### **CNG483 INTRODUCTION TO COMPUTER VISION**

### **PROJECT 3: IRISDEEP**

Mehmet Dağhan NAMLIOĞLU - 2152106

Sedat Ali ZEVİT - 2152221

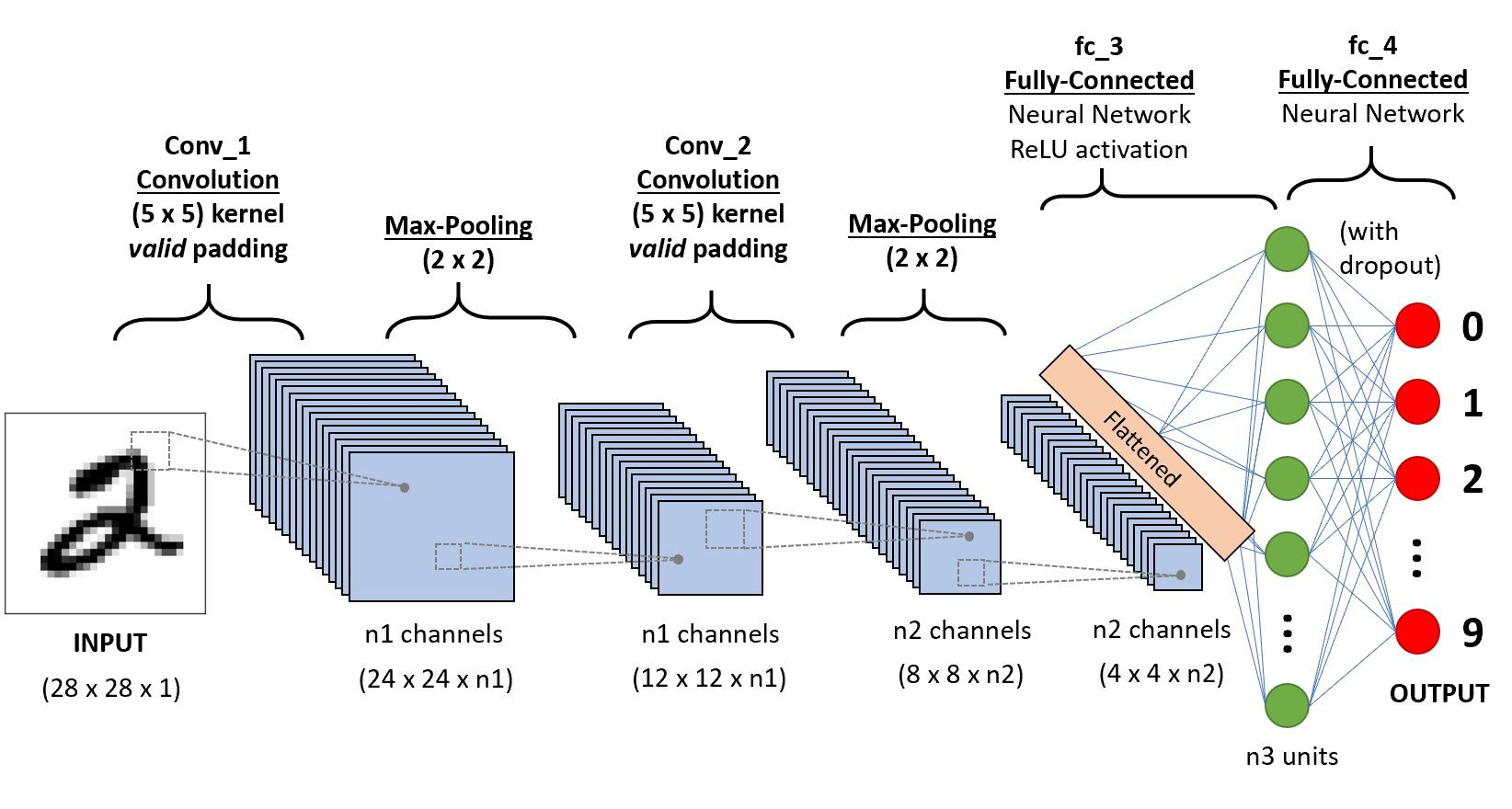
#### **ABSTRACT**

*In this project, we are implementing a convolutional neural network (CNN) for implementing an identity recognition system based on iris recognition. First of all, we arrange datas based on requirements in the assignment. In the assignment, we divide %50 of each person's images for training, %25 images for validation, and the rest of the images are used as validation. Then, we create a model that fits in FIXED network architecture.This means that, the model starts with convolution layer(s), followed by Relu, Pooling, and fully-connected (FC) layers and ends with Softmax layer. In this assignment, we implement multiple convolutional layers with activation functions as Relu. Finally, we tweak hyperparameters to increase accuracy of the model.*

***Index Terms—*** write keywords to describe your project separated with semicolon.

1. **INTRODUCTION**

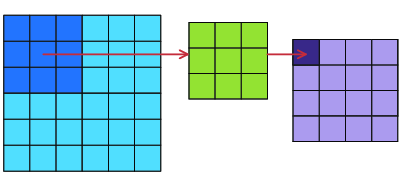
Convolutional neural networks (CNN) are a type of neural networks that are based on convolutions to extract features (or patterns) from the input datas. Convolutional Neural Networks can handle image datas and computer vision very well because CNNs are better than other neural networks (especially ANN) to find local patterns. Also, CNN can learn any local pattern anywhere in the image. A classical CNN is something like in below:



In CNN, there are 3 different layers to implement basically in the model:

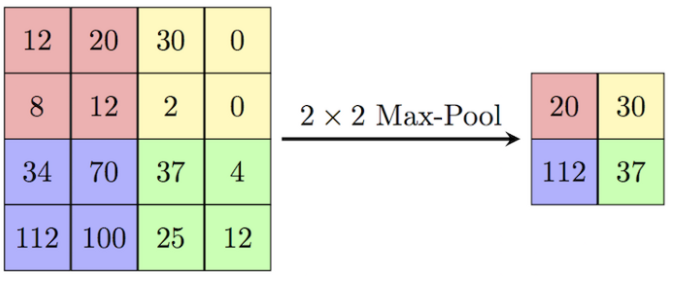
1. **Convolution Layer:**

This layer is the layer to extract features from an input image. Convolution protects the low-level relationships in the image like pixels by learning image features applying small-scaled square parts of input data. It is an operation that takes two inputs such as image matrix and a filter (or kernel). In this layer, we can also change some hyperparameters to tweak better accuracy like input size, kernel size, stride or padding.



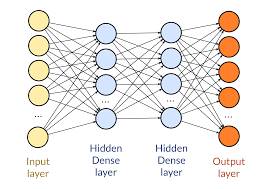
1. **Max-Pooling Layer**

This layer corresponds to a technique of aggressive downsampling of the feature map. This layer generates a new array that takes the maximum value of the specified region in the input array, and puts in the specified location of the new array .



1. **Fully-Connected (Dense) Layer**

This layer is a neural network layer that connects each neuron in the dense layer and receives the input from all neurons of its previous layer. Dense layers perform a matrix-vector multiplication. The values used in the matrix are actually parameters that can be trained and updated with the help of backpropagation, so that the model learns the features of the inputs.



**PREPROCESSING**

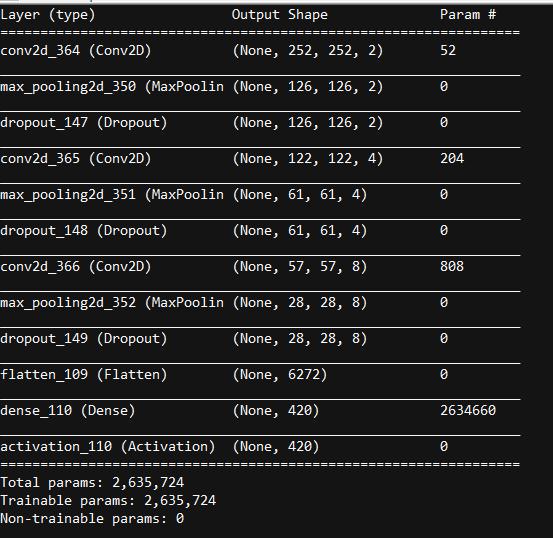
We need to do a set of preprocessing that extracts labels and input images as an array. When we start to design the model, we implement a function that takes each of the images in the dataset folder, and takes the eye pupil location data in the “Parameters.txt” file. And then, this function crops the image based on the coordinates of the eye pupil, and crops as a 256x256 image. The outputs of this function are the array containing image datas and the cropped image array. After that, we normalize each image’s pixel values, and label each person with their images based on the requirements from the assignment. Finally, we divide the datas for train, test and validation based on like this in below for each person with labels:

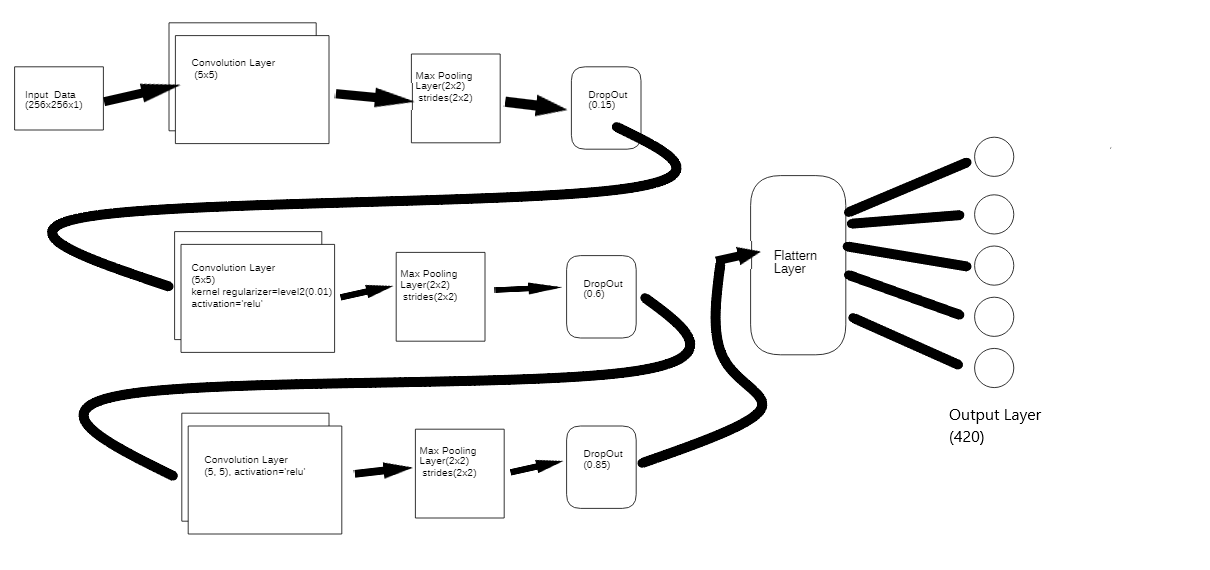
* 50% of images for train the model
* 25% of images for test the model
* 25% of images for validate the model

After all this preprocess, the data is ready for the model.

1. **CNN - ARCHITECTURE**

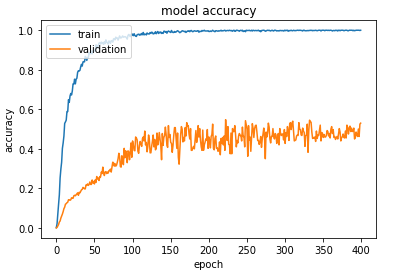
We design CNN Architecture as below:





In this project , we set the epoch values as 400, and we decided to set the number of layers as 2,4 and 8 at the convolutional layers respectively.

1. **TRAINING RESULTS**



Because our number of epochs is very high, our train accuracy goes to 99.82%, even 100% some epochs, train loss is 0.0016 and validation accuracy is reached at 53%. Therefore, the model is overfit.

1. **TESTING RESULTS**

The test result is 54.00 %, and with test loss 3.68. The result is heavily based on the number of convolutional layers and its kernel sizes.

1. **ADDITIONAL COMMENTS AND REFERENCES**

At first, we thought low test accuracy is caused by overtraining. We tried hard to eliminate this problem with decreasing the number of epochs. However, no matter what we do, we couldn’t improve the testing accuracy. We think that the number of convolution layers can change the results.