UNIVERSITI TEKNIKAL MALAYSIA, MELAKA

FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

WORKSHOP 2

COLOURIZING BLACK AND WHITE PHOTOS SYSTEM
(FINAL REPORT)

NAME	HARIHARAN A/LR.MOHAN (B031910477) TANG LI HO (B031810122) AININ SOFIYA HISHAM (B031810081) AHMAD NAUFAL BIN MOHD SALEH (B031810382)
COURSE	BITI
PROJECT TITLE	COLOURIZING BLACK AND WHITE PHOTOS SYSTEM
SUPERVISOR	DR. SOODABEH SAFA

TABLE OF CONTENT

ACKNOWLEDGEMENT

First of all, we would like to thank our supervisor, DR. Soodabeh for guiding us with the project, providing ideas and suggestions to us, and helping us to complete our task..

Not to forget, we would also like to recognize our university, Universiti Teknikal Malaysia Melaka (UTeM) for providing us a chance to test ourselves abilities and skills, and providing us such a good opportunity to show ourselves.

ABSTRACT

Colorization of grayscale images has become a more researched area in recent years, thanks to the advent of deep convolutional neural networks. We attempt to apply this concept to colorization of black and white images without any problem. In this project we have described a method of colorizing a grayscale image in detail. The object and texture content of the image can be as similar as it can be to the target image. Although we can never be perfectly sure that matching of objects from target to image is absolutely correct. Hence to improve this we partially colorize the given image at those points where we are absolutely sure. To implement it, we propose convolutional neural network architectures trained. With AI technology, we can implement it in a systematic way without any problem occurring.

LIST OF FIGURES AND TABLES

1.0 CHAPTER 1 : INTRODUCTION

1.1 Introduction

The technology nowadays aims in imparting tremendous knowledge oriented innovations these days. Colorization of grayscale images has become a more researched area in the recent years. With the implementation of AI technology, the system automatically detects the black and white photo and it colourizes the image. Generally, the idea of colouring a grayscale image is a task that is simple for the human mind, we learn from an early age to fill in missing colours in colouring books, by remembering that grass is green, the sky is blue with white clouds or that an apple can be red or green.

This project provides a system that digitizes a black and white photo for better quality. Most of the old photos sometimes lose its color and it is hard to get the feeling where the photo was taken back then. It is also resentful to have so many beautiful photos before but lost because of no color compared with nowadays. In this modern technology, information is very useful. There is limited information that can be read from the gray photo. Moreover, gray photos can be hard to communicate on an emotional level.

The objective of this project is to digitalize the gray photo to colorize the photo so that it can rebuild the scenery or historical photo as much as possible. Next, the project is also to help people understand the importance of color and how they can use it to better convey the stories they want to tell and the feelings they had at the time of taking the photo to those who view the image. Some people might have trouble with blurry pictures or colorless pictures.

The scope of this project is open for any user who wishes their photo to be colorized and beautiful. To conclude, the system will capture the image or user input the colorless photo and the system will turn it into a colorized photo.

1.2 Problem Statement

- Gray photo lost reality feeling and not attractive to people nowadays
- Resentful to have so many beautiful photo before but lost because of no colour compare with photo nowadays
- Limited information can be read from gray photo
- Gray photos can be hard to communicate on an emotional level.

1.3 Objectives

- To perform digital colorization of black and white photos.
- To rebuild historical scenes or photos into real color photos as much as possible.
- To ease viewers extracting accurate information from the photographs.
- To help people tell the stories captured via photographs and express the feelings they had during photo taking to the viewers.

1.4 Project Scope

1.4.1 Module to be developed

- User Management Module – User preferences, Interface: user-friendly, easier to use for input controls such as using buttons and image fields, and for showing the result by displaying color image matching process.
- Image Management Module – Image gallery: Choose image from image gallery. Contain categories of images
- AI System Construction Module – System: receive image from user and system start to colorize the photo.
- Database Management Module – All the images will be kept in the database as for future improvement.

1.4.2 Target User

- This system and application can be widely used by any user.
- Photographer/Investigator: insert their image, view and send image to the system for colorization process.
- Administration: view all the image, sorting and make image categorization

1.5 Software and Hardware Requirement

SOFTWARE	**HARDWARE**
System and Server: ● SQLite ● python 3.8 **Web:** ● XAMPP ● PHP 5.5 or above **Apps:** ● Android Studio	Laptop

1.6 Project Significance

1.6.1 Motivation and inspiration for the project.

The findings of this study will redound to the benefit of managing the target users to colourize the image as well and also it is an easier task for the target to save as a reference without any problems. For the researcher, this project will help them to use or upgrade for the future finding process without any problems.

The main advantages of the system are:

- When it is compared to black and white images, this system saves the time and also helps target users to analyze it without any problems.
- Colorization would lead to increase in the creation pipeline throughput, saving artists the time spent on colorization while requiring limited amounts of intervention.

1.7 Summary

In summary, it is not easy to discover the exact color of the B&W photos, but it is possible to colorize the photo based on experience and imagination. It can be done by investigating the possible colors of the objects in the photos, for example, the color of clothes, buildings, trees, cars, etc., and colorizing them manually using some software such as Photoshop.

The other approach is to train the computer to understand the colors of different objects by providing a large amount of color photos, then ask the computer to colorize by identifying plausible color for the detected objection.Thanks to the development of neural network, it can be done by using (UNet) which able to localise and distinguish borders is by doing classification on every pixel, so the input and output share the same size.

2.0 CHAPTER 2 : METHODOLOGY

2.1 Intelligent System Development Methodology

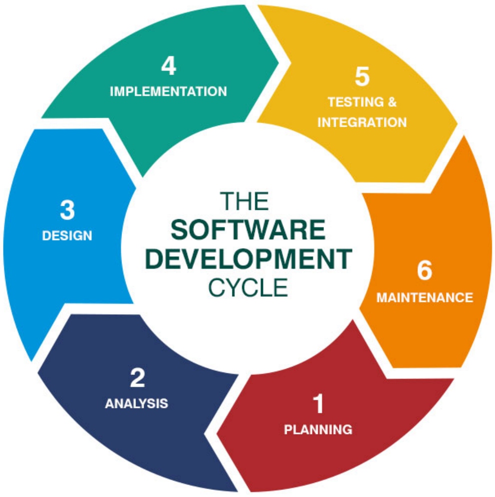


Diagram 1 : The SDLC flow

We are using software development life cycle (SDLC) as our methodology to process our project to ensure that our progress is in control.

2.2 Intelligent System Development Technique

______First and foremost, we may need to reiterate on our team's objective. Image colorization is an image-to-image translation that will receive a gray colour image as input and provide color information to each pixel in the grayscale input image. The input can have many formats like RGB, LAB or even a single channel picture. As the Neural Network is the main trend in the region of artificial intelligence and is thought that it's nearby possible to do anything, it has the ability to learn itself but no need for human resources working on it. Especially in the tasks which relate image processing, NN shows a great performance on it, whether on analysing or generating a new image.

In addition, for normal people who are not a programmer the system will be hard to use because they do not have the basic knowledge on it, so we have to develop a user-friendly interface for them to solve this problem. The system will now have been splitted into backend-system and front-end system. In the final product, supposingly we will have a backend system for the colorization process and a medium server on charge to link the system to the interface.

______As our project is a type of image processing under the region of rebuilding, we have decided to use some convolutional neural networks to help us solve the colorization process. The reason for using CNN as our Neural Network is because the extra convolution layers inside the CNN have great efficiency in extracting an image or rebuilding the image.

As the training on the Neural Network greatly depends on the parameter setting, we will have a basic setting on it and further updating will be applied when we observe some weakness or insufficient, for example not enough accuracy or instability in training result.

2.3 Summary

______In summary, we will initial the project by trying on the U-net CNN model as our backend system’s main core. The parameter will be updated during the training phase to make it able to have better performance. Overall, the system will be written in Python Language except the interfaces.

3.0 CHAPTER 3 : ANALYSIS

3.1 Analysis of Current Application

The neural network is playing an important role in this task, below are some of the researches which focus on colorization.

Resource example:

1) “Colorful Image Colorization“ by Richard Zhang, Phillip Isola, Alexei A. Efros at 2016

   The system is implemented as a feed-forward pass in a CNN at test time and is trained on over a million color images. The result of this project successfully fools humans on 32 % of the trials. This project is using a pre-trained VGG16 network for the feature tracking and reweighting the loss of each pixel at train time based on the pixel color rarity.

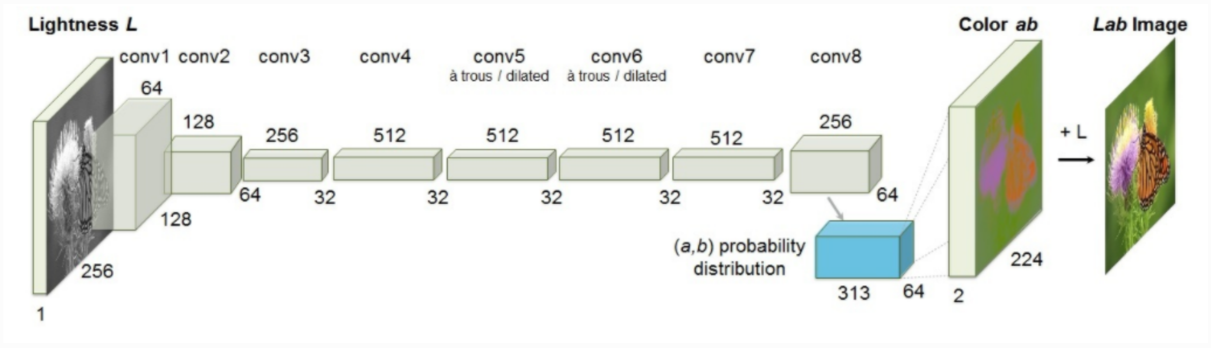


Diagram 2: The CNN network

https://link.springer.com/chapter/10.1007/978-3-319-46487-9_40

2) “Image Colorization Using Generative Adversarial Networks” by Kamyar Nazeri, Eric Ng, Mehran Ebrahimi at 2018

   In this approach, it attempts to fully generalize the colorization procedure using a conditional Deep Convolutional Generative Adversarial Network (DCGAN) for colorizing process. By using a generator with CNN and a simple discriminator to judge whether the outputs from the generator are real or fake. Both models are trained repeatingly in every epoch to improve their accuracy on their job separately.

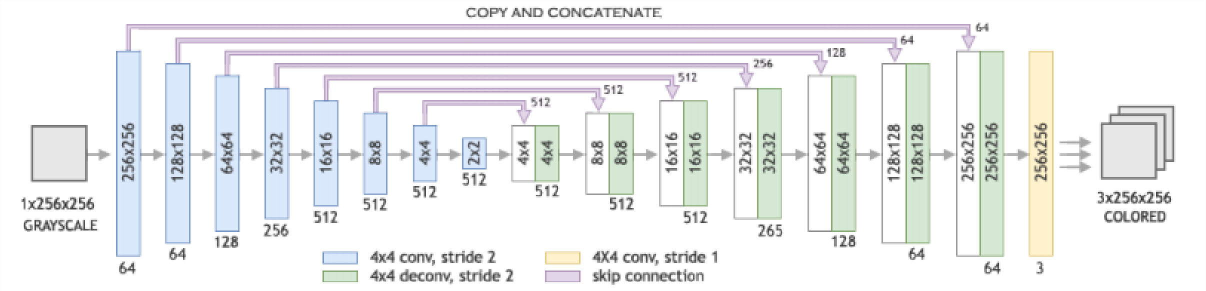


Diagram 3: The generator network

https://arxiv.org/pdf/1803.05400.pdf

3) "Image Colorization with Deep Convolutional Neural Networks" by Jeff Hwang and You Zhou in 2016

Using a convolutional neural network (CNN) that accepts a black-and-white image as an input and generates a colorized version of the image as its output. This project was inspired in part by Ryan Dahl's CNNbased system for automatically colorizing images. The input will be in 3 channels RGB format. It initialized parts of the model with a VGG16 instance that has been pretrained on the ImageNet dataset for further feature extraction.

http://cs231n.stanford.edu/reports/2016/pdfs/219_Report.pdf

In summary, almost all of the existing projects use convolution neural networks as the base model, some of them involve pre-trained networks like VGG16 and VGG19.

3.2 Analysis of Proposed Intelligent System

The proposal system will include a training module and evaluating module. The training module is used to train the neural network's weights, it will cause a lot of time for building up a pre-trained model. The evaluating module is the module to colorize a gray picture by using the pre-trained model, as it does not involve any training process, it can be done in a few seconds for each input picture.

The proposed neural network is convolution network(CNN), which is a neural network with multiple convolution layers in order to pick the features from an image. In processing on image tasks, CNN shows a far improvement in accuracy compared with the common neural network. Hence it will be the main choice in our project. As we need to have an image as output, we have to add in a few max-pooling layers afterward.

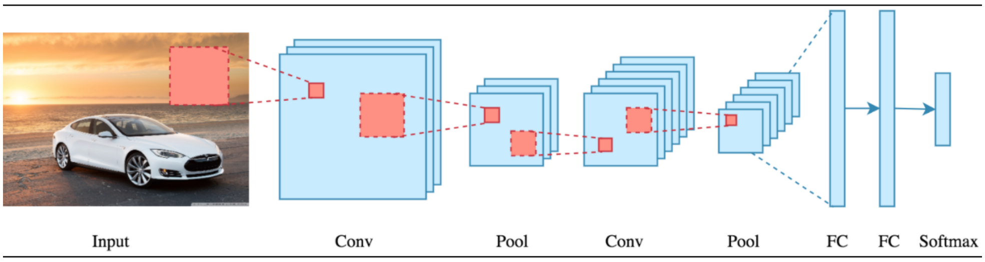


Diagram 4: The CNN structure

Based on our planning, the model will allow one picture as input in the size of 256*256 with 3 channels in format of RGB, it can save our time to prevent preprocessing too much on the images, the final output will be an image with the same size as the input image. The input images for training will be taken from the coco dataset then converted into gray colour, the original images will be the ground truth for loss computation. The loss will be calculated by comparing the generated images and the original images in every pixel.

3.3 Structure Chart of Proposed Intelligent System

The initial design of our system will be like:

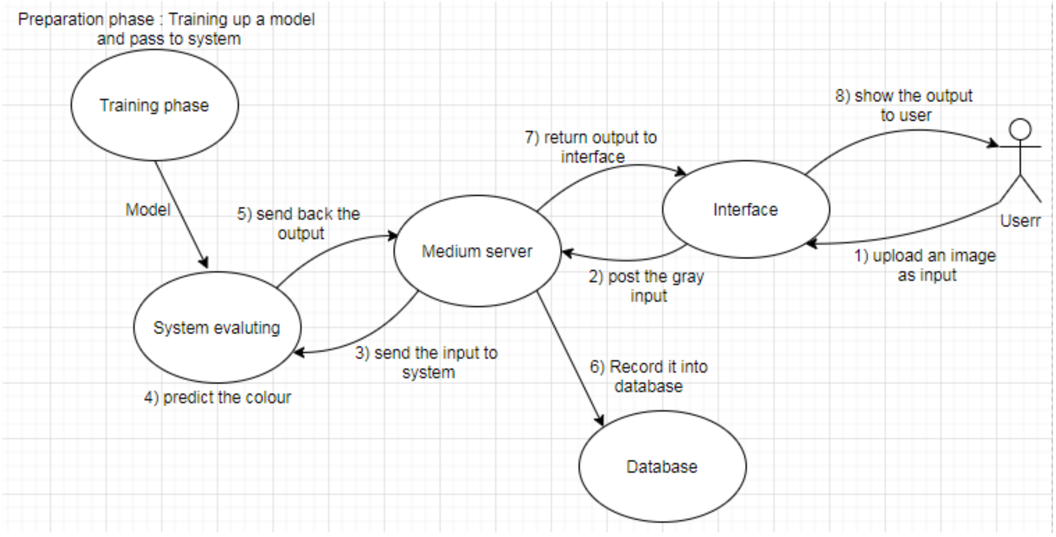
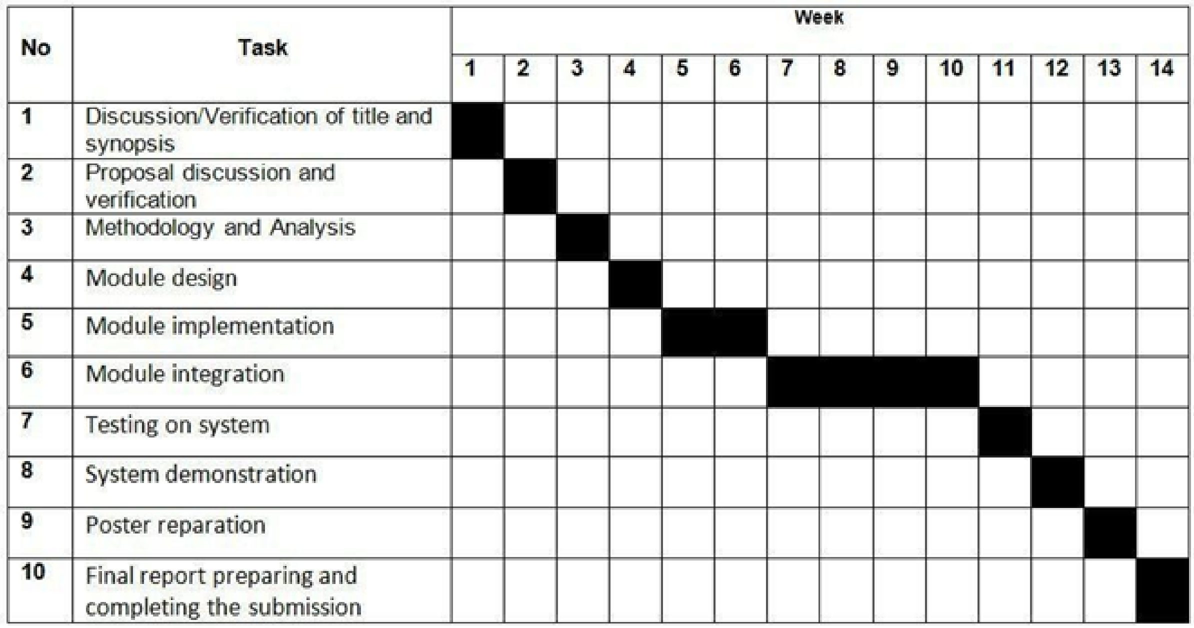

Preparation phase : Training up a model and pass to system
Training phase
Model
System evaluting
4) predict the colour
5) send back the output
3) send the input to system
Medium server
7) return output to interface
2) post the gray input
6) Record it into database
Database
Interface
8) show the output to user
1) upload an image as input
Userr

Diagram 5 : The conceptual design of workflow

3.4 Work Breakdown

Gantt Chart:



No	Task	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Discussion/Verification of title and synopsis	■													
2	Proposal discussion and verification		■												
3	Methodology and Analysis			■											
4	Module design				■										
5	Module implementation					■	■								
6	Module integration							■	■	■	■				
7	Testing on system											■			
8	System demonstration												■		
9	Poster reparation													■	
10	Final report preparing and completing the submission														■

Table 1: Gantt Chart

Work Allocation:

Work	Name
Module 1: Artificial Intelligence System	TANG LI HO
Module 2: Server & Database	AHMAD NAUFAL BIN MOHDSALEH
Module 3: Interface	HARIHARAN A/LR.MOHAN AININ SOFIYAHISHAM

Table 2 : Work Allocation

Work Breakdown:

HARIHARAN A/LR.MOHAN	- H
TANG LI HO	- T
AININ SOFIYA HISHAM	- A
AHMAD NAUFAL BIN MOHD SALEH	- N

No	Task	Name
Planning	Discuss on the workshop title, work distribution and what needs to be done.	All
	Proposal is submitted via ulearn.	All .
Analysis	Identify and discuss the requirements of the system.	All
	Define the project goals.	All
	Identify the risks and limitations of the system.	All
	Discuss and choose which AI techniques are going to be implemented into the system.	T
Design	1. Structured chart	T, H , A

	2. System Methodology 3. Use case diagram	
	1. Flowchart	T
	1. Sequence diagram 2. ERD	A, N
	Design and program Database	T, N
Implementation	Developing the back-end part of the system.	T, N
	Developing the front-end part of the system.	H, A
	Implement AI techniques into the system.	T
Testing and integration	Testing the system by running it.	All
Maintenance	Identify and fix errors.	T
	Improve the system performance.	T
Evaluation	Presentation of the system to the committees.	All

Table 3 : Work Breakdown

3.5 Summary

In summary for this chapter, we are making the preparation phase for the whole project and making the direction of work. We have analyzed how others do the system and develop our own. It is the milestone for our project starting and showing the concept. Also it has shown the initial planning of our team include the time distribution and work for every member.

4.0 CHAPTER 4 : DESIGN

4.1 Introduction

This chapter will talk about the details of our system architecture and modules, also in the purpose of showing what we are planning to apply.

4.2 Intelligent System Architecture

The proposed neural network is U-net, which is a type of convolution neural network specified for image-related tasks.

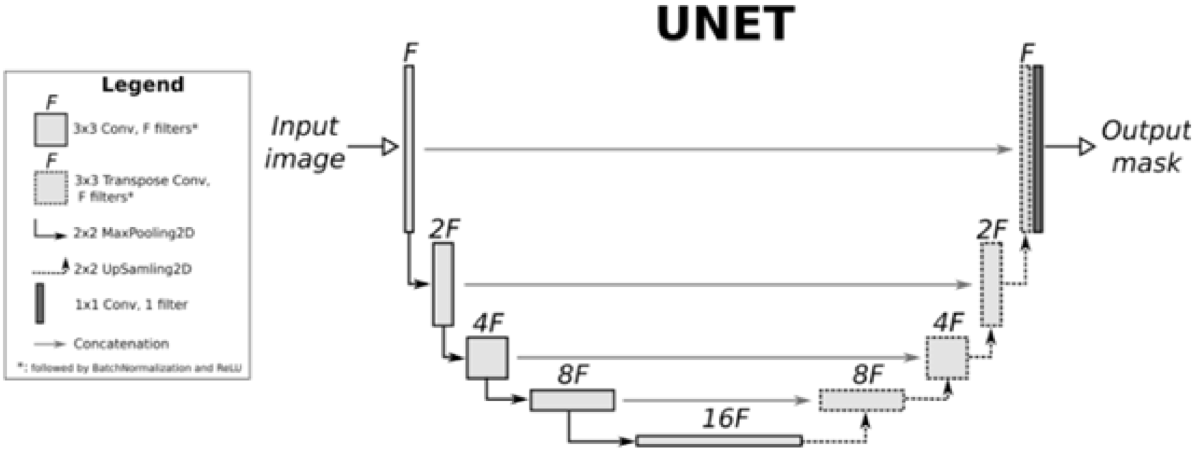


Diagram 6: The structure of U-net

Both the output and input for the model will be 256*256 pixes * 3 channels. The model will have 4 layers of max pooling layers and 4 upsampling layers. This model has a benefit that is the feature of every layer from different views inside the image can be stored and re-used in further evaluation until the last, normally the earlier layers will extract out more surface features and deeper layers extract out a more detailed feature which may not be understood by humans.

For every epoch, the loss will be calculated by using the difference in color in every pixel between original picture and generated picture, which will be used to update the weights on nodes by using back-propagation.

4.3 Module 1

The Intelligent System : Analyze others programmes and researches to find out the method to fulfil the objective and design the programme for both training and testing, including finding out the required dataset for training and making an initial base system for others to follow on.

The main thing to do is handling the risk and quality to make sure it can perform well and stable, based on different accident conditions may need to do improvement on the programme or handling the unexpected issue.

4.4 Module 2

The Server Medium : As our main code is written in Python, it is suggested to use python in the server too to ease the process of linking them together. As we have mentioned before, the server side will be using FastAPI which is good in treating multiple requests and also a hot library today. Other than that, for recording the history of colorizing a database will be used to store all these information including the input output images' path.

The database that we use is SQLite, which is a serverless storage and useful in localhost. SQLite can be accessed by Python code easily and also Xampp Apache has related library inside to connect to it, which can save us a lot of time in connecting the things

4.5 Module 3

The Interface : We have 2 interfaces to be used in this project, Apps and web page. The Apps will be written using Android Studio and website in HTML, PHP and Javascript. The Apps will have the ability to upload an image to the server and receive the output from it. The web page will include 2 pages, one page for uploading and receiving results then display in the same page, another page for showing the previous colorized images.

4.6 Summary

The basic structure for the colorizing system will be using U-net technique. These 3 modules are a completed API when merging together, the server will link the backend system and the frontend interface together so that we can access the system without running through command prompt or other medium software.

5.0 CHAPTER 5 : IMPLEMENTATION

5.1 Introduction

______This chapter will be about how we are going to implement our system including the 3 modules. It will cover the runtime environment for the system and detail of the environment.

5.2 Intelligent System Development Environment

The system is running on Python 3.8 environment with the following library packages:

```
tensorflow==2.3.1
keras==2.4.3
numpy==1.18.5
opencv-python
uvicorn
fastapi
uuid
pydantic
sqlalchemy
```

And also we need to install xampp as our localhost server so that the result can be returned correctly by following the virtual IP address. For connecting the SQLite database, the php version in apache is at least 5.5 because the required library for connecting SQLite does not exist in the earlier version. The newest version of xampp can be installed from “https://sourceforge.net/projects/xampp/”.

SQLite is required too for setting up the database, which can be downloaded from "https://sqlitebrowser.org/dl/".

5.3 Intelligent System Configuration Management

Identification :

The system is currently being tested on linux Ubuntu, window 7 and window 10 with pip installed. For running the AI system, it’s suggested to have at least 8GB RAM in order to make it process well without corruption. The training phase may need a GPU device because the training on images will need a quite huge computing power.

Baseline :
As we have used PHP and SQL inside our programme, it should have Xampp with PHP version higher than 5.5 and SQLite to be accessible to the database. If you want to run the apps on the computer, some extra software like Android Studio and Bluestack may be required for it to run. For the AI part, Python with version 3.7 is required. The required libraries for Python code to run are listed in the requirements text file and new users should install them on themselves.

Version Control :
As we have only one completed product here, no option on choosing another version other than the final version.

Auditing :
As it is a workshop in one semester and there are no github for it open to the public, any updating on the code will be just based on the one who wants to try it.

5.4 Completing Task via Intelligence

We are using the Python library Keras to develop our system. And after a long period of training on the model, it is now able to fit the 118k training image dataset, our system is now able to analyse the contents inside an image and get the characteristics.

The prediction is mostly based on the gray pixels and its overall shape, in normal condition, the input that we try later on in the evaluating and testing phase will not be out of this range, so the model will be able to colorize most of the images well.

5.5 Summary

In short, we are successfully developing an AI system with U-net CNN as the core and performing colourization and finishing training on it. And now we have successfully linked it to the interfaces by the server playing the role of medium.

6.0 CHAPTER 6 : TESTING

6.1 Testing Method

For this chapter, we will validate the result of the implementation of the colorizating system that has hit the requirements specification of the project. Since this project is not a body in development phase, we will need to cover the individual tasks and make sure all of them work well then only we can do testing on the next step. Hence, in our testing phase it will involve 3 parts, they are unit, integration and system testing. These testing helps us to ensure the system is bugless and proof of the tested scenario of the entire system from its beginning towards end.

a) Unit Testing

In this section, we will do testing on every task separately. It is on purpose to make sure that the individual parts can work well, if simply combined all together it will be hard to do checking and testing when bugs are happening or intend to make some change. So in this phase we will do testing on the AI backend system, frontend interface including apps and web page and also the medium server.

AI backend : After finishing training, the loss and accuracy in training will be collected and used for recording, some parameters will be updated to improve the performance of the system. A graph will be plotted out easier for us to observe the result and become part of our report. After that when the model is trained up, the ai system will be simplified into one or few functions so that we can call it from another file easily, then the programme will be executed again to find out the bug and mistake until it works perfectly.

Server medium : Set up a basic server and web page for testing the ability of sending request and return results, there will be no AI function inside but instead using a hard code function inside it, then check on whether the backend and frontend are connected successfully. After connecting, the database will be involved and tested on whether it can insert a new row of information into it.

Interface frontend : Build up the design and implement it then testing to post requests to an inexistent server, then observe whether it can post requests to somewhere. After the interface outlook is completed, do checking on the javascript part to make sure it’s working. Then try to connect the database and get the information to display it.

b) Integration Testing

When integration testing, the different parts will be integrated with another related part.

Backend System and Server : To involve the colorizing function into the server part and executing in hard code, it’s on purpose to observe whether the system can be applied when run through the medium programme.

Server and Frontend interface : Send request from interface to the server to see whether the request reaches the selected target, then return the same image from the server to ensure that the interface can receive output from the server and response for it.

c) System Testing

System testing will be involved in the end of the testing. The testing will be like can the system provide the workflow exactly like the workflow diagram. For example, can the system get user interaction then perform until the end of the system also does not occur any unexpected fault.

6.2 Test Result Analysis

As only the AI system have result, the record in training will be shown at below :

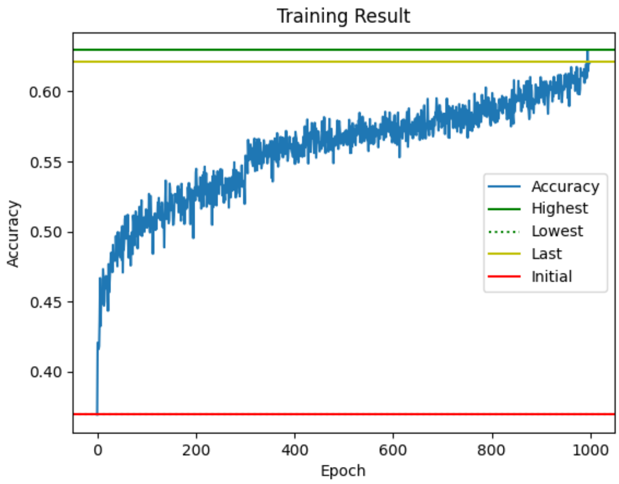


Diagram 7 : Accuracy over period

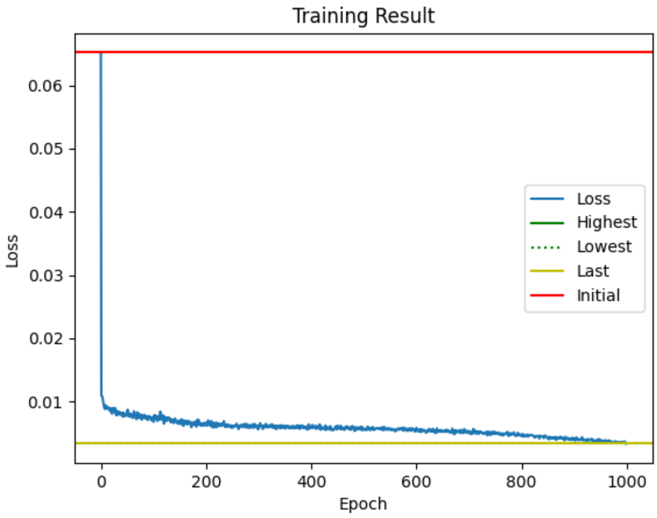
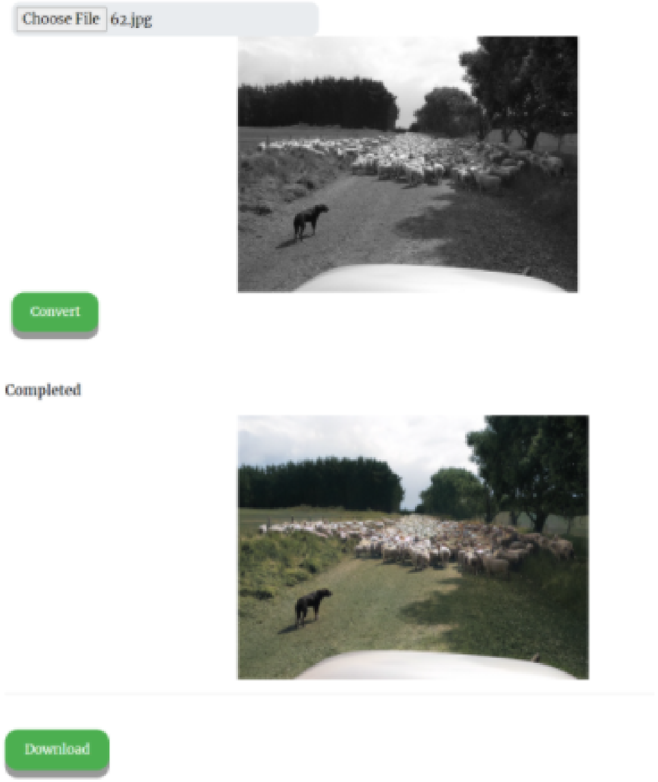


Diagram 8 : Loss over period

	Original Code	Our system (After adjustments)
Accuracy	55.14%	62.08%
Dataset	CIFAR-10	COCO2017
Image Size	32*32	256*256

Table 4 : Comparison on our system and original one

Result for interfaces:



Completed

Diagram 9 : The webpage design(main page)

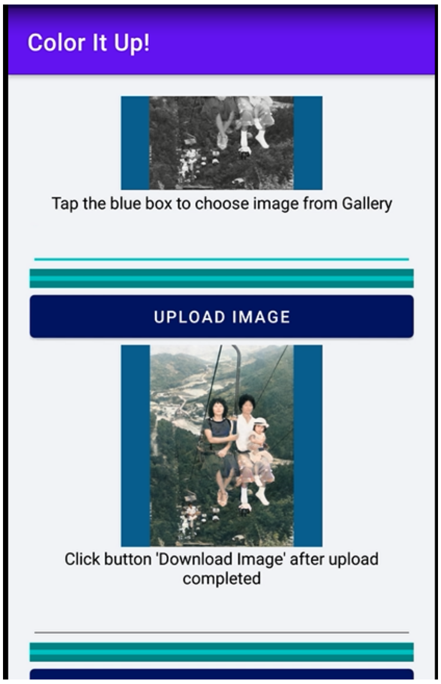

Color It Up!
Tap the blue box to choose image from Gallery
UPLOAD IMAGE
Click button 'Download Image' after upload completed

Diagram 10 : The Apps design

6.3 System Constraints

As this system is only local only, so that others will not be able to access this system through the internet. It's because if making it online there will be a lot of trouble for example the online server cost and the environment, it will be hard to handle if it's not in localhost.

Other than that, actually this system is still unable to handle all images/sceneries yet because of limitations in training time and platform.

6.4 Summary

After completing the testing task, all tasks are workable even after merging together, the testing and checking will be repeated if the code is updated to ensure everything is working well. This also benefits us in controlling the performance of the system.

7.0 CHAPTER 7 : CONCLUSION

7.1 Intelligent System Advantages and Commercial Values

As the past technology is not as good as nowadays especially when the concept of camera and photo has just appeared in the world, our ancients have no ability to take a colourful photo like now but just black and white images. All these photos are without colour and lose their original face, as human's eye is designed to be able to observe multiple colours, we are more likely to get information from colour but these colourless photos just can give us a feeling of unreal. Black and white photos have lost too much content and can't give us more impact or feeling than on spot. Hence, it's our turn to make it take back its original face and gloriousness. By the help of AI, our system can now colorize these black and white photos by just a few seconds of action.

I believe that there will be a lot of people/companies with many memorial old photos but don't know how to treat them, most of them will be willing to pay for rebuilding their photos into colorful photos.

Our system will be attractive to the art collectors too, they will not mind to pay for more collections or try to see what if they put their black and white images into the system. We also can collect the outputs on ourselves and sell it in the form of a collection book or art book.

Lastly, our project has great potential in future works, by using the same concept we can colorize the old video and relight their market values, providing new incomes without any cost. By colorizing the black and white drawing art, we can make it in another form of product, for example make up an anime quickly and easily by colorizing on the drawing outline. More and more projects can be extended from it for example to recover the broken images, enhance the resolution, and also transfer art style onto photos.

7.2 Intelligent System Weaknesses

Because of the training time is limited, the model is just running on images with 256*256 pixels as it is the setting during training phase, it means all input image will be resized to 256*256 pixels before the processing. After the colorizing process it will be resized to original size but there will be some loss in resolution too. And also as the structure of U-net requires returned values from every layer to another matched level layers, it has high demand on the size of input image, causing it hard to fit images of all sizes.

7.3 Suggestions for System Improvement

There will be few improvements on this project, the first and easiest method to improve its performance is to increase the number of training epochs, as the dataset for current training is large it causes the model hard to converge.

Secondly, a pre-trained model for extracting features may be included into the system as part of the network. Actually most of the time of training is not to colorize the image but training for extracting the feature, the usage of pre-trained models from others projects can save a lot of time on it but with high accuracy meanwhile.

7.4 Summary

Overall. we have successfully run the programme and built up user-friendly interfaces for non-programmer users. As we analyze it there is still some improbable space for us in this project, so it might be the next planning to do improvement and enhancement on the future work.

REFERENCES

Original code retrieved from

https://github.com/thevarunsharma/Image-Colorization

“Image Colorization Using Generative Adversarial Networks” by Kamyar Nazeri, Eric Ng, Mehran Ebrahimi at 2018

Retrieved from

https://arxiv.org/pdf/1803.05400.pdf

“Image Colorization with Deep Convolutional Neural Networks” by Jeff Hwang and You Zhou in 2016

Retrieved from

http://cs231n.stanford.edu/reports/2016/pdfs/219_Report.pdf

“Colorful Image Colorization“ by Richard Zhang, Phillip Isola, Alexei A. Efros at 2016

Retrieved from

https://link.springer.com/chapter/10.1007/978-3-319-46487-9_40

APPENDICES

```
import os

img_path_train = []
img_path_test = []

for dirname, _, filenames in os.walk('./coco2017/train2017/'):
    for filename in filenames:
        img_path_train.append(os.path.join(dirname, filename))

img_path_test = []
for dirname, _, filenames in os.walk('./coco2017/val2017/'):
    for filename in filenames:
        img_path_test.append(os.path.join(dirname, filename))

TOTAL_TRAIN = len(img_path_train)
```

```
TOTAL_TEST = len(img_path_test)
TOTAL = TOTAL_TEST + TOTAL_TRAIN

print(TOTAL)

#################################################################
IMPORT LIBRARY
import numpy as np
import random
import os
import sys

import matplotlib.pyplot as plt
import gc
import tensorflow as tf
keras = tf.keras

from keras.layers import Conv2D, MaxPool2D, BatchNormalization,
LeakyReLU, Concatenate, Activation, Input
from keras.layers import Conv2DTranspose as Deconv2D
from keras.models import Model

import cv2

#################################################################
FUNCTION DECLARATION

s = tf.compat.v1.InteractiveSession()

def input_layer(n, input):
    layer = Conv2D(n, (3, 3), padding='same')(input)
    layer = BatchNormalization()(layer)
    layer = LeakyReLU(alpha=0.2)(layer)
    layer = Conv2D(n, (3, 3), strides=1, padding='same')(layer)
    layer = BatchNormalization()(layer)
    layer = LeakyReLU(alpha=0.2)(layer)

    return layer
```

```
def maxPool(n, input):
    layer = MaxPool2D((2, 2), strides=2)(input)

    for i in range(2):
        layer = Conv2D(n, (3, 3), padding='same')(layer)
        layer = BatchNormalization()(layer)
        layer = LeakyReLU(alpha=0.2)(layer)

    return layer

def upConv(n, input_1, input_2):
    layer = Deconv2D(n, (2, 2), strides=2)(input_2)
    layer = Concatenate()([input_1, layer])

    for i in range(2):
        layer = Conv2D(n, (3, 3), padding='same')(layer)
        layer = BatchNormalization()(layer)
        layer = Activation('relu')(layer)

    return layer

#define the model
def UNet(x_shape):
    input = Input(x_shape)

    #convolution
    X0 = input_layer(64, input)
    X1 = maxPool(128, X0)
    X2 = maxPool(256, X1)
    X3 = maxPool(512, X2)
    X4 = maxPool(1024, X3)

    #up convolution
    X4 = upConv(512, X3, X4)
    X3 = upConv(256, X2, X4)
    X2 = upConv(128, X1, X3)
    X1 = upConv(64, X0, X2)
    #to 3 channel
    X0 = Conv2D(3, (1, 1), strides=1)(X1)

    #define the output model
    model = Model(inputs=input, outputs=X0)

    return model

#load the previous training history
def load_previous_epoch(path_history):
    file = open(path_history, "r")
    previous_epoch = 0

    CoList = file.read().split("\n")

    file.close()
    for i in CoList:
        if i:
            previous_epoch += 1
    return previous_epoch

#testing and show the images
def test_sample(color, gray, w, e):
    output = model.predict(gray)

    #convert from BGR to RGB
    output = output[...,::-1]
    color = color[...,::-1]
    gray = gray[...,::-1]

    n=1
    for i in range(3):
        plt.subplot(3, 3, n)
        plt.imshow(color[i].reshape((w, w, 3)), cmap="gray",
interpolation='none')
        plt.subplot(3, 3, n+1)
        plt.imshow(gray[i].reshape((w, w)), cmap="gray",
interpolation='none')
        plt.subplot(3, 3, n+2)
        plt.imshow(output[i].reshape((w, w, 3)), cmap="gray",
interpolation='none')
        n += 3

    #save_test_path = '/content/gdrive/My
Drive/colorization/256/img/Epoch_'+str(e)+'.jpg'
    #plt.savefig(save_test_path)
    plt.show()

#read images and convert to array form
def batch_generator():
    images = []

    train_list = random.sample(range(TOTAL_TRAIN),
TRAIN_IMAGES_PER_EPOCH)
    test_list = random.sample(range(TOTAL_TEST),
TEST_IMAGES_PER_EPOCH)
    t_load = TRAIN_IMAGES_PER_EPOCH + TEST_IMAGES_PER_EPOCH
    path_list = train_list + test_list

    i = 0
    for id in train_list:
        filename = img_path_train[id]
        img = cv2.imread(filename)

        img=cv2.resize(img,(w,w))
        images.append(img)

        sys.stdout.write("\rLoading training img {}/{}".format(i+1,
TRAIN_IMAGES_PER_EPOCH))
        sys.stdout.flush()

        i = i + 1

    i = 0
    for id in test_list:
        filename = img_path_train[id]
        img = cv2.imread(filename)

        img=cv2.resize(img,(w,w))
        images.append(img)

        sys.stdout.write("\rLoading testing img {}/{}".format(i+1,
TEST_IMAGES_PER_EPOCH))
        sys.stdout.flush()

        i = i + 1

    images = np.array(images)
    images = images/255.

    gray_images = np.mean(images, axis=-1)
    gray_images = gray_images.reshape((*gray_images.shape, 1))

    return images[:TRAIN_IMAGES_PER_EPOCH],
gray_images[:TRAIN_IMAGES_PER_EPOCH],
images[-TEST_IMAGES_PER_EPOCH:], gray_images[-TEST_IMAGES_PER_EPOCH:]

######################################################################
######## DEFINE PARAMETERS
EPOCH = 2000
BATCH_SIZE = 16
TRAIN_IMAGES_PER_EPOCH = 2048
TEST_IMAGES_PER_EPOCH = 512
load_PreTrain = True

w = 256

######################################################################
####### LOADING PREVIOUS INFORMATION
x_shape = (w,w,1)
model = UNet(x_shape)

model.compile('adam', loss='mean_squared_error', metrics=['mae',
'acc'])
#model.summary()

if load_PreTrain:
    model.load_weights("./model.hdf5")
    previous_epoch = load_previous_epoch('./history.txt')

else:
    previous_epoch = 0

print('Start from Epoch : {}'.format(previous_epoch))

######################################################################
######## TRAINING
for epoch in range(previous_epoch, previous_epoch+EPOCH):

    print('EPOCH : {}'.format(epoch))

    Y_train, X_train, Y_test, X_test = batch_generator()

    hist = model.fit(X_train, Y_train, batch_size=BATCH_SIZE,
validation_data=(X_test, Y_test))

    record = ("{} {} {} {} {}
{}\n".format(hist.history['loss'][0],hist.history['mae'][0],hist.hist
ory['acc'][0],hist.history['val_loss'][0],hist.history['val_mae'][0],
hist.history['val_acc'][0]))

    with open("./history.txt", "a") as file_object:
        file_object.write(record)

    model.save('./model.hdf5')

    test_sample(Y_test[:3], X_test[:3], w, epoch)
```