Course Project: Handwritten Greek Letter Classification

Project Due: Wednesday, April 23, 2025, 11:59 PM

Group Size: 2-3 individuals

Final Materials Due: final report and code implementation

Description

In the final project, you will develop a machine learning system to classify handwritten (lowercase) greek letters. The data set is to be collected by all students enrolled in this course. You can implement this system yourselves or using a package/library. You can use any packages that come as a default option with Anaconda or PyTorch. We need to be able to run your implementation on our machines. So, be sure to get approval from me for any special packages! If we cannot run the code, you will lose a significant number of points.

Data Set

Each student will collect part of the training set that everyone will use to train their machine learning models. There's a total of 10 greek letters (10 classes). The table below shows the integer encoding we will use for all 10 classes. Each individual will collect 10 images for each greek letter, giving a total of 100 images per student.

Letter	Label	Integer Encoding
α	alpha	0
β	beta	1
γ	gamma	2
δ	delta	3
ϵ	epsilon	4
η	eta	5
θ	theta	6
λ	lambda	7

Letter	Label	Integer Encoding
μ	mu	8
π	pi	9

These images will be edited to a standard size and format using a process detailed in the Data Collection assignment. The easy test set will consist of data from these 10 classes and be used to evaluate everyone's models. The hard test set will also include samples not from any of these classes, to be classified with the integer label -1. Graduate students will be evaluated on both the easy and hard test sets, while undergraduates who choose to be evaluated on the hard test set in addition to the easy test set may receive extra credit.

Project Report

Each group will submit a report that includes the sections listed below. The report should follow the IEEE transactions format (single space, double column). You can find a Word or LaTeX template for the IEEE transactions format at https://www.ieee.org/conferences/publishing/templates.html

The report should focus on training and validation/testing strategies for the contest and any unique elements of your implementation(s). The maximum number of pages for the report is 4. Any pages beyond 4 will not be read nor graded. The report should be written with correct English grammar and spelling, and be precise. Equations and/or pseudo-code can be used for precise explanations.

The following sections must be included for full credit:

- **Abstract** : A summary description of the contents of the report and your findings.
- **Introduction** : Overview of your experiment/s and a literature review.

For the literature review, include any references to any relevant papers for your experiment/s. So, whatever you decide to do, search the ACM and IEEE (or other) literature for relevant papers to read and refer to.

- **Implementation** : Describe and outline any specific implementation

details for your project. A reader should be able to recreate your im- plementation and experiments from your project report. Be sure to describe the methodology with respect to how you will identify unknown classes that were not in the training data (if you are a graduate group or undergraduates going for extra credit).

- **Experiments** : Carefully describe your experiments with the training

data set and any data augmentation set you constructed or procured from existing data sets. Include a description for the goal of each experiment and experimental findings. This is the bulk of what you will be graded on

- if your experimental design is not sound or your experiments do not

make sense, you will lose points.

- **Conclusions** : Describe any conclusions or things you learned from the

project. Your conclusions must follow from what you did. Do not copy something out of a paper or say something that has no experimental support in the Experiments section.

- **References** : List of all references in IEEE bibliography format.

For writing the report, it is recommended that you use Google Docs through your UFL account (if you wish to use a Word document) or Overleaf (if you wish to use LaTeX - association with a UFL e-mail will give you a premium account). Both options will allow multiple team members to work on the report at once, and let you export a PDF.

Project Code Implementation

You can use any packages that come as a default option with Anaconda or PyTorch. We need to be able to run your implementation on our machines. So, be sure to get approval from me for any special packages! If we cannot run the code, you will lose a significant number of points. You can implement your algorithm using Jupyter Notebook or your favorite integrated development environment (IDE). Your final code submission should contain (at least) 3 files:

- **README.txt** : an explanation of your code including anything necessary to properly run it, e.g., packages used, inputs, and outputs.
- **train.py or a Notebook** with a function "train" defined : This function should contain the code used to train your final model
- **test.py or a Notebook** with a function "test" defined : This function should receive data and labels in the same format as the training data and output an accuracy value and predicted labels.
- **If you are going to be evaluated on the hard test set,** you may create a separate file (test_hard.py) with a function defined for testing on the hard test set "test_hard", or include such a function in test.py. This function should also receive data and labels in the same format as the training data and output an accuracy value and the predicted labels.

Grading Details

Your grade (out of 300 points) will be determined using the following rubric

- **(5 pts) Project Group Formation** : You have formed a project group meeting the group requirements by the deadline
- **(25 pts) Project Data Collection** : You have submitted the appropriate amount of properly formatted data (individually graded)
- **(75 pts) Project Code Implementation** : You have turned in code that runs correctly and easily on our machines. This requires a very clear README and easy to modify parameter settings. This also requires clearly listing what packages/libraries are needed to run your code - and checking with me before the due date to ensure we have those libraries. The code should meet any other submission requirements listed in this document or in the Project Code assignment.
- **(75 pts) Accuracy on Test Data Set(s)** :
  - **(EEL 4773)** Full points on this component will be obtained if you correctly classify 90% of the easy test data or have a classification accuracy rate greater than the average classification accuracy rate of the undergraduate class (whichever is lower).
  - **(EEL 5840)** Full points on this component will be obtained if you correctly classify 90% of the easy and hard test data or have a classification accuracy rate greater than the average classification accuracy rate of the graduate class (whichever is lower).
  - **(Both)** Your code should produce the integer-coding label encoding defined in the table in the Data Set section.
- **(120 pts) Project Report** This component will be graded based on the requirements listed in this document and any other requirements in the Project Report assignment.

Extra Credit

The goal of this project is to implement an end-to-end system to perform handwritten math symbols classification. The teams with the best classification accuracy on the hard data set in each class (EEL 4773 and EEL 5840) will earn extra credit. The hard data set will have all 10 classes (labels 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) and class unknown (label -1). There will be test data points from classes that do not appear in the training data. So, you will want to come up with a way to identify when a test point class is “unknown” or was not in the training data. The label you should return for this case is -1. Please have your test function output a class label that matches the class value in the provided training data. These should be: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 and -1.

Submission Details

Turn in your project report and your code to the relevant assignments by Wednesday, April

23 at 11:59 PM.

Again, be sure you include the following files:

- A PDF of your group's project report
- train.py or a Notebook which includes a function "train" that will run your training code on an input data set X and desired output vector T. Any parameter settings must be easy to find and modify.
- test.py or a Notebook which includes a function "test" that will run your testing code on an input data set X. Note: Your test.py code should already be trained and have parameters set! Any parameter settings must be easy to find and modify. It should return a vector with the class label associated with each input data point in X
- A concise README.txt file that clearly illustrates how to properly run your code. As your classification accuracy on a small test data set will

factor into your project grade, if we cannot run the code, the accuracy is 0.