results can be seen in figures 3.29 - 3.35.

The defects now do not have the problem of being larger than they are supposed to. The background too looks natural. The only issue is that for a particular type of defect (figure 3.31), the background looks much darker than it is supposed to.
This can be improved upon by using a U-Net 64 with a crop size of 64 as seen in figure 3.36 but that struggles with maintaining the structure of other types of defects as shown

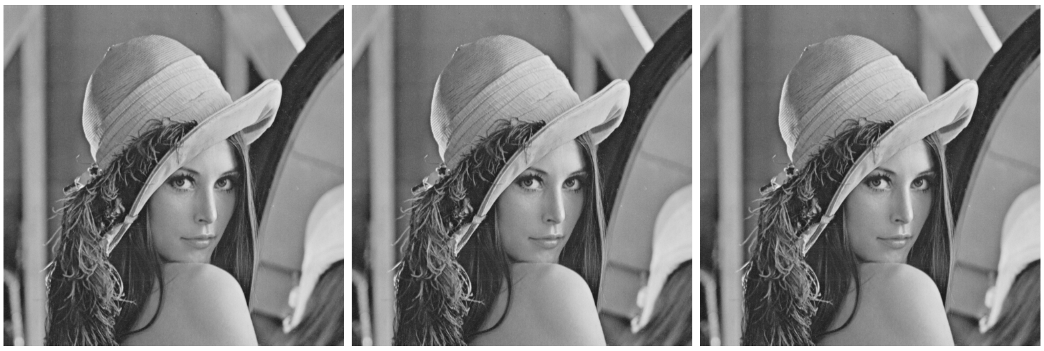


Figure 3.29: LR, SR, and HR Image

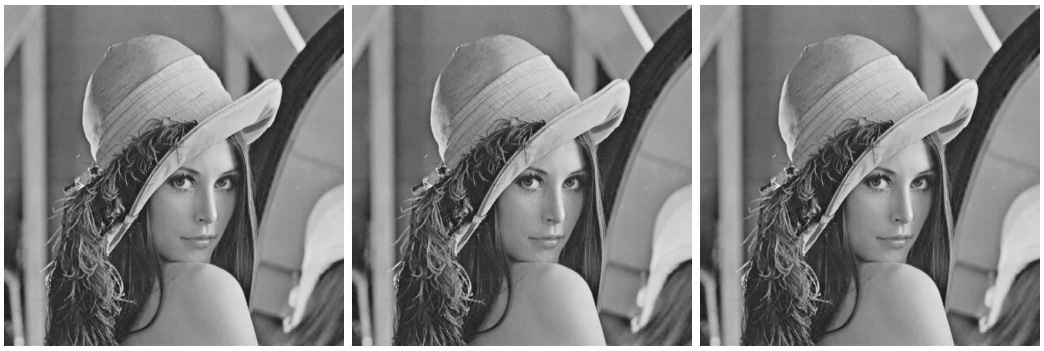


Figure 3.30: LR, SR, and HR Image

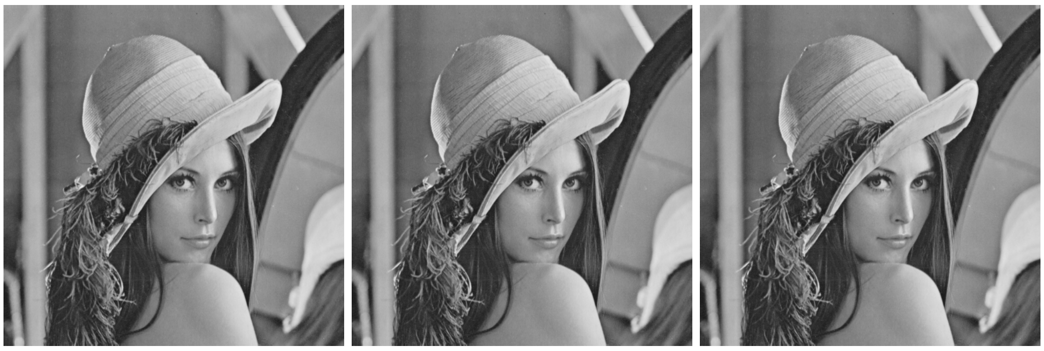


Figure 3.31: LR, SR, and HR Image

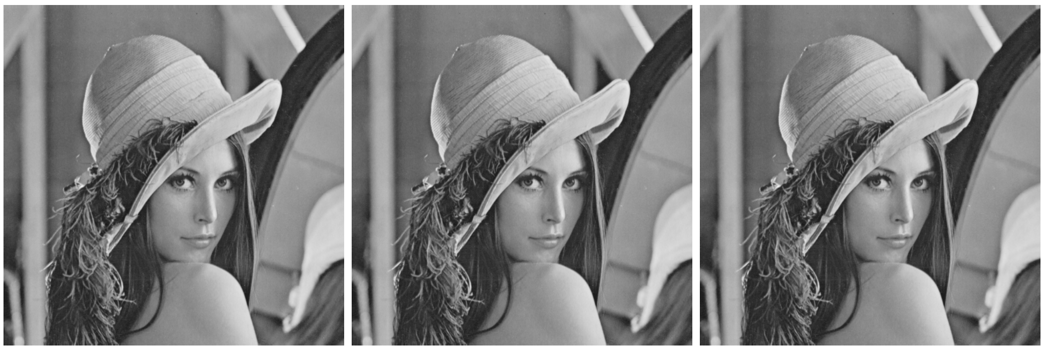


Figure 3.32: LR, SR, and HR Image

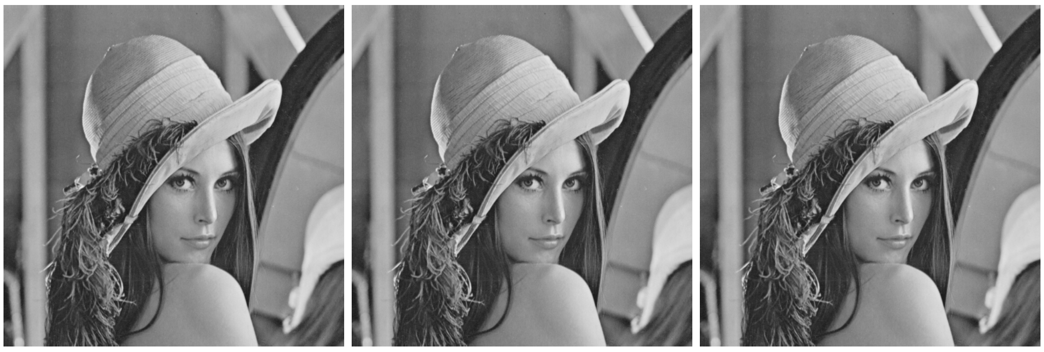


Figure 3.33: LR, SR, and HR Image

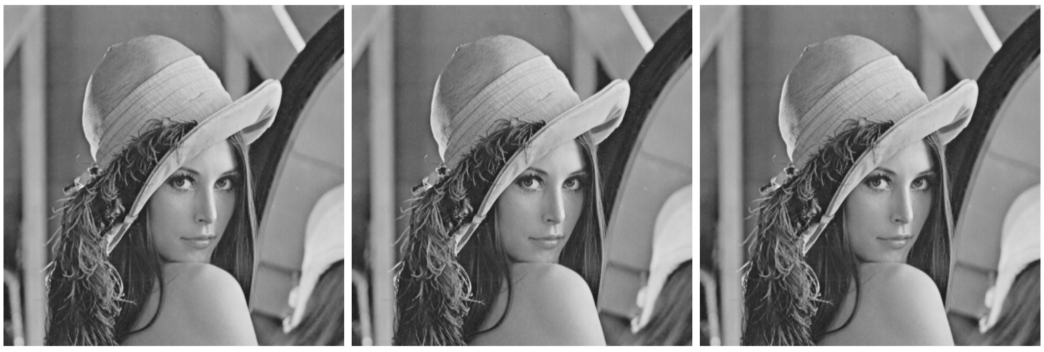


Figure 3.34: LR, SR, and HR Image

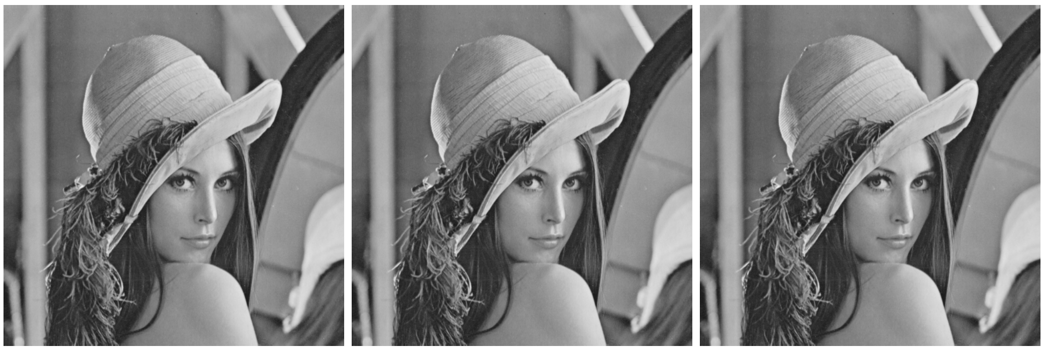


Figure 3.35: LR, SR, and HR Image

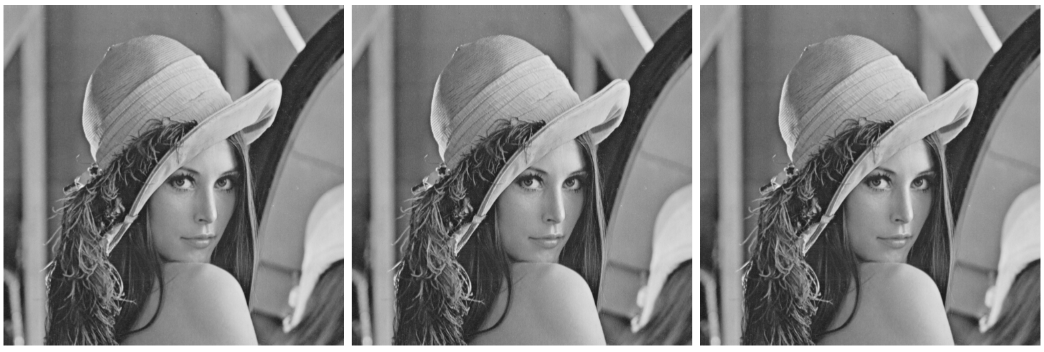


Figure 3.36: LR, SR, and HR Image

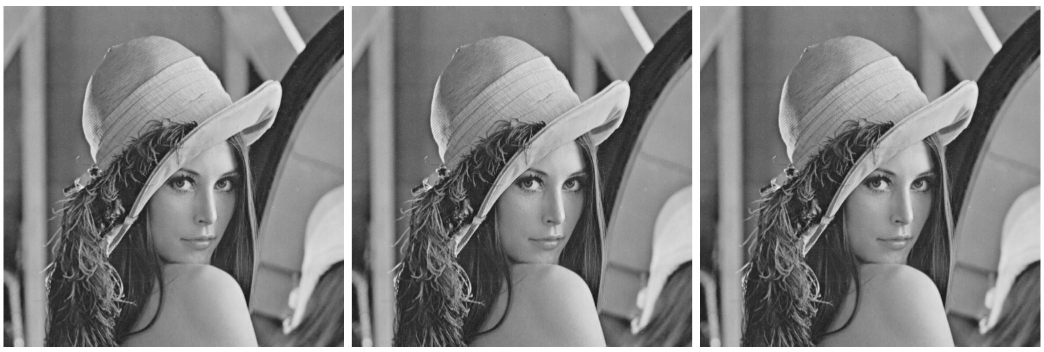


Figure 3.37: LR, SR, and HR Image

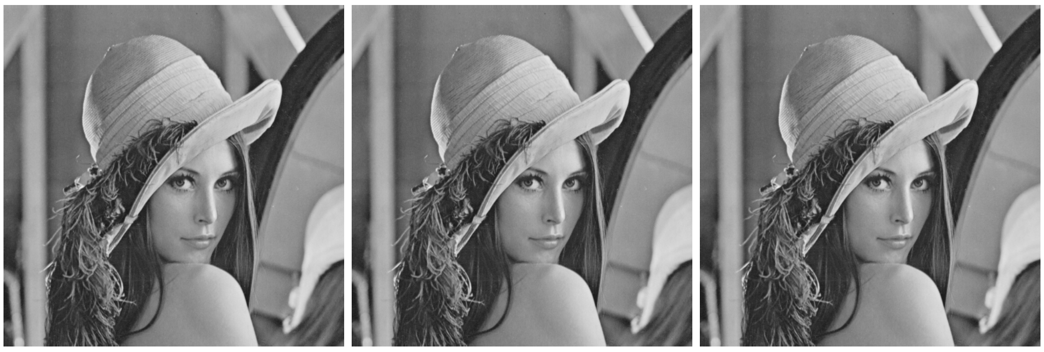


Figure 3.38: LR, SR, and HR Image

in figures 3.37 and 3.38.

After this, we had a meeting with the KLA team. The feedback we got from them was that the speed of the method is of utmost importance. Hence, Blind SR methods, or any method which performs updates during inference is ruled out. Any suggested method must have a high throughput. Hence, we do no proceed further with ZSSR.

3.3 MASA-SR

After the feedback from the KLA team, we tried to see if we could make use of the progress made so far in any manner. We got rid of ZSSR and tried to learn a domain transfer between the bicubically upsampled LR images and the HR images. However, no configuration of generators and training crop size in pix2pix gave us decent results. It seems like the high frequency details maintained by ZSSR compared to plain bicubically upsampling the image were vital for the domain transfer problem. The line of research hence had to be scrapped and we had to start afresh.

Our next idea was to see if we could make use of the development in reference based SR methods for our problem. However, there are two things to consider. First, reference based SR methods are usually slower than other deep-learning based methods as they have to perform template matching between the LR image and the reference image. Hence, we choose MASA-SR [Lu *et al.* (2021*b*)], where the Match and Extraction Module (MEM) significantly reduces the computational cost by a coarse-to-fine correspondence matching scheme. The second issue is how to choose the reference image at test time. If we use a reference image which contains the same type of defect as the test image, albeit at a different spatial location, we are making the assumption that the wafer already has a defect and we know which type of defect it is. This in indeed a problem. However, for the time being, we assume we know the type of defect we have in the test image, and provide a reference image containing the same type of defect as the test image during inference too. This is a much easier setting than the actual problem, but it can give us an idea of whether if it is worth to proceed with MASA-SR.

We train the MASA-SR with the same architecture as mentioned at the end of section 2.3. We train the model with the Adam optimizer by setting $\beta_1 = 0.9$ and $\beta_1 = 0.999$. The learning rate is set to 10^{-4}. We use crops from the LR image of size $40x40$ and the corresponding $160x160$ crops from the HR images. As per the recommendation of the authors [Lu *et al.* (2021*a*)], we first train masa-rec only with the reconstruction loss for 250 iterations. After getting masa-rec, we train masa with all losses, which is based on the pretrained masa-rec for 50 more iterations. For now, we choose the reference image to be same as the ground truth HR image. A few results can be seen in figures 3.39 and 3.40.

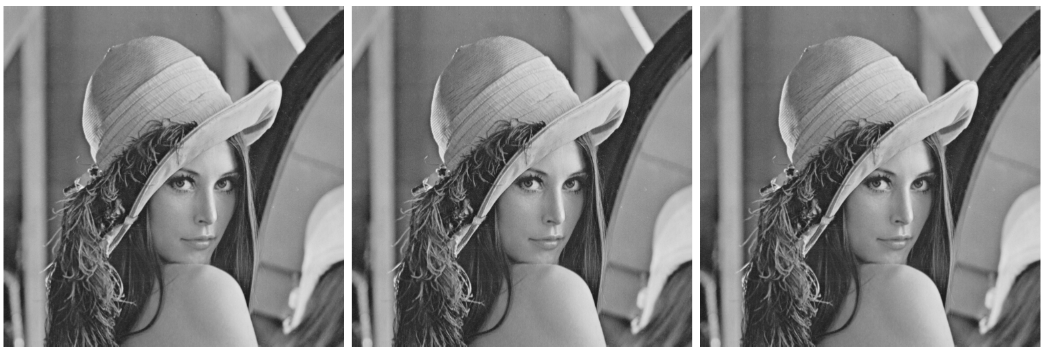


Figure 3.39: LR, SR, and HR Image

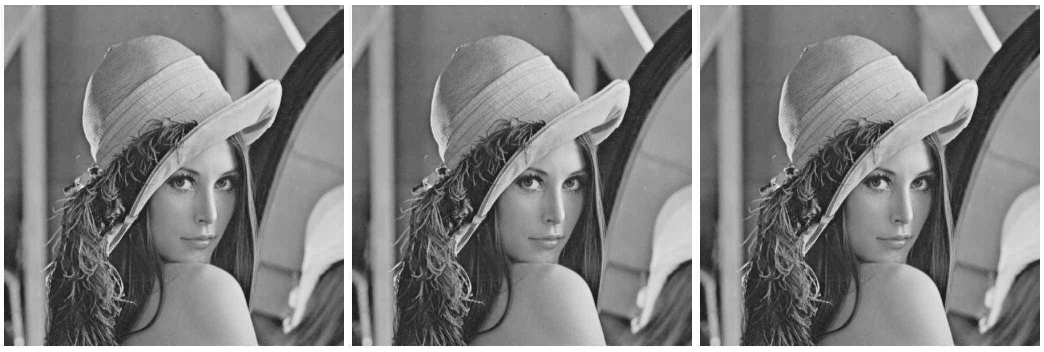


Figure 3.40: LR, SR, and HR Image

Clearly, this did not work out very well. The defects are almost completely gone. We felt this might be due to the fact that given the HR and reference image are the same, which means the reference image is exactly aligned with the LR image. That might cause the MEM to not train properly as it does not have to actually search for best matches, as that is the same location in the reference image as the LR image. Hence we decide to try with a different strategy for choosing the reference image. We choose the reference image as a random patch of the same size as the HR image from any of the different HR images. A few results can be seen in figures 3.41 and 3.42.

We do not see any sort of improvements at all. We could go about performing data augmentation as we did in section 3.2. However, in this case, the network had access to the exact same defect type even during inference. Yet, it did not translate into good results. The actual problem is a lot harder where we will not have access to the defects during inference. Given the method is not giving anything remotely promising, we did not see it wise to proceed with it. We decided to scrap it completely and start fresh again.

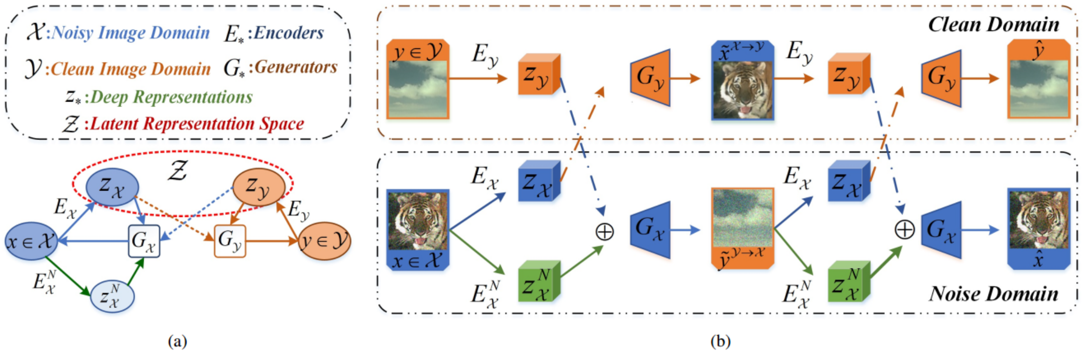
Figure 3.41: LR, SR, and HR Image

Figure 3.42: LR, SR, and HR Image

3.4 Image Translation and Restoration

A major issue with all the methods discussed so far was that they need perfectly aligned data. This is especially a problem in our case because the data that we got from the KLA team was not aligned. We had to manually align the data before training the models. To get rid of this limitation, we decided to experiment with methods which do not require paired training data. We, hence, decided to focus on unsupervised domain transfer and rectification methods which we felt would be useful for our problem statement.
We first tried with the method described in Learning Invariant Representation for Unsupervised Image Restoration [Du *et al.* (2020)] (we will refer to this work as LIR from now on for simplicity). The method has been described in detail in section 2.2. The overall method can be seen in figure 2.4 but we still show it here for the sake of convenience of the reader. It has been shown in figure 3.43.

In the original work, $\mathcal{X}$ and $\mathcal{Y}$ denote the noisy and clean domains respectively. $E_{\mathcal{X}}, E_{\mathcal{Y}}, E^N_{\mathcal{X}}$ represent the content encoder for the noisy domain, content encoder for



$\mathcal{X}$:Noisy Image Domain
E_*:Encoders
$\mathcal{Y}$:Clean Image Domain
G_*:Generators
z_*:Deep Representations
$\mathcal{Z}$:Latent Representation Space
(a)
(b)
Clean Domain
Noise Domain

Figure 3.43: LIR methodology

the clean domain, and the noise encoder respectively. $G_{\mathcal{X}}$ takes the encoded content from the clean domain along with the encoded noise, and tries to generate a sample which should ideally be indistinguishable from the real noisy images. $G_{\mathcal{Y}}$ takes only the content encoded from the noisy image and tries to generate a sample which should ideally be indistinguishable from the real clean images. The second half of the photo where the cross-translation happens is used only to add constraints to the networks during training. During inference, only $E_{\mathcal{X}}$ and $G_{\mathcal{Y}}$ are needed for denoising.

For the initial experiments, we used the model as it is without changing any of the losses. For domain $\mathcal{X}$,we used the LR images upscaled. For domain $\mathcal{Y}$, we used the HR images. It is to be noted that there is no necessity of LR and HR images being aligned as the method is unsupervised. During training, we did not perform any sort of data augmentation. We used the trained $G_{\mathcal{Y}}$ and $E_{\mathcal{X}}$ for inference. Although the results were not that exceptional, especially the background not looking very natural, the defect structure was mostly maintained. This gave us the hope that by modelling the network better for our use case, we can get the defect structures maintained even without paired data.

Modelling the network for our use case We model the network to our use case in the following way.

- We choose $\mathcal{X}$ to be the domain of HR images and $\mathcal{Y}$ be the domain of the LR images bicubically upsampled. We make this choice as we feel the LR upsampled images have only the content information whereas the HR images also have a different style apart from the content.
- We expect $E_{\mathcal{X}}$ to model the content information in the HR images and $E^N_{\mathcal{X}}$ to model the style of the HR images.
- Similarly, we expect $E_{\mathcal{Y}}$ to model the content information in the LR upsampled images.

- During inference, we use $E_{\mathcal{Y}}$ along with $G_{\mathcal{X}}$. It is to be noted that $G_{\mathcal{X}}$ needs the encoded style information along with the content information. For this, we use one of the training HR images as a "reference" image and use $E_{\mathcal{X}}$ to encode the style information from it. This is used as input to $G_{\mathcal{X}}$ for all of the test images.
- We remove some of the losses which do not make sense in our use case. To be precise, we remove the Background Consistency Module (BCM) (equation 2.13) and Semantic Consistency Module (SCM) (equation 2.14). The former is removed because the backgrounds of both the domains in our case is very different and it makes no sense to force them to look the same. The latter is removed because the perceptual loss was found to be rather harmful in a previous project for our use case.

A few results can be seen in figure 3.44 and figure 3.45.

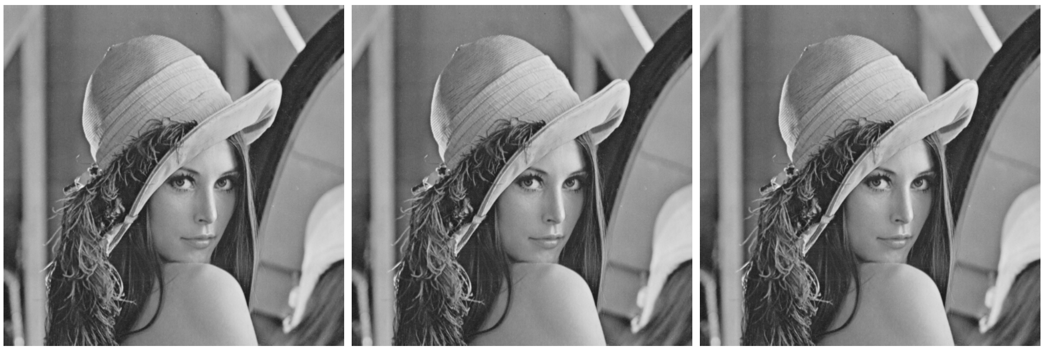


Figure 3.44: LR, SR, and HR Image

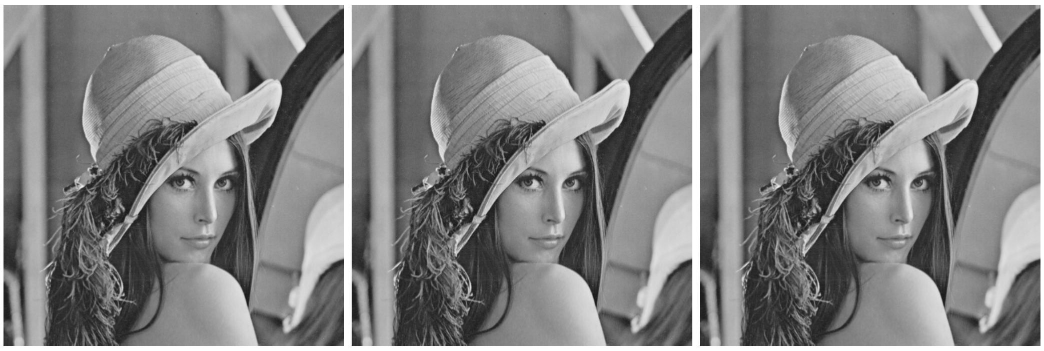


Figure 3.45: LR, SR, and HR Image

As it is clear, the defects are completely gone. But there is something even more worrying. Both the results look exactly the same. On further investigation, it was found that the output image is basically a copy of the reference HR image and has little to no correlation with the input image. This was consistent on training the model multiple times. Our idea was that the model was not adequately constrained. We wanted to ensure that the output is correlated with the input LR upsampled image rather than the

HR reference image. The latter must be used only to get the style features. To try and achieve this, we added a new loss while training.
Now, instead of sampling a single x from domain $\mathcal{X}$ and we sample multiple $x_i, \{i = 1, 2, ..., n\}$ from $\mathcal{X}$. Say the images in domain $\mathcal{X}$ produced by extracting the style features from x_i with the content features from y be $x_{i_y}, \{i = 1, 2, ..., n\}$. Ideally we want these images to be very similar, as the final image must be closely related to the content from the input image, and the reference image must be used only as a guide for the styling. Hence, we introduce the following loss.

$$\Sigma_{j=1}^{j=n}\Sigma_{i=1}^{i=j}\left\|x_{i_y} - x_{jy}\right\| \tag{3.1}$$

Although this loss is a bit stringent, as we expect the output images for different reference images for the same input to be very similar and not exactly the same, for now this is a good start as the model is not being constrained enough and is just copying the output image to be the same as the reference image. A few results can be seen in figure 3.46 and figure 3.47.

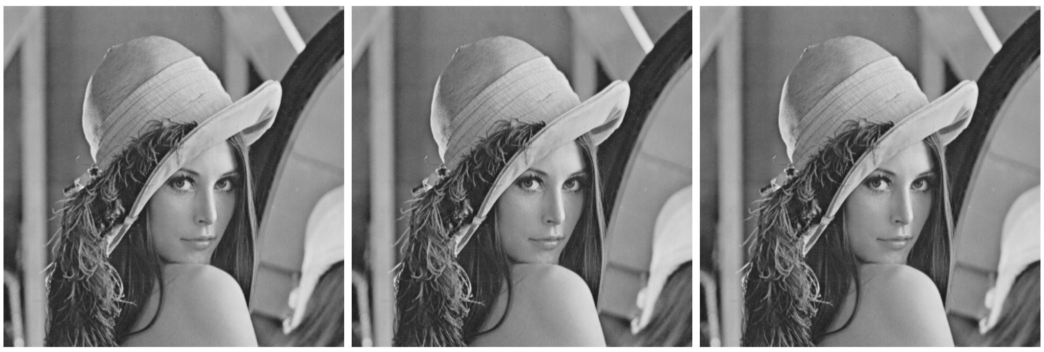


Figure 3.46: LR, SR, and HR Image

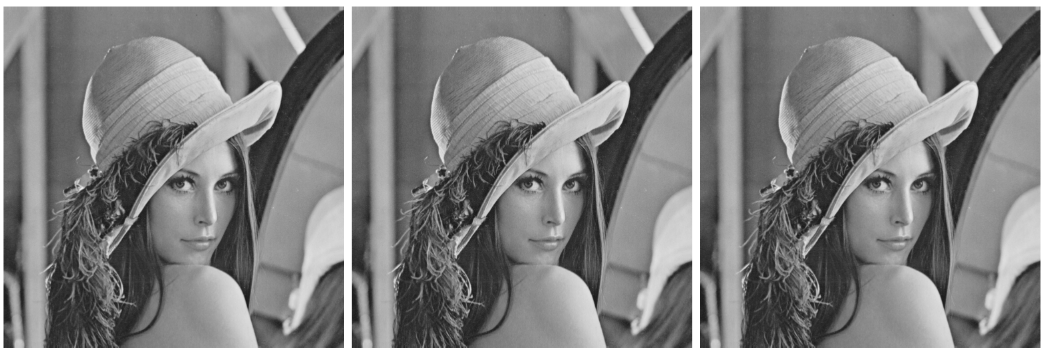


Figure 3.47: LR, SR, and HR Image

The same issue persists. Clearly the model needs to be constrained better. We need to add modules to ensure that the output image is heavily correlated to the input image. We look at the paper The Spatially-Correlative Loss for Various Image Translation Tasks [Zheng *et al.* (2021)] (we will refer to this work as F-LSeSim for brevity) which proposes a spatially-correlative loss that is simple, efficient and yet effective for preserving scene structure consistency while supporting large appearance changes during unpaired image-to-image (I2I) translation. Since the LIR model is heavy and takes a long time to train, we decide to first try the loss on a simpler architecture to see if it is worth proceeding in that direction. In this task, we use the CycleGAN as the reference architecture, but only use half of the pipeline and replace the cycle-consistency loss with the FSeSim loss (refer section 2.2). We used the ResNet based generator with PatchGAN discriminator (refer section 2.2). For the initial experiments, we use a crop size of 256 for training and a patch size of 64 for the spatial correlation loss. A few results are shown in figures 3.48 - 3.49.

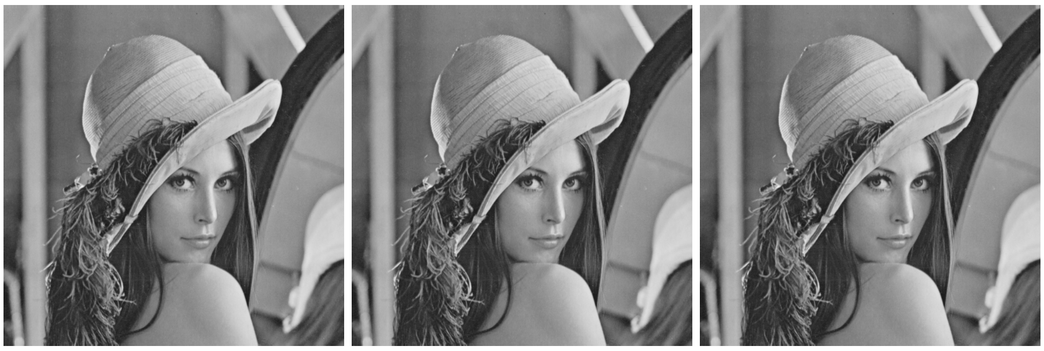


Figure 3.48: LR, SR, and HR Image

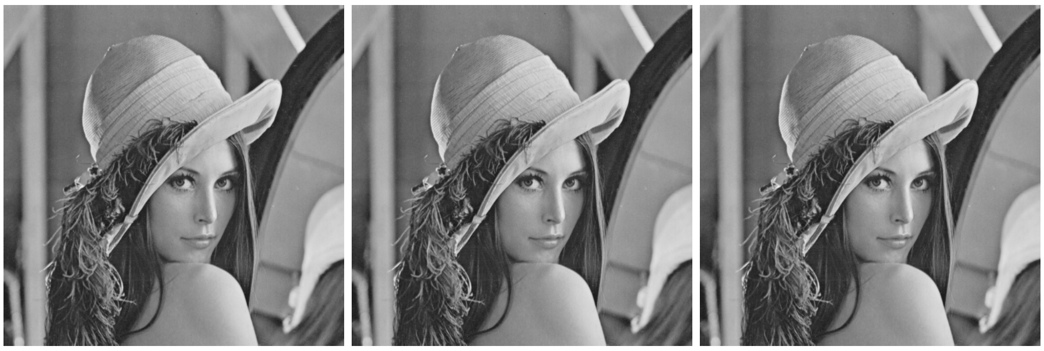


Figure 3.49: LR, SR, and HR Image

Clearly, the results are promising. The defects are maintained to the extent they are recognisable. However, there is still a non-negligible distortion of the defects. Apart, zooming into the background shows that it does not look as natural as the HR images. For the defect distortion, we try data augmentation similar to that suggested in section 3.2. We keep the model same, except for training it on augmented data. A few results are shown in figures 3.50 - 3.51.

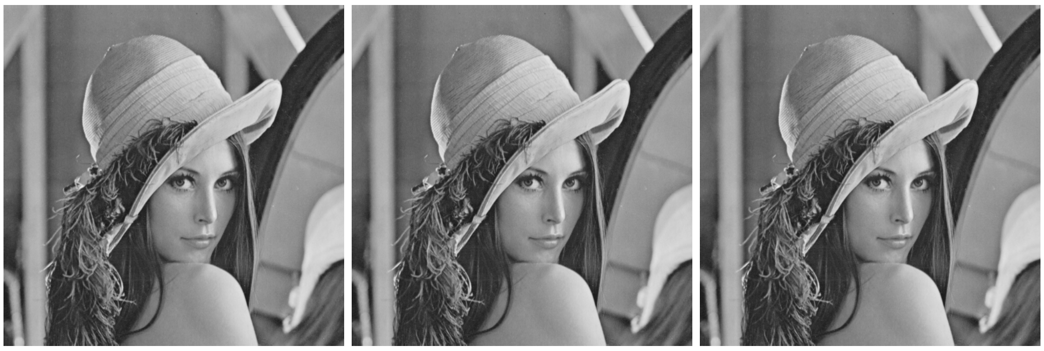


Figure 3.50: LR, SR, and HR Image

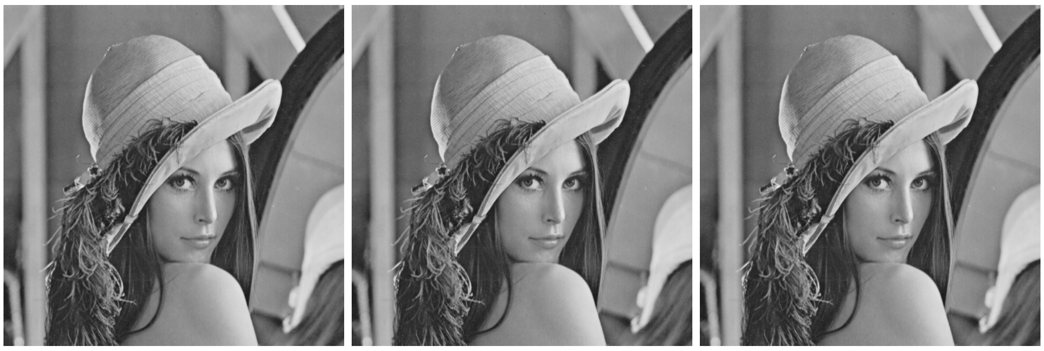


Figure 3.51: LR, SR, and HR Image

The defects are clearly preserved much better now. However, the background still does not look that natural on zooming in. We recall our observations in that the crop size was vital for performance. A crop size of 128 gave the best results overall. In this case, since the spatial correlation loss is calculated on patches on the crop taken during training, we hypothesize using a patch size of 128 could probably help. We try the same experiment with a crop size of 256 and a patch size of 128 without any sort of data augmentation. The results are shown in figures 3.52 - 3.58.

Although there is minor defect distortion, it is nothing catastrophic. The background now looks a lot better. This is important too as the defect detection network is trained

Figure 3.52: LR, SR, and HR Image

Figure 3.53: LR, SR, and HR Image

Figure 3.54: LR, SR, and HR Image

Figure 3.55: LR, SR, and HR Image

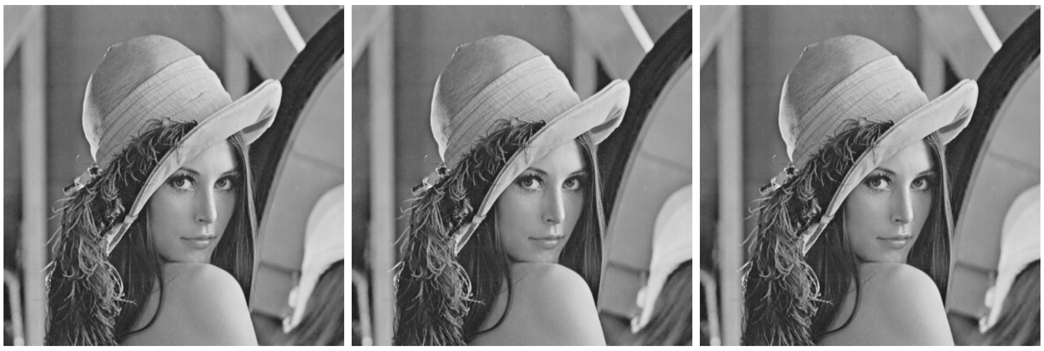


Figure 3.56: LR, SR, and HR Image

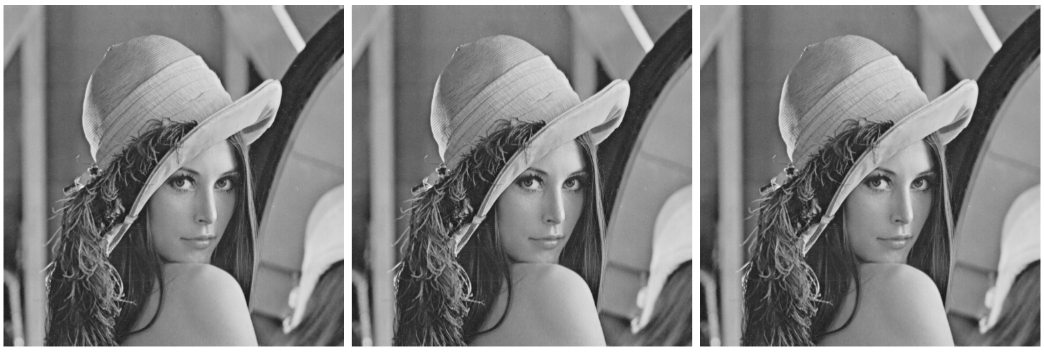


Figure 3.57: LR, SR, and HR Image

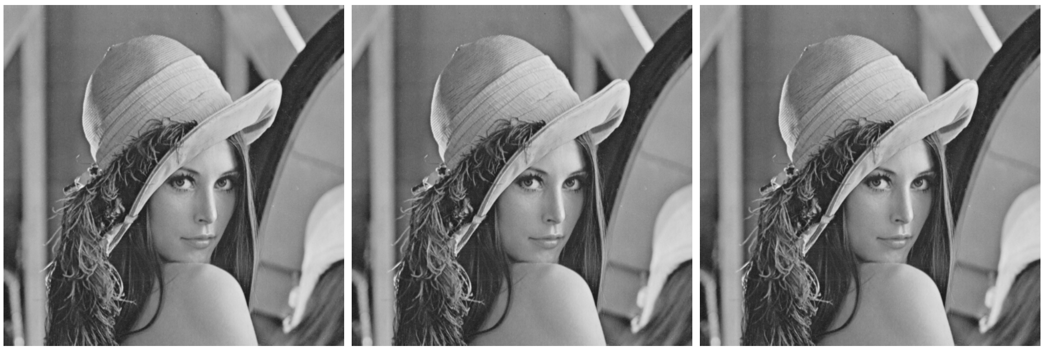


Figure 3.58: LR, SR, and HR Image

on the HR images and a natural looking background ensures it does not detect too many false positives.

We tried using different data augmentation methods as well as architectures for the generators (different U-Nets), however, we did not notice any significant improvement. We also tried the learned spatial correlation loss, but it was extremely unstable. Hence, results from it have not been included.

Chapter 4

Conclusion

In this report, we explore elegant solutions based on image-processing techniques for super-resolving SEM images in face of limited data, especially when there is a large domain shift between the LR and HR images. We show that Blind SR methods coupled with domain transfer methods are an obvious solution. However, they are slow and need paired data. We tackle an even more challenging version of the previous problem, assuming that we do not have any paired data and reply completely on unsupervised methods only. We hence explore different unpaired image to image translation techniques. We show that the spatial correlation loss emerges as a powerful tool to maintain structure while performing domain transfer, and can be used with the simplest of models to give very good results, with limited to no hackery needed.

The focus of this project was large on the image processing side of things: architectures, losses, pipelines etc. However, moving forward we believe a significant focus must lie on the optimization techniques too. Use of more modern and stable GANs is a direction to proceed in to stabilize the training even further. Another manner in which optimization comes into play is with regards to stabilising the learned spatial similarity loss. We feel by using the fixed spatial self similarity loss on the pre-trained VGG model, we are leaving performance on the table. However, we had no other choice as the former loss was unstable to train with, especially on such a small dataset. MoNCE [Zhan *et al.* (2022*b*)] can be a good place to start with stabilizing the learned spatial correlation loss. Even from an image processing perspective, a lot more can be explored along the lines of choosing better losses to compliment the spatial correlation loss. A direction to start might be looking at the StyleGAN [Karras *et al.* (2019)] paper and works based on it. Losses inspired from the StyleGAN paired with the spatial correlation loss may lead to even more natural looking backgrounds while keeping the defects intact during inference. Coordinate MLPs and Implicit Neural Representations are another exciting domain of research in image super-resolution which may be looked at for ideas. Disentanglement Representation Learning too may be used, either as a constraint, or even as an independent architecture by itself. The regularization needed can be provided by the spatial correlation loss.

Appendix A

Interesting Ideas

In this section we describe a few more interesting ideas which look promising. They have been mentioned here and not in the main section because of one or both of the mentioned reasons. First, the idea looks promising for the problem statement but it does not give good results yet. Second, the idea gives good results but more understanding is needed before conclusions can be made.

A.1 QS-Attention

Unpaired image-to-image (I2I) translation often requires to maximize the mutual information between the source and the translated images across different domains, which is critical for the generator to keep the source content and prevent it from unnecessary modifications. The self-supervised contrastive learning has already been successfully applied in the I2I. By constraining features from the same location to be closer than those from different ones, it implicitly ensures the result to take content from the source. However, previous work uses the features from random locations to impose the constraint, which may not be appropriate since some locations contain less information of source domain. Moreover, the feature itself does not reflect the relation with others. [Hu *et al.* (2022)] deal with these problems by intentionally selecting significant anchor points for contrastive learning. A query-selected attention (QS-Attn) module is designed, which compares feature distances in the source domain, giving an attention matrix with a probability distribution in each row. Queries are then selected according to their measurement of significance, computed from the distribution. The selected ones are regarded as anchors for contrastive loss. At the same time, the reduced attention matrix is employed to route features in both domains, so that source relations maintain in the synthesis.

Preliminaries on CUT

In the task of I2I, given an image $I_x \in R^{H\times W\times 3}$ from source domain X, an I2I translation model aims to translate it into $G(I_x)$ in the target domain Y , having no obvious distinctions with the real image $I_y \in R^{H\times W\times 3}$ in that domain. Generally, there are two auto-encoders, one $G_{X\to Y}$ for X to Y and the other $G_{Y\to X}$ for the reverse. CUT [Park *et al.* (2020)] focuses on the single direction translation from X to Y, so it only needs one generator G and one discriminator D, therefore, the subscript is omitted. The objective of adversarial loss L_{adv} can be computed as

$$L_{adv} = \mathbb{E}_{I_y\in Y} \log \mathrm{D}\left(I_y\right) + \mathbb{E}_{I_x\in X} \log\left(1 - \mathrm{D}\left(\mathrm{G}\left(I_x\right)\right)\right) \tag{A.1}$$

CUT also takes advantage of the first half of G as an encoder E to provide the extra constraint for the output from G. Basically, E compares the feature similarities across different domains. It extracts features from both I_x and $G(I_x)$, and establishes the self-supervised contrastive loss as

$$L_{\text{con}} = -\log\left[\frac{\exp\left(q\cdot k^{+}/\tau\right)}{\exp\left(q\cdot k^{+}/\tau\right)+\sum_{i=1}^{N-1}\exp\left(q\cdot k^{-}/\tau\right)}\right] \tag{A.2}$$

where q is the anchor feature from $G\left(I_x\right)$, k^+ is a single positive and k^- are $(N-1)$ negatives. Here τ indicates a temperature hyper-parameter. The anchor q always locates in the fake image, and its positive k^+ is on the same location in the real image I_x. Besides, $(N-1)$ negatives k^- are randomly selected in I_x. The gradient of L_{con} only applies on the anchor q to train the parameters in G, while it is detached on k^+ and k^-, so that G is guided for the single direction of domain translation.
The full objective is expressed as

$$L_G = L_{adv} + L_{con}^{X} + L_{con}^{Y} \tag{A.3}$$

where L_{con}^{X} is the contrastive loss defined in equation A.2, and L_{con}^{Y} is the identity loss, in which the positive k^+ and negatives k^- are from a real target domain image I_y, and the anchor q is from $G\left(I_y\right)$. This identity loss guarantees the features from $G\left(I_y\right)$ are similar with features from I_y, preventing G from making changes on the target domain images.

QS-Attn for Contrastive Learning

QS-Attention [Hu *et al.* (2022)] keeps the simple setting like CUT. E is applied to extract features from I_x and $G\left(I_x\right)$. These features are supposed to establish the con-

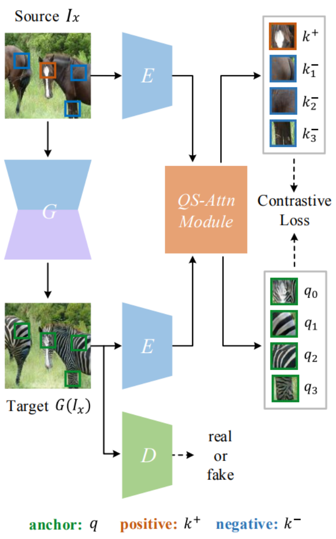


Figure A.1: The overall structure of QS-Attention. The source domain image I_x is translated by the generator G into a target domain image $G(I_x)$. The encoder E extracts features from these two images, then the $QS - Attn$ module selects significant features to establish the contrastive loss. A discriminator D is also used to construct the adversarial loss

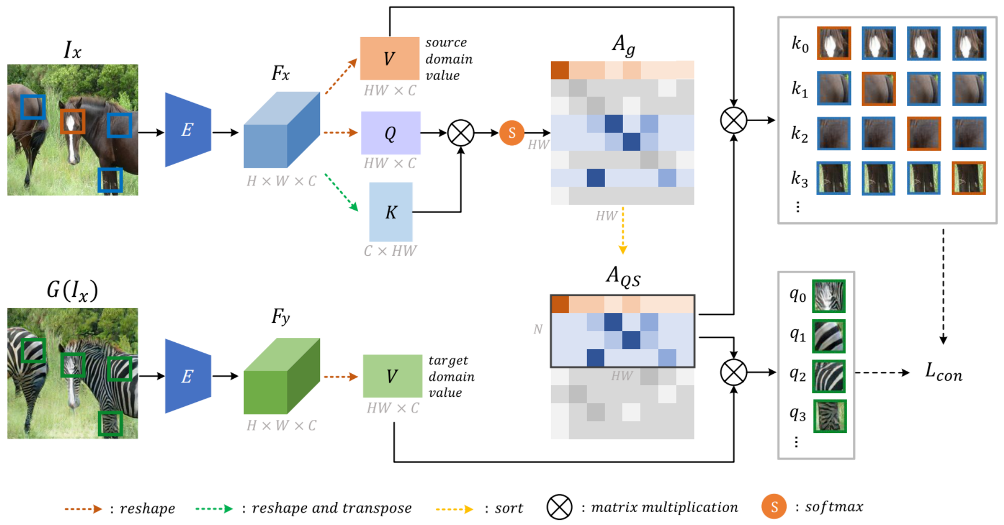


Figure A.2: The details of QS-Attn. The encoder E extracts features F_x and F_y from I_x and $G(I_x)$, then F_x is reshaped and computed to derive the attention matrix A_g. Each row in A_g is sorted by its metric of the significance, and the selected N rows forming the A_{QS}. A_{QS} is further applied to route both source and target domain value features, and obtain positive, negative and anchor features to construct the contrastive loss L_{con}. Positive and negatives are from the real image I_x, while anchors are from the translated image $G(I_x)$. The patches in orange, blue and green indicate the positive, negative and anchor, respectively.

trastive loss L_{con} defined in equation A.2. The key module QS-Attn, setting up L_{con} across two domains, is illustrated in figure A.2. Instead of the simple random strategy, the idea of attention is employed, which first compares a given query with keys, and then selects the query based on the comparison result. However, any separated projection head for query, key and value are not used like the common attention, therefore, without adding extra model parameters in both G and D.

Attention for query selection

CUT randomly selects the anchor q, positive k^+ and negatives k^- to compute the contrastive loss in equation A.2, which is potentially inefficient, because their corresponding patches may not come from the domain-relevant region. Note that some features do not reflect the domain characteristics, they tend to be kept during the translation. Therefore, the L_{con} imposed on them is not vital for G. The intention is to choose the anchor q, and compute L_{con} on the significant ones which contains more domain-specific information. Two forms of attention are to be noted, global attention and local attention.

Global Attention

Based on the above observation, the aim is to define a quantitative value for each potential location, which reflects the significance of the feature. The quadratic attention matrix is adopted, since it exhaustively compares each feature with all other locations, it accurately reflects the similarities with others, as is shown in figure A.2. Particularly, given a feature $F_x \in \mathbb{R}^{H \times W \times C}$ in the source domain, it is first reshaped into a 2D matrix $Q \in \mathbb{R}^{HW \times C}$, and then multiplied by its transposed $K \in \mathbb{R}^{C \times HW}$. Then each row of the matrix is input to the softmax function, leading to a global attention matrix $A_g \in \mathbb{R}^{HW \times HW}$. Consequently, significant features can be measured according to the entropy H_g of each row in A_g, which is computed as in

$$H_g(i) = -\sum_{j=1}^{HW} A_g(i,j) \log A_g(i,j) \tag{A.4}$$

Here i and j are the indices of the query and key, corresponding to the row and column in A_g. When $H_g(i)$ approaches to 0 , it means that in the i-th row, only a very few key locations are similar with the i-th query. Hence it is assumed that it is distinct enough and is important to be constrained by L_{con}. To select all the significant queries, the rows of A_g are sorted by the entropy H_g in the ascending order, and the smallest N rows are selected as the QS-Attn matrix $A_{QS} \in \mathbb{R}^{N \times HW}$. Note that A_{QS} is fully determined by the features in I_x, and has no relation with $G(I_x)$.

Local Attention

Though non-local attention can obtain the global context, it smooths out the detailed context surrounding the queries. Local attention measures the similarity between a query and its neighboring keys within a constant window of $w \times w$ and stride of 1 , which can capture the spatial interactions in local regions, and reduce the computation cost. Given a reshaped query matrix $Q_l \in \mathbb{R}^{HW \times C}$, it is multiplied by the local key matrix $K_l \in \mathbb{R}^{HW \times w^2 \times C}$ and input to the softmax function, leading to a local attention matrix $A_l \in \mathbb{R}^{HW \times w^2}$. The local entropy H_l is computed in each row as

$$H_l(i) = -\sum_{j=1}^{w^2} A_l(i,j) \log A_l(i,j) \tag{A.5}$$

Here i and j are the indices of the query and key. The smallest N rows in A_l are selected by sorting H_l in the ascending order to form A_{QS}. For the value routing, the selected value matrix $V_{ls} \in \mathbb{R}^{N \times w^2 \times C}$ is obtained by selecting the N indices in local value matrix $V_l \in \mathbb{R}^{HW \times w^2 \times C}$.

Cross domain value routing for contrastive learning

The reduced A_{QS} is used as the attention matrix to route the value features from both source and target domains. Here it is emphasized that A_{QS} captures the global or local relation by comparing the query with keys, and it provides useful high-order descriptions about the shape and texture of I_x. Using it to route features helps to enlarge the receptive field of the selected queries, so that better features, which consider the context of I_x, can be formulated. Moreover, the relation defined by A_{QS} is also required to be kept during the image translation. So A_{QS} is imposed on the features from both I_x and $G(I_x)$, routing the corresponding value to form the anchor, positive and negatives. One positive and $(N-1)$ negative features are located in the real image I_x. N anchors are from the fake image $G(I_x)$. The self-supervised contrastive loss is established as equation A.2, using these features to constrain the translation.

A.2 MoNCE

Perceiving the similarity between images has been a long-standing and fundamental problem underlying various visual generation tasks. Predominant approaches measure the inter-image distance by computing pointwise absolute deviations, which tends to estimate the median of instance distributions and leads to blurs and artifacts in the generated images. MoNCE [Zhan *et al.* (2022*b*)] is a versatile metric that introduces image contrast to learn a calibrated metric for the perception of multifaceted inter-image distances. Unlike vanilla contrast which indiscriminately pushes negative samples from the anchor regardless of their similarity, it proposes to re-weight the pushing force of negative samples adaptively according to their similarity to the anchor, which facilitates the contrastive learning from informative negative samples. Since multiple patch-level contrastive objectives are involved in image distance measurement, optimal transport is introduced in MoNCE to modulate the pushing force of negative samples collaboratively across multiple contrastive objectives.
Given images in two domains, image translation aims to translate images from the input domain to appear like images from the output domain. The datasets for training translation model could be unpaired (i.e., unpaired image translation) and paired (i.e., paired image translation), and different loss terms are entailed for image translations with different dataset setting. GAN loss is usually shared across unpaired and paired image translation to fight against artifacts in translated images, and other loss terms are usually designed specifically to fulfill various objectives, e.g., the cyclic consisteny loss [Zhu *et al.* (2017)] for content preservation or the perceptual loss [Johnson *et al.* (2016)] for assessing human perceptual similarity. However, most metrics are designed by computing the absolute mean error which tends to minimize the average deviation to all possible instances and leads to blur in the generated images.
MoNCE formulates the contrastive loss as a versatile metric in various translation tasks, just by properly selecting the positive and negative pairs. For unpaired image translation, the previously proposed PatchNCE [Park *et al.* (2020)] has validated the effectiveness of contrastive learning for the preservation of content. PatchNCE aims to maximize the mutual information between patches in the same spatial location from the generated

image X and the ground truth Y as

$$\mathcal{L}(X,Y) = -\sum_{i=1}^{N} \log \frac{e^{x_i \cdot y_i/\tau}}{e^{x_i \cdot y_i/\tau} + \sum_{\substack{j=1 \\ j\neq i}}^{N} e^{x_i \cdot y_j/\tau}} \tag{A.6}$$

where $X = [x_1, x_2, \cdots, x_N]$ and $Y = [y_1, y_2, \cdots, y_N]$ are encoded image feature sets, τ is the temperature parameter, N is the number of feature patches. Normally, multilayer features (1st, 4th, 8th, 12th and 16th layers of the encoder) are employed in PatchNCE, which is formulated as $\mathcal{L}^m(X,Y) = \sum_{l=1}^{L} \mathcal{L}(X_l, Y_l)$, where X_l and Y_l denote the corresponding feature sets in l layer of the encoder.

Considering the superior performance of PatchNCE for image translation tasks, contrastive learning can serve as a versatile metric in various image translation tasks. However, the vanilla objective of PatchNCE will repel all negative samples indiscriminately regardless of their similarity to the anchor, which tends to be sub-optimal as the inherent information of negative samples is not equal.

Weighted Contrastive Objective (WeightNCE)

As each negative sample poses different similarity to anchor, the pushing force of each negative sample should be accordingly adjusted for better contrastive learning [Wang *et al.* (2019)]. To adjust the pushing force of a negative samples, a simple yet feasible approach is to adjust its weight in the contrastive objective. According to equation A.6, a higher weight of a negative pair indicates a higher importance in contrastive objective, i.e., enlarged pushing force for this negative pair. Thus, the weighted version of equation A.6 (denoted by WeightNCE) can be formulated as

$$-\sum_{i=1}^{N} \log \frac{e^{x_i \cdot y_i/\tau}}{e^{x_i \cdot y_i/\tau} + Q(N-1)\sum_{\substack{j=1 \\ j\neq i}}^{N} w_{ij} \cdot e^{x_i \cdot y_j/\tau}} \tag{A.7}$$

where Q denotes the weight of negative terms ($Q = 1$ by default) in the denominator, $w_{ij}(j \neq i)$ denotes the weight between sample y_j and anchor x_i and is subjected to $\sum_{\substack{j=1 \\ j\neq i}}^{N} w_{ij} = 1, i \in [1, N]$.

The weighting strategy could essentially boil down to two categories: assigning higher weights to hard negative samples (referred as hard weighting w_{ij}^{+}) and assigning higher weights to easy negative samples (referred as easy weighting w_{ij}^{-}). For the contrastive objective of a single patch, hard weighting weights w_{ij}^{+} and easy weighting weights w_{ij}^{-}

are determined with a positive and negative relation to the similarity between sample y_j and anchor x_i as

$$w_{ij}^{+} = \frac{e^{(x_i \cdot y_j)/\beta}}{\sum_{j=1}^{N} e^{(x_i \cdot y_j)/\beta}} \quad w_{ij}^{-} = \frac{e^{(1-x_i \cdot y_j)/\beta}}{\sum_{j=1}^{N} e^{(1-x_i \cdot y_j)/\beta}} \tag{A.8}$$

β is a temperature parameter. [Zhan *et al.* (2022*b*)] conclude that for unpaired image translation, there is few overlap between the similarity histograms of positive and negative pairs after contrastive learning, which indicates that positive and negative pairs can be easily pushed apart. In this end, the hard weighting strategy may help to boost the performance, as the model can focus on learning from more informative negative samples (hard negative samples) which has been proved to be beneficial for contrastive learning [Robinson *et al.* (2020), Wang *et al.* (2019)].

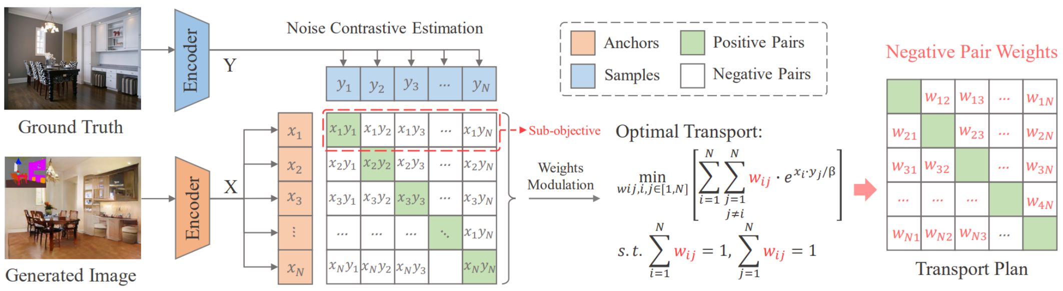
However, for paired image translation, there is severe overlap for the similarity histogram of positive and negative pairs, which indicates many negative samples are hard to be distinguished from the positive samples. In this case, hard weighting may not make for contrastive learning as naively using too hard negative samples may degrade the contribution of moderate ones, yielding worse representation [Jeon *et al.* (2021)]. [Zhan *et al.* (2022*b*)] conjecture that easy weighting may contribute to the contrastive learning in this scenario by assigning lower weights to these hard negative samples which reduces their effects in the contrastive objective. The complete details of validating the above conjecture is beyond the scope of this report and can be found in [Zhan *et al.* (2022*b*)] for those interested.

Modulated Contrastive Objective (MoNCE) As re-weighing strategies are being explored, the total weight associated with a feature (x_i or y_i) is expected to be constant, thus yielding below constraints

$$\sum_{i=1}^{N} w_{ij} = 1, \quad \sum_{j=1}^{N} w_{ij} = 1, \quad i, j \in [1, N] \tag{A.9}$$

Considering the contrastive objective as illustrated in figure A.3, a feature y_j serves as negative sample for multiple sub-objectives. As the total weights associated with y_j is constant (i.e., $\sum_{i=1}^{N} w_{ij} = 1$), there may be conflicts for the weighting strategies of y_j in different sub-objectives, e.g., several sub-objectives all expect a higher weight for y_j while the total weights of y_j is constrained. Therefore, the aim is to modulate the assignment of weights w_{ij} $(i, j \in [1, N], i \neq j)$ across multiple sub-objectives with the constraint of constant total weight.

Targeting to modulate the weights assignment for all negative pairs, a weight modu-



Ground Truth
Generated Image
Encoder
Encoder
Y
X
Noise Contrastive Estimation
y_1 y_2 y_3 ... y_N
x_1 x_2 x_3 ⋮ x_N
Sub-objective
Weights Modulation
Anchors
Positive Pairs
Samples
Negative Pairs
Optimal Transport:
$\min_{w_{ij}, i,j \in [1,N]} \left[\sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} w_{ij} \cdot e^{x_i \cdot y_j / \beta} \right]$
$s.t. \sum_{i=1}^{N} w_{ij} = 1, \sum_{j=1}^{N} w_{ij} = 1$
Negative Pair Weights
Transport Plan

Figure A.3: Framework of MoNCE. There are multiple sub-objectives for the contrastive learning between feature set $X = [x_1, x_2, x_3, \cdots, x_N]$ and $Y = [y_1, y_2, y_3, \cdots, y_N]$. To modulate the weights of negative pairs across multiple sub-objectives, optimal transport with a cost matrix C (defined by $C_{ij} = e^{x_i \cdot y_j / \beta}$ for unpaired translation, $C_{ij} = e^{(1 - x_i \cdot y_j)/\beta}$ for paired translation) is conducted between feature sets X and Y to minimize the total transport cost, yielding an optimal transport plan which serves as the weights of the corresponding negative pairs.

lation goal shared across all contrastive sub-objectives should be determined. The easy weighting strategy is taken as an example to derive the final weight modulation goal. By assigning higher weights to negative pairs with low similarity, the easy weighting strategy for a contrastive sub-objective in equation A.7 is equivalent to reducing the negative term $\sum_{\substack{j=1 \\ j \neq i}}^{N} w_{ij} \cdot e^{x_i \cdot y_j / \tau}$. As the contrastive objectives of all image patches are summed to form the final objective, the shared modulation goal across multiple contrastive objectives can be regarded as reducing the total loss of negatives terms. To derive the expression mathematically, the objective of the modulation goal is formulated as *minimizing* the total loss of negative terms with regards to $w_{ij}, i, j \in [1, N]$ as

$$\min_{w_{ij}, i,j \in [1,N]} \left[\sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} w_{ij} \cdot e^{x_i \cdot y_j / \tau} \right] \tag{A.10}$$

The formation of equation A.10 with constraint in equation A.9 can be regarded as an optimal transport (OT) [Peyré *et al.* (2019)] problem between $[x_1, x_2, \cdots, x_N]$ and $[y_1, y_2, \cdots, y_N]$ with a cost matrix C defined by $C_{ij} = e^{x_i \cdot y_j / \beta}$ for $i \neq j$ and $C_{ij} = \inf$ for $i = j$. Similar to weighting temperature in equation A.8, β in the cost matrix C

serves as a cost temperature that indicates the smoothness of the optimal transport. A smaller β tends to assign higher weights for small cost entries C_{ij} and a large β tends to assign equal weights for all cost entries. Detailed parameter study of β can be found in the experiment part of [Zhan *et al.* (2022*b*)].

The optimal transport aims to retrieve a transport plan T which minimizes the total transport cost as formulated below.

$$\min_{T} \langle C, T \rangle, \quad \text{s.t. } (T\vec{1}) = 1, \left(T^{\top}\vec{1}\right) = 1 \tag{A.11}$$

where $\langle C, T \rangle$ denotes the inner product of C and T. Thus, solving the transport plan T is equivalent to solve the weight parameters as $w_{ij} = T_{ij}$. The Sinkhorn algorithm [Cuturi (2013)] can be applied to equation A.11 for approximating optimal transport solution, yielding the desired optimal transport plan T. With the derived transport plan matrix T as the weights of negative pairs, the modulated objective for easy weighting strategy is accordingly determined. For hard weighting strategy, the modulated objective can be derived similarly, just redefining the cost matrix C in equation A.11 as $C_{ij} = e^{(1-x_i y_j)/\beta}$ for $i \neq j$ and $C_{ij} = \inf$ for $i = j$.

Preliminary Results We paired the MoNCE loss along with the spatial correlation loss as described in section 3.4. We use the authors' code [Zhan *et al.* (2022*a*)] with the only change being the added spatial correlation loss. Since we are considering unpaired image translation, we use the "CUT_MoNCE" code. The parameters are the default ones as present in [Zhan *et al.* (2022*a*)]. The weight used for the spatial correlation loss is 10. The results are shown in figures A.4 - A.10.

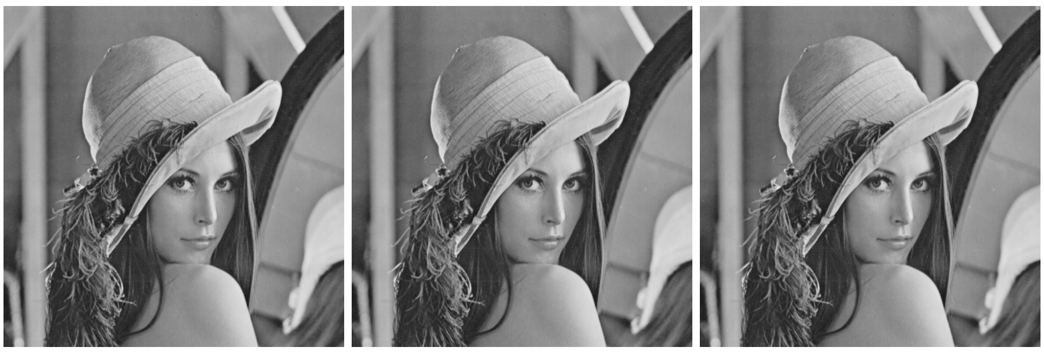


Figure A.4: LR, SR, and HR Image

These are some of the best results we have had so far. In fact, the defects in figure

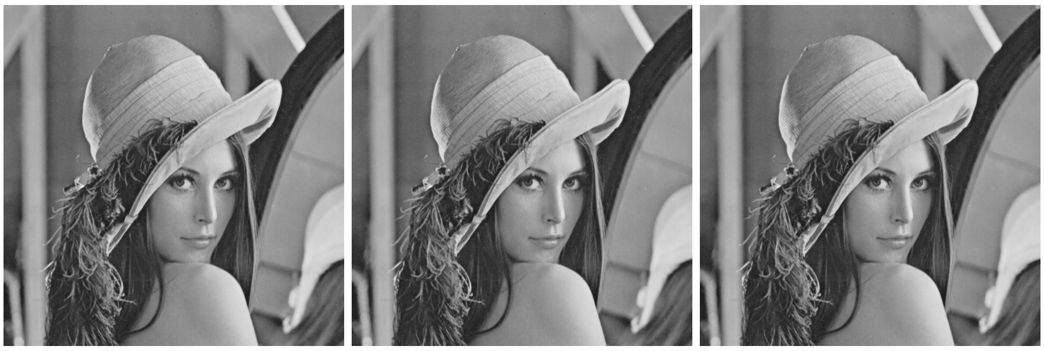


Figure A.5: LR, SR, and HR Image

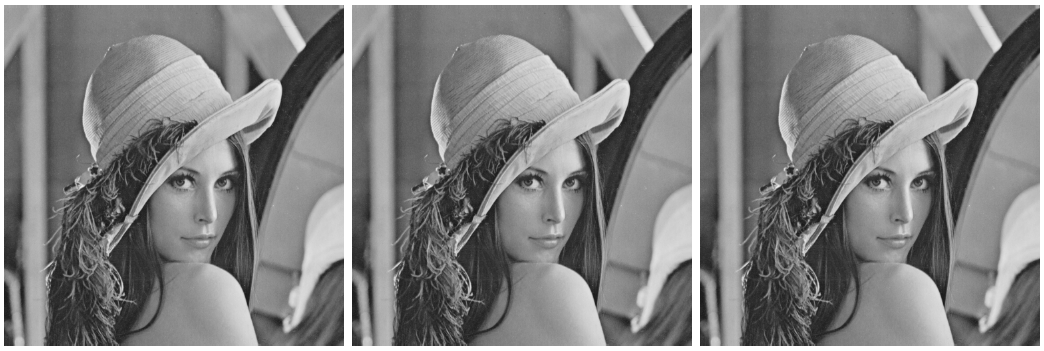


Figure A.6: LR, SR, and HR Image

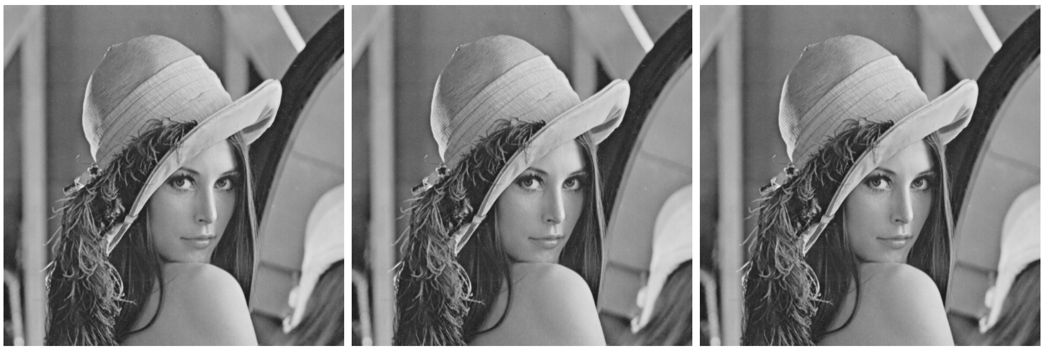


Figure A.7: LR, SR, and HR Image

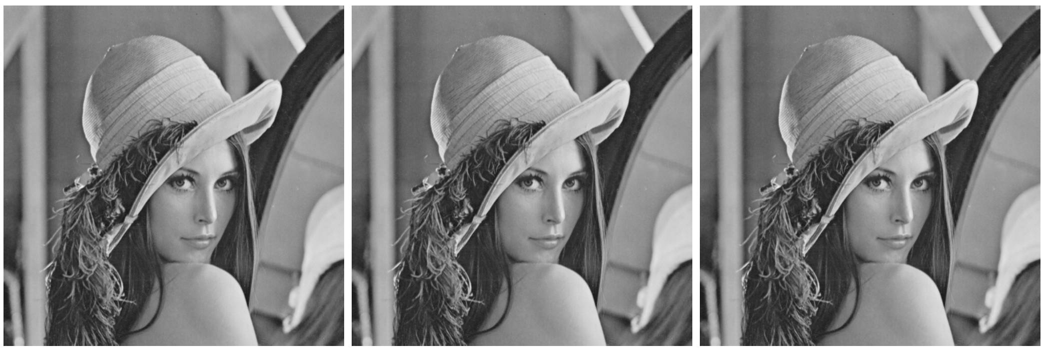


Figure A.8: LR, SR, and HR Image

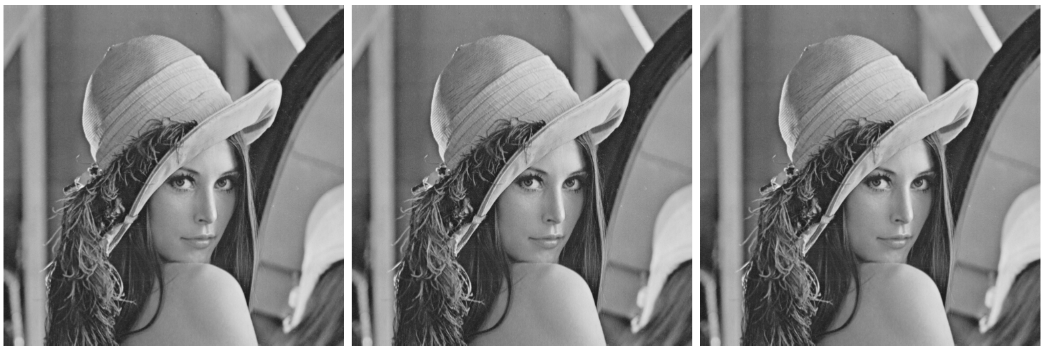


Figure A.9: LR, SR, and HR Image

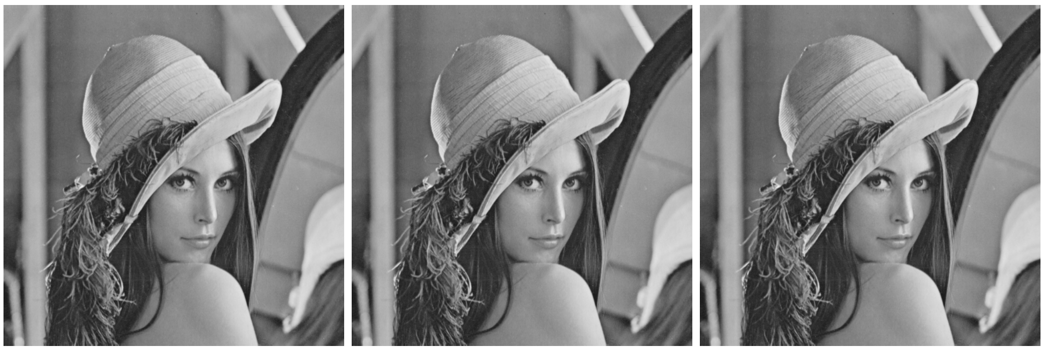


Figure A.10: LR, SR, and HR Image

A.10 looks a lot more natural than what we got using only the spatial correlation loss. However, the reason this is included in the appendix and not the main text is due to the fact that adding this MoNCE loss to the spatial correlation loss makes the training relatively unstable. A few examples of this can be seen in figures A.11 and A.12, where the results on repeating the exact same training again led to a lot worse results. Although the structure of defects is mostly preserved, the background looks really unnatural.

More work is needed to stabilise the training of MoNCE. Another idea is to try and use

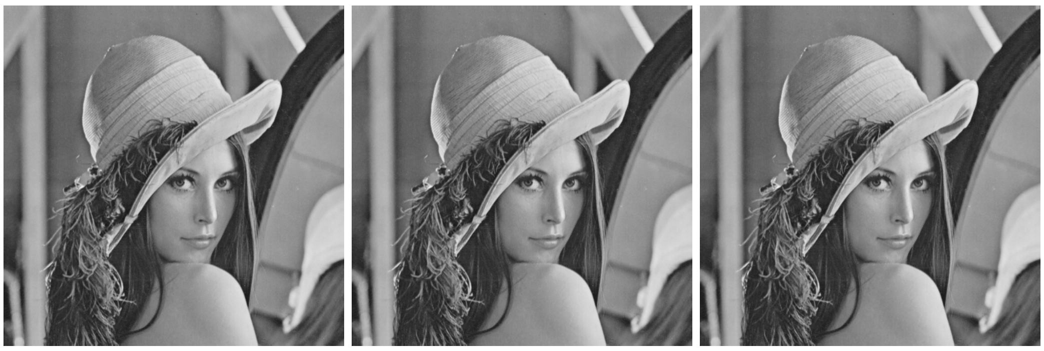


Figure A.11: LR, SR, and HR Image

MoNCE to stabilise the training of the learned self-similarity loss. As of now, MoNCE

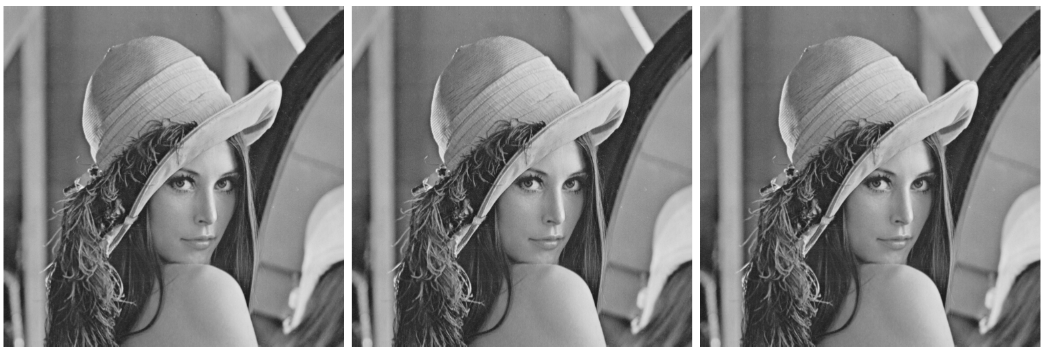


Figure A.12: LR, SR, and HR Image

is a very promising addition but nothing conclusive can be drawn from it yet.

Appendix B

New Set of Images

We obtained a new set of images from KLA on June 3, 2022. There are two folders with names *kla_AEI_2x_2FA_p5m* and *kla_CuCMP_2x_4FA_1p75um*. Each folder has two sub folders with names *512PD* and *1024PD*. The former is a $2\times$ SR dataset whereas the latter is a $4\times$ dataset. The HR images in the folders *1024PD* are of the size 976×976 and the corresponding LR images are of the size 244×244. The HR images in the folders *512PD* are of the size 464×464 and the corresponding LR images are of the size 232×232. The *kla_AEI_2x_2FA_p5m* folder contains 18 LR and HR images for each scale factor. The *kla_CuCMP_2x_4FA_1p75um* folder contains 42 LR and HR images for each scale factor. Unlike the dataset we had been working with so far, there is no obvious domain shift between the LR and HR images in the new dataset for both the folder.

B.1 Visualization

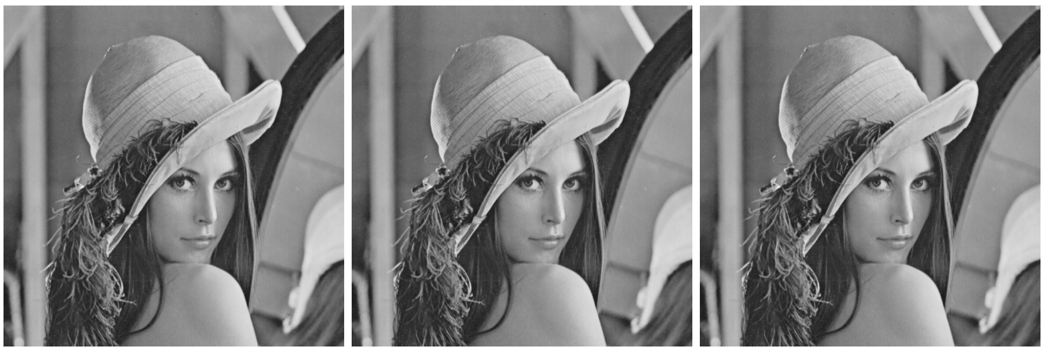


Figure B.1: *1024PD* LR images from the folder *kla_AEI_2x_2FA_p5m*

A few *1024PD* HR images and the corresponding LR images from the folder *kla_AEI_2x_2FA_p5m* can be seen in figures B.2 and B.1 respectively. Similarly, *1024PD* HR and LR images from the folder *kla_CuCMP_2x_4FA_1p75um* can be seen in figures B.4 and B.3 respectively. On a first inspection, although there is no obvious domain shift between the LR and HR images, there are obviously differing noise levels

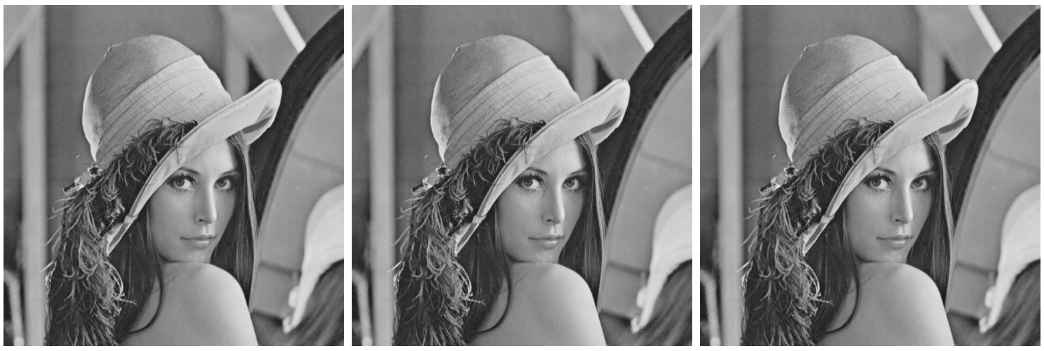


Figure B.2: *1024PD* HR images from the folder *kla_AEI_2x_2FA_p5m*

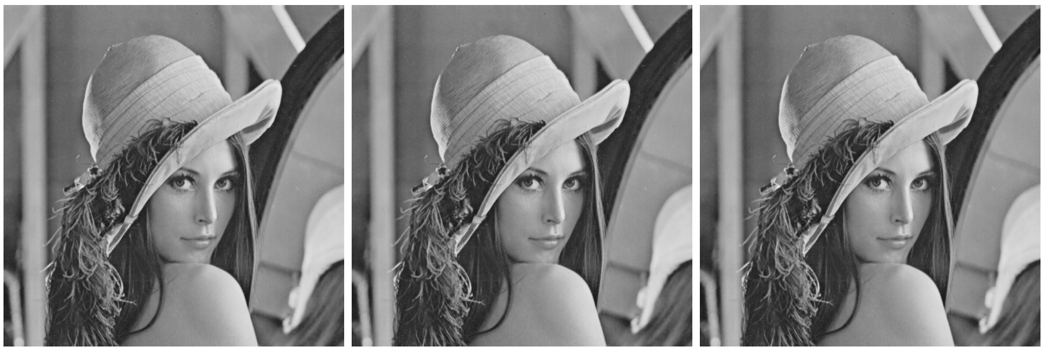


Figure B.3: *1024PD* LR images from the folder *kla_CuCMP_2x_4FA_1p75um*

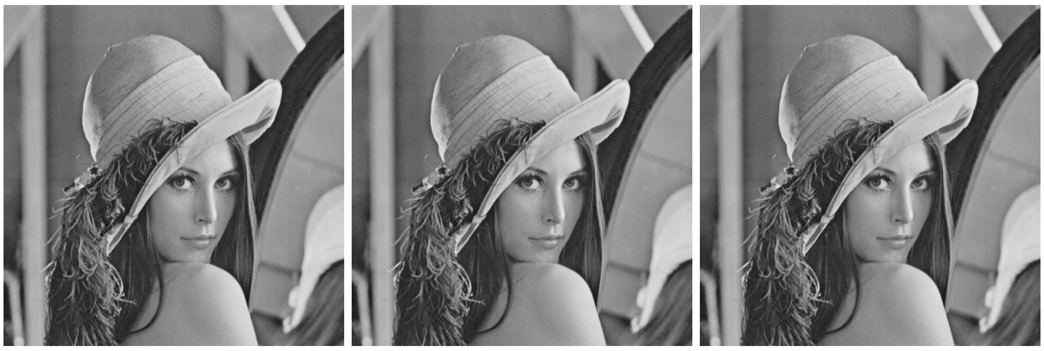


Figure B.4: *1024PD* HR images from the folder *kla_CuCMP_2x_4FA_1p75um*

and sharpness between them. Also, on cycling through a few images, the data looks to be aligned. Although, this will need to be properly verified. It should be noted that the images are not to scale. Given that the HR images are 4× the LR images in both dimensions, it is impossible to represent them to scale

B.2 Preliminary Results

We show results on images from the *kla_CuCMP_2x_4FA_1p75um* dataset. We also restrict to the 4× SR case, as it is easier to spot the difference between a bicubically upsampled image and the HR image at higher SR scales. This will let us know how well is our model doing compared to simply upscaling. Another thing to be noted that, we do not utilise all of the images in the folder under consideration. The reason being one type of image (see the middle images in figures B.4 and B.3) look very different from the rest of the images in the dataset, and we suspect they might throw off an unsupervised domain transfer network. These images have the word *CMPTop* as a part of their names. Excluding these images, we are left with 28 images. We use 21 of these for training and 7 for validation.
As we had been working with image translation methods, we upscale the LR images to be of the same size as the HR images, and train an unsupervised SR network between the LR and HR images. We use the same experiment setting as mentioned in the experiments part of A.2. A few results can be seen in figures B.5 - B.7. We follow the usual format for results: LR image upscaled on the left, the output image in the center, and the ground truth or reference image on the right.

It is clear that the SR image is not the best approximation of the ground truth image. However, is it doing any better than simple upscaling? It does look like that to the naked eyes, but it is hard to quantify. Hence, we use the gradient maps of the LR upscaled, SR, and HR images to explain the observations in a more quantifiable manner. The gradient maps are shown in figures B.8 - B.10. Please zoom into the PDF to view the number on the bar.

Figure B.5: LR, SR, and HR Image

Figure B.6: LR, SR, and HR Image

Figure B.7: LR, SR, and HR Image

Figure B.8: Gradient maps of LR, SR, and HR Image

Figure B.9: Gradient maps of LR, SR, and HR Image

Figure B.10: Gradient maps of LR, SR, and HR Image

We now have something quantifiable. The gradient maps of the ground truth images have thin but strong edges. This is not the case in the input images. The gradient maps of the output images are somewhere in between: better than the input images in general, but not as good as the HR ones. Also, the input image's gradient map has a lot of artefacts which is not present in the gradient map of the HR image, and significantly reduced in the gradient map of the output image.

B.3 Moving Forward

We feel like the performance of the model on part of the new data it was trained and tested on is not adequate. It is better than a plain upscaling, sure. But, the output images have fairly lesser sharpness than the ground-truth images. We feel that given the availability of paired data, an unsupervised image to image translation method might not be the best way to approach this data. There is little to no obvious shift between the LR and HR images. The only difference are the lack of details and difference in noise levels due to data being captured at a different sampling rate. This is similar to other

SR problems which the lab has experience with and for which a well trained SR model sufficed without the need for much hackery.

We feel that for this dataset too, a paired SR model is the way to go. This will also allow the training of the entire dataset together without removal of images which appeared to be from a different domain. We say this because given the unsupervised nature of the training, having images from two domains along with the spatial correlation loss and discriminator loss can confuse the model about what to do with the style of the image. For the discriminator, both styles will be equally valid as it would have seen both styles in the training data. For the spatial correlation too, a change in color while keeping the structure does not increase the loss, hence both are equally valid solutions with regards to this loss too.

A paired SR training additionally constrained with a pixelwise loss like the L1 loss or perceptual loss can prevent the aforementioned issue. The discriminator then has to differentiate between LR and SR samples from the same image, forcing it to differentiate based on blur and noise levels only, and not based on color or domain. The presence of a pixelwise loss also prevents the generator from needlessly flipping the style of the image while keeping the structure intact. Given the presence of a pixelwise loss, we do not see the necessity for the spatial correlation loss; however, we see no reason as to why it should hurt performance.

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. **Cuturi, M.** (2013). Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, **26**.

3. **Du, W.**, **H. Chen**, and **H. Yang**, Learning invariant representation for unsupervised image restoration. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

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

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

6. **Hu, X.**, **X. Zhou**, **Q. Huang**, **Z. Shi**, **L. Sun**, and **Q. Li**, Qs-attn: Query-selected attention for contrastive learning in i2i translation. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

7. **Isola, P.**, **J.-Y. Zhu**, **T. Zhou**, and **A. A. Efros**, Image-to-image translation with conditional adversarial networks. *In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. 2017.

8. **Jeon, S.**, **D. Min**, **S. Kim**, and **K. Sohn**, Mining better samples for contrastive learning of temporal correspondence. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

9. **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.

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

11. **Karras, T.**, **S. Laine**, and **T. Aila**, A style-based generator architecture for generative adversarial networks. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.

12. **Liu, M.-Y.**, **T. Breuel**, and **J. Kautz** (2017). Unsupervised image-to-image translation networks. *Advances in neural information processing systems*, **30**.

13. **Lu, L.**, **W. Li**, **X. Tao**, **J. Lu**, and **J. Jia** (2021*a*). Dvlab-research/masa-sr: Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution (cvpr2021). URL `https://github.com/dvlab-research/MASA-SR/`.

14. **Lu, L.**, **W. Li**, **X. Tao**, **J. Lu**, and **J. Jia**, Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021*b*.

15. **Mechrez, R.**, **I. Talmi**, and **L. Zelnik-Manor**, The contextual loss for image transformation with non-aligned data. *In Proceedings of the European conference on computer vision (ECCV)*. 2018.

16. **Mirza, M.** and **S. Osindero** (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

17. **Park, T.**, **A. A. Efros**, **R. Zhang**, and **J.-Y. Zhu**, Contrastive learning for unpaired image-to-image translation. *In European Conference on Computer Vision*. Springer, 2020.

18. **Pathak, D.**, **P. Krahenbuhl**, **J. Donahue**, **T. Darrell**, and **A. A. Efros**, Context encoders: Feature learning by inpainting. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

19. **Peyré, G.**, **M. Cuturi**, *et al.* (2019). Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, **11**(5-6), 355–607.

20. **Robinson, J.**, **C.-Y. Chuang**, **S. Sra**, and **S. Jegelka** (2020). Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*.

21. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-net: Convolutional networks for biomedical image segmentation. *In International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

22. **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.

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

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

25. **Van den Oord, A.**, **Y. Li**, and **O. Vinyals** (2018). Representation learning with contrastive predictive coding. *arXiv e-prints*, arXiv–1807.

26. **Wang, X.** and **A. Gupta**, Generative image modeling using style and structure adversarial networks. *In European conference on computer vision*. Springer, 2016.

27. **Wang, X.**, **X. Han**, **W. Huang**, **D. Dong**, and **M. R. Scott**, Multi-similarity loss with general pair weighting for deep metric learning. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

28. **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.

29. **Yang, F.**, **H. Yang**, **J. Fu**, **H. Lu**, and **B. Guo**, Learning texture transformer network for image super-resolution. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020.

30. **Zhan, F.**, **J. Zhang**, **Y. Yu**, **R. Wu**, and **S. Lu** (2022*a*). Fnzhan/monce: Modulated contrast for versatile image synthesis [cvpr 2022]. URL `https://github.com/fnzhan/MoNCE`.

31. **Zhan, F.**, **J. Zhang**, **Y. Yu**, **R. Wu**, and **S. Lu**, Modulated contrast for versatile image synthesis. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022*b*.

32. **Zhang, Z.**, **Z. Wang**, **Z. Lin**, and **H. Qi**, Image super-resolution by neural texture transfer. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

33. **Zheng, C.**, **T.-J. Cham**, and **J. Cai**, The spatially-correlative loss for various image translation tasks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2021.

34. **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.