Assignment 2 Report

1 Introduction

The training dataset and validation dataset are given to us, which we can use to train our model. Finally, on the day of submission, we are given the testing dataset on which we must run our model and predict the results. The training dataset consists of 10000 data samples. Each data sample consists of 1000 instances, and each instance is an array of 64 numbers. So the shape of the training dataset is (10000x1000x64). The validation dataset consists of 2000 data samples. Each data sample consists of 1000 instances, and each instance contains an array of 11 numbers whose number gives us information on whether that sound is present at that given instant. The testing dataset consists of 2500 data samples. If no sound is present, the number in the 8th place, representing silence, becomes 1. The shape of the validation dataset is (2000x1000x11). The different sounds in this dataset are Alarm bell ringing, Blender, Cat, Dishes, Dog, Electric_shaver_toothbrush, Frying, Running water, Silence, Speech, and Vaccum_cleaner.

My assignment has three different code files and a neural network model which I trained. This neural network model is in 2 formats hdf5 and sav. I used the hdf5 format to get the results in the Juypter notebook. The code files contain the codes to get the predicted results for the validation dataset, testing dataset, and the other methods I used to compare the performance of the neural network model.

2 Literature Survey

Machine Learning models are trained to learn and predict the results in the testing dataset. We can train our model with different approaches like CNN, Decision tree Classifier, KNN, and Naïve Bayes, ... The hyperparameters are tuned to reduce overfitting as well as to increase the mean f1 score of our model. But as the training dataset size is vast, we must be careful about which methods to train our model. We are given a code block in python which converts the event roll to a multi-hot vector. Using this code block, we can convert a (1000x11) sized validation and predicted samples to a (10) sized array.

3 Methods

The training and validation dataset are given to us separately so we can use the training dataset and validation dataset to find our model's accuracy and mean f1 score. I used google colab and google drive for training the models used in this assignment. For testing the data, I used the Juypter notebook because it is much faster to get the results than on Colab.

After preprocessing the dataset, I used mean f1 score as a metric to determine which model to use. Here preprocessing means changing the dataset to the required format to train and test our model. The mean f1 score is preferred over accuracy because this is a multi-class classification problem. The Neural Network model gave the best mean f1 score, 89.12%.

For the other methods that I tried, the mean f1 score values are as follows:

- Decision tree classifier: 34.91%
- KNN: 32.63%
- Naïve Bayes: 34.23%
- Neural Network Model: 89.12%

From this, we can see that we get the best mean f1 score when using Neural Network Model.

I trained the neural network model in Google colab, and its summary is as follows:

Summary of the built model...
Model: "sequential_8"

Layer (type)	Output Shape	Param #
dense_24 (Dense)	(None, 1000, 64)	4160
activation_24 (Activation)	(None, 1000, 64)	0
dense_25 (Dense)	(None, 1000, 32)	2080
activation_25 (Activation)	(None, 1000, 32)	0
dense_26 (Dense)	(None, 1000, 10)	330
activation_26 (Activation)	(None, 1000, 10)	0

Total params: 6,570
Trainable params: 6,570
Non-trainable params: 0

None

I trained this model for 200 epochs and passed the results to the function eventroll_to_multihot_vector2. I used the validation dataset to find the mean f1 score of this model.

4 My Results

For the Neural network model, the various evaluation metrics on the validation dataset are as follows:

Accuracy: 85% (average accuracy without considering silence)

The accuracy for different sounds excluding silence is: [90, 90, 80, 90, 90, 70, 80, 90, 90, 80]

F1 score(mean): 89.12%

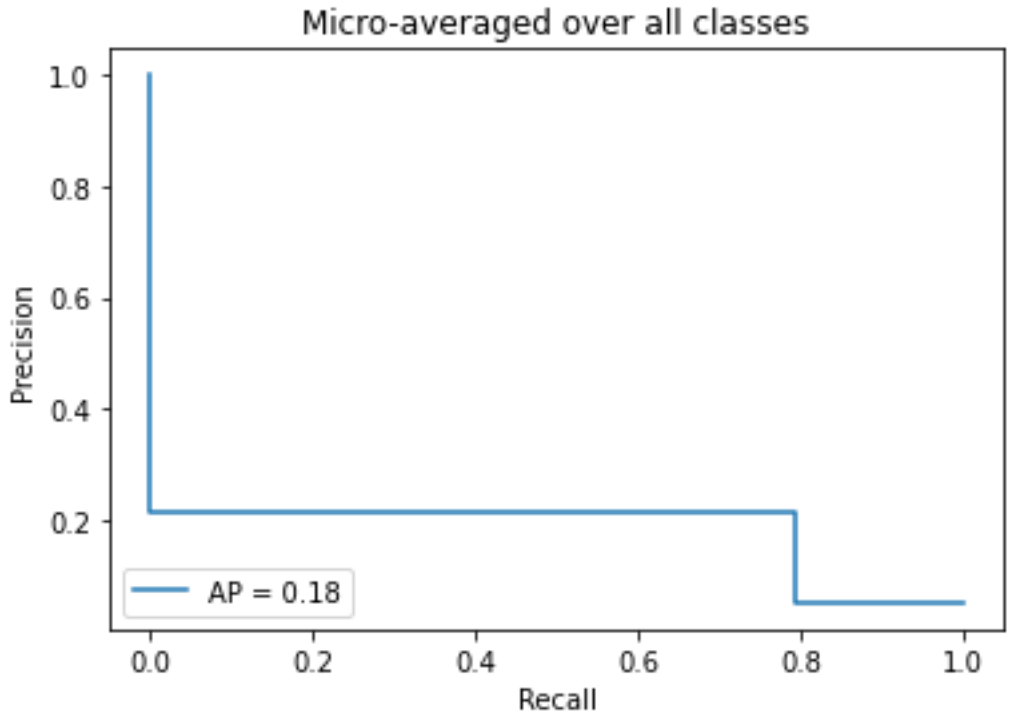
F1 score for individual sounds excluding silence is

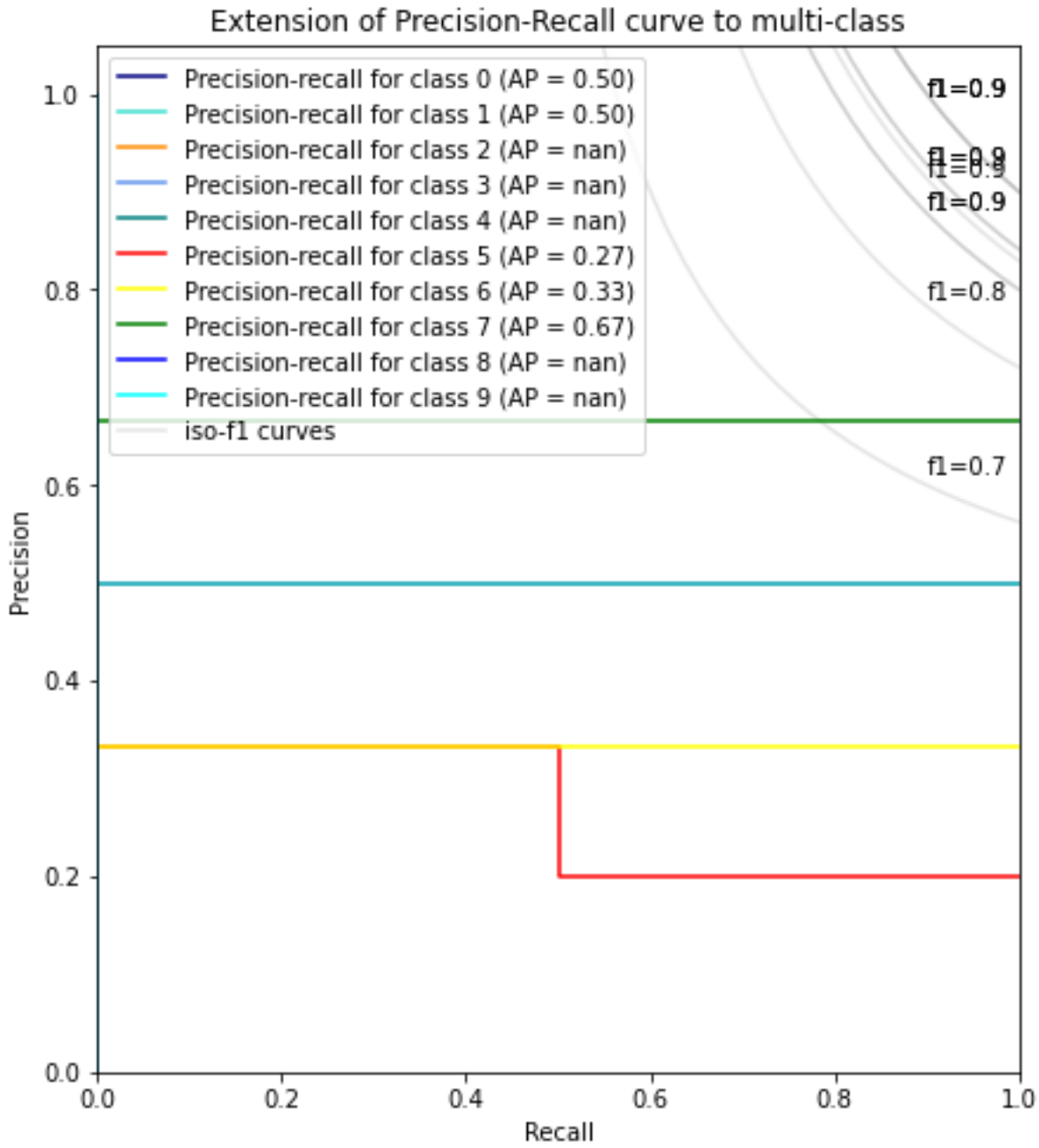
```
[0.9137254901960784, 0.9137254901960784, 0.888888888888889,
0.9473684210526316, 0.9473684210526316, 0.72, 0.8375,
0.9066666666666666, 0.9473684210526316, 0.7411764705882353]
```

```
Confusion Matrix: [[[1608,  392],
         [   0,    0]],

        [[1506,  432],
         [  58,    4]],

        [[1726,  271],
         [   0,    3]],
```

```
[[1556,  444],
 [   0,    0]],

[[1686,  264],
 [   8,   42]],

[[1775,  221],
 [   4,    0]],

[[1870,  130],
 [   0,    0]],

[[1857,  143],
 [   0,    0]],

[[ 615,  486],
 [ 139,  760]],

[[1849,  151],
 [   0,    0]]]
```

Precision vs. Recall Curves:





For the Neural network model, the various evaluation metrics on the testing dataset are as follows:

Accuracy: 85.00% (average accuracy without considering silence)

The accuracy for different sounds excluding silence is: [90, 80, 90, 90, 90, 70, 80, 90, 80, 90]

F1 score(mean): 87.17%

F1 score for individual sounds excluding silence is

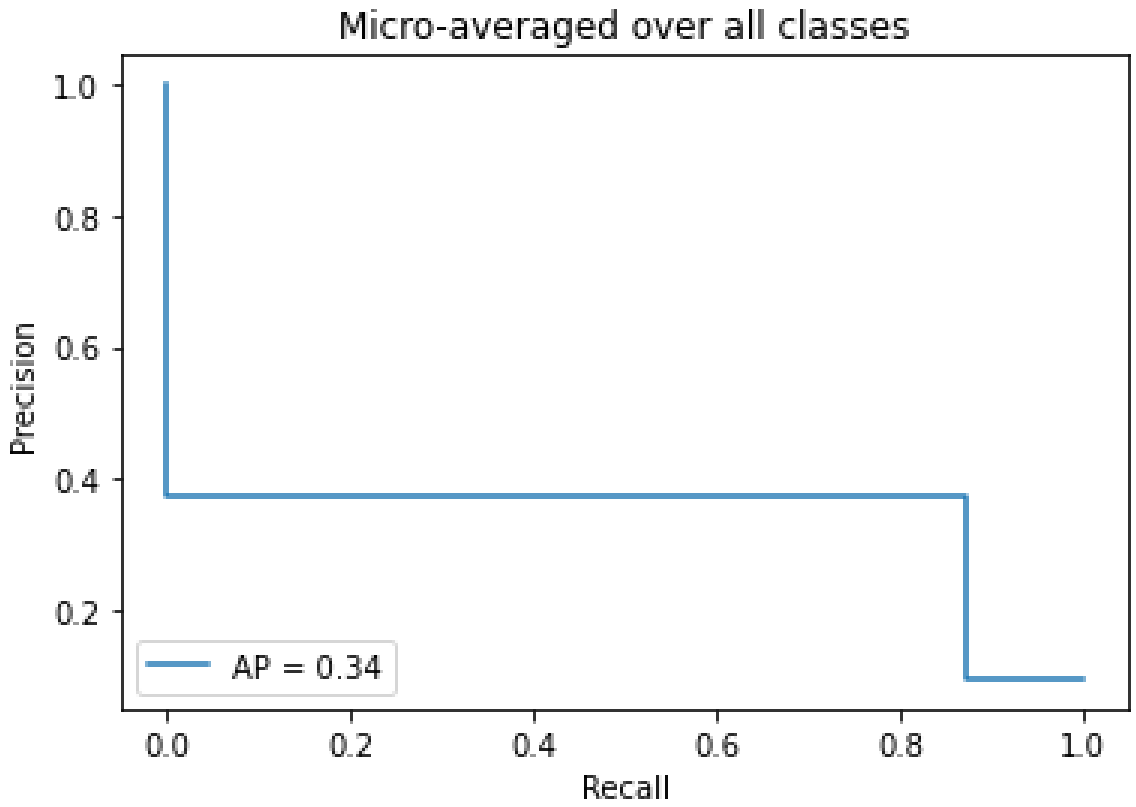
```
[0.9137254901960784,
 0.8375,
 0.9137254901960784,
 0.9137254901960784,
 0.9137254901960784,
 0.76,
```

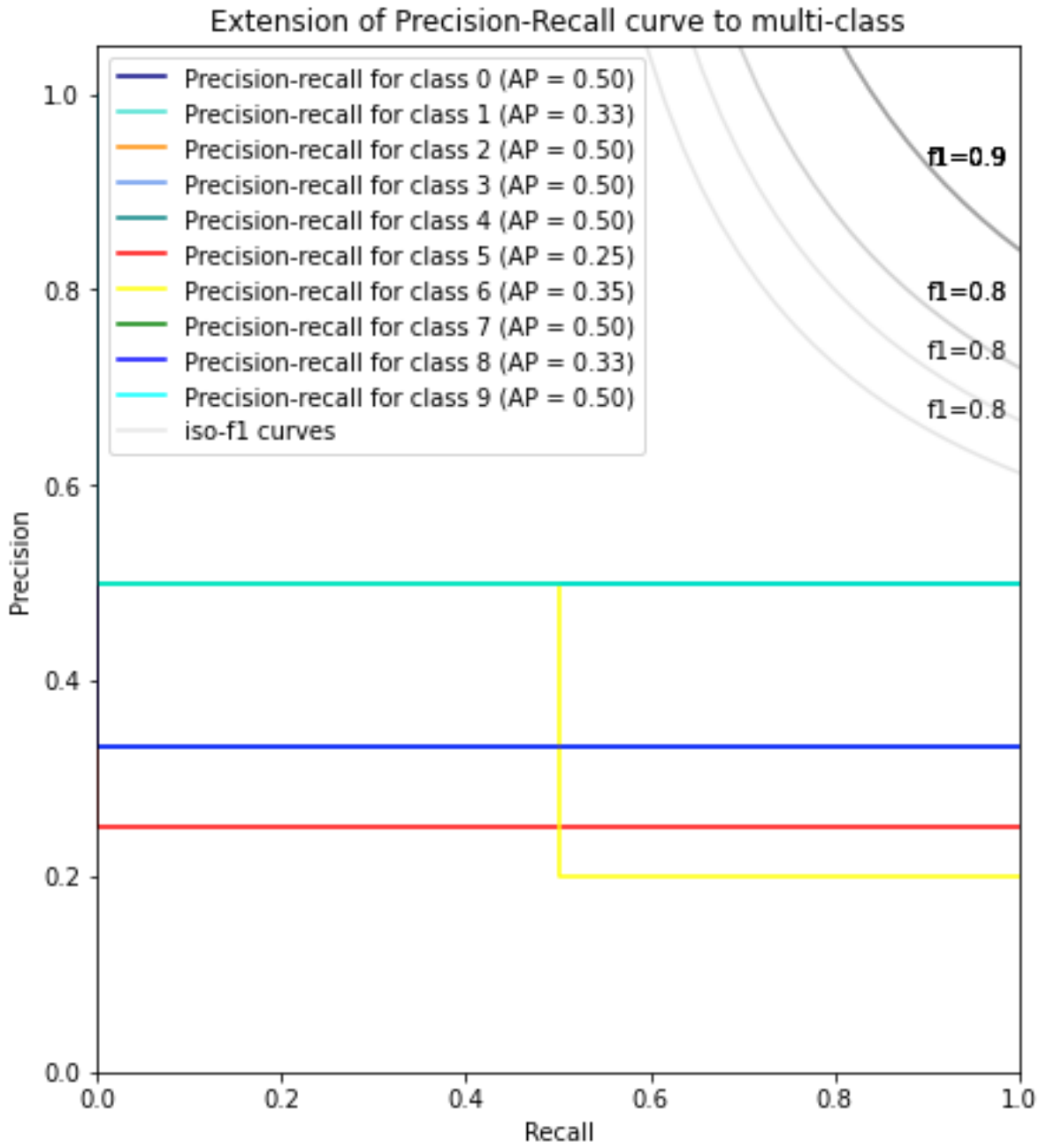
```
 0.8,
 0.9137254901960784,
 0.8375,
 0.9137254901960784]
```

Confusion matrix:

```
[[[2100,  400],
        [   0,    0]],

       [[2044,  257],
        [ 190,    9]],

       [[2194,  277],
        [  22,    7]],

       [[1811,  689],
        [   0,    0]],

       [[2131,  317],
        [  28,   24]],

       [[2211,  280],
        [   6,    3]],

       [[2120,  376],
        [   3,    1]],

       [[2192,  306],
        [   2,    0]],

       [[  74,  337],
        [  53, 2036]],

       [[2249,  251],
        [   0,    0]]]
```

Precision vs recall curves:


Micro-averaged over all classes
Precision
Recall
AP = 0.34
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0



5 Observations and Discussion

Different methods give us different values of mean f1 score when trained and validated on the given dataset. We must select the model that would predict good results in the testing stage. For example, take the Decision tree classifier. We can increase its accuracy by increasing the maximum depth of the decision tree, which consecutively grows the dependency of the model on the training dataset and overfits the data such that we get poor results while predicting the test dataset. Hence, we have to choose the optimal value of depth which is a trade-off between accuracy and overfitting.

In the Neural Network model, we have to choose the optimal value of hidden layers and the number of parameters each hidden layer contains to get more mean f1 score. I used SOFTMAX activation for the last layer because we have to model for multi-class classification. In between the layers, I used ReLU activation.

My model has an excellent f1 score for each class, indicating that the model has high precision and recall.