

1 Introduction

TensorFlow is an open-source library for numerical computation and machine learning, especially deep learning. It allows for the creation and training of neural networks with flexible architecture.

1.1 Installation

```
1 pip install tensorflow
```

1.2 Importing TensorFlow

```
1 import tensorflow as tf
```

2 Tensors

2.1 Creating Tensors

```
1 # Scalar
2 scalar = tf.constant(7)
3 # Vector
4 vector = tf.constant([1, 2, 3])
5 # Matrix
6 matrix = tf.constant([[1, 2], [3, 4]])
7 # Random Tensor
8 tensor = tf.random.normal((3, 3))
```

2.2 Tensor Attributes

```
1 tensor.shape # Shape of tensor
2 tensor.dtype # Data type
```

3 Tensor Operations

3.1 Mathematical Operations

```
1 a = tf.constant([1, 2, 3])
2 b = tf.constant([4, 5, 6])
3
4 add = a + b
5 sub = a - b
6 mult = a * b
7 div = a / b
8
9 # Matrix Multiplication
10 mat_mult = tf.matmul(a[tf.newaxis, :], b[:, tf.newaxis])
```

3.2 Tensor Manipulation

```
1 # Reshaping
2 reshaped = tf.reshape(a, (3, 1))
3 # Slicing
4 slice_a = a[0:2]
5 # Concatenation
6 concatenated = tf.concat([a, b], axis=0)
```

4 Building Models

4.1 Sequential Model

```
1 from tensorflow.keras import Sequential
2 from tensorflow.keras.layers import Dense
3
4 model = Sequential([
5     Dense(64, activation='relu', input_shape=(32,)),
6     Dense(10, activation='softmax')
7 ])
```

4.2 Compiling the Model

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

5 Training Models

5.1 Data Preparation

```
1 # Assuming X_train and y_train are prepared
```

5.2 Model Fitting

```
1 model.fit(X_train, y_train, epochs=10,
2          validation_split=0.2)
```

6 Evaluating Models

6.1 Model Evaluation

```
1 loss, accuracy = model.evaluate(X_test, y_test)
2 print(f'Loss: {loss}, Accuracy: {accuracy}')
```

6.2 Making Predictions

```
1 predictions = model.predict(X_new)
```

7 Advanced Topics

7.1 Custom Layers

```
1 from tensorflow.keras.layers import Layer
2
3 class CustomLayer(Layer):
4     def __init__(self, units=32):
5         super(CustomLayer, self).__init__()
6         self.units = units
7
8     def build(self, input_shape):
9         self.w = self.add_weight(shape=(
10             input_shape[-1], self.units),
11                                   initializer='random_normal')
12         self.b = self.add_weight(shape=(self.units),
13                                   initializer='zeros')
14
15     def call(self, inputs):
16         return tf.matmul(inputs, self.w) + self.b
```

7.2 Callbacks

```
1 from tensorflow.keras.callbacks import EarlyStopping
2
3 early_stopping = EarlyStopping(monitor='val_loss',
4                                patience=3)
5
6 model.fit(X_train, y_train, epochs=10,
7         validation_split=0.2, callbacks=[early_stopping])
```

8 Working with Data Types

8.1 Image Data

```
1 from tensorflow.keras.preprocessing.image import
  ImageDataGenerator
2
3 datagen = ImageDataGenerator(rescale=1./255)
4 train_generator = datagen.flow_from_directory(
5     'data/train', target_size=(150, 150),
6     batch_size=32, class_mode='binary')
```

8.2 Text Data

```
1 from tensorflow.keras.preprocessing.text import
  Tokenizer
2
3 tokenizer = Tokenizer(num_words=10000)
4 tokenizer.fit_on_texts(texts)
5 sequences = tokenizer.texts_to_sequences(texts)
```

9 Model Deployment

9.1 Saving and Loading Models

```
1 # Save model
2 model.save('my_model.h5')
3 # Load model
4 from tensorflow.keras.models import load_model
5 model = load_model('my_model.h5')
```

9.2 TensorFlow Serving

```
1 # Export the model
2 model.save('export/saved_model/1')
3
4 # Start TensorFlow Serving
5 tensorflow_model_server --rest_api_port=8501 \
6     --model_name=my_model --model_base_path="/
  export/saved_model/"
```

10 GPU Utilization

10.1 Setting up GPU Support

```
1 physical_devices = tf.config.list_physical_devices
  ('GPU')
2 if physical_devices:
3     tf.config.experimental.set_memory_growth(
4         physical_devices[0], True)
```

10.2 Performance Optimization

```
1 # Mixed Precision Training
2 from tensorflow.keras.mixed_precision import
  experimental as mixed_precision
3 policy = mixed_precision.Policy('mixed_float16')
4 mixed_precision.set_policy(policy)
```

11 TensorBoard

11.1 Visualization

```
1 from tensorflow.keras.callbacks import TensorBoard
2
3 tensorboard = TensorBoard(log_dir='logs')
4 model.fit(X_train, y_train, epochs=10, callbacks=[
  tensorboard])
```

11.2 Launching TensorBoard

```
1 tensorboard --logdir=logs
```

12 Resources

12.1 Official Documentation

- TensorFlow Documentation

12.2 Courses

- TensorFlow in Practice by Andrew Ng
- Deep Learning with TensorFlow on edX

12.3 Books

- Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* by Aurélien Géron
- Deep Learning with Python* by François Chollet