Assignment 1. Handwritten Digit Recognition using TensorFlow and MNIST dataset

Shantanu Gupta

Arizona State University

Task 1

Task 1 has been performed using a Logistic Regression (LR) model.

Accuracy: 91.95%

Time spent: ~6 hours

Logistic Regression: Logistic Regression is a statistical model which uses a logistic function to model a dependent variable based on the values of independent variables. Basically, the independent variables are the input to any statistical model and the dependent variables are the outcome based on the inputs. A logistic function is a Sigmoid Curve following the equation:

$$f(x) = \frac{1}{1 + e^{-x}}$$

where x is the linear prediction made by the algorithm

Logistic Regression estimates the parameters of a logistic model. We are interested in estimating the probability that a given input belongs to which class. Mathematically, we want to estimate the probability $P[Y_i = 0]$

Our prediction function $f(x)$ returns a probability between 0 and 1. Based on this score, we want to define a decision boundary based on a threshold value. We model the likelihood of the given data using the

$$x = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots\ldots + \beta_k X_k$$

where β_i is the coefficient of sample X_i and β_0 is a constant

We then use a cost function to penalize confident but wrong predictions on the training set, with a low reward for confident and correct predictions. [2]

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} \log\left(h_\theta(x^i)\right) + (1 - y^i)\log(1 - h_\theta(x^i))]$$

Analysis: The Logistic Regression model is a statistical technique well suited to many predictive tasks. Although there have been many more advances in this field, particularly in more difficult tasks like visual and pattern recognition.

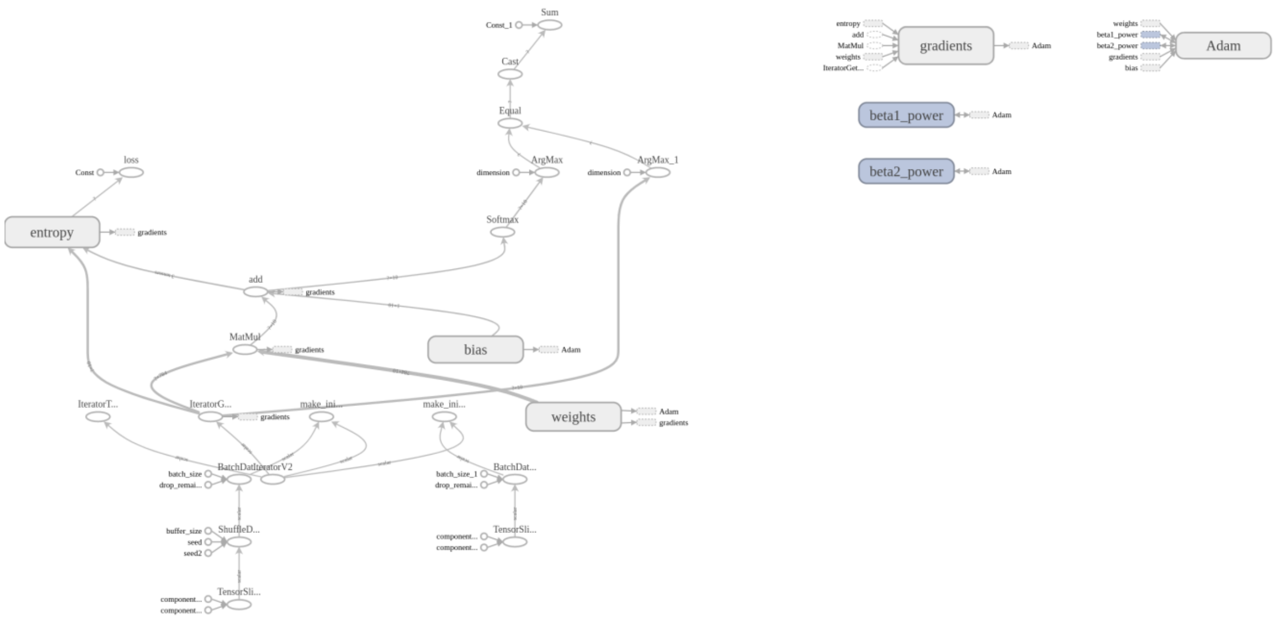

Figure 1: Logistic Regression TensorBoard Graph

Task 2

Task 2 has been performed using a Convolutional Neural Network (CNN) model.

Accuracy: 97.84%

Time spent: ~48 hours

Convolutional Neural Network: CNN is a type of Neural Network which utilizes the information of the neighborhood pixels to correctly classify visual patterns. The model consists of an input layer, a number of hidden layers consisting of Convolutional Layer, Pooling Layer, Fully Connected Layer and an output layer signifying the final class scores. The hidden layers identify visual patterns of the images in a series of patterns increasing in complexity. The number of parameters depends on the product of the size of the filter at each layer. The depth, stride and zero padding control the size of the output layer. The function of pooling layer is to reduce the number of parameters between layers to minimize overfitting. The fully connected layer is used at the end to assign the final prediction class to the inputs.

Analysis: CNN are a very effective technique, particularly suited to the task of visual recognition due to the inherent spatial relationship in images. It keeps the number of computations from increasing exponentially by pooling and still provides a high accuracy.

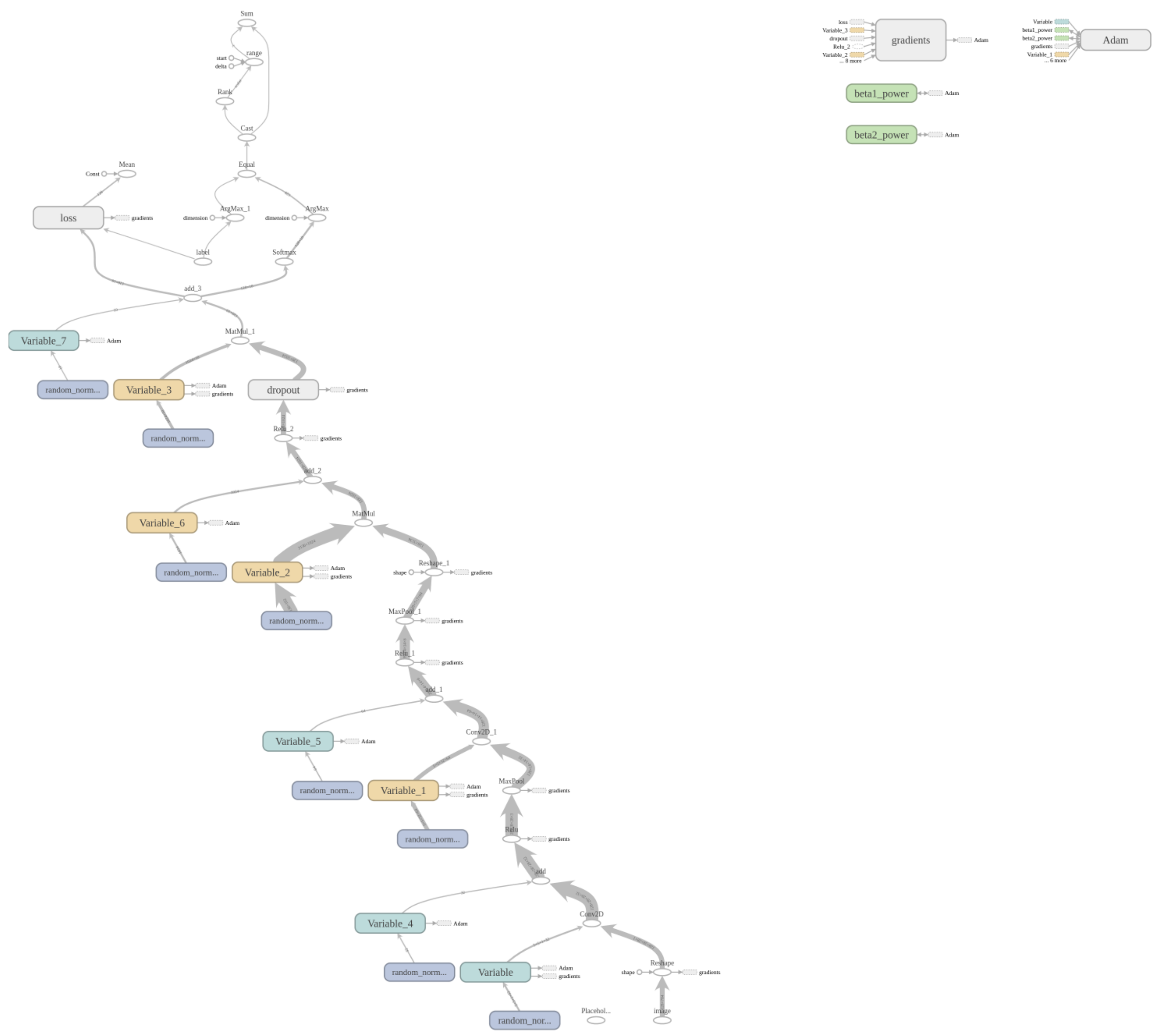

Figure 2: CNN TensorBoard Graph

Problems encountered:

i. Setting up environment with TensorFlow library, the mnist dataset was not available implicitly. I had to checkout the github repository for the dataset and copy it externally to the directory /python3.7/site-packages/tensorflow/examples/tutorials
ii. Removed undefined variable ‘data’ while creating dataset directory in Data Loading Approach 2.
iii. Change os.mkdir() to os.makedirs() to recursively create dataset directory defined by ‘mnist_folder’ in Data Loading Approach 2.
iv. Softmax vs SVM classifier to minimize entropy of distributions
v. I studied about the various popular architectures of CNN layer patterns used for image classification and optimal number of hidden layers keeping the model fast and accurate.
vi. Experimentally find the optimal keep_rate and number of epochs for better accuracy