Final Round Written Report

by Machine Forgetting (Patras)

About this document

# Scope and purpose

In this document we will document how we worked on the topic and the results we got as a team during the Final Round of the EESTech Challenge 2022.

# Intended audience

This document is intended for the evaluators of the teams that participated in the EESTech Challenge.

Table of contents

[About this document 1](#_Toc104483419)

[Table of contents 1](#_Toc104483420)

[1 Data Collection 2](#_Toc104483421)

[2 Preprocessing 3](#_Toc104483422)

[3 Approach and Training 4](#_Toc104483423)

[4 Results 5](#_Toc104483424)

[4.1 Confusion Matrix 5](#_Toc104483425)

[4.2 Accuracy and Validation Accuracy during training 5](#_Toc104483426)

[5 Algorithm complexity 6](#_Toc104483427)

[5.1 Summary of our model 6](#_Toc104483428)

[Revision history 7](#_Toc104483429)

# Data Collection

For data collection, our main concern was to avoid ending up with the model that could perform only under specific circumstances. So, to get a generalized solution , we collected data from many different locations, heights and distances. Its important to note that we collected the same amount of data for all the possible outcomes, so we could avoid any.

# Preprocessing

For the preprocessing we followed a very conservative and safe approach in order to avoid distorting the data. We normalized our data using the Min Max method. We preferred this normalization method over others mainly because it helps to polarize the high and low extreme values even more. So our model for example, will be able to distinguish between normal values and outliers much easier. Thus , a room full people (3) and an empty one should be easier to classify. Also, by scaling our data between [0,1] we can achieve faster convergence. Finally we get a lighter solution which fits better to embedded devices.

# Approach and Training

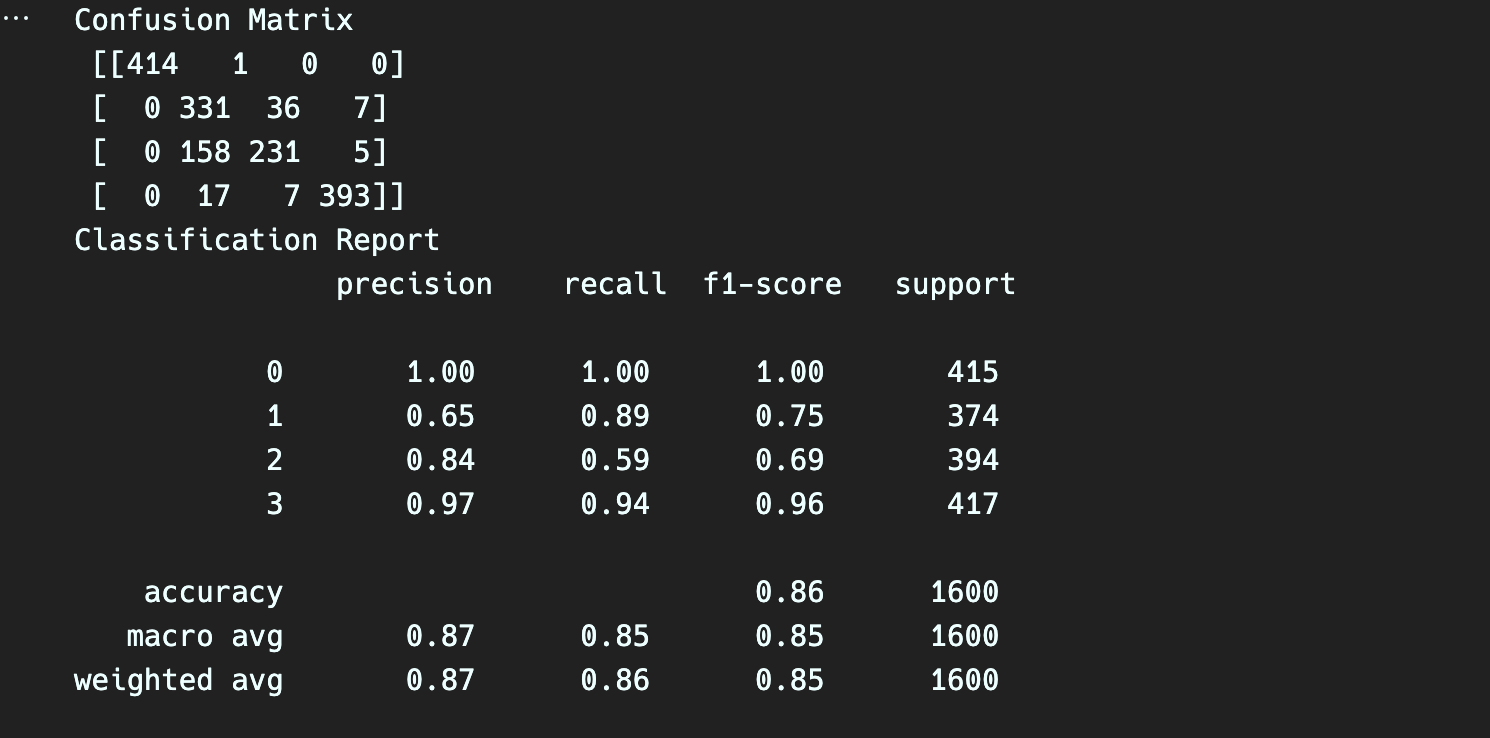
To capture the spatial information of the data we decided to use a classic convolutional network approach. Specifically we do this by applying a filter to an input to create a feature map that summarizes the presence of detected featrures in the input. Also by stacking multiple convolutional layers with increasing size as we go deeper we are able to extract more complex features that are a combination of the features that was extracted in the layer before.

We also used leaky ReLU for the activation in order to avoid vanishing gradients as the learning progresses. Dead neurons could lead to significant information loss. Leaky ReLU does not allow weights to drop to zero.

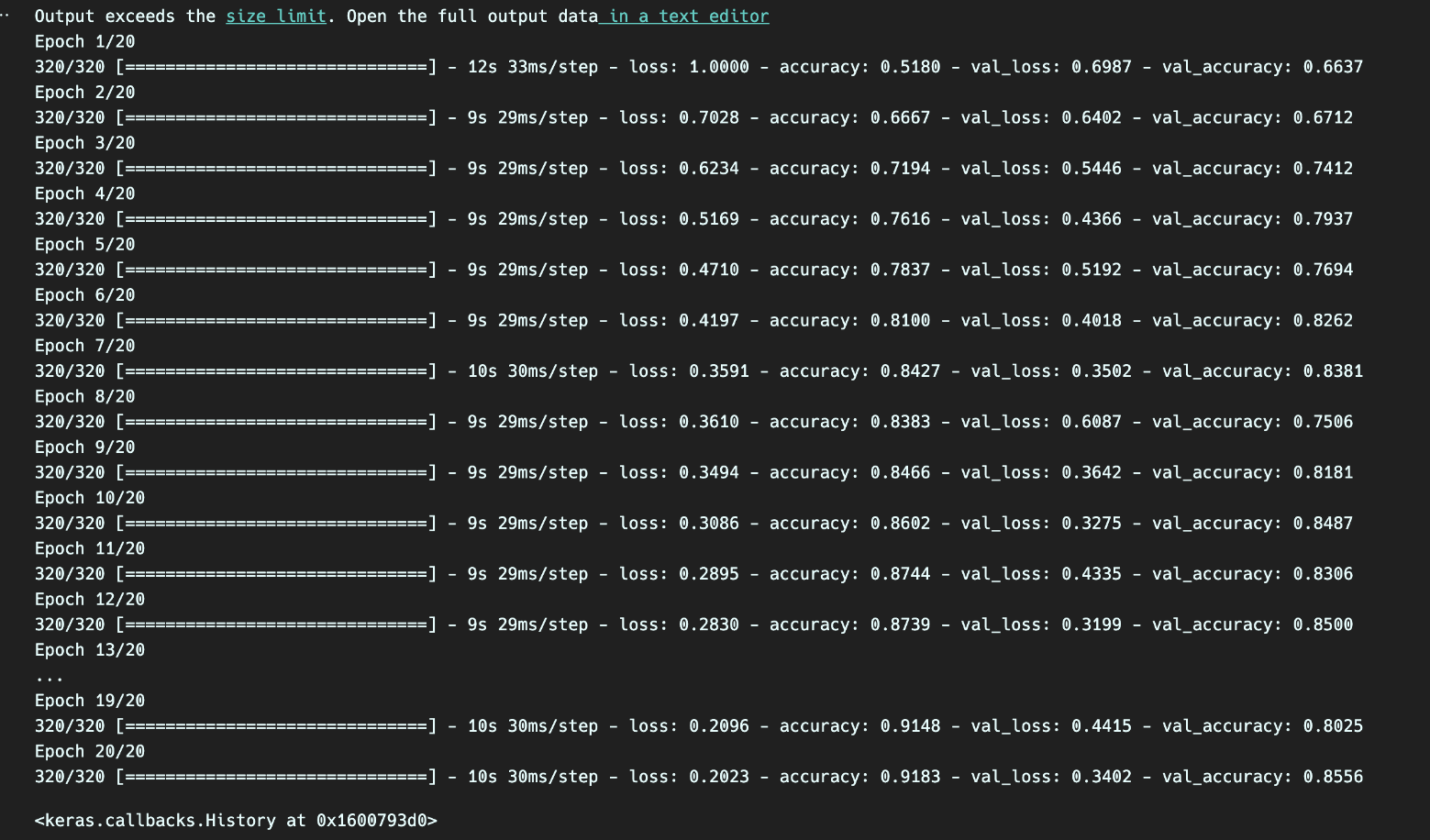
For the training part we splitted our data with a random way and batched them with a size of 10. Small batches like these stabilize the learning process and overall help the model make the most out of the data. The trade off is that the model becomes significantly slower to train. We evaluated the model using data that was never seen on the training process.

# Results

## Confusion Matrix



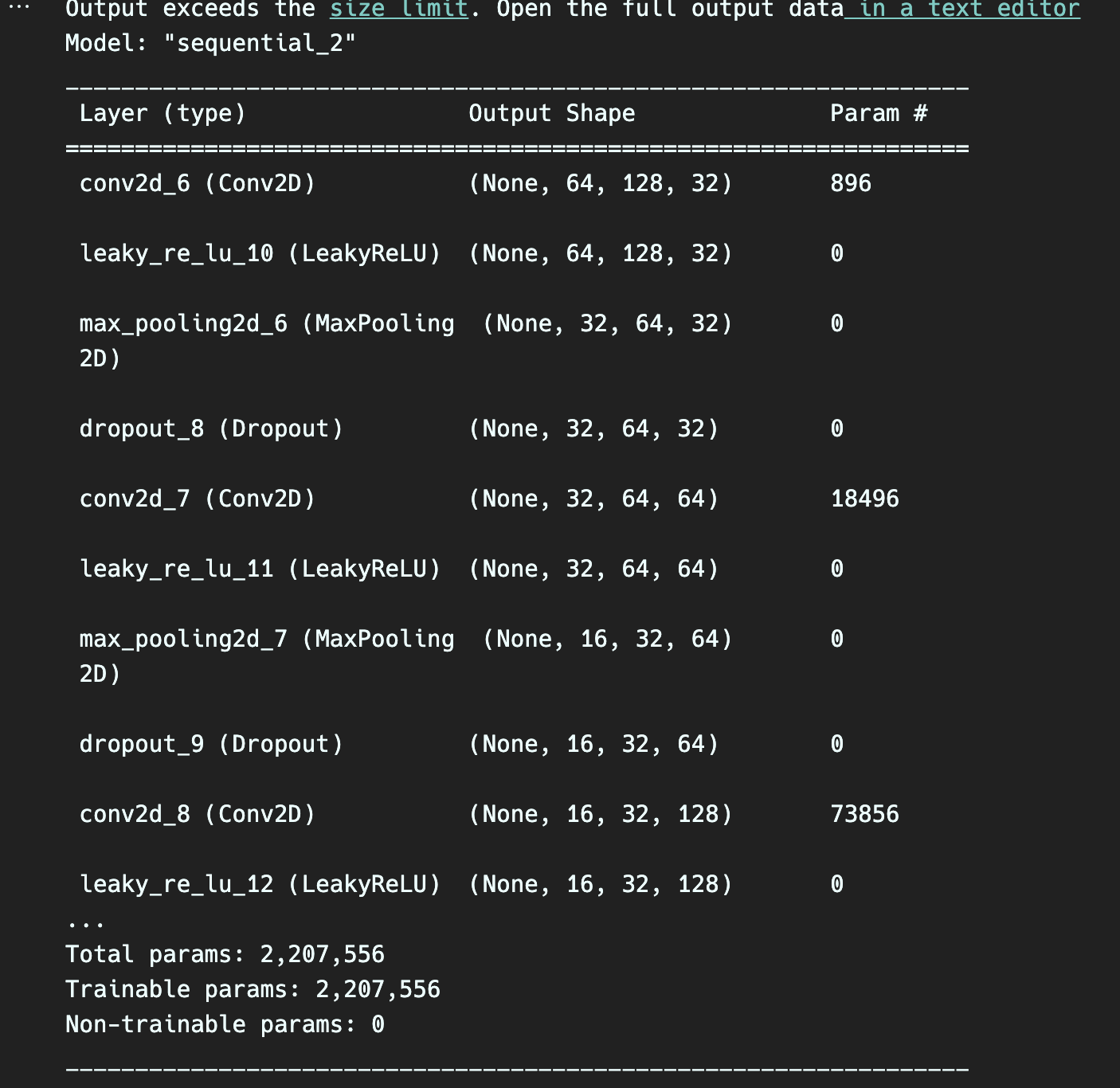
## Accuracy and Validation Accuracy during training



# Algorithm complexity

Time complexity With 2.2 million parameters the model takes around 10 minutes to train.

## Summary of our model



Revision history

| Document version | Date of release | Description of changes |
| --- | --- | --- |
| 1.0 | 26/5/2022 | Initial commit |
|  |  |  |
|  |  |  |