**The sequential model in Keras**

In chapter 3, we used components of the keras API in tensorflow to define a neural network, but we stopped short of using its full capabilities to streamline model definition and training. In this exercise, you will use the keras sequential model API to define a neural network that can be used to classify images of sign language letters. You will also use the .summary()method to print the model's architecture, including the shape and number of parameters associated with each layer.

Note that the images were reshaped from (28, 28) to (784,), so that they could be used as inputs to a dense layer. Additionally, note that keras has been imported from tensorflow for you.

**Instructions**

**100 XP**

* Define a keras sequential model named model.
* Set the first layer to be Dense() and to have 16 nodes and a reluactivation.
* Define the second layer to be Dense() and to have 8 nodes and a reluactivation.
* Set the output layer to have 4 nodes and use a softmax activation function.

# Define a Keras sequential model

model = keras.Sequential()

# Define the first dense layer

model.add(keras.layers.Dense(16, activation='relu', input\_shape=(784,)))

# Define the second dense layer

model.add(keras.layers.Dense(8, activation='relu'))

# Define the output layer

model.add(keras.layers.Dense(4, activation='softmax'))

# Print the model architecture

print(model.summary())

Excellent work! Notice that we've defined a model, but we haven't compiled it. The compilation step in keras allows us to set the optimizer, loss function, and other useful training parameters in a single line of code. Furthermore, the .summary() method allows us to view the model's architecture.