## [F22] MACHINE LEARNING ASSIGNMENT 2

**Due Date:** 16th November 2022, (15:00)
**Submission :** Jupyter notebooks to moodle (**2 notebooks + TensorBoard files in .zip file**).
**Data (Task 1)**: Train, Test

---

### TASK 1

In connection to assignment 1 task for stream quality, where the performance of simple linear models was performed poor and could not learn the pattern of the data. It is important to apply more non-linear models and understand which independent variable affected the stream quality. Using techniques from ensemble learning and deep learning you will predict stream quality more accurately.

#### Task 1.1 (25 points)

As first step, in this task you are required to implement a Deep learning model for predicting stream quality using cloud gaming stream metrics. Secondly, you are required to use ensemble learning with neural networks (it can also be stacking). The main requirement is to achieve accuracy and F1 score more than 90%. Compare the performance of the simple neural network and ensemble of neural networks using appropriate metrics. Training-validation split : Training (80%) and validation (20%).

#### Task 1.2 (25 points)

In previous task you used a neural network model to predict the stream quality, however is quite difficult to explain the reason why the model assigned a certain label to a stream session. Therefore correctly predicting the stream quality is not enough in cases where improving the streaming service is the ultimate goal is also diagnosing the real cause of a low poor stream quality (which can lead to low customer satisfaction). The task is to select, train and evaluate an appropriate ML model that will provide an ability to understand why a specific prediction was achieved. The performance of the model should be more than 85% accuracy and F1 score.

### TASK 2

Detection and recognition of signs is an automobile equipment that reads and interprets permanent and temporary signs located at the edge or over the road, in order to inform the driver in case he could not see them. This technology is now applicable also for self driving cars.

### Task 2.1 (25 points)

In this task you will implement a traffic sign classification convolutional neural network which will be trained on German Traffic Sign Recognition Benchmark dataset. The CNN architecture shown in Table 1 is a baseline model. To accomplish this task you will need to improve the model performance using minimum of three techniques presented in the course (i.e Dropout, Batch Normalization, etc.) except for transfer learning. The base line model is to be trained on 20 epochs, SGD optimizer, learning rate 0.001 and preprocessing of only min-max scaling. The mode performance should at each training epoch should be logged to Tensorboard. The performance metrics to be monitored are training and validation loss, accuracy and F1 score. Dataset can be downloaded from **torchvision.datasets.GTSRB**. Data splits : Training (70%) and validation (10%) and Testing (20%).

| Layer | Layer Type | Kernel size | Stride | Output size |
|---|---|---|---|---|
| 0 | Input | | | 32 x 32 x 3 |
| 1 | Convolutional | 3 x 3 | 1 | 32 x 32 x 32 |
| 2 | Convolutional | 3 x 3 | 1 | 32 x 32 x 32 |
| 3 | Max-pooling | 2 × 2 | 2 | 16 x 16 x 32 |
| 5 | Convolutional | 3 x 3 | 1 | 16 x 16 x 64 |
| 6 | Convolutional | 3 x 3 | 1 | 16 x 16 x 64 |
| 7 | Max-pooling | 2 × 2 | 2 | 8 x 8 x 64 |
| 8 | Convolutional | 3 x 3 | 1 | 8 x 8 x 128 |
| 9 | Convolutional | 3 x 3 | 1 | 8 x 8 x 128 |
| 10 | Max-pooling | 2 × 2 | 2 | 4 x 4 x 128 |
| 11 | Flatten | | | 2048 |
| 12 | Fully connected | | | 512 |
| 13 | Fully connected (softmax) | | | 43 |

Table 1. The details of the baseline CNN model

### Task 2.2 (25 points)

For this task you will need to use transfer learning. You will have to achieve better performance than baseline model in task 2.1 and reduce the number of parameters without loosing accuracy of more than 5 percent. The model training progress should also be monitored using using tensorboard. The performance metrics to be monitored are training and validation loss, accuracy and F1 score.

### GRADING

Each of the task mentioned above will be graded as based on the following :

(1) Exploratory Data Analysis, Feature engineering & preprocessing - 20%;
(2) Machine learning or DNN model definition, training and hyper-parameters turning - 35%;
(3) Comparison of ML models using appropriate evaluation metrics - 30%;
(4) Reporting the goal, approach for training model and the attained ML model results in Jupyter notebook (use markdown) - 15%

## SOURCE CODE (JUPYTER NOTEBOOK FILE)

The implementation should be in python. All deep learning tasks should be implemented using PyTorch framework and Tensorboard should be used to monitor the model performance. All the implementation should be presented in Jupyter notebook files. One jupyter notebook per task. The jupyter notebook file should contain have the following main sections : (i) Data Reading, Exploration and preprocessing; (ii) Machine learning or Deep learning model defining, training and hyper-parameters turning; (iii) Model performance evaluation; (iv) Conclusion and possible improvements;