**COSC 522 (MACHINE LEARNING) PROJECT 2**

**STUDENT NAME: JOSEPH OCHE AGADA**

**NETID: JAGADA**

**PROBLEM ONE**

**Batch and Online Processing**: In batch processing, the entire training sampling is made to pass through the perceptron or network of perceptron before weight update is done. Batch processing is efficient, require a lot of memory/computational resources and take longer for the network to converge to a local minimum. In batch processing, the size of the mini-batch is the same as the size of the entire training set. Batch processing is an extreme case of mini-batching. On the other hand, in online processing, samples are passed through the network individually and for each sample passed through the network, weight update is done. In online processing, weight update and convergence to the local minimum are faster and lesser memory is required. In online learning, mini-batch size is 1. It is on the second extreme od mini-batching.

**Gradient Descent and Stochastic Gradient Descent**: Gradient descent is the optimization algorithm utilized in batch learning.the algorithm computes the gradient of the loss function with respect to all the training examples in the dataset before updating the model’s parameters. This is as against **stochastic gradient descent,** the algorithm computes the gradient and updates the model’s parameters **one example at a time**, instead of waiting for the full dataset. The properties of gradient descent are similar to that of batch learning described above while the properties of stochastic gradient descent are also similar to those of online learning described above.

**The Differences Between Perceptron and Sigmoid Neurons**: Perceptron are the simplest basic network of neurons with an input payer and an output layer. Perceptron have activation function that is a step function. That is the output of the activation function of a perceptron is either o or 1 and input are also usually binary numbers. On the other hand, sigmoid neurons have activation’s function which is a sigmoid function that output real numbers between 0 and 1.

**Feedforward and Backpropagation:** In feedforward network of neurons, training samples are passed through the network from the input layer, through any hidden layers, to the output layer. The network processes the input data by performing a series of matrix multiplications and applying activation functions at each layer. The final result is the network's prediction. The unique aspect on feedforward network is that in every instance, inputs/weighted inputs can only move forward from one neuron to the next neuron and never backward. On the other hand, Backpropagation is the process used to train the neural network by adjusting its weights and biases. It calculates the error of the prediction made during feedforward and propagates this error backward through the network. This is done to update the model parameters in such a way that the error is minimized in the next iteration

**Why We Introduced Bias**: Bias is usually introduced in neural network to ensure that neurons fire. A neuron is said to fire is it output non-zero output. In perceptron, bias also helps to reduce the threshold value to zero. Additionally, the bias term allows the activation function of a neuron to be shifted left or right, which helps the model capture patterns in the data more effectively.

**Difference Between Training, Validation and Test Sets**. In training neural network and some other machine learning models such as boosting and bagging algorithm, there is a validation dataset in addition to the conventional training and testing dataset. Training dataset is used in training used to train machine learning models to recognize pattern in dataset. Validation set is used during the training process to tune hyperparameters or adjust weight and arrive at optimal hyperparameter that ensures that the right model is selected and that there is not overfitting or underfitting. On the other hand, training dataset is used for model evaluation. That is, to carry out the final assessment of the model. While training and validation set are seen by the model during training, the test set is never seen during training. The ability of the model to perform well on the unseen training dataset is the final confirmation of the performance of the model.

**Reason Why Perceptron (MLP) and Sigmoid Neural Network Employ Different Cost Function:** The **Perceptron** employs a **step activation function** because the cost function is designed to simply minimize the number of misclassified points rather than optimize a smooth function. On the other hand, The **MLP** typically uses a **sigmoid activation function** in its hidden layers, which is differentiable and allows for a smoother optimization process using gradient-based methods like backpropagation

**WHY BACKPROPAGATION IS A FASTER ALGORITHM**

In Chapter 2 of Neural Networks and Deep Learning by Michael A. Nielsen (2015), backpropagation is presented as a crucial algorithm for efficiently training neural networks. The key reason backpropagation is faster than alternative methods is its ability to compute the gradient of the cost function with respect to all the network’s weights and biases simultaneously, using a systematic approach. This efficiency arises from the chain rule of calculus, which backpropagation exploits by propagating errors backward through the network layer by layer.

Without backpropagation, one would need to compute the gradient of the cost function individually for each weight and bias in the network, which would be computationally prohibitive, especially for large neural networks with millions of parameters. In contrast, backpropagation reuses intermediate results (such as activations and errors) as it moves backward through the network, thus reducing redundant calculations and saving computational resources.

Furthermore, backpropagation leverages vectorized operations, allowing it to take full advantage of modern hardware such as GPUs, which excel at parallel computations. By efficiently calculating gradients, backpropagation enables the use of gradient descent (or its variants) to update network parameters quickly, making it feasible to train deep networks on large datasets within a reasonable amount of time. Hence, the speed and practicality of neural network training largely rely on the efficient error propagation provided by the backpropagation algorithm