Identify Eight types of Indian Classical

Dance forms with YOLOv4

June 01, 2020

SHARE

Labels

custom dataset

labellmg

training

YOLOv4

Another post starts with you beautiful people!

Thank you all who had followed my last post about install and compile YOLOv4 in Windows10 and could able to successfully set up the Darknet in their machines. As I

promised in last post and you asked for, in this post I am going to share you the steps required for training a custom object with YOLOv4. If you are seeing my blog first time, I recommend you to first follow my last post and then proceed further. For this exercise I have choosen a dataset of eight Indian Classical Dance forms-

1. **Manipuri** from Manipur
2. **Bharatanatyam** from Tamil Nadu

3. **Odissi** from Orissa
4. **Kathakali** from Kerala
5. **Kathak** from Uttar Pradesh
6. **Sattriya** from Assam
7. **Kuchipudi** from Andhra Pradesh
8. **Mohiniyattam** from Kerala

You can download the dataset from this hackethon link. After downloading the dataset , you need to create 8 folders with class name and copy respective images from train folder to there. For this work I have written a simple Python script that

creates eight folders with the images of dance forms. So you can also do the same. After completing this post you will be well aware of building a Neural Network model to classify eight categories of Indian classical dance. Our first step is to label the images in YOLO format. YOLO requires a specific type of labeling. The labelled format of an image should be in **<object-class> <x_center> <y_center> <width> <height>** format. Here **<object-class>** is integer

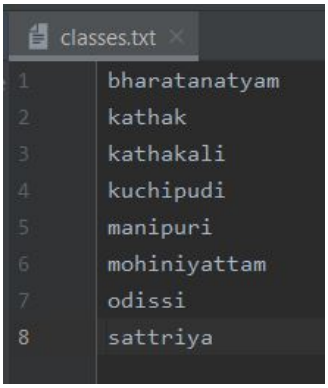
object number from 0 to (classes-1). In our case it will be zero to seven.
<x_center> <y_center> <width> <height> are float values relative to width and height of image which can be equal from [0.0 to 1.0]. This steps is the backbone of the YOLO and so it must be performed with precisely.

To label the image we will use **labellmg** annotation tool. Clone or Download the github code from this link in your machine. Once

cloned/downloaded, open anaconda prompt with admin rights, go to the path of labellmg and install required **pyqt** library using command: **conda install pyqt=5**
After installing pyqt, run following command: **pyrcc5 -o libs/resources.py resources.qrc**
This will install annotation tool in your machine.

Now, create a text file named as **classes.txt** in labellmg root directory and put our target class

names(dance forms) in this format-



Now our labellmg setup is done. We will start annotating the images later. first we will prepare some files and make some changes in our YOLO installed directory for the training of our custom dataset. For this step go to your YOLO installed directory(\vcpkg-master\installed\x64-windows\tools\dark

net\cfg) and follow below steps-

1. Copy the **yolov4-custom.cfg** file, paste there and rename it to **yolo-obj.cfg**

2. Open yolo-obj.cfg file in notepad++ and make below changes-

- change line **batch** to batch=**64**
- change line **subdivisions** to subdivisions=**64**
- change line **max_batches** to **16000**(classes*2000)
- change line **steps** to 80% and 90% of max_batches, f.e.**12800**, **14400**

- set network size **width=416** **height=416** or any value multiple of 32
- change line **classes=8**
- change **[filters=255]** to **filters=39**(classes + 5)x3 in the **3** [convolutional] **before** each [yolo] layer

3. Go to \vcpkg-master\installed\x64-windows\tools\darknet**data** directory and create a file **obj.names.** Copy the content of the classes.txt file there

4. Create a file **obj.data** in the same directory with below content-

Here update the path according to your directory. For backup, create a folder with name backup in the root directory (\vcpkg-master\installed\x64-windows\tools**darknet**). In this folder our custom trained weights files will be saved.

5. Create a folder named as **obj** inside the directory(\vcpkg-master\installed\x6

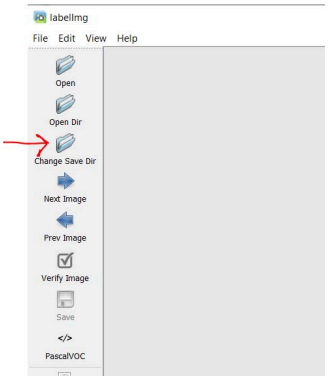
4-
windows\tools\dark
net**data**)

Now your YOLO related changes are done. Next, we will start annotating the images. For this open the anaconda prompt,go to the labellmg directory path and run the following command-

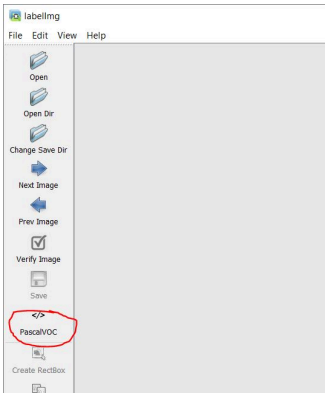
python labelImg.py
/dataset/images/bh
aratanatyam
classes.txt

This command will open labellmg window. Here click on the 'Change Save Dir' icon and select

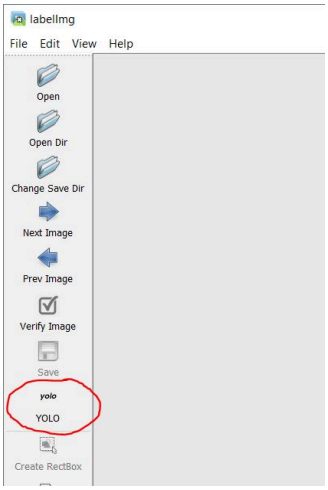
the bharatanatyam folder path (/dataset/images/bharatanatyam)-



Above step will save our annotation txt file in the same location where our images are. So in this case of annotation we will have <img_name>.txt file inside the /dataset/images/bharatanatyam folder. Next click on the 'PascalVOC' icon-

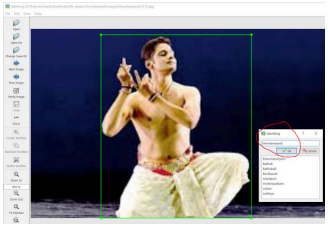


Once you click on it, it will change to YOLO like in below screen-



Since we have already given the image folder path and classes.txt file while running the labellmg command,

the first image from the image folder will be opened by default in labellmg window. Now click on the 'Create RectBox' icon and draw the annotation on the object you want to capture like below-



After drawing the bounding box, select the bharatnatyam option from the small window and click on ok then click on save button. That's it your first annotation file will be save in the location you have chosen while

'Change Save Dir' option. For next image just click on 'Next Image' icon and repeat the same till you get the last image. Once one folder is completed close this labellmg window, go to the anaconda prompt and run the command with change image directory path as below-

python labelImg.py /dataset/images/kathak classes.txt

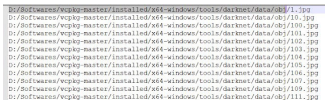
It will again open a labellmg window with showing kathak's folder first

image. Click on 'Change Save Dir' icon and select the /dataset/images/**kat hak** and start annotating images like we did earlier. Repeat the same step for all target classes but don't forget to change the save dir path with every new class. Although this annotation step is time taking but requires your attention so do it carefully. I have done my annotations in 1.5 hours. Once done with all folders, got to each folder copy the .txt and .jpg files and paste in obj

folder you have created in step 5. Do this for all folders. Now in obj folder you will have all image files and their respective txt files. Please make sure you vae not copied the classes.txt file in the obj folder. If yes then remove that file from obj folder.

Next, go to the directory \vcpkg-master\installed\x64-windows\tools\darknet**data** and create a txt file named as **train.txt**. In this file we need to give the image name with file

path. Here file path is **/vcpkg-master/installed/x64-windows/tools/darknet/data/obj** so the whole file will be look like below-



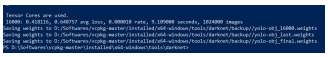
You can write a Python script for this work. With this step we are ready to start the training of our dance form custom data.

Open windows powershell with admin rights and go to the root directory of the Darknet and run the

command **.\darknet.exe detector train data/obj.data cfg/yolo-obj.cfg yolov4.conv.137**

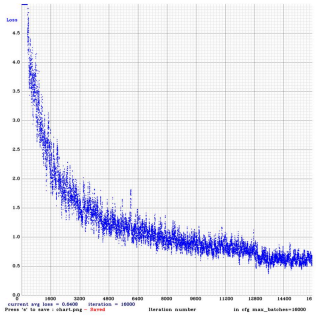
If you have not downloaded the weights yolov4.conv.137, download from this link and put inside root directory of Darknet. Once you run the above command, Darknet will load the layers from this weights file and start the training on our custom data. It finds our data from the obj.data file where we have mentioned number of classes, training

data path and location of saving the trained weights. This training will take quite enough time since we have given max_batches with 16000 and we have total 8 classes. In my machine it took **18 hours** to complete the training so I advise you to disable the sleep property of your machine and start the training in night. After successfully completion of the training you will see following like screen-



This training

command will also save **loss vs iteration number** chart in the root directory of the Darknet like below-



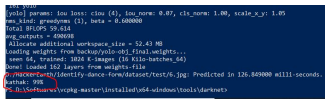
You can see in the starting of the training the average loss was very high but in the end of the training it is 0.6408. The final average loss can be from 0.05 (for a small model and easy dataset) to 3.0 (for a big model and a difficult dataset). So we have a very good model. Now we can

test our model on unseen images. For this run the following command in the same powershell window-

```
.\darknet.exe detector test data/obj.data cfg/yolo-obj.cfg backup/yolo-obj_final.weights /dataset/test/6.jpg -dont_show -out_filename
```

Here just change the test image path according to your system. Above command will predict the class on the given image with confidence in the

powershell console as well as it will save the image with bounding box and class name in the root Darknet directory with name as 'predictions.jpg'-



And it's corresponding prediction image file is as below-

See our model is able to detect the dance form as Kathak for this unseen image which is correct.

Let's check another unseen image using

command **.\darknet.exe detector test data/obj.data cfg/yolo-obj.cfg backup/yolo-obj_final.weights /dataset/test/6.jpg -dont_show -out_filename**

Looks like our model is working as expected. Please note for a very good YOLO model we should preferably have 2000 different images for each class or more, and we

should train 2000*classes iterations or more. Due to my system configurations I have used image width and height as 416 but if you have got bigger avg loss then you should increase the network resolution to 608 or 832. Also if you don't want to detect any specific object then you should also include that object images with **empty txt** files in the obj folder.

That's it guys for today. If you have followed till the end, you have spent your

valuable time in learning a state of the art computer vision technique. Here we are using the original Darknet framework, no any wrapper for our work. There are only very few persons who know using such a state of the art computer vision technique. Now you are one of them but to be a master on a technique you must practice and do experiments. So don't stop yourself just reading here. Do practical on your system, solve issues while doing hands

on, train YOLOv4 on another custom dataset, deploy your custom model as Rest API and showcase in your github repository. In my next post I will share my new learning to you; till then Go chase your dreams, have an awesome day, make every second count and see you later in my next post.

LABELS: CUSTOM DATASET, LABELLMG, TRAINING, YOLOV4

SHARE

Comments

Power Mind Technologies
· July 7, 2020 at 3:30 PM

Thanks for Sharing a Lovely Post for us and Please Keep Updating & Increase Our Knowledge & If any one Want to Grow your Career in Python Language So you Can Contact Here- 9354210902/ 9310806006 Or Visit Website- https://www.powermindtech.com/

Here is the best Training Center for Data Science

python training institute in gurgaon
python training Center in gurgaon

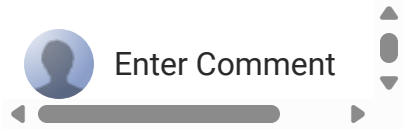
REPLY

theclondmonks Ramakrishna

· November 27, 2021 at 10:34 AM

It is very useful for me. Thanks...
Azure Data Factory Online Training
Azure Data Factory Online Training

REPLY



Popular posts from this blog

How to use opencv - python with Darknet's YOLOv4?

August 14, 2020

Another post starts with you beautiful peo ...

SHARE

82 COMMENTS

READ MORE

How to convert your YOLOv4 weights to Tensor Flow 2.2.0?

June 06, 2020

Another post starts with you beautiful peo ...

SHARE

87 COMMENTS

READ MORE

Machine Learning- Cross Validation & ROC curve

September 27, 2017

Another post starts with

you ...

SHARE

26 COMMENTS

READ MORE

Home
Python Basics- Lists
Python Basics- Dictionaries
Python Basics- Introducing functions
Know your jupyter notebook
Python Basics- if statements
Python Basics- hands on with strings
Python Basics - know about libraries
Python Basics- Hello World!
Getting started by installing Anaconda
Python Advanced- Inroduction to NumPy
Python Advanced- Series

Python Advanced- DataFrame

Followers

Total Pageviews

211,560

About Me

Learn Data Science with PrateekG

Prateek is a Data Scientist, Technology Enthusiast, and Blogger with over 13 years of experience in Machine Learning, Deep Learning, Optical Character Recognition (OCR), Generative AI, and Computer Vision using Python. Currently working as a Computer Vision Eng ...

VISIT PROFILE

Archive

Labels

Report Abuse

Practical Data Science with Jupyter Book: Buy Here