Assignment 3(A)

1)
In TensorFlow, optimizers are used in conjunction with a CNN model to train the model on a dataset.Optimizers are algorithms used to find the optimal set of parameters for a model during the training process. These algorithms adjust the weights and biases in the model iteratively until they converge on a minimum loss value.

1. SGD (Stochastic Gradient Descent)
2. RMSprop (Root Mean Square Propagation)
3. Adagrad (Adaptive Gradient Algorithm)
4. Adadelta (Adaptive Delta)
5. Adam (Adaptive Moment Estimation)
6. Adamax (Adaptive Moment Estimation with Infinity Norm)
7. Nadam (Nesterov Adaptive Moment Estimation)

2)
Gradient Descent:
Parameters are updated after computing the gradient of the error with respect to the entire training set.Since the entire training data is considered before taking a step in the direction of gradient, therefore it takes a lot of time for making a single update. Updates are made using the rule:

$$\theta=\theta-\eta\cdot\nabla J(\theta)$$

Where θ represents the model parameters, η is the learning rate, and $\nabla J(\theta)$ is the gradient of the cost function.It makes smooth updates in the model parameters.

Stochastic Gradient Descent:
Parameters are updated after computing the gradient of the error with respect to a single training example.Since only a single training example is considered before taking a step in the direction of gradient, we are forced to loop over the training set and thus cannot exploit the speed associated with vectorizing the code.The path to convergence is much noisier, leading to

potential oscillations around the minimum and it may need more iterations to converge because each update is less accurate. The parameters are updated using the rule:

$$\theta=\theta-\eta\cdot\nabla J(\theta;x(i),y(i))$$

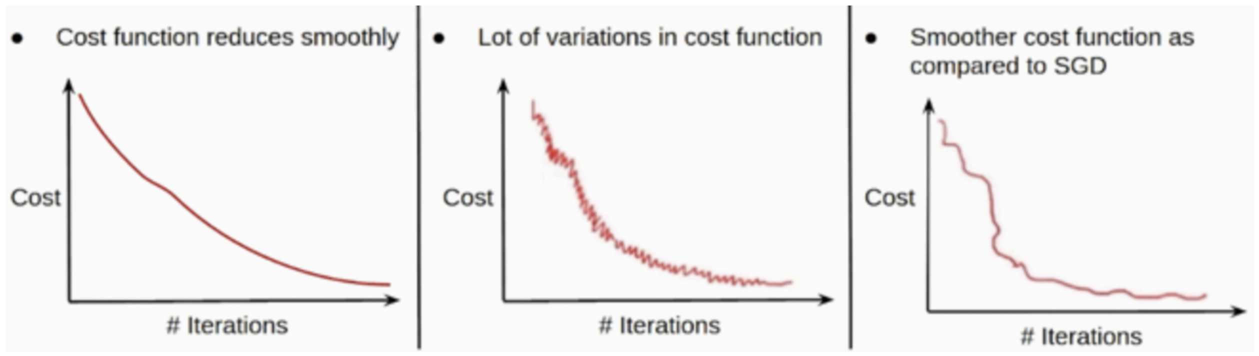
Where (x(i),y(i)) represents a single training example.

Mini Batch Gradient Descent:

Parameters are updated after computing the gradient of the error with respect to a randomly selected subset of the training set.Since a subset of training examples is considered, it can make quick updates in the model parameters and can also exploit the speed associated with vectorizing the code.The parameters are updated using:

$$\theta=\theta-\eta\cdot\nabla J(\theta;X_{\text{mini batch}},Y_{\text{mini batch}})$$

Where (X mini-batch,Y mini-batch) represents the mini-batch of training examples.



:https://www.geeksforgeeks.org/ml-mini-batch-gradient-descent-with-python/

3)

Adaptive Moment Estimation is an algorithm used for optimization in gradient descent.It is a combination of the 'gradient descent with momentum' algorithm and the 'RMSP' algorithm.It uses the squared gradients to scale the learning rate like RMSprop, and it takes advantage of momentum by using the moving average of the gradient instead of the gradient itself, like SGD with momentum. Adam adjusts the learning rates for each parameter individually. It calculates a moving average of the first-order moments (the mean of gradients) and the second-order moments (the uncentered variance of gradients) to scale the learning rates adaptively. Instead of using the same learning rate (the size of the step taken to adjust parameters) for all parameters, Adam adjusts the learning rate individually for each parameter based on the averages it keeps track of.This makes it well-suited for problems with sparse gradients or noisy data.

4)

While Rmsprop adjusts the learning rate for each parameter by dividing the learning rate by an exponentially decaying average of squared gradients , adam computes adaptive learning rates for each parameter, using estimates of first and second moments of the gradients.While both Rmsprop and Adam are adaptive learning rate optimization algorithms, Adam extends Rmsprop by incorporating first moment estimates and bias correction, generally resulting in better performance and faster convergence in practice.Because Adam includes a bias correction step to account for the fact that the moving averages are initialized to zero and therefore biased towards zero initially,the initial steps of Adam more accurate.RMSProp does not include a bias correction.

5)
Selecting an optimizer depends on the specific characteristics of the problem at hand .

- SGD: Good for large-scale data and simple models where computational efficiency is paramount.
- SGD with Momentum/NAG: Useful for models with oscillatory gradients and scenarios requiring faster convergence.

- AdaGrad: Best for sparse data, where infrequent features need larger updates.
- RMSprop: Suitable for RNNs and scenarios where adaptive learning rates are beneficial.
- Adam: Often the default choice due to its robustness and general applicability across various models and datasets.
- AdaMax/Nadam: Variants of Adam, useful in specific cases where Adam might struggle. If Adam converges too slowly on your problem, switching to Nadam might speed up the training process.
- AdaDelta: Useful when learning rates are difficult to set manually, providing adaptive learning without requiring initial learning rates.

6)

A model is said to overfit if it performs very well on the training data but generalizes poorly resulting in low accuracy on the validation set.The curve goes through almost every data point, capturing all the noise and variability. In this case, the model has high variance (performance fluctuates greatly with different data). Underfitting occurs when a model is too simple to capture the underlying patterns in the data, leading to poor performance on both the training and validation/test sets. The curve is too simple, and does not capture the underlying trend in the data. In this case, the model has high bias.

7)

Vanishing gradients occur when the gradients of the loss function with respect to the model parameters become very small. This problem is most prominent when using certain activation functions, such as the sigmoid or tanh.As a result, the weights in the earlier layers of the network do not get updated effectively, slowing down or completely stopping the learning process.During back propagation if the gradient becomes small (<1), the product can eventually tend to zero as it propagates through layers.

Exploding gradients occur when the gradients of the loss function with respect to the model parameters become very large. Backpropagation involves calculating gradients using the chain rule, and as the numbers are multiplied the product can become very large.This results in the weights updating in a very unstable manner, causing the model to diverge.

8)

Batch Normalization normalizes the inputs of each layer across the mini-batch. For a given mini-batch, compute the mean and variance for each feature across the batch.Subtract the mean and divide by the standard deviation to normalize the batch.It acts as a form of regularization and is typically applied before activation function.Normalizes across the batch dimension (i.e., across multiple training examples for each feature) and is particularly effective in CNNs.

Layer Normalization normalizes the inputs of each layer across the features for each individual training example. For a given training example, the mean and variance for each feature across the layer is computed.Subtract the mean and divide by the standard deviation to normalize the features.Commonly used in RNNs, transformers, and situations where batch normalization is less effective. And is typically applied after activation functions.

9)

Regularization techniques in machine learning are strategies used to prevent overfitting and encourage the model to be simpler and more generalizable, improving its performance on unseen data. It prevents the model from memorizing the training data(including all the noise and variance) as this would lead to low accuracy on the validation data.

L1 regularization(lasso):Loss=Original Loss+$\lambda\sum|wi|$ (w is the parameter)

L2 regularization(ridge):Loss=Original Loss+λ∑wi**2

Dropout : It is a technique used primarily in neural networks where, during each training iteration, a random subset of neurons is "dropped out" or set to zero.

Data augmentation: It involves increasing the diversity of the training data without collecting new data by applying random transformations like rotation, scaling, cropping, and flipping to images; or adding noise to data.This is to increase the ability of the model to generalize and improve accuracy on unseen data.

10)

A dropout layer is a regularization technique used in neural networks to prevent overfitting. It works by randomly setting a fraction of the input units to zero at each update during training time. This is done to ensure that the model does not memorize the training data but instead picks up patterns that would help it make accurate predictions on the test data.At each training step, for each neuron in the layer, there is a certain probability (dropout rate) that the neuron will be set to zero.For example, if the dropout rate p is 0.5, then each neuron is dropped with a 50% probability. The neurons that remain are scaled by 2 (i.e.,1/1-0.5). Dropout is applied only during training, not while making predictions.

11)

L1 regularization(lasso): Loss=Original Loss+λ∑∣wi∣ (w is the parameter)
It adds the absolute value of the magnitude of coefficients as a penalty term to the loss function. This encourages sparsity in the model parameters, meaning it drives some coefficients to be exactly zero, effectively performing feature selection and results in simpler models.

L2 regularization(ridge):Loss=Original Loss+λ∑wi**2

It adds the squared magnitude of coefficients as a penalty term to the loss function. This encourages the model to distribute the weights more evenly and discourages large coefficients.Larger values of λ result in smaller weights.This expects all features to contribute to output but reduces overfitting by preventing large weights.

12)

Validation accuracy provides an estimate of how well the model will perform on unseen data. It helps in assessing the generalization ability of the model.Validation accuracy is calculated by comparing the predictions of the model on the validation dataset to the true labels and computing the percentage of correct predictions.

Accuracy=(Number of Correct Predictions/Total Number of Predictions)×100%

Accuracy helps detect overfitting, a low validation accuracy and a high test accuracy implies that the model has high variance while a low test accuracy and low validation accuracy implies the model has high bias. A model with high validation generalizes well on unseen data.

13)

Data augmentation is a technique used to artificially increase the size and diversity of a dataset by applying various transformations to the existing data samples. It is commonly used in tasks like image classification, object detection, and natural language processing. It is done to prevent overfitting and introduce more training examples to help the model generalize better.

Advantages:

By exposing the model to a wider range of scenarios and variations, data augmentation helps improve its ability to generalize unseen data.Augmenting the dataset with diverse examples makes the model more familiar to variations and noise present in real-world data. Data augmentation can often lead to improved accuracy on both the training and test sets.

Disadvantages:

If data augmentation is not used carefully, it can introduce noise into the training data. This can lead to decreased performance on the test set.The quality of augmented data may not always be as high as that of the original data. Data augmentation can increase the computational cost of training a model.Data augmentation can increase the computational cost of training a model, because it has to be trained on a large number of examples.

14)

Transfer learning, used in machine learning, is the reuse of a pre-trained model on a new problem. Instead of training a model from scratch on the new task, transfer learning leverages knowledge gained from the source task to improve learning on the target task and is particularly useful when the target dataset is small or when training a model from scratch would be computationally expensive. Hence a solid machine learning model can be built with comparatively little training data because the model is already pre-trained.

Image Classification:VGG (Visual Geometry Group),ResNet (Residual Network),Inception,MobileNet,EfficientNet

Object Detection:Faster R-CNN,YOLO (You Only Look Once),SSD (Single Shot MultiBox Detector),RetinaNet

Natural Language Processing:BERT (Bidirectional Encoder Representations from Transformers),GPT (Generative Pre-trained Transformer),Transformer-XL ,RoBERTa (Robustly optimized BERT approach)

15)

The bias-variance tradeoff is a fundamental concept in supervised learning that describes the relationship between model complexity and its ability to generalize to new, unseen data. A model with high bias tries to generalize complex real world data with a simple model. Such a model has low train as well as test accuracy. A model is said to have high variance if it is overly complex and fits the training data very well but fails to generalize and accurately predict unseen data.Such a model has high test accuracy and a low validation accuracy. Finding the right balance between bias and variance is important for a machine learning algorithm as increasing one often leads to a decrease in the other.

16)

It is better to train a single neural network with one output neuron for each disease.Training separate models for each disease may result in redundant learning, as features relevant to one disease may also be relevant to others. That approach is also computationally costly and a single network would be more complex but is comparatively less costly to train and maintain.A single model allows for better handling of class imbalances, which might occur if some disease categories are more prevalent in the dataset than others.It can learn shared representations from the data,capturing common patterns across different diseases.Regularization techniques can be applied to the single model to prevent overfitting and improve generalization, preventing overly complex models.

17)

a)There will be 3 weights and bias matrices.

Between input layer and 1st hidden layer: weights-(n1 x n0), bias-(n1x1)

Between 1st and 2nd hidden layer: weights-(n2 x n1), bias-(n2x1)

Between 2nd hidden layer and output layer:weights-(n3 x n2), bias-(n3x1)

b)i)hidden layers: ReLU activation function is used for its simplicity and effectiveness in combating the vanishing gradient problem. (max(0,x))

ii)output layer: Sigmoid is used. It returns the probability of the input data belonging to a particular class. So the output >0.5 can be taken as the prediction.

c)i)hidden layers: ReLU activation function is used for its simplicity and effectiveness in combating the vanishing gradient problem. (max(0,x))

ii)output layer: Softmax is used. It returns the probability of the input data belonging to a particular class and ensures that the sum is 1.So the class with highest probability can be taken as the prediction.

d)i)hidden layers: ReLU activation function is used for its simplicity and effectiveness in combating the vanishing gradient problem. (max(0,x))

ii)output layer: Linear is used. It is equivalent to having no activation function in the dense layer(setting it to none)