**Assignment 5**

*Manpreet Kaur*

*December 2, 2019*

**Class:** ALY6020- \*\*Predictive Analytics Professor:\*\*Marco Montes de Oca

**Introduction:**

**Neural networks** are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input.

All classification tasks depend upon labeled datasets; that is, humans must transfer their knowledge to the dataset in order for a neural network to learn the correlation between labels and data. This is known as supervised learning*.*

A close up of a clock

Description automatically generated

A node layer is a row of those neuron-like switches that turn on or off as the input is fed through the net. Each layer’s output is simultaneously the subsequent layer’s input, starting from an initial input layer receiving your data.

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, Theano or CNTK. It is very popular in the research and development community because it supports rapid experimentation, prototyping, and user-friendly API.

**TensorFlow and Keras in Anaconda**

Tensor flow, is a famous Deep learning library created by google

In the terminal, use the command ``***conda install keras***`` It installed Tensor flow in its backend.

**Activation Function**

Sigmoid function is good for up to 2 hidden layers. However, as hidden layers increase, updating weights becomes very hard and the problem of **vanishing gradient** problem arises. For building deeper neural network, activation functions like ‘Relu’ activation unit is used.

Another problem common is **Exploding Gradient** more often found in RNN(Recurrent Neural Network). This is when the partial derivate of loss function with respect to weight becomes very large due to activation units with values of gradient greater than 1.

ReLu are the best activation function- the default activation function fro MLP. It is the maximum value of (0,z).This is not differentiable at 0. Solves problem of vanishing gradient. The erivative is either 0 or 1. Hence, efficient for deep learning.

**Bias- Variance Tradeoff**

When the number of layers is more, there are more weights and parameters to train. Also, there is a chance of overfitting. This results in high **Variance**.

With few layers, there are fewer weights. There is a chance of underfitting, which means there is a high **Bias**.

For optimization of the model whenever there is a chance of overfitting, you add regularization to the problem.Both L-2 and L-1 regulation can be used.

L-2 regulation has a variable, lambda, whose value ensures there is lesser overfitting. Whereas, L-1 regularization, introduces sparsity in the model.

**Hyperparameter Tuning**

1. Lambda - Higher the value, lower the variance.
2. Number of layers - More the value, more the variance. In sigmoid, we can run into vanishing gradient problem. Hence, we need to keep it just good enough.
3. Epoch­- Going through the data those many times. Selection of number of epochs can be done viewing the train and test errors.
4. Learning late – If the value is high, the number of epochs will be less. And vice-versa. Proper value must be set to avoid over or under fitting.

In Deep Neural Network, overfitting is mostly encountered and not underfitting.

An interesting concept similar to Random Forest selection process in Neural network is ‘**Dropout’**. It was invented for solving the problem of overfitting, where you randomly select the number of neurons by setting the dropout rate. This can be thought of a random subset of features in random forest.

**Problem Statement**

Train a feedforward neural network to implement a binary to hexadecimal decoder to drive a seven-segment display.

**Code Explanation:**

Imported the necessary libraries

A screenshot of a cell phone

Description automatically generated

A picture containing screenshot

Description automatically generated

The input, X is a 4 digit number binary number and output, y is a 7 digit binary display equivalent of a 7 segment decoder. Both of them are of the type array.

A close up of a logo

Description automatically generated

**This is model 1 creation.**

The model has 1 layer which 3 nodes and ‘**Relu**’ activation function. The output layer has ‘**Softmax**’ activation function. We compile it with Adam optimizer and **binary**\_**crossentrophy** loss, which is the same as log loss, as it has output values which are 0/1.

Although ‘Relu’ function is preferred for deep neural network, for the sake of understanding basics, used it for just 1 layer.

A screenshot of a cell phone

Description automatically generated

**A screenshot of a cell phone

Description automatically generated**

A screenshot of a cell phone

Description automatically generated

The resultant plot of epochs vs accuracy is below. After 300 epochs, the accuracy stabalises to 0.71. However, it sharply falls after 400 epochs.

A screenshot of a cell phone

Description automatically generated

**This is model 2 creation.**

It has 1 layer which 2 nodes and ‘**Relu**’ activation function. The output layer has ‘**Sigmoid’** activation function.

Sigmoid function is good for up to 2 hidden layers. However, as hidden layers increase, updating weights becomes very hard and the problem of **vanishing gradient** problem arises.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

The plot stabilizes after 100 epochs at around 0.69 accuracy.

A screenshot of a cell phone

Description automatically generated

**This is model 3 creation.**

It has 1 layer which 2 nodes and ‘**Sigmoid’** activation function. The output layer has ‘**Sigmoid’** activation function.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

The plot stabilizes after 300 epochs at around 0.698 accuracy.

A screenshot of a cell phone

Description automatically generated

**This is model 4 creation.**

It has 1 layer which 2 nodes and ‘**tanh** activation function. The output layer has ‘**Sigmoid’** activation function.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

The plot stabilizes after 200 epochs at around 0.698 accuracy.

A screenshot of a social media post

Description automatically generated

This is model 5 creation.

It has 2 hidden layers. The first has tanh activation function with 3 nodes and the other has ‘relu’ activation function and 2 nodes.

At the output, we have **sigmoid** activation function.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

This has a relatively higher accuracy due to increasing the number of nodes and hidden layers.

The plot stabilizes after 300 epochs at around 0.75 accuracy.

A screenshot of a cell phone

Description automatically generated

This is model 6 creation.

It has 2 hidden layers. The first has tanh activation function with 3 nodes and the other has ‘relu’ activation function and 3 nodes.

At the output, we have **sigmoid** activation function.

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

This has an even higher accuracy due to increasing the number of nodes in the 2nd hidden layer.

The plot stabilizes after 4000 epochs at around 0.85 accuracy.

A screenshot of a cell phone

Description automatically generated

**References:**

\*2018, Retrieved fromA Beginner’s Guide to Important Topics in AI, Machine Learning, and Deep Learning:

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=2ahUKEwi7ouK\_npjmAhXGslkKHe8ZCJ4QFjACegQIDhAG&url=https%3A%2F%2Fskymind.ai%2Fwiki%2Fneural-network&usg=AOvVaw1ogbTOCJ5cNyPaSuwqfDjp

\*2019, Retrieved fromTensorFlow is in a relationship with Keras — Introducing TF 2.0

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=2ahUKEwjKi9L\_n5jmAhXJs1kKHS4SDE4QFjABegQIDRAF&url=https%3A%2F%2Ftowardsdatascience.com%2Ftensorflow-is-in-a-relationship-with-keras-introducing-tf-2-0-dcf1228f73ae&usg=AOvVaw16cc0gGneB508yP3lk3\_nd

\*Vibhu Singh (Jan 24, 2018) , Retrieved from Introduction to K-NN:

<https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwj7yfST0OXlAhVvhOAKHaNbDI0QMwhGKAIwAg&url=http%3A%2F%2Fstrijov.com%2Fsources%2FdemoDataGen.php&psig=AOvVaw2y8-sTxud9a5-4GVf9kwHT&ust=1573680855153767&ictx=3&uact=3>