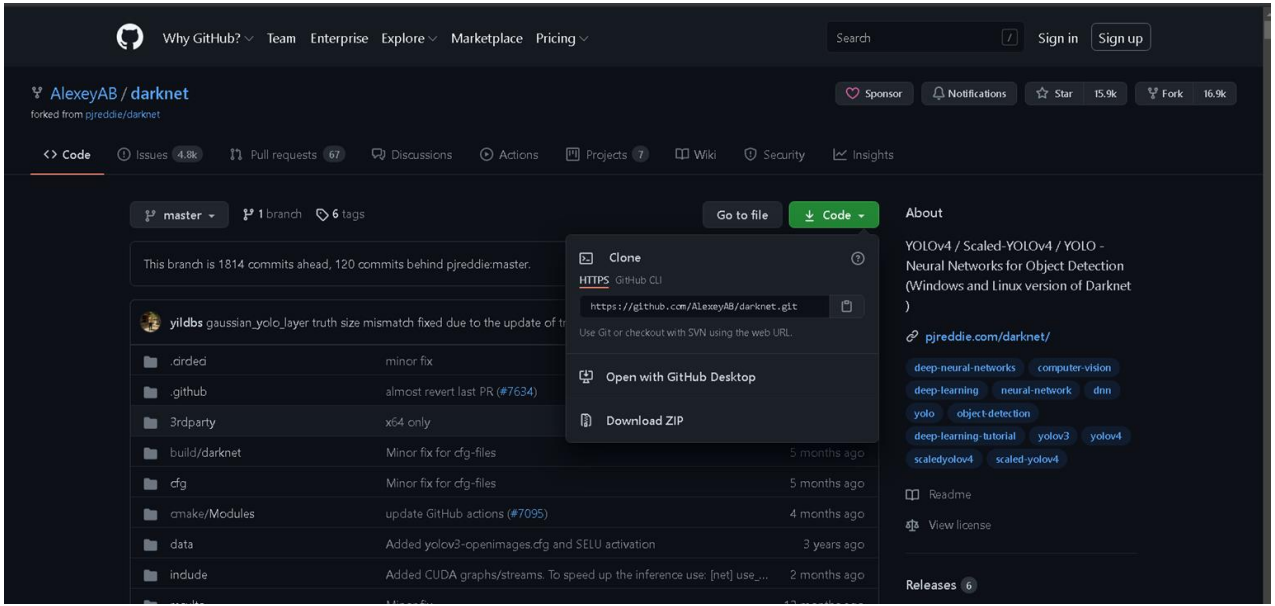
Implementasi

You Only Look Once

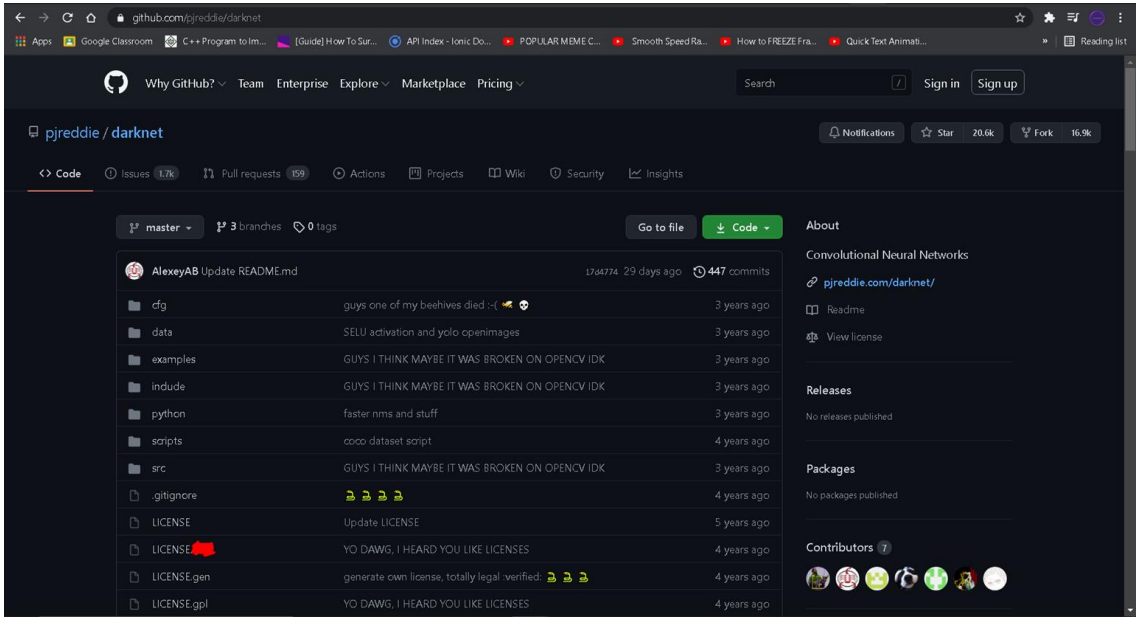
By:
William Sean C14180031
Abraham Imanuel C14180066

1. Framework

- Darknet
- https://github.com/AlexeyAB/darknet.git



1. Framework

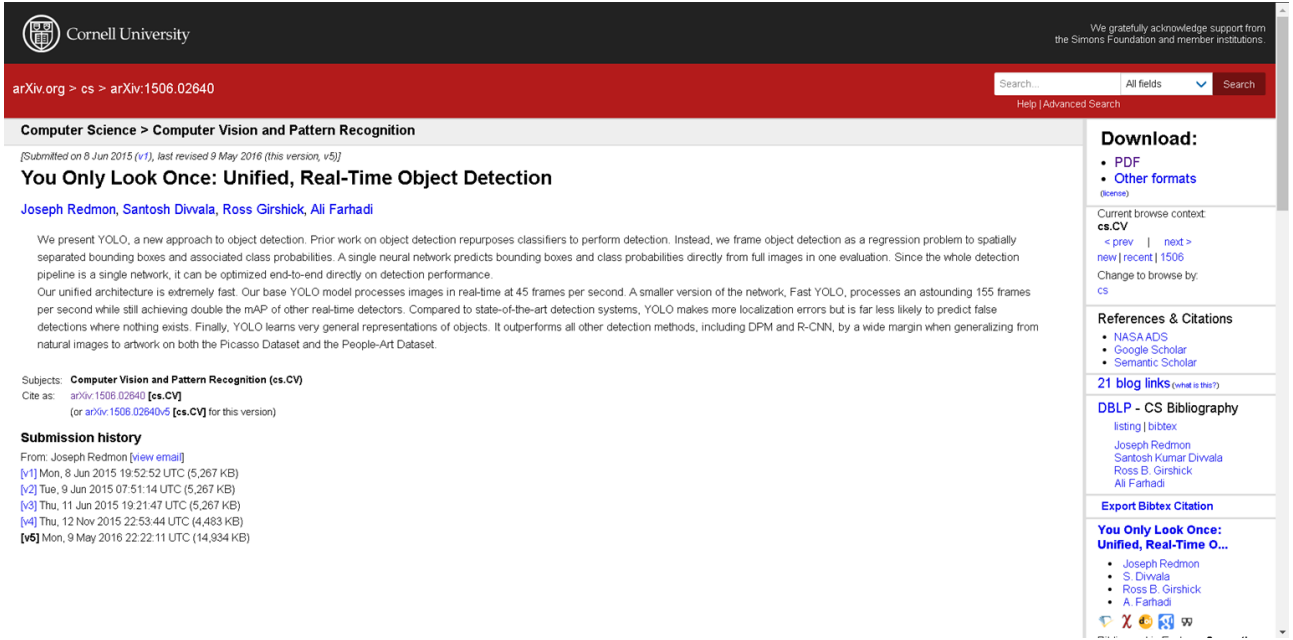

github.com/pjreddie/darknet
Why GitHub? Team Enterprise Explore Marketplace Pricing
Search
Sign in
Sign up
pjreddie / darknet
Notifications
Star 20.6k
Fork 16.9k
Code
Issues 1.7k
Pull requests 159
Actions
Projects
Wiki
Security
Insights
master
3 branches
0 tags
Go to file
Code
AlexeyAB Update README.md
1744774 29 days ago
447 commits
cfg guys one of my beehives died :-(3 years ago
data SELU activation and yolo openimages 3 years ago
examples GUYS I THINK MAYBE IT WAS BROKEN ON OPENCV IDK 3 years ago
include GUYS I THINK MAYBE IT WAS BROKEN ON OPENCV IDK 3 years ago
python faster nms and stuff 3 years ago
scripts coco dataset script 4 years ago
src GUYS I THINK MAYBE IT WAS BROKEN ON OPENCV IDK 3 years ago
.gitignore 4 years ago
LICENSE Update LICENSE 5 years ago
LICENSE YO DAWG, I HEARD YOU LIKE LICENSES 4 years ago
LICENSE.gen generate own license, totally legal :verified: 4 years ago
LICENSE.gpl YO DAWG, I HEARD YOU LIKE LICENSES 4 years ago
About
Convolutional Neural Networks
pjreddie.com/darknet/
Readme
View license
Releases
No releases published
Packages
No packages published
Contributors 7

Source Code YOLO

- http://pjreddie.com/yolo/ (Redmod, J. et al., 2016)

Paper dari YOLO

- https://arxiv.org/abs/1506.02640



Cornell University

We gratefully acknowledge support from the Simons Foundation and member institutions.

arXiv.org > cs > arXiv:1506.02640

Search... All fields Search

Help | Advanced Search

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 8 Jun 2015 (v1), last revised 9 May 2016 (this version, v5)]

You Only Look Once: Unified, Real-Time Object Detection

Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi

We present YOLO, a new approach to object detection. Prior work on object detection repurposes classifiers to perform detection. Instead, we frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance.

Our unified architecture is extremely fast. Our base YOLO model processes images in real-time at 45 frames per second. A smaller version of the network, Fast YOLO, processes an astounding 155 frames per second while still achieving double the mAP of other real-time detectors. Compared to state-of-the-art detection systems, YOLO makes more localization errors but is far less likely to predict false detections where nothing exists. Finally, YOLO learns very general representations of objects. It outperforms all other detection methods, including DPM and R-CNN, by a wide margin when generalizing from natural images to artwork on both the Picasso Dataset and the People-Art Dataset.

Subjects: **Computer Vision and Pattern Recognition (cs.CV)**
Cite as: arXiv:1506.02640 **[cs.CV]**
(or arXiv:1506.02640v5 **[cs.CV]** for this version)

Submission history

From: Joseph Redmon [view email]
[v1] Mon, 8 Jun 2015 19:52:52 UTC (5,267 KB)
[v2] Tue, 9 Jun 2015 07:51:14 UTC (5,267 KB)
[v3] Thu, 11 Jun 2015 19:21:47 UTC (5,267 KB)
[v4] Thu, 12 Nov 2015 22:53:44 UTC (4,483 KB)
[v5] Mon, 9 May 2016 22:22:11 UTC (14,934 KB)

Download:

- PDF
- Other formats (license)

Current browse context: **cs.CV**
< prev | next >
new | recent | 1506
Change to browse by: cs

References & Citations

- NASA ADS
- Google Scholar
- Semantic Scholar

21 blog links (what is this?)

DBLP - CS Bibliography
listing | bibtex
Joseph Redmon
Santosh Kumar Divvala
Ross B. Girshick
Ali Farhadi

Export Bibtex Citation

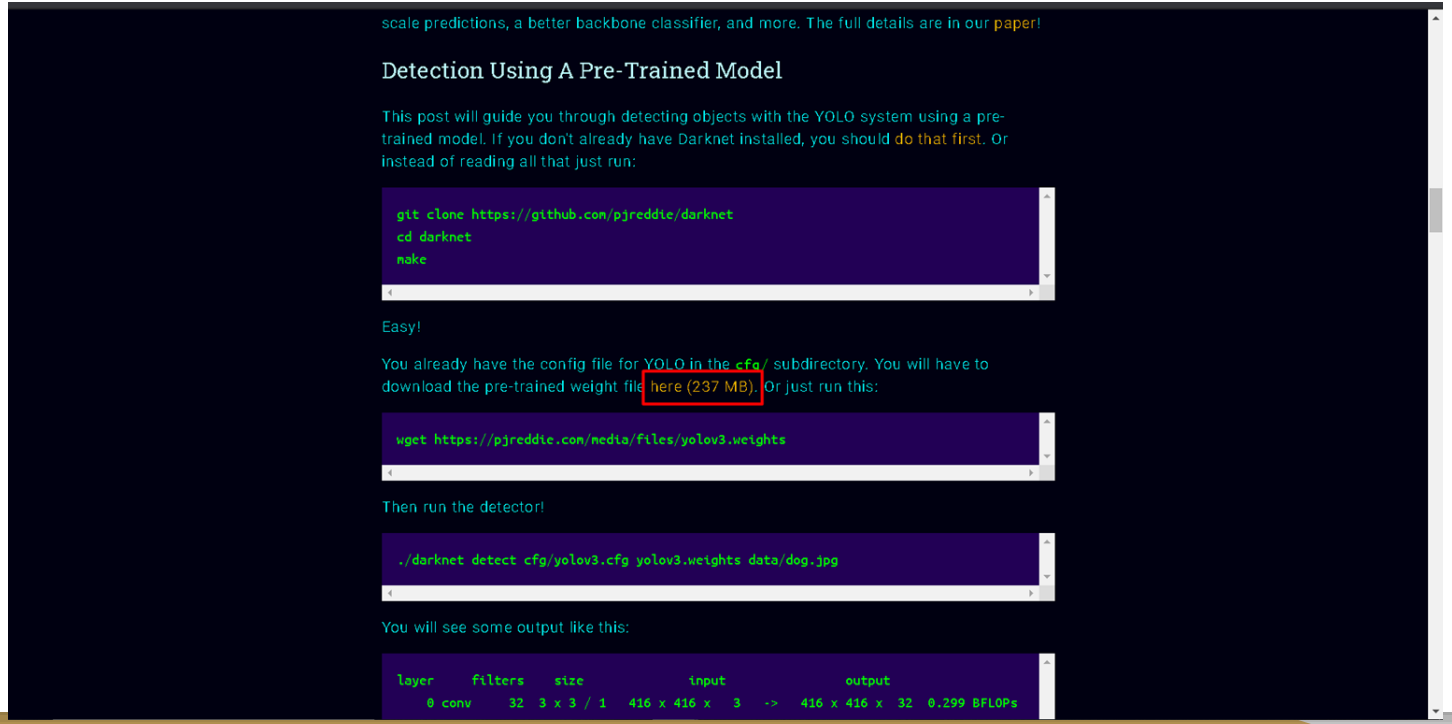
You Only Look Once: Unified, Real-Time O...

- Joseph Redmon
- S. Divvala
- Ross B. Girshick
- A. Farhadi

Requirements

- Windows or Linux
- CMake >= 3.8
- CUDA 10.0 or newer
- OpenCV (Pakai opencv-python boleh)
- cuDNN >= 7.0 for CUDA
- GPU (Untuk video)
- MSVC 2015/2017/2019

Download YOLOV3 Weights



scale predictions, a better backbone classifier, and more. The full details are in our paper!

Detection Using A Pre-Trained Model

This post will guide you through detecting objects with the YOLO system using a pre-trained model. If you don't already have Darknet installed, you should do that first. Or instead of reading all that just run:

```
git clone https://github.com/pjreddie/darknet
cd darknet
make
```

Easy!

You already have the config file for YOLO in the `cfg/` subdirectory. You will have to download the pre-trained weight file here (237 MB). Or just run this:

```
wget https://pjreddie.com/media/files/yolov3.weights
```

Then run the detector!

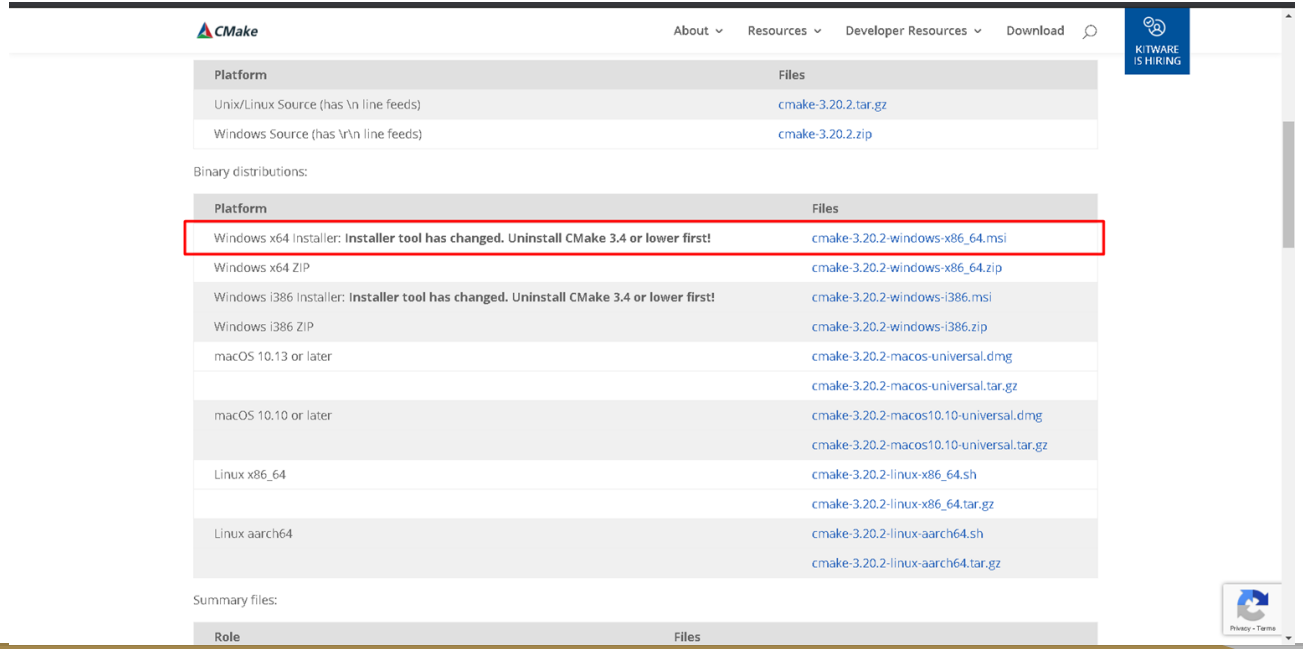
```
./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg
```

You will see some output like this:

```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32  0.299 BFLOPs
```

Download CMake

- https://cmake.org/download/



CUDA & cuDNN

- https://developer.nvidia.com/cuda-downloads (untuk CUDA)
- https://developer.nvidia.com/rdp/cudnn-download (Untuk cuDNN)

How to Check Your CUDA

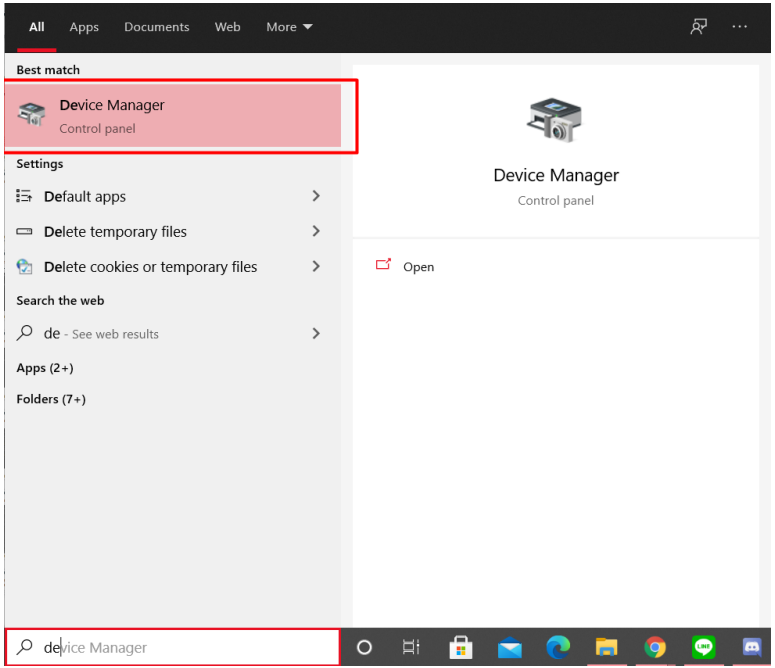
Compute capability (version)	Micro-architecture	GPUs	GeForce	Quadro, NVS	Tesla/Datacenter GPU	Tegra, Jetson, DRIVE
1.0	Tesla	G80	GeForce 8800 Ultra, GeForce 8800 GTX, GeForce 8800 GTS(G80)	Quadro FX 5600, Quadro FX 4600, Quadro Plex 2100 S4	Tesla C870, Tesla D870, Tesla S870	
1.1		G92, G94, G96, G98, G84, G86	GeForce GTS 250, GeForce 9800 GX2, GeForce 9800 GTX, GeForce 9800 GT, GeForce 8800 GTS(G92), GeForce 8800 GT, GeForce 9600 GT, GeForce 9500 GT, GeForce 9400 GT, GeForce 8600 GTS, GeForce 8600 GT, GeForce 8500 GT, GeForce G110M, GeForce 9300M GS, GeForce 9200M GS, GeForce 9100M G, GeForce 8400M GT, GeForce G105M	Quadro FX 4700 X2, Quadro FX 3700, Quadro FX 1800, Quadro FX 1700, Quadro FX 580, Quadro FX 570, Quadro FX 470, Quadro FX 380, Quadro FX 370, Quadro FX 370 Low Profile, Quadro NVS 450, Quadro NVS 420, Quadro NVS 290, Quadro NVS 295, Quadro Plex 2100 D4, Quadro FX 3800M, Quadro FX 3700M, Quadro FX 3600M, Quadro FX 2800M, Quadro FX 2700M, Quadro FX 1700M, Quadro FX 1600M, Quadro FX 770M, Quadro FX 570M, Quadro FX 370M, Quadro FX 360M, Quadro NVS 320M, Quadro NVS 160M, Quadro NVS 150M, Quadro NVS 140M, Quadro NVS 135M, Quadro NVS 130M, Quadro NVS 450, Quadro NVS 420,[40] Quadro NVS 295		
1.2		GT218, GT216, GT215	GeForce GT 340*, GeForce GT 330*, GeForce GT 320*, GeForce 315*, GeForce 310*, GeForce GT 240, GeForce GT 220, GeForce 210, GeForce GTS 360M, GeForce GTS 350M, GeForce GT 335M, GeForce GT 330M, GeForce GT 325M, GeForce GT 240M, GeForce G210M, GeForce 310M, GeForce 305M	Quadro FX 380 Low Profile, Quadro FX 1800M, Quadro FX 880M, Quadro FX 380M, Nvidia NVS 300, NVS 5100M, NVS 3100M, NVS 2100M, ION		
1.3		GT200, GT200b	GeForce GTX 295, GTX 285, GTX 280, GeForce GTX 275, GeForce GTX 260	Quadro FX 5800, Quadro FX 4800, Quadro FX 4800 for Mac, Quadro FX 3800, Quadro CX, Quadro Plex 2200 D2	Tesla C1060, Tesla S1070, Tesla M1060	
2.0	Fermi	GF100, GF110	GeForce GTX 590, GeForce GTX 580, GeForce GTX 570, GeForce GTX 480, GeForce GTX 470, GeForce GTX 465, GeForce GTX 480M	Quadro 6000, Quadro 5000, Quadro 4000, Quadro 4000 for Mac, Quadro Plex 7000, Quadro 5010M, Quadro 5000M	Tesla C2075, Tesla C2050/C2070, Tesla M2050/M2070/M2075/M2090	
2.1		GF104, GF106 GF108, GF114, GF116, GF117, GF119	GeForce GTX 560 Ti, GeForce GTX 550 Ti, GeForce GTX 460, GeForce GTS 450, GeForce GTS 450*, GeForce GT 640 (GDDR3), GeForce GT 630, GeForce GT 620, GeForce GT 610, GeForce GT 520, GeForce GT 440, GeForce GT 440*, GeForce GT 430, GeForce GT 430*, GeForce GT 420*, GeForce GTX 675M, GeForce GTX 670M, GeForce GT 635M, GeForce GT 630M, GeForce GT 625M, GeForce GT 720M, GeForce GT 620M, GeForce 710M, GeForce 610M, GeForce 820M, GeForce GTX 580M, GeForce GTX 570M, GeForce GTX 560M, GeForce GT 555M, GeForce GT 550M, GeForce GT 540M, GeForce GT 525M, GeForce GT 520MX, GeForce GT 520M, GeForce GTX 485M, GeForce GTX 470M, GeForce GTX 460M, GeForce GT 445M, GeForce GT 435M, GeForce GT 420M, GeForce GT 415M, GeForce 710M, GeForce 410M	Quadro 2000, Quadro 2000D, Quadro 600, Quadro 4000M, Quadro 3000M, Quadro 2000M, Quadro 1000M, NVS 310, NVS 315, NVS 5400M, NVS 5200M, NVS 4200M		

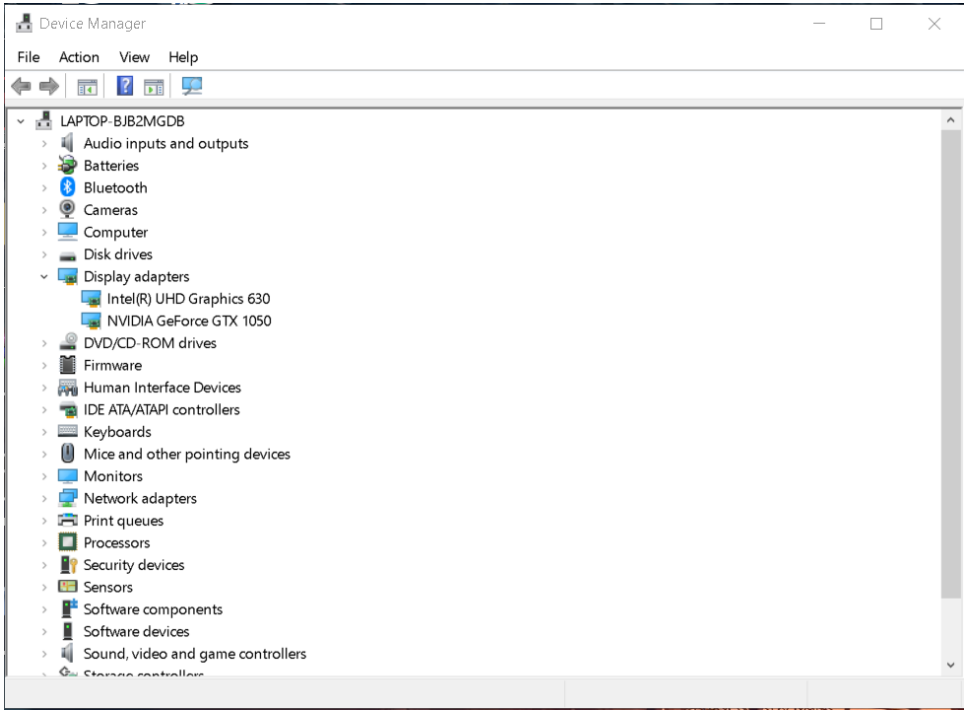
How to Check Your CUDA

						DRIVE PX
6.0	Pascal	GP100		Quadro GP100	Tesla P100	
6.1		GP102, GP104, GP106, GP107, GP108	Nvidia TITAN Xp, Titan X, GeForce GTX 1080 Ti, GTX 1080, GTX 1070 Ti, GTX 1070, GTX 1060, GTX 1050 Ti, GTX 1050, GT 1030, GT 1010, MX350, MX330, MX250, MX230, MX150, MX130, MX110	Quadro P6000, Quadro P5000, Quadro P4000, Quadro P2200, Quadro P2000, Quadro P1000, Quadro P400, Quadro P500, Quadro P520, Quadro P600, Quadro P5000(Mobile), Quadro P4000(Mobile), Quadro P3000(Mobile)	Tesla P40, Tesla P6, Tesla P4	
6.2		GP10B[41]				Tegra X2, Jetson TX2, DRIVE PX 2

- CUDA SDK 8.0 support for compute capability 2.0 – 6.x (Fermi, Kepler, Maxwell, Pascal). Last version with support for compute capability 2.x (Fermi) (Pascal GTX 1070Ti Not Supported).
- CUDA SDK 9.0 – 9.2 support for compute capability 3.0 – 7.2 (Kepler, Maxwell, Pascal, Volta) (Pascal GTX 1070Ti Not Supported. CUDA SDK 9.0 and support CUDA SDK 9.2).
- CUDA SDK 10.0 – 10.2 support for compute capability 3.0 – 7.5 (Kepler, Maxwell, Pascal, Volta, Turing). Last version with support for compute capability 3.x (Kepler). 10.2 is the last official release for macOS, as support will not be available for macOS in newer releases.
- CUDA SDK 11.0 support for compute capability 3.5 – 8.0 (Kepler (in part), Maxwell, Pascal, Volta, Turing, Ampere (in part)).[37] New data types: Bfloat16 and TF32 on third-generations Tensor Cores.[38]
- CUDA SDK 11.1 – 11.3 support for compute capability 3.5 – 8.6 (Kepler (in part), Maxwell, Pascal, Volta, Turing, Ampere).[39]

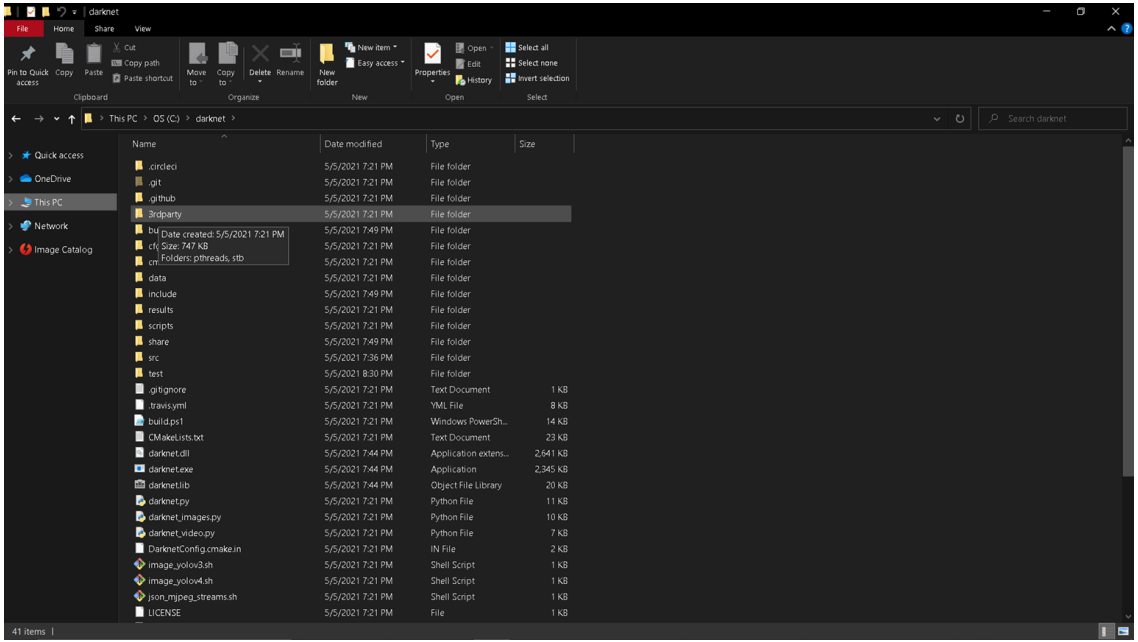
How to Check GPU



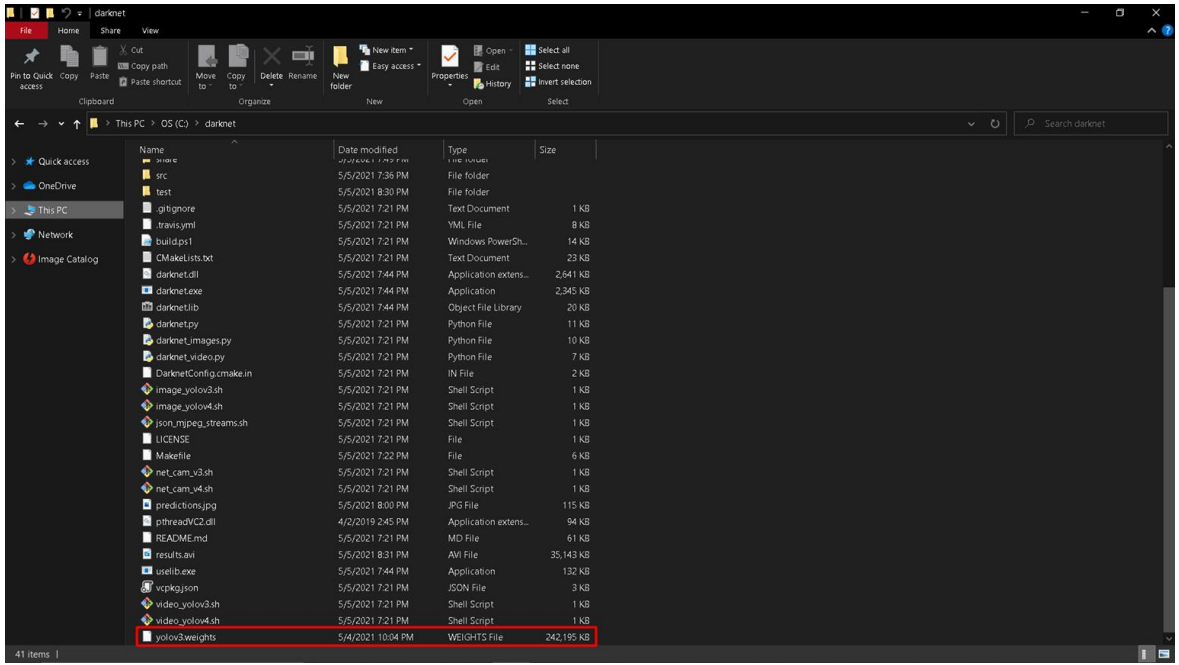


Step 1 - Clone the Framework Repos

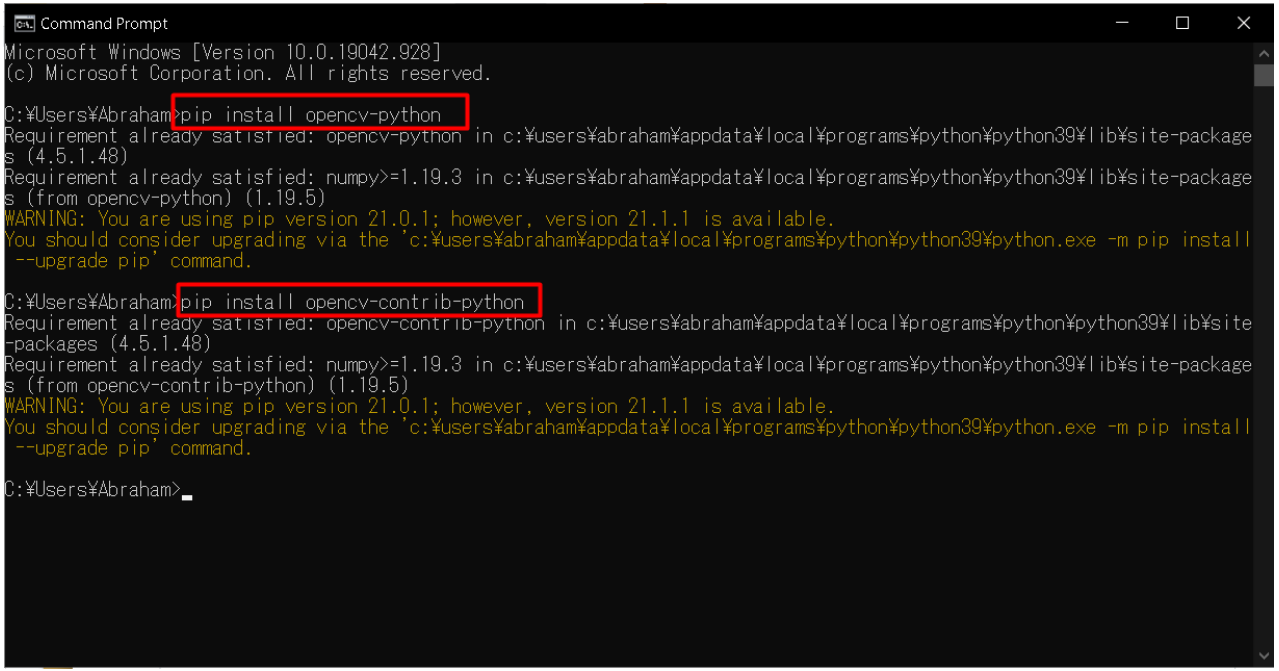
- git clone https://github.com/AlexeyAB/darknet.git



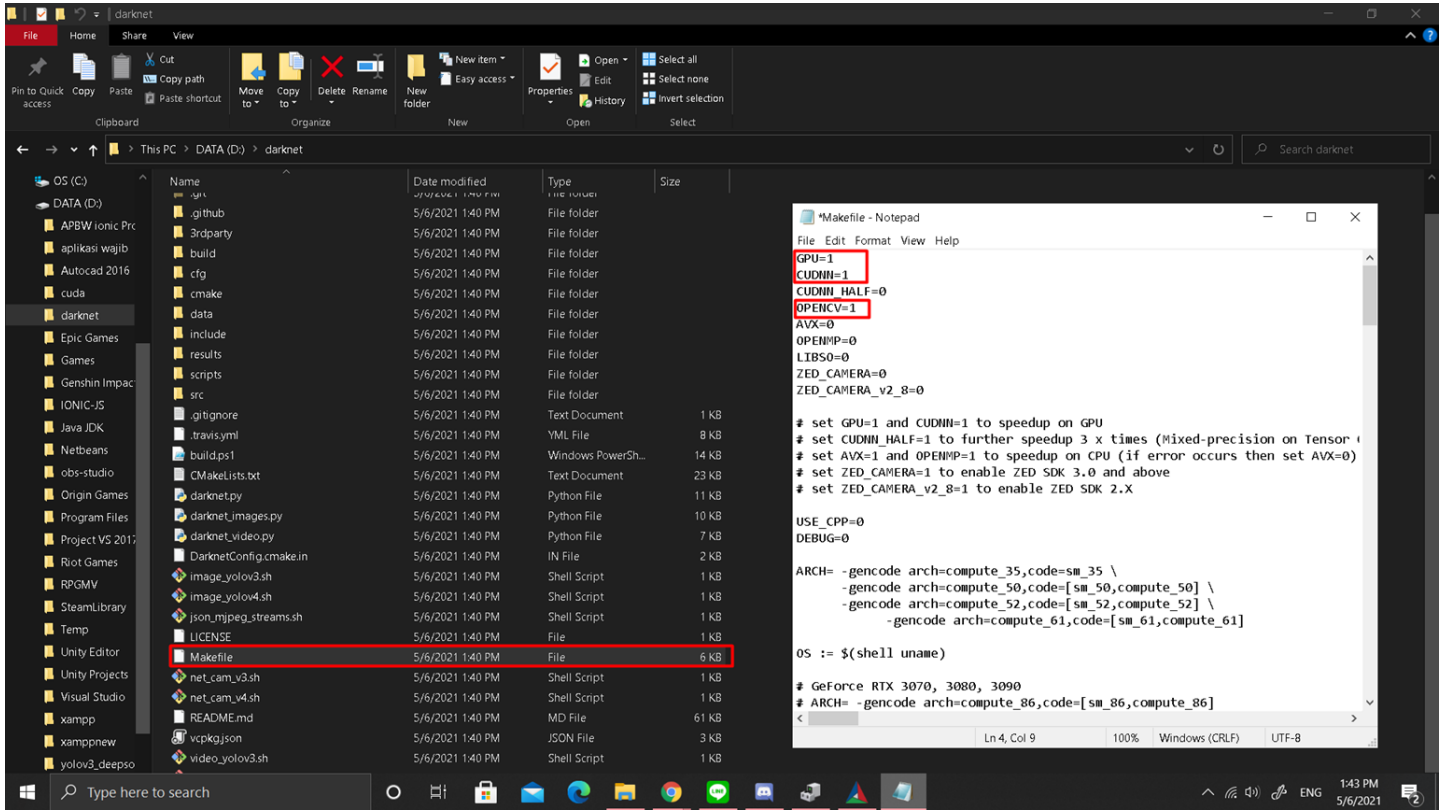
Step 2 - Put Weights File in Repos



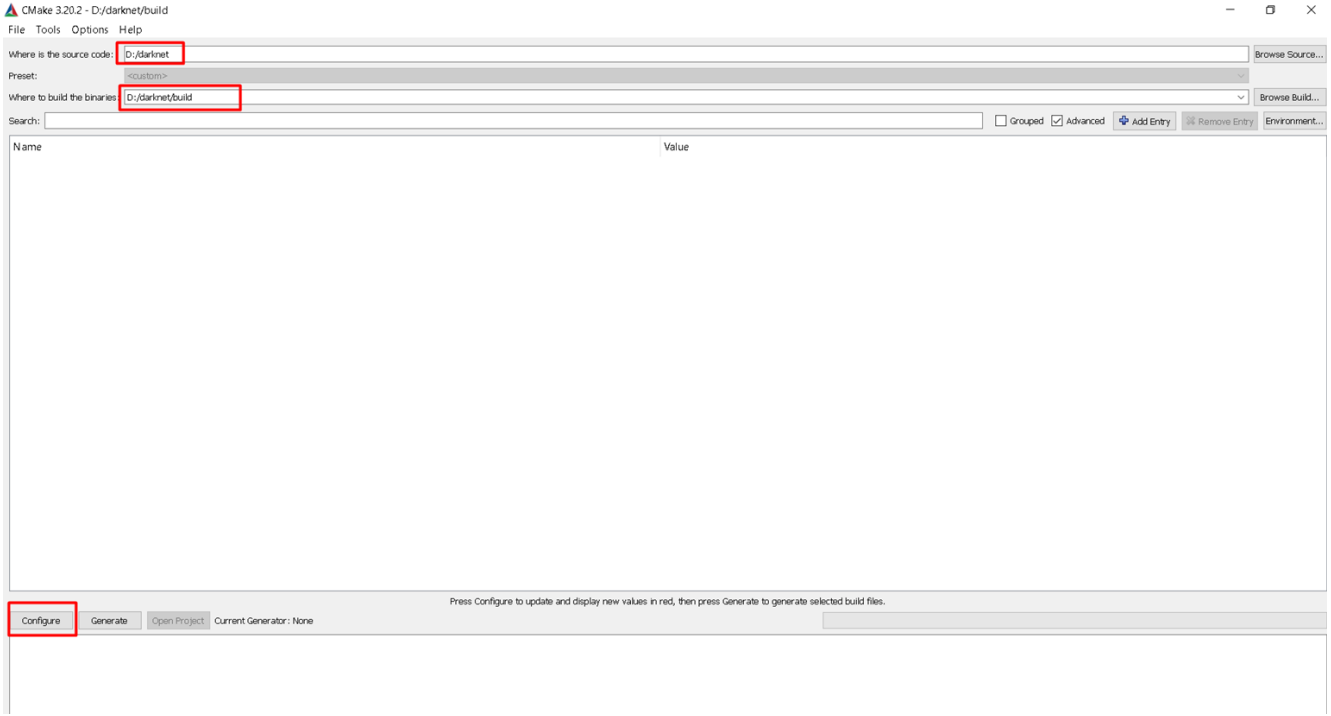
Step 3 - Install OpenCV



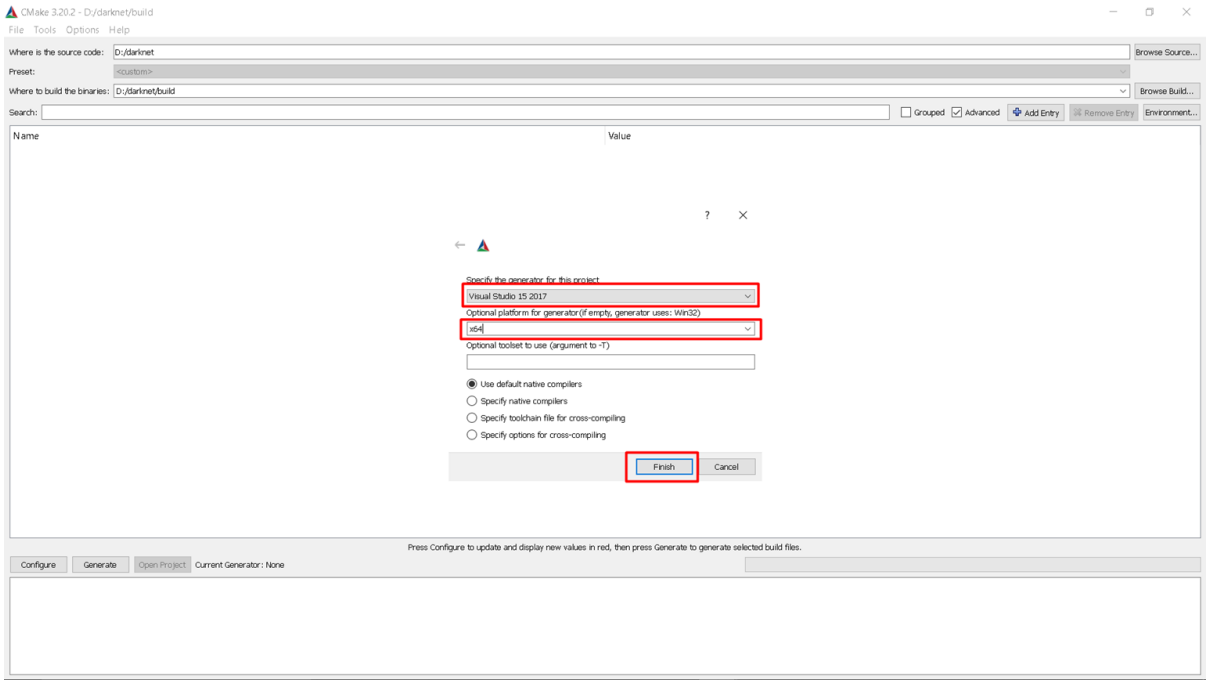
Step 4 - Change “Makefile” file



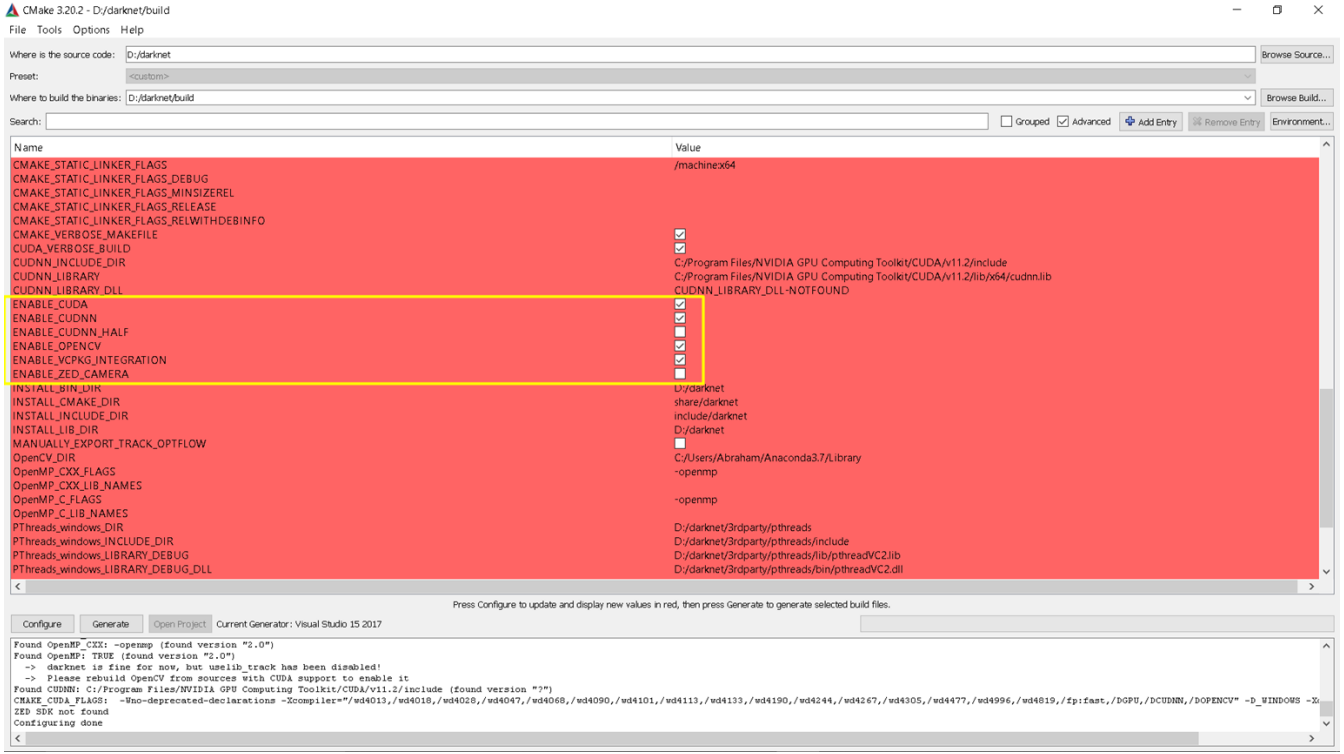
Step 5 - Open CMake, Choose Source Repos and Build repos, and Configure



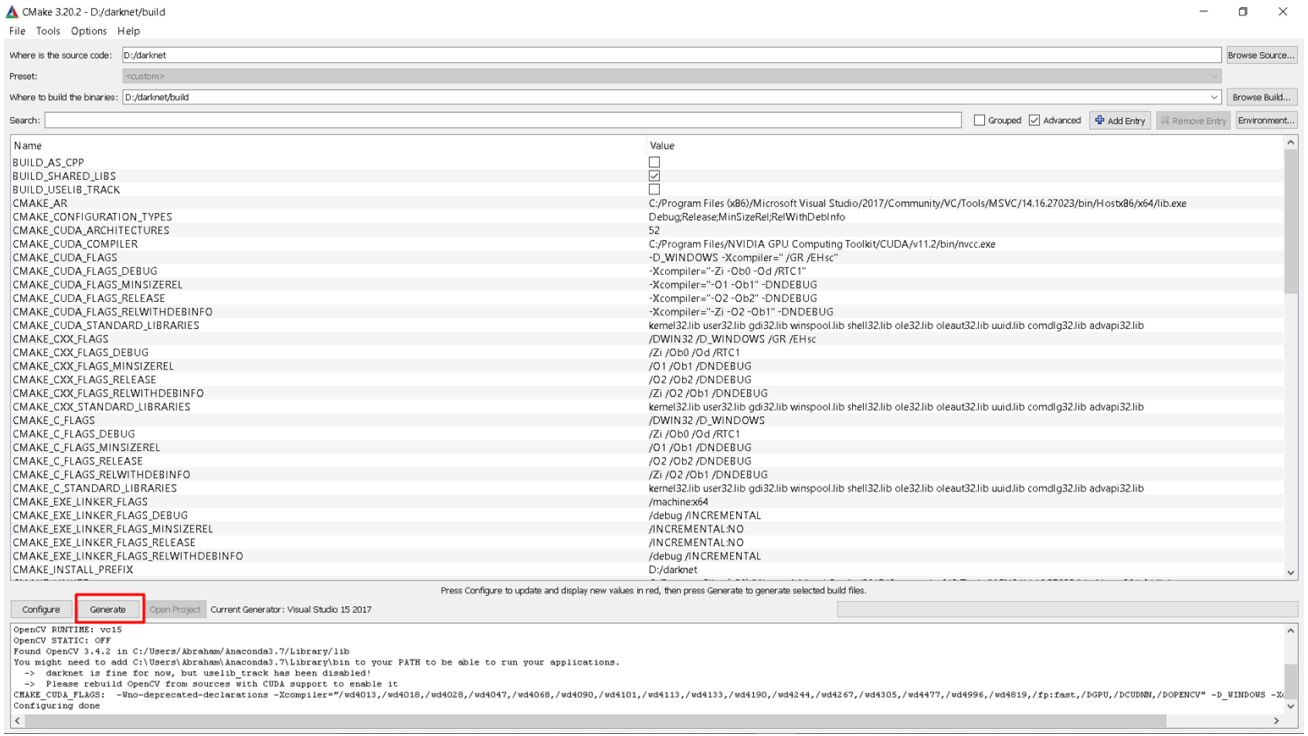
Step 6 - Use Cmake to Configure and Generate Build Code



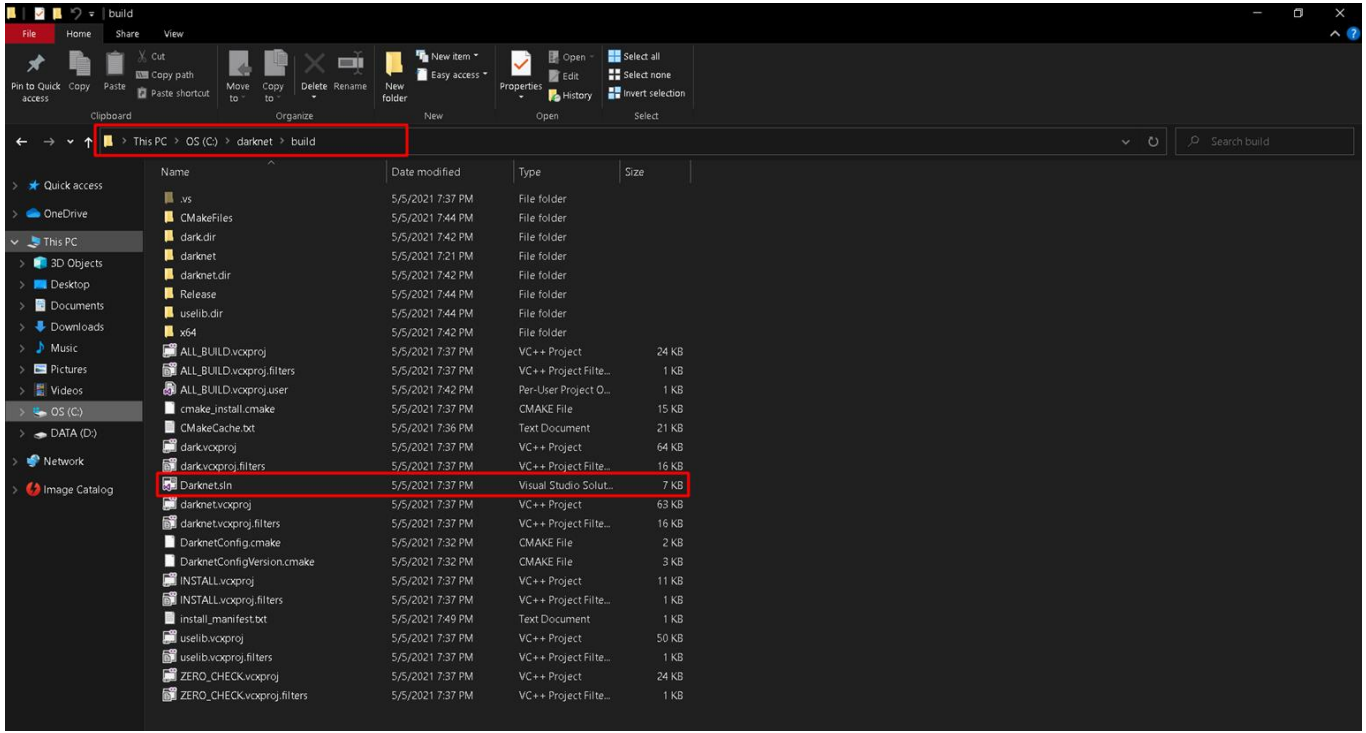
Step 7 - Change some value and Configure again



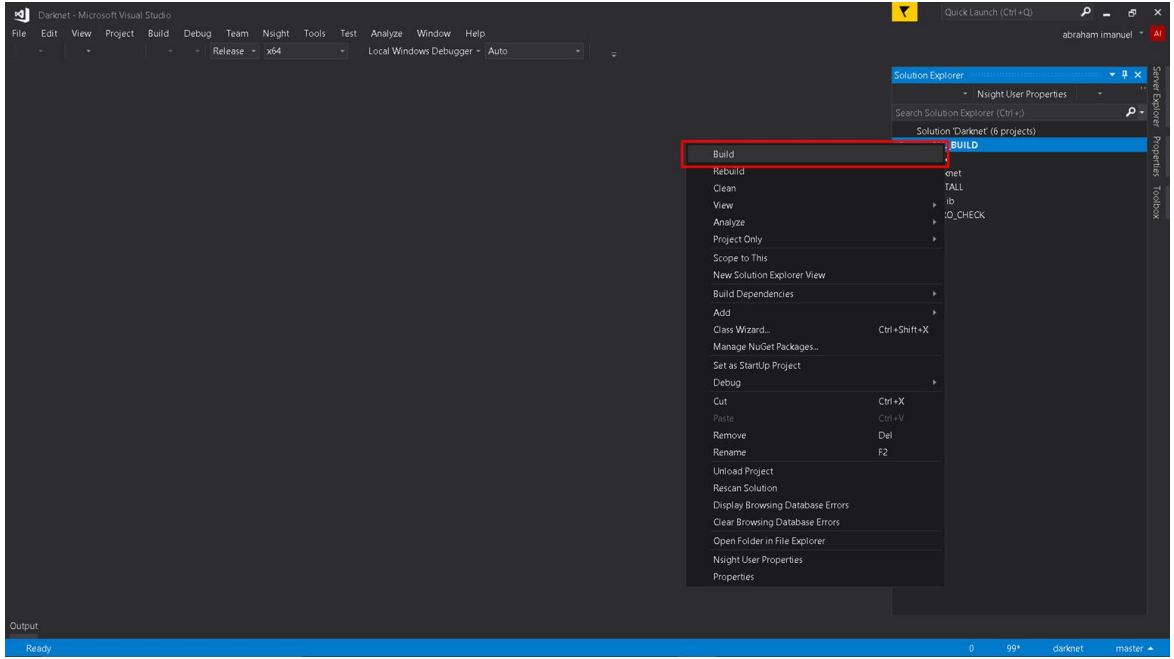
Step 8 - Make sure there aren't any reds and Generate Code



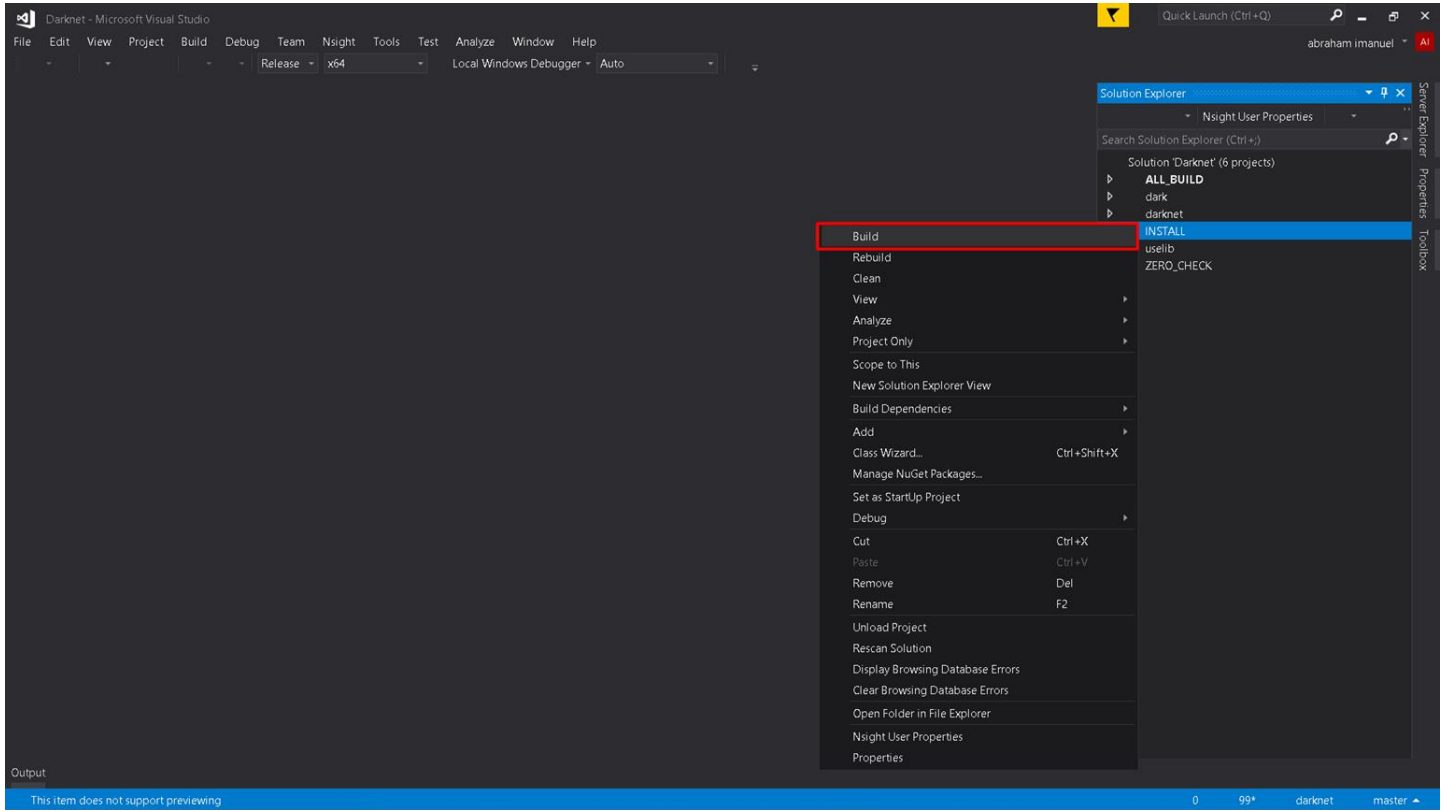
Step 9 - Go to Build Folder and open Darknet.sln



Step 10 - Build “All_Build”



Step 11 - Build “INSTALL”



Step 12 - Demo