Tensorflow and keras

Sunday, 3 March 2024 11:07 PM

Tensorflow is an open source library for machine learning research. Tensorflow offers Keras APIs, which help in creating deep learning models

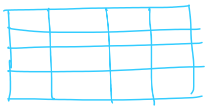
** It is known as tensorflow because it takes multi dimensional array as an input

What is tensor?

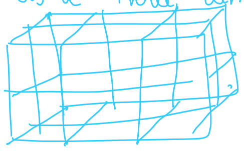
Tensor are multidimensional array

A vector is a one dimensional tensor

A matrix is a two dimensional tensor



A cube is a three dimensional tensor



```
#1-D tensor
  import tensorflow as tf
  x=tf.constant([5,6,8,6])
  # output  <tf.Tensor: shape=(4,), dtype= int32, numpy= array([5,7, 8,6], dtype= int32)
   x
```

```
#2-D tensor
  import tensorflow as tf
  x=tf.constant ([5,6,8,6],[5,6,8,6])
```

```
#1-D tensor to 2D tensor
  import tensorflow as tf
  x=tf.constant([5,6,8,6], shape=(2,2))
```

Deep learning Model

```
import tensorflow as tf
from tensorflow import keras

import numpy as np
import matplotlib.pyplot as plt

# import the dataset
      mnist = keras.datasets.fashion_mnist
```

Train test split

```
# train test split

    (X_train, y_test), (X_test, y_test) = mnist.load_data()

    X_train.shape
# output  (60000, 28,28)

    X_test. shape
# output ( 10000, 28, 28)

    X_train

    X_test

    np. min (X_train)
    np. max (X_train)

    y -train ( )
  # output
  # array ([9, 0, 0, ... , 3, 0, 5], dtype=uint8)

     plt.figure()
     plt. imshow (X_train (0))
     plt. colorbar ()
     plt. show ()

        X_train = X_train /255
        X_test = X_test /255

      plt.figure()
      plt. imshow (X_train (0))
      plt. colorbar ()
      plt. show ()

  from tensorflow.keras import Sequential
  from tensorflow.keras.layers import Flatten , Dense

    model = Sequential ()
     model. add(Flatten ( input_shape=(28,28)))
     model. add ( Dense ( 128, activation= 'relue'))
     model. add ( Dense ( 10, activation= 'softmax'))

     Print ( model. summary())
```

```
      Model : 'sequential'
   layer (type)            output shape          Param
  flatten (Flatten)        (None , 784)          0
  dense ( Dense)           ( None, 128 )         100480
  dense_1 ( Dense)         (None , 10)           1290

     Total params: 101,770
     Trainable params: 101,770
     Non-trainable params: 0
```

784* 128 + 128
128*10 + 10

```
model. compile [optimizer= "adam", loss= "sparse_categorical_crossentropy", metrics =["accuracy"])
```

weight updation $\frac{\partial L}{\partial W}$

loss calculation $y - \hat{y}$

Accuray of model

```
# model training
model.fit (x_train, y_train, epochs = 10)
# testing model on test data-
test - loss, test_accuray = model.evaluate (x_test, y-test)
```