StatusNeo

TensorFlow

KEYWORD

TENSORFLOW

It is an End to End machine learning platform .
Mainly used For training and interference of Deep Neural Network.It is also used for dataflow and differentiable programming

TENSOR

- An algebraic object describing multi linear relationship.
-generalization of vectors and matrices to potentially higher dimensions.
- TensorFlow program work by building graph of Tensor objects
- Each Tensor has a datatype and a shape
- Data Types :Float32,Int32,Strings and others
- Shape :Represents dimension of the data
-Creating a variable : tf.Variable(Value,tf.dtype)
-Type of tensors: Variable,constant,placeholder,Sparse Tensor

LINEAR REGRESSION

- Represents Relationship b/w dependent and independent var
- used when we want to predict the value of a variable based on the value of another variable (NUMERIC VALUE)
- Y = m * X + c (where X=dependent var Y=independent)

CLASSIFICATION

- classification is used to separate data points into classes of different labels.
-Classification comes under "supervised learning".
-Popular Algorithms:-KNN,Decision Tree,SVM,Naive Bayes

CLUSTERING

- involves the grouping of data points,data points in same group have similar prop and in different group have dissimilar prop.
- Clustering comes under "unsupervised learning".
- Most famous one is K Means Clustering.

KERAS

- deep learning API written in Python, running on top of the machine learning platform TensorFlow.
- Features:easy and fast prototyping,Support of RNN and CNN Runs seamlessly on CPU and GPU

NEURAL NETWORK

-are a set of algorithms, That try to mimics working of human brain
- Deep neural network is a layered representation of data.
- Deep refers to presence of multiple layers.
-Building block:Neuron(each neuron takes an input and perform some math on that input and produce an output

LAYERS

- INPUT LAYER :Initial layer where data is passed.first layer of NN.
- OUTPUT LAYER:the last layer which brings us our outputs.
- HIDDEN LAYER:Remaining layers in NN are hidden layer,most of NN consist of at least one hidden but can have unlimited .

ACTIVATION FUNCTION

- Activation functions are simply a function that is applied to the weighed sum of a neuron.
- Activation Function decides weather a Neuron should be activated or not
-Some examples are sigmoid, RELU , SWISH , TANH

EPOCH

- An epoch is simply one stream of our entire dataset
- The number of epochs we define is the amount of times our model will see the entire dataset
- We use multiple epochs in hope that after seeing the same data multiple times the model will better determine how to estimate it.

CORE LEARNING ALGORITHMS

1.LINEAR REGRESSION

1.creating the model:-

```
tf.estimator.LinearClassifier(feature_columns=feature_columns)
```

2.Training the Model:-

```
linear_est.train (INPUT)
```

3.Evaluation of the created Model:-

```
result = linear_est.evaluate(eval_input_fn)
print(result['accuracy'])
```

2.CLASSIFICATION (DNN)

1.creating the model:-

```
# Build a DNN with 2 hidden layers with 30 and 10 hidden nodes each.
classifier = tf.estimator.DNNClassifier(
feature_columns=my_feature_columns,
# Two hidden layers of 30 and 10 nodes respectively.
hidden_units=[30, 10],
# The model must choose between 3 classes.
n_classes=3)
```

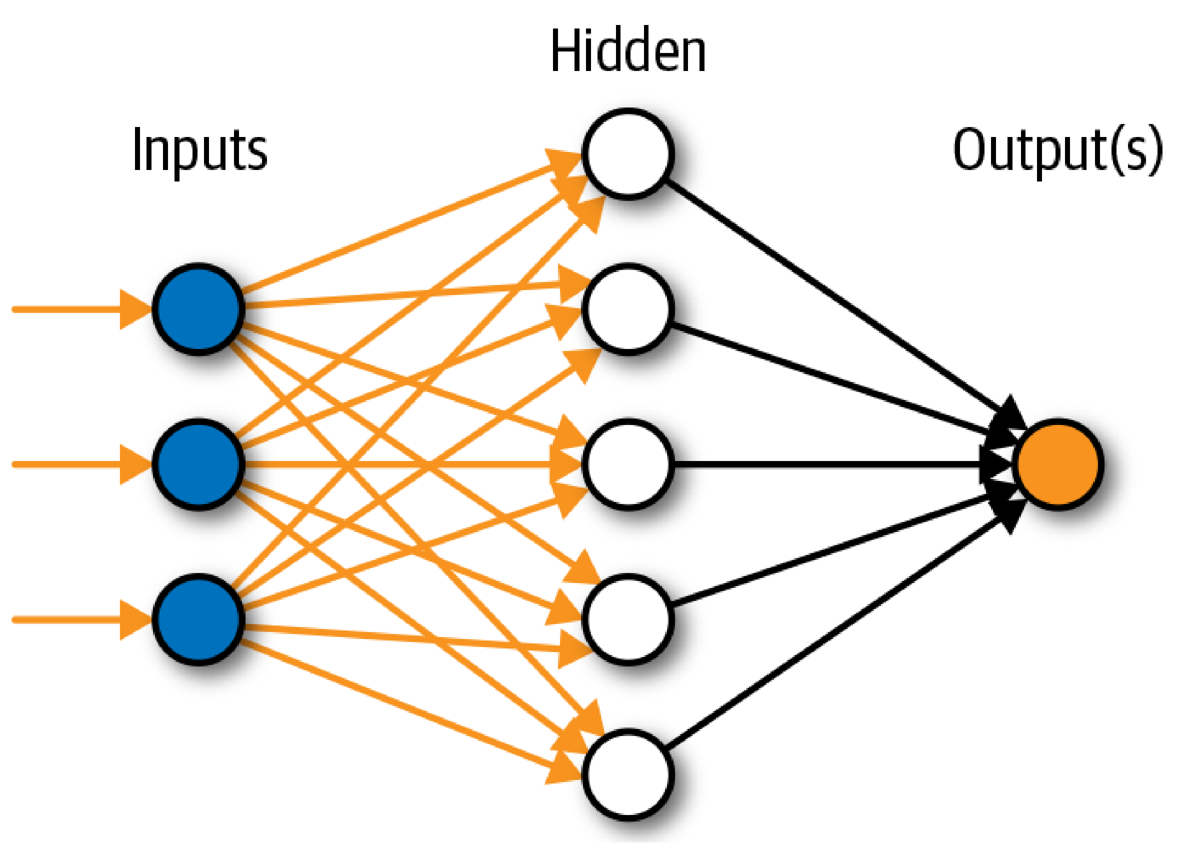
2.Training the Model:-

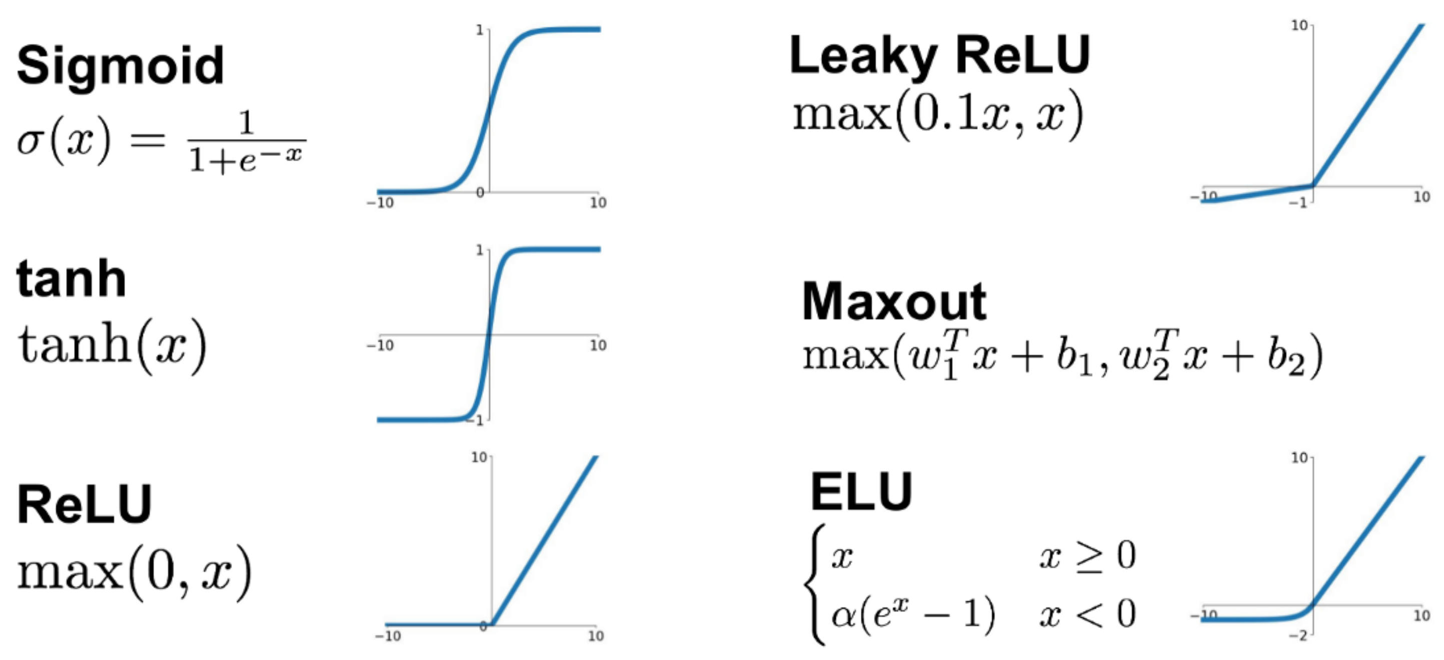
```
classifier.train(input_fn=lambda: input_fn(train, train_y, training=True),
 steps=5000)
```

3.Evaluation of the created Model:-

```
eval_result = classifier.evaluate(input_fn=lambda: input_fn(test, test_y,
training=False))
print('\nTest set accuracy: {accuracy:0.3f}\n'.format(**eval_result))
```

ANN WITH ACTIVATION FUNCTION





1.DEFINING A ANN WITH TENSORFLOW+KERAS

```
model = keras.Sequential([
keras.layers.Flatten(input_shape=(28, 28)),  # input layer (1)
keras.layers.Dense(128, activation='relu'),  # hidden layer (2)
keras.layers.Dense(10, activation='softmax') # output layer (3)
 ])
```

**softmax is used for multiclass Classification

BACKPROPAGATION

The Backpropagation algorithm looks for the minimum value of the error function in weight space using a technique called the delta rule or gradient descent. The weights that minimize the error function is then considered to be a solution to the learning problem.

OPTIMIZER

Optimizers are algorithms or methods used to change the attributes of the neural network such as weights and learning rate to reduce the losses. Optimizers are used to solve optimization problems by minimizing the function.

LOSS/COST FUNCTION

A loss function is used to optimize the parameter values in a neural network model. Loss functions map a set of parameter values for the network onto a scalar value that indicates how well those parameter accomplish the task the network is intended to do.

CONVULUTIONAL NEURAL NETWORK

A convolution is the simple application of a filter to an input that results in an activation. Repeated application of the same filter to an input results in a map of activations called a feature map, indicating the locations and strength of a detected feature in an input, such as an image.

FILTERS

Filters detect spatial patterns such as edges in an image by detecting the changes in intensity values of the image.

PADDING

Padding is a term relevant to convolutional neural networks as it refers to the amount of pixels added to an image when it is being processed by the kernel of a CNN.

STRIDES

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on.

DATA AGUMENTATION

I am a retail marketing consultant seeking a full-time position. With experience in retail market research,
I develop strategies that drive products to success.

RECURRENT NEURAL NETWORK

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

LSTM

Long short-term memory is an artificial recurrent neural network architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points, but also entire sequences of data

TOKENIZING

Tokenization is the process by which a large quantity of text is divided into smaller parts called tokens. These tokens are very useful for finding patterns and are considered as a base step for stemming and lemmatization.

TRANSFER LEARNING

take a model trained on a large dataset and transfer its knowledge to a smaller dataset. For object recognition with a CNN, we freeze the early convolutional layers of the network and only train the last few layers which make a prediction.

REINFORCEMENT LEARNING

Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning.

ANN CONT.

2.COMPILING THE MODEL

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

3.TRAINING THE MODEL

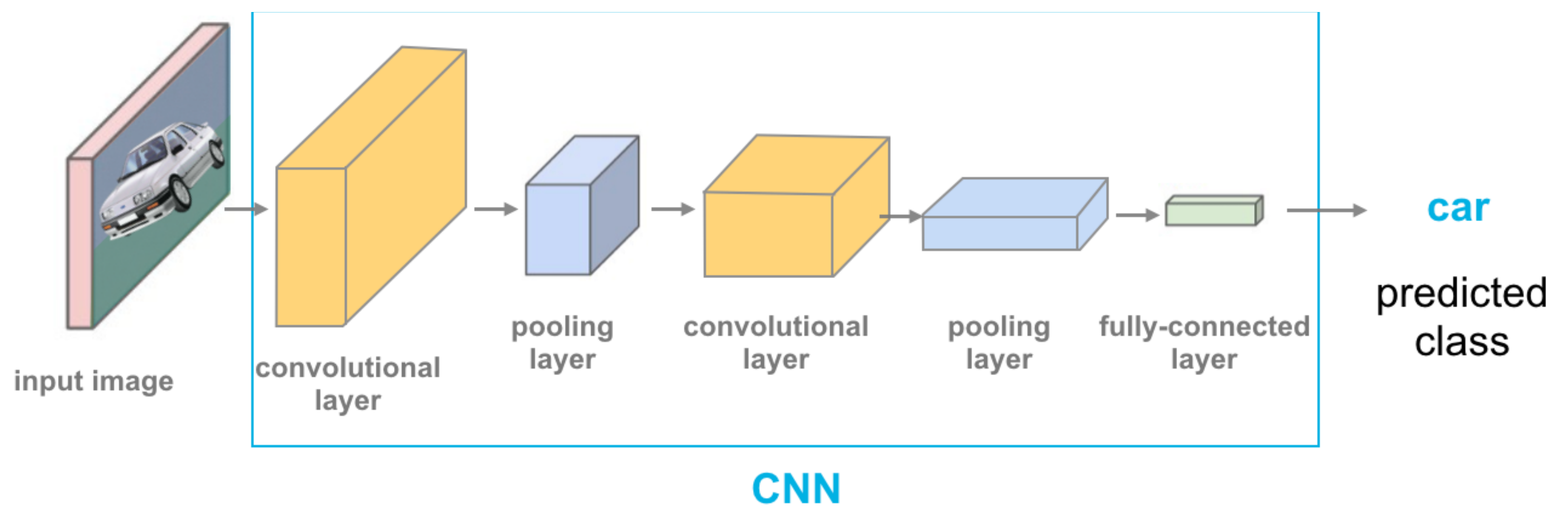
```
model.fit(train_images, train_labels, epochs=10)
```

4.MAKING PREDICTIONS

```
predictions = model.predict(test_images)
np.argmax(predictions[0])
```

CONVULUTION NEURAL NETWORK ARCHITECTURE

```
model=tf.keras.models.Sequential([
tf.keras.layers.Conv2D(16,(3,3),activation='relu',input_shape=150,150,3)),
tf.keras.layers.MaxPool2D(2,2),
tf.keras.layers.Conv2D(32,(3,3),activation='relu'),
tf.keras.layers.MaxPool2D(2,2),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(512,activation='relu'),
tf.keras.layers.Dense(4, activation='softmax')])
```



DATA AGUMENTATION

```
DATAGEN =
IMAGEDATAGENERATOR(
ROTATION_RANGE=40,
WIDTH_SHIFT_RANGE=0.2,
HEIGHT_SHIFT_RANGE=0.2,
SHEAR_RANGE=0.2,
ZOOM_RANGE=0.2,
HORIZONTAL_FLIP=TRUE,
FILL_MODE='NEAREST')
```

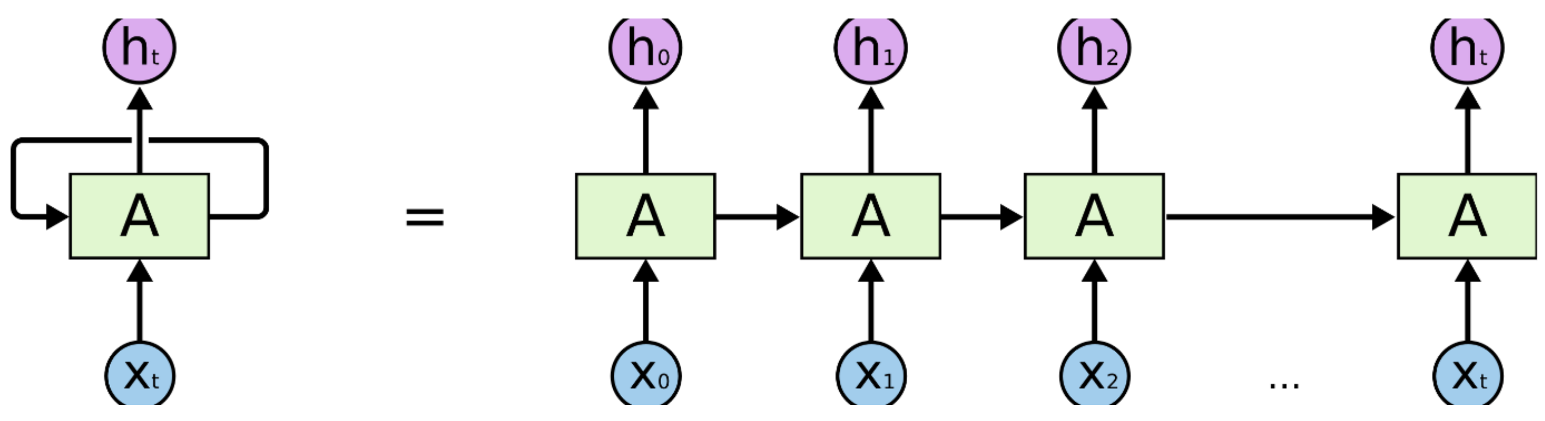
RECURRENT NEURAL NETWORK

1.Tokenizing text

```
TOKENIZER=TOKENIZER(NUM_WORDS=VOCAB_SIZE,OOV_TOKEN=OOV_TOK)
TOKENIZER.FIT_ON_TEXTS(TRAINING_SENTENCES)
WORD_INDEX=TOKENIZER.WORD_INDEX
TRAINING_SEQUENCES=TOKENIZER.TEXTS_TO_SEQUENCES(TRAINING_SENTEN
CES)
TRAINING_PADDED=PAD_SEQUENCES(TRAINING_SEQUENCES,MAXLEN=MAX_LE
NGTH,TRUNCATING=TRUNC_TYPE)
```

2.Creating the model

```
MODEL = TF.KERAS.SEQUENTIAL([
    TF.KERAS.LAYERS.EMBEDDING(VOCAB_SIZE, 32),
    TF.KERAS.LAYERS.LSTM(32),
    TF.KERAS.LAYERS.DENSE(1, ACTIVATION="SIGMOID")
])
```



3.Training and prediction the model

```
MODEL.FIT(PADDED, TRAINING_LABELS_FINAL, EPOCHS=NUM_EPOCHS,
VALIDATION_DATA=(TESTING_PADDED, TESTING_LABELS_FINAL))

SENTENCE = "I REALLY THINK THIS IS AMAZING. HONEST."
SEQUENCE = TOKENIZER.TEXTS_TO_SEQUENCES([SENTENCE])
PADDED = PAD_SEQUENCES(SEQUENCES, MAXLEN=MAX_LENGTH, PADDING='POST',
TRUNCATING=TRUNC_TYPE)
PRINT(MODEL.PREDICT(PADDED))
NP.ARGMAX(MODEL.PREDICT(PADDED))
```