Program Assignment #1

- **Due Date: Nov. 18 (Wed) 18:00**
- **Submission: Source code & Report file**

Classify the given figures into 'triangle', 'tetragon', 'pentagon'. 'circle (including ellipse)', and 'other'. Here, no overlapping is assumed.

We provide an example code (*main.py*). You can modify *main.py* to make your own code.

1. Database

   This data contains images of five labels; triangle, tetragon, pentagon, circle (ellipse), and other. Each image is 300×300 pixels. The data is arranged into five folders corresponding to their labels. The images are labeled 0.png, 1.png, and etc.

   - The labels of samples are defined in the example code (*main.py*).
     - circle ⇔ 0 / triangle ⇔ 1 / tetragon ⇔ 2 / pentagon ⇔ 3 / other ⇔ 4
   - Hand-out data: 25 samples
     - 5 samples / each label
     - You can split this data into training set and validation set for robustness to the test (unseen) data. Because good performance in hand-out data can be the opposite in test data.
   - Test (unseen) data for TA: 25 samples
     - 5 samples / each label
     - This data is not provided.

2. Preprocess ('preprocess' function)

   Implement your own feature extraction and preprocessing algorithm using images.

   - In the example code (*main.py*), we just flatten the images.

3. Make a classifier ('classify function)

   Implement your own classification algorithm using your own features. You can use the tools up to Chapter 4 in the textbook. Rule-based algorithm is also possible. Submit the highest performance algorithm (source code) of your own coding.

   - In the example code (*main.py*), we use KNN algorithm.

Precautions

1. Source code (including *main.py*) and report file (*report.docx*) that explain your preprocessing and classifier methods. If you do not refer our example code (*main.py*), explain how to run your algorithm

   File name: StudentID.zip (or tar.gz)   ex) 20201111.zip (or tar.gz)

2. **Your source code must include 'preprocess' and 'classify' functions.**

   ✓ Submit only one of the best-performing algorithms. (It is impossible to output the best performance after testing with various algorithms in 'classify' function.)

3. Implement your algorithm using python.

   A. You can use python external packages such as *numpy, librosa, skleran, scipy, cv2, and etc.*

4. TA will evaluate final performance of your algorithms using the hand-out data and test (unseen) data.

   A. PA1 score = 0.5 * (Accuracy for hand-out data) + 0.5 * (Accuracy for test data)