Tensorflow

Methods

- TensorFlow Tensors: Tensors are the fundamental building blocks in TensorFlow, representing multi-dimensional arrays of data. They can be constants or variables, and TensorFlow provides methods to perform various operations on tensors, such as addition, multiplication, reshaping, and more.

- TensorFlow Operations: TensorFlow offers a vast collection of operations that can be performed on tensors. These include arithmetic operations, activation functions (ReLU, sigmoid, etc.), matrix operations (matmul), reduction operations (sum, mean, etc.), and many more.

- TensorFlow Variables: Variables are special tensors that hold trainable parameters of a model. They are used to store and update the model's weights during training.

- TensorFlow Graphs: TensorFlow uses a computational graph representation to define and organize operations. The graph defines the structure of the model and how data flows through it.

- TensorFlow Sessions: Before TensorFlow 2.0, you needed to create a session to run operations in the graph. Sessions provided a way to execute computations and evaluate tensors.

- TensorFlow Keras: TensorFlow provides an easy-to-use high-level API called Keras for building and training neural networks. In TensorFlow 2.0 and later versions, Keras is tightly integrated with TensorFlow and serves as the default high-level API for building models.

- TensorFlow Estimators: Estimators are a high-level API for creating TensorFlow models, especially for distributed training on clusters. They simplify the process of building and training models for common tasks like classification, regression, and clustering.

- TensorFlow Data API: TensorFlow offers tools for efficiently managing and loading large datasets. The Data API allows you to create data input pipelines to feed data into your models.

Keras

Methods

- Sequential API: The Sequential API allows you to create a linear stack of layers, where each layer has exactly one input tensor and one output tensor. It is suitable for building simple feedforward neural networks.

- Functional API: The Functional API allows for more flexible model architectures, including multi-input and multi-output models, shared layers, and complex network topologies. It enables you to define models as directed acyclic graphs of layers.

- Layers: Keras provides a wide range of built-in layers that you can use to construct your models. Examples include Dense (fully connected layer), Conv2D (2D convolutional layer), LSTM (Long Short-Term Memory layer), etc.

- Activations: Keras offers various activation functions that can be applied to the outputs of layers, such as ReLU, sigmoid, tanh, softmax, etc.

- Loss Functions: Loss functions are used to measure the error or discrepancy between the predicted output and the actual target. Keras provides a collection of built-in loss functions suitable for different types of tasks, like categorical_crossentropy, mean_squared_error, binary_crossentropy, etc.

- Optimizers: Keras includes popular optimization algorithms to train your models, such as Adam, RMSprop, SGD, and others.

- Callbacks: Callbacks are functions that can be applied during the training process to perform various tasks. They allow you to monitor the training progress, save the best model, adjust learning rates dynamically, and more.

- Model Compilation: Before training, you need to compile your Keras model. During compilation, you specify the loss function, optimizer, and additional metrics to monitor during training.

- Model Training: Keras provides the fit() method to train your model using labeled data. You specify the training data, labels, batch size, number of epochs, and other parameters.