Tensorflow 2.0:
Building Computer Vision Applications

Prof. D. Narayana

Agenda

- Introduction to Tensorflow 2.0
- Tensorflow 1.x vs Tensorflow 2.0
- Computer vision application – Example
- COVID19 detection using chest x-rays
- Tensorflow Serving and Tensorflow Lite
- Q&A

TensorFlow 2.0

- TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications

- TF 2.0 is a major update focused on ease of use

Source: https://www.tensorflow.org

Tensorflow 1.x Vs Tensorflow 2.0

- Tensorflow 2.0 is simpler
- If you find tf.contrib in your code, you are using older version of tensorflow
- tf.contrib is obsolete
- Other layers are not needed
- tf.session is obsolete
- Eager execution is enabled by default
    - No placeholders, session.run(), tf.global_variables_initializer()
- Use @tf.function for the efficiency of compiled graph

Tensorflow 1.x Vs Tensorflow 2.0

- Keras API is now the standard
- To create custom layers and models, subclass the Keras layer
- Building custom layers:
    - You can build custom layers apart from layers provided by Keras API

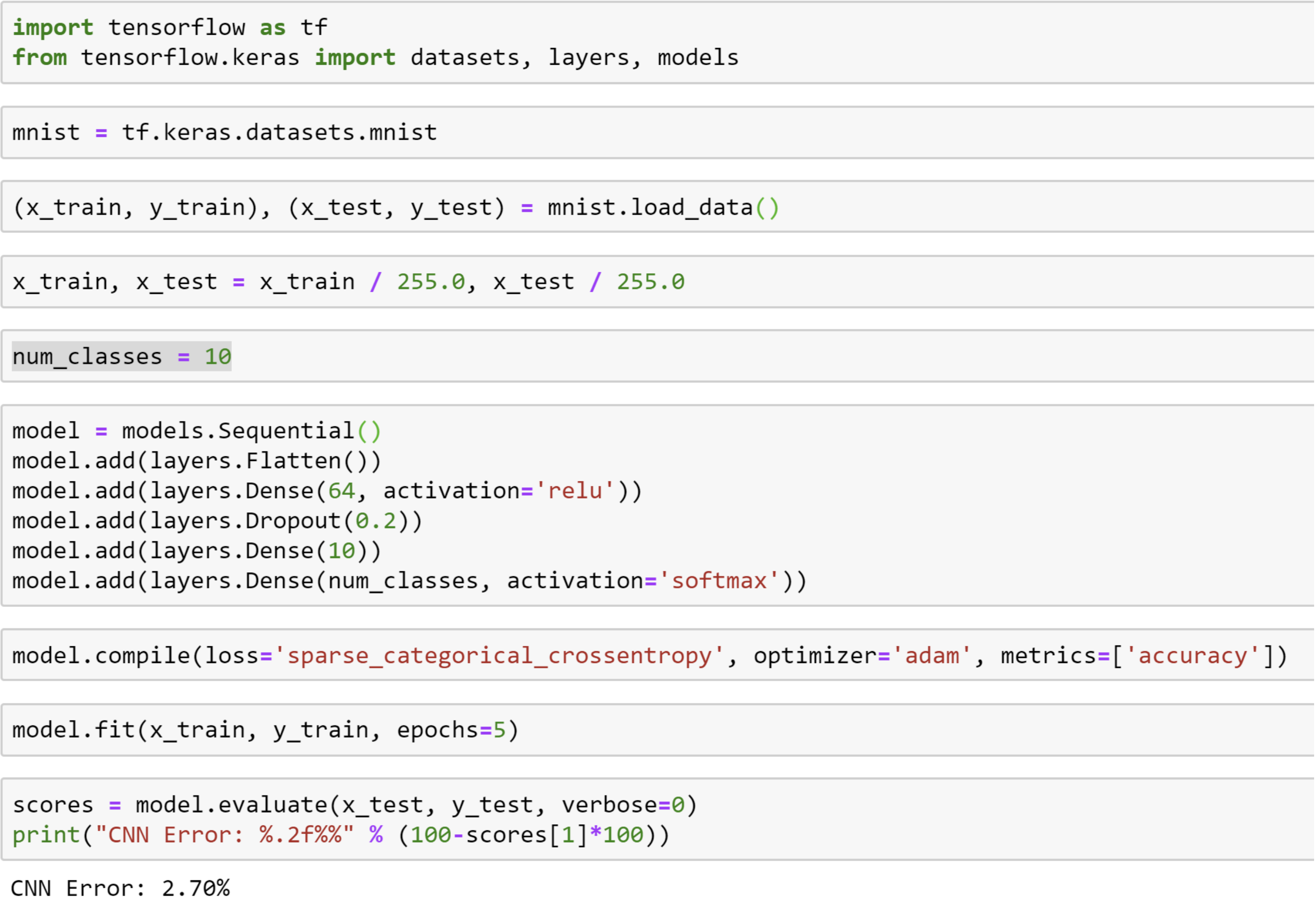
```
class newLayer(tf.keras.layer.Layer):
        “New layer code”
```

    - You need to conform to the Keras API using subclass

Example1: MNIST

- **Problem Definition:**
  - Classify MNIST handwritten digits?

Example1: MNIST



```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
```

```
mnist = tf.keras.datasets.mnist
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
num_classes = 10
```

```
model = models.Sequential()
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(10))
model.add(layers.Dense(num_classes, activation='softmax'))
```

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

```
model.fit(x_train, y_train, epochs=5)
```

```
scores = model.evaluate(x_test, y_test, verbose=0)
print("CNN Error: %.2f%%" % (100-scores[1]*100))
```

```
CNN Error: 2.70%
```

MNIST Classification: Inferences

- MNIST is a widely used dataset for hand-written digit classification
- It consists of 70,000 labeled 28x28 pixel grayscale images of hand-written digits
- Using simple fully connected network, we could get above 97% classification accuracy

Case Study: COVID19 Detection

1. Problem definition
2. Data
3. Classical ML models
4. CNN models

Case Study: COVID-19 Detection

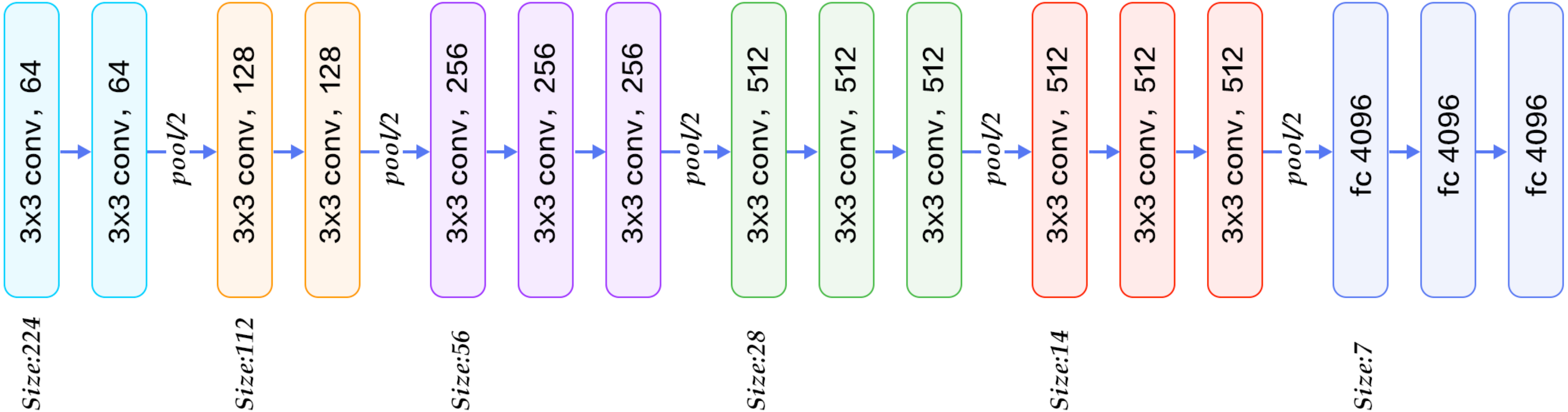
- **Problem Definition:**
  - **Detect COVID-19 induced pneumonia using chest x-ray images**

COVID-19 Detection: Data

- Chexpert largest chest x-ray dataset
  - https://stanfordmlgroup.github.io/competitions/chexpert/
- Kaggle Pneumonia dataset
  - https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia
- COVID Chest x-ray open dataset released by University of Montreal
  - https://github.com/ieee8023/covid-chestxray-dataset

COVID-19 Process Flow and Results

VGG16 Architecture



1.Convolution using 64 filters
2.Convolution using 64 filters + Max pooling
3Convolution using 128 filters
4.Convolution using 128 filters + Max pooling
5.Convolution using 256 filters
6.Convolution using 256 filters
7.Convolution using 256 filters + Max pooling
8.Convolution using 512 filters
9.Convolution using 512 filters
10.Convolution using 512 filters + Max pooling
11.Convolution using 512 filters
12.Convolution using 512 filters
13.Convolution using 512 filters + Max pooling
14.Fully connected with 4096 nodes
15.Fully connected with 4096 nodes
16.Output layer with Softmax activation with No of classes

Results

Architecture	Precision	Recall	F1-Score
VGG16	100%	72%	84%
ResNet50	68%	94%	79%

COVID-19 Detection: Conclusions

- Using the proposed AI techniques based workflow we are in a position to analyze the patient’s condition using x-rays by spending minimal time of radiologists and patient’s money. These sorts of solutions are needed for countries like India where the population size is huge.

TensorFlow Serving

- Is a flexible, high-performance serving system for ML models, designed for production environments
- Makes it easy to deploy new algos while keeping the same server architecture and APIs
- Provides out-of-the-box integration with TF models, but can be easily extended to serve other types of models and data
- Important TensorFlow Serving Concepts:
  - Servables
  - Servable versions
  - Servable streams
  - Models
  - Loaders
  - Sources
  - Aspired versions
  - Managers
  - Core
  - Extensibility

Source: https://www.tensorflow.org/tfx/guide/serving

Tensorflow Lite

- **TensorFlow Lite is an open source deep learning framework for on-device inference**

- **How to build models for mobile devices?**
  - **Pick a new model or retrain an existing one**
  - **Convert a TensorFlow model into a compressed flat buffer with the TensorFlow Lite Converter**
  - **Load compressed .tflite file into a mobile or embedded device**

Source: https://www.tensorflow.org/lite

Summary

- Aadvantages of Tensorflow 2.0
- Tensorflow 1.x vs Tensorflow 2.0
- Computer vision applications development
- COVID19 detection
- Tensorflow Serving
- Tensorflow Lite

Q&A

Thank You