ISTANBUL TECHNICAL UNIVERSITY
ELECTRICAL-ELECTRONICS FACULTY

RGB-TO-IR IMAGE TRANSLATION WITH GENERATIVE ADVERSARIAL NETWORK

SENIOR DESIGN PROJECT

Enes ERDOĞAN

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT

JUNE 2021

ISTANBUL TECHNICAL UNIVERSITY
ELECTRICAL-ELECTRONICS FACULTY

RGB-TO-IR IMAGE TRANSLATION WITH GENERATIVE ADVERSARIAL NETWORK

SENIOR DESIGN PROJECT

Enes ERDOĞAN
(040160231)

ELECTRONICS AND COMMUNICATION ENGINEERING
DEPARTMENT

Project Advisors: Prof. Dr. Hakan Ali ÇIRPAN,
Dr. Sedat ÖZER

JUNE 2021

İSTANBUL TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ

ÜRETKEN ÇEKİŞMELİ AĞLAR KULLANARAK OPTİK GÖRÜNTÜNÜN KIZILÖTESİ GÖRÜNTÜYE DÖNÜŞTÜRÜLMESİ

LİSANS BİTİRME TASARIM PROJESİ

Enes ERDOĞAN
(040160231)

Proje Danışmanları: Prof. Dr. Hakan Ali ÇIRPAN,
Dr. Sedat ÖZER

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ

HAZİRAN, 2021

We are submitting the Senior Design Project Report entitled as “RGB-TO-IR IMAGE TRANSLATION WITH GENERATIVE ADVERSARIAL NETWORK”. The Senior Design Project Report has been prepared as to fulfill the relevant regulations of the Electronics and Communication Engineering Department of Istanbul Technical University. We hereby confirm that we have realized all stages of the Senior Design Project work by ourselves and we have abided by the ethical rules with respect to academic and professional integrity .

Enes ERDOĞAN ....
(040160231)

FOREWORD

I would like to thank my advisors Prof. Hakan Ali ÇIRPAN and Dr. Sedat ÖZER who assisted me to research such a pleasant field. Also, I would like to thank them for their guidance and help throughout the project.

Also, I want to express my gratefulness to my family who has supported me unconditionally.

June 2021 Enes ERDOĞAN

TABLE OF CONTENTS

Page

FOREWORD .... **iv**
TABLE OF CONTENTS .... **vii**
ABBREVIATIONS .... **ix**
SYMBOLS .... **x**
LIST OF TABLES .... **xi**
LIST OF FIGURES .... **xii**
SUMMARY .... **xiii**
ÖZET .... **xiv**
1. INTRODUCTION **1**
  1.1 Basics of Generative Adversarial Networks 1
  1.2 Loss Function of GAN 3
  1.3 About the Infrared Imaging 5
2. CHALLENGES OF TRAINING A GAN **7**
  2.1 Mode Collapse 7
  2.2 Vanishing Gradient 7
  2.3 Disjoint Supports 8
3. CANDIDATE SOLUTIONS AND ADVANCED TRAINING TECHNIQUES **11**
  3.1 Conditioning 11
  3.2 Wasserstein Loss 11
  3.3 Feature Matching 13
  3.4 Minibatch Discriminator 13
  3.5 Least Square Loss 13
  3.6 Progressive Growing Structure and Its Alternatives 14
4. LITERATURE REVIEW **17**
  4.1 Pix2Pix Model 17
  4.2 Pix2PixHD 18
  4.3 MU-Net 20
  4.4 Style Transfer Neural Network 21
5. DESIGNING A GAN MODEL **23**
  5.1 Design Specifications 23
  5.2 The Model 24
    5.2.1 The generator 24
    5.2.2 The multi-scale discriminator 25
  5.3 The Loss Function 25
6. IMPLEMENTATION AND TRAINING **27**
  6.1 FLIR Thermal Dataset 27
    6.1.1 Pre-processing 28
    6.1.2 Image segmentation 29
  6.2 Implementation 29
  6.3 Training Details 30
  6.4 Results 32

7. REALISTIC CONSTRAINTS AND CONCLUSIONS 35
7.1 Practical Application of this Project 35
7.2 Realistic Constraints 35
7.2.1 Social, environmental and economic impact 36
7.2.2 Cost analysis 36
7.2.3 Standards 36
7.2.4 Health and safety concerns 36
7.3 Future Work and Recommendations 36
REFERENCES 38
APPENDICES 41
APPENDIX A 42
CURRICULUM VITAE 44

ABBREVIATIONS

AI : Artificial Intelligence
FM : Feature Matching
GAN : Generative Adversarial Network
GPU : Graphics Processing Unit
IEEE : Institute of Electrical and Electronics Engineers
IR : Infrared image
JS : Jensen-Shannon
KL : Kullback–Leibler
LLN : Law of Large Numbers
LSGAN : Least Square GAN
MU-NET : Multiple U-Net
PEP : Python Enhancement Proposal
Pix2Pix : Pixel to Pixel Translation GAN
ProGAN : Progressive Growing GAN
RGB : Red-Green-Blue (optical) image
TV : Total Variation
UAV : Unmanned Aerial Vehicle
WGAN-GP : Wassertion GAN with gradient penalty

SYMBOLS

$\mathbb{E}$ **:** Expected value
D **:** Discriminator Network
G **:** Generator Network
$\boldsymbol{p}$ **:** Data distribution
$\boldsymbol{\mathcal{L}}$ **:** Loss function
$\boldsymbol{\lambda}$ **:** Coefficient for additional loss terms
L1 : L1 norm
VGG **:** Backbone network for feature extraction

LIST OF TABLES

Page

Table 6.1 : Prerequisites. **27**
Table 6.2 : Annotations in the dataset. **28**

LIST OF FIGURES

Page

Figure 1.1 : Classification of the generative models [1]. **2**
Figure 1.2 : Structure of GAN. **3**
Figure 2.1 : Magnitude of the gradient of the discriminator when the generator trained for only 1, 10, or 25 epochs and fixed [4]. **8**
Figure 2.2 : The manifold of the data in the shape of a Swiss roll is shown [21]. **8**
Figure 3.1 : Comparison between the discriminator and critic [5] **12**
Figure 3.2 : Structure of ProGAN [9]. **15**
Figure 3.3 : Three alternatives of the progressive growing structure proposed in the StyleGAN2 paper [10]. **15**
Figure 4.1 : U-Net structure with skip connections [11]. **17**
Figure 4.2 : Generator structure of Pix2pixHD model [7]. **18**
Figure 4.3 : In the first row transposed convolution is used, and in the second row it is not used [13]. **20**
Figure 4.4 : MU-Net1-b [12]. **21**
Figure 4.5 : Structure of the style transfer network [14]. **22**
Figure 5.1 : Architecture of the generator. **24**
Figure 5.2 : Architecture of the discriminator. **25**
Figure 5.3 : Perceptual loss calculations. **26**
Figure 6.1 : Example RGB and IR image from the dataset. **28**
Figure 6.2 : Semantic segmentation of Figure 6.1. .... **29**
Figure 6.3 : Overall loss change over the iterations. **30**
Figure 6.4 : Loss change for the first scale images. **31**
Figure 6.5 : Loss change for the second scale images. **31**
Figure 6.6 : Loss change of the perceptual loss and feature matching loss (2 scales were shown separately). **32**
Figure 6.7 : Result image. **33**
Figure 6.8 : Result image. **33**
Figure 6.9 : Result image. **34**
Figure 6.10 : Result image. **34**

RGB-TO-IR IMAGE TRANSLATION WITH GENERATIVE ADVERSARIAL NETWORK

SUMMARY

IR imaging has very crucial implementations in various engineering required fields such as flame detection, surveillance, UAVs, search and rescue operations etcetera. To improve their abilities, most of these systems have work with artificial intelligence (AI) in the background. Due to the nature of AI, these systems need a huge amount of data for training. However, IR images are hard to obtain due to the high cost of IR cameras. And even with a camera, obtaining images for specific scenarios could be very time-consuming. To meet the needs of our target group-AI developers, we want to create a model that has the capability to translate a wide range of RGB images to IR images with high quality and high accuracy by using generative adversarial networks (GAN).

As a highly active research area in deep learning, GAN is a type of generative model and in a general sense, these models try to approximate the distribution of the real-world data and then produce samples from this distribution. GAN consists of two neural networks so-called generator and discriminator. The generator produces an image from the noise sampled from the latent space, and its purpose is to make the discriminator believe that the data it generates comes from the same distribution as the real dataset. Meanwhile, the discriminator works as a binary classifier network, and it predicts whether the image is fake or real by looking at real and generated images. During the training, the generator learns how to map latent space into the image space i.e. it learns to extract the structures in the image then learns how to embed course and fine details of the image such as shapes, textures, colors, etc. into the latent space. These two neural networks are trained adversarially to reach the minimum of adversarial loss. The generator network can also take a condition vector as an input in addition to a noise vector so that the output image can be controlled through the condition vector to produce images with desired features. By taking this concept one step further, an image itself also can be conditioned to translate the image to have a different style, texture, and etcetera. This project is also an example of translational-type GAN.

The constructed model will be based upon the Stefan-Boltzmann Law that indicates that the IR emission of the material depends on the material type and its temperature. Considering that, if the material type is known, a model can predict its IR image.

After reviewing the literature briefly, all the knowledge is combined to construct a partially novel model. Results will be discussed and obtained images will be shown in the result section.

ÜRETKEN ÇEKİŞMELİ AĞLAR KULLANARAK OPTİK GÖRÜNTÜNÜN KIZILÖTESİ GÖRÜNTÜYE DÖNÜŞTÜRÜLMESİ

ÖZET

Kızılötesi (termal) kameralar, alev algılama sistemlerinden, insansız hava araçlarına (İHA), arama kurtarma operasyonlarında, güvenlik sistemlerine kadar çeşitli mühendislik alanlarındaki kullanımıyla hayati önem taşımaktadır. Kızılötesi görüntüleme sistemlerin kabiliyetlerini artırmak adına yapay zeka desteğinin kullanılması artık günümüz teknolojisi bir gereğidir. Günümüzde kullanılan derin öğrenme temelli yapay zeka modellerini eğitmek için yüksek miktarda eğitim verisi gerekmektedir. Ancak kızılötesi kameraların yüksek maliyetlerinden dolayı bu alanda proje geliştirmek birçok mühendis/şirket için zordur. Ek olarak kameraya ulaştıktan sonra hedeflenen senaryoya özgü olarak sahneler kurup veriyi toplamak diğer bir problemdir. Bu kapsamda geliştiricilerin ihtiyacını karşılamak adına optik görüntüyü termal görüntüye çevirebilen bir modele ihtiyaç duyulmaktadır. Bu proje kapsamında derin öğrenmenin görece yeni ve aktif bir araştırma alanı olan çekişmeli üretici ağ (ÇÜA) yapısını kullanarak normal kameradan elde edilen görüntünün gerçekçi bir şekilde kızılötesi görüntüye dönüştürülmesi amaçlanmıştır.

Önerilen modelin eğitimi için de belirli bir veri seti gerekse de uzun vadede yeterli miktarda ve çeşitlilikte objelerin olduğu veriler ile eğilirse, çıkarsama zamanında istenilen obje kombinasyonlarını içeren herhangi bir görüntüyü dönüştürebilme kabiliyetine sahip olacaktır.

Bir çeşit üretici model olan ÇÜA, kapalı dağılım kullanması sebebiyle literatürdeki diğer üretici modellerden ayrılır. ÇÜA, temelde üretici ve ayırıcı olarak adlandırılan iki yapay sinir ağının karşılıklı olarak eğitilmesinden müteşekkildir. Amaç eldeki gerçek veri setindeki görüntülerdeki yapıları öğrenerek, gerçek görüntülerin görüntü uzayındaki dağılımına yakınsayacak bir dağılım üretebilmektir. Üretici ağın görevi, belirli boyuttaki rastgele vektörlerden sinir ağları yardımıyla bir görüntü oluşturmaktır. Ayırıcı ağın amacı ise oluşturulan bu sahte görüntü ile eldeki gerçek görüntüyü sinir ağı kullanarak gerçek mi sahte mi olduğunu anlamaktır. Bu anlamda ayırıcı ağ, ikili sınıflandırma ağı olarak düşünülebilir. Üreticinin kullandığı rastgele vektör, saklı uzaydan örneklenir. Dolayısıyla aslında üretici ağın gerçek görevi saklı uzayı sinir ağları kullanılarak resim uzayına eşlemeye çalışmaktır.

ÇÜA modelinin eğitilmesi temelde iki sinir ağının parametreleri arasındaki işbirliksiz minimum-maksimum oyunu olarak tanımlanabilir. Bu oyunun denge noktasına Nash dengesi denir. ÇÜA yapısını oluşturan iki yapıda bir çeşit sinir ağı olduğundan rastgele gradyan inişi temelli bir optimizasyon fonksiyonu kullanılır. Ancak bu durumda Nash dengesi bulmak yerine kayıp fonksiyonunu minimize edecek şekilde bir nokta bulunmaya çalışılmış olur.

ÇÜA eğitilirken ek bir vektör ile koşullanabilir, bu sayede daha kaliteli sonuçlar kontrollü bir şekilde elde edilebilir. Bu vektör, üretilmek istenen veri setindeki bir

sınıfı temsil edebileceği gibi, görüntünün farklı özelliklerini kontrol etmek içinde kullanılabilir. Koşullama işlemi bir adım daha ileri taşınarak bir görüntünün kendisi de koşullanabilir. Bu şekilde herhangi bir görüntün temel yapısı korunup stil, doku gibi özellikleri istenilen şekilde manipüle edilebilir. Projeden kullanılan yapı da temelde bu şekilde çalışmaktadır.

ÇÜA yapılarını eğitmenin kaybolan eğim, mod çökmesi ve ayrık destek uzayı gibi çeşitli zorlukları vardır. Bu projede öznitelik eşleme, alternatif kayıp fonksiyonları kullanma, aşamalı büyüyen yapılar kullanma gibi literatürdeki ileri seviye eğitim metodlar araştırılarak uygulanmış ve modelin eğitimi daha stabil bir hale getirilmiştir.

Bu proje, Stefan-Boltzmann yasasınında belirttiği üzere cismin yaydığı kızılötesi ışınların cismin cinsi ve sıcaklığıyla orantılı olduğu bilgisine dayanarak tasarlanmıştır. Dolayısıyla tasarlanan modelin veri setindeki objeleri tanıyarak dönüştürmesi beklenmektedir. Teoride, tasarlanan sinir ağı modeli, bu objeleri tanıma kapasitesine sahip olsa da eğitmeyi kolaylaştırmak adına görüntünün anlamsal bölümlenmiş hali de modele girdi olarak verilebilir. Bu amaçla eldeki veri setine benzer ortamlarda kaydedilmiş görüntülerden oluşan bir veri setiyle ön-eğitilmiş bir anlamsal bölümleme modeli kullanılmıştır.

Ayrıca literatürdeki ileri görüntü çevirme çalışmaları incelenmiş, kullandıkları modelin yapısı, kayıp fonksiyonları analiz edildikten sonra edinilen bilgiler ışığında kısmen yeni bir model önerilmiştir. Bu model Pytorch kütüphanesi kullanılarak Colab Notebook ortamında FLIR termal görüntü veri seti ile eğitilmiştir. Elde edilen görüntüler raporlanmış ve yorumlanmıştır.

1. INTRODUCTION

This project aims to design a neural network model that can translate optical (RGB) images to infrared (IR) images with high fidelity using generative adversarial networks (GAN). First of all, I introduced the concept of GAN and its typical challenges. Then, I stated notably advanced training techniques of GAN. Afterward, the literature for translation type GAN was briefly reviewed. Finally, combining all this knowledge, the model was proposed, and obtained results of the proposed model were discussed.

1.1 Basics of Generative Adversarial Networks

In the last decade, many breakthrough developments have been achieved in various branches of deep learning. Especially, a subfield called generative models which belongs to semi-supervised learning has become the center of interest. Generative models try to approximate the distribution of the real data, then produce samples from this distribution. The field mainly categorized whether the distribution constructed explicitly or implicitly as can be shown in Figure 1.1. The concern of this project focused on the generative adversarial networks (GAN), which has implicit density i.e. instead of attempting to define data distribution (density) explicitly, in GAN, approximated distribution can only be accessed indirectly by sampling [1]. Although the term data could be anything, in general, GAN is used for image generation tasks.

GAN was first proposed by Ian Goodfellow in 2014, and since then it is a highly active research area. Many applications for various fields came out e.g. creating realistic images such as faces, cars, also image editing, anonymous medical data generation, image translation to different textures, colorization, and restoration, etc.

GAN consists of two neural networks so-called generator and discriminator. The generator produces data (e.g. images) from the noise sampled from the latent space, and then it tries to make the discriminator believe that the data it generates comes from the same distribution as real data i.e. purpose of the generator is to create data with respect to the dataset so realistically that discriminator supposes it is coming from the real dataset. Noise vector sampled from n-dimensional latent space can be denoted as $z \in [0, 1]^n$

Meanwhile, the discriminator fed with real and generated data tries to identify whether the data is real or generated. So, it is basically a binary classifier and trained with labels i.e. it is a supervised model.

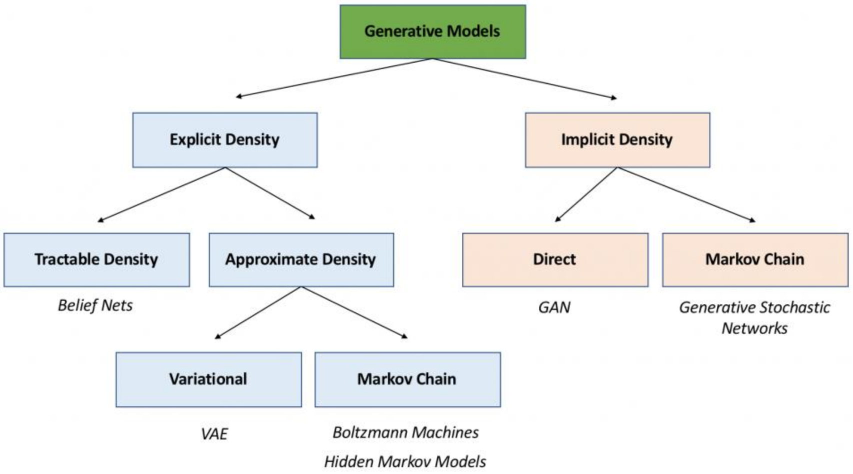

Figure 1.1 : Classification of the generative models [1].

Since both models are neural networks with differentiable parameters, they are trained with gradient descent-based algorithms, and they have been trained alternately. Although the generator does not see any real data directly, it is trained according to the feedback that comes from the discriminator's output.

At the beginning of the training, it is expected that both networks perform poorly, after this adversarial training, the performance of both will get better and better. The word adversarial refers to that both networks use the same loss function, but while the generator tries to minimize this loss, the discriminator tries to maximize it. Finally, there will be no need for a discriminator, and the generator can create images just from the noise.

It is fundamentally a non-cooperative min-max game between the discriminator and the generator. Thus, it requires to find the Nash Equilibrium point between the parameters of these two networks. However, gradient descent minimizes the loss function instead of directly finding Nash Equilibrium since it may not be converged easily [2].

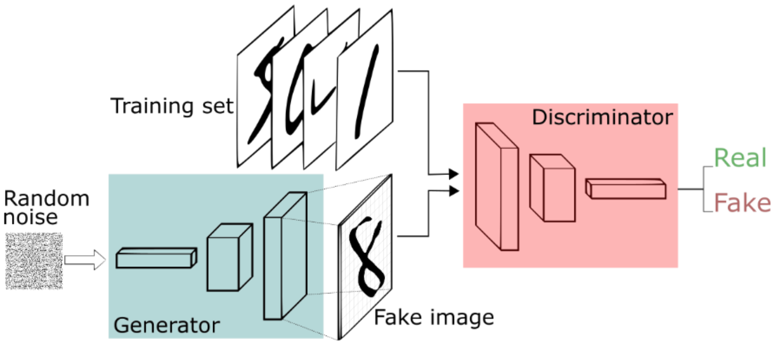

Figure 1.2 : Structure of GAN.

The strength of the GAN comes from the concept of latent space. During the training, in a deeper sense, the generator learns how to map latent space to image space i.e. it learns to extract the structures in the image then learns how to embed course and fine details of the image such as shapes, textures, colors, etc. into the latent space. This makes GAN to acquire the ability to produce infinitely many samples with diverse features instead of just one deterministic sample.

Another advantage of latent space is that moving slightly around the sampled noise is going to correspond to a similar image with different details. Furthermore, by moving in a specific direction, some specific features of the image can be manipulated. For example, a GAN model trained with a face dataset can manipulate the face to make it smiley, or it can change its skin color by moving in the latent space.

How a vanilla GAN can be used to translate images will be explained in the 3.1.

1.2 Loss Function of GAN

Most of the previous generative models have been used a loss function based on the maximum likelihood estimation. It can be shown that it is equivalent to Kullback–Leibler (KL) divergence between real distribution, p_r and approximated distribution, p_θ, where θ represents the parameter of the distribution. KL-divergence is a concept in information theory, and it measures the distance between two distributions. However, contrary to distance, it lacks some properties of the notion of distance e.g. symmetry, triangular inequality.

$$KL(p_r||p_\theta) = \int_x p_r(x) \log \frac{p_r(x)}{p_\theta(x)} dx \tag{1.1}$$

Put it differently, minimizing KL-divergence encourages the model to have p_θ that more resembles the real-data distribution p_r.

On the other hand, the loss function of the vanilla GAN is as follows

$$\min_G \max_D \mathcal{L}(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1-D\big(G(z)\big)\right)\right] \tag{1.2}$$

Also can be written as

$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(z)}\big[\log\big(1-D(x)\big)\big] \tag{1.3}$$

where p_r is the distribution of the real dataset, p_g is the distribution of the generated samples, and p_z is the distribution of the noise.

In the original paper, it has been proven that when the discriminator is optimal i.e. perfect discriminator with an accuracy of one is achieved, the loss function of GAN turns out to be Jensen–Shannon divergence, which is similar to KL divergence but it is symmetric. It was claimed that the success of the GAN compared to the previous generative models comes from its novel loss function [3].

By utilizing the Law of Large Numbers (LLN), it can be transformed into the following form. This form is used during the implementation.

$$\mathcal{L}(D,G) = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} 1\cdot\log\left(D\left(x_i^{(r)}\right)\right) + \lim_{m\to\infty}\frac{1}{m}\sum_{j=1}^{m} 1\cdot\log\left(1-D\left(x_j^{(g)}\right)\right) \tag{1.4}$$

Then the equation 1.4 can be written as follows.

$$= -\frac{1}{m}\sum_{i=1}^{m}\left[CE\left(1, D\left(x_i^{(r)}\right)\right) + CE\left(0, D\left(x_i^{(g)}\right)\right)\right] \tag{1.5}$$

Where CE stands for cross-entropy.

In the literature, many kinds of loss functions have been suggested for different purposes. I will particularly mention the Wasserstein Loss and Least Square Loss in the following chapter.

1.3 About the Infrared Imaging

Before the further investigation of the GAN structure, the infrared imaging and the reasoning of the project were explained in this section.

Every object above the temperature of absolute zero degrees emits electromagnetic radiation due to its internal energy. More specifically, according to Stefan-Boltzmann Law, the amount of this radiated energy is proportional to the emissivity of the material and also proportional to the fourth power of its absolute temperature [17]. This phenomenon which is also called thermal radiation can be detected partially via infrared cameras. Especially objects at room temperature almost only emits infrared light [18].

An infrared camera first captures the energy and then converts it into an electrical signal. Then, the energy information is mapped into a color-coded representation for better visual interpretability. Although IR cameras are monochromatic, the intensity of the energy is represented with pseudo-colors.

It can be concluded from the mentioned law above that material type gives important sight about the amount of infrared radiation since emissivity directly, and temperature of an object at a particular area indirectly depends on the material type. Hence, if the model can learn the material types, it can also tranlate them to the IR image.

The quality of the image taken by the IR camera depends on the camera’s specifications. For instance, the wider the spectral range of the camera, the more accurate it represents the scene, and another important property is the thermal sensitivity of the camera which describes the smallest temperature difference that can be measured [19]. Overall, it can be concluded that different IR cameras may be imaging the same scene differently. This raises an important problem. Since the dataset may be constructed from various sources, normalization may not make sense, and the same object may have very different values.

Note that in literature, thermal camera and IR camera are used interchangeably in general. In this document, the term IR camera was preferred.

2. CHALLENGES OF TRAINING A GAN

2.1 Mode Collapse

GAN is expected to learn and generate images belong to various classes in the dataset. However, in practice, the generator may produce slightly better images in certain classes during the training. In that case, the score of the discriminator for those images will be high, and this situation indirectly encourages the generator to produce more images from these certain classes. Hence, after training a while, generator will only produce these images of certain classes to reduce the loss. This phenomenon is called mode collapse, and it is typical in vanilla GAN. The term mode refers to average (frequent) images of the dataset such as classes. For instance, a GAN network trained with a hand-written digit dataset, after some training, may collapse certain modes. For instance, it produces only digits 3 and 1. This undesirable result may be solved with the Wasserstein Loss function or minibatch discrimination layer discussed in the following section.

2.2 Vanishing Gradient

It was mentioned that, for optimum discriminator, loss function becomes JS divergence. Thus, it can be expected that first training the discriminator to the optimum then training both of them alternately helps the generator to gradually produce better images. Also, in practice, the discriminator usually can reach the optimum point faster than the generator. However, this is not the case. When the discriminator is optimum, the score either gets close to 0 or close to 1, so the gradient of the generator gets vanished. This was experimented by Arjovsky et al. [4] as shown in Figure 2.1.

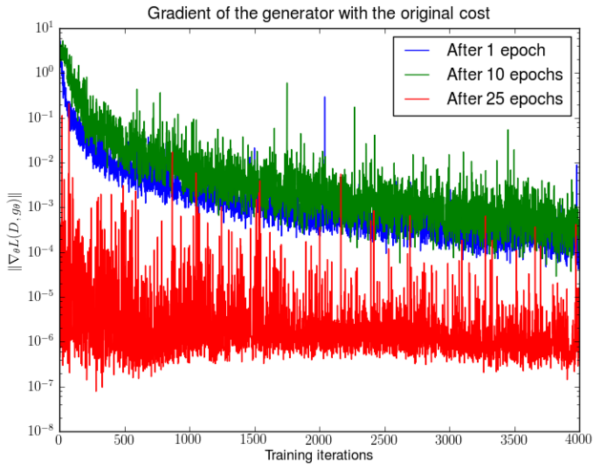

Figure 2.1 : Magnitude of the gradient of the discriminator when the generator trained for only 1, 10, or 25 epochs and fixed [4].

As a result, it is always desired for discriminator and generator to learn balancedly. In some cases, this problem simply can be solved by updating the parameters of the generator multiple times while the discriminator updated once. Also, it is claimed that Wasserstein loss or least square loss can solve this problem.

2.3 Disjoint Supports

According to the manifold hypothesis, the dimension of the real-world data is actually artificially high i.e. High dimensional data lies on the low dimensional manifolds [20]. The formal mathematical definition of the manifold is out of the scope of this project. Instead, its meaning in the machine learning context can be intuitively understood from the simple example in Figure 2.2.

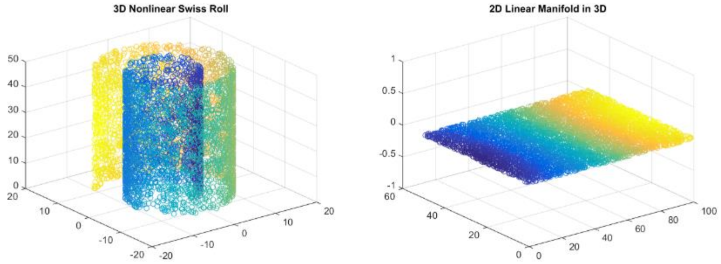

Figure 2.2 : The manifold of the data in the shape of a Swiss roll is shown [21].

In the first part, data are 3-dimensional. However, we can find a transformation to get the second part, which shows that data is actually artificially high dimensional, i.e. it is a 2-dimensional manifold in 3-D space.

By extending this idea, an image can be also considered as a high dimensional point in the R^{m*n} space where m and n are the width and height of the image. So, an image dataset with the distribution p_r is a low dimensional manifold in this high dimensional space. The purpose of the generator is to map the latent space z into the p_g such that ultimately tries to approximate p_r. Since the dimension of the latent space is typically much lower than image space, for a generator, it is impossible to cover the real image manifold, and this is independent from the number of parameters of the generator [4]. Furthermore, it was stated that this may lead to p_r and p_g to become disjoint. For disjoint distributions, JS divergence is constant $\log 2$ meaning that it cannot provide meaningful feedback to the generator, and this is quite a challenge for the training GAN.

3. CANDIDATE SOLUTIONS AND ADVANCED TRAINING TECHNIQUES

In this section, I want to briefly mention about notable training techniques such that they stabilize the training and increase the fidelity of the generated images. Also, they could be a remedy to the mentioned challenges in the previous section.

3.1 Conditioning

When the dataset has countable classes or labels like in a hand-written digit dataset, generator and discriminator can be conditioned with the class information so that image of the desired class can be generated. This is usually done by concatenating a one-hot-encoded label vector and label matrix to the noise and image as given in equation 3.1.

$$\mathcal{L}(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D\,(x,y)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\big(G(z,y)\big)\right)\right] \tag{3.1}$$

Where y denotes the label information.

Conditioning makes the GAN controllable. Also, it improves the performance of the model significantly mainly because the generator and discriminator could learn common class patterns more easily.

Furthermore, for translation purposes, an image itself can be conditioned to generate a new image with desired additional characteristics. For instance, image colorization, labels to image translation, sketch to image translation, image stylization are some of the well-known applications. In this project, also translation-type GAN is used.

3.2 Wasserstein Loss

To remedy some of the mentioned problems, Wasserstein distance was proposed by Arjovsky et al. as an alternative loss function [5]. Informally, Wasserstein distance can be defined as the minimum cost of transforming a pile of soil from one shape to another. Here, shapes can be thought of as probability distributions. The formula of the Wasserstein distance is over complicated to mention. Instead, its dual form derived by using Kantorovich-Rubinstein duality is given in equation 3.2. During the implementation also, this form is used.

$$W(P_r, P_g) = \sup_{|f|_L \leq 1} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)] \tag{3.2}$$

where f refers to the critic. Instead of discriminator, the term critic was preferred because its range is not limited between 0 and 1, it can give any value. And, $|f|_L \leq 1$ refers to the Lipschitz continuity. Lipschitz continuity expresses the limit of the change of the function. Ensuring the Lipschitz continuity prevents the sudden jumps and falls in the loss function which is a problem because it makes the result of the gradient update unpredictable.

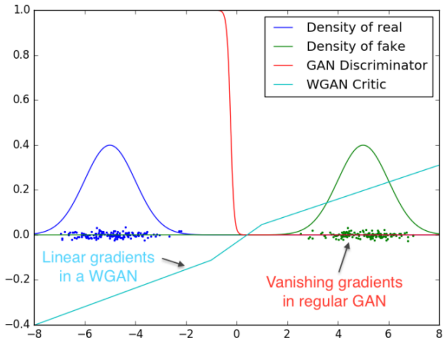

Figure 3.1 : Comparison between the discriminator and critic [5]

To satisfy the Lipschitz continuity condition, in the original paper, a gradient clipping method was suggested [5]. However, even they admitted that it is not an ideal solution. Another approach is adding a term to the loss function so-called gradient penalty that encourages the gradient to get closer to one [6].

The effect of Wasserstein loss can be observed from the toy example in Figure 3.1. There are two data distributions that are disjoint. The gradient of the discriminator does not help the generator to get optimized due to saturation. On the other hand, WGAN can effectively give feedback to the generator even if real and generated data are disjoint.

3.3 Feature Matching

To increase the stability of the training, the following term can be added to the loss function.

$$\mathcal{L}_{\mathrm{FM}}(G,D) = ||\mathbb{E}_{x\sim p_{\mathrm{r}}} f(x) - \mathbb{E}_{z\sim p_{z}(z)} f\big(G(z)\big)||_2^2 \tag{3.3}$$

where f represents the activations of the intermediate layers of the discriminator. These activations contain discriminatory information about the real and generated images. Hence, feature matching is simply a way to minimize the statistics between the generated images and real images [2]. This technique was used in the so-called Pix2PixHD, an image translation GAN model [7].

3.4 Minibatch Discriminator

A discriminator is not aware of the variety of the generated images, so there is no mechanism to prevent the generator from mode collapse. To solve this problem, a procedure that calculates the variance of the input images was proposed, and the result can be concatenated to the intermediate layer's output [2]. Since real images are very variable naturally, this will push the generator to produce images with high diversity.

3.5 Least Square Loss

Another alternative loss function called least square GAN was proposed by Mao et al. [8] to solve the vanishing gradient problem. They assert that using fake samples -that are discriminated as real- to update the generator will produce almost no backpropagation even if these fake samples are far from the real samples. This is the case partially due to the nature of the sigmoid function that is used to standardize the output of discriminator between zero to one, so in this scenario vanishing gradient is inevitable.

On the other hand, by using their proposed least square loss, samples discriminated as real but far from real samples will also be penalized.

The Least square loss function has the following form:

$$\min_{D} \mathcal{L}_{\text{LSGAN}}(D) = \frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}(x)}[(D(\boldsymbol{x})-1)^2] + \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}[(D(G(\boldsymbol{z})))^2] \tag{3.4}$$

$$\min_{G} \mathcal{L}_{\text{LSGAN}}(G) = \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}[(D(G(\boldsymbol{z}))-1)^2] \tag{3.5}$$

They proved that the least square loss function is equivalent to minimization of Pearson χ^2 divergence between $p_r + p_g$ and $2 \cdot p_g$. As mentioned before divergence is a statistical similarity measurement between two distributions. They have experimented that using least square loss increases the stability and produces better quality images. Also, they showed this loss function could be a remedy to the mode collapse issue.

Note that to implement least square loss, the sigmoid activation function at the end of the discriminator network is removed.

3.6 Progressive Growing Structure and Its Alternatives

In vanilla GAN, although sharp images can be acquired, high-resolution images are hard to generate. Progressive growing GAN (ProGAN) simplifying the generation process significantly [9]. It starts with the low-resolution image and gradually adds a new layer and increases its resolution. For example, instead of directly generating 1024 by 1024 image, it generates first 4 by 4 images then add a new layer to generate 8 by 8 image so on so forth. While a new layer is adding, there exist two paths. First, the output of the 4 by 4 block is taken as an input by 8 by 8 block, and the second image simply upsampled from the 4 by 4 image as shown in part b of Figure 3.2. Then they are added with the weight of alpha and 1-alpha, and the alpha value gradually is increased.

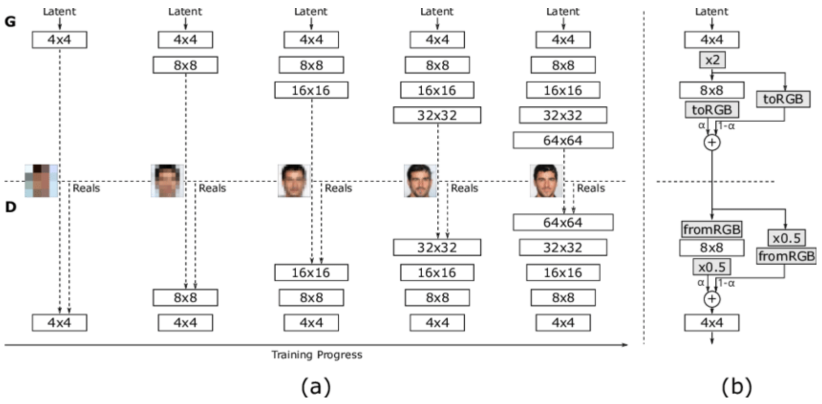

Figure 3.2 : Structure of ProGAN [9].

Alternatively, different models have been proposed that use the same idea as shown in Figure 3.3.

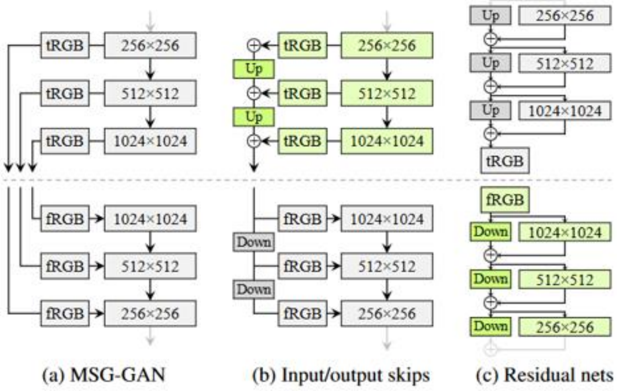

Figure 3.3 : Three alternatives of the progressive growing structure proposed in the StyleGAN2 paper [10].

4. LITERATURE REVIEW

In this section, well-known translation-type GAN models are reviewed, their structures and loss functions are analyzed.

4.1 Pix2Pix Model

Pix2Pix model was proposed by Isola et al. [11] and its purpose is to create a general model that can combine different kinds of image translation tasks e.g. semantic label to photo, black-white to a color image, edge to image, thermal to color images, etc. The generator part is called U-Net, and these images that are desired to be translated are given as an input. Instead of inputting noise directly, stochasticity is acquired from the dropout layer because the authors claimed that the model learns to ignore the noise in such a network. Since it is a translation, there are direct relations between input and output in terms of the structure of the images. This structural low-level information is carried by the skip-connections as shown in Figure 4.1.

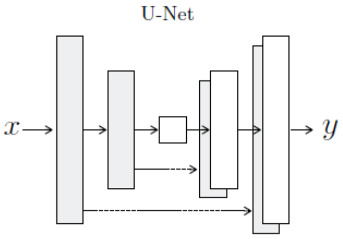

Figure 4.1 : U-Net structure with skip connections [11].

The discriminator is called the Markovian discriminator (also known as Patch discriminator). It penalizes only patch-size errors, and instead of outputting only one value, it gives a matrix of values that correspond to different parts of the image.

Its loss function has two components as seen in equation 4.1.

$$\min_{G} \max_{D} \mathcal{L}_{CGAN}(G,D) + \lambda\mathcal{L}_{L1}(G) \tag{4.1}$$

Where

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[||y - G(x,z)||_1], \tag{4.2}$$

and

$$\mathcal{L}_{CGAN}(G,D) = \mathbb{E}_{x,y}[\log D\,(x,y)] + \mathbb{E}_{x,z}\big[\log\big(1 - D\big(x, G(x,z)\big)\big)\big] \tag{4.3}$$

x refers to the input image-the condition, and y is the output. L1 loss directly penalizes pixel differences between the input and output. They claimed that patch discriminator penalizes the high-frequency structures, and L1 penalizes the low-frequency structures. Although their results are very successful, images may contain hallucinated objects.

4.2 Pix2PixHD

Over the previous Pix2Pix model, Wang et al. [7] developed the more advanced image translation network such that they can produce images with 2k by 1k resolution without any hallucinating objects in the image. Also, they designed a more controllable image editing mechanism i.e. object-level editing.

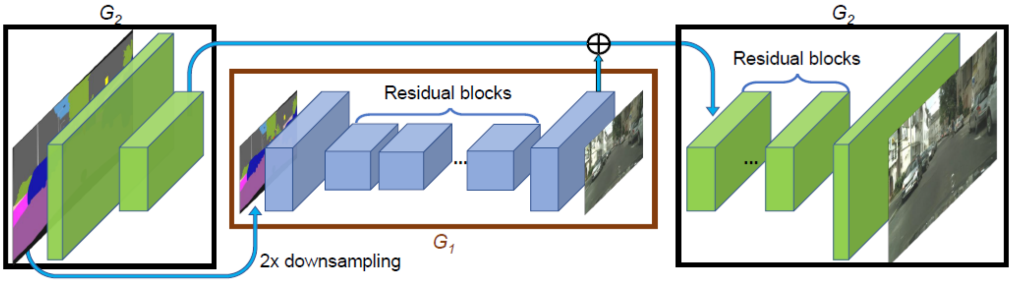

Figure 4.2 : Generator structure of Pix2pixHD model [7].

Similar to a progressive growing structure, they implemented the generator with a coarser-to-fine methodology. As shown in Figure 4.2, the generator consists of two networks: Global generator and Local enhancer, denoted by G1 and G2 respectively.

First, only the global generator is trained with downsampled images. Then, they are trained jointly to double the resolution. Contrary to the previous model, instead of concatenation, extracted global information was passed with a summation.

In the generator, residual convolutional layers were preferred. Residual networks are known for their ability to easily learn identity function. Since input and output share some common low-level structures, it is reasonable to use residual connections.

Since final images have high resolution, the discriminator should have a larger receptive field compared to the previous work to accurately evaluate its score. Receptive field size has a direct impact on the spatial coherency of the image. However, increasing the receptive field requires a deeper network or larger convolutional filters meaning that computational cost increases too. To deal with this, they use a multi-scale discriminator. They simply downsampled the images with the ratio of 2 and 4, then give it as the input to the discriminator network which was a PatchGAN discriminator as before. This structure improves the final image quality both globally and locally.

As discussed in previous sections, they used feature matching as an additional term in the loss. Also, they used perceptual loss for further improvement which will be explained in the next section. Hence, the loss function turned out to be in the following form.

$$\min_{G}\left(\left(\max_{D_1,D_2,D_3}\sum_{k=1,2,3}\mathcal{L}_{\mathrm{GAN}}(G,D_k)\right)+\lambda\sum_{k=1,2,3}\mathcal{L}_{\mathrm{FM}}(G,D_k)\right) \tag{4.4}$$

As just mentioned, there are three discriminators for each scale denoted with D_k. Instead of using adversarial loss, they preferred the least square loss. Also, FM loss represents the feature matching loss with the following formula.

$$\mathcal{L}_{\mathrm{FM}}(G,D_k)=\mathbb{E}_{(\mathrm{s},\mathrm{x})}\sum_{i=1}^{T}\frac{1}{N_i}\left[\left\|D_k^{(i)}(\mathrm{s},\mathrm{x})-D_k^{(i)}\big(\mathrm{s},G(\mathrm{s})\big)\right\|_1\right] \tag{4.5}$$

In equation 4.5, s represents the semantic label input, and x represents the corresponding natural image. To measure the distances between features, the L1 loss was preferred.

With this high-scale model, they achieved very satisfying results compared to the previous model.

4.3 MU-Net

The MU-Net model proposed by Iwashita et al. [12] to create a model that translates RGB images to IR images in order to use it in the Mars rovers since it is not going to have an IR camera anymore. Since it is for the Mars mission, they focused on the terrain objects. They also attempted to explain the background theory of how this translation is possible. Briefly, it is claimed that the type of the terrain object affects its emission which is captured by the IR camera.

They were using the transposed convolutional layer which may cause a checkboard effect [13]. Independent from this model, examples of the checkboard effect are shown in Figure 5.3.

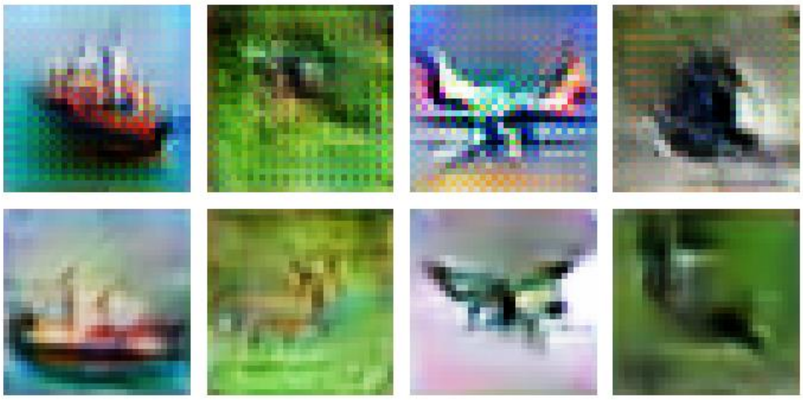

Figure 4.3 : In the first row transposed convolution is used, and in the second row it is not used [13].

Its loss function is as follows

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda\mathcal{L}_{MSE} \tag{4.6}$$

where cross-entropy is for annotation image and mean-square-error is for IR image. Instead of conditioning the annotation image, the model tries to produce it. Thus, during training, annotation information is extracted, and keep in parameters of the expansive unit. Also, in this paper, camera parameters are considered since the dataset was gathered from different cameras. Thus, the homography matrix approximately was calculated, and used to standardize the dataset.

Four similar models were proposed, and the most successful one is shown in Figure 4.4.

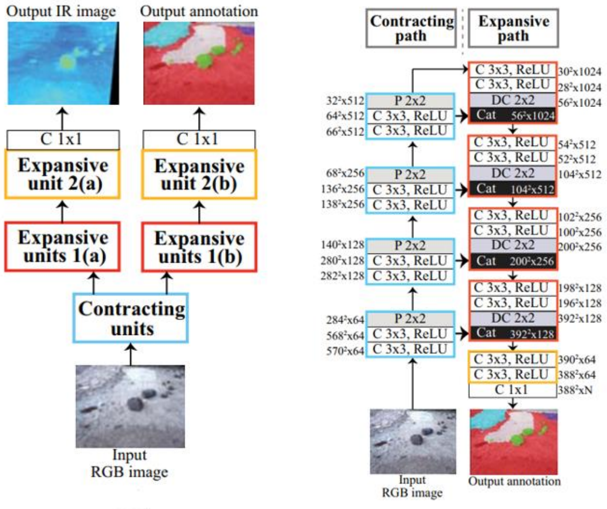

Figure 4.4 : MU-Net1-b [12].

The results of the paper were not that satisfying.

4.4 Style Transfer Neural Network

Although it was not a real GAN, the image stylization and super-resolution model proposed by Johnson et al. [14] gives important ideas about the training of translation-type GANs.

Instead, as shown in Figure 5.5, they use an image transformation network to generate the stylized image and a pre-trained VGG-16[1] network to calculate the loss. As the name implies, the purpose is to change the style of the image while preserving the content.

---

[1] Widely-used generic image 16-layer classification network proposed by Visual Geometry Group Lab.

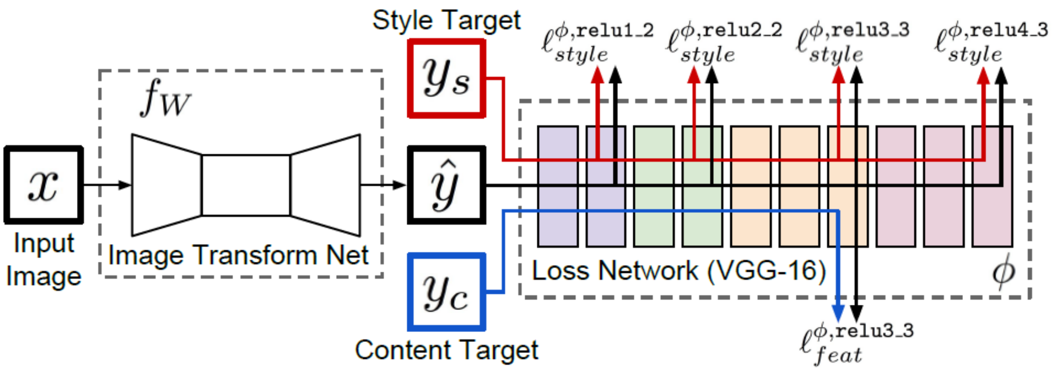
Style Target
y_s
$\ell_{style}^{\phi,relu1\_2}$
$\ell_{style}^{\phi,relu2\_2}$
$\ell_{style}^{\phi,relu3\_3}$
$\ell_{style}^{\phi,relu4\_3}$
f_W
x
Input Image
Image Transform Net
$\hat{y}$
y_c
Content Target
Loss Network (VGG-16)
ϕ
$\ell_{feat}^{\phi,relu3\_3}$

Figure 4.5 : Structure of the style transfer network [14].

Since VGG-16 was trained with ImageNet dataset which consists of a diverse range of objects, its intermediate features learned the highly compressed representations of the real-world perceptual information. Thus, it can identify the characteristics of the objects. To extract the style information, the model uses features from 4 different levels while to extract content information they only use features from a high level. This is because higher-level features have larger receptive fields, so content information was expected to get from higher levels of the network.

The loss function of the model is as follows

$$\hat{y} = arg\min_{y} \lambda_c \ell^{\phi}_{\text{feat}}(y, y_c) + \lambda_s \ell^{\phi}_{\text{style}}(y, y_s) + \lambda_{TV} \ell_{TV}(y) \tag{4.7}$$

Note that to stick with the original notation, the loss is represented with ℓ as an exception.

As the paper named, they represent feature reconstruction loss, style reconstruction loss, and total variation loss respectively. To improve the image smoothness, they add a total variation regularizer to the loss. Since content is supposed to be preserved, $y_c = x$ and y_s is target style image.

Note that this is different from the training GAN model with a pre-trained discriminator. Discriminator works like a classifier with a sigmoid cross-entropy loss function while this loss network directly compares the statistics of generated and target image.

In our model, a perceptual loss can be used to improve the training of the network as an additional loss.

5. DESIGNING A GAN MODEL

Until this point general and some advanced aspects of GAN were discussed. Additionally, some related image translation GANs in the literature were analyzed. By utilizing all this information, now it is objected to design a GAN model capable to translate RGB images to IR images.

After mentioning the specifications, an empirically designed model will be presented.

5.1 Design Specifications

Before directly constructing a model, first, it is important to define the design requirement of our model. The following objectives were deduced from the chapter 0 and 0and it will guide us throughout to design.

- Upsampling/downsampling layers: Even if input and output have the same size, sampling layers help to decrease computational complexity, also increases the receptive field without any cost. The importance of the downsampling was also mentioned in section 4.2.

- The transposed convolutional layer should not be preferred because it was shown that in that case output image tends to have checkboard distortion as mentioned before. Instead, to expand the feature maps, convolution with an upsampling could be used.

- Among the various normalization layers, in style transfer networks, it was claimed that using instance normalization -also called contrast normalization- produced more satisfactory results [15]. This is mainly because it gets rid of instance-specific contrast information. Since RGB to IR image transfer can also be thought of as style transfer by considering the RGB image as content image, instance normalization layer is an intuitive choice.

- Although creating a large-scale image is not our primary purpose, multiscale discriminators will allow us to use smaller discriminator structures. Its reasoning explained in the section 4.2.
- In addition to adversarial loss, feature matching loss and perceptual loss could be implemented.
- As mentioned before, residual connections should be used.

5.2 The Model

The structures of the generator and discriminator are explained. Note that dimensions of the blocks roughly represent the shape of feature maps — size-wise and channel-wise.

5.2.1 The generator

As a generator network, the image stylization network proposed by Johnson et al. was preferred. This was also the part of Pix2PixHD model. However, to avoid checkboard effects early in the training, the direct convolutional layer was preferred instead of the transposed convolution. Also, the number of residual blocks was decreased considering the resolution of the dataset. Each block contains a convolution layer, leaky rectified linear activation function, and instance normalization.

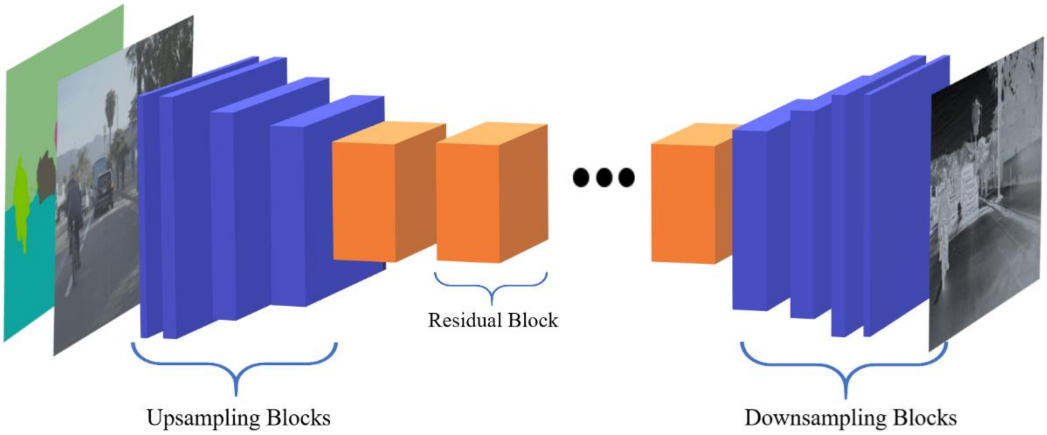

Figure 5.1 : Architecture of the generator.

Figure 5.1 illustrates the generator network. Downsampling and upsampling blocks are represented with the blue blocks. Between these, there are 6 residual blocks to refine the feature maps over and over.

5.2.2 The multi-scale discriminator

As mentioned earlier, a discriminator is simply a binary classifier, so by using fundamental layers such as a convolutional layer, batch normalization, and dropout discriminator network was designed as illustrated in Figure 5.2.

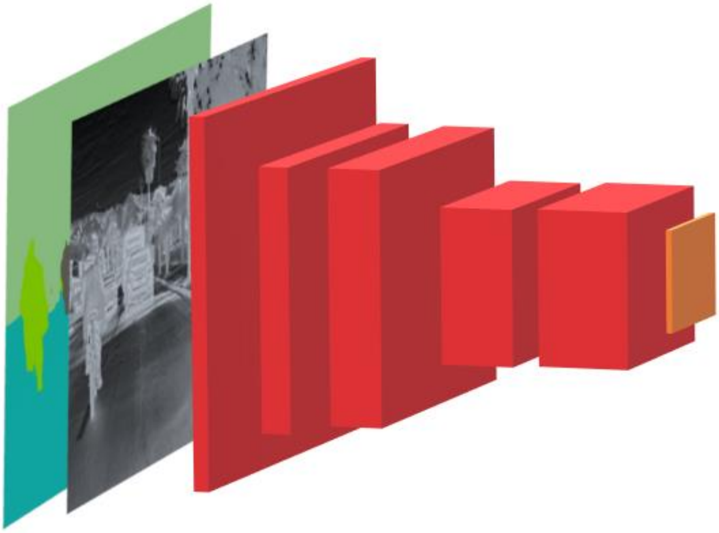

Figure 5.2 : Architecture of the discriminator.

To implement a multi-scale discriminator, 2 different discriminators are used. The first one processes the original images, and the second one processes the images that are resized by the ratio of 1/2.

Notice that instead of just one prediction value, the discriminator outputs a prediction matrix which was shown as an orange block in Figure 5.2. Each value in the matrix focuses on the 80 by 80 pixel in the image.

5.3 The Loss Function

To optimize the parameters of generator and discriminator, the least-square loss was preferred which was discussed in section 3.5. To improve its performances, additionally, feature matching loss and perceptual loss were used. As a result, the overall loss function turned out to be following form for the generator:

$$\min_{G}\left(\sum_{k=1,2}\mathcal{L}_{GAN}(G,D_k)+\lambda_{FM}\sum_{k=1,2}\mathcal{L}_{FM}(G,D_k)+\lambda_P\,\mathcal{L}_P\left(y_r,y_g\right)\right) \tag{5.1}$$

And for the multi-scale discriminator:

$$\max_{D_1,D_2} \sum_{k=1,2} \mathcal{L}_{LSGAN}(G, D_k) \tag{5.2}$$

FM loss is used as given in equation 4.5.

To implement perceptual loss, the VGG-19 network was used to extract the features. First, outputs after every pooling layer were multiplied with the indicated weights and then L1 distance as shown in Figure 5.3.

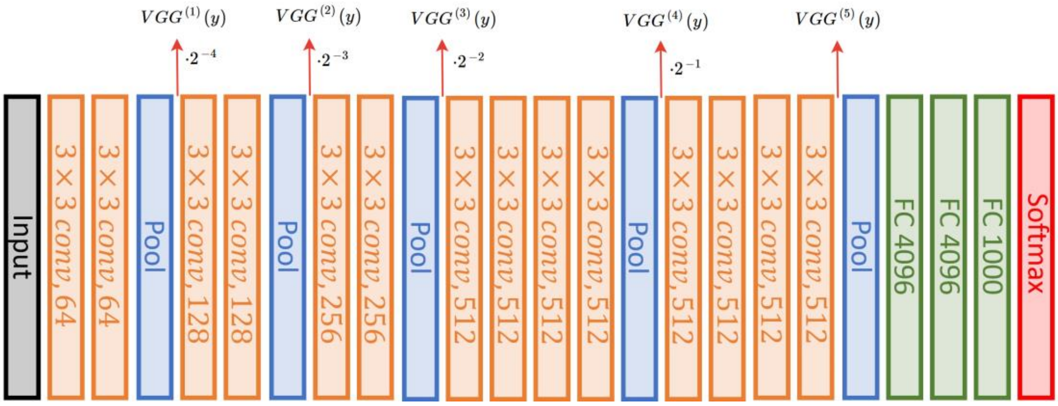

Figure 5.3 : Perceptual loss calculations.

Thus, its formula can be written as follows

$$\mathcal{L}_P(y_r, y_g) = \sum_{i=1}^{5} \frac{1}{2^{5-i}} \left[\left\| VGG^{(i)}(y_r) - VGG^{(i)}(y_g) \right\|_1 \right] \tag{5.3}$$

6. IMPLEMENTATION AND TRAINING

The model has been implemented in Python (3.8) programming language by using the Pytorch module. Pytorch module contains all of the required sophisticated functions to build and optimize any type of neural network. Although the model was developed at the local computer, all of the training and testing stages were completed on Google Pro Colabratory Notebook environment which enables us to use high-power GPUs. Hence, the installation steps will be omitted. To reproduce the model, versions of the non-built-in modules are shown in Table 6.1. Note that the last 2 modules are only required for the image segmentation module.

Table 6.1 : Prerequisites.

Module Names	Recommended Versions
Pytorch	1.6 or higher
scikit-image	0.16.2
Opencv	4.1.2
tqdm	4.41.1
keras-segmentation	0.3.0
Tensorflow	2.4.1
Keras	2.4.3

Additionally, the cost of the cloud computing service will be mentioned in the next chapter.

6.1 FLIR Thermal Dataset

The model was trained with the dataset known as FLIR Thermal Dataset [22]. It consists of synced RGB and IR images captured from cameras mounted onto a car such that their optical axes are 5 centimeters apart from each other. In terms of the lighting conditions, images have been captured in a day (60%) and night (40%) driving on Santa Barbara, CA area streets and highways. It contains approximately 10k images, and they include annotations of person, car, bicycle, dog, and other vehicles as can be seen in Table 6.1 in detail.

Table 6.2 : Annotations in the dataset.

Object	Total number of annotations
Person	28151
Car	46692
Bicycle	4457
Dog	240
Other Vehicles	2228

In Figure 6.1, an example image from the dataset is shown.

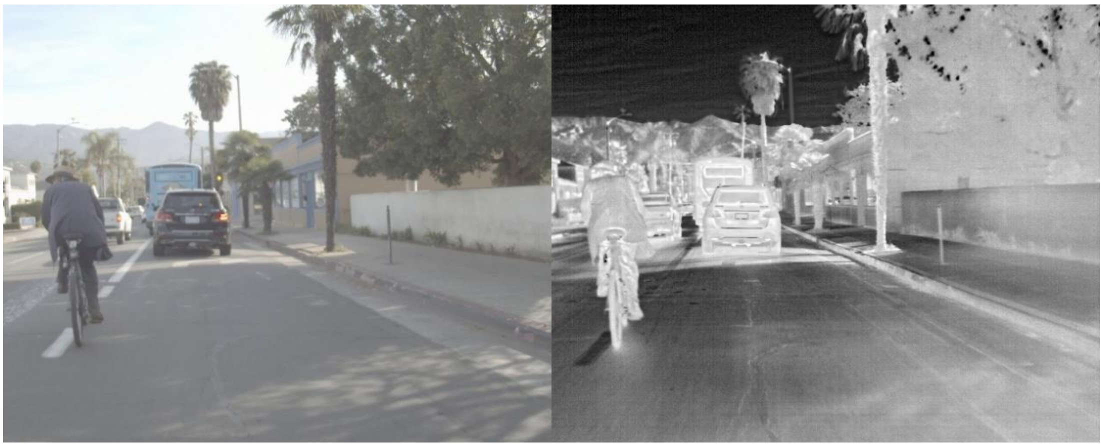

Figure 6.1 : Example RGB and IR image from the dataset.

Although the dataset is free to use, to access the dataset, its license agreement must be agreed upon. Then, it can be downloaded from the directed address.

Since this is a licensed dataset, I cannot give a direct link to the dataset.

6.1.1 Pre-processing

RGB images in the dataset were captured by different cameras, so they have different field of view (FOV) angles and different sizes. Thus, to empirically register the images, some of them were cropped and resized so that all of them have 640 by 512 pixels. Cropping values were chosen empirically by comparing the corresponding IR images.

6.1.2 Image segmentation

As mentioned earlier, material type contains significant information to translate RGB image to IR image. Hence, to improve the performance of the model, semantic segmentation of the RGB images can also be given as concatenated with the RGB images.

To obtain a segmented version of the dataset, PSPNET[2] pre-trained on the Cityscapes dataset was used. Object types and scenes in the training dataset and Cityscapes dataset are very similar so obtained segmentations are expected to be accurate.

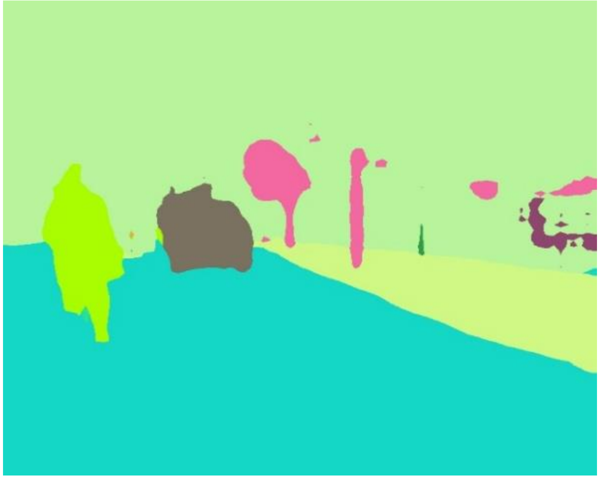

Figure 6.2 : Semantic segmentation of Figure 6.1.

To run this segmentation network, a ready-to-use Python module called keras-segmentation was preferred. An example segmented image is shown in Figure 6.2.

6.2 Implementation

The code mainly was written with the object-oriented understanding i.e attention has been paid to encapsulation, inheritance, abstraction, etc. Additionally, it was designed to continue the training of the partially-trained model. This is particularly useful when there is limited computational power.

---

[2] Pyramid Scene Parsing Network: A semantic segmentation network proposed by Zhao et al. in 2017 [16].

All of the codes including the pre-processing and segmentation scripts were uploaded to the project Github page [23]. To use it, one can clone the code to his/her machine with the following command.

```
!git clone https://github.com/eneserdo/RGB-to-IR-Translation-with-GAN
```

Then, he must download the dataset from the FLIR's website. Afterward, to resize and to get a segmented dataset, the following commands should be run. Notice that "RGB_dir" and "IR_dir" are used as a placeholder to refer to the directory files.

```
!python img_resize.py -rgb RGB_ dir -ir IR_ dir
```

```
!python img_segment.py -i RGB _dir
```

Finally, to train the model for 100 epochs with the segmented dataset, run the following command

```
!python train.py  -te 100 -seg True
```

To see the other parameters and their usage:

```
!python train.py -h
```

The exclamation mark (!) at the beginning of every command is only required when working in the notebook environment, otherwise, it can be ignored.

6.3 Training Details

Weights of both networks were initialized from a Gaussian distribution with 0 mean and 0.02 standard deviation, and biases are set to zero. Then, the model trained for 100 epochs with a batch size of 8. To optimize the parameters the Adam optimizer with momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$ is used. The learning rate was 2e-4 and 4e-5 for generator and discriminator respectively to train both balancedly.

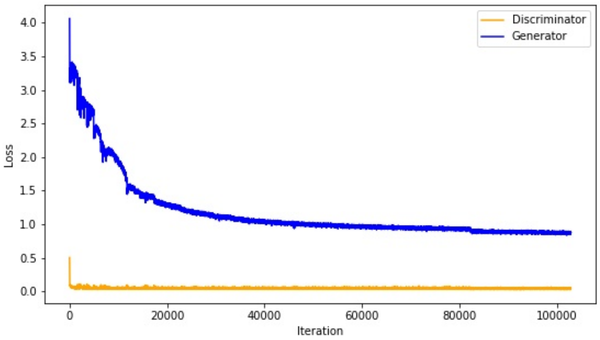

Figure 6.3 : Overall loss change over the iterations.

However, as shown in Figure 6.3, still the loss of the discriminator dropped immediately. The reason behind this immediate drop could be the adaptive learning rate adjustment ability of the Adam optimizer.

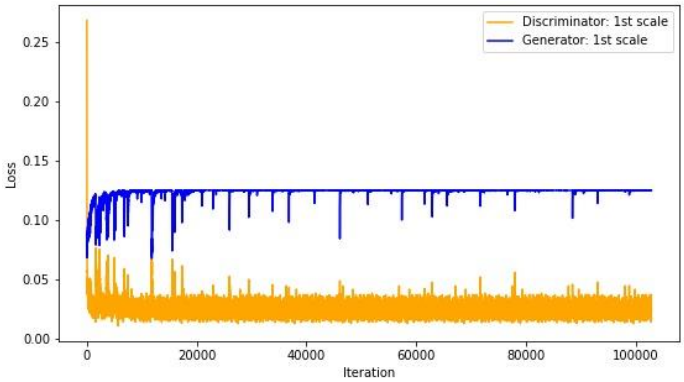

Figure 6.4 : Loss change for the first scale images.

Feature matching loss coefficient, λ_{FM} and perceptual loss coefficient λ_P were set to 10.

Due to the limited computational power, all images were resized to 320 x 256 pixels in runtime.

It can be concluded that the decrease of the loss mainly due to the feature matching and perceptual loss, other losses were not minimized well.

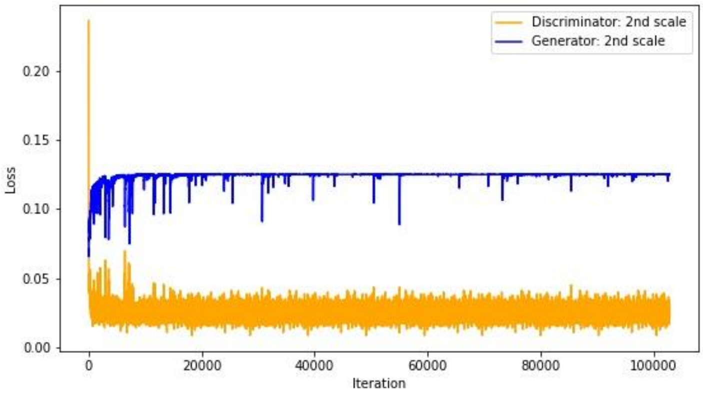

Figure 6.5 : Loss change for the second scale images.

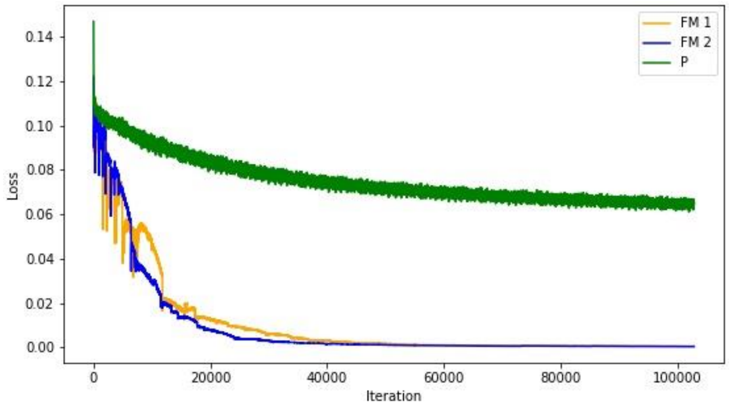

Figure 6.6 : Loss change of the perceptual loss and feature matching loss (2 scales were shown separately).

Contrary to other subfields of deep learning, in general, there is no generally agreed-upon evaluation metric for GANs — although there are some for different GAN types. Thus, comparing two different training configurations or comparing the results of two models that have different trained for a different number of epochs is a challenging task. So, it is hard to say where to stop the training. Furthermore, for GANs, loss itself is not a direct measure of the visual quality of the result and the accuracy of the translation. As a result, the quality of the results can only be evaluated by visual examination manually.

Note that to smoothen the noise in loss graphs, all of the graphs were filtered with the exponential moving average with $\alpha = 0.1$

6.4 Results

To run the test script:

!python test.py -ce 100 –segment True

Four example results were shown in Figure 6.7 to 6.10. Note that or all figures, 1st row: predicted IR, Ground truth IR, and 2nd row: RGB, segmented RGB.

As can be seen, two types of distortion in the images can be noticed — ghosting and a kind of fixed-pattern noise. Also, the image segmentation network did not work perfectly, so it affected the performance of the model.

For more example results, see Appendix A.

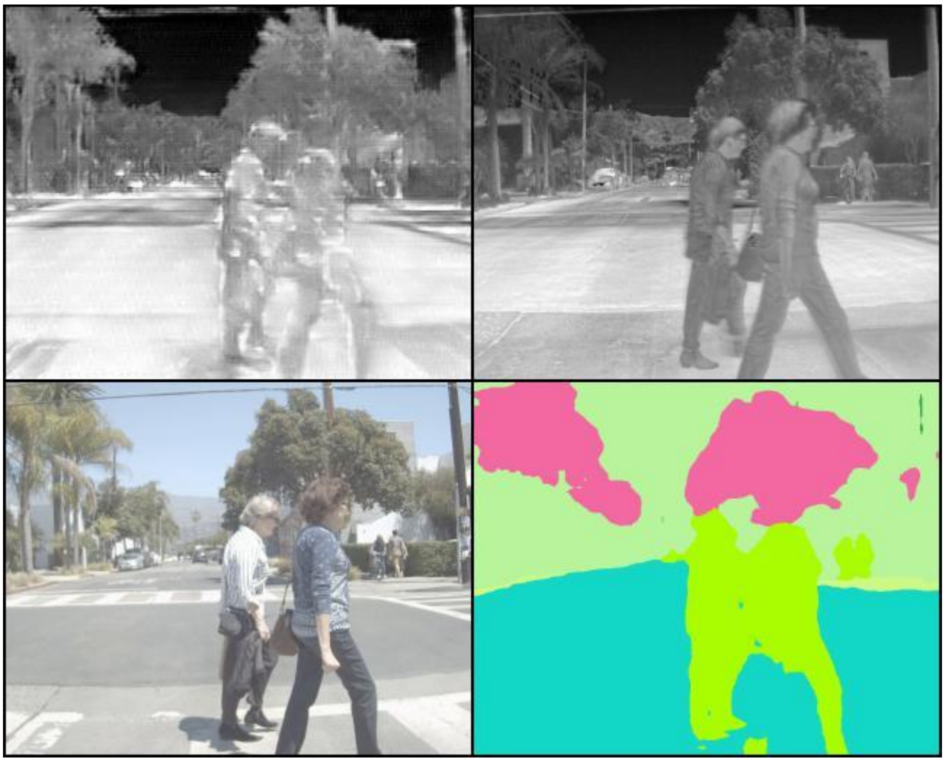

Figure 6.7 : Result image.

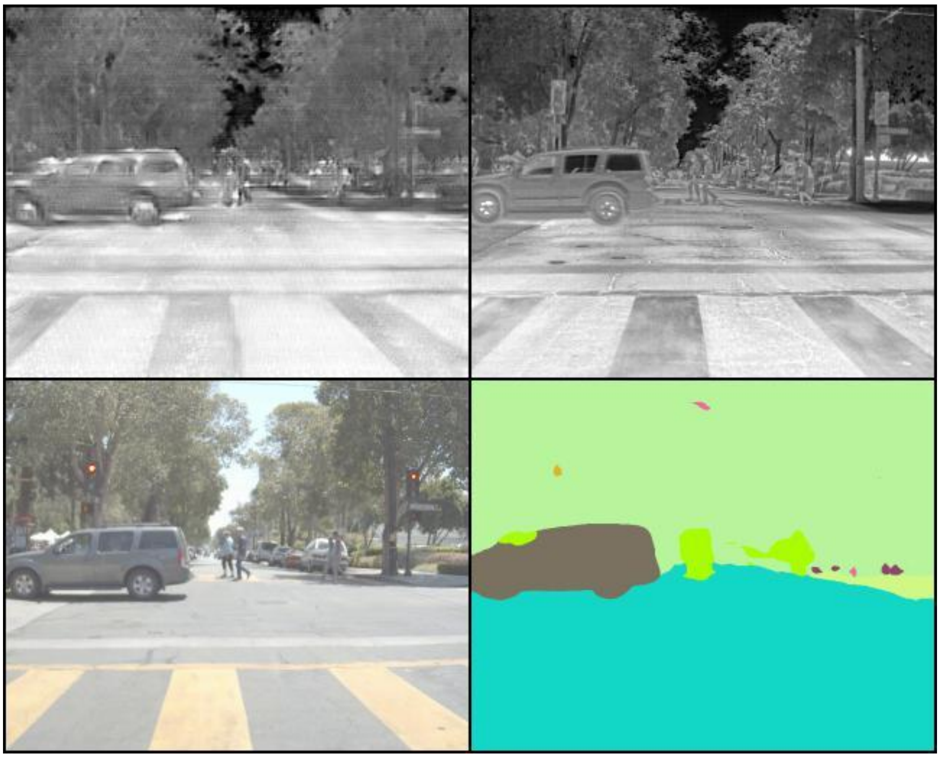

Figure 6.8 : Result image.

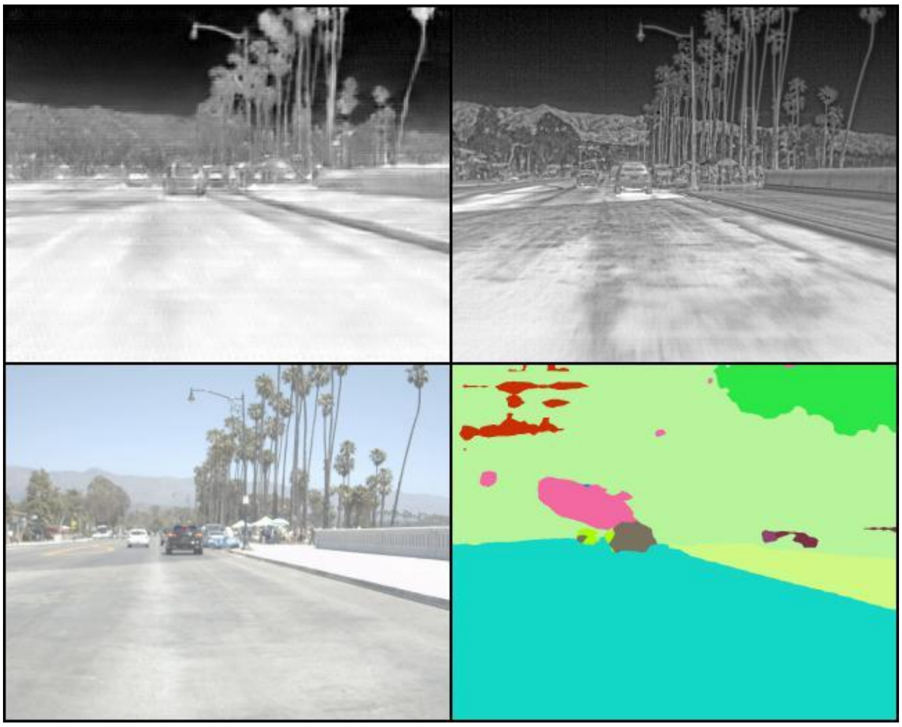

Figure 6.9: Result image.

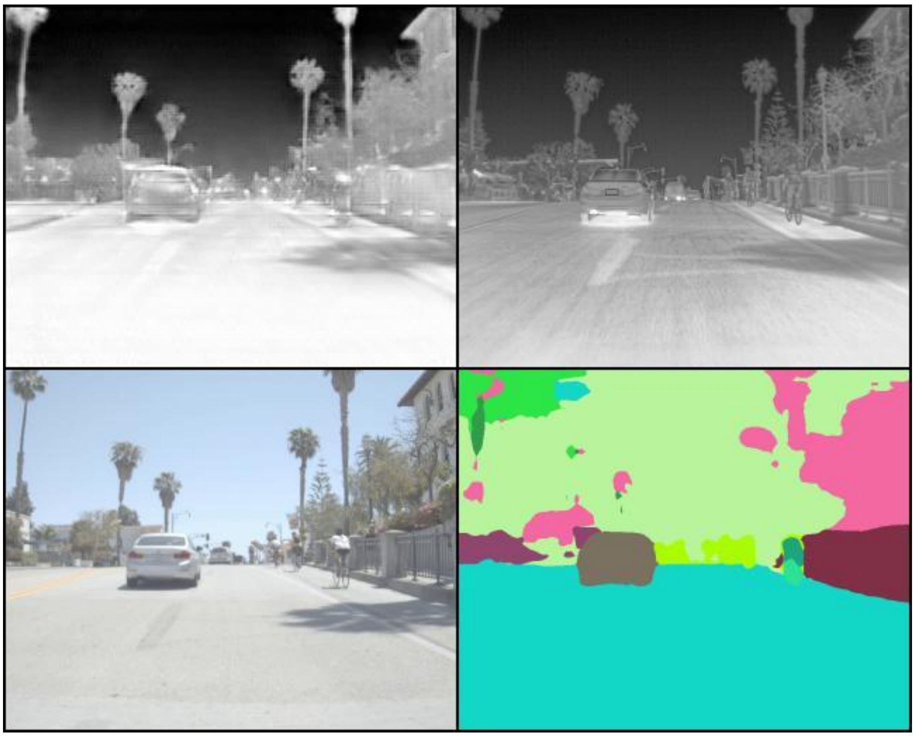

Figure 6.10: Result image.

7. REALISTIC CONSTRAINTS AND CONCLUSIONS

7.1 Practical Application of this Project

IR imaging has very crucial implementations in various engineering required fields such as flame detection, surveillance, UAVs, search and rescue operations etcetera. To improve their abilities, most of these systems have work with artificial intelligence (AI) in the background. And, due to the nature of AI, these systems need a huge amount of data for training. However, IR images are hard to obtain due to the high cost of IR cameras. And even with a camera, obtaining images for specific scenarios could be very time-consuming. To meet the needs of our target group-AI developers, we want to create a model that has the capability to translate a wide range of RGB images to IR images with high quality and high accuracy.

7.2 Realistic Constraints

To explain the reasoning behind this project, it can be said that in principle, to turn the RGB image to IR image, it is assumed that IR information is hidden inside the original image, and it is waiting to be extracted. Since the features of an object in an IR image depend on its temperature and emissivity, this mentioned prior information could be the material type that can be acquired from the RGB image. This is because emissivity is a characteristic of material type. Also, the temperature of the object could be identifier by assuming the lightning conditions are similar.

On the other hand, the main limitation of the project comes when there is no prior information exists. For instance, RGB images of obstacles e.g. a picture of a wall does not contain any information about the objects behind the wall which was normally detectable by a real IR camera. In this sense, it is important to point out that this project aims to translate RGB images to IR images only in case of required information exists. Thus, the main limitation is to find an appropriate dataset.

Also, limited computational power did not let us do many training experiments.

Finally, the dataset that was selected is not very diverse and there was an alignment problem.

7.2.1 Social, environmental and economic impact

This project, if successful, tries to make AI developers to create their own IR image dataset without having a real IR camera. So, its main benefit is that it can help them to save money.

There is either none or very little social and environmental impact.

7.2.2 Cost analysis

The project requires high computational power to train the network, but instead of a decent computer, cloud computing services can be used.

Main cost items:

- Google Cloud Platform - Deep Learning Instance: 300 Dollars per month
- Junior Engineer Salary: 600 Dollars per month

Thus, for a 28-week project total cost will be 5900 $

Note that I used Google Colab Pro services to train and test the model. It cost 10 dollar per month, which is a very budget-friendly alternative.

7.2.3 Standards

In the required parts, IEEE standards were followed. Also, since implementation was done in Python language, PEP standards were considered.

7.2.4 Health and safety concerns

Regarding the project, there is neither a health nor safety issue.

7.3 Future Work and Recommendations

To improve the results, there are many hyper-parameters to tune. Also, loss function can be changed to Wasserstein loss or another loss in the literature. Additionally, since Adam has adaptive learning rate ability, controlling the learning rate did not result in as expected, so other optimization functions could be used.

Another issue is to understand the importance of each component of the loss function by doing an ablation study. For instance, the results of the two models trained with and without the perceptual loss can be compared.

Lastly, the dataset at the hand is not very comprehensive. Training with different datasets may improve the results.

REFERENCES

[1] **I. Goodfellow**, “NIPS 2016 tutorial: Generative adversarial networks,” *arXiv*. arXiv, Dec. 31, 2016, Accessed: Jan. 23, 2021. [Online]. Available: http://www.iangoodfellow.com/slides/2016-12-04-NIPS.pdf.

[2] **T. Salimans**, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved Techniques for Training GANs,” *Adv. Neural Inf. Process. Syst.*, pp. 2234–2242, Jun. 2016, Accessed: Jan. 28, 2021. [Online]. Available: http://arxiv.org/abs/1606.03498.

[3] **F. Huszár**, “How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?,” Nov. 2015, Accessed: Jan. 29, 2021. [Online]. Available: http://arxiv.org/abs/1511.05101.

[4] **M. Arjovsky** and L. Bottou, “Towards Principled Methods for Training Generative Adversarial Networks,” *arXiv*, Jan. 2017, Accessed: Jan. 28, 2021. [Online]. Available: http://arxiv.org/abs/1701.04862.

[5] **M. Arjovsky**, S. Chintala, and L. Bottou, “Wasserstein GAN,” *arXiv*, Jan. 2017, Accessed: Jan. 29, 2021. [Online]. Available: http://arxiv.org/abs/1701.07875.

[6] **I. Gulrajani**, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved Training of Wasserstein GANs,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-December, pp. 5768–5778, Mar. 2017, Accessed: Jan. 29, 2021. [Online]. Available: http://arxiv.org/abs/1704.00028.

[7] **T. C. Wang**, M. Y. Liu, J. Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2018, pp. 8798–8807, doi: 10.1109/CVPR.2018.00917.

[8] **X. Mao**, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “Least Squares Generative Adversarial Networks,” *Proc. IEEE Int. Conf. Comput.*

Vis., vol. 2017-October, pp. 2813–2821, Nov. 2016, Accessed: Apr. 16, 2021. [Online]. Available: http://arxiv.org/abs/1611.04076.

[9] **T. Karras**, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” *arXiv*. arXiv, Oct. 27, 2017, Accessed: Jan. 29, 2021. [Online]. Available: https://youtu.be/G06dEcZ-QTg.

[10] **T. Karras**, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2020, pp. 8107–8116, doi: 10.1109/CVPR42600.2020.00813.

[11] **P. Isola**, J. Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-January, pp. 5967–5976, doi: 10.1109/CVPR.2017.632.

[12] **Y. Iwashita**, K. Nakashima, S. Rafol, A. Stoica, and R. Kurazume, “MU-net: Deep learning-based thermal IR image estimation from RGB image,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2019, vol. 2019-June, pp. 1022–1028, doi: 10.1109/CVPRW.2019.00134.

[13] **A. Odena,** V. Dumoulin, and C. Olah, “Deconvolution and Checkerboard Artifacts,” *Distill*, vol. 1, no. 10, p. e3, Oct. 2016, doi: 10.23915/distill.00003.

[14] **J. Johnson,** A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Mar. 2016, vol. 9906 LNCS, pp. 694–711, doi: 10.1007/978-3-319-46475-6_43.

[15] **D. Ulyanov**, A. Vedaldi, and V. Lempitsky, “Instance Normalization: The Missing Ingredient for Fast Stylization,” Jul. 2016, Accessed: Apr. 18, 2021. [Online]. Available: http://arxiv.org/abs/1607.08022.

[16] **H. Zhao**, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-January, pp. 6230–6239, doi: 10.1109/CVPR.2017.660.

[17] “Stefan-Boltzmann Law.” http://hyperphysics.phy-astr.gsu.edu/hbase/thermo/stefan.html (accessed Apr. 08, 2021).

[18] “Infrared - Wikipedia.” https://en.wikipedia.org/wiki/Infrared (accessed Apr. 11, 2021).

[19] “Thermal Camera Specs You Should Know Before Buying | FLIR Systems.” https://www.flir.eu/discover/professional-tools/thermal-camera-specs-you-should-know-before-buying/ (accessed Apr. 08, 2021).

[20] “Manifold Hypothesis Definition | DeepAI.” https://deepai.org/machine-learning-glossary-and-terms/manifold-hypothesis (accessed Jan. 28, 2021).

[21] “Unsupervised Learning - Artificial Inteligence.” https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/unsupervised_learning (accessed Jan. 29, 2021).

[22] “FREE - FLIR Thermal Dataset for Algorithm Training | FLIR Systems.” https://www.flir.com/oem/adas/adas-dataset-form/ (accessed Apr. 19, 2021).

[23] “eneserdo/RGB-to-IR-Translation-with-GAN: Translational GAN model.” https://github.com/eneserdo/RGB-to-IR-Translation-with-GAN (accessed Jun. 18, 2021).

APPENDICES

APPENDIX A: More Results

APPENDIX A

For all figures, 1st row: predicted IR, Ground truth IR, and 2nd row: RGB, segmented RGB.

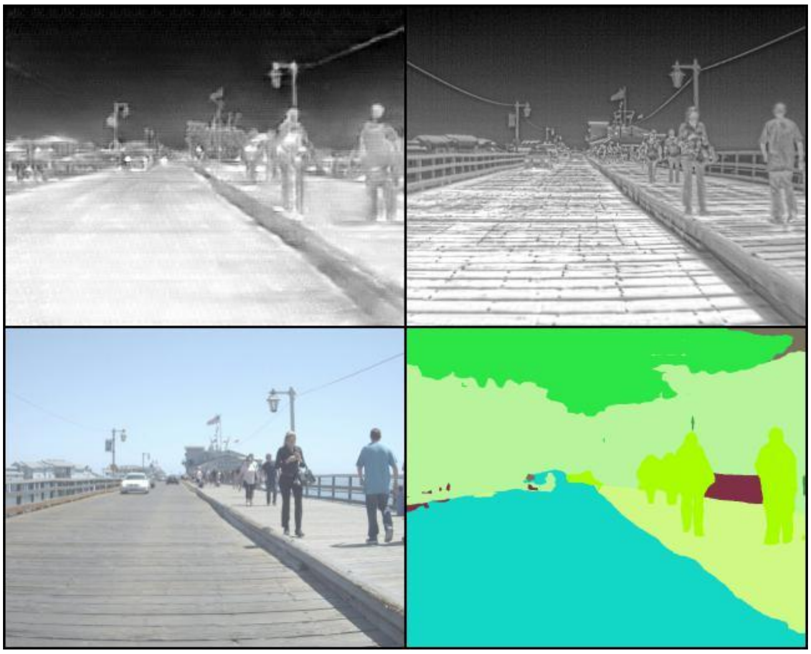

Figure A.1

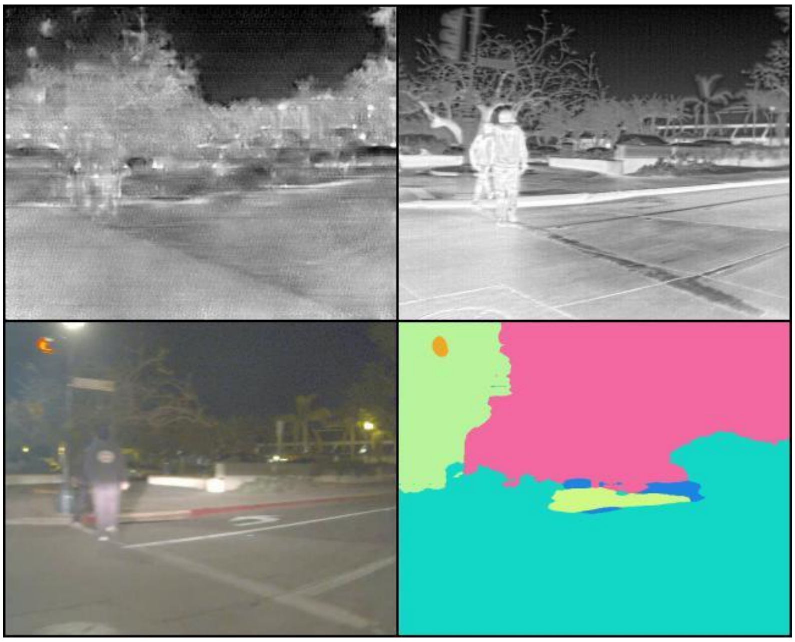

Figure A.2

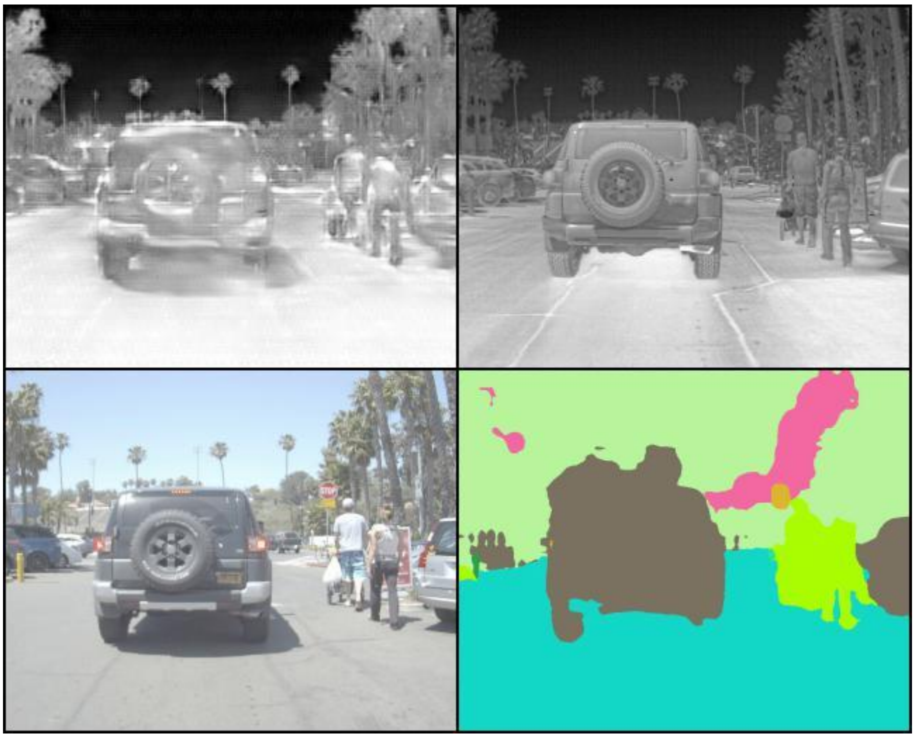

Figure A.3

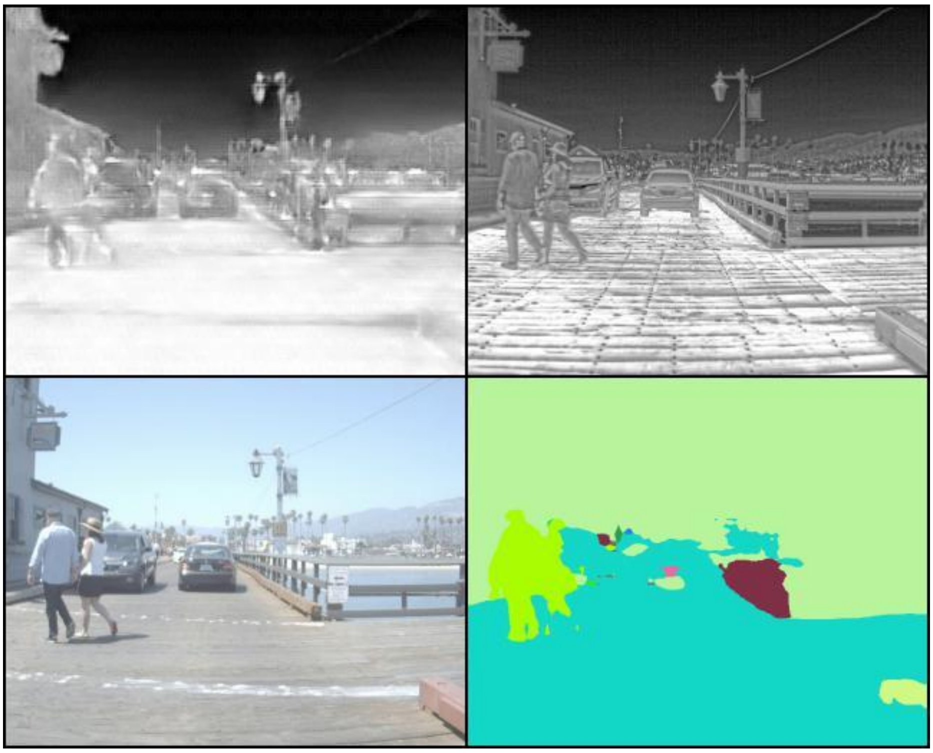

Figure A. 4

CURRICULUM VITAE

Name Surname : **Enes Erdoğan**

Place and Date of Birth : **Konya, May 25th 1998**

E-Mail : **erdogane16@itu.edu.tr**

Enes Erdoğan is currently a senior year student at Electronics and Communication Engineering in Istanbul Technical University. He worked as an intern at ASELSAN A. Ş. and Baykar Defence. He mainly interested in deep learning and image processing.