E0 250 - Deep Learning

Project 1 - FizzBuzz

Deepesh Virendra Hada
M.Tech, Department of CSA
SR: 17196

27 January, 2020

Network Architecture

The neural network constructed is a *fully connected neural network* (FCNN) with 3 layers, an *input* layer, a *hidden* layer and an *output* layer. Following are the number of neurons in each layer:

- Hidden Layer (Logistic) - 10
- Output Layer (Softmax) - 4

The input, which is an integer, is first converted into its binary representation in 10 bits. It is then passed on to the hidden layer with certain *weights*, given by the *weight*1 matrix in the code. Also, a *bias* is applied to each neuron in the hidden layer, given by the *bias*1 vector. Initially, all the weights are randomly set and all the biases are set to 0. Each neuron, hence, takes a linear combination of the input variables, and applies a *logistic sigmoid* function to that combination. The output of each neuron is given as follows:

$$g(x) = \frac{1}{1 + e^{(w^T x + b)}} \tag{1}$$

All these outputs from the hidden layer are then given to the output layer, consisting of 4 neurons, with some weights and bias, given by the *weight*2 matrix and *bias*2 matrix, respectively. These 4 neurons correspond to the four output classes:

- Class 0 - **Fizz**
- Class 1 - **Buzz**
- Class 2 - **FizzBuzz**
- Class 3 - **Other**

Now, the final outputs are probabilities of an input belonging to any one of these four classes. Since the logistic function doesn't satisfy the properties of a probability distribution, the *softmax* function is applied to the *activation outputs* of the hidden layer neurons. For example, the probability of class j for an input x is given as:

$$p_j = \frac{e^{-z_j}}{\sum_{i=0}^{3} e^{-z_i}} \tag{2}$$

Hyperparameters

1. **Loss function:** The loss function used is the *Mean Sqaured Error* loss.
2. **Regularizer:** To prevent overfitting, *early stopping* regularizer has been used. The training of the model was intentionally stopped before the *loss* got minimized to 0 after a certain threshold loss.
3. **Epochs:** The number of epochs for which the training was done was based on the threshold loss given by the regularizer.
4. **Learning Rate, η:** We have used a fixed learning rate for training the model , after experimenting with the loss obtained with each.

Metrics

- **Accuracy:** The max output probabilities of the 4 classes were compared with the true labels of the training examples. The number of inversions were measured through these comparisons and the accuracy of the model was hence, calculated.

The following table lists the accuracy obtained with various hyperparameter tunings. Note that the individual class accuracy, i.e., $fizz, buzz, fizzbuzz$ and *other* class accuracy are calculated on the **test data (numbers from 1 - 100)**.

η	**Epochs**	**Loss**	**Train Accuracy**	**Test Accuracy**
8	17366	0.005	90.11%	84%
9.5	6921	0.025	95.44%	95%
11	22025	0.03	94.33%	95%
12	4420	0.04	90.66%	91%
8 till 4000 epochs, 3 after 4000	6713	0.045	91.33%	97%

Conclusion

The number of hidden layers used is 1 with 10 neurons and *logistic* activation. The output layer consists of 4 neurons and *softmax* activation.
The *hyperparameters* used are, $\eta = 8$ till 4000 epochs and 3 afterwards and final training loss = 0.045.
The accuracy obtained with this setting are:

- Training accuracy = 91.33%
- Test accuracy = 97%