# CPSC 585 - Artificial Neural Networks

Project 5, Spring 2021

due April 14

*Last updated Thursday April 2, 2:35 am PDT*

**Caution**: the version of the EMNIST Letters dataset provided by Tensorflow Datasets is incorrect. If you downloaded emnist\_letters.npz prior to Friday April 2, follow the link below to download a new version. The corrected version should have size 68,058,233 bytes and MD5 checksum 36ea18093d8de0d8f1d9d2601276a4f8. Please re-read the updated dataset description below, and note that experiment *(3)* has changed to use a dedicated validation set rather than specifying a validation split.

The character recognition problem in previous projects has used low-resolution bitmap images of letters from dot-matrix fonts. This project tackles a much more complex character recognition problem using higher-resolution grayscale images of hand-written letters. This problem will require a significantly more complex network, and will require more advanced approaches to training and tuning.

The project may be completed individually, or in a group of no more than three students.

## Platforms

The platform requirements for this project are the same as for [Project 1](https://docs.google.com/document/d/1UING3xL5oPf0zRLnBNyfX4N4mdDO7Aeo6XzuNgRMQ1A/edit?usp=sharing), but the project uses a larger dataset that will require significantly more training. If you do not have access to a physical machine with a GPU, you should consider using Google Colab with a [GPU](https://colab.research.google.com/notebooks/gpu.ipynb) or [TPU](https://colab.research.google.com/notebooks/tpu.ipynb) or another cloud service such as [Kaggle Notebooks](https://www.kaggle.com/docs/notebooks) ([GPU](https://www.kaggle.com/docs/efficient-gpu-usage), [TPU](https://www.kaggle.com/docs/tpu)), or [Gradient Community Notebooks](https://medium.com/@HelloPaperspace/introducing-gradient-community-notebooks-easily-run-ml-notebooks-on-free-gpus-f3fa36336b3c).

## Libraries and Code

This project should use the [NumPy](https://numpy.org/) and [Keras](https://keras.io/) libraries. You may also wish to use other components of [TensorFlow](https://www.tensorflow.org/) and Python libraries such as [scikit-learn](https://scikit-learn.org/) and [pandas](https://pandas.pydata.org/).

Code from [*A Whirlwind Tour of Python*](https://jakevdp.github.io/WhirlwindTourOfPython/) and from the library documentation may be reused. All other code and the results of experiments must be your own original work or the original work of other members of your team.

## Warmup

As described in Section 1.8 of the textbook, the MNIST database of handwritten digits is a standard benchmark for computer vision. This dataset is [included with Keras](https://www.tensorflow.org/api_docs/python/tf/keras/datasets/mnist) and is a good introduction to working with larger datasets.

[This notebook](https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/2.1-a-first-look-at-a-neural-network.ipynb) by Francois Chollet creates a simple Multilayer Perceptron as described in Section 2.1 of *Deep Learning with Python*. (Recall that this book is available [from the library](https://xerxes.calstate.edu/fullerton/databases/proxy/59).) Update the code to use [tf.keras](https://www.tensorflow.org/guide/keras/), then try running it to see if you can match the reported accuracy of 97.8%

## Dataset

The Extended MNIST or [EMNIST dataset](https://www.nist.gov/itl/products-and-services/emnist-dataset) expands on the original MNIST, adding handwritten letters as well as additional samples of handwritten digits. There are several “splits” of the data by various characteristics. We will be using the “EMNIST Letters” dataset, which contains data split into 27 classes, one unused (class 0) and one for each letter in the English alphabet.

*Note*: Some classes in this dataset can be challenging to recognize because each class contains images of both upper- and lower-case letters.

Download [emnist\_letter.npz](https://drive.google.com/file/d/1m6FN1CrVDlQSvrbcLQmI-J81w5kDxX8P/view?usp=sharing). This file can be opened with the [numpy.load()](https://numpy.org/doc/1.19/reference/generated/numpy.load.html) method. The data contains six arrays: 'train\_images', 'train\_labels', 'validate\_images', 'validate\_labels', 'test\_images', and 'test\_labels'. The arrays from the original dataset have been altered to match Chollet’s MNIST notebook:

* The images have been transposed and scaled to floating point.
* The labels have been one-hot encoded.

## Experiments

Run the following experiments in a Jupyter notebook, performing each action in a [code cell](https://jupyter-notebook.readthedocs.io/en/stable/notebook.html#code-cells) and answering each question in a [Markdown cell](https://jupyter-notebook.readthedocs.io/en/stable/notebook.html#markdown-cells).

1. Use [plt.imshow()](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.imshow.html) to verify that the image data has been loaded correctly and that the corresponding labels are correct.
2. Begin by applying the network architecture from Chollet’s MNIST notebook to the EMNIST Letters data. What accuracy do you achieve? How does this compare with the accuracy for MNIST?.
3. Keeping the same number of layers in the network (i.e. an MLP with a single hidden layer), modify the architecture to improve the accuracy. You will need to decide on an appropriate number of neurons in the hidden layer. Keep in mind that:
   * There are 27 classes rather than 10, so you will need a larger hidden layer than the MNIST network.
   * In addition to having more classes, EMNIST Letters mixes upper- and lowercase letters within each class, so even with enough neurons in the hidden layer, your accuracy is likely to be lower. See the details in the [EMNIST paper](https://arxiv.org/abs/1702.05373) for the kind of performance you might reasonably expect.
   * The Keras [fit()](https://keras.io/api/models/model_training_apis/#fit-method) method can take a validation\_data parameter in order to evaluate metrics on the validation set.
4. Once you have settled on the size of the hidden layer, use the techniques you learned in Chapters 3 and 4 of the textbook to obtain the highest accuracy you can on the validation set. These might include:
   * Preprocessing
   * Weight initialization
   * Choice of activation function
   * Optimizer
   * Batch Normalization
   * Regularization
   * Data augmentation
   * Dropout
   * Early Stopping

You may find the [slides for Chapter 3](http://www.charuaggarwal.net/Chap3slides.pdf) helpful, particularly the presentation “Neural Network Training [Initialization, Preprocessing, Mini-Batching, Tuning, and Other Black Art].”

1. Add additional [Dense](https://keras.io/api/layers/core_layers/dense/) hidden layers as appropriate to improve the accuracy. Note that you may need to adjust your hyperparameters or other aspects of the network architecture in response to these changes. How does the accuracy for your deep network compare with the accuracy you achieved in experiment *(4)*?.
2. When finished tuning, evaluate your results on the test set. Compare the test performance of your original network from experiment (2) and the final networks for experiments *(3)* and *(4)*.
3. Use plt.imshow() to view some of the misclassified images and examine their labels. Describe what you think might have gone wrong.

## Submission

A Markdown cell at the top of the notebook should include project summary information [as described in the Syllabus](https://docs.google.com/document/d/1TG4qngijN2ZNLpvok1_NIeE-iELeN6F2WYiFVZprbBQ/edit#heading=h.5162uorf65x7) for README files.

Since you may be actively editing and making changes to the code cells in your notebook, be certain that each of your code cells still runs correctly before submission. You may wish to do this by selecting *Run All* from the drop-down menu bar.

Submit your Jupyter .ipynb notebook file through Canvas before class on the due date.

If the assignment is completed by a team, only one submission is required. Be certain to identify the names of all students on your team at the top of the notebook. See the following sections of the Canvas documentation for instructions on group submission:

* [How do I join a group as a student?](https://community.canvaslms.com/t5/Student-Guide/How-do-I-join-a-group-as-a-student/ta-p/468)
* [How do I submit an assignment on behalf of a group?](https://community.canvaslms.com/t5/Student-Guide/How-do-I-submit-an-assignment-on-behalf-of-a-group/ta-p/294)