A PROJECT REPORT ON

Astronomical Image colorization and super-resolution using GANs

SUBMITTED TOWARDS THE
PARTIAL FULFILMENT OF THE REQUIREMENTS OF

BACHELOR OF ENGINEERING (Computer Engineering)

BY

Shreyas Kalvankar	Exam No:
Hrushikesh Pandit	Exam No:
Pranav Parwate	Exam No:
Atharva Patil	Exam No:

Under The Guidance of

Prof. Dr. S.M. Kamalapur

Department of Computer Engineering
K. K. Wagh Institute of Engineering Education & Research
Hirabai Haridas Vidyanagari, Amrutdham, Panchavati,
Nashik-422003
Savitribai Phule Pune University

A. Y. 2020-21 Sem I

K. K. Wagh Institute of Engineering Education and Research
Department of Computer Engineering

CERTIFICATE

This is to certify that the Project Titled

Astronomical Image colorization and super-resolution using GANs

Submitted by

Shreyas Kalvankar	Exam No:
Hrushikesh Pandit	Exam No:
Pranav Parwate	Exam No:
Atharva Patil	Exam No:

is a bonafide work carried out by Students under the supervision of Prof. Dr. S.M. Kamalapur and it is submitted towards the partial fulfilment of the requirement of Bachelor of Engineering (Computer Engineering) Project during academic year 2020-21.

Prof. Dr. S.M. Kamalapur
Internal Guide
Department of Computer Engineering

Prof. Dr. S. S. Sane
Head
Department of Computer Engineering

Abstract

Automated colorization of black and white images has been subject to much research within the computer vision and machine learning communities. Beyond simply being fascinating from an aesthetic and artificial intelligence perspective, such capability has broad practical applications. It is an area of research that possesses great potentials in applications: from black and white photo reconstruction, image augmentation, video restoration to image enhancement for improved interpretability.

Image downscaling is an innately lossy process. The principal objective of super resolution imaging is to reconstruct a low resolution image into a high resolution one based on a set of low-resolution images to rectify the limitations that existed while the procurement of the original low-resolution images. This is to insure better visualization and recognition for either scientific or non-scientific purposes. No matter how good an upscaling algorithm is, there will always be some amount of high frequency data lost from a downscale-upscale function performed on the image. Ultimately, even the best upscaling algorithms cannot effectively reconstruct data that does not exist. Traditional methods for image upsampling rely on low-information, smooth interpolation between known pixels. Such methods can be treated as a convolution with a kernel encoding no information about the original image. A solution to the problem is by using Generative Adversarial Networks (GANs) to hallucinate high-frequency data in a super-resolved image that does not exist in the smaller image. Although they increase the resolution of an image, they fail to produce the clarity desired in the super-resolution task. By using the above mentioned method, not a perfect reconstruction can be obtained albeit instead a rather plausible guess can be made at what the lost data might be, constrained to reality by a loss function penalizing deviations from the ground truth image. A huge number of raw images lie unprocessed and unseen in the Hubble Legacy Archives. These raw images are typically low-resolution, black and white and unfit to be shared with the world. It takes huge amounts of hours to process them. This processing is necessary because astronomers often struggle to distinguish objects from the raw images. Random and synthetic noise from the sensors in the telescope, changing optical characteristics in

the system and noise from other bodies in the universe all make the processing further necessary. Furthermore, colorization is needed to help highlight small features that ordinarily wouldn't be able to be picked out against noise of the image. The processing of the images is so time consuming that the images are rarely seen by human eyes. The problem is only likely to get worse. Not only is new data being continuously produced by Hubble Telescope, but new telescopes are soon to come online. A simplification of image processing by using artificial image colorization and super-resolution can be done in an automated fashion to make it easier for astronomers to visually identify and analyze objects in Hubble dataset.

Acknowledgments

please enter text here.

Shreyas Kalvankar
Hrushikesh Pandit
Pranav Parwate
Atharva Patil
(B.E. Computer Engg.)

INDEX

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 PROJECT IDEA

- The idea of the project is to create a efficient mathematical model for image colorization and super resolution using Generative Adversarial Networks (GANs)
- Having two networks compete will stimulate greater performance by the virtue of minimization of loss functions that traditional Convolutional Neural Network cant do

1.2 MOTIVATION OF THE PROJECT

- Image colorization seems to be evolving as computers get better and better at predicting missing variables. Different methods and techniques have been applied for colorizing images that have shown promising results
- Introduction of convolutional neural networks has made this task even more precise and accurate. Convoluting grayscale images to RGB provides unprecedented results with reference to visual inspection
- With the introduction of Generative Adversarial Networks, this particular task can be developed and modified to increase the efficiency and precision for colorizing images instead
- Furthermore, image upscaling is another problem that has been under research in the computer vision community. It has been studied and many approached have been developed to accurately predict the missing pixel values while upscaling an image
- Application of GANs to this discipline has successfully improved the performance and computers are getting better and better at predicting accurate missing pixel values and upscaling images many folds the original size
- All this computation power can be used for astronomical research by processing large data archives

- A large number of images lie dormant in most of the space survey data archives which never go through any kind of processing and are low resolution and black & white. These images could be processed automatically by an algorithm that will colorize and super-resolve the images which can make it easier for astronomers to visually inspect the images

1.3 LITERATURE SURVEY

1.3.1 Image Colorization

1.3.1.1 Hint Based Colorization

[2] proposed using colorization hints from the user in a quadratic cost function which imposed that neighboring pixels in space-time with similar intensities should have similar colours. This was a simple but effective method but only had hints which were provided in form of imprecise colored scribbles on the grayscale input image. But with no additional information about the image, the method was able to efficiently generate high quality colorizations. [3] addressed the color bleeding issue faced in this approach and solved it using adaptive edge detection. [4] used luminescence based weighting for hints to boost efficiency. [5] extended the original cost function to enforce color continuity over similar textures along with intensities.

[6] had proposed another approach that reduced the burden on the user by only requiring a full color example of an image with similar composition. It matched the texture and luminescence between the example and the target grayscale image and received realistic results as long as the example image was sufficiently similar.

Regardless of the scribble based or example based approach, the algorithms still needed sufficient human assistance in form of hand drawn or colorized images.

1.3.1.2 Deep Colorization

Owing to recent advances, the Convolutional Neural Networks are a de facto standard for solving image classification problems and their popularity continues to rise with continual improvements. CNNs are peculiar in their ability to learn and differentiate colors, patterns and shapes within an image and their ability to associate

them with different classes.

[7] proposed a per pixel training for neural networks using DAISY [8], and semantic [9] features to predict the chrominance value for each pixel, that used bilateral filtering to smooth out accidental image artifacts. With a large enough dataset, this method proved to be superior to the example based techniques even with a simple Euclidean loss function against the ground truth values.

Finally, [10] successfully implemented a system to automatically colorize black & white images using several ImageNet-trained layers from VGG-16 [11] and integrating them with auto-encoders that contained residual connections. These residual connections merged the outputs produced by the encoding VGG16 layers and the decoding portion of the network in the later stages. [12] showed that deeper neural networks can be trained by reformulating layers to learn residual function with reference to layer inputs. Using this *Residual Connections*, [12] created the *ResNets* that went as deep as 152 layers and won the 2015 ImageNet Challenge.

1.3.1.3 Generative Adversarial Networks

[13] introduced the adversarial framework that provides an approach to training a neural network which uses the generative distribution of $p_g(x)$ over the input data x.

Since it's inception in 2015, many extended works of GAN have been proposed over years including DCGAN [14], Conditional-GAN [15], iGAN [16], Pix2Pix [17].

[14] applied the adversarial framework for training convolutional neural networks as generative models for images, demonstrating the viability of *deep convolutional generative adversarial networks*.

DCGAN is the standard architecture to generate images from random noise. Instead of generating images from random noise, Conditional-GAN [15] uses a condition to generate output image. For e.g. a grayscale image is the condition for colorization of image. Pix2Pix [17] is a Conditional-GAN with images as the conditions. Besides learning the mapping from input image to output image, it can also learn a separate loss function to train this mapping. Pix2Pix is considered to be the

state of the art architecture for image-image translation problems like colorization.

1.3.2 Image Upscaling

1.3.2.1 Frequency-domain-based SR image approach

[18] proposed the frequency domain SR method, where SR computation was considered for the noise free low resolution images. They transformed the low resolution images into Discrete Fourier transform (DFT) and further combined it as per the relationship between the aliased DFT coefficient of the observed low resolution image and that of unknown high resolution image. Then the output is transformed back into the spatial domain where a higher resolution is now achieved.

While Frequency-domain-based SR extrapolates high frequeny information from the low resolution images and is thus useful, however they fall short in real world applications.

1.3.2.2 The interpolation based SR image approach

The interpolation-based SR approach constructs a high resolution image by casting all the low resolution images to the reference image and then combining all the information available from every image available. The method consists of the following three stages (i) the registration stage for aligning the low-resolution input images, (ii) the interpolation stage for producing a higher-resolution image, and (iii) the deblurring stage for enhancing the reconstructed high-resolution image produced in the step ii).

However, as each low resolution image adds a few new details before finally deblurring them, this method cannot be used if only a single reference image is available.

1.3.2.3 Regularization-based SR image approach

Most known Bayesian-based SR approaches are maximum likelihood (ML) estimation approach and maximum a posterior (MAP) estimation approach.

While [19] proposed the first ML estimation based SR approach with the aim to find the ML estimation of high resolution image, some proposed a MAP estimation

approach. MAP SR tries to takes into consideration the prior image model to reflect the expectation of the unknown high resolution image.

1.3.2.4 Super Resolution - Generative Adversarial Networks (SR-GAN)

The Genrative Adversarial Network [13], has two neural networks, the Generator and the Discriminator. These networks compete with each other in a zero-sum game. [20] introduced SRGAN in 2017, which used a SRResNet to upscale images with an upscaling factor of 4x. SRGAN is currently the state of the art on public benchmark datasets.

CHAPTER 2

PROBLEM DEFINITION AND SCOPE

2.1 PROBLEM STATEMENT

The problem can be divided into two sub-problems:

- Create an efficient model to colorize grayscale images
- Take a colorized image and upscale it n times the original size

2.1.1 Goals and objectives

Goal and Objectives:

- Auto-Colorization:
  - The first model will be given input a grayscale, low resolution image of dimensions $(64 \times 64 \times 1)$
  - The model will perform a series of mathematical operations that will increase the channel width of the image from 1 (single channel grayscale image) to 3 (RGB)
  - The output of the model will be a colorized version of the input image with dimensions $(64 \times 64 \times 3)$
- Upscaling/super-resolution:
  - The input to the model will be a colorized image of shape $(64 \times 64 \times 3)$
  - The model will increase the dimensions of the image from (64×64) to $((64 \cdot n) \times (64 \cdot n))$ by performing a series of upscaling operations and predicting information that may be lost while downscaling
  - The output of the model will be an upscaled RGB image with dimensions $((64 \cdot n) \times (64 \cdot n) \times 3)$
- The models may be combined to form a single model that will take a low resolution, grayscale image as its input and produce a high resolution, colorized image as its output

2.1.2 Statement of scope

- The model will consist of neural networks implemented using deep learning frameworks that will accept images of input format *JPEG*
- The input will be grayscale images of size 64×64
- Input bounds:
  - Lower bound: $64 \times 64 \times 1$
  - Upper bound: no limit
- The output will be produced in two phases:
  - A colorized output of model 1 with shape $64 \times 64 \times 3$
  - A upscaled output of model 2 from the colorized output of model 1 with shape $(64 \cdot n) \times (64 \cdot n) \times 3$
- The model will:
  - take input black & white images
  - produce colorized images of the same size
  - produce upscaled images of size *n* times the input size (currently 64)
- The model will **not**:
  - take a colorized image as an input
  - take an image of size less than (64×64) in size
  - produce accurate upscaling or coloring albeit merely make a guess at what the lost values might be

2.2 MAJOR CONSTRAINTS

- The astronomical image data required for training purposes is mostly raw. There exists no structured dataset that is already cleaned. The unavailability of a dataset is a major constraint for the project

- Scraped data from the archives is noisy and requires heavy processing and cleaning in order to be usable by the model
- The images available for download are of low resolution, which sets an upper bound on the maximal upscale factor
- The image data is large and needs high computation power to process
- The data needs to be cleaned manually as there exist no methods to automatically do this particular task
- The model involves neural networks which heavily rely on computation power for its training. The hardware required for training is not readily available because of absence of a workstation supporting heavy computations
- The training part requires large amount of memory
- Absence of an NVIDIA workstation GPU will slow down the training further

2.3 METHODOLOGIES OF PROBLEM SOLVING AND EFFICIENCY ISSUES

- Data gathering and processing
  - Data Scraping
    - Owing to unavailability of a dataset, raw data can be acquired by the means of web scraping
    - Images from the snapshots of entire night sky can be obtained in such a way from the Hubble Legacy Archive
  - Data Cleaning
    - The scraped data consists of snapshots of the entire night sky with 1 degree deviation of the telescope
    - This results in large amount of noisy, overexposed, irregular data images

    - This data needs to be cleaned manually before it can be used for any kind of study

- Image colorization
  - The problem of image colorization has been solved using multiple methodologies
  - [10] used convolutional neural networks with residual encoders using the VGG16 architecture
  - Though the system performs extremely well in realistically colorizing various images, it consisted of $L2$ loss which was a function of the Euclidean distance between the pixel's blurred color channel value in the target and predicted image

$$L2loss = \sum_{i=1}^{n} (y_{true} - y_{predicted})^2 \tag{2.1}$$

  - This is a regression based approach and the pixel-wise L2 loss will impose an averaging effect over all possible candidates and will result in dimmer and patchy colorization
  - Generative Adversarial Networks introduced by [13] use a minimax loss which is different than the $L2$ loss as it will choose a color to fill an area rather than averaging. This is similar to a classification based approach

- Image Upscaling
  - One of the most popular approach to image upscaling was sparse-coding. This approach assumes that images can be sparsely represented by a dictionary of atoms in some transformed domain [21]. The dictionary is learned during the training process.
  - The main drawback for this was that the optimization algorithm was computationally expensive
  - Dong et. al explored super-resolution using convolutional neural network and calling it SRCNN [22]. They explained how CNN had many

similarities to the sparse-coding-based super-resolution.

  - Kim et. al improved upon SRCNN's results using their very own model inspired from the VGG-net architecture[23].
  - After the introduction of GANs, Ledig et. al applied them to super-resolution (SRGAN) using a network inspired by the ResNets [20][12].
  - SR-GAN works well with for single image super-resolution as it also uses an intelligent content loss function that uses pre-trained VGG-net layers. However, Ledig et. al noted that further information could be used if this network were to be adapted to a video, such as temporal information.

- A generative network, G, is meant to learn the underlying distribution of a data set, Y. For e.g. we can train a GAN over face images to generate images similar to those faces. With just a generative network however, we must visually assess the quality of network outputs and judge how we can adapt the network to produce more convincing results.

- With a discriminative network D, we can incorporate this tweaking directly into training. The discriminative network takes in both fabricated inputs generated by G and the real inputs from Y. It's sole purpose is to classify if the input has come from G or Y.

- The key idea is back propagation of the gradients from the results of D's classification to G so that G gets better at producing images and in turn fooling D.

- For the project, we split the data into two categories: X that serves as the data for the Y, which are its corresponding labels.

- G_1 takes in a low resolution $x \in X$ which is black & white and produces $\hat{y}$, a colorized version of x. The descriminator D, in turn takes in a colorized image and outputs the probability that the image comes from Y, instead of as outputs from G, $G(x)$. As such, if the discriminator is fooled by out generator,

it should output a probability greater than 0.5 for the set of inputs coming from $G(x)$ and a probability less than 0.5 for images coming from Y.

- The same is the process for generator G_2 with the only difference being that the X is the set of colorized images but having low resolution and Y is the set of high resolution images that serve as the labels for underlying mapping of X. G_2 takes in the low resolution image $x \in X$ and produces $\hat{y}$ and the discriminator outputs a probability determining whether the image is super-resolved by G_2 or the ground truth images from Y.

2.4 SCENARIO IN WHICH MULTI-CORE, EMBEDDED AND DISTRIBUTED COMPUTING USED

A deep learning algorithm is a software construct that has certain steps that may be hardware intensive. Generative Adversarial Networks require huge amount of computing prowess to complete multiple passes of forward and backward propagation in order to train themselves. A network may consist of millions and billions of parameters which are associated with hundreds of thousands of graph nodes. To actually be able to train a network with more than a billion parameters, we need appropriately large amount of memory. Furthermore, the operations of forward and backward propagation are mathematical operations that adjust the parameters based on the gradient of the cost function to minimize the cost. This calculation, although heavy, is independent of each node and can be performed in a parallel framework. NVIDIA CuDA enabled GPUs have a CuDNN (CuDA Deep Neural Network) library that hooks the training algorithm onto the GPU memory for processing, deploying thousands and hundreds of thousand parallel threads to perform independent calculations of optimizing gradients. Such an infrastructure is expensive and requires a dedicated set up for running deep learning algorithms. For normal use cases, one can run into the problems of memory overflows while allocating tensors in the process of creation of graphs. In such cases, it is costly to buy more GPUs. One can make use of cloud services provided by Google Colab, AWS, Azure and more. These services can host runtimes that will allow users to run their deep learning algorithms over their hardware which will ensure fast and efficient training.

2.5 OUTCOME

- An efficient mathematical model to be created which will describe mappings required to colorize and super-resolve low resolution grayscale images
- A brief albeit descriptive study of different approaches towards image colorization and super-resolution
- Study presenting the benefits of certain GAN architectures and their edge over other kinds of neural networks in image colorization and super-resolution

2.6 APPLICATIONS

- Currently, given the number of the raw and unprocessed images in Hubble Legacy Archives, much of the images are not workable for scientific evaluation. The main application of building a GAN and automating the upscaling and colorization of these images is to help in visual classification for astronomers. Through a high resolution and colourized image, astronomical objects which would've been imperceptible to the human eye could be now visible for visual inspection. While upscaling is expected to address the poor quality of the original images, colourization will help distinguish astronomical objects and activites from the noise generated by various factors.

2.7 HARDWARE RESOURCES REQUIRED

The project is based Machine Learning, and the use of Tensorflow-GPU brought forward the need for a very high end hardware. Google Colab (or Colaboratory) is a free Jupyter notebook environment offered by Google which runs notebooks from Python kernels and uses Google Drive for storage.

Sr. No.	Parameter	Minimum Requirement	Justification
1	GPU type	NVIDIA CuDA enabled GPU	Training the model
2	GPU memory	>6 GB	Batch training

Table 2.1: Hardware Requirements

2.8 SOFTWARE RESOURCES REQUIRED

Platform :

1. Operating System: Windows/Linus
2. IDE: Jupyter Notebook
3. Programming Language: python3, javascript
4. Frameworks: Node.js, Tensorflow, plotting libraries, openCV

CHAPTER 3

PROJECT PLAN

3.1 PROJECT ESTIMATES

Use Waterfall model and associated streams derived from assignments 1,2, 3, 4 and 5(Annex A and B) for estimation.

3.1.1 Reconciled Estimates

3.1.1.1 Cost Estimate

The model followed is the Constructive Cost Model (COCOMO) for estimating the efforts required in the completion of the porject. Like all estimation models, the COCOMO model requires sizing information. This information can be specified in the form of:

- Object Point
- Function Point(FP)
- Lines of Source Code(KLOC)

For our project, sizing information in the form of Lines of Source Code is used. The total lines of code,
KLOC = 750
Equations: The initial effort(Ei) in man-months is calculated using equations:

$$E = ax(KLOC)^b$$

where, a = 3.0, b = 1.12, for a semi-detached project E = Efforts in person-hours
E = 4.5 PM

$$D = ax(E)^b$$

Where, a = 2.5,
b = 0.35, for a semi-detached project
D = Duration of Project in months

D = 4 Months

3.1.1.2 Time Estimates

$$C = D * Cp * hrs$$

Where, C = Cost of project

D = Duration in Hours

Cp = Cost incurred per person-hour

hrs = hours

Total of 4.5 person-months are required to complete the project successfully.

Duration of Project D = 6 Months

The approximate duration of the project is 4 months

3.1.2 Project Resources

Project resources [People, Hardware, Software, Tools and other resources] based on Memory Sharing, IPC, and Concurrency derived using appendices to be referred.

3.2 RISK MANAGEMENT

This section discusses Project risks and the approach to managing them.

3.2.1 Risk Identification

1. Dataset needs to be processed in order to get clean data
2. Vanishing Gradients
3. Mode Collapse
4. Failure to Converge

3.2.2 Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	Garbage images in dataset	Medium	Low	High	High
2	Vanishing Gradients	Low	Low	High	High

Table 3.2: Risk Table

Probability	Value	Description
High	Probability of occurrence is	$> 75\%$
Medium	Probability of occurrence is	$26 - 75\%$
Low	Probability of occurrence is	$< 25\%$

Table 3.3: Risk Probability definitions [1]

Impact	Value	Description
Very high	$> 10\%$	Schedule impact or Unacceptable quality
High	$5 - 10\%$	Schedule impact or Some parts of the project have low quality
Medium	$< 5\%$	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 3.5: Risk Impact definitions [1]

3.2.3 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk

Risk ID	1
Risk Description	There are Garbage images in dataset which can lead to improper training of discriminator. It can lead to undesired outputs
Category	Pre processing.
Source	Software requirement Specification document.
Probability	Low
Impact	High
Response	Mitigate
Strategy	To manually remove garbage images from the dataset
Risk Status	Occurred

Risk ID	2
Risk Description	If discriminator is too good , then generator training can fail due to Vanishing gradients
Category	Requirements
Source	Software Design Specification documentation review.
Probability	Low
Impact	High
Response	Mitigate
Strategy	Wasserstein loos and Modifed minimax loss are desigend to prevent vanishing gradients.
Risk Status	Identified

3.3 PROJECT SCHEDULE

3.3.1 Project task set

Major Tasks in the Project stages are:

- Task 1: Colledting Dataset
- Task 2: Cleaning Dataset
- Task 3: Creating GAN model of image colorization and super-resolution

- Task 4: Training the GAN model
- Task 5: Fine tuning the GAN model

3.4 TEAM ORGANIZATION

- Team of 4 members
- 1 Project guide
- 1 Project coordinator

3.4.1 Team structure

Team of 4 members

3.4.2 Management reporting and communication

- Weekly meeting with guide about the work
- Weekly meeting among team for updates and work distribution
- Daily updates on work progress

CHAPTER 4

SOFTWARE REQUIREMENT SPECIFICATION

4.1 INTRODUCTION

4.1.1 Purpose and Scope of Document

The purpose of this project document is to create a comprehensive documentation about the methodology used in implementing the upscaling and colorization of the HSA images. The document explores the goals, objectives, resources, project plan, SRS and finally the summary and conclusion. It is hoped that this document would be beneficial in answering every query that any individual may have about this research project.

4.1.2 User profiles

Two actors have been defined in the Use case Diagram:
User: User preprocesses the image with the help of the GAN and facilitates the whole process.
GAN: It takes the input images, colorizes the image and feeds it to another GAN. The second GAN then upscales the image and a final output image is generated.

4.1.3 Use-cases

¡¡¡¡¡¡¡ HEAD =======

¿¿¿¿¿¿¿ Shreyas

Sr No.	Use Case	Description	Actors	Assumptions
1	Input image	Image is loaded	User	Image of size 64x64 and format jpg/jpeg is uploaded
2	Colourising	GAN colourises the image	GAN	The image is colourised from its black and white format
3	Upscaling	GAN upscales the image	GAN	The image is upscaled to resolution 768x768 and is not pixelated.

Table 4.2: Use Cases

4.1.4 Use Case View

Use Case Diagram. Example is given below

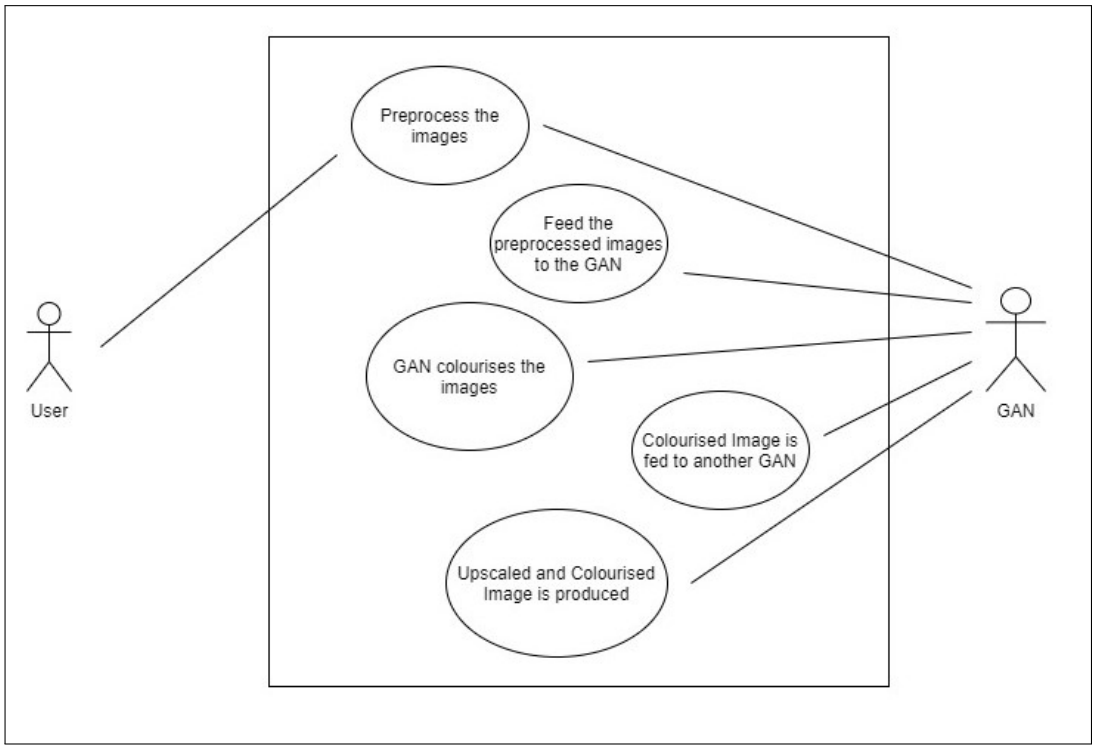

Figure 4.1: Use case diagram

4.1.5 Data Flow Diagram

4.1.5.1 Level 0 Data Flow Diagram

4.1.5.2 Level 1 Data Flow Diagram

4.1.6 Activity Diagram:

4.1.7 Non Functional Requirements:

- Accessibility: The dataset generated would be accessible for everyone to access in open source.
- Availability: The code would be available on github
- Performance: The performance of all the GANs is recorded.

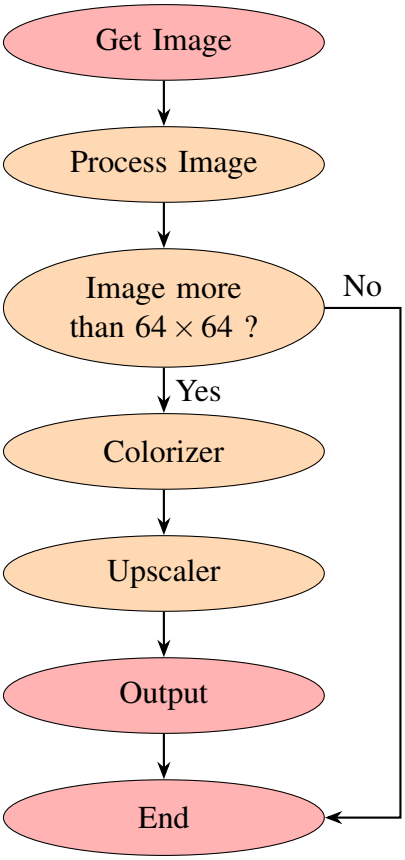

Figure 4.2: Activity Diagram

4.1.8 State Diagram:

State Transition Diagram

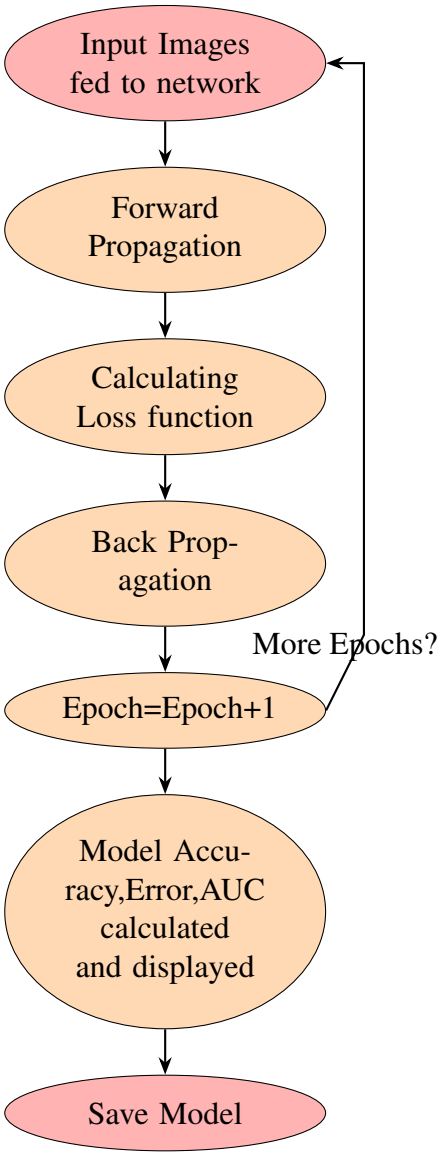

Figure 4.3: State Transition Diagram

CHAPTER 5

DETAILED DESIGN DOCUMENT

5.1 INTRODUCTION

The project is largely inspired by Christian Ledig's SRGAN paper [20] and Dong et. al [22] implementation of SRGANs using Tensorflow. Dahl et. al [10] introduction of residual encoding using VGG architecture and adaptation of GANs as conditional GANs by Mirza et al. [15] proved to be quite effective for implementing colorization of images. We provide detailed architectural design for each respective GANs and other networks that it will be compared with.

5.2 ARCHITECTURAL DESIGN

Generative Adversarial Networks (GANs) have two competing neural network models. The generator takes in the input and generates fake images. The discriminator gets the image from both the generator and the label along with the grayscale image and it determines which pair contains the real colored image. During training, the generator and the discriminator are playing a continuous game. At each iteration, generator can produce more realistic photo, while the discriminator is getting better at distinguishing fake photos. Both models are trained together in a minimax fashion and the goal is to train a generator to be indistinguishable from the real data.

5.2.1 Image Colorization

With conditional GAN, both generator and discriminator are conditioning on the input x. Let the generator be parameterized by θ_g and the discriminator be parameterized by θ_d. The minimax objective function can be defined as:

$$\min_{\theta_g}\max_{\theta_d}\left[\mathbb{E}_{x,y\sim p_{data}}\log D_{\theta_d}(x,y)+E_{x\sim p_{data}}\log(1-D_{\theta_d}(x,G_{\theta_g}(x))\right]$$

Where, G_{θ_g} is the output of the generator and D_{θ_d} is the output of the discriminator. We're currently not introducing any noise in our generator to keep things simple for the time being. Also, we consider $L1$ difference between input x and output y in generator. On each iteration, the discriminator would maximize θ_d according

to the above expression and generator would minimize θ_g in the following way:

$$\min_{\theta_g} \left[-\log(D_{\theta_d}(x, G_{\theta_g}(x))) + \lambda \| G_{\theta_g}(x) - y \|_1 \right]$$

With GAN, if the discriminator considers the pair of images generated by the generator to be a fake photo (not well colored), the loss will be back-propagated through discriminator and through generator. Therefore, generator can learn how to color the image correctly. At the final iteration, the parameters θ_g will be used in our generator to color grayscale images.

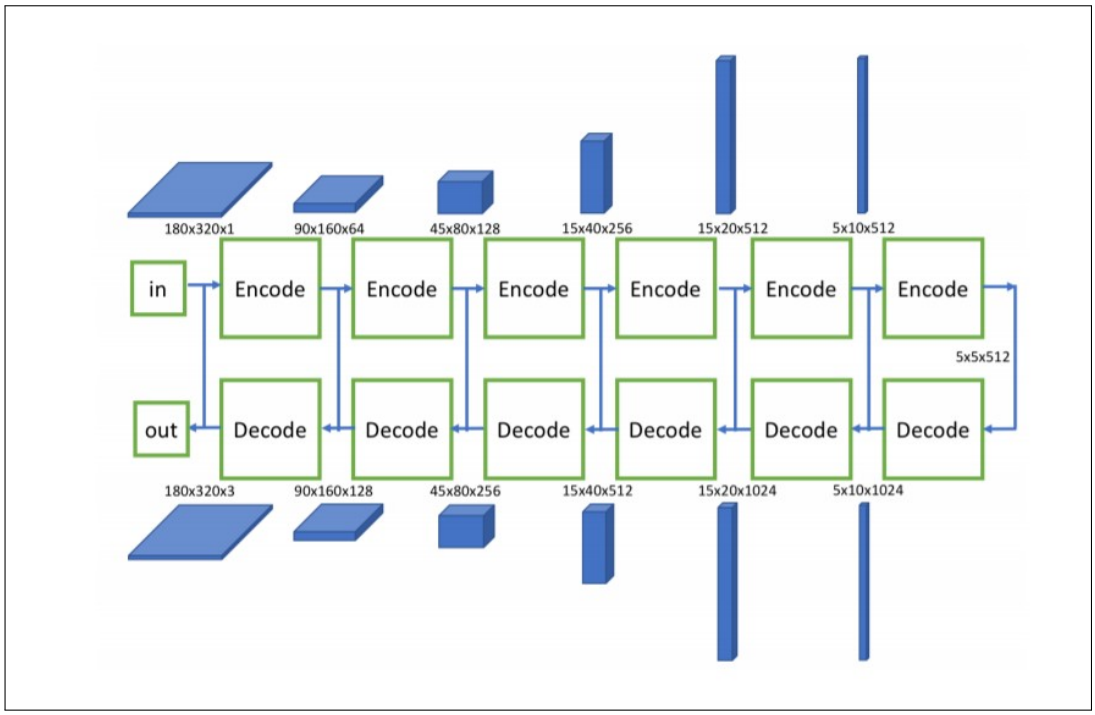

Figure 5.1: Encoder-Decoder ConvNets in Generator

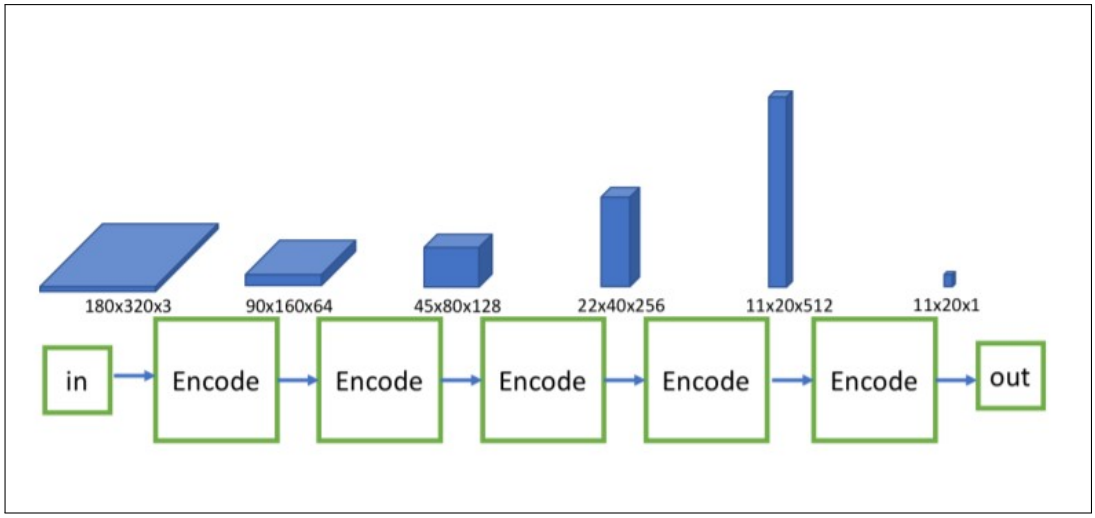

Figure 5.2: Discriminator

5.2.2 Image Super-resolution

We use the SRResNet as the generator in the SRGAN model as used by Ledig et. al [20]. It contains both the residual blocks and the skip connections, as seen in Figure 5.3. Within each residual block, there are two convolution layers followed by a Batch Normalization layer and a parametric ReLU layer. Finally, the image is then upscaled 4 times using two sub-pixel convolution layers [24].

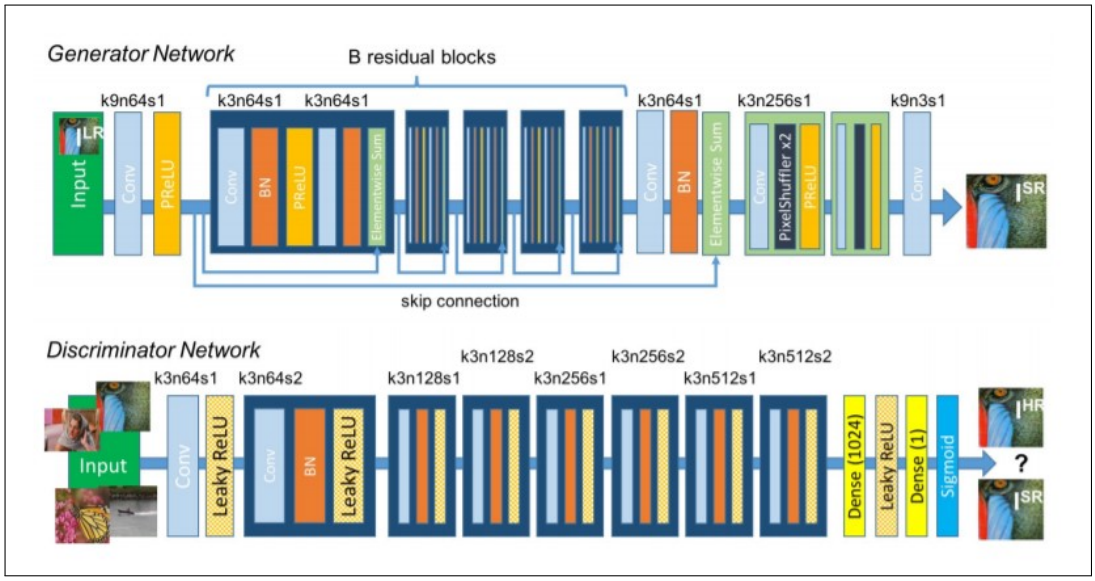

Figure 5.3: SRGAN model: SRResNet generator and discriminator

The generator's goal is to produce high resolution images to fool the discriminator of the GAN into thinking that it is receiving real instead of fake images. On the

other hand the discriminator's goal is to classify the images it has received as either real images or generated images from the generator. The GANs objective function is a minimax game as mentioned in the previous section. We define the minimax function for this task with trivial changes in notation and express it as:

$$\min_{\theta_g}\max_{\theta_d}\mathbb{E}_{I^{HR}\sim p_{train}(I^{HR})}\left[\log D_{\theta_d}(I^{HR})\right]+\mathbb{E}_{I^{LR}\sim p_g(I^{LR})}\left[\log(1-D_{\theta_d}(G_{\theta_g}(I^{LR})))\right]$$

where, $I^{HR}p_{train}(I^{HR})$ are the high resolution images. $I^{LR}p_g(I^{LR})$ are the input low resolution images, G_{θ_g} is the output of the generator and D_{θ_d} is the output of the discriminator. We use the perpetual loss function for VGG based content losses introduced by Ledig et. al [20] which is a weighted sum of a content loss l_X^{SR} and an adversarial loss component $(10^{-3}l_{Gen}^{SR})$.

For the content loss, we aim to use the VGG loss introduced by Ledig et. al[20] which is the euclidean distance between the feature representations of a reconstructed image $G_{\theta_g}(I^{LR})$ and the reference image I^{HR}:

$$l_{VGG_{i,j}}^{SR}=\frac{1}{W_{i,j}H_{i,j}}\sum_{x=1}^{W_{i,j}}\sum_{y=1}^{H_{i,j}}\left(\phi_{i,j}(I^{HR})_{x,y}-\phi_{i,j}(G_{\theta_g}(I^{LR}))_{x,y}\right)^2$$

where $W_{i,j}$ and $H_{i,j}$ represent the dimensions of the respective feature maps within VGG19 network. The adversarial generative loss l_{Gen}^{SR} is defined on the probabilities of the discriminator $D_{\theta_d}(G_{\theta_g}(I^{LR}))$ over all the training samples as:

$$l_{Gen}^{SR}=\sum_{n=1}^{N}-\log D_{\theta_d}(G_{\theta_g}(I^{LR}))$$

$D_{\theta_d}(G_{\theta_g}(I^{LR}))$ is the probability that the reconstructed image $G_{\theta_g}(I^{LR}))$ is a natural HR image. For beter gradient behavior, we minimize $-\log D_{\theta_d}(G_{\theta_g}(I^{LR}))$ instead of $\log\left[1-D_{\theta_d}(G_{\theta_g}(I^{LR}))\right]$.

CHAPTER 6

DATASET AND EXPERIMENTAL SETUP

Initially, we started by scraping the data off Hubble Legacy Archive. Using puppeteer(headless chrome), we scraped off hundreds of thousands of colorized images it has available. The Hubble Legacy archive is slow and produces grainy images with lots of noise and unprocessed images. A filter for M101 (Messier 101) galaxy rendered more than 80 thousand images with a 1 degree difference between consecutive right ascension. The data is large and has no particularly efficient way to clean without human investment. Cleaning tens of thousands of images by handpicking noiseless and well colored images is time consuming. For training the SRGAN, we need high resolution, well colored images.

To that end, we scraped the Hubble Heritage project instead. The Hubble Heritage project releases the highest-quality astronomical images. They are all stitched together, colorized and processed to eliminate noise. Hubble Heritage then selects the best, most striking of these for public release. However, there are only ~ 150 of these images that are actually useful. To increase the amount of data we had, we scraped images from the main Hubble website as well. This provided an extra approximately ~ 1000 images. Each image is a JPG image with dimensions of 256×256 pixels and contains 3 channels of RGB.

CHAPTER 7

SUMMARY AND CONCLUSION

7.1 SUMMARY

Low Resolution and Black and White Images in the Hubble Legacy Archive lie dormant and are needed to be upscaled and colourized for better visual discernment of the astromers. Images are initially scraped from the Hubble Legacy Arhive and the Hubble Legacy Project and a dataset is formed which is cleaned manually and is then fed to two Generative Adversarial Networks consecutively. The images of size $(64 \times 64 \times 1)$ are initially colorized using a first GAN and an output image of size $(64 \times 64 \times 3)$ with three RGB channels is obtained. This is then upscaled to a size of $((64 \cdot n) \times (64 \cdot n) \times 3)$. The output images are visually superior as well as colorized. These images are expected to be of greater use for astronomers than the older ones.

7.2 CONCLUSION

The images are upscaled and colourized using a completely automated algorithm which uses Deep Learning. Generative Adversarial Networks are successfully used for the implementation and the images obtained are in public forum to be used for research, it is anticipated that this will aid the astronomers vastly in their efforts.

REFERENCES

[1] R. S. Pressman, *Software Engineering (3rd Ed.): A Practitioner's Approach*. New York, NY, USA: McGraw-Hill, Inc., 1992.

[2] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *ACM SIGGRAPH 2004 Papers*, pp. 689–694, 2004.

[3] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu, "An adaptive edge detection based colorization algorithm and its applications," pp. 351–354, 01 2005.

[4] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1120–1129, 2006.

[5] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 1214–1220, 2006.

[6] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, pp. 277–280, 07 2002.

[7] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," 2016.

[8] E. Tola, V. Lepetit, and P. Fua, "A fast local descriptor for dense matching," *Proc. CVPR*, 06 2008.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2015.

[10] R. Dahl, "Automatic colorization," 2016.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[14] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2016.

[15] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” 2014.

[16] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, “Generative visual manipulation on the natural image manifold,” 2018.

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” 2018.

[18] R. TSAI, “Multiframe image restoration and registration,” *Advance Computer Visual and Image Processing*, vol. 1, pp. 317–339, 1984.

[19] Tom and Katsaggelos, “Reconstruction of a high-resolution image by simultaneous registration, restoration, and interpolation of low-resolution images,” in *IEEE International Conference on Image Processing* (Anon, ed.), vol. 2, pp. 539–542, IEEE, jan 1996. Proceedings of the 1995 IEEE International Conference on Image Processing. Part 3 (of 3) ; Conference date: 23-10-1995 Through 26-10-1995.

[20] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” 2017.

[21] Jianchao Yang, J. Wright, T. Huang, and Yi Ma, “Image super-resolution as sparse representation of raw image patches,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[22] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 184–199, Springer International Publishing, 2014.

[23] J. Kim, J. K. Lee, and K. M. Lee, “Accurate image super-resolution using very deep convolutional networks,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1646–1654, 2016.

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, 2016.

ANNEXURE A

MATHEMATICAL MODEL

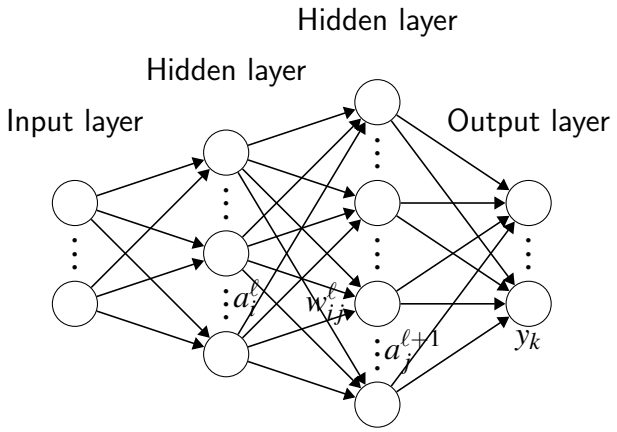

Figure A.1: A feed forward neural network

A.1 ARTIFICIAL NEURAL NETWORKS

Figure A.1 shows a schematic of the simplest multi-layer perceptron network, i.e. a feed-forward neural network. The network is composed of an input layer, hidden layers and output layer. Define a_i^ℓ as the i^{th} neuron of the ℓ^{th} layer and $a_j^{\ell+1}$ as the j^{th} neuron of the $(\ell+1)^{th}$ layer and w_{ij}^ℓ, b_j^ℓ is the weight and bias connecting the two neurons, then the output of the ℓ^{th} layer is given by:

$$a_j^{\ell+1} = g(\sum_{i \in N^\ell} (w_{ij}^\ell a_i^\ell + b_j^\ell)) \tag{A.1}$$

The loss is calculated using a loss function such as cross entropy function especially for binary classification.

$$loss(y', y) = -y \log y' - (1-y) \log (1-y'). \tag{A.2}$$

where $y' \in \{0,1\}, y \in (0,1)$. This objective is to minimize this cross entropy over a batch of all training data. This is done using gradient descent where the parameters viz. weights and biases are updated to reduce the overall cross entropy loss.

A.2 CONVOLUTIONAL NEURAL NETWORKS

A convolution is a linear operation that can be viewed as a multiplication or dot product of matrices. The input is a tensor of shape *height* x *width* x *channels* and the convolution operation abstracts the image to a feature map (also called a kernel)

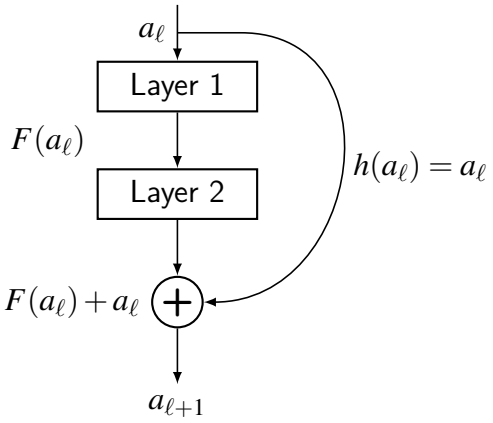

Figure A.2: Residual Block

of shape *kernelsize* x *kernelsize* x *kernelchannels*. The layers can be computed by:

$$a_j^{\ell+1} = g(\sum_{i \in F_j} (a_i^\ell \times k_{ij}^{\ell+1} + b_j^{\ell+1})) \tag{A.3}$$

where ℓ is the layer, g is the activation function ReLU, F_j is the receptive field, k is the convolutional kernel and b is the bias.

A.3 RESIDUAL NETWORKS

Figure A.2 shows a building block of a residual network. The residual blocks can be expressed mathematically as follows. Let $h(a)$ be an underlying mapping that is to be fit by a set of layers, where a is the input to these layers. If we hypothesize that multiple non-linear transformations by these layers can approximate the layer functions, then one can also hypothesize that a similar approximation can be made for the residual functions, i.e. $h(a) - a$, which have the same input and output dimensions. So instead of letting the underlying mapping be $h(a)$ we approximate a residual function $F(a) = h(a) - a$. Thus, the actual function becomes $h(a) = F(a) + a$. We can achieve this approximation in a feed forward neural network using a series of skip connections that perform identity mapping and jump over a few layers. Adding the outputs of these two connections gives the final output layer. Thus, the residual unit can be defined as,

$$a_{\ell+1} = h(a_\ell) + F(a_\ell, W_\ell) \tag{A.4}$$

where $a_{\ell+1}$ and a_ℓ represent the input and output for the ℓ^{th} layer and F is the residual function. The h denotes the operation in the identity mapping. The output of the layer is generally processed using an activation function such as ReLU. Let f be the ReLU activation and replacing F with the definition of feed forward activation, the residual block thus can be defined as,

$$a_{\ell+1} = f(h(a_\ell) + W_2 f(W_1 a_\ell)) \tag{A.5}$$

For the sake of simplicity, we consider no operation being performed in the identity mapping. Thus, equation A.4 and equation A.5 become,

$$a_{\ell+1} = f(a_\ell + F(a_\ell, W_\ell)) \tag{A.6}$$

A.4 GENERATIVE ADVERSARIAL NETWORKS

A generative network, G, is supposed to learn the underlying distribution of a latent space, Y. Instead of visually assessing the quality of network outputs and judge how we can adapt the network to produce convincing results, we incorporate automatic tweaking during training by introducing a discriminative network D. The network D takes in both the fabricated outputs generated by G and real inputs from the underlying distribution Y. The network produces a probability of the image belonging to the real or fabricated space.

Let $x \in X$ be a low resolution/grayscale image and $y \in Y$ be it's underlying distribution from the latent space Y. Generator G takes in input x and produces an output $\hat{y}$. We define the mapping $x \to \hat{y}$ in the following manner:

$$G(x) = \hat{y}$$

The discriminative network D is fed the fabricated mapping $x \to \hat{y}$ and the underlying distribution of x i.e. $y \in Y$. The network D then produces a result that is a probability distribution of the input space indicating the class of the image that it

thinks the input belongs to. We define this as:

$$D\big(G(x), y\big) = p$$

where $p \in (0,1)$ is the probability that the image is fabricated or real. With conditional GAN, both generator and discriminator are conditioning on the input x. Let the generator be parameterized by θ_g and the discriminator be parameterized by θ_d. The minimax objective function can be defined as:

$$\min_{\theta_g} \max_{\theta_d} \Big[\mathbb{E}_{x,y\sim p_{data}} \log D_{\theta_d}(x,y) + E_{x\sim p_{data}} \log(1 - D_{\theta_d}(x, G_{\theta_g}(x)) \Big]$$

Where, G_{θ_g} is the output of the generator and D_{θ_d} is the output of the discriminator. We're currently not introducing any noise in our generator to keep things simple for the time being. Also, we consider $L1$ difference between input x and output y in generator. On each iteration, the discriminator would maximize θ_d according to the above expression and generator would minimize θ_g in the following way:

$$\min_{\theta_g} \Big[-\log(D_{\theta_d}(x, G_{\theta_g}(x))) + \lambda \| G_{\theta_g}(x) - y \|_1 \Big]$$

ANNEXURE B

PLAGIARISM REPORT

ANNEXURE C

PAPER PUBLISHED (IF ANY)

ANNEXURE D

SPONSORSHIP DETAIL (IF ANY)