7. Deep Learning – Save the model

Contents

7. Deep Learning – Save the model

1. Save the model

- ✓ Based on requirement we can save the model
- ✓ Generally, in deep learning process,
  - model will be saved into json file
  - model weights will be saved into HDF5(Hierarchical Data Format)
- ✓ HDF is more convenient for storing large arrays of real values, as we have in the weights of neural networks.

Program Name: Save the model
Input file: demo1.py
pima-indians-diabetes.csv

```
# importing required libraries
from numpy import loadtxt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from keras.models import model_from_json

# load the dataset
dataset = loadtxt('pima-indians-diabetes.csv', delimiter = ',')

# split into input (X) and output (y) variables
X = dataset[:, 0:8]
y = dataset[:, 8]

# define the keras model

model = Sequential()

model.add(Dense(12, input_shape = (8, ), activation = 'relu'))
model.add(Dense(8, activation = 'relu'))
model.add(Dense(1, activation = 'sigmoid'))

# compile the keras model
model.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics =
['accuracy'])

model.fit(X, y, epochs = 150, batch_size = 10)

model_json = model.to_json()

# saving the model
with open("model.json", "w") as json_file:
        json_file.write(model_json)

# saving the model weights
model.save_weights("model.weights.h5")

print("Saved model to disk")

print("Done")
```

Output

```
Epoch 147/150
77/77 [==============================] - 0s 1ms/step - loss: 0.5480 - accuracy: 0.7201
Epoch 148/150
77/77 [==============================] - 0s 1ms/step - loss: 0.5500 - accuracy: 0.7227
Epoch 149/150
77/77 [==============================] - 0s 1ms/step - loss: 0.5493 - accuracy: 0.7292
Epoch 150/150
77/77 [==============================] - 0s 1ms/step - loss: 0.5509 - accuracy: 0.7279
Saved model to disk
Done
```

model.json

✓ The json format of the model looks like the following:


```
{
 "class_name": "Sequential",
 "config": {
  "name": "sequential",
  "layers": [
   {
    "class_name": "InputLayer",
    "config": {
     "batch_input_shape": [
      null,
      8
     ],
     "dtype": "float32",
     "sparse": false,
     "ragged": false,
     "name": "dense_input"
    }
   },
   {
    "class_name": "Dense",
    "config": {
     "name": "dense",
     "trainable": true,
     "batch_input_shape": [
      null,
      8
     ],
     "dtype": "float32",
     "units": 12,
```

```
      "activation": "relu",
      "use_bias": true,
      "kernel_initializer": {
       "class_name": "GlorotUniform",
       "config": {
        "seed": null
       }
      },
      "bias_initializer": {
       "class_name": "Zeros",
       "config": {}
      },
      "kernel_regularizer": null,
      "bias_regularizer": null,
      "activity_regularizer": null,
      "kernel_constraint": null,
      "bias_constraint": null
     }
    },
    {
     "class_name": "Dense",
     "config": {
      "name": "dense_1",
      "trainable": true,
      "dtype": "float32",
      "units": 8,
      "activation": "relu",
      "use_bias": true,
      "kernel_initializer": {
       "class_name": "GlorotUniform",
       "config": {
        "seed": null
       }
      },
      "bias_initializer": {
       "class_name": "Zeros",
       "config": {}
      },
      "kernel_regularizer": null,
      "bias_regularizer": null,
      "activity_regularizer": null,
      "kernel_constraint": null,
      "bias_constraint": null
     }
    },
    {
     "class_name": "Dense",
```

```
                "config": {
                 "name": "dense_2",
                 "trainable": true,
                 "dtype": "float32",
                 "units": 1,
                 "activation": "sigmoid",
                 "use_bias": true,
                 "kernel_initializer": {
                  "class_name": "GlorotUniform",
                  "config": {
                   "seed": null
                  }
                 },
                 "bias_initializer": {
                  "class_name": "Zeros",
                  "config": {}
                 },
                 "kernel_regularizer": null,
                 "bias_regularizer": null,
                 "activity_regularizer": null,
                 "kernel_constraint": null,
                 "bias_constraint": null
                }
               }
              ]
             },
             "keras_version": "2.8.0",
             "backend": "tensorflow"
            }
```

Program Name: Loading the model from json file

Input file: demo2.py

Input file: pima-indians-diabetes.csv

```
# importing required libraries
from keras.models import model_from_json
import numpy as np

# load the dataset
dataset = np.loadtxt('pima-indians-diabetes.csv', delimiter = ',')

# split into input (X) and output (y) variables
X = dataset[:, 0:8]
y = dataset[:, 8]

json_file = open('model.json' , 'r')

model_j = json_file.read()
model = model_from_json(model_j)
model.load_weights("model.weights.h5")
print("Loaded model from disk")

model.compile(loss = 'binary_crossentropy' , optimizer = 'rmsprop' ,
metrics=['accuracy'])
score = model.evaluate(X, y)

print(score)
```

Output

```
Loaded model from disk
24/24 [==============================] - 0s 924us/step - loss: 0.4692 - accuracy: 0.782
[0.46923649311065674, 0.7825520634651184]
```