REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

GRAY-SCALE IMAGE COLORIZATION USING DEEP LEARNING

19011915 — Parsa Kazerooni
19011069 — Utku Mehmetoğlu

COMPUTER PROJECT

Advisor
Assist. Prof. Dr. Hamza Osman İLHAN

Mayıs, 2022

ACKNOWLEDGEMENTS

We would like to acknowledge and give our thanks to Assist. Prof. Dr. Hamza Osman İLHAN who made our work come to a formal shape and gave us a different perspective that made this work possible.

Parsa Kazerooni
Utku Mehmetoğlu

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ADAM	Adaptive Moment Estimation
BatchNorm	Batch Normalization
CIE	International Commission on Illumination
CNN	Convolutional Neural Network
CUDA	Compute Unified Device Architecture
GAN	Generative Adversarial Network
MSE	Mean Squared Error
MSSIM	Mean Structure Similarity
PSNR	Peak Signal-to-Noise Ratio
ResNet	Residual Learning Network
VGG	Visual Geometry Group

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

GRAY-SCALE IMAGE COLORIZATION USING DEEP LEARNING

Parsa Kazerooni
Utku Mehmetoğlu

Department of Computer Engineering
Computer Project

Advisor: Assist. Prof. Dr. Hamza Osman İLHAN

Our project aims to build a deep-learning based model in order to colorize gray-scale images. Since our main goal is to satisfy a human viewer in an automatic way, we chose study methods that predicts colors that are appealing to the eye. The system is implemented as a Self-supervised feed-forward pass at test time in a pre-trained Convolutional Neural Network. Its performance is measured by both numeric analysis such as DeltaE and by a "Human Turing Test" survey that checks for human satisfaction rate.

Keywords: Image Colorization, Deep Learning, CNN, ResNet, Delta E

ÖZET

Derin Öğrenme ile Griseviye Fotoğrafların Renklendirmesi

Parsa Kazerooni
Utku Mehmetoğlu

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Dr. Ögr. Üyesi Hamza Osman İLHAN

Projemiz, gri seviye resimleri renklendirmek için derin öğrenme tabanlı bir model oluşturmayı amaçlıyor. Buradaki ana amacımız bir insanı otomatik olarak, tatmin etmek olduğundan, göze hoş görülen renklendirmeler üreten bir yaklaşım izlendi. Sistem, test zamanında bir CNN'de ileri besleme geçişi olarak uygulanır ve bir milyondan fazla renkli görüntü üzerinde eğitildi. Performans, DeltaE gibi matematiksel yaklaşımlar kullanarak çıktıları test eden yaklaşımlar ve insan geri bildirimini inceleyen bir "insan Turing testi" anketi kullanarak değerlendirilir.

Anahtar Kelimeler: Renklendirme, Derin Öğrenme, CNN, ResNet, Delta E

1
Introduction

Gray scale images are images whose pixels are constructed only by the amount of brightness and nothing else. However human eye perceives colors in different aspects, brightness, color, intensity and etc. Thus such information on a gray-scale photo is lost. Our aim in this project is to recover these lost colors as much as possible using the power of deep-learning. A human can paint a gray-scale image based on their past experiences which is a lifetime of familiarity for colourful objects, landscapes and life forms. But given such case a person may not paint a gray-scale photo to the exact same colors which the photo originally consist of, but still the painted image in most cases will look pleasant to human eye and satisfies a viewer. In case of teaching a machine exposing it to a large image dataset, which we prepared before, and feeding it gray-scale channels as input and other channels as output to teach our model is our goal in this project. We aim to produce a pleasant to look, vivid and sharp images and to test the pleasantness to the human eye we are going to use a "Human Turing test" where we survey a group of people with mostly artificially colored and native images where we ask our test group whether the image is actually natural or colored by machine.

1.1 Motivation

By having a model that colorizes a gray-scale image accurate enough, multiple outcomes can be achieved. Historical footage can be brought to life [1], old pictures can be seen the way the cameraman did back then, and all other use-cases. Also it can be served as a "lossy compression" method which will be discussed in Future works as well (Chapter 9).

1.2 Organization

Chapter 3 looks into the system's feasibility. Chapter 5 examines the training plan and requirements. Chapter 6 discusses about the training process and benchmarks.

Chapter 7 investigates on different ways to measure the accuracy of colorization in a numeric sense and applies the best one. Chapter 8 studies what users think of the generated results. Chapter 9 gives us a brief conclusion and potentials for the future.

2
Preview

For years, artists have been actively colorizing historical footage, old pictures. To help the process, the usage of artificial intelligence was introduced. Automatic Image colorization is still an ongoing research area with multiple solutions and methods. Some of them are discussed below:

2.1 User-guided colorization

User-guided networks was introduced in 2004 in an article proposed by Levin et al. [2]. The method needed user's provided scribbles with the target colors, then it colorized by filling each region with the related scribble using least-square method. Then, Huang et al. [3] propose an adaptive edge-detection based colorization method to reduce the amount of artifacts created by poor region detection. Also texture features were implemented by Qu et al. [4] and Luan et al. [5] to reduce the amount of scribbles needed. [6]

2.2 Plain Networks

This was an interesting method at the time and it was used by artists and animators. But the results were heavily depended by users abilities. The utilization of CNNs for image processing in the last years, grew significantly. Deep Colorization Method proposed by Z. Cheng [6] was introducted by using image descriptors as an input for a deep neural network. Iizuka's method was introduced a fusion layer for combining global priors and local image features as an input [7]. But most of the most of the methods were generating desaturated results. Zhang el al. [8] found a solution to this issue by using classification methods. To increase the diversity of colors in the final output, class-rebalancing is used during training. [9]

2.2.1 Deep Colorization

In this study various images from image-sets are clustered into different image colorization (ie. landmark images, city images, view images, person images) using an adaptive image colorization technique. Later chrominance values for the data-set is used to train the model. The model basically provides a chrominance value for a given image based on clusters. The architecture used in this model consists of a neural network which has the equal size of neurons in both input and output neurons, further the neuron size in the hidden layer is set to half of that count. ReLU is utilized as activation function in this method. In feature design work, images are separated as 3 levels of features as low high and mid. Adaptive Image Clustering technique is also applied. The reference images are clustered with k-means algorithm and peak signal to noise ratio is computed between the original image and colorized result. If the computed result fails to meet a certain threshold the image is removed from the image-set. Also semantic clustering is applied to clustering method which helps to differentiate between images that are globally similar and semantically different from each other. [6]

2.2.2 Colorful Colorization

The network stacks two or three convolutional layers, ReLU layer and Batch normalization layer together to form eight blocks. To have vibrant colors, the method rebalances the loss based on the pixel rarity in training time. By doing this, less colorful backgrounds such as sky, won't affect the colorization process. It's a multi-modal scheme that each pixel has a probability distribution for each color. [8] [10]

2.3 Various methods

It's worth mentioning some different approaches and specific use cases as well.

2.3.1 Domain-specific Networks

Such as Infrared Colorization by Limmer et al. [11] which colorizes RGB images to Near Infrared (NIR) images using a multi-branch CNN. Additionally Song et al. proposed Radar image colorization [12] which acts as a feature-extractor on raw single-polarized radar images. Since domain knowledge is applied, their performance is usually high, but the use case is specific and it can not be generalized.

2.3.2 Text-based methods

Text-based Networks apply methods which can be a part of user-guided techniques, But instead of colored scribbles, they are guided through text hints. Manjunatha et al. [13] deployed a model by using Colorful Colorization's classifier FCNN [8] to be trained with image captions. Text2Color was also introduced by Bahng et al. [14]. It's implemented by using two conditional GANs, to generate a pallete by words, and to use the pallete for the colorization. They can be advantageous in the situations were image descriptions are already available, but they require an accurate text input. [10]

3
Feasibility Study

3.1 Technical Feasibility

3.1.1 Hardware Feasibility

We tried experimenting with multiple runners such as MobileNet-V3, EfficientNet, SqueezeNet and ResNet and etc., we trained them on GPU with Nvidia CUDA libraries of PyTorch. due to the size of dataset, Google Colab or other free cloud-based providers, couldn't handle the process because it required high amounts of dedicated resources. Therefore we switched from cloud-based provider to local hardware as shown in the Table 3.1.

Table 3.1 Hardware Specifications

Hardware	Specifications
GPU	Nvidia GTX 1660Ti
Clock Speed	1860 MHz
Virtual RAM	6 Gib VRAM

3.1.2 Software Feasibility

Python appears to be the language of choice for artificial intelligence applications, according to studies. Our project was likewise built using Python and its open-source libraries. Following is a list of the libraries we used:

- NumPy: NumPy is the most important Python package for scientific computing. It provides multidimensional array objects and different derived objects such as matrices. It has a speed advantage over native Python arrays due to the fact that it is developed in C.
- PyTorch: PyTorch is an open-source python machine learning framework. it is mainly used for applications such as computer vision and natural language

processing. Also it has CUDA support for Nvidia graphics cards which accelerates the training and testing processes.

- Fast.ai: It's a library made by none-profit organization named "fast.ai". The library's purpose is to make training of deep neural networks as easy as possible and make the training process fast and accurate.
- Matplotlib: Matplotlib is a plotting library for Python. It provides customizable plots and helps to plot charts and images.

3.2 Time Feasibility

Figure 3.1 shows the project's Gantt diagram in details.

3.3 Legal and Economic Feasibility

All the materials and tools used in this project are free-to-use and most of them are open-source and those that require affiliation are mentioned. Also since this is a research-based project, it doesn't serve any commercial usage and it will be non-profit as well.

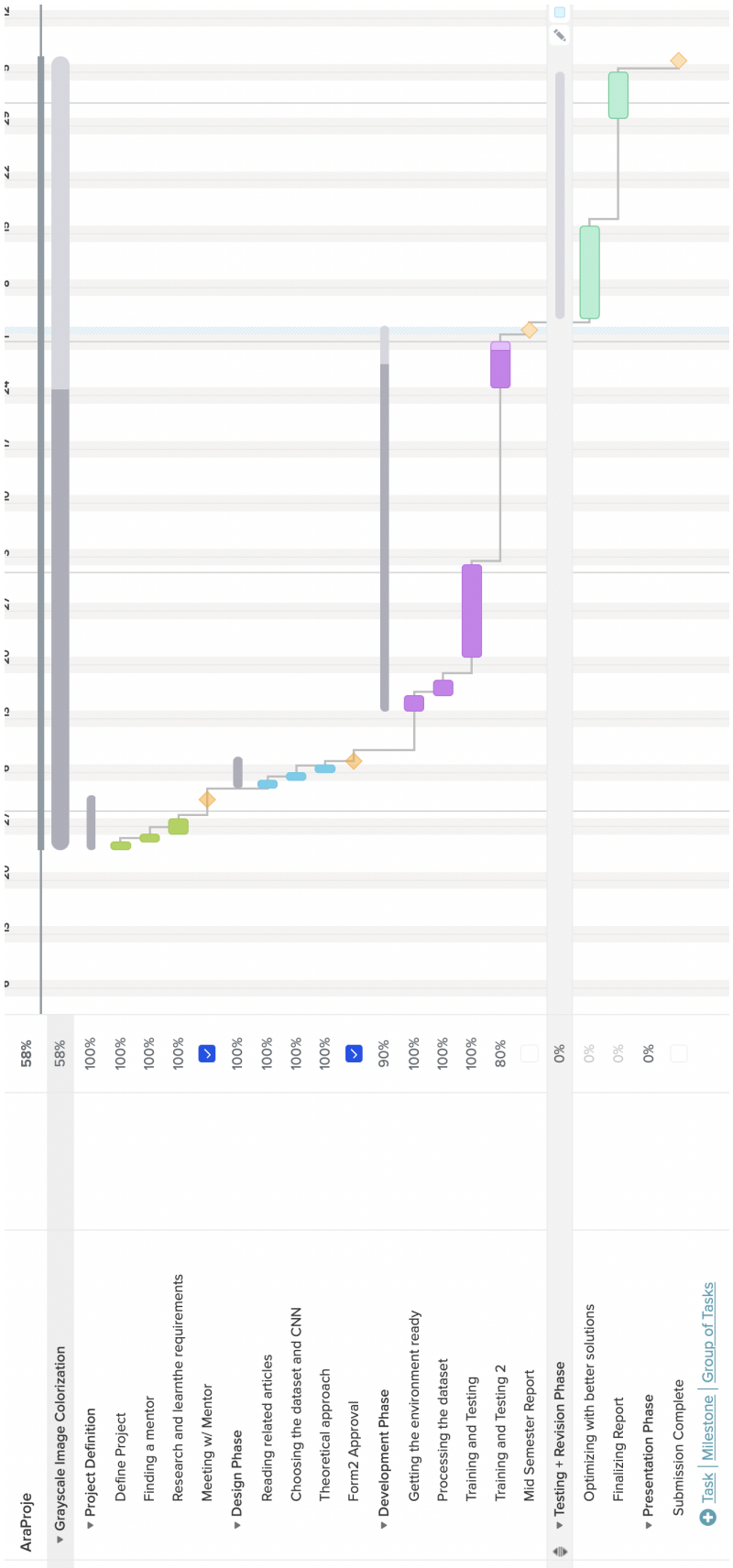

Figure 3.1 Gantt Diagram(Sideways)

4
System Analysis

The colors of an image can be represented different. The most common color spaces are RGB (Red, Green, Blue), CMYK (Cyan, Magenta, Yellow, Key). CIELAB is a color space which represents the colors by their L* as luminance, and a* channel as red-green pair, and b* as yellow-blue pair. CIELAB is a good representation to human vision, since our aim is to generate colors to seem natural to humans, CIE Lab color space is preferred. Also, by using CIE Lab, the luminance channel can be directly achieved by the gray-scale image. This means that our model only needs to predict ab channels. But using RGB color space, forces us to predict three different values.

To have a better usability, it's better to design a system that doesn't restrict the user to only use images with CIE Lab color space. Figure 4.1 demonstrates a simple system that adapts common color spaces as well. The system's input is a gray-scale image with RGB color space. And the output is expected to return a image with the same structure, Therefore, the predicted ab values are combined with the original L channel to create the output image, then converted back to RGB color space to achieve a seamless user experience.

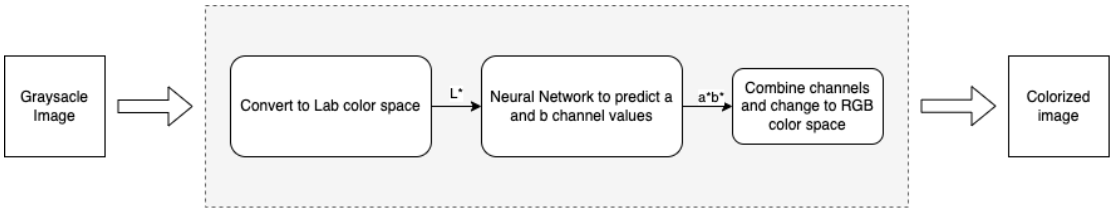

Figure 4.1 System's diagram

5
System Design

Given a image in CIE Lab colorspace with the size of $H \times W$, the input is its Lumminance channel demonstrated as $X_L \in \mathbb{R}^{W \times H \times 1}$. The target is to predict a and b channel values. The target is to obtain a accurate mapping between Luminance values and ab values for prediction. denoted as below

$$\mathscr{F} : \mathbf{X_L} \rightarrow \left(\hat{\mathbf{X}}_\mathbf{a}, \hat{\mathbf{X}}_\mathbf{b}\right) \tag{5.1}$$

The colorized image, $\hat{X} \in \mathbb{R}^{W \times H \times 3}$, is constructed by combining X_L with $\hat{X}_a$ and $\hat{X}_b$.

$$\hat{X} = (X_L, \hat{X}_a, \hat{X}_b) \tag{5.2}$$

5.1 Model Design

5.1.1 Objective Function

MSE (Mean Squared Error) is used as the objective function. it's defined by taking the arithmetic mean for the squared distance of each pixel's individual channel values between the prediction and the ground truth.

$$L = \frac{1}{2HW} \sum_{k \in \{a,b\}} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(X_{k_{i,j}} - \hat{X}_{k_{i,j}}\right)^2 \tag{5.3}$$

We also considered to use CMSSIM as the objective function but its details is explained in the Section 4.1.1.

5.1.2 Network Architecture

ResNet-18: is a CNN trained on ImageNet dataset, presented in the article "Deep Residual Learning for Image Recognition" [15]

The Network Architecture is shown in Figure 5.1.

layer name	output size	18-layer
conv1	112×112	
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$
	1×1	
FLOPs		1.8×10^9

Figure 5.1 ResNet18 Architecture [16]

5.2 Dataset Prepration

As mentioned, we use MirFlicker[17] as the dataset to train and test our network. it contains pictures varying from green landscapes to human-made objects.

Preprocessing: Converted to Lab colorspace, extracted L and ab channels into individual numpy arrays.

After the training is done, our network tries to predict the best combination of ab values. By adding the lightness channel (which is given as input), we would be able to construct the full picture. (Figure 5.2)

Figure 5.2 a picture with ab channels only, added lightness values to recreate the original picture

6
Implementation

6.1 Training The Model

The dataset is splitted into 20% dedicated as train set and 80% as test set. Each training epoch consists of 1250 loops and the Batch Size of 16. Total of 15 Epochs are iterated through the learning process. On our hardware, it took time of 2:30 hours to train the model. (15 minutes per epoch, 15 epochs in total)

Figure 6.1 demonstrates the learning progress in different stages. Until 3rd epoch, results are not acceptable at all, As they are mostly colored like old Sepia pictures, but after some iterations, accuracy increases and after the 8th epoch, most of the results are accurate enough.

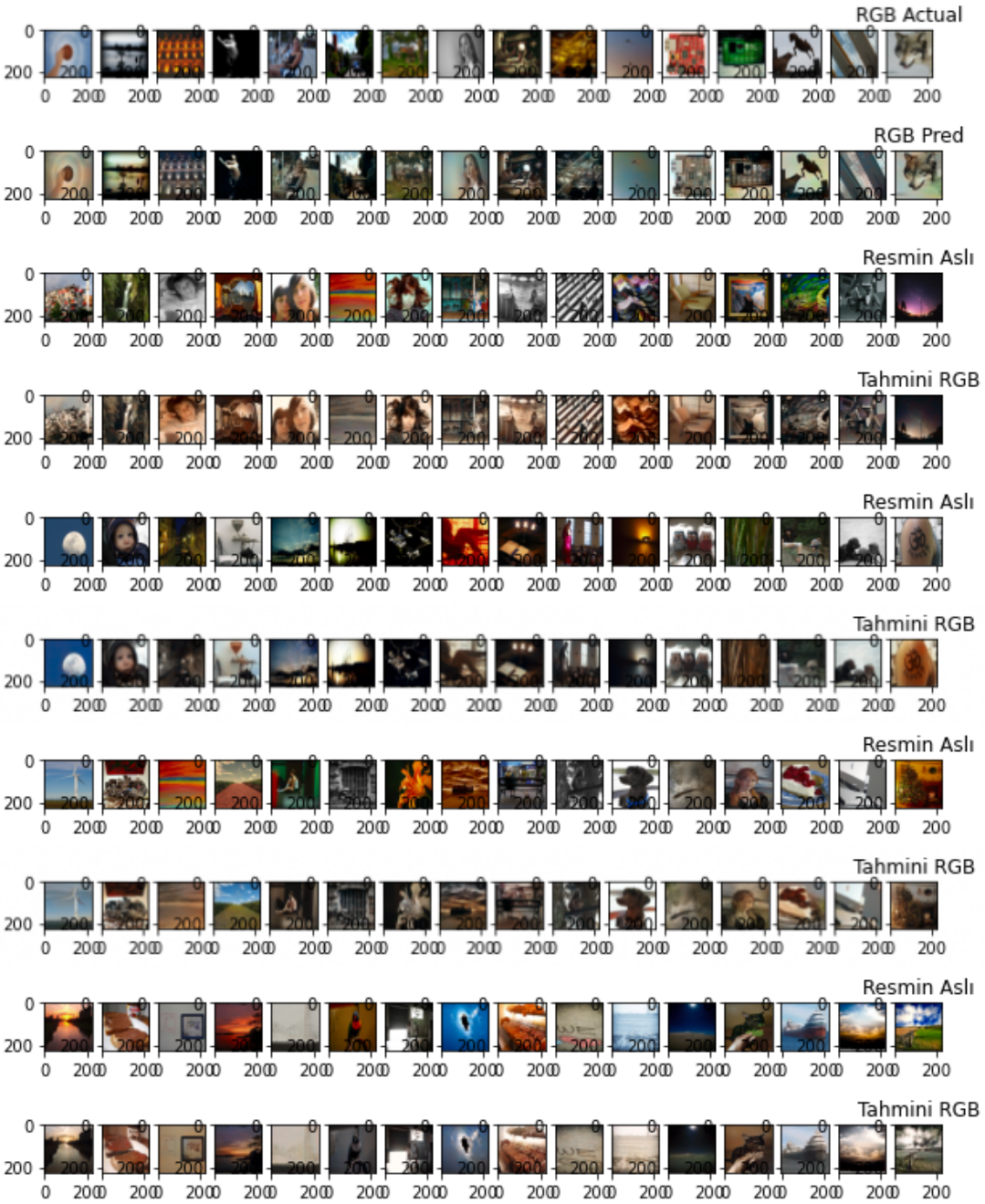

Figure 6.1 Model predictions at epoch 1, 3, 8, 14, 15

6.2 Testing

As shown in figures 6.2 and 6.3, the model predictions are quite accurate or at least the fact that they're artificial is unnoticeable. But as it's demonstrated, some of the human-made objects are in a completely different color, some of those aren't usually linked with a specific color and no human would tell if it's in a "wrong" color, but some of them do have a globally-known color such as Turkish Flag. The Classification Method improves the colorization of these kind of objects.

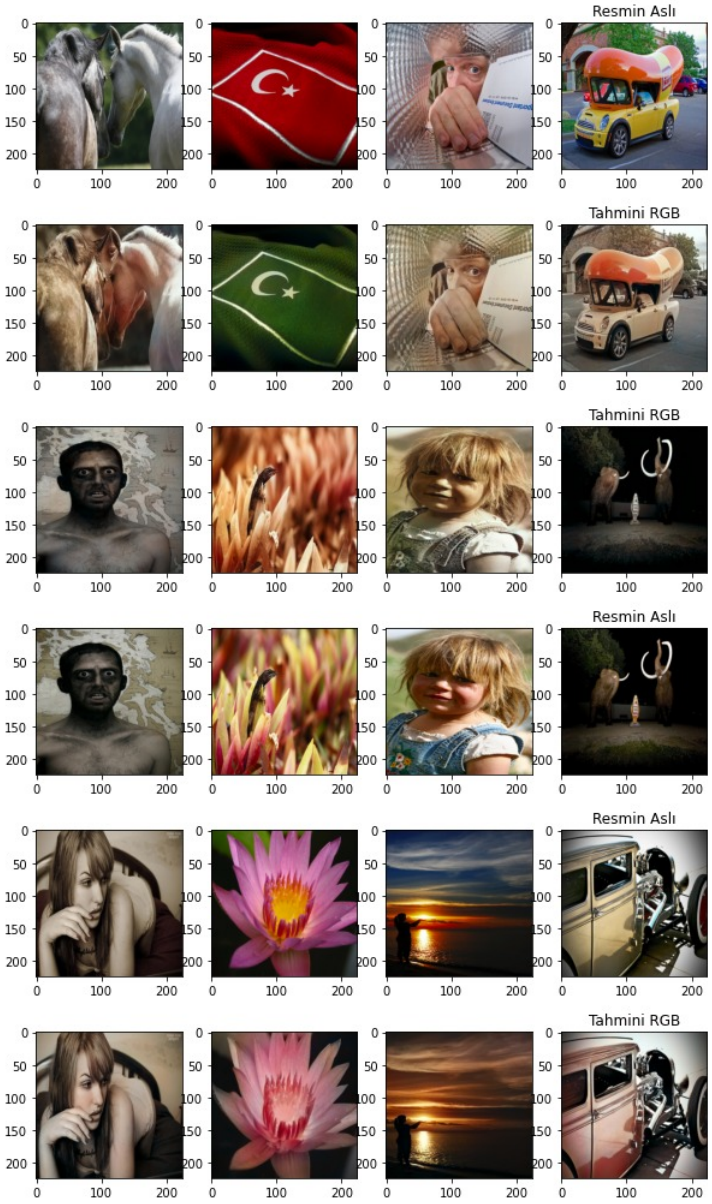

Figure 6.2 Some of the model's results

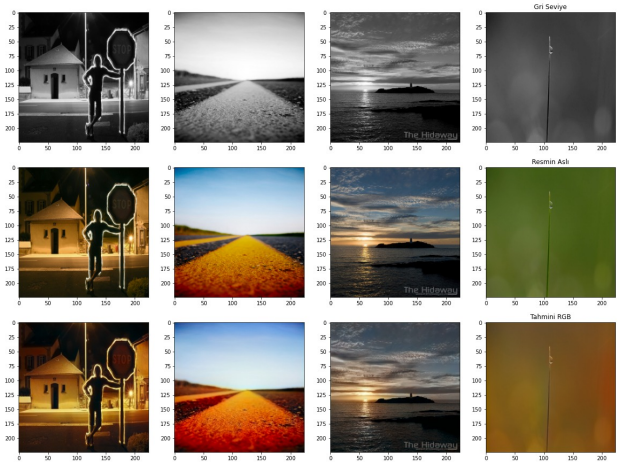

Figure 6.3 Gray Scale, Actual and Predicted image

6.3 Model Benchmarks

As shown in figure 6.4, train loss tends to decrease over time, but test loss isn't decreasing significantly. However the prediction's accuracy is measured later on and it's satisfying enough. But further research is planned.

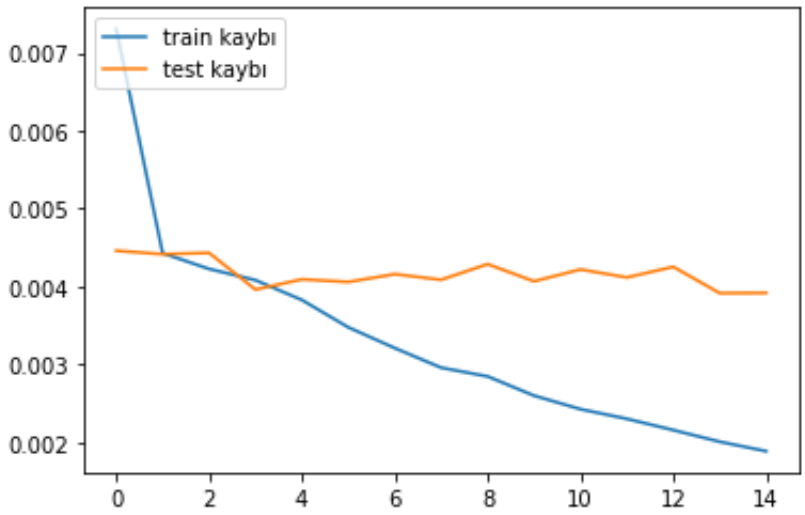

Figure 6.4 The plot of train and test loss in each epoch

7
Performance Analysis

To measure how close the predicted image is, there are many approaches. e.g. Evaluating based on human feedback such as "Colorization Turing Test" used by Richard Zhang's Colorful Colorization method [8]. It determines the realness of the generated image by the percentage of the 'yes' answers of the question "Is it real?" in a survey. It can be a good metric since it's directly related to the target of this research. But it can be a slow process. Therefore We investigate and define autonomous methods.

7.1 Color Distance Metrics

7.1.1 CMSSIM and other Metrics

Multiscale Structural Similarity (MSSIM) is evaluated by Luminance, Contrast, Color, Gradient and other properties of two images to determine their similarity [18]. By calculating the Euclidean distance between two colors and applying a threshold filter, we can estimate how accurate the color prediction performed [19]. But it can be inconsistent if not performed correctly, more details are examined in [20].

7.1.2 Mean Delta E

Delta E (Empfindung) is used to measure the difference between two colors perceived by human. Table 7.1 describes the Delta E values. [21] It has the potential to be the perfect candidate metric to determine the similarity of a generated image with the ground truth.

Table 7.1 Delta E ranges and Perception [21]

Delta E	**Perception**
≤ 1.0	Not perceptible by human eyes.
1 - 2	Perceptible through close observation.
2 - 10	Perceptible at a glance.
11 - 49	Colors are more similar than opposite
100	Colors are exact opposite

It is usually applied to "perceptually uniform" colorspaces such as CIELab. there are three versions of deltaE, dE76, dE94, and dE00.

dE76 is very similar to Euclidiean Distance Metric as shown below.

$$\Delta E^*_{ab} = \sqrt{(L^*_2 - L^*_1)^2 + (a^*_2 - a^*_1)^2 + (b^*_2 - b^*_1)^2} \tag{7.1}$$

18 years later, **dE94** came out as an improvement. It introduced weighted parameters such as luminance, hue, chroma. The formula is shown below

$$\begin{aligned}
&\Delta E^*_{94} = \sqrt{\left(\frac{\Delta L^*}{k_L S_L}\right)^2 + \left(\frac{\Delta C^*_{ab}}{k_C S_C}\right)^2 + \left(\frac{\Delta H^*_{ab}}{k_H S_H}\right)^2} \\
&\Delta L^* = L^*_1 - L^*_2 \\
&C^*_1 = \sqrt{a^{*2}_1 + b^{*2}_1} \\
&C^*_2 = \sqrt{a^{*2}_2 + b^{*2}_2} \\
&\Delta C^*_{ab} = C^*_1 - C^*_2 \\
&\Delta H^*_{ab} = \sqrt{\Delta E^{*2}_{ab} - \Delta L^{*2} - \Delta C^{*2}_{ab}} = \sqrt{\Delta a^{*2} + \Delta b^{*2} - \Delta C^{*2}_{ab}} \\
&\Delta a^* = a^*_1 - a^*_2 \\
&\Delta b^* = b^*_1 - b^*_2 \\
&S_L = 1 \\
&S_C = 1 + K_1 C^*_1 \\
&S_H = 1 + K_2 C^*_1
\end{aligned} \tag{7.2}$$

Finally in 2000, **dE2000** or **dE00** was introduced, it solved some issues with eD94 but it's much more complex. [22] [23]

We used dE94 because it's a good balance between accuracy and complexity. Since It's a per-pixel metric, there should be a generalization method for the whole image. By

taking mean of pixel delta E values, we can obtain a single metric to discuss the color similarity of the prediction and ground truth.

Also, We can visualize the color differences to obtain more details on performance by plotting all the delta E values as a 2d colored map. Figure 7.1 demonstrates the idea.

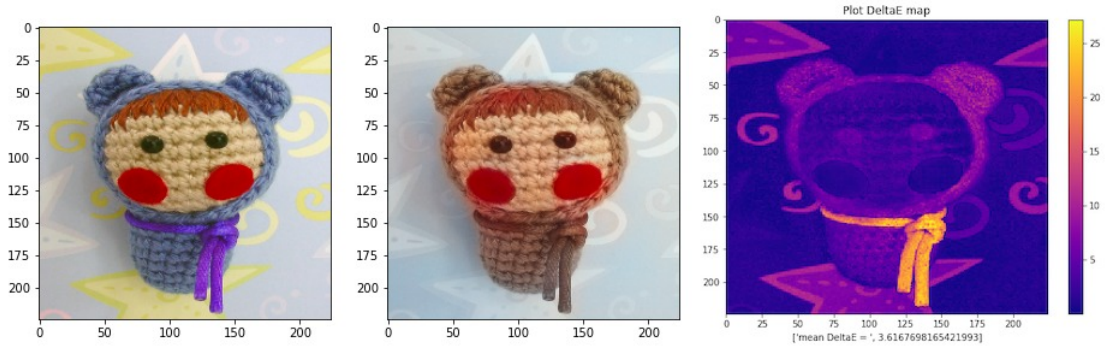

Figure 7.1 Ground Truth, Prediction, and Delta E Heat-map

Figure 7.2 gives us a good understanding of the output's color probability distribution. Since it takes the arithmetic mean, some images that are mostly a plain background and a small object, can be misrepresented as a accurate prediction since the impact of the small object is low. This can be fixed by considering weights when calculating overall delta E. But for now, it can be a good approximation for a color similarity measurement.

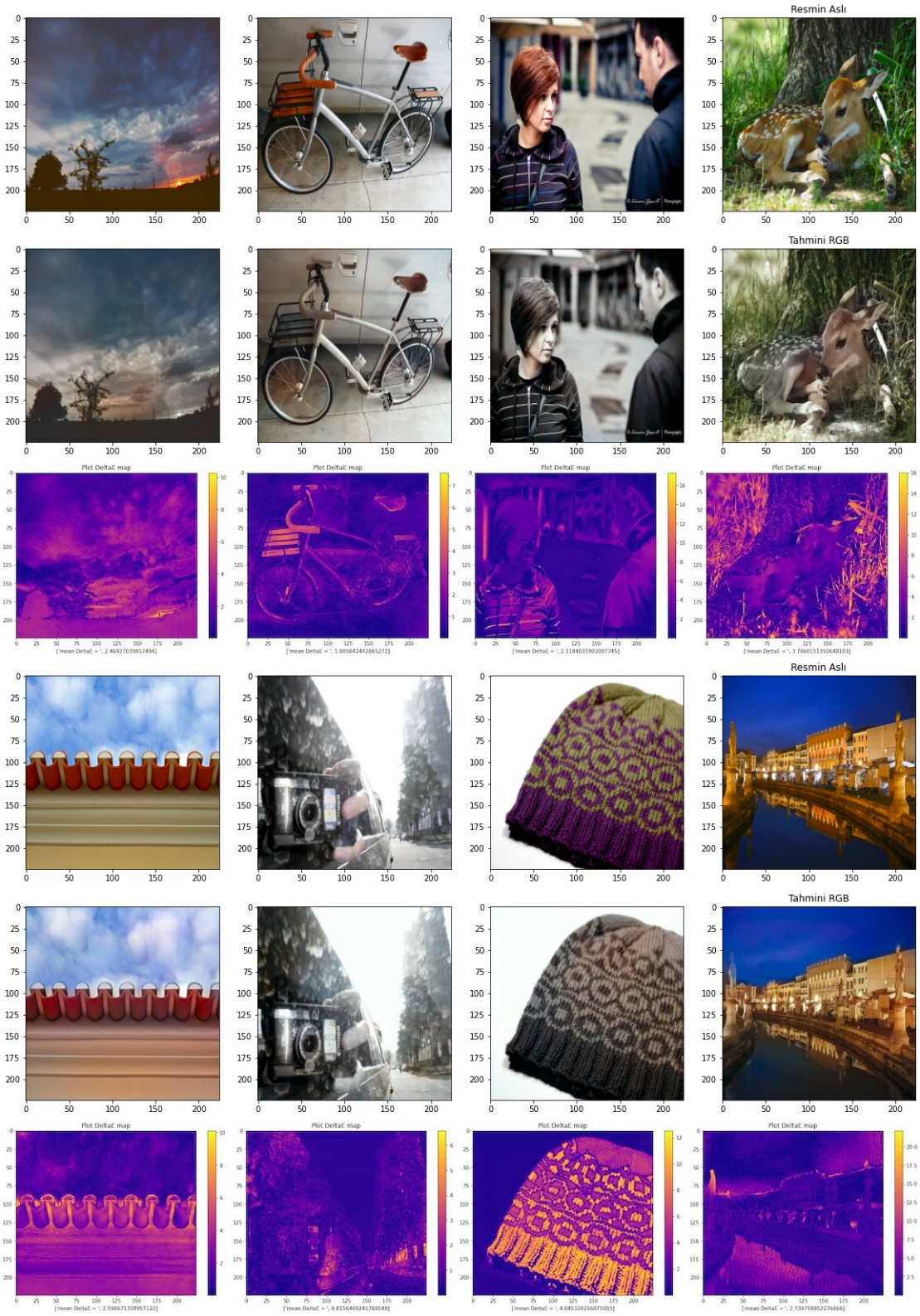

Figure 7.2 Delta E map for the tests

8
Experimental Results

As mentioned before, to determine how real the colorization performed, a survey was prepared to study humans' opinion. Since the results are a meaningful determination, our model's ability to trick humans is also evaluated by a survey.

8.1 Human Feedback Study

A simple survey was prepared with 9 randomly selected images consisting colorized outputs and natural images. the performance was measured by asking *Is the shown image real?* and the yes/no ratio of the answers. The images were not cherry-picked, but most of them contained sky and landscapes which could result to a biased sample, since the colorizing them is usually well performed. Submissions were registered by 40 people uniformly selected from Computer Engineering Students, Animation And Graphic Design Students, people not related to a specific academic area and older citizens. Total of 360 answers were given, containing 224 false answers, thus the total human error rate is 62%. This is a good sign because it shows that the colorized images were close to a real image and it tricked most humans.

Figure 8.1 contains the results of the survey. Each image is evaluated by the percentage of the people who thought the image was real.

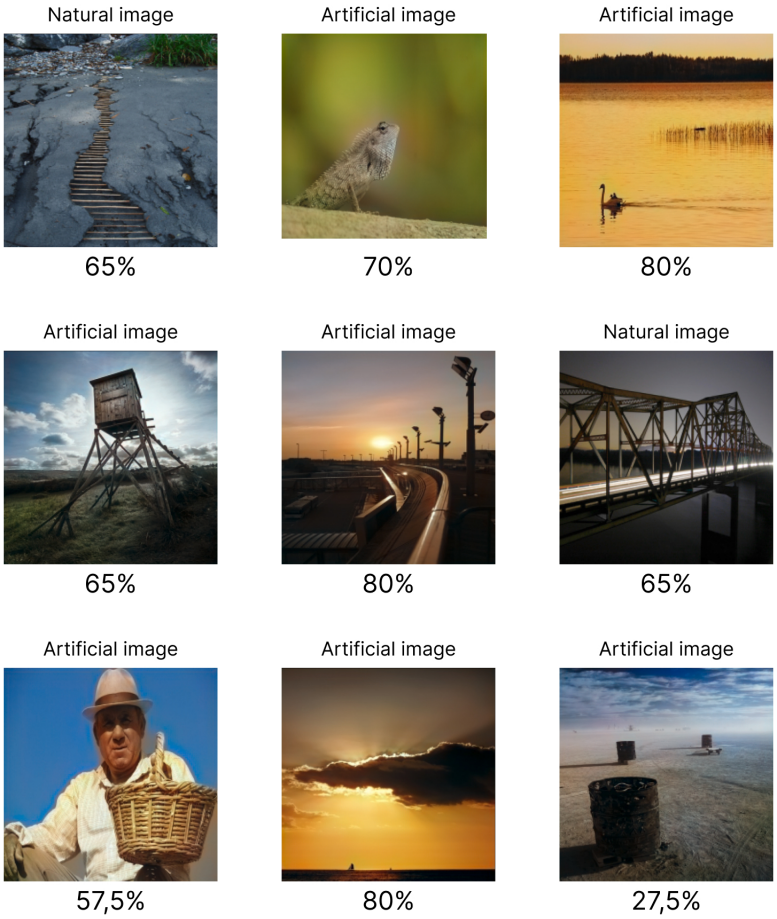

Figure 8.1 Results of the survey

9
Conclusion

9.1 Future Work

By analyzing the aspects of the problem, the inevitable fact that Self-supervisied Colorization require some supervised task to be performed to be more close to the reality, tends to be a challenge that leads to a large number of researches on a more precise Automatic process for the future.

Also, by utilizing the model, we can deliver a colorization service to the users. A simple web application is planned to be developed. Additionally, our model could be used to save space for different services, since gray-scale images contain less information than a colorful image, a service might use our model as a "lossy compression" method to store images that their color accuracy is not concerned.

9.2 Conclusion

To conclude, we trained a model that is color accurate, good looking and sharp in most cases. Our further analysis showed us that the colorization technique might not be perfectly accurate but it can provide us a good representation of what the grayscale image might looked like if it was a colorful one. Also in some cases our model struggled with unnatural or extremely vivid colours but nevertheless given the hardware limitations and the colours that disappear due to grayscale conversion we think our model predicted most of the colors right and provided pleasant looking images.

References

[1] M. Simon, "Ai magic makes century-old films look new," Aug. 2020.

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

[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," in *Proceedings of the 13th annual ACM international conference on Multimedia*, 2005, pp. 351–354.

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

[5] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proceedings of the 18th Eurographics conference on Rendering Techniques*, 2007, pp. 309–320.

[6] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 415–423.

[7] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (ToG)*, vol. 35, no. 4, pp. 1–11, 2016.

[8] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*, Springer, 2016, pp. 649–666.

[9] L. R.-G. Federico Baldassarre Diego Gonzalez-Morin, "Deep-koalarization: Image colorization using cnns and inception-resnet-v2," *ArXiv:1712.03400*, Dec. 2017. [Online]. Available: `https://arxiv.org/abs/1712.03400`.

[10] S. Anwar, M. Tahir, C. Li, A. Mian, F. S. Khan, and A. W. Muzaffar, "Image colorization: A survey and dataset," *arXiv preprint arXiv:2008.10774*, 2020.

[11] M. Limmer and H. P. Lensch, "Infrared colorization using deep convolutional neural networks," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2016, pp. 61–68.

[12] Q. Song, F. Xu, and Y.-Q. Jin, "Radar image colorization: Converting single-polarization to fully polarimetric using deep neural networks," *IEEE Access*, vol. 6, pp. 1647–1661, 2017.

[13] V. Manjunatha, M. Iyyer, J. Boyd-Graber, and L. Davis, "Learning to color from language," *arXiv preprint arXiv:1804.06026*, 2018.

[14] H. Bahng *et al.*, "Coloring with words: Guiding image colorization through text-based palette generation," in *Proceedings of the european conference on computer vision (eccv)*, 2018, pp. 431–447.

[15] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. DOI: 10.48550/ARXIV.1512.03385. [Online]. Available: https://arxiv.org/abs/1512.03385.

[16] "Resnet pytorch." (), [Online]. Available: https://pytorch.org/hub/pytorch_vision_resnet/.

[17] Liacs. "Mirflickr dataset." (), [Online]. Available: https://press.liacs.nl/mirflickr/.

[18] J. Nilsson and T. Akenine-Möller, *Understanding ssim*, 2020. DOI: 10.48550/ARXIV.2006.13846. [Online]. Available: https://arxiv.org/abs/2006.13846.

[19] M. Hassan Husain and C. Bhagvati, "Structural similarity measure for color images," *International Journal of Computer Applications*, vol. 43, pp. 7–12, Apr. 2012. DOI: 10.5120/6169-8590.

[20] A. Fatima, W. Hussain, and S. Rasool, "Grey is the new rgb: How good is gan-based image colorization for image compression?" *Multimedia Tools and Applications*, vol. 80, pp. 1–17, Jan. 2021. DOI: 10.1007/s11042-020-09861-y.

[21] Zschuessler. "Delta e 101." (), [Online]. Available: http://zschuessler.github.io/DeltaE/learn/.

[22] M. Luo, G. Cui, and B. Rigg, "The development of the cie 2000 colour-difference formula: Ciede2000," *Color Research Application*, vol. 26, pp. 340–350, Oct. 2001. DOI: 10.1002/col.1049.

[23] J. Nobbs, "A lightness, chroma and hue splitting approach to ciede2000 colour differences," in Apr. 2002, vol. 5, pp. 46–53.

Curriculum Vitae

FIRST MEMBER

Name-Surname: Parsa Kazerooni
Birthdate and Place of Birth: 22.06.2001, Tahran
E-mail: parsa.kazerooni@std.yildiz.edu.tr
Phone: 0531 400 83 10
Practical Training:

SECOND MEMBER

Name-Surname: Utku Mehmetoğlu
Birthdate and Place of Birth: 01.11.2001, İstanbul
E-mail: utku.mehmetoglu@std.yildiz.edu.tr
Phone: 0541 940 37 62
Practical Training: Defne Telecommunication

Project System Informations

System and Software: Windows/Linux/MacOS, Python
Required RAM: 8GB
Required Disk: 256MB