GTU COMPUTER ENGINEERING AGE PREDICTION FROM FACE USING DEEP LEARNING MODEL

CSE 495
FINAL PRESENTATION

Süleyman Gölbol

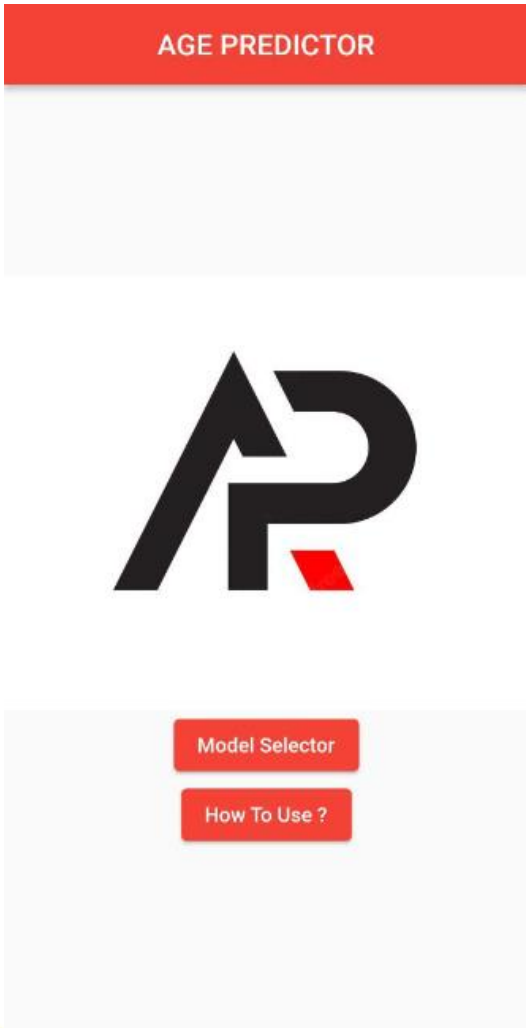
Project Supervisor: Assistant Prof. Dr. Burcu YILMAZ

January 2023

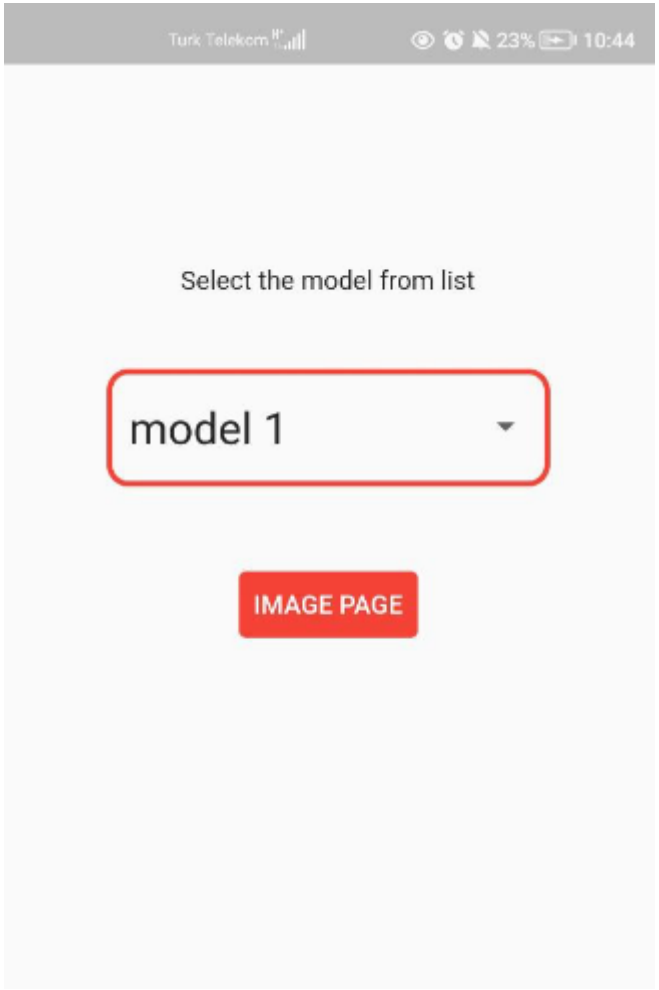
Contents

- Mobile App Schema
- Border Detection
- Azure Connection
- Github Connection
- Dataset Details
- Model Details
- Model Results
- Success Criteria
- Timeline
- References

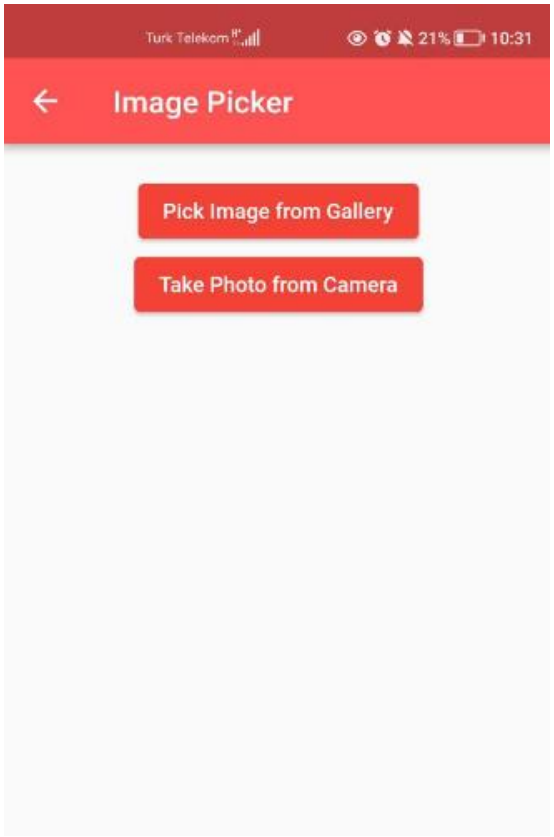
Mobile App Schema

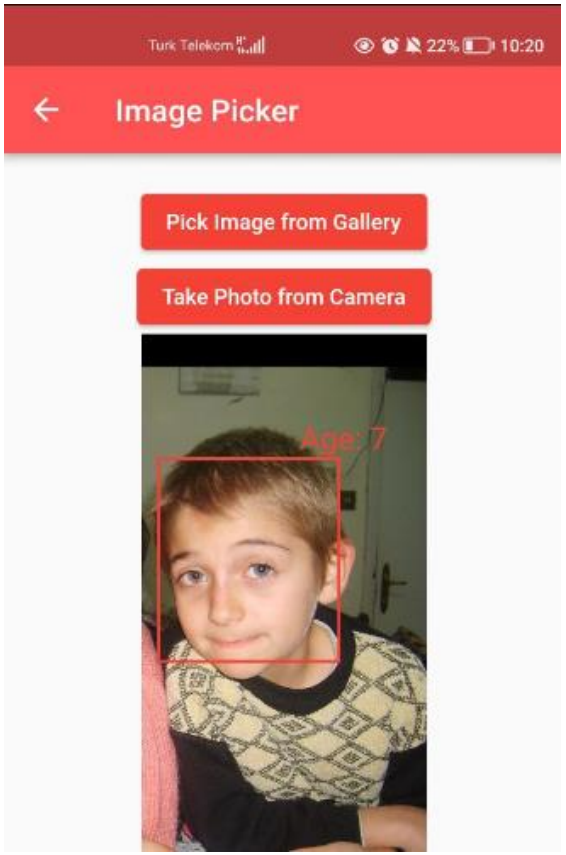


After first meeting I've started to create a mobile application for the project using Flutter.



Mobile App Schema

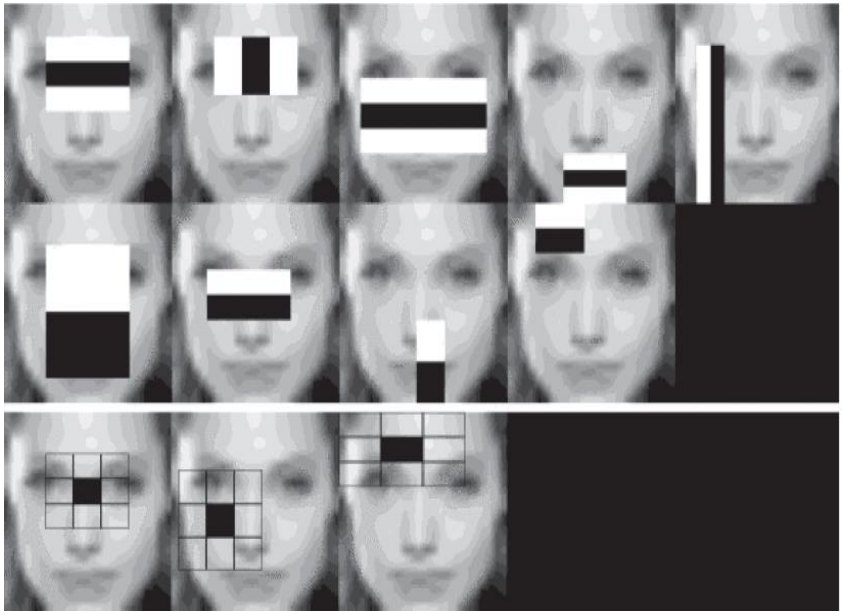




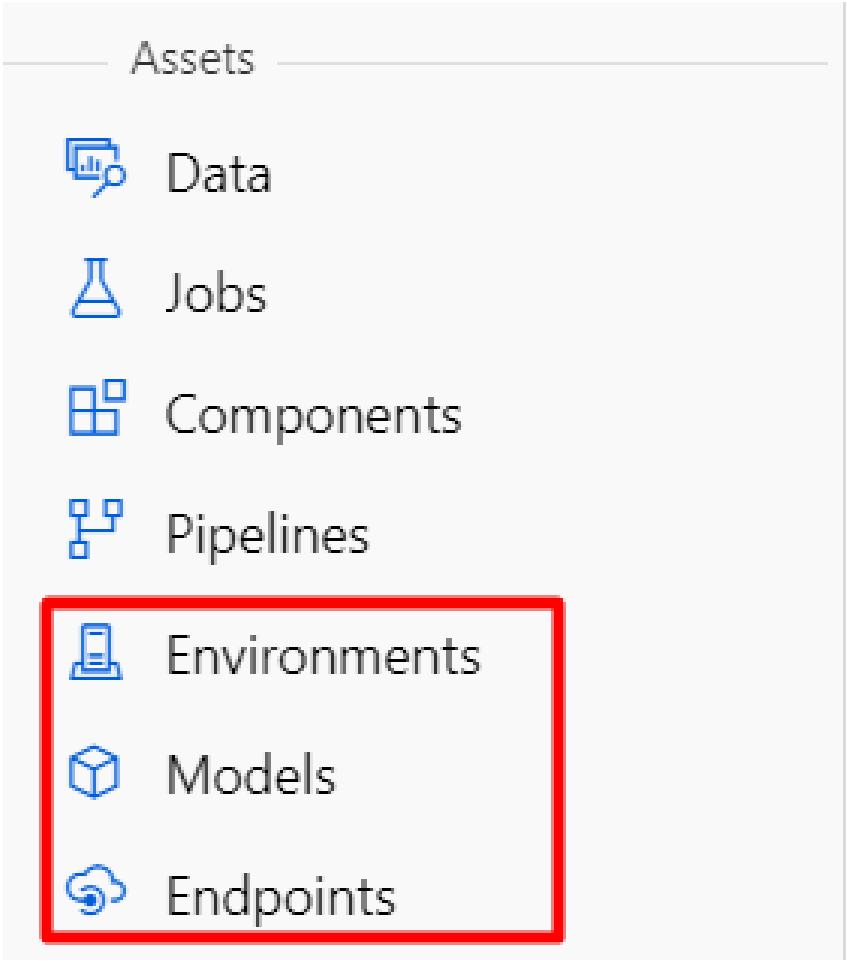
- After clicking on Image Page button, it opens Image Picker page, and it allows us to take a photo or select image from gallery. (Just preview)

Face Border Detection

- To get the face border coordinates I used Haar cascade frontal face xml. It helps to detect face borders to draw rectangle on it.
- Haar cascade uses the cascading window, and it tries to compute features in every window and scoring them to classify whether it could be a face.



Azure

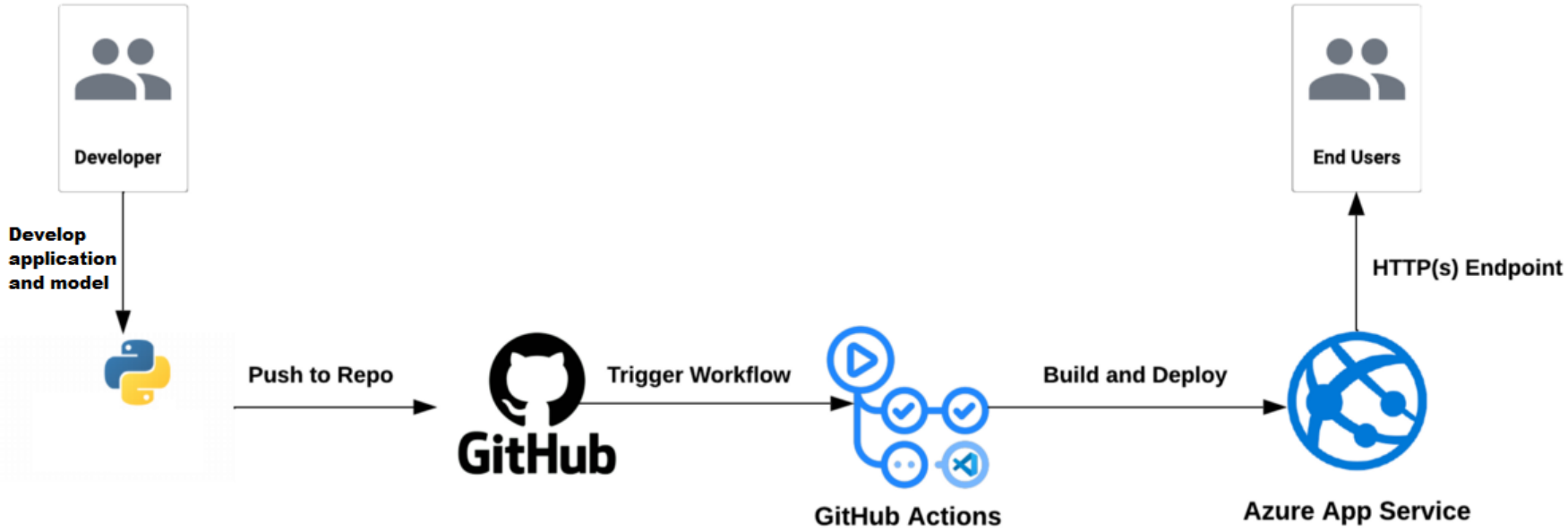


To get the age results I needed to connect the mobile app to service storing model.

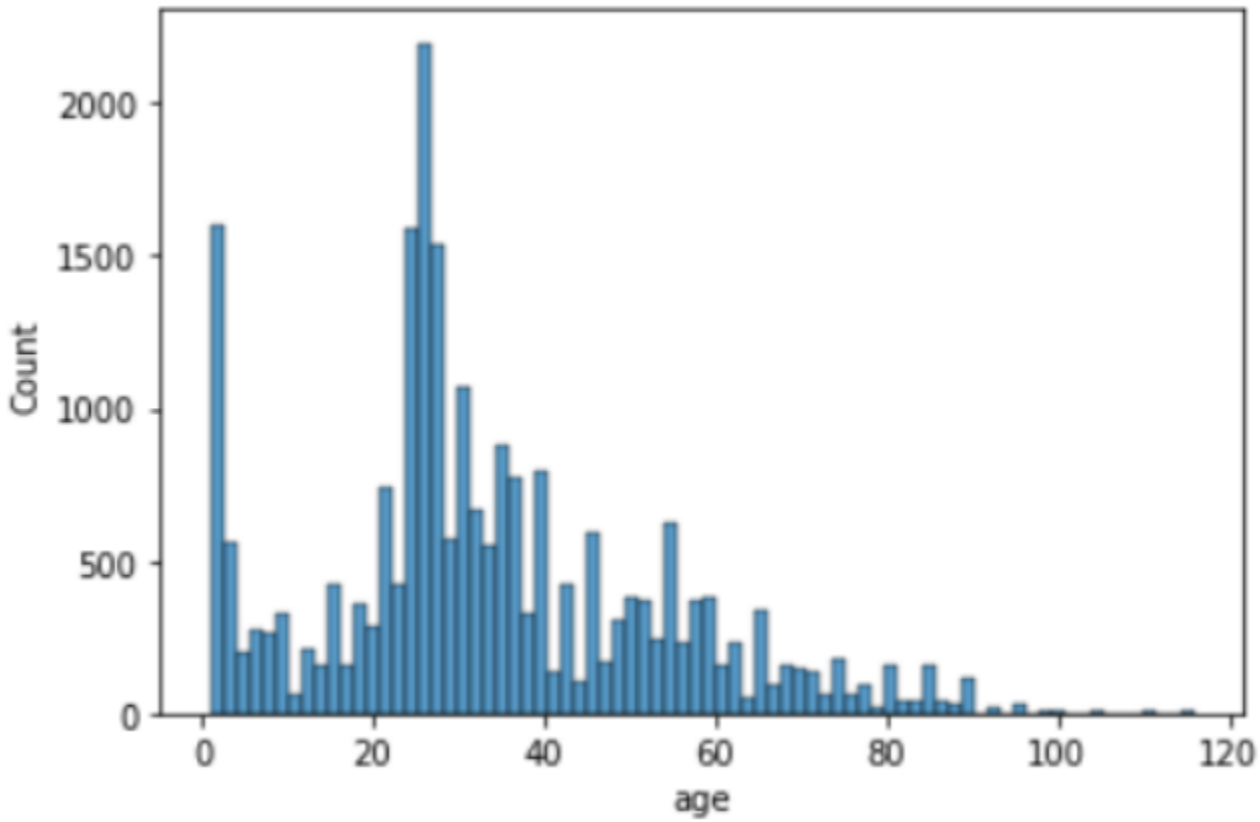
To deploy the model, I am using Microsoft Azure Machine Learning Web Function App and Github Actions. (Using Yaml and Python languages)

After saving the model, I upload the model to the Github Actions section. Then I create a docker image for environment and connect with script.

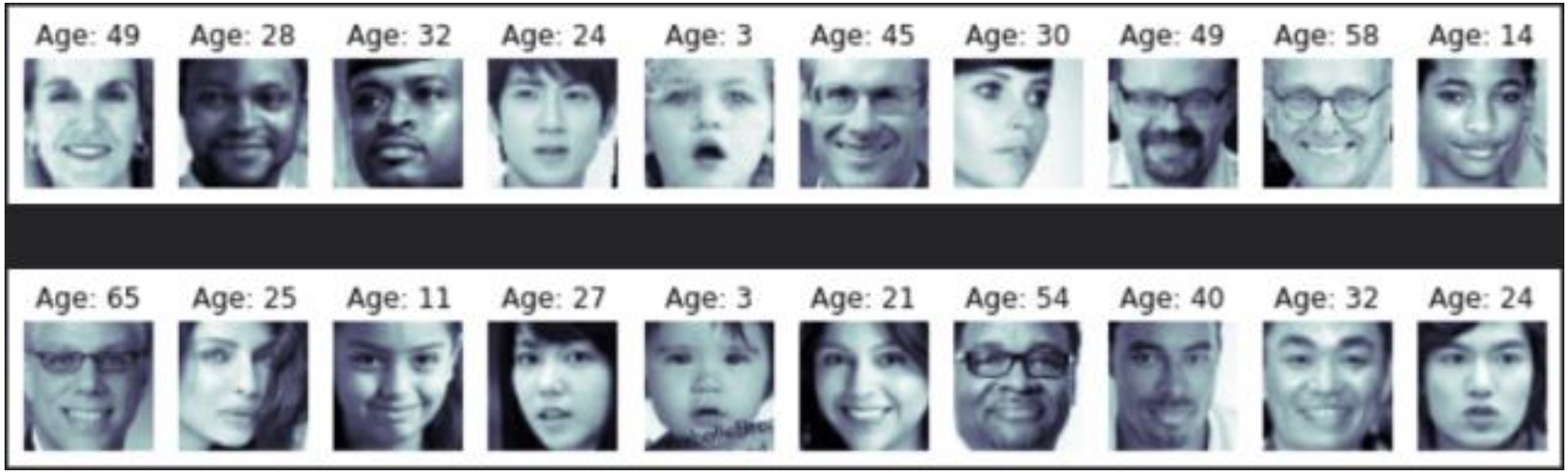
Connection Schema



Dataset Age Distribution



Some Images From Dataset



Model Details

- For preprocessing, I converted images to pixel values. ☑
- Reshaping every image to become (48,48,1) size. ☑
- Cleansing datasets if it contains unnecessary data. ☑
- Splitting datasets into training and test data. ☑
- Finding suitable algorithms that detects the face features. ☑
- Creating/updating model using different layers and activation functions. ☑
- Training the faces with multiage pictures of people. ☑
- Sending model over the internet to camera device for prediction. ☑
- Detecting face and using photo as input in the model. ☑
- Detecting the age of person using a mobile application for testing. ☑

Model Details

```
model = keras.Sequential()
model.add(Conv2D(64, kernel_size=(3,3),
          input_shape, activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(BatchNormalization())
```

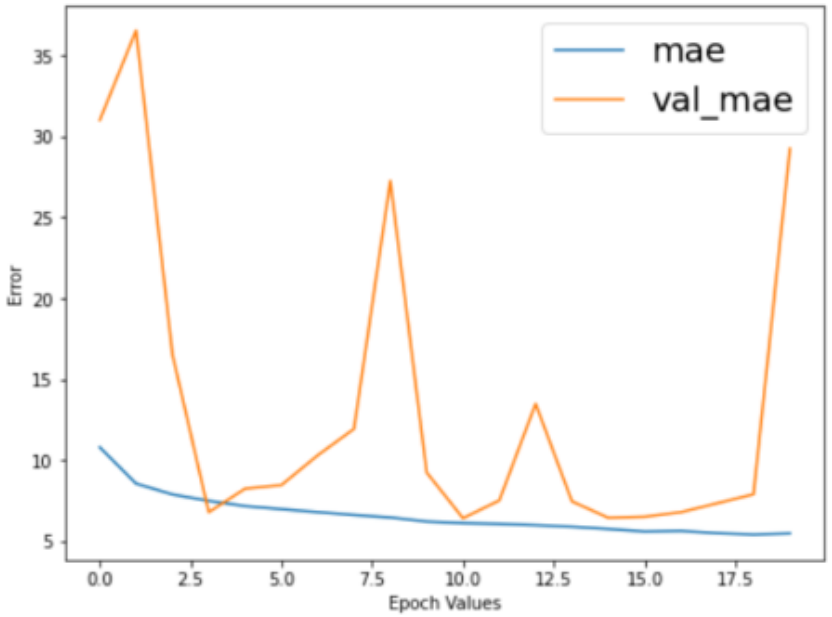
- First, I added a 2D convolutional layer with 64 filters, a kernel size of (3,3). The activation function used relu (rectified linear unit) is used to introduce non-linearity to the model.
- Then I added 2D max pooling layer with a pool size of (2,2). Max pooling is a technique for reducing the spatial dimensions of an input while retaining important information. It does this by only retaining the maximum value from each pooling window.
- Then I added batch normalization layer. Batch normalization is a technique used to improve the stability and performance of neural networks by normalizing the inputs of each layer. This helped to improve the optimization process and reduce computation time.

Model Details

```
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(1))
```

- At the end, I added Flatten layer so that it flattens output from the previous layers into a 1D array, so that it can be input into a fully connected (dense) layer.
- Then I added dense layer with 'relu' activation function. The dense layer is used to learn high-level features from the output of the previous layers.
- Then I added dropout layer with a rate of 0.3. Dropout is a regularization technique that randomly sets a fraction of input units to 0 at each update during training time, which helped prevent overfitting.
- In the last line, model.add(Dense(1)), adds the final layer with a single output unit. This is the output layer of the model, which produces a single output value for each input.

Model Results for 20 Epochs



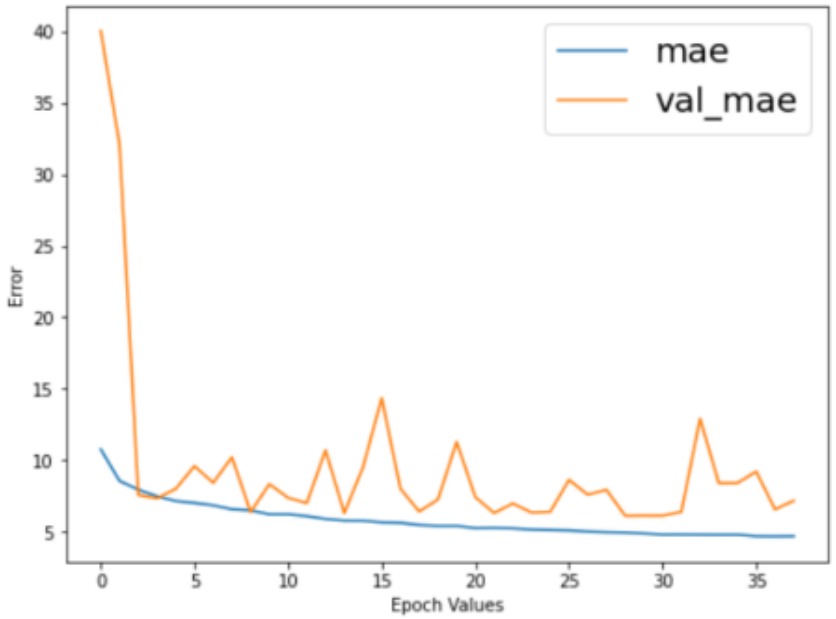


Table 5.1: Comparison of errors.

Model	Epochs	MSE VAL LOSS	MAE
My Model	20	1043.81	29.25
My Model	38	104.86	7.14

Success Criteria

1. Absolute error below 10 using model for a dataset over 15000 values.
2. Detection should be made at most 5 seconds.
3. Absolute error below 12 using different model with transfer learning.

Timeline

1st Meeting (Preliminary Presentation) – Gathering datasets, creating model. – Starting to create mobile application.	October 26, 2022, Wednesday
2nd Meeting – Training, model fixes on project. – Connecting application to deployed model for testing.	December 07, 2022, Wednesday
Report Submission	January 15, 2023, Sunday
Trailer Submission	January 15, 2023, Sunday
3rd Meeting (Final Presentation)	January 18, 2023, Wednesday
Demo	January 19, 2023, Thursday

References

1. Sumit Mund, Microsoft Azure Machine Learning, 2015
2. Sidra Mehtab, Jaydip Sen, Face Detection Using OpenCV and Haar Cascades Classifiers, March 2020
3. P. Valenzuela-Toledo and A. Bergel, "Evolution of GitHub Action Workflows," 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 2022, pp. 123-127
4. Jeff Heaton, “Applications of Deep Neural Networks with Keras”, September 2020
5. Kinsman, Timothy and Wessel, Mairieli and Gerosa, Marco A. and Treude, Christoph , How Do Software Developers Use GitHub Actions to Automate Their Workflows?, arXiv, 2021