**HW 4 – Multi-label Classification Using Trainer (10 Points)**

**Required Submission files:**

***Submit the following two files:***

1. **FirstName\_LastName\_HW4.ipynb**
2. **FirstName\_LastName\_HW4.pdf (pdf version of the above file)**

**TASK: Multilabel Classification (10 Points)**

In this HW, you will identify tags for stack exchange Questions. This data is a subset of data available in [Kaggle](https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data) Competitions. The given dataset contains the different questions asked in the StackExchange website for various tech domains. We have fetched only those question that contains top 10 individual tags. Each question can have more than one tag. **This means that this is a multi-label classification problem.** These are the ten categories for tags in the data.

|  |  |
| --- | --- |
| 0 | c# |
| 1 | java |
| 2 | php |
| 3 | javascript |
| 4 | android |
| 5 | jquery |
| 6 | c++ |
| 7 | python |
| 8 | iphone |
| 9 | asp.net |

**Data:**

* The dataset is available from eLearning: **0\_Data folder** from the file: **df\_multilabel\_hw\_cleaned.joblib**.
* The dataset is a pandas dataframe. You can use joblib.load(file) to load the dataset.
* No preprocessing is required, you are provided with the cleaned dataset.
* You will use this data set to train your classification model**.**
* Hint1: The Tag\_Number column is your labels. However, the data type of list elements is object. To convert it to integers; use ast.literal\_eval() function from ast library. You will need to import this library first.
* Hint2: Since this is a multilabel dataset – you will need to do the one-hot encoding of your dependent variable. You can use MultiLabelBinarizer from sklearn.
* **You will create train/valid/test splits from this data set.** Use 60% for train, 20 % for the test, and the remaining 20% for the validation dataset.

**Model:**

* Your Network should have the following layers.

EmbeddingBag → Hidden\_Layer1 →ReLU → Dropout\_Layer1 → BatchNorn\_Layer1 → Hidden\_Layer2 → ReLU → DropoutLayer2 → BatchNorm\_Layer2 → Output Layer

* You will use the following hyperparameters to train our model:

    EMBED\_DIM=300,

    VOCAB\_SIZE=len(your\_vocab),

    OUTPUT\_DIM=10,

    HIDDEN\_DIM1=200,

    HIDDEN\_DIM2=100,

    # training

    EPOCHS=5,

    BATCH\_SIZE=128,

    LEARNING\_RATE=0.001,

    WEIGHT\_DECAY=0.000,

    CLIP\_TYPE = 'value',

    CLIP\_VALUE = 10,

    PATIENCE=5,

Note:  For Gradient Clipping. you will need to add the following line in the step function we discussed in class.

loss.backward()

# Clip gradients before the optimizer step

clip\_grad\_value\_(model.parameters(), clip\_value=10.0)

# Update parameters

optimizer.step()

Instead of accuracy use Hamming Loss as a metric, since this is a multilabel classification problem. Hint (see Task 4 of HW3 for reference).

You can use 3\_imdb\_complete\_example.ipynb from Lecture 3 as a starting point. You will need to make changes to the Training Function to adapt it to the multilabel Classification problem.