Homework 1: MNIST with NumPy

CS 1470/2470

Due September 25, 2019 at 11:59pm

1 Conceptual Questions

1. What is the difference between the "Perceptron Learning Algorithm" and the algorithm we implemented in this assignment? (2-5 sentences)

   Hint: Consider how and when we update weights.

2. In lecture, the update rule for weights in a single layer, multi-class neural network with cross entropy loss was defined as follows:

   Let w_{ji} be the weight that corresponds to the jth class and the ith input feature, x_i. Let c equal the correct class for a given input. Our loss is then:

   $$L = -\log(P_c)$$

   If $j = c$, then:

   $$\frac{\partial L}{\partial w_{ji}} = (P_j - 1)x_i$$

   Otherwise:

   $$\frac{\partial L}{\partial w_{ji}} = P_j x_i$$

   We use these partial derivatives to descend our gradients as follows:

   $$w_{ji} = w_{ji} - \frac{\partial L}{\partial w_{ji}} \cdot \alpha$$

   where α is the learning rate.

   Derive the above values for $\frac{\partial L}{\partial w_{ji}}$ from cross entropy loss.

   Hints:

   (a) *Consider the two cases of j.*

   (b) *Start by expanding out L.*

3. Why do we use a bias vector in our forward pass? (2-4 sentences)

4. Why are GPUs invaluable for training neural networks? What properties of neural networks make them well-suited for execution on GPUs? (2-5 sentences)

5. (Optional) Have feedback for this assignment? Found something confusing? We'd love to hear from you!

2 Ethical Implications

1. What are some qualities of MNIST that make it a "good" dataset for a classification problem? (2-3 sentences)

2. **Algorithms in the real world:** Suppose you are an administrator of the US Postal Service in the 1990s.

   (a) What positive effects (if any) would result from deploying an MNIST-trained neural network to recognize handwritten zip codes on mail? (1-4 sentences)

   (b) What negative effects (if any) might result from deploying such an architecture? (1-4 sentences)

3 CS2470-only Questions

1. Cross entropy is a useful loss function for classification, but it is not appropriate for all tasks. Another common loss function is the *squared error loss*:

   $$L(\mathbf{y}, \mathbf{a}) = (\mathbf{y} - \mathbf{a})^2$$

   where $\mathbf{a}$ is the answer. This type of loss is useful for e.g. training a neural to predict temperature given input data about a place on earth (e.g. its latitude and longitude, the time of year, etc.).
   Derive $\frac{\partial L}{\partial w_i}$ for a single-layer network with this loss function.

2. In your introductory calculus class, you were likely exposed to the following simple algorithm for finding the minimum of a function: take the derivative of the function, set it to zero, then solve the equation. Neural networks are differentiable, so why don't we use this algorithm (instead of gradient descent) to minimize the loss? (1-4 sentences)

3. Prove that SGD with a batch size of 1 gives an unbiased estimate of the 'true gradient' (i.e. the gradient calculated over the entire training set). Assume that the batch is selected uniformly at random from the full training set.
   Hints:

   (a) *Recall that an estimator $\hat{f}$ is an unbiased estimator of a function f if $\mathbb{E}[\hat{f}] = f$.*

(b) *Both expectation and differentiation are linear operators.*