MPA-MLF – Final Project

Classification of room occupancy

Evann DURAIN – Kilian GEOFFROY

Introduction :

In this rapport we will be discussing about the algorithm we created to classify the number of persons in a room from 60 GHz signal transmission.

The dataset used consist of snapshots of signals in the delay-Doppler domain that represent the reflections from the targets (humans and machines) at some distance from the receiver moving at some velocity. The dataset is created in figures in .png format, where each file represents one two-dimensional snapshot in the delay-Doppler domain.

The goal of this project is to classify the number of persons present in the room from a given snapshot. Therefore we will have 4 categories.

1) Machine only, zero persons in the room
2) One person in the room
3) Two persons in the room
4) Three persons in the room

Here are the files we will be working on :

1) x_train.zip - training dataset
2) y_train.csv - ground truth values for the training data
3) x_test.zip - data for testing
4) submission_example.csv – example of data format that is accepted by Kaggle

Different steps :

Files unity :

The first thing we want to address in the creation of our model is the file unity. To ensure that, the images will be preprocessed (ex : resize, normalize, denoise).

For that we will be making sure that every snapshot is a fixed size (128 per 128 pixels) by resizing every image.

Architecture :

Since this project involves image classification we thought that using CNN (Convolutional Neural Network) would be more appropriate for different reasons :

CNN is very effective in capturing spatial patterns present in images. It also uses filter (or kernels) to detect local patterns. They will be able to identify the delay-Doppler effect wherever it can be on the image and how many persons will be in the images.

One last thing is CNN's ability to automatically learn relevant features directly from the data.

All of those reasons make CNN perfect for the task we will be attributing it.

Datasets :

Unfortunately, we weren't able to use the complete test dataset because our computers weren't powerful enough to go through the testing without being hours or days long.

Because of that we split the x_train.zip and y_train.csv datasets into a training dataset and a validation dataset. The model will be trained in the first one and learn to classify the images. After that it will be tested on the validation dataset.

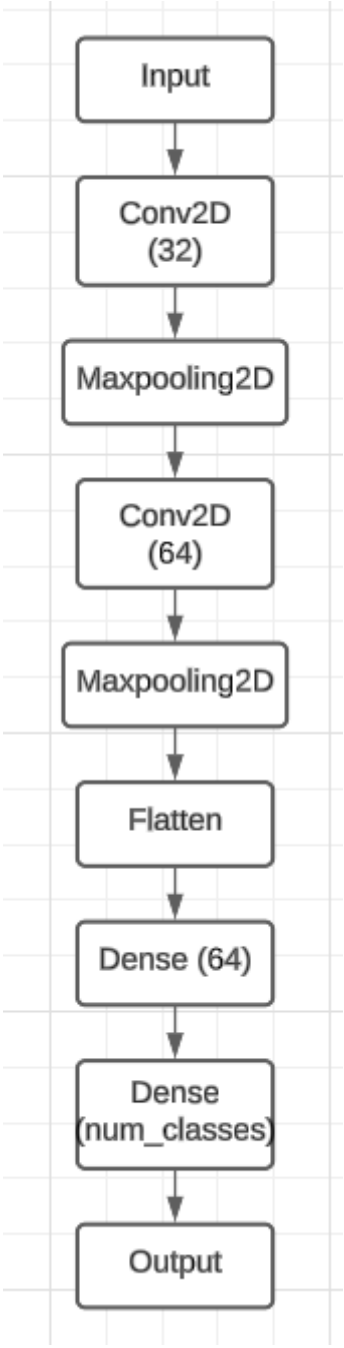
We are also going to label our data in 4 classes, one for every possible snapshot.

Model :

Durin our tests we found that using 20 epochs for our model was the best. We didn't want to run into an overfit problem by using too much epochs.

We don't have the different testing scores because we didn't use Kaggle in time.

Flow chart of the model :



What could we improve ?

We could always try to improve the accuracy, precision, recall and F1 score of our model.

What we could also do in the future is to fine tune depending on the results of each iterations of test. Then test the model on unknown data to see if it is working.

To keep the model updated in the long run we would need to run iterations of those steps to ensure that the model's performance stays at its best.