Bi-Weekly Report (1/25/22~2/8/22)

Accomplishments

- Finalize the project proposal.
- Retrieve all datasets from Kaggle and TensorFlow Keras and then store them in the Google Drive folder.
- Finalize the train and test split ratio for the dataset, which is 0.7:0.3. The split ratio is kept constant throughout the datasets in this project in order to keep consistency. The training and test set split ratios are the same route as what authors of BAGAN-GP used in their paper.
- Implement interactive visualization for the dataset with class distribution so that we have better understanding of what's the sample size in each class.
- Manipulate the MRI Tumor Image dataset by splitting the dataset into train and test sets with a ratio of 0.7:0.3.
- Manipulate the FER-2013 Facial Expression Image dataset by:
    - Merging the predefined train and test sets so that we can re-split the train and test set in the ratio we want.
    - Splitting the dataset into train and test sets with a ratio of 0.7:0.3.
- Save MRI Tumor Image and FER-2013 Facial Expression Image dataset in "*npy*" format for quicker read in the future.

Upcoming Goal

- Construct a Convolutional Neural Network (CNN) binary classification model for MRI Based Brain Tumor Images.
- Construct two CNN multiclass classification models for:
    - FER-2013 Facial Expression dataset
    - Fashion MNIST dataset

For all above mentioned three models, we will train it with the unbalanced set utilizing Keras and Tensorflow. During evaluation of the model, k-fold validation will be utilized so that we don't have to manually specify a ratio/split for the validation set.

Issue & Barriers

- Long execution time - when performing split and merge for the dataset, it takes a very long time to complete the execution. The same thing happened when we tried to save the FER-2013 training set in "*npy*" file format. On average, it takes 30 minutes. However, one fact could cause an impact is we have a Colab that uses a limited compute instance. On top of that, the dataset is loaded in the cloud instance's memory, which makes the execution time longer. Therefore, utilizing GPU would be ideal for both preprocessing and model evaluations. Luckily, we found an open source library RAPIDS GPU and received technical support from Ashwin Srinath, Nvidia Software Developer, (URL: https://github.com/rapidsai/cudf/issues/10187) to set up the environment on the Colab.

- When performing train and test split sampling with "*random.sample()*" method, the program returns an error by saying sample size cannot be greater than population size or a negative number. However, when we track the sample size and population, there should not be any issues. After trying with the "*random.choices()*" method mentioned in Stack Overflow, it did not work as well. Luckily, we were able to find a method from Kaggle (URL: https://www.kaggle.com/questions-and-answers/102677) and modified the code to fit our needs.