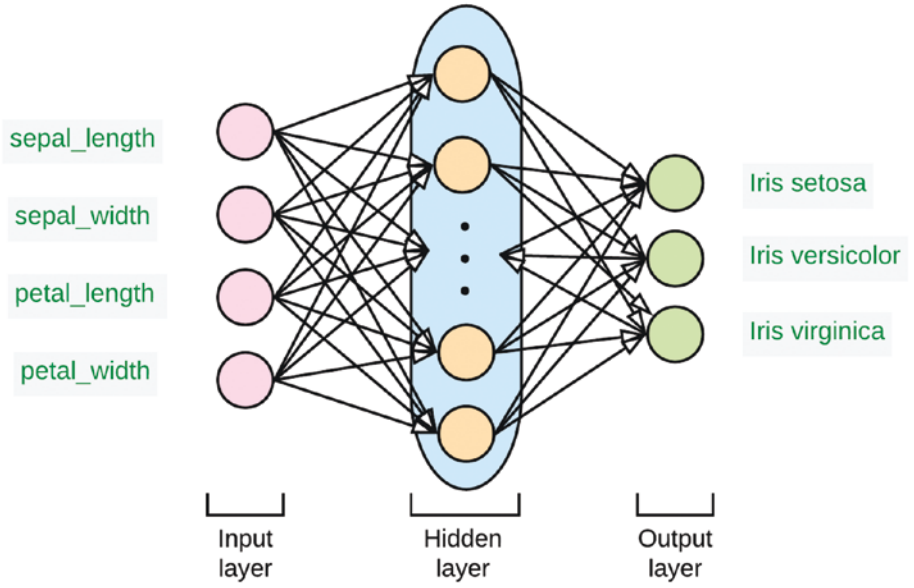

Figure 30-11. *Iris dataset – neural network architecture*

Using the Keras Sequential API

This code segment will construct a neural network model with the Sequential API using the method **'tf.keras.Sequential()'** to stack layers on each other. The model creates a hidden layer with 32 neurons and an output layer with 3 output units because the Iris target contains 3 classes.

```
!pip install -q tensorflow==2.0.0-beta0
```

```
# import packages
import tensorflow as tf
import pandas as pd
from sklearn.preprocessing import OneHotEncoder
```

```
# dataset url
train_data_url = "https://storage.googleapis.com/download.tensorflow.org/
data/iris_training.csv"
test_data_url = "https://storage.googleapis.com/download.tensorflow.org/
data/iris_test.csv"
```

```
# define column names
columns = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']

# download and load the csv files
train_data = pd.read_csv(tf.keras.utils.get_file('iris_train.csv', train_
data_url),
                                skiprows=1, header=None, names=columns)

test_data = pd.read_csv(tf.keras.utils.get_file('iris_test.csv', test_data_url),
                                skiprows=1, header=None, names=columns)

# separate the features and targets
(X_train, y_train) = (train_data.iloc[:,0:-1], train_data.iloc[:,-1])
(X_test, y_test) = (test_data.iloc[:,0:-1], test_data.iloc[:,-1])

# apply one-hot encoding to targets
y_train=tf.keras.utils.to_categorical(y_train)
y_test=tf.keras.utils.to_categorical(y_test)

# create the sequential model
def model_fn():
    model = tf.keras.Sequential()
    # Add a densely-connected layer with 32 units to the model:
    model.add(tf.keras.layers.Dense(32, activation='sigmoid', input_dim=4))
    # Add a softmax layer with 3 output units:
    model.add(tf.keras.layers.Dense(3, activation='softmax'))

    # compile the model
    model.compile(optimizer=tf.keras.optimizers.SGD(),
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
    return model

# parameters
batch_size=50
```

```
# use tf.data to batch and shuffle the dataset
train_ds = tf.data.Dataset.from_tensor_slices(
    (X_train.values, y_train)).shuffle(len(X_train)).repeat().batch(batch_size)
test_ds = tf.data.Dataset.from_tensor_slices((X_test.values, y_test)).
batch(batch_size)

# build train model
model = model_fn()

# print train model summary
model.summary()

# train the model
history = model.fit(train_ds,steps_per_epoch=5000)

# evaluate the model
score = model.evaluate(test_ds)

print('Test loss: {:.2f} \nTest accuracy: {:.2f}%'.format(score[0],
score[1]*100))

'Output':
Test loss: 0.22
Test accuracy: 96.67%
```

Using the Keras Functional API

The general code pattern for the Functional API is structurally the same as the Sequential version. The only change here is in how the network model is constructed. We also demonstrated the Keras feature for printing the graph of the model in this example. The output is illustrated in Figure 30-12.

```
!pip install -q tensorflow==2.0.0-beta0

# import packages
import tensorflow as tf
import pandas as pd
from sklearn.preprocessing import OneHotEncoder
```