|  |
| --- |
| **HW2: Classifying digit images (MNIST)** |
| **Published Date:** August 14,2020 |
| **Deadline Date:** October 19th, 4:30 p.m. |
| **Description:**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **This can be done in teams of up to two (2) people.**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **Overview and Assignment Goals:**  The objective of this assignment are the following:   * Implement the classification methods described below to predict digits in 28x28 images. * Choose the Best Model and submit predictions on the Test data to the leaderboard .   **Detailed Description:**  The MNIST dataset is considered the standard of handwritten digit recognition.  For the purposes of this assignment, you are given a labeled dataset of 60,000 images of 28x28 pixels each (links below) and their corresponding labels. You are also given test images of the same kind, with labels. All this data is already in the form of [numpy arrays](https://www.geeksforgeeks.org/numpy-load-in-python/) .  Your job is to implement the following models:   * A Neural Network (ANN) model using [Keras](https://keras.io). You have to consider and test at least **three** Keras models by varying the number of layers, activation functions in the hidden layers, etc. Tune those parameters with cross validation. We provide template file named [pa2Template.py](http://cs.gmu.edu/~dbarbara/CS584/pa2Template.py) which should be used as a starting point for each model. NOTE that another template file, named [pa2pre.py](http://cs.gmu.edu/~dbarbara/CS584/pa2pre.py), must be copied for each model, creating a new file (pa2pre m1.py for model 1, pa2pre m2.py for model 2, etc.). This file must contain ALL preprocessing performed on your dataset for the model (i.e, reshaping the data). The template file , pa2Template.py, is already configured to call   the method inside pa2pre.py , however, you will need to change the import statement to reflect your specific model (e.g., “from pa2pre m1 import”). Preprocessing needs to be performed on both the training and test datasets.  Each python file (for each design) should store your computed model to a h5 file (using model.save('')). Describe in your report how to run your python so that it creates the h5 files.  • Use the template file for each of your models and name them m1.py, m2.py, and m3.py.  • Name your model files m1.h5, m2.h5, and m3.h5.  Submit your best model (h5 file) and your pa2pre.py file as a zip. **Rename your py file pa2PreBest.py and your h5 file PA2BestModel.h5. Name the zip file PA2.zip.**  (Zip this zip file with all the rest of the submission.) Make sure that the submission does not have any directory information within it. I will run your model on a “secret” test set.  For every single model, analyze your results on the test data Xtest1 and use the labels provided to evaluate it. Construct a 10x10 confusion matrix. Create a heatmap with annotation. A template for producing a heatmap is available ([confusionMatrixHeatmap.py](http://cs.gmu.edu/~dbarbara/CS584/confusionMatrixHeatmap.py)).     * Add noise to the training data by using the script [addNoise.py](http://cs.gmu.edu/~dbarbara/CS584/addNoise.py) that takes the following parameters:   + --inputFile (Name of the numpy file (input))   + --sigma (the value of σ (used when adding the noise))   + --outputFile (name of the output file)   The script contains the line np.random.seed(1671). This is **VERY** important as it makes the pseudo-random numbers produced the same each time the program is executed (and thus, produces consistent output). The script adds “simulated” noise to each pixel in your dataset by sampling from a Gaussian distribution (µ = 0 and σ = taken from the command line parameters). addNoise.py uses the numpy.random.normal method to generate the simulated noise.  Use your best Keras model to perform validation for this new data and compare the results to what you got without noise. Create a confusion matrix and heatmap.   * Generate a bar plot that illustrates the performance of each of the models you created along the way. The script [barPlotTemplate.py](http://cs.gmu.edu/~dbarbara/CS584/barPlotTemplate.py) helps you produce this. Include it in your report.     **Rules:**   * This is a team assignment. Discussion of broad level strategies is allowed but any copying of prediction files and source codes will result in honor code violation.   **Deliverables:**   * **Blackboard Submission of Source Code and Report:**   + Create a folder called HW2\_LastName(s) \* include both names if you’re working in a team \*   + Create a subfolder called src and put all the source code there.   + Include in that sub folder your saved model. This should be a zip file of your best (h5 file) and the pa2pre.py file. (Naming conventions above; Follow them!)   + Create a subfolder called Report and place a 2-Page, single-spaced report describing details regarding the steps you followed for developing the classifier for predicting the product review sentiments. Be sure to include the following in the report:   1. Team Name Registered on miner web-site.   2. Rank & Accuracy score for your submission (at the time of writing the report).   3. Your Approach   4. Your methodology of choosing the approach and associated parameters.   5. All the plots you generated.   6. Archive your parent folder (.zip or .tar.gz) and submit via Blackboard for HW2.     **Grading:**  Grading for the Assignment will be split on your implementation (50%), report (20%) and ranking results (30%). |
| **Files:**   * *Train Data:*[Download File](http://cs.gmu.edu/~dbarbara/CS584/MNISTXtrain1.npy) * *Training Data labels:* [Download File](http://cs.gmu.edu/~dbarbara/CS584/MNISTytrain1.npy) * *Test Data:*[Download File](http://cs.gmu.edu/~dbarbara/CS584/MNIST_X_test_1.npy) * *Test Data labels:*[Download File](http://cs.gmu.edu/~dbarbara/CS584/MNIST_y_test_1.npy) |