cs577 Assignment 2 : Program Report

Pushpendra Singh Chauhan

Department of Computer Science

Illinois Institute of Technology

March 13, 2021

Abstract

This is a report for programming question in cs577 Assignment 2. It will contain the description of the problem I am trying to solve, the algorithms I employed to solve the problem, implementation details like program design issues and the results obtained. The results obtained will be analyzed for correctness. The performance of the algorithm will be evaluated and discussed.

1 Problem Statement:

- Implement and test a three layer neural network (i.e. two hidden layers and one output layer) for a three class-classification using python and without using a GPU framework. Use categorical cross entropy for loss, sigmoid activation function for hidden layers and softmax for the output layer. The computation graph and its forward and backward traversal should be hard coded in the program without having to support dynamic configurations. Evaluate the network you implemented using similar steps to that of the previous assignment. Use two or more datasets. It is essential that you evaluate performance properly. Make sure to explain the results you obtain and do not unnecessarily repeat similar results. The code you write should

be modular and well documented. The program needs to be written in python. Your implementation should include the following components:-

- A loss function and an evaluation function.
- A class for each node type with methods for computing a forward and backward pass.
- Forward and backward traversal of the computation graph where the data batches are pushed forward and the gradient loss values are pushed backward.
- Implementation of stochastic gradient descent with learning rate and decay parameters for updating model weights.

2 Proposed Solution:

- Used ‘to_categorical()’ function of keras library for encoding the known class labels.
- Implemented a ‘LinearLayer’ class with three methods: ‘forward’ – forward pass, ‘backward’ – backward pass and ‘update_params’ – performs gradient descent.
- Implemented a ‘’SigmoidLayer” class with two methods: ‘forward’ – forward pass and ‘backward’ – backward pass.
- Implemented a “SoftmaxLayer” class with two methods: ‘forward’ – forward pass and ‘backward’ – backward pass.
- Implemented “compute_cost” function for computing ‘categorical cross entropy’ loss in case of three class-classification and ‘binary cross entropy’ loss in case of binary classification problem.
- Implemented “accuracy” function for calculating the accuracy in order to evaluate the performance of the classification.
- Used ‘matplotlib’ library for plotting training and validation loss as a function of epochs, training and validation accuracy as a function of epochs.
- Used ‘StandardScaler’ of ‘sklearn’ library for normalizing the features.
- Used ‘train_test_split’ function of ‘sklearn’ library for splitting the data into training and testing subsets.

- Used ‘numpy’ library for dealing with tensor operations and data manipulation.
- Used ‘CIFAR10’ dataset form keras for three class-classification.
- Used spam email data from the UCI repository “spambase” for binary classification.

3 Implementation Details:

- FOR ‘CIFAR10’ DATASET (THREE CLASS CLASSIFICATION):- (a). Load the CIFAR10 data from keras datasets. (b). Selected subset of three classes from train and test data. (c). Split the training data into training and validation subset. (d). Convert the class labels to categorical. (e). Vectorize and normalize the image data. (f). Designed a fully connected neural network for three-class classification manually using computation graph nodes. It consists of two hidden layers and one output layer. (g). Used ‘softmax’ activation function and 3 neurons in output layer as it is multi-class classification problem. Used ‘categorical_crossentropy’ as loss function because it is multi-class classification problem. Used ‘sigmoid’ activation function for hidden layers. (h). Plot the training and validation loss as a function of epochs. (i). Plot the training and validation accuracy as a function of epochs. (j). Trained the network on training data and validated it with validation data. (k). Evaluated the model on test data.
- FOR SPAM EMAIL DATA (BINARY CLASSIFICATION):- (a). Download spam email data from UCI repository ‘spambase’ and put the two files named “spambase.data” and “spambase.names” in the ‘data’ folder. (b). Define a function named ‘load_spam_data’ to load and split the data. It will return the train and test data after normalizing the features. (c). Calling the ‘load_spam_data’ function. (d). Splitting the training data into training and validation subsets. (e). Design a fully connected neural network for binary classification manually using computation graph nodes. It consists of two

hidden layers and one output layer. (f). Used ‘sigmoid’ activation function and 1 neuron in output layer as it is binary classification problem. Used ‘binary_crossentropy’ as loss function because it is binary classification problem. Used ‘sigmoid’ activation function for hidden layers also. (g). Plot the training and validation loss as a function of epochs. (h). Plot the training and validation accuracy as a function of epochs. (i). Tuned model hyper parameters to improve the performance. (j). Trained the network on training data and validated it with validation data. (k). Evaluated the model on test data.

- Folder structure: AS2 (main folder) →

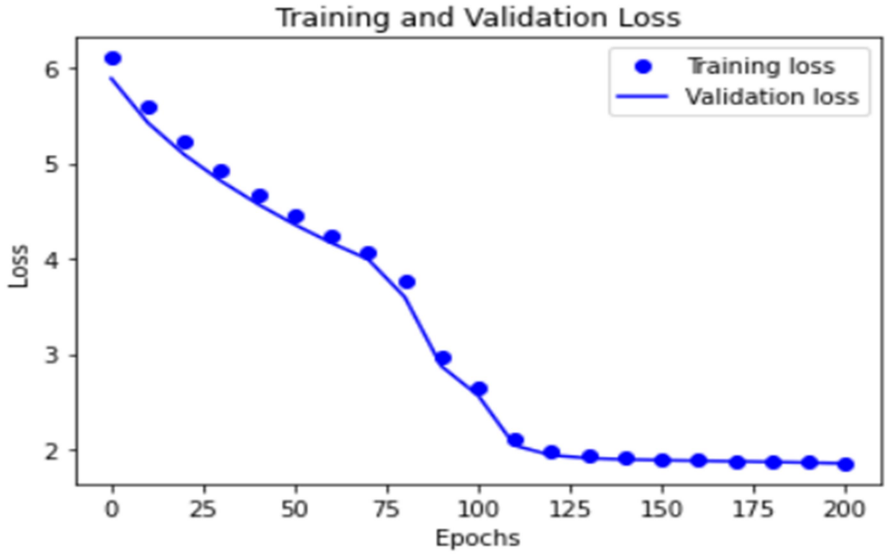
  (1). src (sub-folder) – It contains source code files (‘3-class classification.ipynb’, ’Binary_Classification.ipynb’)

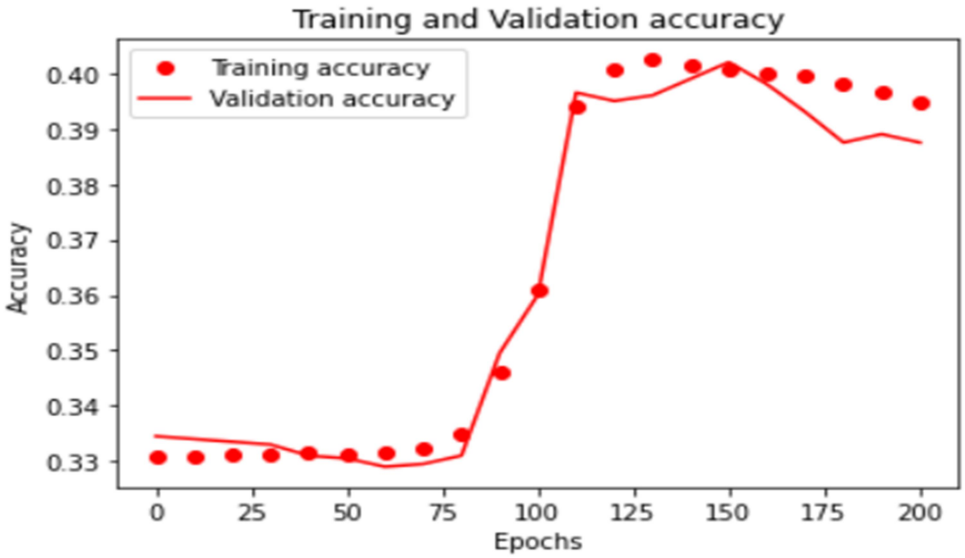
  (2). data (sub-folder) – Download spam data from UCI repository “spambase” and put the two files named ‘spambase.data’ and ‘spambase.names’ in this folder. Links to the dataset will be given in ‘data.txt’ file. ‘data.txt’ file will reside in this folder.

  (3). doc (sub-folder) – It contains two pdf files (‘review questions answers.pdf’ and ‘program report.pdf’).

4 Results and discussion:

- CIFAR10 DATASET (three class classification):- Training and validation loss, training and validation accuracy plot after training the network:

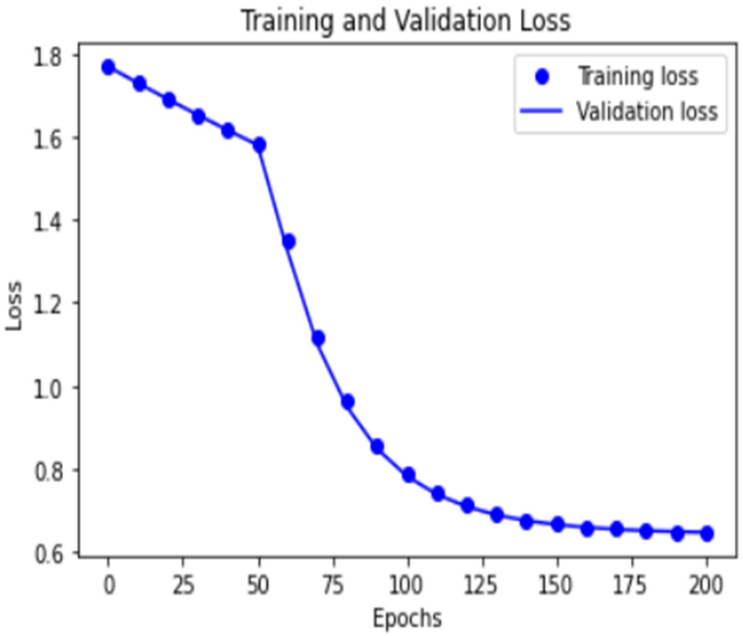


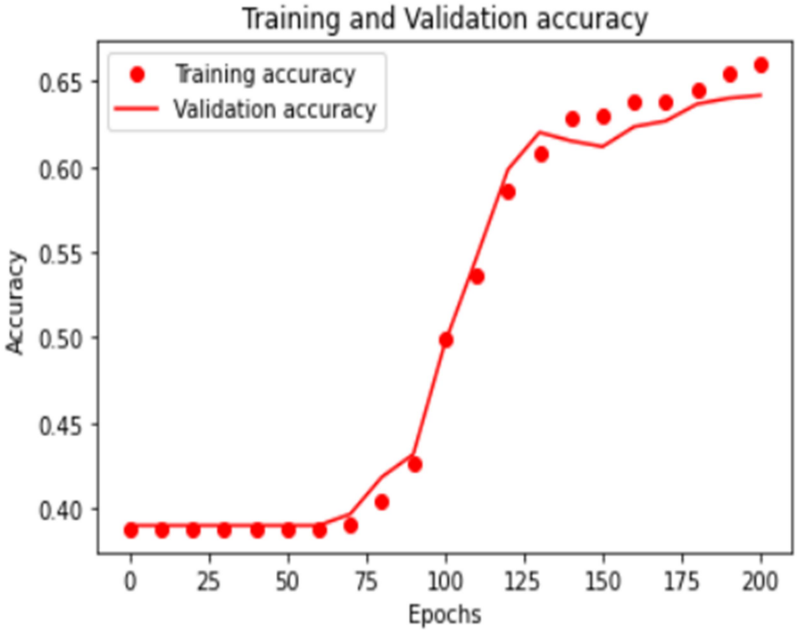


Test Accuracy: 38.4% Test Loss: 1.8777535310570248

From above plots, we can see that training and validation loss decreases with the increase in number of epochs. We can infer that our model is able to update the parameters so that loss can be decreased; it means network is working properly.

- SPAM EMAIL DATA (Binary classification):- Training and validation loss, training and validation mean absolute error after training the network:





Test Accuracy: 63.95% Test Loss: 0.66582689

From above plots, we can see that training and validation loss decreases with the increase in number of epochs. We can infer that our model is able to update the parameters so that loss can be decreased; it means network is working properly.

5 References:

- https://pub.towardsai.net/building-neural-networks-from-scratch-with-python-code-and-math-in-detail-i-536fae5d7bbf
- https://www.tutorialspoint.com/python_deep_learning/python_deep_learning_computational_graphs.htm
- https://medium.com/tebs-lab/deep-neural-networks-as-computational-graphs-867fcaa56c9
- https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/
- https://www.codingame.com/playgrounds/9487/deep-learning-from-scratch---theory-and-implementation/computational-graphs
- https://www.kdnuggets.com/2019/08/numpy-neural-networks-computational-graphs.html