Neural Networks 1

- Due No due date
- Points 140
- Questions 14
- Time Limit None
- Allowed Attempts Unlimited

Instructions

You can have multiple attempt on this quiz to improve your score. Only the highest score will be recorded.

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	15 minutes	130 out of 140

Score for this attempt: 130 out of 140

Submitted Nov 2 at 7:13pm

This attempt took 15 minutes.

Question 1

10 / 10 pts

Which of these are terms used to refer to components of an artificial neural network?

- [ ] axon

Correct!

- [x] layers

Yes, a layer is a grouping of neurons in a neural network

Correct!

- [x] activation function

Yes, an activation is the number calculated by a neuron (and “activations” in the figure above is a vector that is output by a layer that contains multiple neurons)

Correct!

- [x] neurons

Yes, a neuron is a part of a neural network

Question 2

10 / 10 pts

True/False? Neural networks take inspiration from, but do not very accurately mimic, how neurons in a biological brain learn.

Correct!

True

Artificial neural networks use a very simplified mathematical model of what a biological neuron does.

False

Question 3

10 / 10 pts

For a neural network, what is the expression for calculating the activation of the third neuron in layer 2?

$a_3^{[2]} = g(\vec{w}_2^{[3]} \cdot \vec{a}^{[2]} + b_2^{[3]})$

$a_3^{[2]} = g(\vec{w}_2^{[3]} \cdot \vec{a}^{[1]} + b_2^{[3]})$

Correct!

$a_3^{[2]} = g(\vec{w}_3^{[2]} \cdot \vec{a}^{[1]} + b_3^{[2]})$

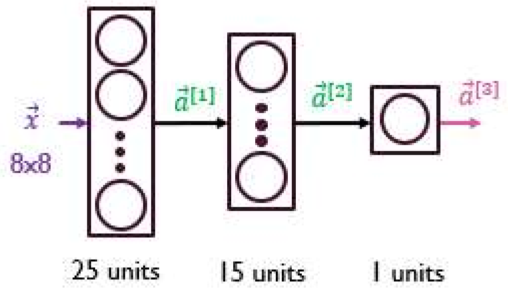
Yes! The superscript [2] refers to layer 2. The subscript 3 refers to the neuron in that layer. The input to layer 2 is the activation vector from layer 1.

$a_3^{[2]} = g(\vec{w}_3^{[2]} \cdot \vec{a}^{[2]} + b_3^{[2]})$

Question 4

10 / 10 pts

For the handwriting recognition task discussed in lecture, what is the output $a_1^{[3]}$?



A vector of several numbers that take values between 0 and 1

A number that is either exactly 0 or 1, comprising the network’s prediction

A vector of several numbers, each of which is either exactly 0 or 1

Correct!

The estimated probability that the input image is of a number 1, a number that ranges from 0 to 1.

Yes! The neural network outputs a single number between 0 and 1.

Question 5

10 / 10 pts

How many layers the following neural network has?

model = Sequential([Dense(units=25, activation="sigmoid"),Dense(units=15, activation="sigmoid"),Dense(units=10, activation="sigmoid"),Dense(units=1, activation="sigmoid")])

25

10

- 1

Correct!

- 4

Question 6

0 / 10 pts

How do you define the second layer of a neural network that has 4 neurons and a sigmoid activation?

- Dense(units=4)

You Answered

- Dense(layer=2, units=4, activation = 'sigmoid')

Correct Answer

- Dense(units=4, activation='sigmoid')
- Dense(units=[4], activation=['sigmoid'])

Question 7

10 / 10 pts

For which type of task would you use the binary cross entropy loss function?

`

model.compile(loss=BinaryCrossentropy())

`

Correct!

- binary classification (classification with exactly 2 classes)

Yes! Binary cross entropy, which we've also referred to as logistic loss, is used for classifying between two classes (two categories).

- A classification task that has 3 or more classes (categories)
- BinaryCrossentropy() should not be used for any task.
- regression tasks (tasks that predict a number)

Question 8

10 / 10 pts

Here is code that you saw in the lecture:

```
```
model = Sequential([
    Dense(units=25, activation='sigmoid’),
    Dense(units=15, activation='sigmoid’),
    Dense(units=1, activation='sigmoid’)
    ])
```

model.compile(loss=BinaryCrossentropy())
```

model.fit(X,y,epochs=100)

`

Which line of code updates the network parameters in order to reduce the cost?

Correct!

- model.fit(X,y,epochs=100)

Yes! The third step of model training is to train the model on data in order to minimize the loss (and the cost)

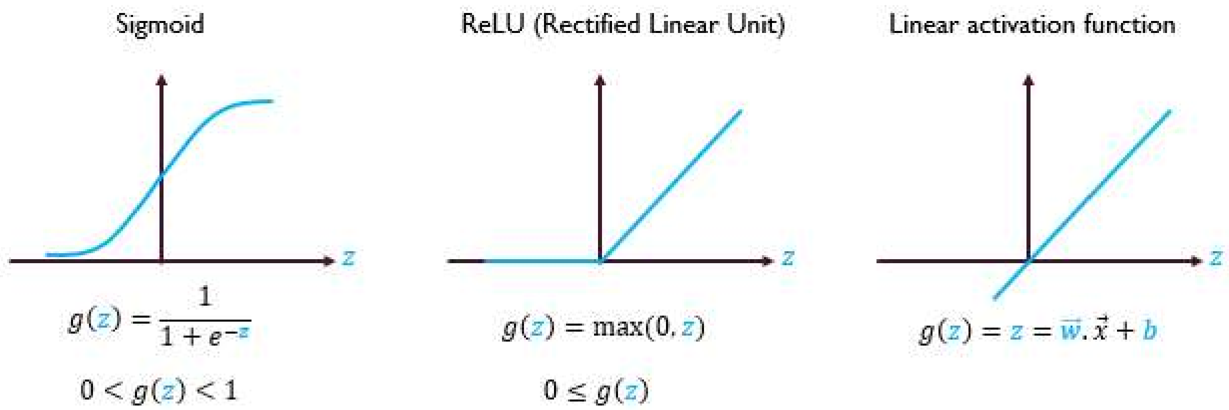
- None of the above -- this code does not update the network parameters.
- model.compile(loss=BinaryCrossentropy())
- model = Sequential([...])

Question 9

10 / 10 pts

Which of the following activation functions is the most common choice for the hidden layers of a neural network?

$$z = \vec{w}.\vec{x} + b$$



- Linear
- Sigmoid
- Most hidden layers do not use any activation function

Correct!

- ReLU (rectified linear unit)

Yes! A ReLU is most often used because it is faster to train compared to the sigmoid. This is because the ReLU is only flat on one side (the left side) whereas the sigmoid goes flat (horizontal, slope approaching zero) on both sides of the curve.

Question 10

10 / 10 pts

For the task of predicting housing prices, which activation functions could you choose for the output layer? Choose the 2 options that apply.

Correct!

- [x] linear

Yes! A linear activation function can be used for a regression task where the output can be both negative and positive, but it's also possible to use it for a task where the output is 0 or greater (like with house prices).

Correct!

- [x] ReLU

Yes! ReLU outputs values 0 or greater, and housing prices are positive values.

- [ ] Sigmoid

Question 11

10 / 10 pts

A neural network with many layers but no activation function (in the hidden layers) is not effective; that’s why we should instead use the linear activation function in every hidden layer.

○ True

Correct!

◉ False

Yes! A neural network with many layers but no activation function is not effective. A linear activation is the same as "no activation function".

Question 12

10 / 10 pts

- For K different category each category we will define a different z parameter.

$$z_j = \vec{w}_j . \vec{x} + b_j \qquad j = 1, \ldots, K$$

- Then the probability of the feature belonging to each category can be modeled as.

$$a_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} = \frac{e^{z_j}}{e^{z_1} + e^{z_2} + \cdots + e^{z_K}} = \mathrm{P}(y = j|\vec{x})$$

For a multiclass classification task that has 4 possible outputs, the sum of all the activations adds up to 1. For a multiclass classification task that has 3 possible outputs, the sum of all the activations should add up to ….

○ Less than 1

○ More than 1

Correct!

◉ 1

Yes! The sum of all the softmax activations should add up to 1. One way to see this is that if $e^{z_1} = 10, e^{z_2} = 20, e^{z_3} = 30$, then the sum of $a_1 + a_2 + a_3$ is equal to $\frac{e^{z_1} + e^{z_2} + e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}$ which is 1.

- It will vary, depending on the input x.

Question 13

10 / 10 pts

- For Logistic Regression:

$$a_1 = g(z) = \frac{1}{1+e^{-z}} = \mathrm{P}(y=1|\vec{x})$$
$$a_2 = 1 - a_1 \qquad = \mathrm{P}(y=0|\vec{x})$$

- We defined the loss as follow:

$$Loss = \begin{cases} -\log(a_1) & if \quad y=1 \\ -\log(1-a_1) & if \quad y=0 \end{cases}$$

$$Loss = \begin{cases} -\log(a_1) & if \quad y=1 \\ -\log(a_2) & if \quad y=0 \end{cases}$$

- For softmax regression:

$$a_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} = \mathrm{P}(y=j|\vec{x}) \qquad j = 1, \dots, K$$

- Similarly, the loss will be defined as:

$$Loss = \begin{cases} -\log(a_1) & if \quad y=1 \\ -\log(a_2) & if \quad y=2 \\ & \vdots \\ -\log(a_K) & if \quad y=K \end{cases}$$

- This is called Crossentropy loss.

For multiclass classification, the cross entropy loss is used for training the model. If there are 4 possible classes for the output, and for a particular training example, the true class of the example is class 3 (y=3), then what does the cross entropy loss simplify to? [Hint: This loss should get smaller when a_3 gets larger.]

Correct!

- $-log(a_3)$

Correct. When the true label is 3, then the cross entropy loss for that training example is just the negative of the log of the activation for the third neuron of the softmax. All other terms of the cross entropy loss equation $(-log(a_1), -log(a_2), and - log(a_4))$ are ignored

- $\frac{-log(a_1)+-log(a_2)+-log(a_3)+-log(a_4)}{4}$
- $\frac{z_3}{z_1+z_2+z_3+z_4}$
- z_3

Question 14

10 / 10 pts

model	`import tensorflow as tf`
	`from tensorflow.keras import Sequential`
	`from tensorflow.keras.layers import Dense`
	`model = Sequential([`
	`  Dense(units=25, activation='relu')`
	`  Dense(units=15, activation='relu')`
	`  Dense(units=10, activation='linear') )]`

loss	`from tensorflow.keras.losses import`
	`  SparseCategoricalCrossentropy`
	`model.compile(...,loss=SparseCategoricalCrossentropy(from_logits=True) )`

fit	`model.fit(X,Y,epochs=100)`
predict	`logits = model(X)` `f_x = tf.nn.softmax(logits)`

For multiclass classification, the recommended way to implement softmax regression is to set from_logits=True in the loss function, and also to define the model's output layer with…

Correct!

- a 'linear' activation

Yes! Set the output as linear, because the loss function handles the calculation of the softmax with a more numerically stable method.

- a 'softmax' activation

Quiz Score: 130 out of 140