Noisy Visual Training:

Learning Robust Low-level Image Prior

A PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

by

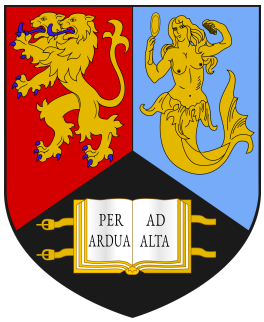
HAO CHEN

h.chen.12@bham.ac.uk

Under the guidance of

PROF. JIANBO JIAO

j.jiao@bham.ac.uk



School of Computer Science

University of Birmingham

Birmingham, UK

DEDICATION

Dedicated this thesis to the beloved Prof. Gilbert Strang, who interested me in the nature of mathematics.

Contents

1 Abstract

Recently, self-supervision has been applied to low-level computer vision tasks as a powerful tool that no longer requires image pairs. However, directly applying self-supervision on deep estimators would lead to poor image fidelity due to lack of effective supervision. In this work, we study self-supervised deep denoisers and develop a new pre-training scheme with several plug-and-play modules to directly reconstruct invariant representation under unknown noise domains. Experiments on synthetic and real noise removal tasks shows the proposed scheme can improve current state-of-the-art self-supervised methods by a large margin and can achieves comparable performance with supervised methods. Code has been released in GitLab.

Keywords: Invariant Representation Learning, Pre-training Scheme, Disentangled Learning, Image Denoising, Self-supervised Learning, Mutual Information, Image Post-processing

2 Background Introduction

Deep Learning (DL) is our attempt to mimic the behaviour of the human brain, allowing it to learn from large amounts of data. We have witnessed great progress in Deep Learning in the last decades, albeit still far from matching our ability.

Currently, Deep Learning (DL) has been incorporated into many critical real-world systems. For example, DL can be used to create Computer Vision systems that can produce photo-realistic avatars or Natural Language Processing systems that can talk with human. DL did a great job in boosting the development of Artificial Intelligence in many fields, both in industry and academia.

Despite the rhetoric about DL applications remains almost messianic, they are rated low on the reliability and generalizability scale. Most deep models work around a simple task and can work well in a few fixed scenarios with lots of data available [30]. The critical nature of these system makes them vulnerable with data inconsistencies and domain-gap, which results in high test error rates. For example, poor illumination, strong reflections or other adversarial weather conditions [4] are challenging for scene understanding ability of existing autonomous vehicles systems.

Some in-depth studies have revealed the failure of Deep Learning may also due to the existence of spurious statistical shortcuts in data. *Ribeiro et el.* [25] shows that deep classifier with high precision may rely on the background to distinguish animal species (Figure 1). *Marcus et el.* [20] demonstrate that question-answering agents tend to reply with the high-frequency words appeared, e.g. names or nouns. This spurious features is non-invariant and can change arbitrarily across domains that slight data shift can have a significant impact on deep agents [15, 18].

Thus, we start to think why deep learning agents fails, for example: why the deep self-driving agent which has seen millions of training examples may still perform out of expectation [30], while human who only have practised driving for two weeks can drive in extreme weather conditions which they never encountered on their lifetime?

We attribute this unique human ability to the prior knowledge that human acquire in the process of evolution and individual growth. Human brain is the product of billions of years of evolution, and it is a bunch of primitive constructs that you can use as the basic ingredients of your unconscious mind. This structure allows us to unconsciously grasp all kinds of general knowledge in our daily life. This knowledge prior facilitates us to understand the world and extract meaningful feature fast when we facing new tasks even with only a few examples available.

Many researchers try to mimic the same kinds of mechanisms as human used, like contrastive learning [16] and meta learning [13], etc. These representation learning methods served as a kind of pre-training schemes, equipping the model with prior knowledge acquired from the given image itself, without labels as humans do. Nowadays, pre-training has been extensively applied to and marked

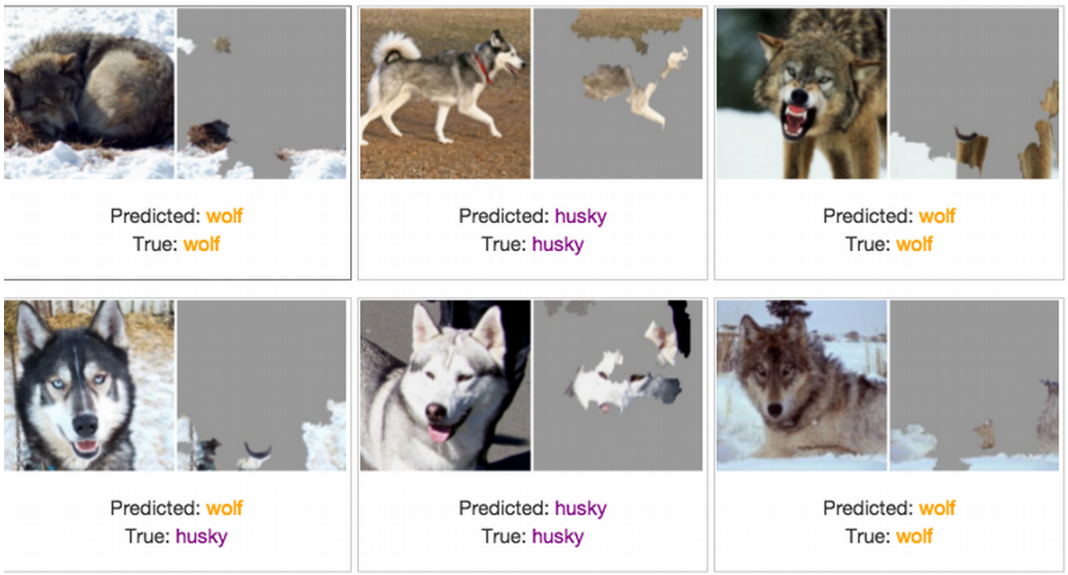


Figure 1: Each image pairs are combine with input image (left) and decision basis (right). The high precision husky/wolf classifier are making decision based on detecting the snow rather than the animals. [25]

as an essential element in high-level computer vision tasks. Most of the current representation learning methods is designed to group unlabeled image in an unsupervised way that forcing them to understand images in high-level semantically. Subsequent models can use these high-level features to understand the semantic information image and well classify and localize objects in the images.

However, rather that utilizing high semantic image features, low-level tasks use pixel-level features to understand the distribution of pixel intensities, thus requiring different pre-training schemes. Unlike high-level semantic pre-training, few attempts have been made to investigate how pixel-level pre-training act in image processing tasks.

In this paper, we mainly research on one-shot *denoising* tasks, where available training information is extremely inadequate and often consider as a ill-posed problem that result in model overfitting and under-train (in Sec. 3.5). We make a step further in this direction and propose a full pixel-level pre-training pipeline called *Noisy Visual Training* (NVT), which is designed to provide models with prior knowledge for one-shot *denoising* tasks and can be generalized to other low-level down-streams. The details of NVT in illustrated in Sec. 7, where we explain our approach from a theoretical perspective.

To comprehensively diagnose the influence of pre-training, we use the residual image for visualizing the noise/image separating ability of models. Extensively quantitative and qualitative experiments are done on *denoising* tasks, showing the NVT can well support single image-based *denoising*. All experiments are summarized in Sec. 8.

We also provide tables for terminology and mathematical notation used in there work in Sec. 5 and Sec. 6 for rapid reference.

3 Related Work

3.1 Pre-training and Deep Prior

Pre-training is a common approach in the field of deep learning and is widely used in computer vision, natural language processing and speech [9]. For example, the supervised ImageNet pre-trained backbones are widely used for object detection and segmentation tasks with the help of transfer learning and fine-tuning.

One of the most notable benefit of pre-training is its usability and therefore gaining importance in the field of deep learning literature. Generally, pre-training allows models to equip with prior knowledge, or Deep Prior. Model with Deep Prior can learn and be optimized quickly without requiring much data comparing to the one starting from scratch.

However, pre-training does not always lead to better performance and may not be adaptable. For example, ImageNet, which is a dataset based on classification problem, is much easier benchmark than most of the visual tasks such as the object detection problem, thus requiring different pre-training schemes for more challenging tasks.

Recently, there is some new research directions on the notion of Unsupervised Representation Learning (URL) and has shown promising results in the Deep Learning literature. Generally, URL acquires knowledge and encourages exploratory behaviour by shifting the focus from targeting effects to understanding causes without task-specified tagged data. One way is the Contrastive Learning, which is a principle of contrasting samples from multiple augmentations against the same input image to learn attributes that are *common* between different image transformations. The Contrastive Learning will be illustrated in Section 3.2. The idea of Invariant Feature Learning is similar to Contrastive Learning but with more strict constraint of the learned representation regarding the multiple augmentations.

The other promising is the Disentangled Representations Learning which works toward learning of separating high-level abstractions into chunks that representing each interpretable features alone. This follows the ambitious objective of disentangling the underlying causal factors to control the output in the desired way by directly changing the representations (usually unexplainable), which will be discussed in Section 3.2.2.

There are also some noteworthy comparisons between the random initialization and pre-trained models. Research by *He et al.* [11] shows that, pre-training has an advantage over random initialization at first, but as the iterations increase, the performance gets closer and closer, eventually reaching the similar result.

3.2 Unsupervised Learning and Self-supervised Learning

One of the main challenges for AI remains learning in the wild, at which humans are much better than machines. The reason why human outperforms DL agents in unseen tasks can be attributes to the prior knowledge we have, i.e., our ingenuity, experiences and insights of the world and so forth.

In the past 20 years, scientists spend lots of effort putting engineered knowledge into specific prior knowledge designed for agents. But it is laborious and sensitive to domain shift. Thus, research proposed Unsupervised Learning and Self-supervised Learning aiming to learn consciousness prior from unlabelled data like human did in their daily life without any human intervention.

The notations of *Unsupervised Learning* (UL) and *Self-supervised Learning* (SSL) are similar literally but in the different end of the spectrum. Generally, they both do not involving the any human annotation and since the model do not have correct labels, they end up using different training monitoring scheme.

Unsupervised Learning helps model to find the underlying high-level patterns or structures within the given data that can [6]. For example, language model with huge amount of param-

eters often requires unsupervised learning for pre-training. These trained concepts brought to consciousness often correspond to understanding words or short phrases and the thought itself can be transformed into a brief linguistic expression, like a sentence. Just like our intuitive or from experience to understand the texts.

Self-supervised learning [14] instead of finding semantically high-level patterns for clustering, it attempts to use the input itself as the supervision signal and trying to solve tasks that are traditionally dominated by supervised learning. In general scenario of self-supervised learning, each image instance can be treated as a classification group that independent with each other. Basically, contrastive learning and disentangled representation learning are terms of self-supervised learning.

This work mainly focus on *self-supervised learning*.

3.2.1 Contrastive Learning

One would expect that given similar inputs, the deep representation should be similar. That is the goal of contrastive learning [5, 15], where we explicitly train our model by requiring that the representation deviation of samples from same distribution to be small and, conversely, dissimilar those samples belonging to different distributions. In one word, a contrastive learning model encodes semantically similar data with similar encoding state and makes the coding results pulled against each other for different group of data.

In contrastive learning, we want to consider the perturbations in the input as noise and would like the model to be robust against it. The variation of possible perturbations is treated as a distribution among contrastive learning methods. For example, given a image x and a set of augmentation $\mathcal{T}$, the augmented view $\tilde{x} \in \mathcal{T}(x)$ of the original x is considered as similar since they are essentially different versions of the same image. Thus, all augmented views $\tilde{x} \in \mathcal{T}(x)$ are consider a distribution that the encoding of all the views should be close in representation field.

Generally, contrastive learning is a self-supervised representation learning method without the need of access to labels. This natural assets allow contrastive learning to enjoy the massive number of unlabled data and makes it a powerful task-independent Deep Prior with a general understanding of the data.

3.2.2 Disentangled Representation Learning

Representations Learning is a critical building block of deep learning scenario. Traditional representation learning has been conducted with a focus task performance rather than the distribution of features in representation domain. In this way, these per-domain models tend to produce sparse and low cognitive features that are representative enough for the given task but not interpretable for human to understand and modify.

To fill the blank, researchers incorporate Disentangled Representation Learning (DRL) to build up independently controllable latent space in terms of casual factors in images, such as face expressions and hair style in generated head avatars. Simply put, DRL helps model learns a representation space that consists of multiple linear sub-spaces that each of which controls one aspect variation of the world.

In order for doing this, DRL explicitly include domain exchange and restore to preclude each representation variable. For example, *Du et al.* [7] utilize DRL to drive image back and forward between clean domain and noisy domain that the noise inside the image can be represented as a adjustable variable that used for image denoising.

@articlehjelm2018learning, title=Learning deep representations by mutual information estimation and maximization, author=Hjelm, R Devon and Fedorov, Alex and Lavoie-Marchildon, Samuel and Grewal, Karan and Bachman, Phil and Trischler, Adam and Bengio, Yoshua, journal=arXiv preprint arXiv:1808.06670, year=2018

3.3 Mutual Information

In the view of probability theory, the mutual information (MI) is a metric to measure interdependence between two variables, or in intuitively, how much information of one variable we can get from observing another one. The concept of MI is closely related to the concept of information entropy, which is a fundamental concept in information theory that quantifies the amount of information contained in a random variable. Recent works have focused on maximizing MI as a mean to measure similarity and perform representation learning, such as to maximize the mutual information between given data and the learned representation. Given two random variables X and Y from unknown distribution, the mutual information between them can be calculated by the following equation:

$$I(X,Y) = \int_{\mathcal{X}} \int_{\mathcal{Y}} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) dxdy$$

In this way of calculation, the greater the MI, the more information X can provide about Z, and vice versa. Although MI is a theoretically elegant quantity, the calculation of MI is usually exists in high-dimensional space. Especially for the representation in deep learning, it is very difficult to accurately compute both the edge distribution and the joint distribution at the same time in many cases that makes calculation of notoriously computational hard. Some recent estimators use deep models to approximate the joint and edge distribution, then can be used in calculating MI. Auch as estimate Kullback-Leibler divergence and Jensen-Shannon divergence [2, 12, 26] .

MI is also widely use in the field of disentanglement. To accomplish this task, the MI is minimized between one and another given latent variables.

3.4 Image Post-processing

Image post-processing is routine operation step in image capturing process, involving white balance, demosaicing, denoising and tone mapping, etc [24]. Similar to any data processing pipeline, images must be prepared and be further processed for human or machine interpretation. The incorporation of image post-processing in general CMOS camera convert the captured RAW image into human visual system (RGB domain) and can largely improve the image quality.

Deep learning-based methods have also enjoy the benefits of image post-processing, such as in medical image and super resolution [8]. In general, the include of post-processing into deep image processing aiming to improve image quality in terms of qualitative or quantitative metrics by adjusting image pixel distributions.

3.4.1 Unsharp Masking-based Methods

The Unsharpen Masking (USM) is a originally implemented in darkroom photography but is now commonly used in digital image processing software. In digital image processing, USM can be regarded as a spatial sharpen filtering consisting of two steps. The first step is to use a low-pass filtering layer, which is usually a fixed Laplacian of Gaussian (LoG) kernel L, over the image X to get the blurred or unsharp negative image B.

$$B = L \otimes X$$

where *otimes* is the convolution operation. In the second step, we can get the detail layer through a difference of $D = X - B$, and fuse it back to the original image to strengthen the details with a weight λ:

$$\tilde{X} = X + \lambda * (X - L \otimes X)$$

In deep learning view, the USM can be seen as a skip connection with one fixed weight convolution layer. Inspired by which, many USM-based approaches are grouped in deep learning research [3], proposing to change the method of calculating detail layer D or use a differentiable USM to act as a prior for deep learning, especially in pixel-wise classification schemes.

3.5 Image Denoising

Denoising is an everlasting topic in the science of Digital Image Processing. Good image denoising increases the visual quality and better reflects the information carried by the collected image which can positively affect subsequent image analysis and processing tasks. But the treatment of real noise in images is domain-dependent and non-trivial.

Plenty of denoising methods have been developed, originating from probability theory, statistical models, linear and nonlinear filtering etc. Many so-called classic methods more or less rely on the explicit or implicit assumption of prior noise distribution, which is usually unknown in a real-world scenario.

To mitigate this problem, a series of learning-based denoising methods without the requirement of knowing any noise model is proposed. Traditionally,the models are trained in supervision fashion with pairs of corrupted images and clean images, so that the network learns to remove the corruption. Despite their great ability to learning a mapping from corresponding noisy image to clean images from toy datasets, the performance of their real-world applications heavily depends on a large number of real noisy-clean image pairs for training. Unfortunately, collecting such an aligned pairwise dataset is extremely challenging and expensive in real-world photography and prone to dataset biases as the conditions for obtaining paired images are limited and harsh.

Some works tend to relieve the above cumbersome by synthesising noise or adjusting camera gain to obtain images with different noise levels. However, the real raw image noise is a complex combination of many noise sources and varies greatly among different fields. Existing methods are laborious and unable to target and model all noise sources accurately, the performance of model trained with synthetic pairs degrade greatl due to the domain gap between synthetic and real noise.

A promising path for relieving the above cumbersome is the recently emerged *Self-supervised* denoising, where the noisy image is used both as input and ground truth to train the network. It is much based on the assumption that the noise is i.i.d and independent of the image signal. Thus, statistics tell us if the model is expressive enough and if enough data is given, it is possible that we can recover the image signal from noise input.

As self-supervision only assume the availability of one single unorganized image, it is more applicable for real-world denoising problems across uncountable subfields in science and engineering.

Recent papers have shown competitive performance to supervised learning, such as Deep Image Prior [28], Self2Self [23] and Noisier2Noise [22].

3.5.1 Image Noise Modeling

Noise is an integral and significant part of image capturing and hot to model it accurately is a long-standing problem in computer vision. Well-established parametric noise models are critical in many computer vision applications. For instance, noise model estimation can incorporated as a noise prior for training models to be robust in the presence of realistic noise [19] . Noise modeling is also necessary in image post-processing for removing fixed pattern effects from CMOS sensors [1].

Early work has shown that noise in wild can be expressed as a Noise Flow that combine with a series of noise such as signal-independent shot noise, additive heteroscedastic Gaussian and quantization noise [21].

So recent work on using Noise Flow [1, 19] for synthesizing noisy data for pre-training denoiser has yielded significantly improvement upon the supervised baseline denoising approaches. The results indicate the potential of using Noise Flow in pre-training self-supervised denoisers.

3.5.2 Deep Denoisiers

Deep denoisers have boosted recent development on image denoising with the utilization of deep learning. Using unsupervised architecture, it is currently possible to take advantage of the large number of unpaired noisy images for training. However, a versatile existing methods show poor performance under practical setting due to their depressed discernment of noise and image texture.

Lacking of ground truth makes the training procedure hard to converge. Many works focusing on how to blind some information of the original image from the network to prevent it from learning a identity mapping. In this paper, we largely inspired by the following papers:

3.5.3 Deep Image Prior

The important feature of Deep Image Prior [28] or DIP is that it only use a completely randomized deep convolutional network to learn to replicate a corrupted image (e.g., an image with noise added). This network never look at any other images, nor does it look at normal images for training from start to finish. But the final result is still quite satisfying that it shows the network will automatically learn how to reconstruct the image by itself without ant training.

Deep convolutional neural networks (CNN) has ability to represent local regularity and self-similarity that they have an innate ability to learn the "uncorrupted, natural part" of input noisy y before it learns the "corrupted part". In other words, CNN understands how a natural image should look like, which makes CNN structure has high noise impedance and is a very good prior for image processing tasks.

DIP use this self-similarity of images represented by a randomized CNN and achieve high noise reduction performance, defining a network structure-base prior. Presenting a task of minimizing the distance and penalty over prediction $\hat{x} = f_\theta(y)$ given image y:

$$\arg\max_{\hat{x}} \mathbf{E}(\hat{x}; y) + R(\hat{x})$$

Conventional learning approaches minimizing this loss function by iteratively go through the data set and update the weight θ of the network until reaching a compromising point that achieves the minimum loss.

However, DIP using a distinct path that using a embedding z to generate $\hat{x} = g_\theta(z)$ by the random initialized model instead of input the noisy y. In this process, the weight θ is fixed with the z in parameter space varies. In one word, other models vary the weights optimize result, on the contrary, DIP minimizing the given loss function by updating the input z.

The parameterization is somehow not so intuitively that one may wonder this will result in the same noisy image y which serves as the target. By this way, the model is basically just over fitting the given noisy image.

Each curve in (Figure 3) represents loss trend in training. The different learning speed of natural image and noise image shows the model is more sensitive to the natural structure inside the image, while later over-fitting the input noise. By using this idea, DIP apply the cutoff mechanism that stop training at specific epoch that prevent from over-fitting noise inside the given image.

With this prior, DIP can do more than the denoising task, but also Super-Resolution, Image Impainting and Enhancement without requiring training set.

3.5.4 Self2Self with Dropout

Self2Self [23] or S2S is a self-supervised denoiser which only rely on the input noisy image and would only have access to is the input image itself. This promising path relax the requirement of collecting a large number of clean/noisy image pairs, which is difficult and expensive. And S2S goes a step further by lifting the requirement to use the training set.

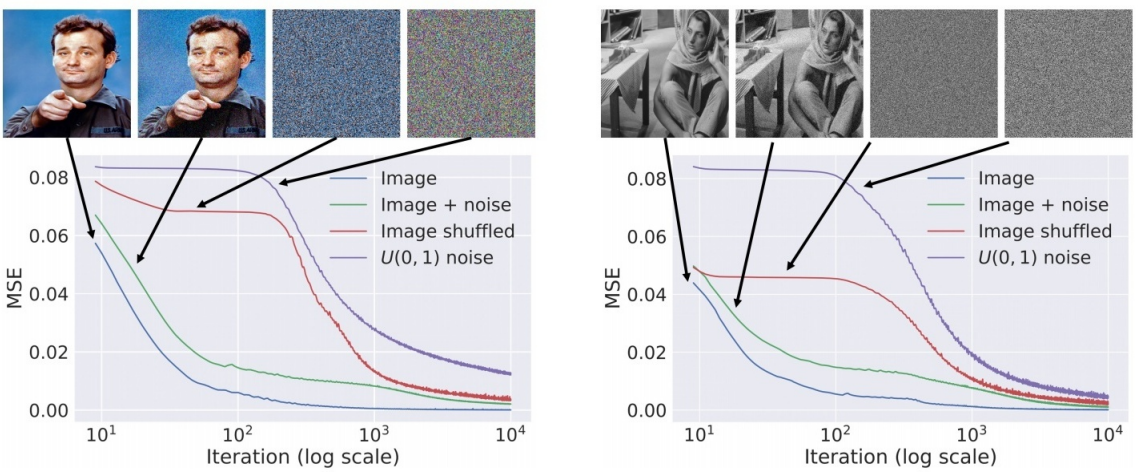


Figure 2: Experiments in [28] show the learning curves for reconstruction tasks with additive i.i.d noise. Natural images result in a faster converge speed, while the structure of random noise is learned in a much later iterations.

The S2S designs a dropout-kind of Bernoulli-sampled Map B to drop some pixels within the input image, and ask the network to learn the masked pixels from their neighbors under the assumption that all noise are i.i.d and all pixels are related to their neighbor pixel in Informatics. S2S iterates the noisy image itself multiple times to get performance competitive to the supervised learning.

$$\arg\min_{\theta} \mathbb{E}_{y,B}||\mathcal{F}(B \odot y, \theta) - ((1 - B) \odot y)||^2$$

As the widely-used distance metric Mean Square Error (MSE) can be seen as a combination of bias and variance, the S2S use the Dropout technique to reduce the variance by introducing Bernoulli-sampled Map B. The model uncertainty brought by Dropout (both in input and the model) makes the predictions of these models have independent statistical certainty, so the average of these predictions generated over multiple dropout instances of the input image can reduce the *variance* of the results.

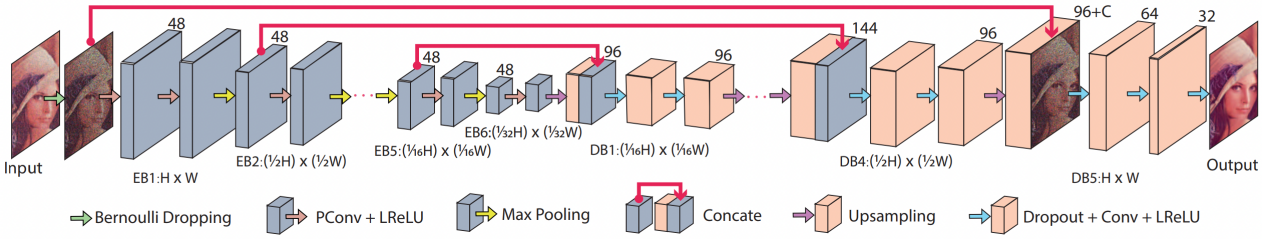


Figure 3: Self2Self architecture figure from [23]. It implements a simple U-Net structure and takes a $B \odot y$ and predicted the masked region $(1 - B) \odot y$.

Although S2S is the current SOTA of unsupersived denoising method, the trade-off is a surprisingly slow run time. The time for S2S to perform one single denoising procedure is around 2H (Default 1.5×10^5 iterations). Another issue is that existing self-supervised deep denoisers suffer from inefficient training and loss of useful information. The mask introduce information loss in the masked pixel region, makes the training hard to converge to true value.

The idea of using mask in training is worth noticing, *Lin et al.* also use [17] the random mask to destroy some structures to achieve the purpose of denoising. Intuitively, repairing the mask requires predicting the vacancy value based on the surrounding information, which is a weighted averaging

operation and also quite an interpolation task, where the image frequency cannot be raised out of and will normally decreases monotonically or ideally equal. A similar work but not related to denoising is discussed in Sec. 3.5.6.

3.5.5 Noisier2Noise

Noisier2Noise [22] outlines adding new noise n' to an original corrupted noisy image y can potentially make them a training pair.

$$z = y + \tilde{n} = x + n + \tilde{n}$$

Thus, the objective function renewed to the following formula:

$$\arg\min_{\theta} \mathbb{E}_z ||\mathcal{F}(z, \theta^*) - y||^2.$$

The true optimal strategy is to simple subtract n' from the z. But note that it is impossible if the network cannot distinguish the differences between original noise n and the artificial counterpart n'. So the best achievable optimal is instead to maximize $\mathbb{E}[y|z]$:

$$\mathbb{E}[y|z] = \mathbb{E}[x|z] + \mathbb{E}[n|z]$$

where an estimation of clean image $\mathbb{E}[x|z]$ is possible to be extracted.

Recall the assumption $\mathbb{E}[n|z] = \mathbb{E}[n'|z]$, the predicted target $\mathbb{E}[y|z]$ can be transformed into:

$$\begin{aligned} 2\mathbb{E}[y|z] &= \mathbb{E}[x|z] + (\mathbb{E}[x|z] + \mathbb{E}[n|z] + \mathbb{E}[n'|z]) \\ &= \mathbb{E}[x|z] + \mathbb{E}[x + n + n'|z] \\ &= \mathbb{E}[x|z] + z. \end{aligned}$$

In other words, we can estimate $\mathbb{E}[x|z] = 2\mathbb{E}[y|z] - z$. Note that, in practice the $\mathbb{E}[x|z]$ will not be an exact reconstruction of the one true clean image x, but instead a mean of all plausible clean images within the distribution.

Our insight is similar to the *Super Resolution*, which can be seen as learning a up-sampling mapping to a single image from its corrupted low-resolution counterpart, the aim of Noisier2Noise can be seen as learning a similar procedure. Learning a mapping from a noiser version (doubly-noise image) to the original noisy image, and can be applied to the noise image for denoising. This procedure is also experimented under the Noisier2Noise framework, it can get a higher PSNR score but with less detail recovered.

However, Noisier2Noise places a strong assumption that the artificial noise and the original noise are similar in order to have $\mathbb{E}[n|z] = \mathbb{E}[n'|z]$. Thus, the above denoising formula only perform well in mathematical play, the true optimization the network will do is not more than learning how to extract n in one y to mimic n', and add it to another y in order to perform $z = y + n'$ in the formula.

This formula only works if we know the noise modeling, otherwise the introduced additional noise n' will further corrupt the noisy image y.

3.5.6 Masked Autoencoder

The idea of Mask Autoencoders (MAE) [10] is fairly simple: masking random patches within the input image alongside with the location information to let network reconstruct the missing pixels.

This scalable approach force networks to learn high-level semantic information for impainting problem, and can generalize well to other high-level problems.

However, the aim of MAE is not to fully recover the image but to learn a scalable learner for computer vision. But in our task, the retrieval of low-level information is critical as to retain as much image signal as possible.

4 Observation

Our idea comes from one observation: Given one noisy image y, human can easily identify texture and color information inside the image. For example, we can identify the face expression, the structural details of the violin and also their original color in the noisy Figure 4. When conducting denoising, people want to remain those information in the output image.

So we start with one question: How can we do that? One may notice that we rely on the knowledge of the world we already learned. Here we can define prior knowledge more intuitively as our basic beliefs or understanding in the absence of information. e.g. In the case of the Figure 4, a prior knowledge over the noisy images basically represents what we think the natural/ original image should look like.

Albeit the early successes in Deep Denoising showing prospective future in this field *pre-se*, the drawbacks are prominent. The current denoising methods tend to perform regional blurring for rendering denoised result, suffering from loss of useful information. Figure 5 shows the denoised result by Self2Self [23], where the facial features and violin details are undermined, and the colors has not been rendered photorealistically.

Figure 4: Noisy image example from BSD68 dataset with Additive Gaussian Noise ($\sigma^2 = 35$)

Figure 5: Self2Self [23] denoising output stoped at the 50000^{th} iteration

Thus, our insight is that if the denoising estimator can identify critical features inside the noisy image y, we can perform denoising without destroying those features. So, it can also be said that we want the denoiser to obtain the prior knowledge through the some observation over the data set.

Our second observation is that we, human, have multiple views available over same object or scene in the real world scenario to understand them thoroughly while the deep networks only with one observation for each data that makes them incomparable to us regarding understand the world. Thus, we design as easier training procedure that similar to the "Matching Game for kids" that given two noisy image y_a and y_b sampled from same one clean image x, we want the network to output what is the invariant features across them. More details see Sec. 7.

5 Abbreviations Reference

Abbreviation	**Original text**	**Reference**
mutual information (MI) NVT	Noisy Visual Training	Feature encoder
D	$\mathbb{R}^{\vec{\theta}}$	Feature decoder

6 Mathematical Settings

For convenience, In this proposal, I keep an original denoising setting, which is assuming the noise n is the i.i.d additive white Gaussian noise (AWGN). Thus the relation between corrupted noisy image y and clean image x can be formulated as below:

$$y = x + n,\ n \sim \mathcal{N}(0, \sigma^2),$$

Consider a denoising network $\mathcal{F}$ with optimal parameter θ^*, which is expected taking noisy y and output the corresponding clean x:

$$x = \mathcal{F}(y, \theta^*) = \mathcal{F}(x + n, \theta^*),$$

In this work, we also assume the synthesize noise $\tilde{n} \propto n \sim \mathcal{N}(0, \sigma^2)$.
A summary of the symbols used, their dimension, their meaning, and availability.

Symbol	**Dimension**	**Meaning**	**Availability**
E	$\mathbb{R}^{\vec{\theta}}$	Feature encoder	$\hat{Z} = \mathcal{E}(Y, \theta)$
D	$\mathbb{R}^{\vec{\theta}}$	Feature decoder	$\hat{X} = \mathcal{D}(Z, \theta)$
$\mathcal{X}$	$\mathbb{R}^N$	Collection of clean images	$X \in \mathcal{X}$
$\mathcal{Y}$	$\mathbb{R}^M$	Collection of observed noisy images	$Y \in \mathcal{Y}$
X	$\mathbb{R}^{H \times W \times C}$	Clean image we would like to estimate	
$\hat{X}$	$\mathbb{R}^{H \times W \times C}$	Estimated clean image	
Y	$\mathbb{R}^{H \times W \times C}$	Noisy image for which we observe	
$\hat{Y}$	$\mathbb{R}^{H \times W \times C}$	Reconstructed noisy image	
$\mathcal{T}$	$\mathbb{R}^{H \times W \times C}$	Noise contamination with unknown distribution	
Y_p	$\mathbb{R}^{H \times W \times C}$	Same X contaminated by $\mathcal{T}_p$	$Y_p \triangleq \mathcal{T}_p(X)$
Y_q	$\mathbb{R}^{H \times W \times C}$	Same X contaminated by $\mathcal{T}_q$	$Y_q \triangleq \mathcal{T}_q(X)$
$Z_\theta^{\{q,p\}}$	$\mathbb{R}^d$		

$Z_{\omega}^{\{q,p\}}$	$\mathbb{R}^d$	
$Z_{\xi}^{\{q,p\}}$	$\mathbb{R}^{2d}$	
$\mathcal{N}$		Multivariate normal distribution
$\mathcal{P}$		Poisson distribution
$\mathcal{U}$		Uniform distribution
μ		
Σ^2		
λ		
$O_{k,d}$	$\mathbb{R}^{k,d}$	Zero vector subscripted by dimensions
$\oplus$		Concatenation of features

7 Methodology

7.1 Description

In this section, we describe in detail of our proposed model/method-agnostic methods. Our goal is to design plug-and-play modules that can enhance the current approach in a variety of ways, yet are easily adaptable to any model without much effort.

Noisy Representation Learning (NRL) is a way to pre-train the model that force the model to separate the image feature and noise feature from noisy image. Mutual Information Optimization (MIO) is designed to minimize the mutual information between image feature and noise feature, thus reducing the information loss during separating.

Pyramid Features Injection and Constraint (PFIC) can restore and provide underlying clean structure information of noisy image in a pyramid way. Unsharpen Masking-based Feature Post-processing (UEFP) is a post-processing that maximize the posterior probability of the output.

Here, we provide a overview workflow of proposed methods and how can them combined with any given model in a plug-and-play way:

Step 1. Plug in *PFIC* in the both input side and output side of the model.

Step 2. Pre-train the model with *NRL* and optimized by *MIO*.

Step 3. Train the model as in usual.

Step 4. Post-processing the result by *UEFP*.

The workflow requires several steps but is simple and easy to follow effortlessly. Each module is described in detail in the subsequent subsections.

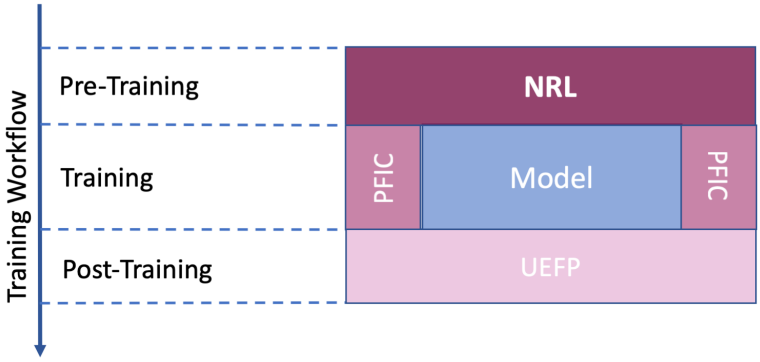


Figure 6:

In Figure 6, we demonstrate a *Jigsaw* that wrap in a blue building block called "Model". It is a visualization of the lifetime of the proposed plug-and-play modules. The blue arrow in the left showing the order of the three stage: Pre-training (Step 2), training (Step 3) and post-training (Step 4). By this way, we want to show that our proposed are surrounding around the model without directly modify its architecture.

7.2 Noisy Representation Learning

NRL's goal is to learn an invariant representation z_θ from different noisy samples observed within the same scene, which can then be used for downstream low-level tasks.

As described previously, our NRL is comprised of three online networks: an *encoder* E_θ, one auxiliary *encoder* E_ω and one *decoder* D_ζ, as shown in Figure 7. We also have three corresponding memory networks, and their parameters $\bar{\xi} \ni \{\bar{\theta}, \bar{\omega}, \bar{\zeta}\}$ are an exponential moving average of the online parameters $\xi \ni \{\theta, \omega, \zeta\}$.

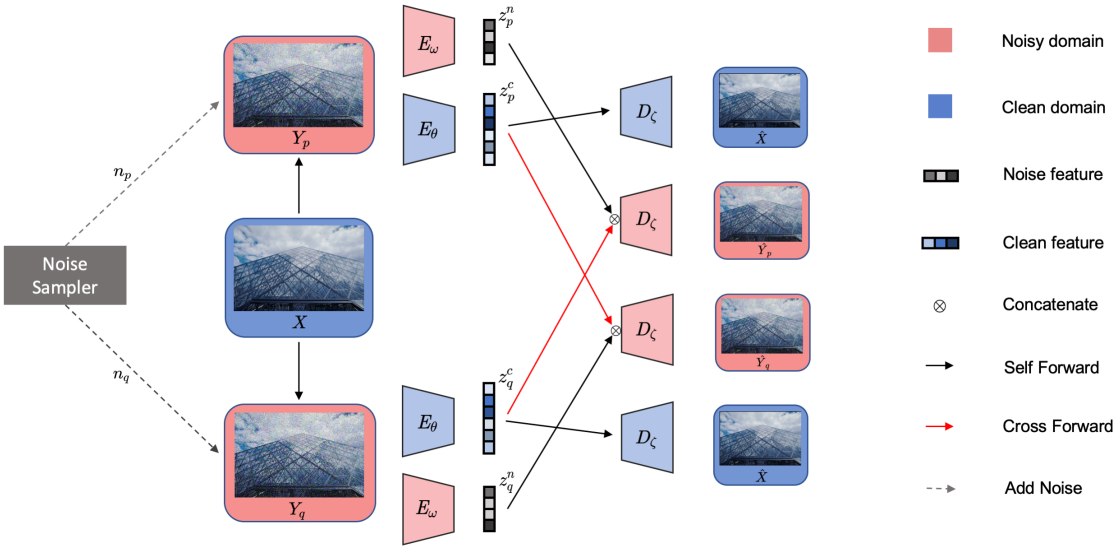


Figure 7: TODO

Each training step has three stages: noisy representation generation, pixel-level restoration with constraint and parameters memorization. More precisely, given a set of images $\mathcal{X}$ and two noise distribution $\mathcal{T}_p$ and $\mathcal{T}_q$, NRL produces two contaminated views $Y_p \triangleq \mathcal{T}_p(X)$ and $Y_q \triangleq \mathcal{T}_q(X)$ from an image $X \sim \mathcal{X}$ sampled uniformly from $\mathcal{X}$. For each view, the E_θ and E_ω estimate corresponding *representation* $Z_\theta \triangleq E_\theta(Y)$ and $Z_\omega \triangleq E_\omega(Y)$. We then ℓ_2-normalize both Z_θ and Z_ω to $Z_\theta \triangleq Z_\theta / \|Z_\theta\|_2$ and $Z_\omega \triangleq Z_\omega / \|Z_\omega\|_2$ and perform regularization.

Since we want to extract invariant feature across different noise disturbance, hereby we define the regularization between a pair of image *features* $\{Z_\theta^p, Z_\theta^q\}$ given the noisy input set $\{Y^p, Y^q\}$ and regularization strength γ:

$$\min_\theta \ \mathcal{L}_\theta^{reg} \triangleq \gamma \|Z_\theta^p - Z_\theta^q\|_2^2 = 2\gamma - 2\gamma \cdot \frac{\langle Z_\theta^p, Z_\theta^q \rangle}{\|Z_\theta^p\|_2 \cdot \|Z_\theta^q\|_2}. \tag{1}$$

The *decoder* D_ζ output $\hat{X}_\zeta \triangleq D_\zeta(Z_\theta \oplus O_d)$ or $\hat{Y}_\zeta \triangleq D_\zeta(Z_\theta \oplus Z_\omega)$ depending on concatenated features input. Note that this decoder is applied to crossed feature when estimating $\{\hat{Y}^p, \hat{Y}^q\}$ in the training step, for example, $\hat{Y}_\zeta^p \triangleq D_\zeta(Z_\theta^q \oplus Z_\omega^p)$ and vice versa. We impose domain-based loss $\mathcal{L}_{\theta,\zeta}^{clean}$ and $\mathcal{L}_\xi^{noisy}$ within clean domain $\{X, \hat{X}_{\theta,\zeta}^p, \hat{X}_{\theta,\zeta}^q\}$ and noisy domain $\{Y^p, Y^q, \hat{Y}_\xi^p, \hat{Y}_\xi^q\}$ separately, and perform a stochastic optimization step to minimize the overall error $\mathcal{L}_\xi^{NRL} = \mathcal{L}_{\theta,\zeta}^{clean} + \mathcal{L}_\xi^{noisy} + \mathcal{L}_\theta^{reg}$ with respect to ξ. NRL's dynamics are summarized as

$$\xi \leftarrow \text{optimizer}\left(\xi, \nabla_\xi \mathcal{L}_\xi^{NRL}, \eta\right), \tag{2}$$

where η is a learning rate. Finally, the memory networks take the parameters ξ from online networks and perform the following update with given target decay rate $\tau \in [0, 1]$,

$$\bar{\xi} \leftarrow \tau\bar{\xi} + (1-\tau)\xi.$$

During inference-time, we only consider the weights $\bar{\xi}$ in the final memory networks. The full architecture is provided in Sec. 7, and overall loss function is summarized in Sec. ??

7.3 Mutual Information Optimization

By visualization, we observe the existing methods prone to eliminate the high-frequency component (image-related) with noise when conducting denoising, which can be attributed to the remaining entanglement of noise and image features. Since our goal is to separate noise (random) representations Z_ω and find out the shared (image-related) latent variable Z_θ from a pair of noisy images, which can be formulated as a Mutual Information Optimization (MIO) problem.

Our basic mutual information framework follows [26], which builds shared representation and then exclusive representation in an incremental way. Given a clean image X, a pair of contaminated views Y_p and Y_q, their shared image *features* $\{Z_\theta^p, Z_\theta^q\}$ and the exclusive noisy features $\{Z_\omega^p, Z_\omega^q\}$, the MIO can be calculated in two stages similar to *MinMax* Optimization.

In the first maximization stage, we estimate and maximize the mutual information between the clean image and their shared representation via statistics networks $T_{\phi_X}^{sh}$:

$$\max_{\phi_X} \mathcal{I}_{\phi_X}^{sh} \triangleq T_{\phi_X}^{sh}(X, Z_\theta^q) + T_{\phi_X}^{sh}(X, Z_\theta^p),$$

and maximize the crossed mutual information between the image views and their corresponding noise representation via the statistics network $T_{\phi}^{ex}{}_Y$:

$$\max_{\phi_Y} \mathcal{I}_{\phi_Y}^{ex} \triangleq T_{\phi_Y}^{ex}(Y_p, Z_\theta^q \oplus Z_\omega^p) + T_{\phi_Y}^{ex}(Y_q, Z_\theta^p \oplus Z_\omega^q).$$

In our expectation, the representation Z_ω must not contain information captured by the representation Z_θ when maximizing the mutual information between Y and $Z_\theta \oplus Z_\omega$. Therefore, the mutual information $\mathcal{I}(Z_\theta, Z_\omega)$ between Z_ω and Z_θ must be minimized, which can be calculated using Kullback-Leibler (KL) Divergence:

$$\begin{aligned} I(Z_\theta, Z_\omega) &:= \mathrm{D}_{KL}(\mathbb{P}(Z_\theta, Z_\omega)||\mathbb{P}(Z_\theta)\mathbb{P}(Z_\omega)) \\ &:= \int\int \mathbb{P}(Z_\theta, Z_\omega) \log\left(\frac{\mathbb{P}(Z_\theta, Z_\omega)}{\mathbb{P}(Z_\theta)\mathbb{P}(Z_\omega)}\right) dZ_\theta\, dZ_\omega, \end{aligned}$$

where the $\mathbb{P}(Z_\theta, Z_\omega)$ is the joint Probability Density Function (PDF), and $\mathbb{P}(Z_\theta)$ and $\mathbb{P}(Z_\omega)$ stand for marginal distributions of Z_θ and Z_ω. Albeit KL Divergence is theoretically brief, the calculation of joint and marginal distributions are computational burdensome. Thus, instead of KL divergence, which concern with its precise value, we use Jensen-Shannon (JS) MI estimator as [12] to estimate this quantity.:

$$\begin{aligned} \hat{I}(Z_\theta, Z_\omega) &:= \mathrm{D}_{JS}(\mathbb{P}(Z_\theta, Z_\omega)||\mathbb{P}(Z_\theta)\mathbb{P}(Z_\omega)) \\ &:= \mathbb{E}_{\mathbb{P}(Z_\theta, Z_\omega)}\left[-\log(1+e^{T_\rho(Z_\theta, Z_\omega)})\right] - \mathbb{E}_{\mathbb{P}(Z_\theta)\times\mathbb{P}(Z_\omega)}\left[\log(1+e^{T_\rho(Z_\theta, Z_\omega)})\right], \end{aligned}$$

where the T_ρ is a statistics networks T defined by a set of weights ρ which trained as a discriminator to classify representations drawn from distribution $P_{Z_\theta Z_\omega}$ or $P_{Z_\theta}P_{Z_\omega}$ as fake samples or real samples.

Thus, minimizing $\mathcal{I}(Z_\theta, Z_\omega)$ is equivalent to minimizing the $\hat{I}(Z_\theta, Z_\omega)$ which can be achieved in an adversarial manner:

$$\min_{\rho} \ \mathcal{L}_\rho^{adv} = \mathbb{E}_{p(z_\theta)p(z_\omega)} \left[\log T_\rho(Z_\theta, Z_\omega)\right] + \mathbb{E}_{p(z_\theta, z_\omega)} \left[\log\left(1 - T_\rho(Z_\theta, Z_\omega)\right)\right].$$

Samples from distribution $P_{Z_\theta Z_\omega}$ are obtained by passing the image views $\{Y^p, Y^q\}$ through the encoders E_θ and E_ω to extract the joint (Z_θ, Z_ω). Samples from $P_{Z_\theta} P_{Z_\omega}$ are obtained by shuffling the noise representations Z_ω within the batch sampled from $P_{Z_\theta Z_\omega}$.

This method under assumption that if we can generate noisy representation Z_ω that combined with Z_θ looks like sample drawn from $P_{Z_\theta} P_{Z_\omega}$, which is equal to Z_ω is a random variable that independent from Z_θ and does not contain any useful information from image.

The overall MIO can be concluded as:

$$\max_{\phi_X, \phi_Y} \ \min_{\rho} \ \mathcal{I}_{\phi_X}^{sh} + \mathcal{I}_{\phi_Y}^{ex} - \mathcal{L}_\rho^{adv}.$$

7.4 Pyramid Features Injection and Constraint

As indicated by the statistical analysis in [31], there exists the clean signal in the noise-contaminated natural images. This strong phenomenon can be viewed when down-scaling the contaminated natural images and can serve as a statistical prior for separating the clean signal from the noise.

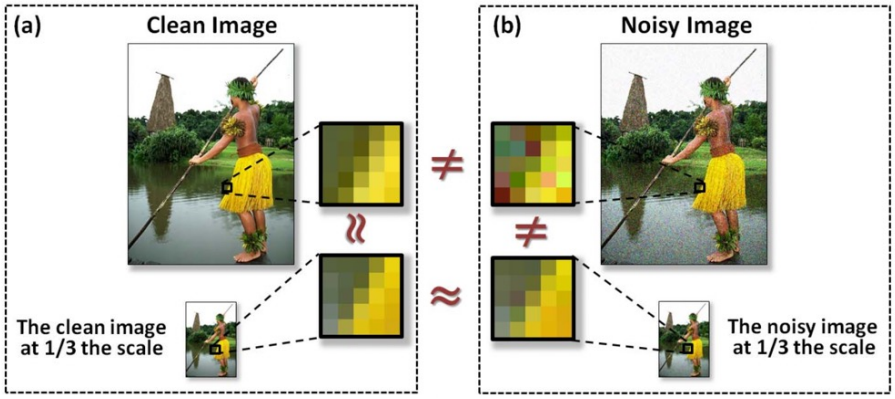


Figure 8: (a) and (b) are corresponding clean image and noisy image and their coarser counterpart at 1/3 resolution. The extracted 5 x 5 patches from all 4 images at the same relative coordinates are displayed in the center. Note that they differ prominently in the finer resolution patches, but the coarser patches are similar. This figure is originated from [31].

With this prior, we design multi-scale pre-prior injection and post-prior constraint that likes like two reversed Pyramid , thus, we call it Pyramid Features Injection and Constraint (PFIC) module. The PFIC is designed to be portable that can plug-and-play into any model architecture such as U-net in Figure 9. The applied isotropic pyramid architecture is depending on the scale and re-scale process of the given model. By incorporating this module, we provide model with structural information on the multi-scale input pyramid and guide denoising in a resolution-wise behaviour on the output pyramid.

The reconstruction error is yielded across-scale in the output isotropic pyramid regarding to the contribution. For example, taking the target resolution at the top of the output pyramid as 1x, whose contribution denotes as 1. A layer which with $\frac{1}{2}$x resolution only contributes $\frac{1}{4}$ as the total number of pixel is only a quarter of the top layer.

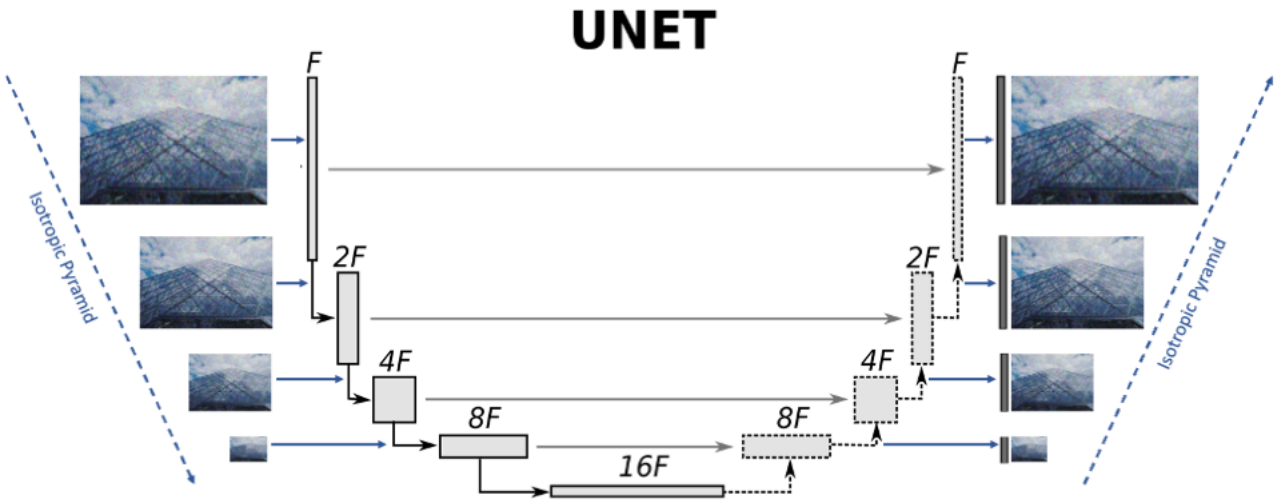


Figure 9: Plug in *Pyramid Features Injection and Constraint* module into U-net structure.

7.5 Unsharpen Masking-based Feature Post-processing

The current denoising method inevitably blurs the image, causing undesired structural information loss. Inspired by Unsharpen Masking (USM) which strengthen the intensity of pixels with large gradient variations, thus strengthening the edge signal at high frequencies. The traditional USM is an image processing technique first practiced in darkroom photography that use to create image that is less blurred than the original one. USM involves two steps: first, it create an negative image which is treated as a unsharpen *mask* by the meaning of blurring the original image. The unsharpen *mask* is then merged with the positive image using a merging weight amount α:

$$\text{sharpened} = \text{original} + (\text{original - blurred}) \times \text{amount}.$$

In our implementation, we design a USM-based Edge Feature Post-processing (UEFP) in a optimizing way that maximizes the posterior probability. Our idea comes from the fact that we can subsequently recover the hidden high frequency information from the blurred denoised output [27]. In practice, we use a Laplacian of Gaussian (LoG) kernel as low-pass filter that denote as L_w with learnable weight w, the UEFP can be formulated as:

$$\tilde{X} = \hat{X} + \alpha(\hat{X} - L_w * \hat{X})$$

where $\hat{X}$ is the clean estimation output by the network and $\tilde{X}$ is the maximum posterior likelihood estimation of $\hat{X}$ using UEFP. The symbol $*$ represents convolution operation and α is a learnable amount.

The UEFP assume that the noise contamination is center at zero with unknown variation, thus, remaining the same statistical mean and median between X and Y. Where X is the clean image underlying and Y is the noisy image we observed. While traditional USM performing in RGB domain, the statistical quantities is more robust in gray-scale domain which combine the distributed RGB channels into one distribution.

Based on these observations, we convert the optimization problem into maximizing the posterior probability in Value channel of HSV domain (hue, saturation and value), which is similar to gray-scale domain representing lightness in intensity but is reversible. The equivalent UEFP optimization can be formulated as:

$$\arg\max_{w,\alpha} P(\tilde{X}|\hat{X}, Y)$$

7.6 Physics-based Noise Modeling

In extreme environment, image capturing process can be affected by many kinds of noise. Majority of denoising tasks nowadays usually format the noise as additive Gaussian noise, which isn't complelling with the real-life environments.

We consider our camera photosensor model primarily based upon the CMOS sensor, which is the dominating imaging sensor nowadays, as the setting in [29]. we systematically analyse the noise statistics in terms of the CMOS sensor physical process in imaging pipeline.

In the following, we will further describe the details of noise resource and the noise introduction procedure.

7.6.1 Shot and Read Noise

During capturing process, CMOS first get incidentally exposed to photons that reflected by the surface of objects. Such exposure liberates photoelectrons in each pixel area on the CMOS that are proportional to the to the quantum intensity hitting some specific area on camera sensor. The capture process of digital sensor raw image Y can be formulated as:

$$Y = K * I + \mathcal{N}$$

where I is the number of photoelectrons that generated proportional to the scene irradiation, K represents the overall camera system gains. And $\mathcal{N}$ is a analog of noise composition.

In nature, the number of photons $\tilde{I}$ captured by camera exist an inevitable uncertainty, which is usually modeled as Poisson random variable:

$$\tilde{I} \sim \mathcal{P}(\tilde{I})$$

where $\mathcal{P}$ denotes the Poisson distribution.

In the photon-to-electron stage, this uncertainty of photons imposes shot noise, which is also following Poisson distribution, and read noise, which can be approximated as a zero-mean Gaussian random variable ("21 Clipped Noisy Image: Heteroskedastic Modeling"). Together, the distribution of electrons can be seen as a single heteroscedastic Gaussian random variable, where mean is equal to the true light intensity $\tilde{I}$ and the variance is parameterized by the read noise λ_r, plus the shot noise λ_s:

$$I \sim \mathcal{N}(\mu = \tilde{I}, \sigma^2 = \lambda_r) + \mathcal{P}(\lambda_s)$$

Read noise is generally assumed to follow Gaussian, but analysis in [29] tells a long-tailed nature of its shape, where using a Tukey lambda distribution (TL) that can approximate a number of common distributions including Gaussian $N(0, 2.142)$ and Cauchy $C(0, \pi)$ with some certain parameters. However, we find Student's T distribution more suitable and flexible with Gaussian-liked long tail assumption.

7.6.2 Camera System Noise

In electrons-to-voltage stage, we notice when analog signal to digital signal, it exists quantization noise which follow uniform distribution with noise interval λ_q.

$$N_q \sim \mathcal{U}(-\lambda_q/2, \lambda_q/2)$$

In addition to this, we introduce N_r to account for banding pattern noise N_b, with scale parameter σ_r for each row/column, which is a camera-dependent noise. It often appears as horizontal or vertical lines in measurement. We simulate it as a zero-mean Gaussian variable with variance λ_b, *i.e.*:

$$N_b \sim \mathcal{N}(mean = 0, \sigma^2 = \lambda_b)$$

then adding it as an fixed offset to all pixel within each row/column.

7.6.3 Color-biased noise

Although zero-mean noise assumption is generally applicable in most situations, we find it break down under extreme noisy settings, because if the non-negligible direct current (DC)(i.e. zero-frequency) noise component. This component originated from the current noise, the averaged number if thermally generated electrons, which renders the noise distribution no long zero-centered. Thus we include a pattern noise component N_c for simulating this color-biased noise. In physical setting, this noise component is fixed that stay constant throughout all images. However, we find that letting it fixed can lead to this fixed parameter be learned and overfitted by the model, thus we experimentally measure and modify it during training.

7.7 Loss Functions Design

In this section we divided the optimization function into three parts: Reconstruction Loss, Regularization Loss and Perceptual Loss.

7.7.1 Reconstruction Loss

Deep image restoration can be seen as a regression problem in the dimension of $\mathbb{R}^{m \times n}$ that not conveniently convex in most of the cases. The intuitive solution is to select the weights that best explain the existing data observations , which is equivalent to maximizes the posterior likelihood function using maximum likelihood estimation (MLE).

Mean Square Error. Mean square error (MSE), which is the error measurement of least squares estimation (LSE) in regression analysis of overdetermined systems. Statistically speaking, MLE can be reduced to minimize the Mean Square Error (MSE) for Gaussian distribution. In the ideal case, the posterior distribution optimized by MSE is normally distributed with bias zero and variance independent of the input variables. In statistical view, the MSE can be seen as:

$$MSE = bias^2 + variance$$

To approximate a Gaussian posterior, the estimators use no non-linear kernel function or activation function in the output and interpret it as mean of a normal distribution. In our implementation, the output activation function is set to $ReLU$ which has value range $[0, \infty)$, for preventing the negative value in restored images. But this setting up comes with several problems.

First is this MLE approximation is also based on the assumptions that the error term must follow Gaussian distribution with constant variance, which is usually not complied in a real-world scenario. Secondly, many standard distribution would go singular (bell shape) at the value of 0 results in the alternative loss functions.

7.7.2 Binary Cross Entropy

Binary Cross-entropy (BCE), or log loss, is usually used to measure the probability value between 0 and 1. One common source of the confusion among deep learning practitioners is that you should use MSE for regression analysis when predicting any numeric value and use BCE for binary classification when the ground truth label are either 0 or 1. Indeed, BCE has the most slippery slope around the value of 0.5 (normally used as the binary classification threshold) that can push-away two clusters makes it suitable for binary classification problem.

The elegance of using BCE for binary classification is that the loss value would be zero if the predictions are correct, in both cases. Under regression scenario, BCE no longer has this property due to the ground-truth labels is numerically in range [0, 1]. Note that in this case, the minimum value of BCE may not be zero at all, but it still indicating the distance of point predictions and true labels which enable its usage in regression problem. For a multivariate Gaussian model with varying mean and constant diagonal covariance, BCE is equivalent to MSE in mathematics.

Assuming that the target data has been normalized to [0, 1], using Sigmoid function as the output activation can prevent numerical out range which is common when not activation is applied. Albeit its superiority of limiting value range and differentiable, it can lead to loss saturation that prevent gradient-based learning algorithms from making progress. In this case, BCE which has logarithm to undo the exp in Sigmoid is preferred.

7.8 Regularization Loss

7.8.1 L1 Regularization

L1 Regularization is commonly used in regression problem for preventing over-fitting. In statistical view, the deviation of model can be measured by bias and variance. For a well generalised model, the bias is associated with its directly performance (e.g. prediction accuracy) while one should also keep the predicting variants low that won't greatly varies the output. Traditional statistical models are designed to be unbiased, and thus, low variance is considered to be an element of a good model.

L1 Regularization hereby introduced in these models by penalise the large variance in the output prediction $\hat{x}$ with the weight λ:

$$\arg\min \lambda \sum_i \|\hat{x}\| .$$

which is adding as a additional term to the loss function, it is also use to learned model parameters θ or other constraint. In our pre-training scheme NRL, we define the L1 regularization between image *features* $\{Z_\theta^p, Z_\theta^q\}$ to explicitly constrained them to be the same:

$$\min_\theta \ \mathcal{L}_\theta^{reg} \triangleq \lambda \|Z_\theta^p - Z_\theta^q\|_2^2 = 2\lambda - 2\lambda \cdot \frac{\langle Z_\theta^p, Z_\theta^q \rangle}{\|Z_\theta^p\|_2 \cdot \|Z_\theta^q\|_2}. \tag{3}$$

7.8.2 Kullback-Leibler Divergence

The Kullback-Leibler Divergence score (KL divergence score) is used to quantify similarity between distributions. This statistical distance is widely used as shortcut to compute other statistical measurement methods, such as mutual information or cross entropy. In essence, the KL divergence distance between two random variables ρ and $\hat{\rho}$ is notated as follows:

$$KL(\hat{\rho} \| \rho)$$

The divergence operator between two variables is written using the || notation, which can be calculated as the sum of all probability over each given point x of distribution $\hat{\rho}$ multiply the log of of the division of probability of $\hat{\rho}$ given point over the probability of $\hat{\rho}$ given event:

$$D_{\mathrm{KL}}(\hat{\rho} \| \rho) = \sum_{x \in \mathcal{X}} \hat{\rho}(x) \log \left(\frac{\hat{\rho}(x)}{\rho(x)} \right)$$

In deep learning, KL divergence can be leveraged as a loss function in forcing the latent space to approximate specific ideal distribution. For example, the latent space of deep models is usually unrestricted that makes it a unexplainable. Some researchers propose to use KL divergence to constrain the latent space to be standard Gaussian that is interpolatable and continuous everywhere. In this way, we can understand the space by interpolation within the latent space and interpret it to output the desired results.

7.9 Perceptual Loss

Perceptual losses are used for measuring the distance in feature space between two different images. For example, if shifting the image by one pixel, the *L2* distance between the shifted version and the original image will be a large due to the pixel-level difference. However, perceptually they are the same due to they are not different within the delivered information.

The Perceptual Loss is proposed for amending the MSE measurement by encouraging perceptually pleasing results, and thus, is widely use in style transferring and image restoration. Normally,

the Perceptual Loss extracts features from images by feed-forwarding them through VGG ϕ network. Then, the high-level distance is calculated by simply measured between the obtained features:

$$\ell_{f\,\text{eat}}^{\phi,j}(\hat{x}, x) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{x}) - \phi_j(x)\|_2^2$$

where C, H and W are channel, width and height of collected feature maps. The y and $\hat{y}$ are target image and estimation which I want to measure distance between.

Tradition VGG-based Perceptual Loss works in many cases, however, the VGG is trained on ImageNet classification task which relies on high semantic feature to determine the decision boundaries. Instead, we use the image coding model from CLIP which is designed for image translation that the feature space is more robust and transferable for image generation. We apply similar approach in defining a new Fréchet Inception Distance (FID) which also use a classification problem-trained model.

8 Experiments

8.1 Discussion of Implementation Details

In this part, we detailed show the implementation details of proposed modules from the step stone of Self2Self model. We adopt a three-stage training scheme: pre-training, training and post-training stages. First, we pre-train the model with PFIC modules attached using NRL. The pre-trained weight is gained by from online model with parameters $\xi \ni \{\theta, \omega, \zeta\}$, which applied exponential moving average (EMA) through the pre-training. Then, the weight is used as initial weight for standard training. Finally Post-processing the result by *UEFP*.

In pre-training stage, we utilize a learning rate of $1e-4$ and weight decay of $1e-5$ on *Adam* optimizer. The training batch is 32 and a step scheduler is applied with step size 10000. We pre-train the model for 50000 iterations on PASCAL VOC 2007 dataset, which has 9963 images (Sec. A.I). Flip, Rotation and MixUp has been applied as image augmentation. All experiments were conducted on NVIDIA Tesla 32G-V100 GPU.

Quantitative performance are measured based on four metrics: peak signal-to-noise ratio (PSNR), structure similarity (SSIM), multi-scale structural similarity (MS-SSIM) and frechet inception distance(FID). Metrics descriptions and calculation details are summarized in Sec. B.

8.2 Experiment of Interpolation

We notice the interpolation method is sensitive in *PFIC*. We compare two widely-used methods: Bilinear, Bicubic, Area and Lanczos4.

Hereby we summary the quantitative comparisons between these methods in Table 3 and Table 4 based on PSNR and SSIM measurement. The experiments are conducted in CBSD68 dataset that we apply different interpolation methods with down-scaling and up-scaling on noisy images with sigma $\sigma = 50$ and then compared with clean counterpart. All interpolations are using algorithms from opencv-python.

DOWN \UP	BILINEAR	AREA	BICUBIC	LANCZOS4
BILINEAR	20.4755	19.2703	19.0238	18.5726
AREA	22.5783	21.7483	21.7866	21.5292
BICUBIC	19.1597	17.5342	17.3654	16.8355
LANCZOS4	18.7355	17.0557	16.8842	16.3516

Table 3: PSNR result of the interpolation experiments.

In Table 3, the vertical direction shows the interpolation methods used in down-scaling while the horizontal direction is for up-scaling. In this table, the red metric shows the highest score which using Area interpolation for down-scaling and Bilinear for up-scaling. The description of these two methods are in Sec. B.IX and Sec. B.VIII.

DOWN \UP	BILINEAR	AREA	BICUBIC	LANCZOS4
BILINEAR	0.3617	0.2987	0.2992	0.2798
AREA	0.4771	0.4162	0.4197	0.4021
BICUBIC	0.2985	0.239	0.2395	0.2212
LANCZOS4	0.2781	0.2237	0.2219	0.2045

Table 4: SSIM result of the interpolation experiments.

Similar finding also appears in Table 4 that the best score is achieved by using Area interpolation for down-scaling and Bilinear for up-scaling. As indicated by this experiment, we apply Area

interpolation in the down-scaling and Bilinear for up-scaling of *PFIC*.

8.3 Evaluation of Self2Self

We conduct experiments on several images sampled from CBSD68 dataset with Self2Self on low-level denoising task. By the limiting of computational power, in this part, we sample 5 examples from CBSD68 dataset involving simple and complex background.

The visual quanlitative results are shown in Table 10.

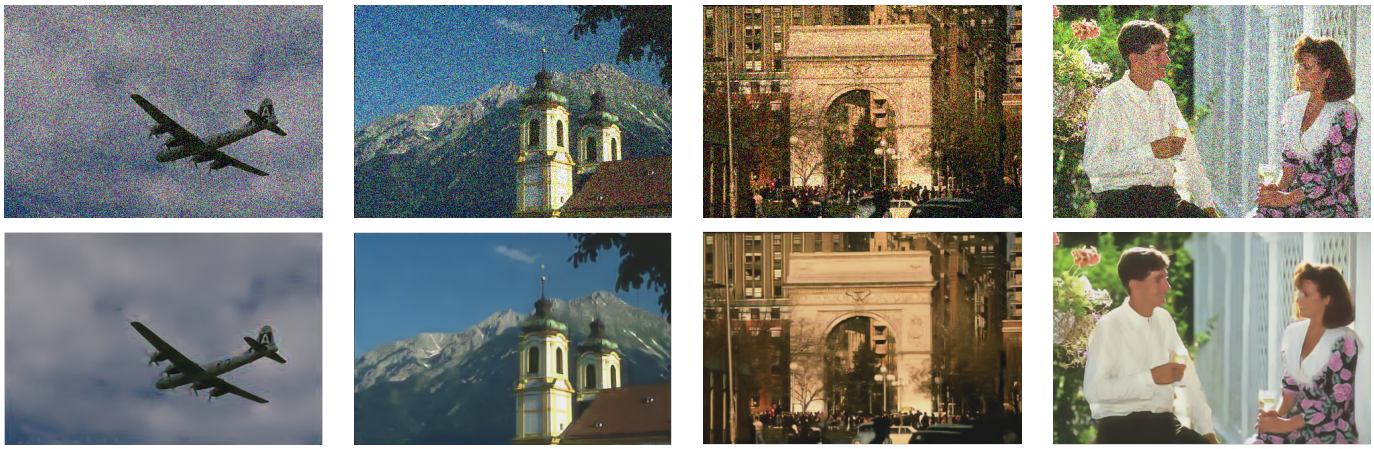


Figure 10: Quantitative result on Self2Self. The first row is the input noisy images from CBSD68 dataset with noisy $\sigma = 50$. The second row is the output from Self2Self with the input image above.

In the Table 5 below, we provide quantitative experiment using three metrics: peak signal-to-noise ratio (PSNR), structure similarity (SSIM) and multi-scale structural similarity (MS-SSIM). The arrow in the table shows the performance improvement direction.

ID	0000	0018	0020	0027	0028	Mean
PSNR/dB ↑	32.0635	25.5077	27.9148	23.8167	25.2757	26.9157
SSIM ↑	0.9405	0.8129	0.8063	0.6871	0.7854	0.8064
MS-SSIM ↑	0.9674	0.9542	0.9360	0.9137	0.9272	0.9397

Table 5: Self2Self quantitative result on CBSD68 dataset with a noise level $\sigma = 50$. ID in the first row means the sample ID from CBSD68 dataset.

Generally, Self2Self can eliminate most of the noise in the input that the noise within the output is not obvious to human visual system. The quantitative results are also the superiority of this single image-based method. The overall performance on CBSD68 dataset in the original paper is 28.70/0.8030 and 25.92/0.6990 of PSNR/SSIM on noisy sigma 25 and 50 separately.

8.4 Experiments on Noisy Representation Learning

In this part, we conduct the experiments of our proposed method by add one module at each time based on the step stone of Self2Self, which results shows in the above subsection.

At first, we testify our pre-training scheme: Noisy Representation Learning. The details structure are described in Figure 7. We have applied several image augmentations for training, including image vertical and horizontal flipping, MixUp and noise injection.

8.4.1 Ablation Study on MixUp

Now that MixUp was originally using on classification problem, but we have transferred it into the image restoration task. It is used to combine two given image into one that contains the objects and structure of both images only with pixel intensity decreased. The motivation of this transferring is that as it can make the boundary of the objects less obvious and make the training harder. It is under the assumption that if the model can successfully find the less obvious contour of training image, handling natural image will be a easier task. The quantitative result in Table 7 also shows utilizing MixUp for training can improve objective score. The experiments in Table 7 are conducted with various noise injection and test on Gaussian noise corrupted image with given level.

Noise Level	25	50	75
w/o MixUp	27.62	24.85	22.63
w/ MixUp	27.97	25.37	23.90

Table 6: PSNR score with different noise level comparison between using the MixUp augmentation or not.

All experiments are conducted on CBSD68 dataset.

8.4.2 Ablation Study on Noise Injection

As we want our model suitable for handling unknown noise types with unknown noise level, we utilize a variable noise injection. In our implementation, each noise type is sampled to a given probability independently with a random sampled noise level within a specific numeric range, e.g. sigma is sampled from 25 to 80 for Gaussian Noise, and 15 to 40 for row and column noise. The types of noise level is detailed discussed in Sec. 7.6. We discover that the introducing of various noise types and noise level can severely undermine the performance of pre-trained model regarding the PSNR. For example, a pre-trained model using only Gaussian noise and sigma 25 would result in a PSNR of 30.42 dB, while applying a different sigma would drop the score to 29.27 dB. Similar discovery also found on random applying multiple noise type. This experimental findings are intuitive as single noise injection is easier compared to multiple type noise injection with variable noise levels.

Noise Level	25	50	75
w/ Fixed Noise	30.42	28.13	25.94
w/ Varying Types	28.35	26.32	24.49
w/ Varying Levels	29.27	27.44	25.21
w/ both	27.97	25.36	23.90

Table 7: Noise Injection experiment of Pre-trained model on Gaussian noise with given noise level.

We can witness varying noise types of levels can causing performance decreasing in pre-trained model. We also observe the trend that when noise level increased, the influence of varying noise level decreased. Further experiment on how noise injection will effect the subsequent model needs more work to verify, details will be discussed in *Future Work* (Sec. 10).

As we intent to equip the model with ability of handling multiple types of noise type and level, we use the full-varying NRL pre-training scheme for subsequent experiments.

8.4.3 Ablation Study on Architecture

In the details of architecture, the common image restoration networks usually apply U-Net like structure which equips one encoder to down-scale the feature maps in order to compress the information and reduce the amount of calculation and a decoder to recover the given feature-maps

and summarised them into the final output with given desired resolution. As shown in the figure, we are implementing two encoders and one shared decoder, which need one more auxilary encoder to conduct this pre-training scheme. In practical we duplication the given encoder and applied as the auxilary encoder. Our figure shows we are using **concatenation** for combining two features space from two separate encoders. The other possible way includeing simply **Add** operation or a combination of **Add** and **Concatenation**. Our experiments show that using only *concatenation* can effectively isolate the two given feature distributions compared with other two methods. But it is worth noting that, the **concatenation** double the feature dimension in the decoder which is said the original output dimension from the encoder is said $C \times W \times H$, but after the operation the dimension will becomes $2C \times W \times H$ which embeds more parameters in the encoders and demand more calculation power. This way also requires to prune the additional input dimension in each layers in order to obtain the original decoder structure for the standard training stage. For example, in the smallest resolution, which usually treated as the bottle neck in a U-Net like structure, we prune the input dimension from $2C$ to C, while in other layers which contains the input from a lower scaled layer ($2C$) and the encoder ($2C$), we prune the input from encoder to C in order to comply with in original structure. In practical, using the *Add* operation instead can omit the step above which makes it more usable and less pre-training information lost. Due to the time limited, we didn't conducted experiment on Add operation but which is included in the future work that is worth on exploring.

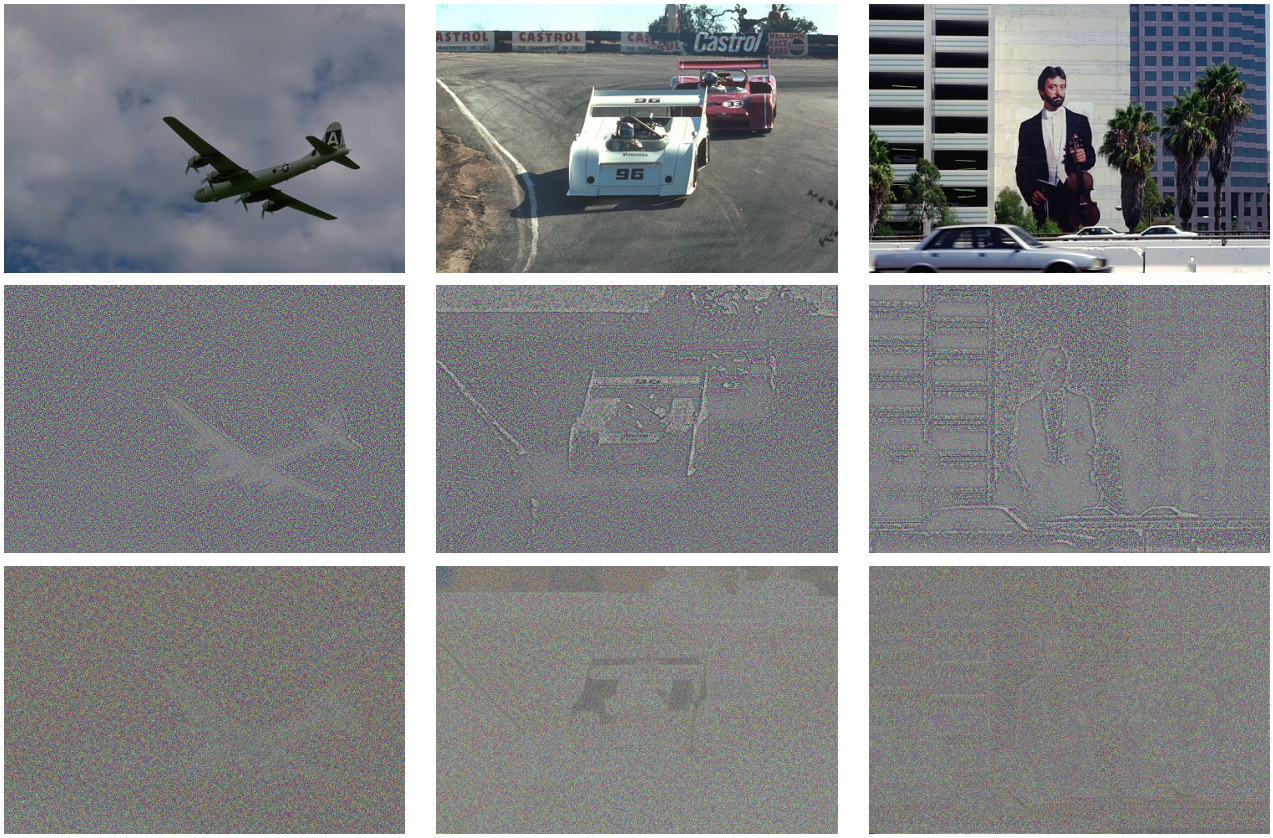


Figure 11: Residual experiments conducted on traditional pre-training scheme and our NRL. The first row is the corresponding clean underlying image. The second row is the residual noisy images from traditional pre-training scheme. The third row is the residual noisy images from NRL. The lower intensity and blurred object contour in the third row shows our method has saved more useful information while denoising.

In Figure 11, we shows the residual experiments of the pre-trained model. In this part, the

residual noisy images R_n are visualization of the eliminated part of the model by conducting input noisy image Y minus the output estimation $\hat{X}$:

$$R_n = Y - \hat{X}$$

In our expectation, the residual noisy images R_n should be similar to completely random noise patterns that do not contain any useful or structural information inside. For comparison, we apply *traditional* image restoration pre-training scheme which use originally given structure and synthesizing noisy image as input and make the output as similar as possible to the underlying clean images (1^{st} row), the results are shown in row 2. The results of our NRL are shown in row 3. As shown in Figure 11, we can easily recognize the objects from the result residual image, e.g. the plane in the center-left image, the car in the center-middle image and the people in the center-right image. Although we can still tell the contour of the object in the corresponding images in the 3^{rd} row, but the signal intensity is decreased, shows we have saved more signals by applying NRL comparing to the traditional pre-training scheme.

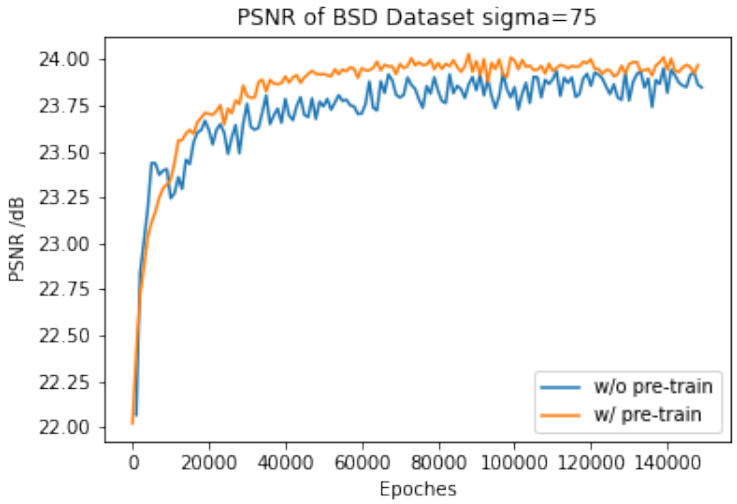


Figure 12: PSNR reported comparing w/ and w/o pre-training.

In the Figure 12, We show a training sample of applying our pre-trained NRL weights to Self-2Self compared with randomized initial weight.The orange line shows the result with pre-trained NRL weight while the blue line shows the result without. This result shows in some cases, using pre-trained NRL weights can give a more stable training trend compared to randomized weight and also improve PSNR to a small margin. Although this improve was not always witnessed in other cases, this more stable training enabled us to a reliable early-stopping. Note that the first 20,000 or 30,000 iterations already gave an acceptable result, while the subsequent iterations over 100,000 changed only two decimal places of the PSNR. In some cases where such high objective score results are not required, this steady early stop can produce a fairly good result to meet the need, so as to save a lot of training time.

In this case, use impression workhorse So improved PSR to toe margin but this can this not always to emerge as shown in some other cases, using prefer model will give me a prominent improvement compared with the randomized with the quantitative results showing the flow table with noise level 25 to certify these most simple training

8.5 Quantitative Experiments

In this part, we provide quantitative experiment using three metrics: peak signal-to-noise ratio (PSNR), structure similarity (SSIM) and multi-scale structural similarity (MS-SSIM).

By the limiting of computational power, in this part, we sample 5 examples from CBSD68 Dataset involving simple and complex background.

Table 8 shows the result of original Self2Self [].

ID	0000	0018	0020	0027	0028	Mean
PSNR/dB ↑	32.0635	25.5077	27.9148	23.8167	25.2757	26.9157
SSIM ↑	0.9405	0.8129	0.8063	0.6871	0.7854	0.8064
MS-SSIM ↑	0.9674	0.9542	0.9360	0.9137	0.9272	0.9397

Table 8: Self2Self quantitative result on CBSD68 dataset with a noise level $\sigma = 50$.

Table 9 shows the result of original Self2Self pluged in with USM post-processing.

ID	0000	0018	0020	0027	0028	Mean
PSNR/ dB ↑						
S2S w/o USM	32.0635	25.5077	27.9148	23.8167	25.2757	26.9157
S2S w/ USM	32.8959	26.5644	28.5094	24.5327	26.1089	27.7223
SSIM ↑						
S2S w/o USM	0.9405	0.8129	0.8063	0.6871	0.7854	0.8064
S2S w/ USM	0.9368	0.8284	0.8122	0.7094	0.7982	0.8170
MS-SSIM↑						
S2S w/o USM	0.9674	0.9542	0.9360	0.9137	0.9272	0.9397
S2S w/ USM	0.9667	0.9589	0.9397	0.9238	0.9345	0.9447

Table 9: Quantitative result on CBSD68 dataset with $\sigma = 50$ and USM parameter $\alpha = 0.2$.

Table 10 shows the result of original Self2Self with PFIC module and USM post-processing.

ID	0000	0018	0020	0027	0028	Mean
PSNR/ dB ↑						
S2S w/o Pyramid	32.0635	25.5077	27.9148	23.8167	25.2757	26.9157
S2S w/ Pyramid	31.9782	25.1394	27.6210	23.4700	24.9063	26.6230
S2S w/ USM	32.8959	26.5644	28.5094	24.5327	26.1089	27.7223
S2S w/ both	33.2086	26.4256	28.4659	24.3840	25.9560	27.6880
SSIM ↑						
S2S w/o Pyramid	0.9405	0.8129	0.8063	0.6871	0.7854	0.8064
S2S w/ Pyramid	0.9554	0.8008	0.8013	0.6721	0.7768	0.8013
S2S w/ USM	0.9368	0.8284	0.8122	0.7094	0.7982	0.8170
S2S w/ both	0.9600	0.8197	0.8094	0.6961	0.7910	0.8153
MS-SSIM ↑						
S2S w/o Pyramid	0.9674	0.9542	0.9360	0.9137	0.9272	0.9397
S2S w/ Pyramid	0.9717	0.9502	0.9348	0.9100	0.9240	0.9382
S2S w/ USM	0.9667	0.9589	0.9397	0.9238	0.9345	0.9447
S2S w/ both	0.9722	0.9563	0.9392	0.9211	0.9319	0.9442

Table 10: Self2Self quantitative result on example images sampled from CBSD68 dataset with a noise level $\sigma = 50$ and ablation study result of USM with parameter $\alpha = 0.2$ and Pyramid architecture.

While with these prospective advantages of the self-supervision *pre-se*, learning without external supervision remains difficult and faces severe practical barriers.

8.6 Numerical Instability Analysis

In the training process, we notice that a frequently occurred behaviour: cost function turning into *Not a Number* (NaN) after a certain number of iterations, normally after the 5000^{th} iteration. This numerical instability could result in non-functional network that undermines the training efforts.

Often times this common problem is introduced by gradient blowing up in updating layer weights and biases that exceeds numerical precision. In practice, these trainable weights take on the invalid value of a NaN because of numerical overflow or underflow.

Some possible reasons include:

- Poor learning rate selection results in exploding weight updates.
- Poor data pre-processing results in target differences or unexpected values (e.g. 0/0 in normalising full zero input image).
- Poor loss function design that allowing the calculation of large error derivative.

Since we have applied Gradient Clipping to limit the error derivative before propagating it *backward* and update the weights, it dramatically decreasing the likelihood of an weight overflow or underflow when updating backwards.

We attribute this problem to Function Numerical Instability in *forward* propagation. One common condition encountered with is the numerical errors of logarithm of 0. In our experiment network, the output activation function is set to ReLU which prevent the negative value in restored images. We apply Binary Cross-Entropy (BCE) as the distance measurement (as discussed in the Sec. **??**) whose probability term should never have zero-values due to the exponential. Since the ReLU function can produce exactly zero (e.g. a negative value gets clipped to 0), the numerical instability of BCE off-tracks the forward process. This behaviour is unexpected in binary classification problem due to the limitation of binary labels, which reduce this instability by $0 \times log(0) = 0$.

Our simple solution is applying *Sigmoid* as output activation and numerical offset $\epsilon = e^{-15}$ to the input of logarithm for numerical stability.

For this reason it is usually common practice to add a small value to assist off-the shelf optimizers and for numerical stability.

9 Research Purpose and Objective

The main objective of this work to develop representation learners that are resilient to many different types of noise with only noisy image. Our final goal is to mitigate the heavy burden of building denoising methods specific to some kind of noise models in different scientific fields.

These representation learner that trained with noisy self-supervision are capable of extracting image signal within noisy images regardless of many kind of noise, thus makes it easy-transferable denoising method. They may can also be applied to some down-stream low-level tasks beyond the denoising, such as colorization, segmentation or other computational imaging problems. Implementation details and mathematical proof will be provided in Section 7. we want to develop a representation learner that is resilient to many different types of noise.

10 Limitations and Future Work

In this work, we have discovered some existing denoising method and have done experiments on some models and noise types. The experiments so far have proved the superiority of the proposed modules. However, different types of noise involving different noise problem formulation, especially for real-world noise, the analysis of the proposed novel training scheme should also be conducted in real life scenario. The influence of using different settings of noise for pre-training is still unknown.

Future works can include ablation study on the influence of pre-training noise setting, how is will impact the sub-sequent tasks and also other testament, e.g. the model performance on unseen noise type, severer noise level or if only trained with single or less noise types. The influence of the amount of pre-training data and the transferring ability of each model architecture (e.g. U-Net, transformer, and MLP).

Secondly, training deep models with only one image is promising but still an open problem that requires a lot more effort into it. We notice the current single image method may affected by the severe noise that cause color distortion and artifacts, some future work may can along this way to improve it.

The third is that since the shared feature among low-level tasks, this methodology designed for denoising can also be expanded to other tasks such as Super-Resolution, Image Impainting and Image Debluring. This is also a promising direction of future work to propose a transferable scheme that can be combined with these methods without requiring modification.

11 Conclusions

As the requirements and complexity of image processing is increasing, the research in more applicable method in this field is still in great demand.

Recently, the rise of deep learning has replaced the traditional methods in many fields, which has created a new development branch, leading significant advances in image processing, including denoising, super-resolution, deblurring and impainting.

In this work, we have investigated the failure of existing single image based denoising task in recovering the image details and showed that a key reason is that current single-image model overfitting the available clean pattern in image that lost the ability to uncover the blinded image details.

To remedy this problem, we propose three modules in pre-training, training and post-training stage separately, aiming to add prior knowledge and maximize the posterior probability of the result image. We showed that these modules stabilizes the training and can significantly improve the details of result images. Therefore, our principal for adding prior and posterior maximization would be beneficial in this scenario. There are some promising directions of future work, which are already discussed in Sec. 10.

12 Acknowledgement

I would like to extend my sincere gratitude to all those who have directed me into the field of Computer Vision and provided me with so much guidance and encouragement throughout the period of my postgraduate study.

My deepest gratitude goes to my supervisor, Prof. Jianbo Jiao, whose rigorous knowledge, thought-provoking guidance, and invaluable spur in various viewpoints have always been treasures for me in the writing process and beyond. Apart from being my supervisor, he is also a wonderful mentor. I have also benefited much from him for his humour and unique ways of thinking. Thanks

to his valuable instruction and warm support, I am able to pursue this study and accomplish it in time. My gratitude is also reserved for my parents and friends. Special thanks are given to my girlfriend, Miss Weiman Chen for his unselfish help and unfailing support.

Though immense help has been received from those whose contributions have been acknowledged above, I take sole responsibility for any errors and failings I have not been able to correct.

References

[1] Abdelrahman Abdelhamed, Marcus A Brubaker, and Michael S Brown. Noise flow: Noise modeling with conditional normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3165–3173, 2019. 3.5.1

[2] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018. 3.3

[3] Jose Carranza-Rojas, Saul Calderon-Ramirez, Adán Mora-Fallas, Michael Granados-Menani, and Jordina Torrents-Barrena. Unsharp masking layer: injecting prior knowledge in convolutional networks for image classification. In *International Conference on Artificial Neural Networks*, pages 3–16. Springer, 2019. 3.4.1

[4] Marie-Neige Chapel and Thierry Bouwmans. Moving objects detection with a moving camera: A comprehensive review. *Computer science review*, 38:100310, 2020. 2

[5] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020. 3.2.1

[6] Happiness Ugochi Dike, Yimin Zhou, Kranthi Kumar Deveerasetty, and Qingtian Wu. Unsupervised learning based on artificial neural network: A review. In *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, pages 322–327. IEEE, 2018. 3.2

[7] Wenchao Du, Hu Chen, and Hongyu Yang. Learning invariant representation for unsupervised image restoration. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 14483–14492, 2020. 3.2.2

[8] Alma Eguizabal, Mats U Persson, and Fredrik Grönberg. A deep learning post-processing to enhance the maximum likelihood estimate of three material decomposition in photon counting spectral ct. In *Medical Imaging 2021: Physics of Medical Imaging*, volume 11595, pages 1080–1089. SPIE, 2021. 3.4

[9] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pretraining help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010. 3.1

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 3.5.6, C

[11] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019. 3.1

[12] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 3.3, 7.3

[13] Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021. 2

[14] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020. 3.2

[15] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 2, 3.2.1

[16] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020. 2

[17] Huangxing Lin, Yihong Zhuang, Yue Huang, Xinghao Ding, Xiaoqing Liu, and Yizhou Yu. Noise2grad: Extract image noise to denoise. In *IJCAI*, pages 830–836, 2021. 3.5.4

[18] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2507–2516, 2019. 2

[19] Ali Maleky, Shayan Kousha, Michael S Brown, and Marcus A Brubaker. Noise2noiseflow: Realistic camera noise modeling without clean images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17632–17641, 2022. 3.5.1

[20] Gary Marcus and Ernest Davis. *Rebooting AI: Building artificial intelligence we can trust*. Vintage, 2019. 2

[21] Kristina Monakhova, Stephan R Richter, Laura Waller, and Vladlen Koltun. Dancing under the stars: video denoising in starlight. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16241–16251, 2022. 3.5.1

[22] Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12064–12072, 2020. 3.5, 3.5.5

[23] Yuhui Quan, Mingqin Chen, Tongyao Pang, and Hui Ji. Self2self with dropout: Learning self-supervised denoising from single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1890–1898, 2020. 3.5, 3.5.4, 3, 4, 5

[24] Abdul Muntakim Rafi, Uday Kamal, Rakibul Hoque, Abid Abrar, Sowmitra Das, Robert Laganiere, Md Kamrul Hasan, et al. Application of densenet in camera model identification and post-processing detection. In *CVPR workshops*, pages 19–28, 2019. 3.4

[25] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016. 2, 1

[26] Eduardo Hugo Sanchez, Mathieu Serrurier, and Mathias Ortner. Learning disentangled representations via mutual information estimation. In *European Conference on Computer Vision*, pages 205–221. Springer, 2020. 3.3, 7.3

[27] Michael W Tao, Jitendra Malik, and Ravi Ramamoorthi. Sharpening out of focus images using high-frequency transfer. In *Computer Graphics Forum*, volume 32, pages 489–498. Wiley Online Library, 2013. 7.5

[28] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018. 3.5, 3.5.3, 2

[29] Kaixuan Wei, Ying Fu, Yinqiang Zheng, and Jiaolong Yang. Physics-based noise modeling for extreme low-light photography. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 7.6, 7.6.1

[30] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. 2

[31] Maria Zontak, Inbar Mosseri, and Michal Irani. Separating signal from noise using patch recurrence across scales. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1195–1202, 2013. 7.4, 8

Part
Appendix

Appendix A Dataset

The same with .. we using PASCAL VOC 2007 dataset for training.

Testing set using BSD68, KODAK dataset.

In this section, we provide more information about them.

A.I PASCAL VOC 2007

The PASCAL VOC dataset is serving as one of the benchmarks for object detection methods.

There are 20 classifications in this dataset. The dataset contains 9963 images for training and validation, with 24,640 objects in the region of interest.

PASCAL VOC provides an excellent set of standardized datasets for image recognition and classification, and an image recognition challenge was held every year from 2005 to 2012. the main goal of the challenge is to identify some classes of objects in real scenes. In this challenge, it is a supervised learning problem and the training set is given in the form of images with labels.

Unlike other method using this dataset for high-level tasks, we think this dataset contains dense objects and texture that can be a good basis for providing sufficient and diversified image information for learning invariant features.

A.II BSD68

Color BSD68 dataset for image denoising benchmarks It is part of The Berkeley Segmentation Dataset and Benchmark https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

This benchmark dataset is widely used for measuring image denoising algorithms performance, however I could not find it easily. It includes the original .jpg files, converted to lossless .png, and noisy with Additive White Gaussian Noise of different levels.

A.III KODAK

The pictures below link to lossless, true color (24 bits per pixel, aka "full color") images. It is my understanding they have been released by the Eastman Kodak Company for unrestricted usage. Many sites use them as a standard test suite for compression testing, etc. Prior to this site, they were only available in the Sun Raster format via ftp. This meant that the images could not be previewed before downloading. Since their release, however, the lossless PNG format has been incorporated into all the major browsers. Since PNG supports 24-bit lossless color (which GIF and JPEG do not), it is now possible to offer this browser-friendly access to the images.

A.IV SIDD

The last decade has seen an astronomical shift from imaging with DSLR and point-and-shoot cameras to imaging with smartphone cameras. Due to the small aperture and sensor size, smartphone images have notably more noise than their DSLR counterparts. While denoising for smartphone images is an active research area, the research community currently lacks a denoising image dataset representative of real noisy images from smartphone cameras with high-quality ground truth. We address this issue in this paper with the following contributions. We propose a systematic procedure for estimating ground truth for noisy images that can be used to benchmark denoising performance for smartphone cameras. Using this procedure, we have captured a dataset, the Smartphone Image

Denoising Dataset (SIDD), of 30,000 noisy images from 10 scenes under different lighting conditions using five representative smartphone cameras and generated their ground truth images. We used this dataset to benchmark a number of denoising algorithms. We show that CNN-based methods perform better when trained on our high-quality dataset than when trained using alternative strategies, such as low-ISO images used as a proxy for ground truth data.

Appendix B Metrics

B.I PSNR

The widely used loss metric in demonising, and in the majority of other image processing fields is the Mean Square Error (MSE), which is mathematically equal to Peak Signal-to-Noise Ratio (PSNR), a widely recognised objective metric.

B.II SSIM

The structural similarity index measure (SSIM) is a method for predicting the perceived quality of digital television and cinematic pictures, as well as other kinds of digital images and videos. SSIM is used for measuring the similarity between two images. The SSIM index is a full reference metric; in other words, the measurement or prediction of image quality is based on an initial uncompressed or distortion-free image as reference.

The SSIM index is calculated on various windows of an image. The measure between two windows x and y of common size $N \times N$ is: [4]

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

with: - μ_x the average of x; - μ_y the average of y; - σ_x^2 the variance of x; - σ_y^2 the variance of y; - σ_{xy} the covariance of x and y; - $c_1 = (k_1 L)^2, c_2 = (k_2 L)^2$ two variables to stabilize the division with weak denominator; - L the dynamic range of the pixel-values (typically this is $2^{\#\text{ bits per pixel}} - 1$); - $k_1 = 0.01$ and $k_2 = 0.03$ by default.

Quantitative results. Recent work [27] finds that FID measured by InceptionNet [18] may not correlate with the perceptual quality, as the model is initially trained for ImageNet classification. Following [27], we calculate FID [18] with the CLIP model [39] whose feature space is more robust and transferable. Table 1 shows our method consistently outperforms the model without pretraining by a large margin. Compared with the leading approach, OASIS [44], on mask-to-image synthesis, our method obtains significant improvements (5:9 on ADE20K, 3:6 on COCO, and 4:4 on Flickr) in terms of FID. Our general approach also shows promising performance on sketch-to-image and geometry-to-image synthesis tasks

B.III Fréchet inception distance (FID)

The Fréchet inception distance (FID) is a metric used to assess the quality of images created by a generative model, like a generative adversarial network (GAN).[1][2] Unlike the earlier inception score (IS), which evaluates only the distribution of generated images, the FID compares the distribution of generated images with the distribution of real images that were used to train the generator.[1][3]

The FID metric was introduced in 2017,[1] and is the current standard metric for assessing the quality of generative models as of 2020. It has been used to measure the quality of many recent models[3] including the high-resolution StyleGAN1[4] and StyleGAN2[5] networks.

Definition [edit] For any two probability distributions over $\mathbb{R}^n$ having finite mean and variances, μ, ν, their Fréchet distance is [6]

$$d_F(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{\mathbb{R}^n\times\mathbb{R}^n} \|x-y\|^2 \, \mathrm{d}\gamma(x,y)\right)^{1/2},$$

where $\Gamma(\mu,\nu)$ is the set of all measures on $\mathbb{R}^n \times \mathbb{R}^n$ with marginals μ and ν on the first and second factors respectively. (The set $\Gamma(\mu,\nu)$ is also called the set of all couplings of μ and ν..). In other words, it is the 2-Wasserstein distance on $\mathbb{R}^n$. For two multidimensional Gaussian distributions $\mathcal{N}(\mu,\Sigma)$ and $\mathcal{N}(\mu',\Sigma')$, it is explicitly solvable as [7]

$$d_F\left(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu',\Sigma')\right)^2 = \|\mu-\mu'\|_2^2 + \operatorname{tr}\left(\Sigma+\Sigma'-2\left(\Sigma^{\frac{1}{2}}\cdot\Sigma'\cdot\Sigma^{\frac{1}{2}}\right)^{\frac{1}{2}}\right)$$

This allows us to define the FID in pseudocode form: INPUT a function $f : \Omega_X \to \mathbb{R}^n$. INPUT two datasets $S, S' \subset \Omega_X$. Compute $f(S), f(S') \subset \mathbb{R}^n$. Fit two gaussian distributions $\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu',\Sigma')$, respectively for $f(S), f(S')$. RETURN $d_F\left(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu',\Sigma')\right)^2$ In most practical uses of the FID, Ω_X is the space of images, and f is an Inception v3 model trained on the ImageNet, but without its final classification layer. Technically, it is the 2048dimensional activation vector of its pool3 layer. [3]

https://machinelearningmastery.com/how-to-implement-the-frechet-inception-distance-fid-from-scratch/

B.IV Inception Score (IS)

What Is the Inception Score? The Inception Score, or IS for short, is an objective metric for evaluating the quality of generated images, specifically synthetic images output by generative adversarial network models. The inception score was proposed by Tim Salimans, et al. in their 2016 paper titled "Improved Techniques for Training GANs." In the paper, the authors use a crowd-sourcing platform (Amazon Mechanical Turk) to evaluate a large number of GAN generated images. They developed the inception score as an attempt to remove the subjective human evaluation of images. The authors discover that their scores correlated well with the subjective evaluation.

The inception score involves using a pre-trained deep learning neural network model for image classification to classify the generated images. Specifically, the Inception v3 model described by Christian Szegedy, et al. in their 2015 paper titled "Rethinking the Inception Architecture for Computer Vision." The reliance on the inception model gives the inception score its name.

A large number of generated images are classified using the model. Specifically, the probability of the image belonging to each class is predicted. These predictions are then summarized into the inception score.

The inception score is calculated by first using a pre-trained Inception v3 model to predict the class probabilities for each generated image.

These are conditional probabilities, e.g. class label conditional on the generated image. Images that are classified strongly as one class over all other classes indicate a high quality. As such, the conditional probability of all generated images in the collection should have a low entropy.

As an alternative to human annotators, we propose an automatic method to evaluate samples, which we find to correlate well with human evaluation

B.V Learned Perceptual Image Patch Similarity (LPIPS)

https://torchmetrics.readthedocs.io/en/v0.8.2/image/learned$_p erceptual_i mage_p atch_s imilarity.html$

B.VI Subjective

B.VII Interpolation

B.VIII Bilinear Interpolation

Bilinear interpolation is one of the basic resampling techniques in computer vision and image processing, where it is also called bilinear filtering or bilinear texture mapping.

As the name suggests, the bilinear interpolant is not linear; but it is linear (i.e. affine) along lines parallel to either the x or the y direction, equivalently if x or y is held constant. Along any other straight line, the interpolant is quadratic. Even though the interpolation is not linear in the position (x and y), at a fixed point it is linear in the interpolation values, as can be seen in the (matrix) equations above.

The result of bilinear interpolation is independent of which axis is interpolated first and which second. If we had first performed the linear interpolation in the y direction and then in the x direction, the resulting approximation would be the same.

Repeated linear interpolation [edit] We first do linear interpolation in the x-direction. This yields

$$f\left(x, y_1\right) = \frac{x_2 - x}{x_2 - x_1} f\left(Q_{11}\right) + \frac{x - x_1}{x_2 - x_1} f\left(Q_{21}\right),$$
$$f\left(x, y_2\right) = \frac{x_2 - x}{x_2 - x_1} f\left(Q_{12}\right) + \frac{x - x_1}{x_2 - x_1} f\left(Q_{22}\right).$$

We proceed by interpolating in the y-direction to obtain the desired estimate:

$$\begin{aligned} f(x, y) &= \frac{y_2 - y}{y_2 - y_1} f\left(x, y_1\right) + \frac{y - y_1}{y_2 - y_1} f\left(x, y_2\right) \\ &= \frac{y_2 - y}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f\left(Q_{11}\right) + \frac{x - x_1}{x_2 - x_1} f\left(Q_{21}\right)\right) + \frac{y - y_1}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f\left(Q_{12}\right) + \frac{x - x_1}{x_2 - x_1} f\left(Q_{22}\right)\right) \\ &= \frac{1}{\left(x_2 - x_1\right)\left(y_2 - y_1\right)}\left(f\left(Q_{11}\right)\left(x_2 - x\right)\left(y_2 - y\right) + f\left(Q_{21}\right)\left(x - x_1\right)\left(y_2 - y\right) + f\left(Q_{12}\right)\left(x_2 - x\right)\left(y - y_1\right) + \right. \\ &= \frac{1}{\left(x_2 - x_1\right)\left(y_2 - y_1\right)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f\left(Q_{11}\right) & f\left(Q_{12}\right) \\ f\left(Q_{21}\right) & f\left(Q_{22}\right) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}. \end{aligned}$$

Note that we will arrive at the same result if the interpolation is done first along the y direction and then along the x direction. [11]

Alternative matrix form [edit] Combining the above, we have

$$f(x, y) \approx \frac{1}{\left(x_2 - x_1\right)\left(y_2 - y_1\right)} \begin{bmatrix} f\left(Q_{11}\right) & f\left(Q_{12}\right) & f\left(Q_{21}\right) & f\left(Q_{22}\right) \end{bmatrix} \begin{bmatrix} x_2 y_2 & -y_2 & -x_2 & 1 \\ -x_2 y_1 & y_1 & x_2 & -1 \\ -x_1 y_2 & y_2 & x_1 & -1 \\ x_1 y_1 & -y_1 & -x_1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \\ xy \end{bmatrix}.$$

B.IX Area Interpolation

Appendix C Possible Risks

Possible problems I foresee with this capstone is the proposed methods won't work just like the assumption. Beyond that technical issues with potential algorithm design problem could lead to issues in the long run.

The potential solution is to from re-implementing the current workable method to my own algorithm. Thus can prevent basic design problem prevent algorithm convergence, and it is more easy to identify the issues.

Some other problem may barrier the project is the capability of the model. In MAE [10], the high-capability *ViT-Huge* was used to perform representation learning. Model of this size may be the reason why MAE outperforms current methods, but it is nearly impossible for this project to employ a large model like this. This capability issue may be potential problem in the future, which cannot be identify at this time.