Get Trained to Train

Jyotishko Banerjee

***Abstract*—**
According to the project, in a distant future, we live in a world where we possess the excellence of machine learning. With an urge to know our past and the origin of AI/Ml. We have set to a journey of finding ancient knowledge of how the great tools that assist us were created. After a lot searching, we find a USB device and when we read, we find a zip file with 100 images and The Commandments. They throw us a challenge as follows.

Keeping aside the storyline, according to the The Commandments, we are supposed to make a Machine Leaning Model using Google Colab by using GPU from Google Servers and train the model to identify 'ball' from an image by the user. The training must be done by first feeding in the 100 images after making Annotations and let the model learn to detect balls, then make some validations. And then finally when the user feeds in an image, the model should successfully identify the ball. The detailed procedures are explained in the following sections.

I. INTRODUCTION

The project has been designed to teach us basics of machine learning models, training them from scratch, to understand their deep formulas, their methods of predictions and how they work. The task also teaches us how to use tools like Google Colab and labelImg.

Google Colab is a very efficient way to run high-end codes without putting much load on our own system. Basically, it's like online Jupyter notebook. Moreover, we do not even need to install the libraries, we can directly import them to script. It uses the Google servers to execute our codes. But before that, successful connections and memory allocations are necessary to run codes successfully. It can use 2 types of processors, CPU and GPU. While CPU are just sufficient for running normally loaded codes, for heavily loaded codes, GPU becomes a necessity as it gives better processing power.

LabelImg is a GUI based tool that lets the user annotate images that are to be fed to model to train it. Here, annotations mean marking the co-ordinates of place of where the object to be detected it present. In LabelImg, we can easily do it by clicking and making boxes and typing in the lables and saving them. It offers 3 types of lables making- YOLO, PascalVOC and CreateML. YOLO being one of the most popular one, I chose to annotate them in YOLO.

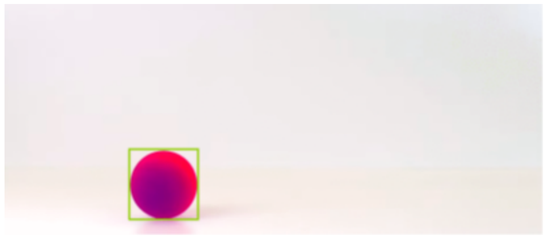
Pytorch is a library that is used to use machine learning models in Data Analysis and Computer Vision. It is a huge library can be used to used to not only to train our model, but also to use pre-trained models to analyse images.

II. Problem Statement

In a distant future, we have the power of machine learning running through our veins. However, a hunger for knowledge to know its origin, we set on a journey to unravel its ancient secrets. Finally, we find a USB drive, we read it and discover a zip folder with 100 images in it and a pamphlet of The Commandments that gives us a challenge. We are supposed to follow the instructions to make the project. The Commandments explain about the journey of Pytorch with its faithful companions ethereal cloud Colab and LabelImg.

After working out the main problem statement from the sea of metaphors, the following was understood. We have to train a Machine Learning model ourselves with the images given so that when an image like following is given,

Output like the following is generated.



However, there are some conditions.
• The whole script needs to be written in Google Colab, using the servers of google
• Among the 2 types of processors available in Colab – CPU and GPU, as explained in INTRODUCTION, we are instructed to use GPU
• Of course, we need to use Pytorch library only, although other libraries like TensorFlow are available
• Before putting the images to the model, we are supposed to LabelImg to annotate the images to mark the co-ordinates of the object to be identified i.e. ball and label them.

• We are not allowed to use pre-trained models, but train our own using the images supplied to us in the zip file.

Finally, images uploaded by the user should identify the ‘ball’ as explained in a previous paragraph.

III. Related Work

Now, Pytorch has been used to make a lot of machine learning models to make predictions in data and computer vision. The mechanism of our the Neural networks identify the required object is fascinating. Its usage is greatly found in medical instruments, security cameras and other purposes. The rough idea of how models work is given in FINAL ATTEMPT. The most basic implementation of models can be used in predicting value of a function when some values are fed into the model, by one of the simplest models Linear Regression.

I have been working on OpenCV module for a few months now, but this one is of the most challenging tasks I found. Merging Machine Learning models with OpenCV is a challenging task. But, as I did not have much knowledge of ML models, I had to first learn about a few ML models using normal arithmetic data as explained in the INITIAL ATTEMPTS section.

IV. INITIAL ATTEMPTS

As written before, I did not have much exposure to Machine Learning models before. So, I focused on understanding our models work on arithmetic data. I learnt about Linear Regression model and training it, its internal mathematics and working. I watched various tutorials to understand them whose links I have provided in REFERENCES. I have learnt about tensors, gradients, nn models. I also tried a few pre-trained models and tested them.

But, overall, I was confused about how to make the final project successful, as I was getting huge number of resources but not exactly what I wanted. After a lot of learning and finding, I realised that I need to time to understand Pytorch more deeply and finally make the project successful. So, in FINAL ATTEMPTS, I am going to explain what I did till now on it.

V. Final Approach

A typical model works like the following –

• We insert the dataset in any form

• Then, we call a model from nn of torch

• Then using the dataset, we train the model in the following way

• First feed in the data one by one, as wished, we may put in multiple layers of the Neural Network

• At certain intervals, the model is made is made to predict the value and compared with actual value of it, then the loss is calculated

• Accordingly, the weights of the data are determined by the model

• The above iterations performed huge number of times trains the model to predict the almost close to the real value

• Finally, when we give model to generate output by giving in initial values, almost correct value is given

Now, why is it almost close? Why not perfect value? Because most of the time, a few wrong values have been fed in while training. When the training process goes on, the model will ultimately decrease its weight, but won’t ultimately be eliminated. Hence, a slight error usually always remains.

Now, as in case of this project, first I had to annotate the images provided in .zip folder through LabelImg. Initially, I had problems in installing and running LabelImg as neither PyCharm or Command Prompt had ‘pip’ environment installed. Even I had installed labelImg in PyCharm, but could not open it. Finally, I could install and open it from VS Code. I annotated all the 100 images using labelImg and saved it as YOLO type of annotations in another folder.

I could not completely make the final project as I could not completely get familiar with the syntaxes yet. But the flow of code is roughly as explained above, except the dataset loaded will be the images and the annotations to train the model. Hence, overall, the data type that the model deals with is matrix. The libraries imported for training the model are torch, numpy, transform, cv2, matplotlib, etc

VI. Results and Observation

The results after successful training of model will be roughly as was displayed in PROBLEM STATEMENT.

VII. Future Work

I have to work and learn more on the topic of training and knowing about ML models and their syntaxes and implementations in Python. I hope I will be able to learn everything that is required to complete the project and also make advanced projects with Combination of ML Models and Computer Vision in the future.

CONCLUSION

Overall, I found this task challenging as this opened up a new dimension of skills that I need to explore to enhance my Computer Vision skills. As we know, AL has and will more become an important part of our lives, its utilisation in tasks and learning about it becomes necessary to solve complicated problems.

References

[1] Eli Stevens, Luca Antiga and Thomas Viehmann, forward by Soumith Chintala, ”Deep Learning with Pytorch”, Manning Publications Co., 2020

[2] Patrick Loeber, ”Deep Learning With PyTorch - Full Course”, YouTube, 2021

[3] Nicholas Renotte, ”Tensorflow Object Detection in 5 Hours with Python”, YouTube, 2021

[4] The PyTorch Foundation, ”PyTorch”