Welcome!

https://pjreddie.com/darknet/yolo/

Welcome to **Perform Real-Time Object Detection with YOLOv3**. This is a project-based course which should take approximately 1.5 hours to complete. Before diving into the project, please take a look at the course objectives and structure:

Course Objectives

In this course, we are going to focus on four learning objectives:

1. Perform real-time object detection with YOLOv3
2. Use OpenCV to manipulate video data
3. Use pre-trained models to perform real-time and passive inference with a GPU
4. Develop a simple command line application with Python for inference

By the end of this course, you will be able to detect objects in videos using the YOLO system.

Course Structure

This course is divided into 3 parts:

1. Course Overview: This introductory reading material.
2. Perform Real-Time Object Detection with YOLOv3**:**This is the hands on project that we will work on in Rhyme.
3. Graded Quiz: This is the final assignment that you need to pass in order to finish the course successfully.

Project Structure

The hands on project on Performing Real-Time Object Detection is divided into following tasks:

Task 1: Introduction and Overview

* Introduction to the data and and overview of the project.
* See a demo of the final product you will build by the end of this project.
* Introduction to the Rhyme interface.
* Import essential modules and helper functions from [NumPy](https://numpy.org/" \t "_blank), [Matplotlib](https://matplotlib.org/" \t "_blank), and [Keras](https://www.tensorflow.org/guide/keras" \t "_blank).

Task 2: Explore the Dataset

* Display some images from every expression type in the Emotion FER [dataset](https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data).
* Check for class imbalance problems in the training data.

Task 3: Setup Training and Validation Data Generators

* Generate batches of tensor image data with real-time data augmentation.
* Specify paths to training and validation image directories and generates batches of augmented data.

Task 4: Create a Convolutional Neural Network (CNN) Model

* Design a convolutional neural network with 4 convolution layers and 2 fully connected layers to predict 7 types of facial expressions.
* Use Adam as the optimizer, categorical crossentropy as the loss function, and accuracy as the evaluation metric.

Task 5: Train and Evaluate Model

* Train the CNN by invoking the **model.fit()**method.
* Use **ModelCheckpoint()** to save the weights associated with the higher validation accuracy.
* Observe live training loss and accuracy plots in Jupyter Notebook for Keras.

Task 6: Save and Serialize Model as JSON String

* Sometimes, you are only interested in the architecture of the model, and you don't need to save the weight values or the optimizer.
* Use **to\_json()**, which uses a JSON string, to store the model architecture.

Task 7: Create a Flask App to Serve Predictions

* Use open-source code from "[Video Streaming with Flask Example](https://github.com/log0/video_streaming_with_flask_example)" to create a flask app to serve the model's prediction images directly to a web interface.

Task 8: Create a Class to Output Model Predictions

* Create a FacialExpressionModel class to load the model from the JSON file, load the trained weights into the model, and predict facial expressions.

Task 9: Design an HTML Template for the Flask App

* Design a basic template in HTML to create the layout for the Flask app.

Task 10: Use Model to Recognize Facial Expressions in Videos

* Run the **main.py** script to create the Flask app and serve the model's predictions to a web interface.
* Apply the model to saved videos on disk.

Meet the Instructor

I'm Snehan Kekre, a machine learning and data science instructor at Rhyme. I will graduate in 2021 with a BSc in Computer Science and Artificial Intelligence from Minerva Schools at KGI, based in San Francisco. My passion to grow the ML and AI community drives me to create accessible, skills-based, machine learning projects at Rhyme.

About Rhyme

This course runs on Coursera's hands-on platform called Rhyme. On Rhyme, you do projects in a hands-on manner in your browser. You will get instant access to pre-configured cloud desktops that have all the software and data you will need. So, you can just focus on the learning. For this project, this means instant access to a cloud desktop with Python, Jupyter, and TensorFlow pre-installed.