

**Module 22**

**Deep Neural Networks 1**

* [Video Transcripts](https://student.emeritus.org/courses/4765/files/3621221?wrap=1)
* [Download Video Transcripts](https://student.emeritus.org/courses/4765/files/3621221/download?download_frd=1)
* [Quick Reference Guide](https://student.emeritus.org/courses/4765/files/3621222?wrap=1)
* [Download Quick Reference Guide](https://student.emeritus.org/courses/4765/files/3621222/download?download_frd=1)

A deep neural network is an artificial neural network (ANN) with multiple layers between the inputs and outputs. Neural networks have many types, but they all have the same components: neurons, synapses, weights, biases, and functions. These networks process data in complex ways by employing sophisticated math modeling.

Computer vision, speech processing, machine translation, and reinforcement learning have all been revolutionized by deep learning neural networks. Thus, deep neural networks have become indispensable for any data scientist's toolkit.

**Glossary**

**Batch Size**

The number of samples that will be passed through to the network at one time

**Dropout Rate**

A technique that involves ignoring randomly selected neurons during training

**Epoch**

Consists of an entire dataset being passed forward and backward through the neural network only once

**Keras**

A high-level API of TensorFlow; an open-source library that provides an artificial neural network interface for Python

**KerasTuner**

A library for tuning the hyperparameters of a neural network that helps pick optimal hyperparameters in your neural network implement in TensorFlow

**Layer**

A general term that applies to a collection of 'nodes' operating together at a specific depth within a neural network

**Node (or Neuron)**

A place where computation happens, loosely patterned on a neuron in the human brain

**Rectified Linear Unit (ReLU)**

The number of samples that will be passed through to the network at one time

**Regularization Rate**

A number of techniques that reduce the complexity of a neural network model during training to reduce overfitting

**Softmax Layer**

The activation function in the output layer of neural network models that predicts a multinomial probability distribution

**Tanh**

A zero-centered activation function having a range between -1 to 1

**Notes:**

pip install **tensorflow**

pip install -U --trusted-host pypi.org --trusted-host files.pythonhosted.org **tensorflow**

**Best Practices**

While an organization may set its own best practices for developing deep learning models, here is a list of some important ones to keep in mind:

1. **Batch normalization** is a technique for standardizing inputs into a network, applied to either the activations of a previous layer or direct inputs
2. A **residual connection**is a skip-connection in which the residual functions are learned with reference to the layer inputs rather than learning unreferenced functions
3. **Hyperparameter optimization** algorithms find the hyperparameter tuple that minimizes a predefined loss function based on independent data
4. **Ensemble learning** reduces the variance of predictions and generalization error by combining predictions from multiple neural network models

**Popular Hyperparameters**

In machine learning, hyperparameters determine the model's training process and topology. During the training process, these variables remain constant and directly affect the performance of your ML program. There are two types of hyperparameters:

* Model hyperparameters that affect the model selection, such as the number and width of hidden layers
* Algorithm hyperparameters that affect the quality and speed of the learning algorithm, such as the number of neighbors for a K-nearest neighbors (KNN) classifier and the learning rate for stochastic gradient descent (SGD)

**Dropout Rate**

The technique of dropout involves ignoring randomly selected neurons during training. Instead, they are randomly ‘dropped out.’ The result is that their contribution to downstream activation is removed in the forward propagation, and their weights are not updated in the backpropagation.

**Regularization Rate**

The term 'regularization' refers to a number of techniques that reduce the complexity of a neural network model during training to reduce overfitting. Two prevalent and efficient regularization techniques are L1 and L2.

**L1 regularization** penalizes the absolute magnitude of the coefficients. To put it differently, it limits the coefficient size. As a result, L1 can produce sparse models (with few coefficients). Some coefficients can become zero and can be eliminated. Lasso regression uses this technique.

In **L2 regularization**, the square of the magnitude of the coefficients is added to the penalty. As a result, the L2 model does not yield sparse models, and all coefficients are shrunk by the same factor (none are eliminated). Regression and support vector machines use this technique.

**Module Issues:**

Codio 22.1 Problem 5:

**Quizes:**

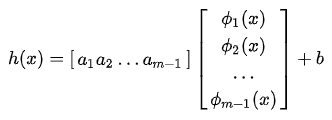
The term HOG stands for Histogram Optional Grade. : False

*You are correct! The answer “*False*” is correct because HOG stands for Histogram of Oriented Gradients.*

What is the standard benchmark for comparison of preprocessing techniques and algorithms for image classification? : ImageNet

*You are correct! The answer “I*mageNet*” is correct because the most famous benchmark dataset, which consists of a massive database of images, is known as ImageNet.*

The vector notation for a linear regression model equation is shown below:



**h(x)=[**

|  |
| --- |
| **a**  1  **a**  2  **…**  **a**  m  −  1 |

]

**[**

|  |
| --- |
|  |
| **ϕ**  1  **(x)** |
| **ϕ**  2  **(x)** |
| ... |

]+b : True

*You are correct! The answer “*True*” is correct because the vector notation for linear regression equation can be defined as a weight vector capital A, a feature vector phi(x), and an offset b represented as h(x) = AT phi(x) + b.*

Which of the following is not an activation function for neural networks? : TanU: Φ(x)=max(x,0)

*You are correct! The answer “*TanU: Φ(x)=max(x,0)*” is correct because this is not an activation function for neural networks.*

Adding a sigmoid activation function to the output node transforms the regression model into a (blank). : Binary classification model

*You are correct! The answer “*Binary classification model*” is correct because by adding a sigmoid activation function to the output node, the regression model is transformed into a binary classification model.*

The training problem for neural networks is to find all of the “A” and “b” coefficients, such that the loss function is maximized. : False

*You are correct! The answer “*False*” is correct because the training problem for neural networks is to find all of the A and b coefficients, such that the loss function is minimized.*

What is the import statement in Python to import the layers library? : from tensorflow.keras import layers

*You are correct! The answer “*from tensorflow.keras import layers*” is correct because this is the correct statement to import the layers library.*

Consider the given Python code:

model = keras.Sequential([

    layers.Dense(3, activation="relu"),

    layers.Dense(4, activiation="relu"),

    layers.Dense(2, activation="relu"),

    layers.Dense(1, activation="sigmoid")

])

How many neurons does the third layer have? : 2

*You are correct! The answer “2” is correct because moving sequentially from top to bottom, the third layer has the first constructor as “2”, which gives the number of neurons in the third layer.*

From the given Python code:

model = keras.Sequential([

    layers.Dense(3, activation="relu"),

    layers.Dense(4, activation="relu"),

    layers.Dense(2, activation="relu"),

    layers.Dense(2, activation="sigmoid")

])

Is the neural network a binary classification model? : True

*You are correct! The answer “*True*” is correct because the outer layer of the neural network built from the Python code has the activation function as sigmoid, which makes it a binary classification model.*

Which of the following Python statements set the optimization function as “rmsprop”, the loss function as “binary\_crossentropy”, and the metric as “accuracy”? : model.compile(optimizer="rmsprop", loss="binary\_crossentropy", metrics=["accuracy"])

*You are correct! The answer "*model.compile(optimizer="rmsprop", loss="binary\_crossentropy", metrics=["accuracy"])*” is correct because this is the right syntax to manually specify the optimization function, the loss function, and metrics.*

Which of the following is not a binary classification problem? : Orange, apple, or pear

*You are correct! The answer “*Orange, apple, or pear*” is correct in that it is not an example of a binary classification problem. A binary classification problem is a problem with two classes whereas this statement has more than two.*

Given a multiclass problem with four classes, how many nodes should be in the softmax layer? : Four nodes

*You are correct! The answer “*Four nodes*” is correct because in neural networks for multiclass problems the nodes in the output layer are equal to the number of classes.*

According to the softmax function, if “xi <xj” then “yi <yj”. : True

*You are correct! The answer “*True*” is correct because the softmax function preserves the order that if xi <xj then yi <yj.*

The softmax function fits perfectly with one-hot encoding. : True

*You are correct! The answer “*True*” is correct because it produces values between 0 and 1 that can also be interpreted as a certainty value for each of the classes.*

Consider the given Python code:

model = Sequential([

    Dense(5, input\_dim=2, activation='relu'),

    Dense(2, actication="softmax") ])

model.compile(loss="categorical\_crossentropy", metrics=["accuracy"])

history = model.fit(Xy\_ohe, epochs=500)

Is it correct for building a multiclass neural network for Iris data? : False

*You are correct! The answer “*False*” is correct because in the output layer, the nodes defined are two, whereas the Iris data has three output classes.*

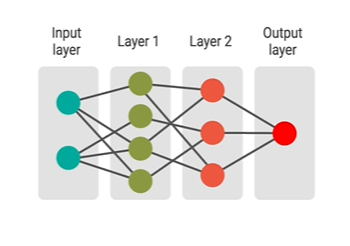
What is the loss function for building a neural network that uses integer encoding? : sparse\_categorical\_crossentropy

*You are correct! The answer “*sparse\_categorical\_crossentropy*” is correct because this is the loss function used for integer encoding in neural networks.*

The hyperparameter that refers to the number of neurons in each layer is known as the depth. : False

*You are correct! The answer “*False*” is correct because the number of neurons in each layer is known as the width.*

Consider the following neural network:



What is the depth of this neural network? : 3

*You are correct! The answer “*3*” is correct because the depth is the number of layers, which include 2 hidden layers and one output layer, which equals 3.*

What is the tool used for selection of hyperparameters for neural networks? : Cross-validation

*You are correct! The answer “C*ross-validation*” is correct because it is the most important tool for deciding on hyper parameters in neural networks.*

When using cross-validation on neural networks, the model that has the (blank) cross-entropy loss on the development set is chosen. : Lowest

*You are correct! The answer “*Lowest*” is correct because in neural networks, the model that has lowest cross-entropy loss on the development set is chosen.*

Instead of manually generating a table of validation set loss values, Keras has a tool called (blank) that allows you to search a pre-specified set of hyperparameters. : KerasTuner

*You are correct! The answer “*KerasTuner*” is correct because this is the tool used to search a pre-specified set of hyperparameters.*

What is the algorithm that computes the gradients in small steps called? : Mini-batch

*You are correct! The answer “*Mini-batch*” is correct because the algorithm computing the gradients in small steps is called a mini-batch algorithm.*

Each epoch of stochastic gradient descent should consider a straight ordering of points. : False

*You are correct! The answer “*False*” is correct because each epoch of stochastic gradient descent should consider a random ordering of points.*

What is batch losses? : An array that stores loss values after every batch

*You are correct! The answer “*An array that stores loss values after every batch*” is correct because to track the loss at every batch, an array is created to store loss values known as batch losses.*

What is the constructor used in the fit() function in Python to declare the batch size? : batch\_size

*You are correct! The answer “*batch\_size*” is correct because this constructor is used in the Python function to declare batch size.*