Neural Network Explanatory Text

Andy Xu

September 2020

The python file that this text supplements is a neural network that I implemented from scratch. I programmed all of the functions needed to initialize, train, evaluate, and make predictions with the network. The trained neural network labels images of handwritten letters. I opted to use a subset of an Optical Character Recognition (OCR) dataset that includes 10 letters, though the neural net can equally well handle 26 letters.

Model Definition The model is a single-hidden-layer neural network with a sigmoid activation function for the hidden layer, and a softmax on the output layer. Let the input vectors $\mathbf{x}$ be of length M, the hidden layer $\mathbf{z}$ consist of D hidden units, and the output layer $\hat{\mathbf{y}}$ be a probability distribution over K classes. That is, each element y_k of the output vector represents the probability of $\mathbf{x}$ belonging to the class k. See the next page for a graphical representation.

$$\hat{y}_k = \frac{\exp(b_k)}{\sum_{l=1}^{K} \exp(b_l)}$$
$$b_k = \sum_{j=0}^{D} \beta_{kj} z_j$$
$$z_j = \frac{1}{1 + \exp(-a_j)}$$
$$a_j = \sum_{m=0}^{M} \alpha_{jm} x_m$$

The objective function I will use for training the neural network is the average cross entropy over the training dataset $= \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}$:

$$J(\boldsymbol{\alpha}, \boldsymbol{\beta}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_k^{(i)} \log(\hat{y}_k^{(i)}) \tag{1}$$

To train, I will optimize this objective function using stochastic gradient descent (SGD), where the gradient of the parameters for each training example is computed via backpropagation.

Neural Network

Neural Network for Classification