UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Multi-task learning for image restoration

Author:
Tetiana MARTYNIUK

Supervisor:
Orest KUPYN

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

UKRAINIAN CATHOLIC UNIVERSITY

Abstract

Faculty of Applied Sciences

Master of Science

Multi-task learning for image restoration

by Tetiana MARTYNIUK

We present an efficient end-to-end pipeline for general image restoration. The setting has a generic encoder and separate decoders so that our model can benefit from the shared low-level feature representations between the tasks. We also introduce the new architecture for the generator inspired by the feature pyramid networks for dealing with multi-scale degradations. We train the models for solving three particular image restoration problems: deblurring, dehazing, and raindrop removal.

Acknowledgements

I owe a big thanks to Orest Kupyn for supervision and unfailing support throughout the research, for sharing the insights and cheering up constantly, for generating new ideas and giving valuable feedback. Special gratitude goes out to Oles Dobosevych for always having my back and exploring my interest in computer vision and image restoration in particular. I express my sincere thanks to Oleh Smolkin for all the late-night debug sessions in Sheptytsky Center. I thank Prof. Zhangyang Wang and his PhD student Junru Wu from Visual Informatics Group at Texas A&M University (VITA) for the collaboration and bringing their experience, especially regarding the generation of datasets (Section 4.1.2). I also want to thank Let's Enhance and Oles Dobosevych for providing the computational resources. I am extremely grateful to Grammarly, Inc and Ukrainian Catholic University for granting me scholarships to cover my tuition fees. Finally, my deepest gratitude goes to my family as they reminded me of this thesis every single day as if I could forget.

Contents

List of Figures

List of Tables

List of Abbreviations

cGAN	**c**onditional GAN
CNN	**C**onvolutional **N**eural **N**etwork
FPN	**F**eature **P**yramid **N**etwork
GANs	**G**enerative **A**dversarial **N**etworks
ILSVRC	**I**mageNet **L**arge **S**cale **V**isual **R**ecognition **C**hallenge
MLP	**M**ulti-**L**ayer **P**erceptron
PSNR	**P**eak **S**ignal-to-**N**oise **R**atio
SOTA	**S**tate-**O**f-**T**he-**A**rt
SR	**S**uper-**R**esolution
SSIM	**S**tructural **Sim**ilarity
WGAN	**W**asserstein GAN
WGAN-GP	**W**asserstein GAN with **G**radient **P**enalty

To my caring family

Chapter 1

Introduction

Image restoration is one of the key computer vision tasks that aims to transform (or somehow manipulate) the corrupted image so it becomes as close to the original target as possible with respect to the overall visual appearance, perceptual quality, visual content, or resulting performance of any other algorithm that uses image data as an input (say, object detection) and requires image restoration techniques as a preprocessing step. It has a counterpart, image enhancement, the task that is focused on transforming the ground truth image, so it becomes more pleasing to the eye or gets better quality overall. One of the common problems for both tasks is the evaluation of the performance, as the words “quality”, “pleasant”, “perception” have little to do with measurability.

The topics related to as image restoration and image enhancement tasks are diverse, among which super-resolution, denoising, deblurring, dehazing, enlightening (low-to-normal lighting enhancement), raindrop and rain strikes removal, snow removal, and the list goes on. They are witnessing huge interest from the computer vision communities, as the solutions to these problems are required as high-tech systems become more autonomous and sophisticated. For instance, video surveillance and security systems have to deal with various environmental and weather conditions and imperfections of camera lenses (raindrops on the lens, motion blur caused be movements and drag, low resolution of camera sensors, graininess, focus problems, etc.) There are many other applications: autonomous driving, enhancement of visual media content, biomedical processing; even Airbnb claims that hosts with high-quality photos earn 40% more than other hosts in their area.

The state-of-the-art image restoration and enhancement approaches heavily rely on deep learning models, as they are able to learn data representations directly from data in a hierarchical layer-based structure. More precisely, CNNs serve as a basic tool in computer vision models as they capture images’ spatial locality. Lately, huge progress was achieved with applying GANs [Goodfellow et al., 2014] to these tasks and treating them as image-to-image translation problems conditioning on the input [Isola et al., 2016]. Even though cGANs and pix2pix [*pix2pix*] framework were designed as a general-purpose solution, producing reasonable results on a wide variety of image-to-image translation problems, the research papers are still mostly focused on single-task solutions. As was already mentioned, the metrics are a tough issue in this domain, and the approaches based on GANs often lag behind "numerically" (SRGAN loses to SRResNet in PSNR and SSIM on traditional benchmarks [Ledig et al., 2016], DeblurGAN loses to Nah, Kim, and Lee, 2016 in PSNR), as they try to hallucinate realistic details and make their outputs indistinguishable from the images drawn from the real-data distributions, but the common-used metrics are still MSE-based, favouring for rather blurry results obtained averaging all plausible outputs. Nevertheless, GAN-based models produce perceptually convincing results in super-resolution [Ledig et al., 2016; Wang et al., 2018; Bulat, Yang, and Tzimiropoulos, 2018], deblurring [Kupyn et al., 2017; Madam Nimisha, Sunil, and Rajagopalan, 2018], dehazing [Li et al., 2018; Du and Li, 2018; Engin, Genç, and Ekenel, 2018; N. and N, 2018], underwater image restoration [Yu, Qu, and Hong, 2019; Fabbri, Islam, and Sattar, 2018], deraining [Zhang, Sindagi, and Patel, 2017], denoising [Chen et al., 2018; Tripathi, Lipton, and Nguyen, 2018] and many others.

Traditionally, these tasks are treated separately, developing new architecture blocks for each specific problem, introducing additional loss components, using some auxiliary constraints. **Our contribution** here is that we develop a pipeline that benefits from tackling these problems altogether, as they all are low-level computer vision tasks and they all aim to map tensors to tensors, no more no less. We present an end-to-end learned method for several image restoration tasks and show results for three specific ones: deblurring, dehazing, and raindrop removal. The setting has a generic encoder and separate decoders so that the model both shares the low-level feature representations between the tasks, and, afterwards, concentrates every branch (decoder) on its specific problem, allowing to learn task-specific details. The only thing that differs for those decoders is data that goes through, so to re-train the model for a different subset of tasks one only has to provide corresponding (paired) datasets. Each "branch" is considered to be a generator and has its own discriminator, so the whole setting is trained in conditional GANs manner. Such setting is flexible architecture-wise, so we try different generator architectures to compare the performance of the models. We introduce new FPN-inspired [Lin et al., 2016] generator architecture with InceptionResNetV2 backbone [Szegedy, Ioffe, and Vanhoucke, 2016] to deal with degradations at different scales. We use VGG-feature based losses [Johnson, Alahi, and Li, 2016; Mechrez, Talmi, and Zelnik-Manor, 2018] for perceptual quality assurance.

Our approach differs from the existing ones: it does not rely on the physical models of the degradations [Pan et al., 2018], it does not restore the images with specific multiple degradations [Dong et al., 2018; Zhang et al., 2018], it does not rely on a specific image generator structure [Ulyanov, Vedaldi, and Lempitsky, 2017]. Our approach is not limited to specific tasks: it takes advantage of the common features that image degradations share and leaves room for each of the tasks to learn its specific restoration parameters.

The rest of the thesis is organized as follows. In Chapter 2 we describe related work on image restoration and multi-task learning in general and background knowledge on neural networks and specific architectures. In Chapter 3 we go into details regarding our proposed pipeline, network architectures, and objective functions. In Chapter 4 we describe the datasets, discuss the choice of metrics, and describe technical details about the experiments, providing quantitative and qualitative evaluation with examples of the networks' outputs. In Chapter 5 we conclude what was done and described in previous chapters.

Chapter 2

Related work

2.1 Neural networks

An artificial neural network is a brain-inspired complex system which is designed to replicate the way we, humans, learn. The neural network normally consists of the input layer, a hidden layer, and the output layer; if it has multiple hidden layers, it is sometimes called "deep neural network", though nowadays "deep" is usually omitted as deep learning is at the peak of AI hype cycle. The simplest deep neural network is *multilayer perceptron*, where all neurons of neighbouring layers are connected between each other:

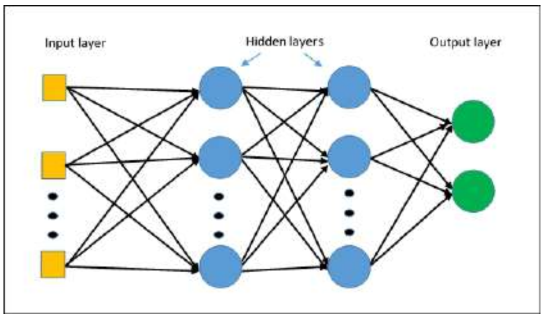


FIGURE 2.1: MLP architecture. [*Image source*]

The nodes of the neural network are extremely interconnected and have activation functions that provide non-linearities. The training of the neural network consists of forward and backward passes. In the forward pass, we push the input through the layers and calculate the error function. In the backward pass, we figure out how each weight affects the total error computing the gradients via the chain rule, and modify the network weights to decrease the error. In the perfect setting, we continue training until the error function converges to zero.

Neural networks are perfect tools to investigate hidden patterns, especially in high-dimensional spaces, and they can solve tremendously complex problems that require more than hand-crafted features and heuristics.

2.1.1 CNNs

Convolutional neural networks [Lecun et al., 1998] are such neural networks that have at least one layer with convolutional operations instead of standard matrix multiplication. [Goodfellow, Bengio, and Courville, 2016]

CNNs are a subcategory of deep learning models that have proven very powerful in computer vision tasks. With local receptive fields, neurons can extract elementary visual features such as oriented edges, end-points, corners (or similar features in other signals such as speech spectrograms). These features are then combined by the subsequent layers in order to detect higher-order features. [Lecun et al., 1998]

There were 4 main building blocks in the well-known LeNet-5, a classical CNN architecture:

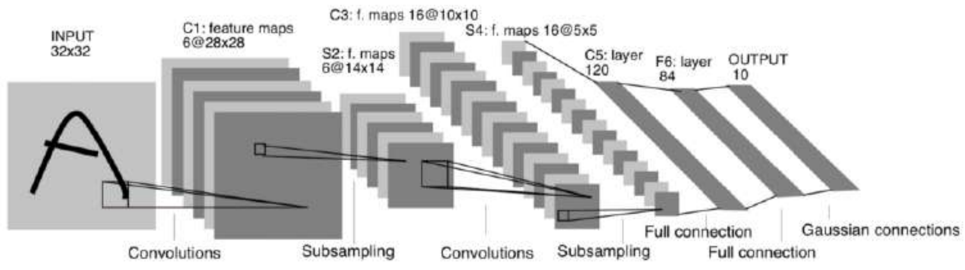


FIGURE 2.2: LeNet architecture. [Lecun et al., 1998]

- convolutional layer. Computes a dot product between the kernel weights and a small patch of the input, has trainable parameters.
- subsampling layer. Performs downsampling along the spatial dimensions, doesn't have trainable parameters.
- non-linearity, or activation function. Applies non-linear activation function, e.g. ReLU [Nair and Hinton, 2010], Leaky ReLU [Maas, 2013], tanh; doesn't have trainable parameters.
- fully-connected layer. Standard MLP-style layer, has trainable parameters.

There are few things that make CNNs so suitable for the computer vision tasks (in comparison to multi-layer perceptron architecture): images as the inputs are very large (thousands or millions of pixels), so feeding the image directly to the fully-connected layer will lead to memory issues and lack of training data to catch up with the number of trainable parameters of the network. Moreover, we would like the network to be robust to the local distortions and translations of the input.

Fully-connected architectures ignore the topology of the input, while images have strong 2D local structure. CNNs force the extraction of local features by restricting the receptive fields of hidden units to be local.

Once a feature has been detected the relevance of its exact position goes down, as now only its approximate location relative to other features matters.

Few specific architectures that need to be mentioned:

- AlexNet [Krizhevsky, Sutskever, and Hinton, 2012] became a breakthrough in deep learning as Krizhevsky et al. presented a much deeper and wider network that won ILSVRC [*ILSVRC*] by a large margin. All current widespread applications of CNNs in computer vision owe AlexNet their success.
- VGGNet [Simonyan and Zisserman, 2014] has shown that the depth matters and is still a critical component for performance. Its feature maps are used for designing perceptual losses due to the depth and semantic richness.
- ResNet [He et al., 2015] has shown very powerful representational ability and overcame the 'vanishing gradient' problem by introducing a skip-connection that acts as a shortcut for one or more layers.

2.1.2 Deep Generative Models. GANs.

Deep generative models are neural networks that can learn and imitate the data distribution one feeds to them (basically, the distribution of the input data of any nature the task is based on, from images to texts). When training a deep generative network one tries to find parameters that make their model closest to real but unknown data distribution. [Nalisnick et al., 2019]

There are two dominant and efficient types of generative models: VAE [Kingma and Welling, 2013] (Variational Autoencoders), and GANs [Goodfellow et al., 2014] (generative adversarial networks). The latter are of particular interest of ours.

In the generative model setup, we know that the samples come from different distributions, but finding a two-sample test objective in high dimensions is hard. The key idea behind GANs then is to learn a statistic that maximises a suitable notion of distance between the two sets of the sample. [*CS236*]

The Goodfellow's GANs framework [Goodfellow et al., 2014] consists of two models: a discriminator D and a generator G, and a two-player minimax game is played between them. The discriminator estimates how likely the given data comes from the real dataset (D is also called 'critic' sometimes), and is optimised to distinguish fake samples from the real ones. The generator receives noise as the input and outputs artificial samples; it is trained to 'trick' the discriminator capturing the real data distribution and making those synthetic samples as difficult to tell from the true ones as possible.

FIGURE 2.3: source: *Quora*

Yann LeCun, Director of AI Research at Facebook and Professor at NYU
Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook and Huang Xiao

Adversarial training is the coolest thing since sliced bread.

A generator is a directed, latent variable model with a deterministic mapping between the input noise and the generated output. It minimises a two-sample test objective. Unlike a variational autoencoder, there is no inference network which can learn a variational posterior over latent variables. A discriminator is any function; in particular, it may be a neural network, that maximises the two-sample test objective.

The minimax game with the value function $V(D,G)$ then is nothing else but [Goodfellow et al., 2014]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[\log(1-D(G(z)))\right]$$

At the early stages of training, when G is still too primitive to fool the discriminator D, the confidence of D about rejecting the sample is very high, thus $D(G(z))$ is nearly zero leading to the saturation of the second term $\log(1-D(G(z)))$. As that is the only term dependent on G, one can change the task of minimising $\log(1-D(G(z)))$ to maximising $log(D(G(z)))$. The new objective then helps with vanishing gradients on the early stages of the training process.

There are still many issues that stand on the way of near-to-perfect GAN training, among which mode collapse, evaluation, an absence of stopping criteria in practice, unstable optimization, etc.

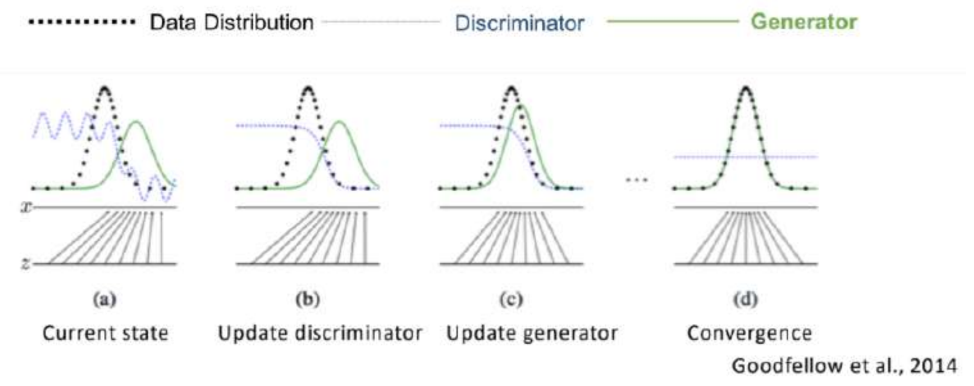
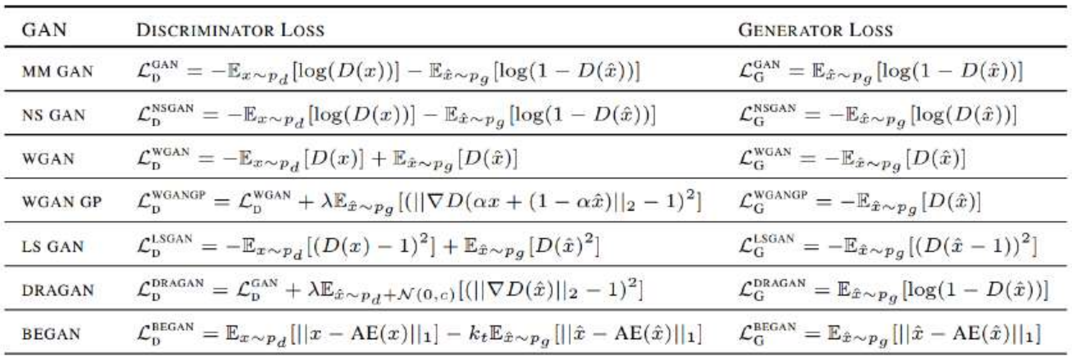


FIGURE 2.4: Simultaneous training of GANs. source: [CS236]

There is still room for creativity and improvement regarding the choice of the objective function. Lucic et al., 2017 conducted research on comparison and evaluation of different GANs configurations, here are the most popular objectives (see Fig. 2.6):



GAN	DISCRIMINATOR LOSS	GENERATOR LOSS
MM GAN	$\mathcal{L}_{\text{D}}^{\text{GAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_{\text{G}}^{\text{GAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
NS GAN	$\mathcal{L}_{\text{D}}^{\text{NSGAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_{\text{G}}^{\text{NSGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_{\text{D}}^{\text{WGAN}} = -\mathbb{E}_{x\sim p_d}[D(x)] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$	$\mathcal{L}_{\text{G}}^{\text{WGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
WGAN GP	$\mathcal{L}_{\text{D}}^{\text{WGANGP}} = \mathcal{L}_{\text{D}}^{\text{WGAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_g}[(\|\nabla D(\alpha x + (1-\alpha\hat{x})\|_2 - 1)^2]$	$\mathcal{L}_{\text{G}}^{\text{WGANGP}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
LS GAN	$\mathcal{L}_{\text{D}}^{\text{LSGAN}} = -\mathbb{E}_{x\sim p_d}[(D(x)-1)^2] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})^2]$	$\mathcal{L}_{\text{G}}^{\text{LSGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[(D(\hat{x}-1))^2]$
DRAGAN	$\mathcal{L}_{\text{D}}^{\text{DRAGAN}} = \mathcal{L}_{\text{D}}^{\text{GAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_d+\mathcal{N}(0,c)}[(\|\nabla D(\hat{x})\|_2 - 1)^2]$	$\mathcal{L}_{\text{G}}^{\text{DRAGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
BEGAN	$\mathcal{L}_{\text{D}}^{\text{BEGAN}} = \mathbb{E}_{x\sim p_d}[\|x - \text{AE}(x)\|_1] - k_t\mathbb{E}_{\hat{x}\sim p_g}[\|\hat{x} - \text{AE}(\hat{x})\|_1]$	$\mathcal{L}_{\text{G}}^{\text{BEGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\|\hat{x} - \text{AE}(\hat{x})\|_1]$

FIGURE 2.5: Generator and discriminator loss functions. source: [Lucic et al., 2017]

WGAN and WGAN-GP

If one follows the progress of the GANs, they might notice that the key things that have influenced improvements in the results were not only the new architectures but also the introduction of different loss functions.

The loss function that Goodfellow et al., 2014 use for training the discriminator is known as the *Kullback–Leibler divergence*:

$$KL(\mathbb{P}_r||\mathbb{P}_\theta) = \int \log\left(\frac{P_r(x)}{P_\theta(x)}\right) P_r(x)d\mu$$

Here $\mathbb{P}_r, \mathbb{P}_\theta \in Prob(X)$ – two distributions, where $Prob(X)$ denotes the space of probability measures defined on a compact metric space X.

This loss function has several disadvantages that lead to the instability of the GAN training, one of which is non-differentiability, and even discontinuity, for some distributions with respect to the parameter θ – weights of the neural network G.

In [Arjovsky, Chintala, and Bottou, 2017] the authors suggest using different function called the *Earth-Mover (EM) distance* or *Wasserstein-1*

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \inf_{\gamma\in\Pi(\mathbb{P}_r,\mathbb{P}_\theta)} \mathbb{E}_{(x,y)\sim\gamma}[||x-y||]$$

Here $\Pi(\mathbb{P}_r, \mathbb{P}_\theta)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are $\mathbb{P}_r$ and $\mathbb{P}_\theta$ respectively. It is called the "earth-mover" because $\gamma(x, y)$ indicates how much "mass" must be transported from x to y to transform $\mathbb{P}_r$ into $\mathbb{P}_\theta$ in the continuous probability space. The Wasserstein distance can provide a smooth measure even for the cases when other metrics fail (for instance, when the two distributions are located in lower dimensional manifolds without overlaps, which was illustrated by the trivial example in the paper), so it helps to maintain a stable learning process using gradient descents. One may use the *Kantorovich-Rubinstein duality* to rewrite the formula so it becomes more approachable:

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)]$$

Here the supremum is taken over *1-Lipschitz* functions $f : X \to \mathbb{R}$. Even though WGAN became a breakthrough, the suggested approach was still not stable enough, and the reason for that was the *weight clipping* used to enforce a Lipschitz constraint for the discriminator (the authors even encouraged further investigation as were aware that the weight clipping was a "clearly terrible" approach). Weight clipping is simply clamping the weights to a fixed interval after each gradient update so they lie in a compact space, guaranteeing f to be 1-Lipschitz.

The authors of [Gulrajani et al., 2017] suggested to penalize the norm of gradient of the discriminator with respect to its input to avoid undesired clipping. Their objective function now consists of two parts: original discriminator loss and the gradient penalty (as we need the function to be 1-Lipschitz):

$$L = -\mathbb{E}_{x \sim \mathbb{P}_r}[D(x)] + \mathbb{E}_{x \sim \mathbb{P}_\theta}[D(G(x))] + \lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}\left[\left(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1\right)^2\right]$$

Here $\mathbb{P}_{\hat{x}}$ is defined implicitly – sampling uniformly along straight lines between pairs of points sampled from the real data distribution $\mathbb{P}_r$ and the hallucinated data distribution $\mathbb{P}_\theta$.

cGANs

Conditional GANs [Isola et al., 2016] are an extension of GANs for multimodal inputs. They add extra information to both generator and discriminator to train both networks better. Unlike regular GANs when the generator deals with a random noise vector, here it receives some constraint of any kind, in case of image-to-image translation - an image.

2.1.3 FPN

Feature pyramids are a basic component in recognition systems for detecting objects at different scales. They were heavily used in the era of hand-engineered features. The principle advantage of featurizing each level of an image pyramid is that it produces a multi-scale feature representation in which all levels are semantically strong, including the high-resolution levels. [Lin et al., 2016] FPN uses a top-down architecture with lateral connections to build an in-network feature pyramid from a single-scale input. The bottom-up pathway is the feedforward computation of the backbone ConvNet, which computes a feature hierarchy consisting of feature maps at several scales with a scaling step of 2. [Xu et al., 2018] The network stage defines as the layers producing output maps of the same size. The output of the last

layer of each stage is reference set of feature maps, which will enrich to create pyramid, because the deepest layer of each stage should have the strongest features. The top-down pathway hallucinates higher resolution features by upsampling spatially coarser, but semantically stronger, feature maps from higher pyramid levels. These features are then enhanced with features from the bottom-up pathway via lateral connections. Each lateral connection merges feature maps of the same spatial size from the bottom-up pathway and the top-down pathway. This process is independent of the backbone convolutional architectures.
This method showed significant improvements over several strong baselines and competition winners.

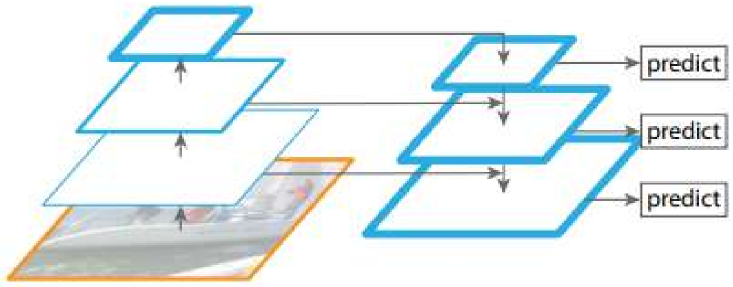


FIGURE 2.6: Feature pyramid network. source: [Lin et al., 2016]

Chapter 3

Proposed approach

Our primary goal was to create an efficient pipeline for general image restoration: given several images $\tilde{g}_1, ..., \tilde{g}_m$ with m different types of degradations as an input, with no prior knowledge or other constraints regarding the nature of degradation, to provide the recovered images $D_i(E(\tilde{g}_i)), i = 1...m$ as close as possible to the real-data distributions p_{g_i}, where E is a generic encoder and D_i are corresponding decoders. Also, during training, we introduced separate discriminator networks for each encoder-decoder branch and trained the whole setting in an adversarial manner.

3.1 Pipeline

Multi-task learning (MTL) is a paradigm for solving domain-related tasks via shared representations. [Ruder, 2017] One can use what is learned for one problem to help in solving others in parallel, utilising the extra information about the domain contained in the training samples of the related tasks.

Such a technique allows embeddings developed in the hidden layers for one task to be used by the rest of the tasks. It also allows features to appear in such a way which was not possible in any single-task net, trained separately. Importantly, MTL-backpropagation also allows some hidden units to become specialised for just one or a few tasks; other tasks can ignore features they think of being redundant by keeping the weights connected to them close to zero. [Caruana, 1997]

Our setting has an encoder-decoder-like architecture (see Fig. 3.1) with a generic encoder (the part of the network where the weights are shared) and separate decoders for each image restoration task in the pipeline. The images are processed in batches, and it is important that each forward-backward cycle 'sees' images from all the datasets; otherwise the network will substitute previously acquired knowledge with the last-seen task information only, forgetting the rest of them. We make forward passes for each task-specific batch separately one-by-one, accumulating the generator loss, and only then backpropagate it, when all m batches have passed through the network. The decoders share the same architecture, though learn different parameters during the training. Every single image has its predefined decoder; the gradient backpropagates through the corresponding connections of the network. That is how we learn shared low-level features.

We also use separate discriminators with the same architectures for each branch so that they are concentrated only on single real-data distribution.

Another advantage of this approach is that it is flexible to the choice of the generator architecture: one can use any reasonable architecture that works well on one of the tasks and doesn't have task-specific constraints, and split it into two parts, imitating encoder and decoder.

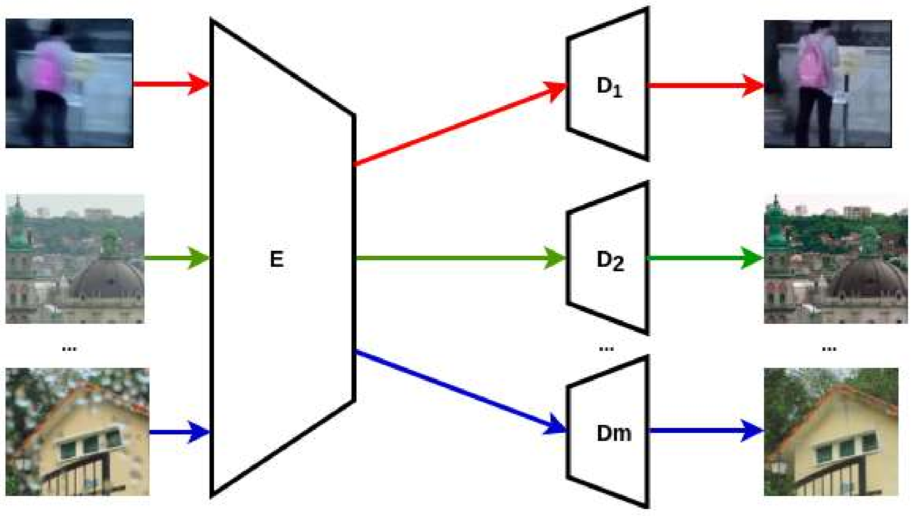


FIGURE 3.1: Generic encoder and separate decoders pipeline

It is important to emphasise that this approach is not only valuable in the context of achieving better results for the tasks with massive rich datasets, but it also helps a lot with the degradation tasks where it is challenging to collect many image pairs, so the procedure can benefit from such an encoder trained on a diverse large-scale dataset.

While the encoder helps to learn generic low-level features that show the common background for the tasks, the decoders, on the contrary, help to learn task-specific features, as still, the types of degradation have lots of differences in visual and perceptual representation.

3.2 Architecture

We use ResNet-like architecture used in [Kupyn et al., 2017] for the baseline experiments.

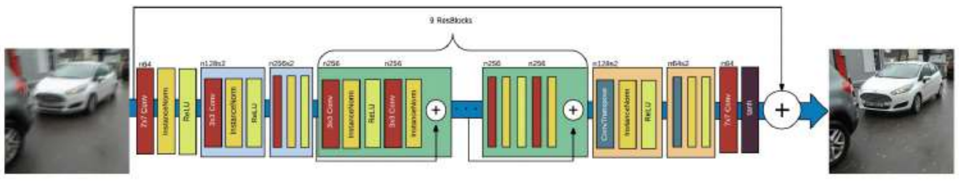


FIGURE 3.2: DeblurGAN generator architecture [Kupyn et al., 2017]

We split it in two as follows: 7×7 convolutional layer, 2 strided convolutions, and 9 residual blocks [He et al., 2015] go to the encoder E, while the rest - 2 transposed convolutional blocks and 7×7 convolutional layer - go to the corresponding decoders D_i.

The ResNet-like architecture proposed in DeblurGAN still had lots of disadvantages and room for improvement. First of all, it could be optimized a lot in depth. Secondly, it worked poorly with blur at different scales. That is why we decided to use the pyramid-like architecture as another option for this task and present in the work results of both experimental settings.

For the feature extraction part we use feature-pyramid network [Lin et al., 2016] with InceptionResNetV2 backbone [Szegedy, Ioffe, and Vanhoucke, 2016], which outputs multi-scale feature maps $M_0, ..., M_4$, that are further processed as is shown in Fig. 3.3.

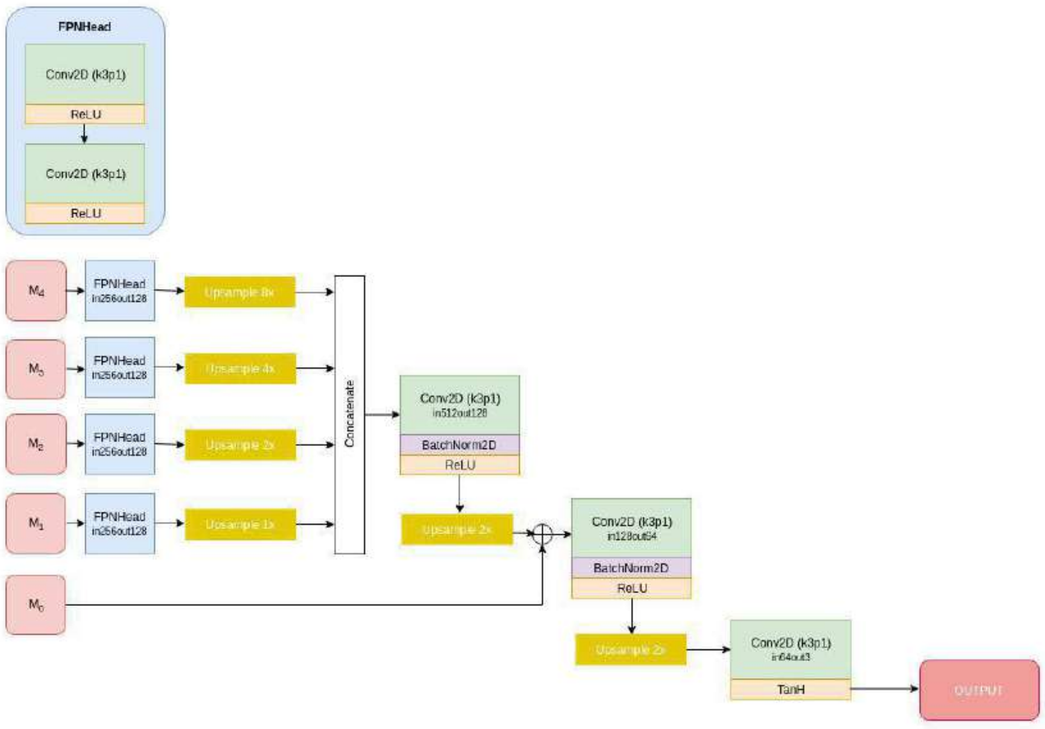


FIGURE 3.3: FPN Inception generator architecture

All the blocks prior to the "concatenation" block are treated as the encoder part E, while the rest is treated as the decoder part D_i. As one of the contributions of DeblurGAN [Kupyn et al., 2017] was extra skip-connection that helped to achieve better results, we do not give up on this improvement in the multi-task setting as well, as skip connections were useful for MLT in recent work of Ruder et al., 2017. As you may see in the Experiment section (here goes link to the figure), skip-connections give a significant boost in PSNR.

We do not change the discriminator architecture used in [Kupyn et al., 2017]: it is WGAN-GP discriminator identical to PatchGAN [Isola et al., 2016].

3.3 Loss functions

We will elaborate only on the generator loss here, as WGAN-GP discriminator loss was already discussed in Chapter 2. In the basic setting, we set the gradient penalty constant to be equal to 10.

In the problems where we deal with restoring the images, we need to find a proper way to compare the images on the training stage – the reconstructed and the original ones. One of the components for the loss function of the generator has to be the so-called “content” loss L_X. The simplest approach would be to use either L_1 or L_2 distance as L_X. Obviously, they lead to bad results as they take into account the raw pixel data and inherently tend to push the network to generate blurry outputs (the blur accumulates). Using L_2 (as well as L_1) works really poorly with the multi-modal distributions. [Mathieu, Couprie, and LeCun, 2015] If you have two equally likely modes for output (the pixel intensity), the average value of those two gives you an even smaller value of the L_2 objective, thus forces the network to stick to that unlikely though technically "better" option.

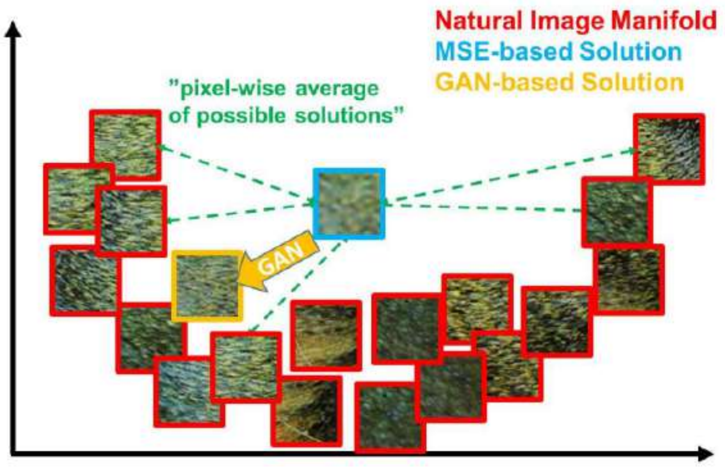


FIGURE 3.4: Illustration of patches from the real-data manifold (red) and SR-patches with MSE (blue) and GAN (orange) objective. The MSE-based solution appears overly smooth due to the pixel-wise average of possible solutions in the pixel space, while GAN pushes the output towards the natural image manifold producing sharper and more realistic solutions. [Ledig et al., 2016]

In [Kupyn et al., 2017] the perceptual loss [Johnson, Alahi, and Li, 2016] is used instead. It is similar to the L_2, but deals with the images' VGG-19 [Simonyan and Zisserman, 2014] feature representations, namely, *conv3_3* feature maps. It is defined as follows:

$$L_X = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} \left(\phi_{i,j}\big(g(x)\big)_{x,y} - \phi_{i,j}\big(G(\tilde{g}(x))\big)_{x,y} \right)^2$$

where $\phi_{i,j}$ denotes the feature map before the i-th maxpooling by the j-th convolution within the VGG-19 network pretrained on ImageNet [Deng et al., 2009]. The feature maps are usually taken after activation, though Wang et al., 2018 claim that the feature maps before activation preserve more semantics, and this is a valid point to build new experiments upon for further exploration.

So, the basic setting for the experiments had 3 loss components: the VGG-based perceptual loss [Johnson, Alahi, and Li, 2016] for the general content, the L_1 loss for preserving colors, and the adversarial loss for the texture details.

$$L_G = L_X + 0.5 * L_1 + 0.01 * L_{adv}$$

It is worth mentioning that the discriminator loss also was multiplied by 0.01 so the updates would have been on the same scale.

The issue with the perceptual loss is that it doesn't take into account the semantics of the image. It is highly noticeable in the style transfer [Gatys, Ecker, and Bethge, 2016] problems.

For instance,

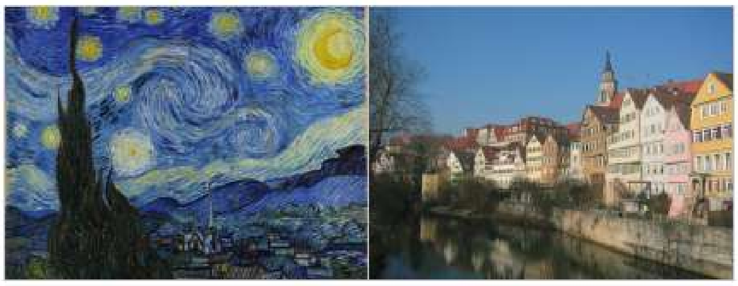


FIGURE 3.5

leads to

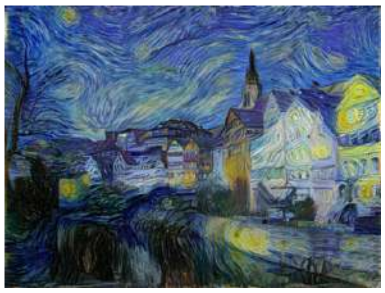


FIGURE 3.6

which is clearly "non-Van Gogh", as the yellow sun spots appear not only in the sky, but also on the river bank, and some of the buildings are weirdly corrupted, though on the original Van Gogh picture they seem to be perfectly OK.

[Mechrez et al., 2018] suggested loss function that cares about the image semantics – Contextual loss:

$$L_{CX} = -\log\left(\frac{1}{N}\sum_j \max_j A_{ij}\right)$$

Here A_{ij} stands for the affinity between features x_i and y_j. Affinity is the function that describes the importance of the geometry of the image for the feature distribution and has the following formula:

$$A_{ij} = \frac{exp\left(1 - \frac{\tilde{d_{ij}}}{h}\right)}{\sum_l exp\left(1 - \frac{\tilde{d_{il}}}{h}\right)}$$

Here $h > 0$ is a bandwidth parameter, and $\tilde{d_{ij}}$ is the normalized cosine distance between x_i and y_j. The authors, though, use not the abundantly discussed above Wasserstein-1 distance, but stick to more classic KL-Divergence. Hence, we came into conclusion that the combination of the two previously introduced functions – the Earth-Mover distance and the Contextual loss – might show even better results, both dealing with the semantics and smoothness for stable learning. In Chapter 4 we show the results of the experiment with additional component $0.5L_{CX}$ of the generator loss.

Chapter 4

Experiments

4.1 Datasets

4.1.1 GORPO Dataset

GOPRO Dataset [Nah, Kim, and Lee, 2016] consists of 3214 pairs of blurry and sharp images, 2103 pairs in the training set and 1111 pair in the test set. It was generated from the 240 FPS videos via GOPRO4 Hero Black camera, blurry images by averaging different number (7-13) of the successive frames, so the strength of blur does not follow the same pattern. In other words, that means that the sharp image is taken at a shutter speed equal to 1/240, while the blurry one simulates the photo taken at a shutter speed equal to n/240, where n is the number of successive latent frames used. The pairs are aligned in such a manner that the sharp image is the mid-frame among those used for creating the blurry one.

4.1.2 GOPRO 3840 FPS Dataset

GOPRO 3840 FPS Dataset was created based on the GOPRO Dataset via interpolation method introduced in [Niklaus, Mai, and Liu, 2017b]. Given two frames, it uses adaptive convolution [Niklaus, Mai, and Liu, 2017a] in a spatially-adaptive separable manner to interpolate the intermediate frame. It generates the output pixels for frame synthesis approximating the 2D convolution kernels with a pair of 1D kernels, horizontal and vertical. The method employs a fully convolutional neural network [Long, Shelhamer, and Darrell, 2014] that produces the separable kernels for all output pixels at once. The important thing here is that as the output is full-frame, one can use perceptual loss functions [Johnson, Alahi, and Li, 2016] to improve the perceptual quality of the results. The loss that was used for this method was based on the *relu4_4* layer of the VGG-19 network [Simonyan and Zisserman, 2014]. Besides GOPRO 3840 FPS Dataset there were also generated two other datasets, namely 960 FPS and 1920 FPS, but we used the one with the highest FPS rate for our experiments. The blurred images obtained via frame interpolation and averaging tend to be smoother and possess fewer artifacts, making the input image more realistic.

4.1.3 D-HAZY Dataset

D-HAZY [Cosmin Ancuti, 2016] is a dataset of 1472 pairs of clear and hazy images of real indoor scenes composed from the Middleburry [Scharstein et al., 2014] (23 pairs) and the NYU-Depth V2 [Nathan Silberman and Fergus, 2012] (1449 pairs) datasets. The haze on the images is synthesized via the corresponding depth maps using the physical model of a hazy medium. We use only the NYU-Depth V2 part of the D-HAZY dataset, splitting it in 1000 training and 449 testing pairs randomly.

4.1.4 Raindrop Dataset

Raindrop Dataset [Qian et al., 2017] consists of 1110 pairs of "rainy" and clear images with various background scenes. Those were obtained using two different cameras, namely Sony A6000 and Canon EOS 60, and two pieces of the same 3mm glass attached to the camera lenses, sprayed with water and clean. The diversity in the raindrops is achieved varying the distance from the lens to the glass from 2 to 5 cm. The dataset was already divided into *train* (861 pairs), *test_b* (249 pairs), and *test_a* (58 pairs, randomly selected subset of *test_b*) subsets; we used *test_b* subset as a test set.

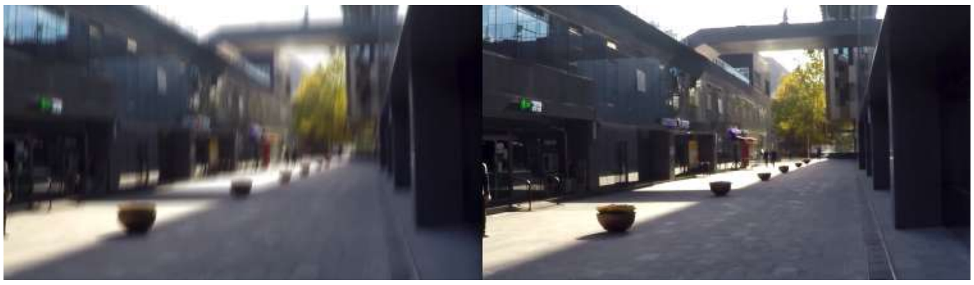


FIGURE 4.1: Example of a blurry-sharp pair of images from GOPRO 3840 FPS Dataset test set

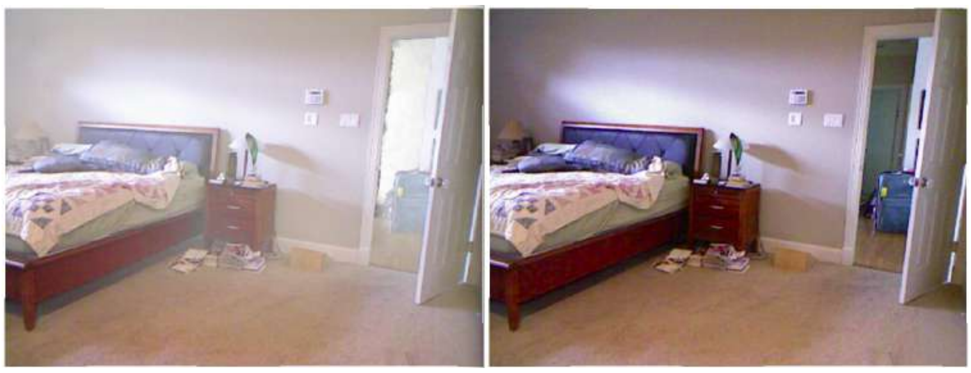


FIGURE 4.2: Example of a hazy-clear pair of images from D-HAZY test set

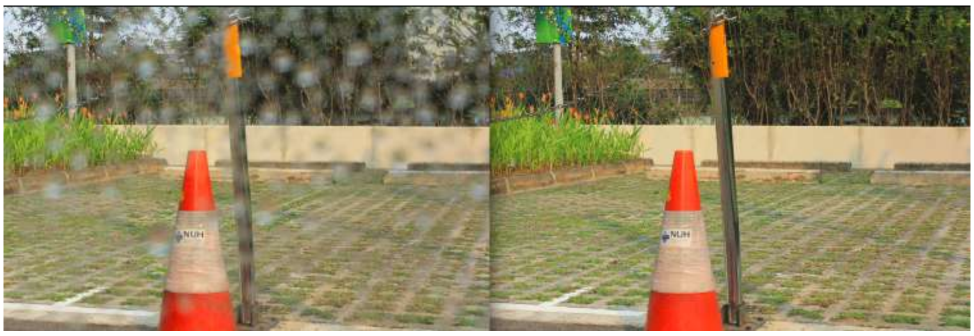


FIGURE 4.3: Example of a rainy-clear pair of images from Raindrop Dataset test set

4.2 Metrics

Image restoration algorithms are typically evaluated by some distortion measures, e.g. PSNR, SSIM, or by subjective evaluation of human observers that quantify perceived perceptual quality [Blau and Michaeli, 2017].

Intuitively, distortion measure is a criterion that measures "quality" of image restoration [Becker, 2000]. More formally, it is some real-valued nonnegative function of two arguments g and $\tilde{g}$ that represents the "accuracy from g to $\tilde{g}$".

The last decades witnessed tremendous progress in developing image restoration algorithms of many kinds, both in visual (perceptual) quality and in common-used distortion measures. As one of the main problems in this domain is ill-posedness of the tasks, sometimes it is unclear what should the community consider to be the best result or SOTA. Different challenges, such as NTIRE [*NTIRE 2019*] and PIRM [*PIRM 2018*], have separate tracks for evaluation of the model performance, focused either on perceptual quality or on the widely used distortion metrics. Indeed, lately, the improvement in visual quality could not always catch up with improvement in reconstruction accuracy, as was shown in [Ledig et al., 2016] where the authors conducted extensive mean-opinion test to prove their point. One of the reasons for that is that the approaches based on pixel-wise differences, such as PSNR, tend to output over-smoothed textures without sufficient high-frequency content details, causing fundamental disagreement with the human opinion scores. Even if one finds a 'perfect' metric, there is no guarantee that we can directly optimize for it, because it also should be differentiable.

The perceptually-oriented criteria are often no-reference, meaning that the quality of the output is measured without depending on the reference image, but based on estimating deviations from natural image statistics. Among such are the Ma's score [Ma et al., 2016] and NIQE [Mittal, and Bovik, 2012], a linear combination of which was used as a metric for PIRM 2018 challenge, as it has shown the highest Spearman correlation coefficient (0.83) with the human-opinion scores, while the SSIM and RMSE measures were anti-correlated [Blau et al., 2018, Fig. 9]. Even more unusual one, the evaluation method used in [Kupyn et al., 2017] measured the quality of deblurring algorithm based on results of object detection on a pretrained YOLO [Redmon et al., 2015] network.

Finally, in [Blau and Michaeli, 2017] the authors claim that there is a certain trade-off between perception and distortion, and it exists for all distortion measures (see Fig. 4.4).

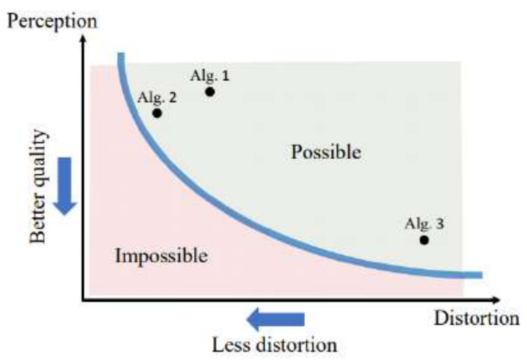


FIGURE 4.4: There exists a region in the perception-distortion plane which cannot be attained, regardless of the algorithmic scheme. ([Blau and Michaeli, 2017])

Even though the PSNR and SSIM are not perceptually best criteria, those are the widely-used ones in the majority of image restoration tasks, and to compare our method to the SOTA ones, we will use those two measures for evaluation alongside with providing visual results. Moreover, these are appealing because they are mathematically convenient in the context of optimization.

4.2.1 PSNR

The PSNR quantifies how faithful is the distorted image to the original one [Fardo et al., 2016]. For grey-scale images,

$$PSNR = 20 \log_{10} \frac{MAX_g}{\sqrt{MSE}}$$

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [g(i,j) - \tilde{g}(i,j)]^2$$

where MAX_g is the highest possible pixel value, g and $\tilde{g}$ are the two images.

PSNR is measured in decibels (dB). The higher PSNR - the better. If the pixels are encoded using 8 bits, $MAX_g = 255$. For color images with three channels, the formula for PSNR is the same except the sum is taken over all squared value differences and is divided by an additional factor of three. For color images with luminance component (such as YCbCr), only that component is used.

Unlike MSE, PSNR does not strongly depend on the image intensity scaling as it scales the MSE according to the image range. PSNR is an adequate measure of image restoration for the same image, but the comparison of PSNR between the images makes no sense.

4.2.2 SSIM

SSIM index is based on the similarities of luminance, contrast and structure between local patches **x** and **y** extracted from an original and a corrupted image:

$$l(\mathbf{x},\mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

$$c(\mathbf{x},\mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

$$s(\mathbf{x},\mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}$$

where μ, σ and σ_{xy} are the mean, standard deviation and covariance of the image patches, respectively, and C_1, C_2, $C_3 > 0$ are stability constants that prevent the fraction from exploding when the denominator approaches zero [Wang et al., 2004]. The local SSIM index is then obtained as

$$SSIM(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^\alpha [c(\mathbf{x},\mathbf{y})]^\beta [s(\mathbf{x},\mathbf{y})]^\gamma$$

where $\alpha, \beta, \gamma > 0$ stand for the relative importance for each of the three factors. The last term here is responsible for the structural distortions and the first two terms - for non-structural ones, so SSIM is capable of differentiating between them as the components are relatively independent. This equation, though, does not take into account the distance to the observer. Therefore, the SSIM index depends on the scale it is applied to. A multi-scale index derived from SSIM is called Multi-Scale Structural Similarity, a.k.a. MS-SSIM.

The SSIM index is usually simplified by taking $\alpha = \beta = 1$, and $C_3 = C_2/2$. The above equation then reduces to

$$SSIM(\mathbf{x}, \mathbf{y}) = \left(\frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}\right)\left(\frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}\right)$$

The SSIM index of the image then is calculated by (weighted) averaging the SSIM indices of image patches via the sliding window. The motivation behind that is that the image statistical features are usually highly spatially non-stationary, and image distortions may also be space-variant.

The last formula is usually applied only on the luminance component, though it may also be applied to color or chromatic values. The range of SSIM function is $[-1; 1]$, the maximum value reachable only for two identical images. 0 stands for no structural similarity.

Although not perfect as a perceptual image quality metric, it still outperforms MSE and its derivatives [Rehman, 2013].

4.3 Training and evaluation details

We implemented all of our models using *PyTorch* [*PyTorch*]. The training was performed on GeForce® GTX 1080 GPU and NVIDIA Tesla P100 GPU. The models were trained on random crops of size 256x256. As all the models are fully convolutional and are trained on image patches, they can be applied to images of arbitrary size. We use Adam optimizer [Kingma and Ba, 2014] and perform several experiments varying the ratio of gradient descent steps for the discriminator and generator in Wasserstein-GAN fashion [Arjovsky, Chintala, and Bottou, 2017]. We also vary the ratio between the learning rates for the generator and the discriminator. The learning rate for the generator remains unchanged and is set to 0.0001. The learning rate for discriminator is the same if not stated otherwise. After the first 40 epochs, we linearly decay the learning rate to $1e - 7$ over the next 40 epochs. We train the deblurring branch of the generator with the batch size 2 and other two branches with the batch size 1.

The SSIM evaluation was performed with skimage [*scikit-image*] library, averaging the value per each channel. The PSNR measure also treated the images as RGB 3-channel ones.

4.4 Results

Results on image deblurring via FPN Inception network are in Table 4.1. We perform single-task training, deblurring+dehazing training, and 3-tasks-training.

TABLE 4.1: PSNR and SSIM, mean values over the GoPro test dataset (deblurring task).

	Nah, Kim, and Lee, 2016	Xu, Zheng, and Jia, 2013	Kupyn et al., 2017	Ours (FPN)		
Metric				*1 task*	*2 tasks*	*3 tasks*
PSNR	**28.3**	25.1	27.2	28.277	27.3	27.38
SSIM	0.916	0.89	**0.954**	0.928	0.81	0.8

Results of ResNet-like architecture training in different settings: with discriminator learning rate equal to 0.0002, with additional contextual loss component, and with WGAN-like discriminator update ratio equal to 5.

TABLE 4.2: PSNR and SSIM, mean values over the corresponding datasets.

	Metric	$lr_D = 2lr_G$	$+0.5cx$	$lr_D = lr_G$ $D_{upd} = 5$
deblur	PSNR	27.13	26.56	26.85
	SSIM	0.791	0.779	0.78
dehaze	PSNR	16.92	16.92	16.6
	SSIM	0.695	0.718	0.699
derain	PSNR	23.98	23.96	23.97
	SSIM	0.765	0.756	0.761

Here are some image results:

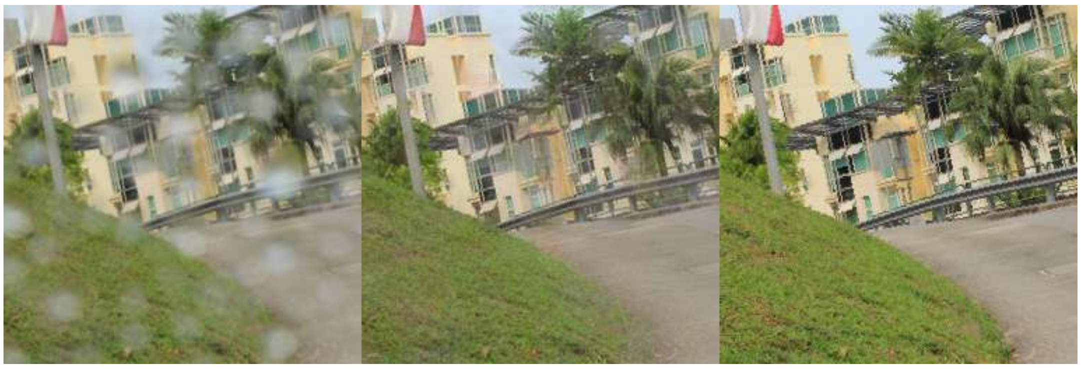


FIGURE 4.5: FPN Inception trained on 3 tasks deals with raindrops. Left to right: input, output, ground truth.

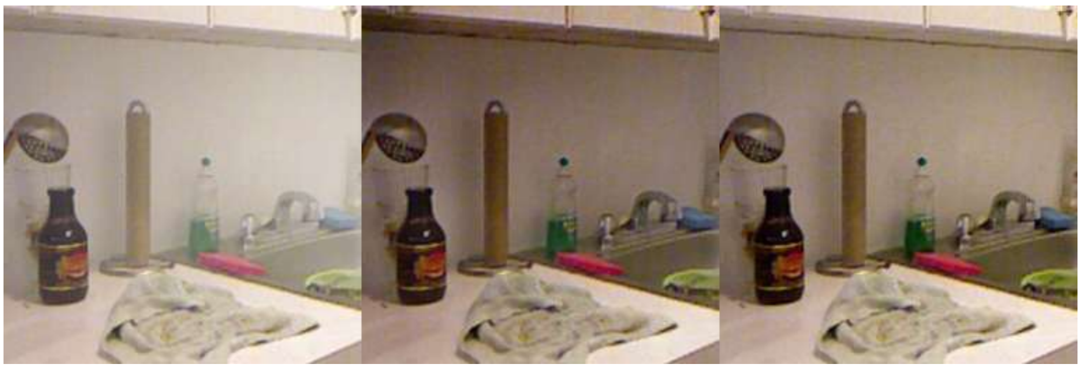


FIGURE 4.6: FPN Inception trained on 3 tasks deals with haze. Left to right: input, output, ground truth.

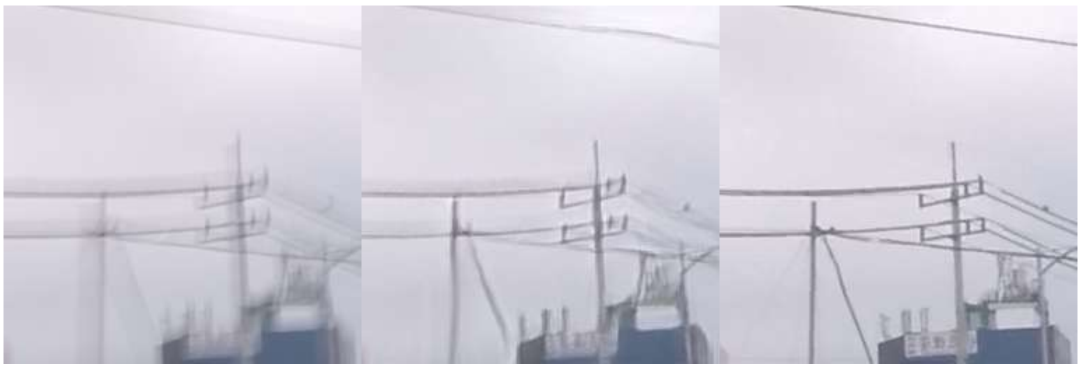


FIGURE 4.7: FPN Inception trained on 3 tasks deals with blur. Left to right: input, output, ground truth.

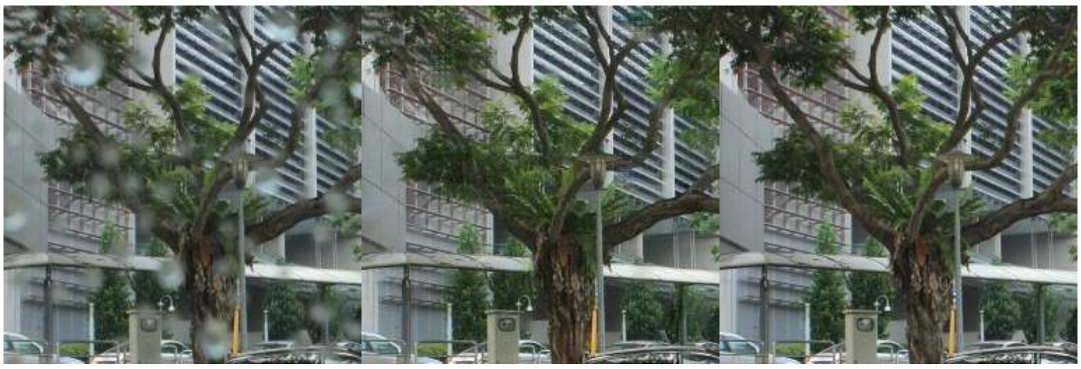


FIGURE 4.8: ResNet-like generator trained on 3 tasks with contextual loss deals with raindrops. Left to right: input, output, ground truth.

Chapter 5

Conclusion

We described an end-to-end multi-task learning pipeline for image restoration based on GANs training. The approach is promising due to its flexibility and to the fact that it contributes to the extremely fast-emerging field of deep learning - image-to-image translation via GANs. The new FPN-inspired architecture shows results very close to SOTA, though, there is still room for improvement and new experiments regarding the training procedure, objective functions, and network architectures.

The code is available on *github*.

Bibliography

In:

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). “Wasserstein GAN”. In: URL: https://arxiv.org/pdf/1701.07875.pdf.

Becker, Axel (2000). “A review on image distortion measures”. In: *Fraunhofer - ITWM*. URL: https://kluedo.ub.uni-kl.de/frontdoor/deliver/index/docId/1472/file/bericht19.pdf.

Blau, Yochai and Tomer Michaeli (2017). “The Perception-Distortion Tradeoff”. In: *CoRR* abs/1711.06077. arXiv: 1711.06077. URL: http://arxiv.org/abs/1711.06077.

Blau, Yochai et al. (2018). “2018 PIRM Challenge on Perceptual Image Super-resolution”. In: *CoRR* abs/1809.07517. arXiv: 1809.07517. URL: http://arxiv.org/abs/1809.07517.

Bulat, Adrian, Jing Yang, and Georgios Tzimiropoulos (2018). “To learn image super-resolution, use a GAN to learn how to do image degradation first”. In: *CoRR* abs/1807.11458. arXiv: 1807.11458. URL: http://arxiv.org/abs/1807.11458.

Caruana, Rich (1997). “Multitask Learning”. In: *Machine Learning* 28.1, pp. 41–75. ISSN: 1573-0565. DOI: 10.1023/A:1007379606734. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.121.8707&rep=rep1&type=pdf.

Chen, Jingwen et al. (2018). “Image Blind Denoising With Generative Adversarial Network Based Noise Modeling”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. URL: http://openaccess.thecvf.com/content_cvpr_2018/papers/Chen_Image_Blind_Denoising_CVPR_2018_paper.pdf.

Cosmin Ancuti Codruta O. Ancuti, Christophe De Vleeschouwer (2016). “D-HAZY: A DATASET TO EVALUATE QUANTITATIVELY DEHAZING ALGORITHMS”. In: *IEEE International Conference on Image Processing (ICIP)*. ICIP’16. Pheonix, USA. URL: https://ieeexplore.ieee.org/document/7532754/.

CS236. https://deepgenerativemodels.github.io/.

Deng, J. et al. (2009). “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

Dong, Weisheng et al. (2018). “Joint Demosaicing and Denoising with Perceptual Optimization on a Generative Adversarial Network”. In: *CoRR* abs/1802.04723. arXiv: 1802.04723. URL: http://arxiv.org/abs/1802.04723.

Du, Yixin and Xin Li (2018). “Perceptually Optimized Generative Adversarial Network for Single Image Dehazing”. In: *CoRR* abs/1805.01084. arXiv: 1805.01084. URL: http://arxiv.org/abs/1805.01084.

Engin, Deniz, Anil Genç, and Hazim Kemal Ekenel (2018). “Cycle-Dehaze: Enhanced CycleGAN for Single Image Dehazing”. In: *CoRR* abs/1805.05308. arXiv: 1805.05308. URL: http://arxiv.org/abs/1805.05308.

Fabbri, Cameron, Md Jahidul Islam, and Junaed Sattar (2018). “Enhancing Underwater Imagery using Generative Adversarial Networks”. In: *CoRR* abs/1801.04011. arXiv: 1801.04011. URL: http://arxiv.org/abs/1801.04011.

Fardo, Fernando A. et al. (2016). "A Formal Evaluation of PSNR as Quality Measurement Parameter for Image Segmentation Algorithms". In: *CoRR* abs/1605.07116. arXiv: 1605.07116. URL: http://arxiv.org/abs/1605.07116.

Gatys, L. A., A. S. Ecker, and M. Bethge (2016). "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423. DOI: 10.1109/CVPR.2016.265.

github. https://github.com/t-martyniuk/MTL_image_restoration.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. The MIT Press. ISBN: 0262035618, 9780262035613.

Goodfellow, Ian J. et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.

Gulrajani, Ishaan et al. (2017). "Improved Training of Wasserstein GANs". In: *CoRR* abs/1704.00028. arXiv: 1704.00028. URL: http://arxiv.org/abs/1704.00028.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385. arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

ILSVRC. http://www.image-net.org/challenges/LSVRC/.

Image source. https://www.oreilly.com/library/view/getting-started-with/9781786468574/ch04s04.html.

Isola, Phillip et al. (2016). "Image-to-Image Translation with Conditional Adversarial Networks". In: *CoRR* abs/1611.07004. arXiv: 1611.07004. URL: http://arxiv.org/abs/1611.07004.

Johnson, Justin, Alexandre Alahi, and Fei-Fei Li (2016). "Perceptual Losses for Real-Time Style Transfer and Super-Resolution". In: *CoRR* abs/1603.08155. arXiv: 1603.08155. URL: http://arxiv.org/abs/1603.08155.

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980. arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980.

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114. arXiv: 1312.6114. URL: http://arxiv.org/abs/1312.6114.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Kupyn, Orest et al. (2017). "DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks". In: *CoRR* abs/1711.07064. arXiv: 1711.07064. URL: http://arxiv.org/abs/1711.07064.

Lecun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE*, pp. 2278–2324.

Ledig, Christian et al. (2016). "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *CoRR* abs/1609.04802. arXiv: 1609.04802. URL: http://arxiv.org/abs/1609.04802.

Li, Runde et al. (2018). "Single Image Dehazing via Conditional Generative Adversarial Network". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. URL: http://openaccess.thecvf.com/content_cvpr_2018/papers/Li_Single_Image_Dehazing_CVPR_2018_paper.pdf.

Lin, Tsung-Yi et al. (2016). "Feature Pyramid Networks for Object Detection". In: *CoRR* abs/1612.03144. arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2014). "Fully Convolutional Networks for Semantic Segmentation". In: *CoRR* abs/1411.4038. arXiv: 1411.4038. URL: http://arxiv.org/abs/1411.4038.

Lucic, Mario et al. (2017). "Are GANs Created Equal? A Large-Scale Study". In: *CoRR* abs/1711.10337.

Ma, Chao et al. (2016). "Learning a No-Reference Quality Metric for Single-Image Super-Resolution". In: *CoRR* abs/1612.05890. arXiv: 1612.05890. URL: http://arxiv.org/abs/1612.05890.

Maas, Andrew L. (2013). "Rectifier Nonlinearities Improve Neural Network Acoustic Models". In:

Madam Nimisha, Thekke, Kumar Sunil, and A. N. Rajagopalan (2018). "Unsupervised Class-Specific Deblurring". In: *The European Conference on Computer Vision (ECCV)*. URL: http://openaccess.thecvf.com/content_ECCV_2018/papers/Nimisha_T_M_Unsupervised_Class-Specific_Deblurring_ECCV_2018_paper.pdf.

Mathieu, Michaël, Camille Couprie, and Yann LeCun (2015). "Deep multi-scale video prediction beyond mean square error". In: *CoRR* abs/1511.05440. arXiv: 1511.05440. URL: http://arxiv.org/abs/1511.05440.

Mechrez, Roey, Itamar Talmi, and Lihi Zelnik-Manor (2018). "The Contextual Loss for Image Transformation with Non-Aligned Data". In: *CoRR* abs/1803.02077. arXiv: 1803.02077. URL: http://arxiv.org/abs/1803.02077.

Mechrez, Roey et al. (2018). "Learning to Maintain Natural Image Statistics". In: *CoRR* abs/1803.04626. arXiv: 1803.04626. URL: http://arxiv.org/abs/1803.04626.

Mittal, A, Soundarajan, and Alan Bovik (2012). "Making a "completely blind" image quality analyzer". In: *IEEE Signal Processing letters* 20, pp. 209–212. URL: https://www.researchgate.net/publication/308158013_Making_a_completely_blind_image_quality_analyzer.

N., Bharath Raj and Venkateswaran N (2018). "Single Image Haze Removal using a Generative Adversarial Network". In: *CoRR* abs/1810.09479. arXiv: 1810.09479. URL: http://arxiv.org/abs/1810.09479.

Nah, Seungjun, Tae Hyun Kim, and Kyoung Mu Lee (2016). "Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring". In: *CoRR* abs/1612.02177. arXiv: 1612.02177. URL: http://arxiv.org/abs/1612.02177.

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML'10. Haifa, Israel: Omnipress, pp. 807–814. ISBN: 978-1-60558-907-7. URL: http://dl.acm.org/citation.cfm?id=3104322.3104425.

Nalisnick, Eric et al. (2019). "Do Deep Generative Models Know What They Don't Know?" In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=H1xwNhCcYm.

Nathan Silberman Derek Hoiem, Pushmeet Kohli and Rob Fergus (2012). "Indoor Segmentation and Support Inference from RGBD Images". In: *ECCV*. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/shkf_eccv2012.pdf.

Niklaus, Simon, Long Mai, and Feng Liu (2017a). "Video Frame Interpolation via Adaptive Convolution". In: *CoRR* abs/1703.07514. arXiv: 1703.07514. URL: http://arxiv.org/abs/1703.07514.

— (2017b). "Video Frame Interpolation via Adaptive Separable Convolution". In: *IEEE International Conference on Computer Vision*. URL: https://arxiv.org/abs/1708.01692.

NTIRE 2019. http://www.vision.ee.ethz.ch/ntire19/.

Pan, Jinshan et al. (2018). "Physics-Based Generative Adversarial Models for Image Restoration and Beyond". In: *CoRR* abs/1808.00605. arXiv: 1808.00605. URL: http://arxiv.org/abs/1808.00605.

PIRM 2018. https://www.pirm2018.org/.

pix2pix. https://phillipi.github.io/pix2pix/.

PyTorch. https://pytorch.org/.

Qian, Rui et al. (2017). "Attentive Generative Adversarial Network for Raindrop Removal from a Single Image". In: *CoRR* abs/1711.10098. arXiv: 1711.10098. URL: http://arxiv.org/abs/1711.10098.

Quora. https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-unsupervised-learning.

Redmon, Joseph et al. (2015). "You Only Look Once: Unified, Real-Time Object Detection". In: *CoRR* abs/1506.02640. arXiv: 1506.02640. URL: http://arxiv.org/abs/1506.02640.

Rehman, Abdul (2013). "SSIM-Inspired Quality Assessment, Compression, and Processing for Visual Communications". In: URL: https://core.ac.uk/download/pdf/144146749.pdf.

Ruder, Sebastian (2017). "An Overview of Multi-Task Learning in Deep Neural Networks". In: *CoRR* abs/1706.05098. arXiv: 1706.05098. URL: http://arxiv.org/abs/1706.05098.

Ruder, Sebastian et al. (2017). "Sluice networks: Learning what to share between loosely related tasks". In:

Scharstein, Daniel et al. (2014). "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth". In: vol. 8753, pp. 31–42. DOI: 10.1007/978-3-319-11752-2_3. URL: https://link.springer.com/chapter/10.1007/978-3-319-11752-2_3.

scikit-image. https://github.com/scikit-image/scikit-image.

Simonyan, Karen and Andrew Zisserman (2014). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556. arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556.

Szegedy, Christian, Sergey Ioffe, and Vincent Vanhoucke (2016). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". In: *CoRR* abs/1602.07261. arXiv: 1602.07261. URL: http://arxiv.org/abs/1602.07261.

Tripathi, Subarna, Zachary C. Lipton, and Truong Q. Nguyen (2018). "Correction by Projection: Denoising Images with Generative Adversarial Networks". In: *CoRR* abs/1803.04477. arXiv: 1803.04477. URL: http://arxiv.org/abs/1803.04477.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor S. Lempitsky (2017). "Deep Image Prior". In: *CoRR* abs/1711.10925. arXiv: 1711.10925. URL: http://arxiv.org/abs/1711.10925.

Wang, Xintao et al. (2018). "ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks". In: *CoRR* abs/1809.00219. arXiv: 1809.00219. URL: http://arxiv.org/abs/1809.00219.

Wang, Zhou et al. (2004). "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *Image Processing, IEEE Transactions on* 13, pp. 600 –612. DOI:

10.1109/TIP.2003.819861. URL: http://www.cns.nyu.edu/pub/lcv/wang03-preprint.pdf.

Xu, Li, Shicheng Zheng, and Jiaya Jia (2013). "Unnatural L0 Sparse Representation for Natural Image Deblurring". In: URL: http://www.cse.cuhk.edu.hk/leojia/projects/l0deblur/.

Xu, Yingxue et al. (2018). "Chinese Herbal Recognition based on Competitive Attentional Fusion of Multi-hierarchies Pyramid Features". In:

Yu, Xiaoli, Yanyun Qu, and Ming Hong (2019). "Underwater-GAN: Underwater Image Restoration via Conditional Generative Adversarial Network: ICPR 2018 International Workshops, CVAUI, IWCF, and MIPPSNA, Beijing, China, August 20-24, 2018, Revised Selected Papers". In: pp. 66–75. ISBN: 978-3-030-05791-6. DOI: 10.1007/978-3-030-05792-3_7.

Zhang, He, Vishwanath Sindagi, and Vishal M. Patel (2017). "Image De-raining Using a Conditional Generative Adversarial Network". In: *CoRR* abs/1701.05957. arXiv: 1701.05957. URL: http://arxiv.org/abs/1701.05957.

Zhang, Xinyi et al. (2018). "Gated Fusion Network for Joint Image Deblurring and Super-Resolution". In: *CoRR* abs/1807.10806. arXiv: 1807.10806. URL: http://arxiv.org/abs/1807.10806.