Intelligent Systems Engineering
ECE 449
Fall 2023

Laboratory Exercise #3

Resources: *Machine Learning with Python*, from Cuantum Technologies (required text).

- Chapter 6, Introduction to Neural Networks and Deep Learning
- Chapter 7, Deep Learning with Tensorflow
- Chapter 8, Deep Learning with Keras

[1] Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner: Gradient Based Learning Applied to Document Recognition, Proceedings of IEEE, 86(11):2278–2324, 1998.

Summary: In this project, you will explore image classification using Convolutional Neural Networks (CNNs). Working with a well-known dataset (MNIST Digits, distributed with Keras), you will design and parameterize a basic CNN (specifically, the LeNet architecture using some modern design choices) using a stratified cross-validation experimental design. You will evaluate the performance of your best design on a predetermined out-of-sample test set (this is common in deep learning research).

Discussion: At this point in time, CNNs are the most effective algorithms known for image classification / computer vision. While CNNs have been applied to many other problems, image classification is in fact what they were originally designed for in [1]. Tensorflow, and its companion library Keras, are one of the most important platforms for training CNNs and other deep learning algorithms. In my experience, the simplified Keras interface is the appropriate choice for a simple deep learning project such as this one.

Requirements: In this project, you will solve an image classification task using an updated version of the architecture of [1] (which is a simple sequence of alternating convolution and pooling layers). The updates are:

- Use a ReLU activation instead of a sigmoid one.
- In you pooling layers, use the MAX() function instead of the arithmetic mean.
- For classification, use a single dense layer followed by a softmax layer.

For your experiments, you will use the MNIST-Digits dataset (packaged with Keras). The dataset is provided already split into a training and a testing set. Normalization is not necessary for the CNN, and there are no missing values. You must, however, convert the class labels into a one-hot encoding. While it would be preferable to do so in an automated pipeline, testing on Google Colab indicates that this results in an out-of-memory error. More details are provided in the lab lecture.

Your task will be to design and parameterize the CNN in Tensorflow / Keras using the training dataset, and then evaluate your final, best design on the test dataset. You will employ a 5-fold stratified cross-validation design for exploring your hyperparameters in the training set (running

MNIST-Digits is computationally demanding, so we will prefer a 5-fold design instead of the usual 10 folds). Once you have determined the best design, train the CNN one more time on the entire 60,000-image training set, and then report your performance on the out-of-sample test set (10,000 images). The scientific literature on MNIST-Digits almost always just reports the classification accuracy on the test dataset; this is what you should report as well.

Submission: All of the above (code to execute the data ingestion, preprocessing, and parameter exploration, final conclusion) should be submitted as a Jupyter notebook, with the parameter exploration taking place in an automated pipeline. The TAs will grade your work by executing the notebooks on a fresh copy of the dataset, downloaded from the same source. Submit the notebooks via eClass in your lab section before the deadline.

Grading

CNN trained via cross-validation:	75%
Pipeline automated:	25%