AMCS 215
Mathematical Foundations of Machine Learning
Assignment 6, Due October 17, 2023

Reading material:

- Chapter 9 from Reference 1
- Chapter 5 from `https://d2l.ai`

Notes:

- Because of the mid-semester break, this assignment is due on Tuesday instead of the normal Sunday. We will get back to a Sunday schedule the following week.
- Exam is scheduled for Nov 1. Please bring your laptops to the exam.

1. Step length for gradient descent.

When the loss function f has an L-smooth gradient with a known Lipschitz constant L, the learning rate α can be chosen to ensure a quantifiable reduction in loss, which we derive in this exercise.

- combine a Taylor expansion with the L-smoothness of the gradient to derive the following:

$$f(x + \alpha d) \leq f(x) + \alpha \nabla f(x)^T d + \alpha^2 \frac{L}{2} \|d\|^2$$

- show that if we move in the negative gradient direction $d = -\nabla f(x^k)$ the step length $\alpha = 1/L$ minimizes the expression on the right
- show that this learning rate guarantees a reduction in the loss that is at least $\frac{1}{2L} \|\nabla f(x^k)\|^2$

2. Comparing three classifiers.

In this exercise we will build three different classifiers for a dataset: a multiclass logistic classifier; a generative classifier that fits a Gaussian mixture to the data; and a multilayer perceptron with one hidden layer.

You will find on blackboard a starter file that defines a dataset with $D = 2$ features and $C = 3$ classes.

- generate a multiclass logistic classifier for the dataset
- generate a Gaussian to fit each class $p(x|y = c; \theta_c)$ and build the class posteriors $p(y = c|x; \theta_c)$ to produce a 3-class classifier.
- build and train a feedforward network with one hidden layer (consisting of 4 neurons) and `relu` activation.
- how would you assess the quality of these classifiers? Generate appropriate tables and comment.
- compare the performance of these three classifiers on the given dataset. Explain the performance.

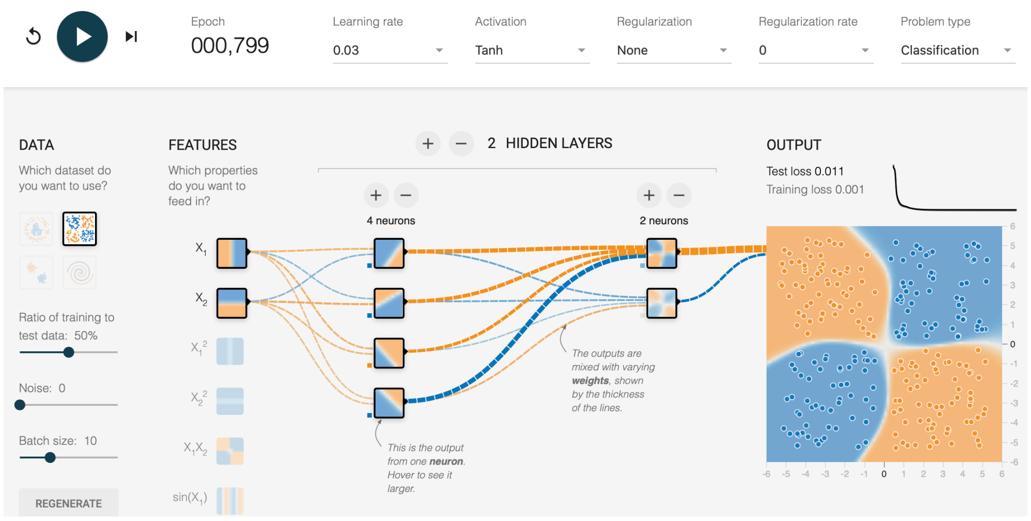
3. Computational complexity of backpropagation

Consider an MLP (aka feedforward) deep neural network, with k dense hidden layers each consisting of n neurons with ReLU activations, and a final dense layer with a single neuron whose value is used in a binary classification task. For simplicity, we assume the input is a vector of size n. We train the network with batches of size b input samples.

- in terms of k, n, and b, how many floating point operations (FLOPS) are needed to evaluate the network for a batch? (write down the dominant term only, in $O()$ notation)
- in terms of k, n, and b, how many FLOPS are performed during the backward phase of backpropagation for computing the gradient of the loss from a batch?

- in terms of k, n, and b, how much memory is needed during the forward/backward passes of backpropagation?

4. Interpretation of neurons response.
`playground.tensorflow.org` has a visual interface to an MLP network for binary classification of datasets whose elements consist of two features. There are a few sample datasets illustrated. Consider the one that consists of a class of points in the first and third quadrants, and a second class in the second and fourth quadrants (see screenshot below). Use a trained network similar to the one below to answer the following questions.


Epoch 000,799
Learning rate 0.03
Activation Tanh
Regularization None
Regularization rate 0
Problem type Classification

Ratio of training to test data: 50%
Noise: 0
Batch size: 10

Test loss 0.011
Training loss 0.001

The outputs are mixed with varying **weights**, shown by the thickness of the lines.

This is the output from one **neuron**. Hover to see it larger.

- give a very brief description of the features that each of the four neurons of the first hidden layer appear to be detecting? The description should be in terms of the geometry of the classification. (recall that the output of a neuron in this layer is $z_j = \varphi(w_j^T x)$, i.e., it "lights up" when the input x matches the template defined by its w_j weights—we are basically trying to give a geometric interpretation of each of these w_j templates)
- consider the neuron of the second hidden layer that most strongly affects the final output. Can you briefly explain how the feature that it is detecting is obtained from a combination of the features of the neurons from the first layer? Could this feature have been generated by a direct combination of the input features x, or is the first layer necessary here?

5. MLP for MNIST data.
You will on Blackboard a starter file for training an MLP network on the MNIST dataset, where each image is represented as vector of 784 values.

- define a network consisting of two hidden dense layers of 128 neurons each with ReLU activations, and a dense output layer of 10 neurons.
- it is typical in practice to have the model **not** output a probability distribution, but rather a vector of logits (10 in this case). Turning these logits into a probability distribution via the softmax function is then done inside the loss function. Explain briefly why this is done, i.e., what difficulties does this avoid?
- write down the mathematical expression of the cross entropy loss function you will use to train this model
- how many trainable parameters are there in this model?
- write a training loop using stochastic gradient descent with a batch size of 32.
- train the model for a few epochs and plot a few images along with their predicted labels. You may use the following hyperparameters (learning rate 10^{-2}, momentum 0.9) for the training.
- evaluate the performance of the trainedmodel by computing the percentage of correct predictions it generates on unseen (test) images