

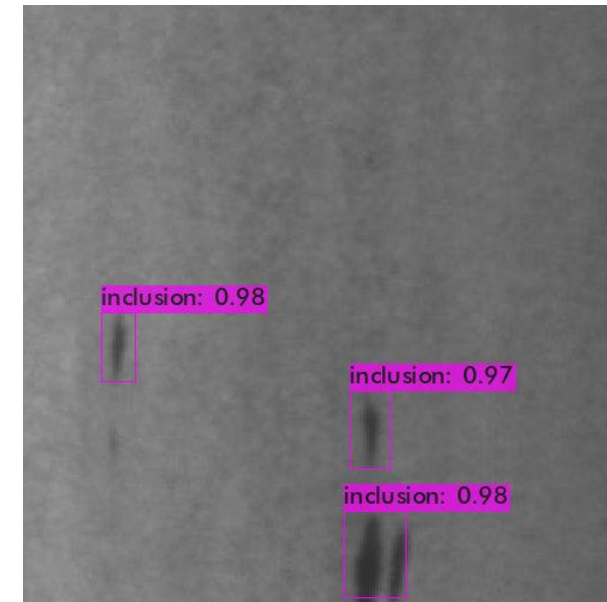
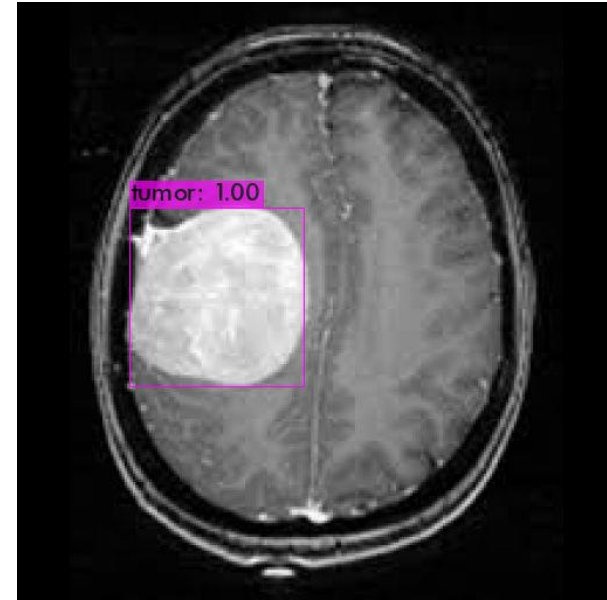
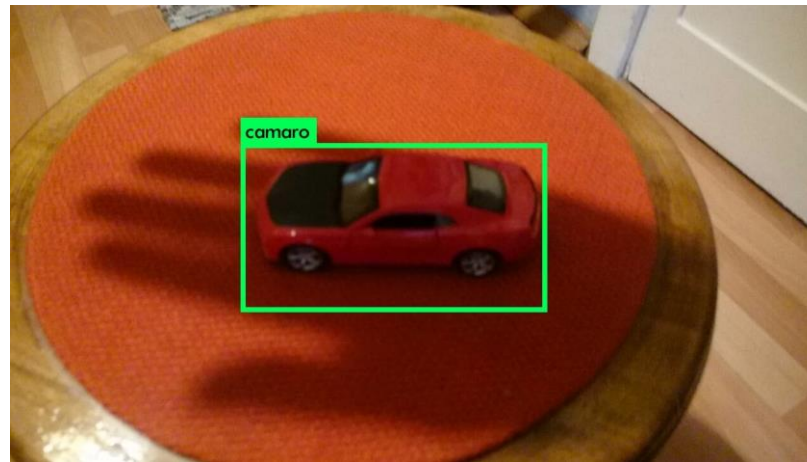
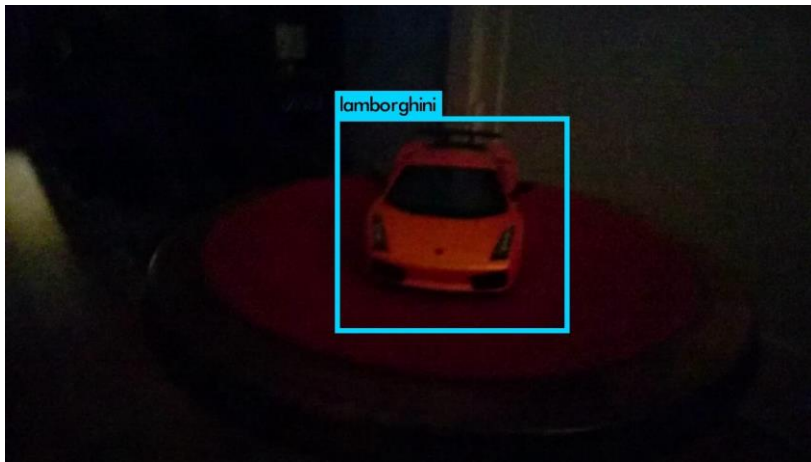
Image-Object-Detection-YOLOv4- CPP-Jupyter

Russian Alexey Bochkovskiy, the maintainer of YOLO Darknet, found that the CSPNet detector developed by Wang Jianyao, the post-doc of the Chinese Academy of Sciences and director Liao Hongyuan, was fast and good, so he invited the Chinese Academy of Sciences to develop YOLOv4 for the backbone, and did various parts of the previous generation of YOLOv3. The improvement can not only maintain a certain detection speed, but also greatly improve the detection accuracy and reduce the usage of hardware.

Version 20230223

Applications

- YOLOv4 solutions can be applied to factory defect detection, medical image analysis, biological image analysis, industrial security image analysis, mask image analysis, luxury car brand image analysis, etc.



How to use

The main process is:

Annotate images -> Prepare files for training -> Training -> Inference

🏠 / Jupyter-Image-Object-Detection-YOLOv4-CPP-14 /

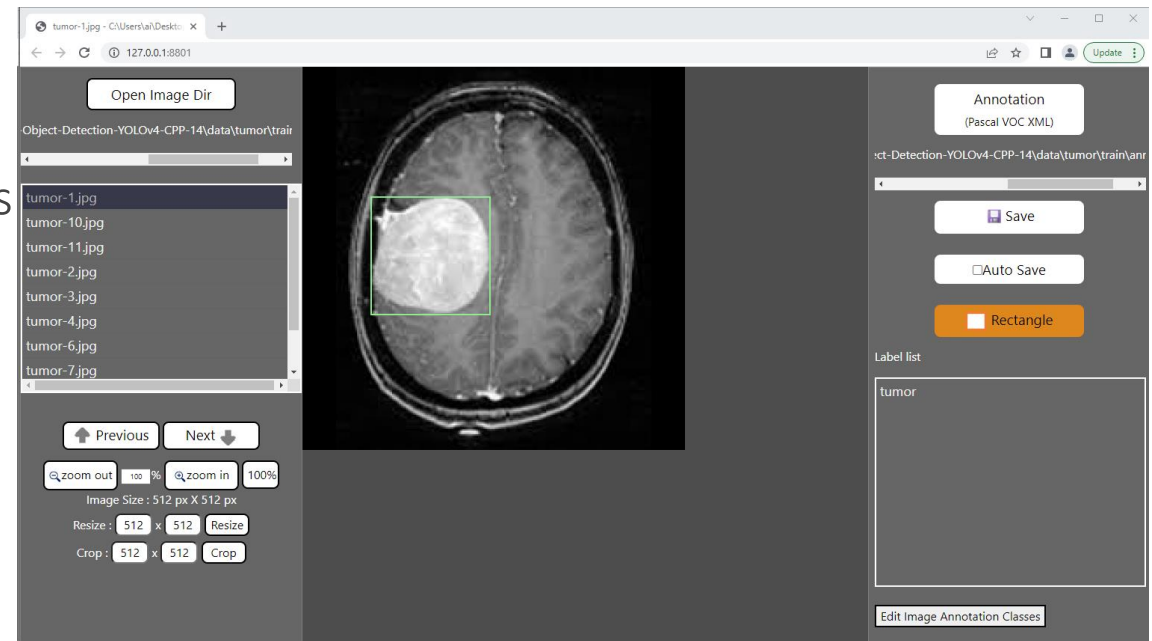
Name
bin
data
src
1_annotation_pascal_voc_xml.ipynb
2_calculate_anchors_GPU.ipynb
3_convert_yolo_format.ipynb
4_prepare_train_txt.ipynb
5_prepare_val_txt.ipynb
6_prepare_config_file.ipynb
7_train_CPU.ipynb
7_train_GPU.ipynb
8_inference_CPU.ipynb
8_inference_GPU.ipynb
9_inference_webcam_CPU.ipynb
9_inference_webcam_GPU.ipynb
10_inference_folder_1_CPU.ipynb
10_inference_folder_1_GPU.ipynb
11_inference_folder_demo_CPU.ipynb
11_inference_folder_demo_GPU.ipynb
12_YOLOv4_auto_labeling_GPU.ipynb

1_annotation_pascal_voc_xml.ipynb

Open the webpage for image annotation.

ipynb parameter:

- “port” is the port used by the webpage. If the port is occupied by the user, please change another port value by yourself.
- “image_folder” is the image path.
- “annotation_path” is the path to the annotation archive.



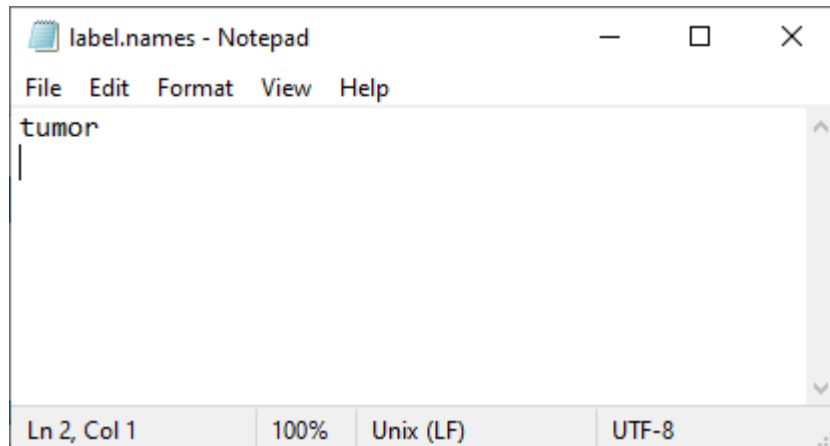
See Annotation.pdf for how to use annotation pages.

2_calculate_anchors_GPU.ipynb

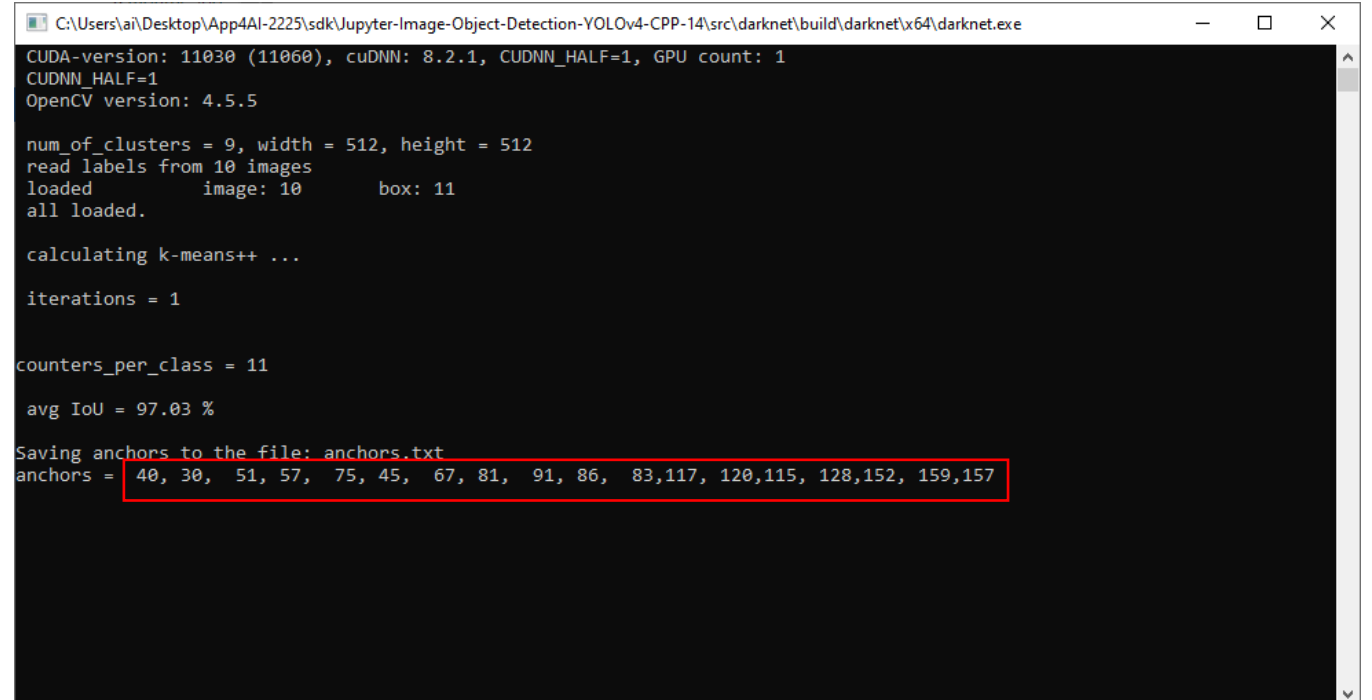
Calculate the anchor value suitable for your dataset. Before running, please confirm the label.names in your dataset and whether the category filled in the content is correct.

supplement:

The content of label.names is the category name, excluding background, and the format distinguishes multiple categories by wrapping lines.



```
label.names - Notepad
File Edit Format View Help
tumor
|
Ln 2, Col 1 100% Unix (LF) UTF-8
```



```
C:\Users\ai\Desktop\App4AI-2225\sdk\Jupyter-Image-Object-Detection-YOLOv4-CPP-14\src\darknet\build\darknet\x64\darknet.exe
CUDA-version: 11030 (11060), cuDNN: 8.2.1, CUDNN_HALF=1, GPU count: 1
CUDNN_HALF=1
OpenCV version: 4.5.5

num_of_clusters = 9, width = 512, height = 512
read labels from 10 images
loaded image: 10 box: 11
all loaded.

calculating k-means++ ...

iterations = 1

counters_per_class = 11

avg IoU = 97.03 %

Saving anchors to the file: anchors.txt
anchors = 40, 30, 51, 57, 75, 45, 67, 81, 91, 86, 83,117, 120,115, 128,152, 159,157
```

Set yolov4.cfg anchor parameters

Fill in the value generated after running 2_calculate_anchors_GPU.ipynb into the yolov4.cfg anchor point parameter in the dataset dataset.

The image shows three Notepad windows displaying the configuration file `yolov4.cfg`. The top window shows the initial configuration with the `anchors` parameter set to a list of 18 values: `anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192, 243, 459, 401`. The middle window shows the same configuration but with the `filters` parameter updated to `filters=18#255` and the `mask` parameter updated to `mask = 6,7,8`. The bottom window shows the final configuration with the `mask` parameter updated to `mask = 3,4,5`. Blue arrows point from a text box "Fill in the value displayed in the CMD" to the `anchors` parameter in each of the three windows, indicating that the values should be updated based on the output of a command.

```
File Edit Format View Help
yolov4.cfg - Notepad
pad=1
#filters =(classes + 5)x3
filters=18
activation=linear

[yolo]
mask = 0,1,2
anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192, 243, 459, 401
classes=1
num=9
jitter=.3
```

```
File Edit Format View Help
yolov4.cfg - Notepad
stride=1
pad=1
#filters =(classes + 5)x3
filters=18#255
activation=linear

[yolo]
mask = 6,7,8
anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192, 243, 459, 401
classes=1
num=9
```

```
File Edit Format View Help
yolov4.cfg - Notepad
pad=1
#filters =(classes + 5)x3
filters=18#255
activation=linear

[yolo]
mask = 3,4,5
anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192, 243, 459, 401
classes=1
num=9
jitter=.3
```

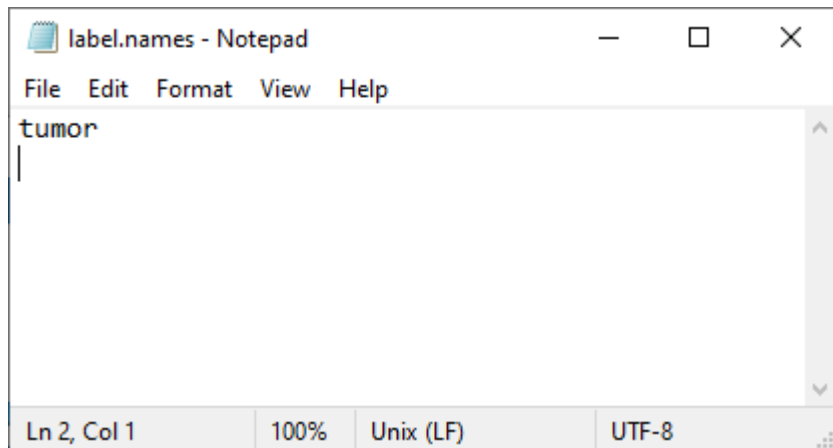
Fill in the value displayed in the CMD

3_convert_yolo_format.ipynb

Convert the voc xml label file to the yolo format. Before running, please confirm label.names under the label_file path in #parameters and whether the content filled in the category is correct.

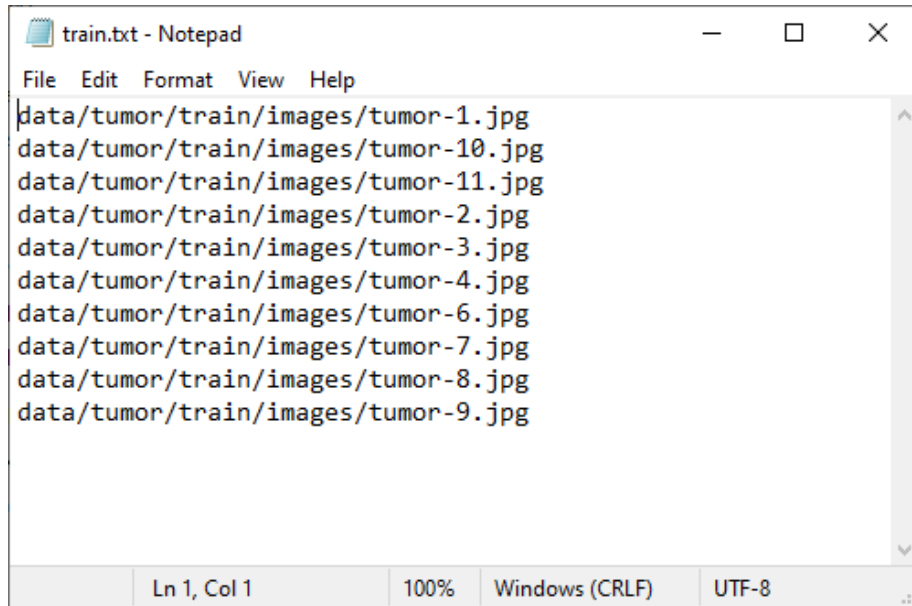
supplement:

The content of label.names is the category name without background.



4_prepare_train_txt.ipynb

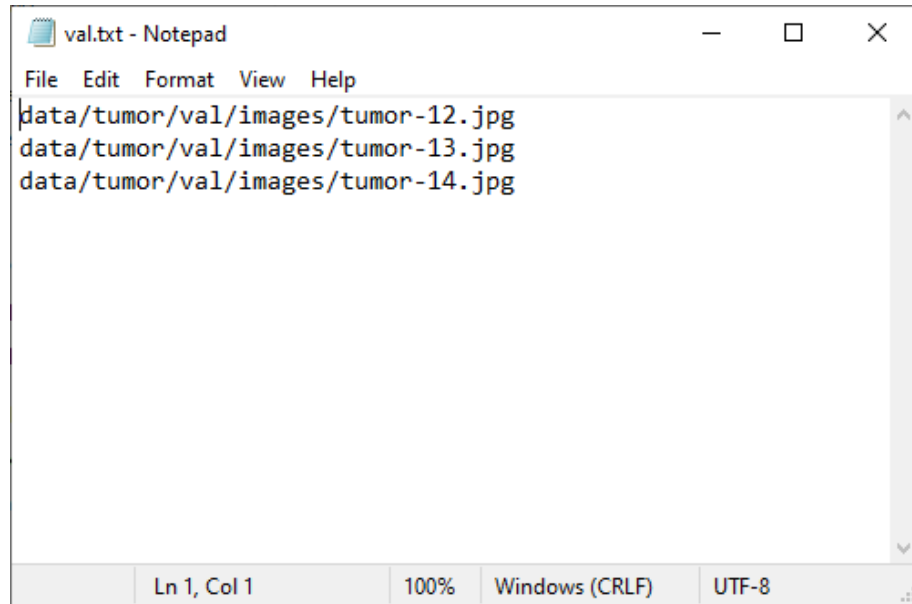
Generate training image path files “train.txt”.



```
train.txt - Notepad
File Edit Format View Help
data/tumor/train/images/tumor-1.jpg
data/tumor/train/images/tumor-10.jpg
data/tumor/train/images/tumor-11.jpg
data/tumor/train/images/tumor-2.jpg
data/tumor/train/images/tumor-3.jpg
data/tumor/train/images/tumor-4.jpg
data/tumor/train/images/tumor-6.jpg
data/tumor/train/images/tumor-7.jpg
data/tumor/train/images/tumor-8.jpg
data/tumor/train/images/tumor-9.jpg
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```


5_prepare_val_txt.ipynb

Generate validation image path files “val.txt”.



A screenshot of a Notepad window titled "val.txt - Notepad". The window contains three lines of text, each representing a file path for a validation image:

```
data/tumor/val/images/tumor-12.jpg  
data/tumor/val/images/tumor-13.jpg  
data/tumor/val/images/tumor-14.jpg
```

The status bar at the bottom of the window shows "Ln 1, Col 1", "100%", "Windows (CRLF)", and "UTF-8".

6_prepare_config_file.ipynb

Set ipynb parameters, such as dataset name, number of categories.

```
import os

# parameters
classes= 1
train = 'data/tumor/train.txt'
valid = 'data/tumor/valid.txt'
names = 'data/tumor/label.names'
backup = 'data/tumor/model'
filename = 'data/tumor/voc.data'
```

Set the number of labels

set dataset name

Set training yolov4.cfg related parameters

Set the parameters related to the number of file types in the yolov4.cfg file in the dataset.

The image displays three Notepad windows showing the configuration of a YOLOv4 training file (yolov4.cfg). The windows illustrate how to set parameters related to the number of categories in the dataset.

Top Window: Shows the configuration for the [yolo] section. The `filters=18` parameter is highlighted in red. A red arrow points from this line to the text: "Set to (number of categories + 5) x 3 example: Suppose there is a label to train So (1 + 5) x 3 = 18". The `classes=1` parameter is highlighted in blue. A blue arrow points from this line to the text: "Set the number of categories example: Suppose there is a label to train So fill in 1".

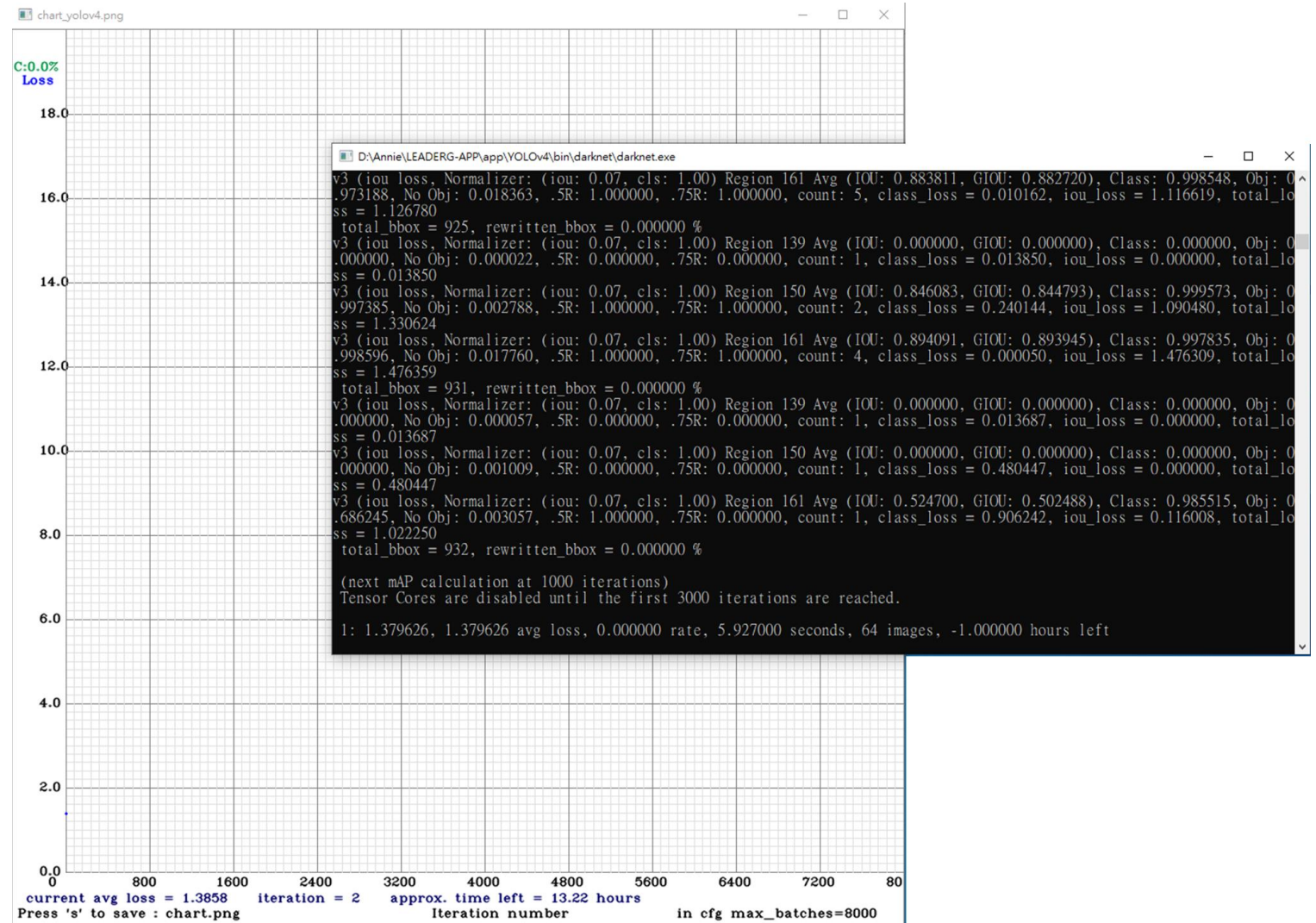
Middle Window: Shows the configuration for the [yolo] section. The `filters=18#255` parameter is highlighted in red. A red arrow points from this line to the text: "Set to (number of categories + 5) x 3 example: Suppose there is a label to train So (1 + 5) x 3 = 18". The `classes=1` parameter is highlighted in blue. A blue arrow points from this line to the text: "Set the number of categories example: Suppose there is a label to train So fill in 1".

Bottom Window: Shows the configuration for the [yolo] section. The `filters=18#255` parameter is highlighted in red. A red arrow points from this line to the text: "Set to (number of categories + 5) x 3 example: Suppose there is a label to train So (1 + 5) x 3 = 18". The `classes=1` parameter is highlighted in blue. A blue arrow points from this line to the text: "Set the number of categories example: Suppose there is a label to train So fill in 1".

7_train_GPU.ipynb

Start training.

The training is divided into two types: GPU and CPU. You can choose GPU or CPU mode according to your own hardware.



8_inference_GPU.ipynb

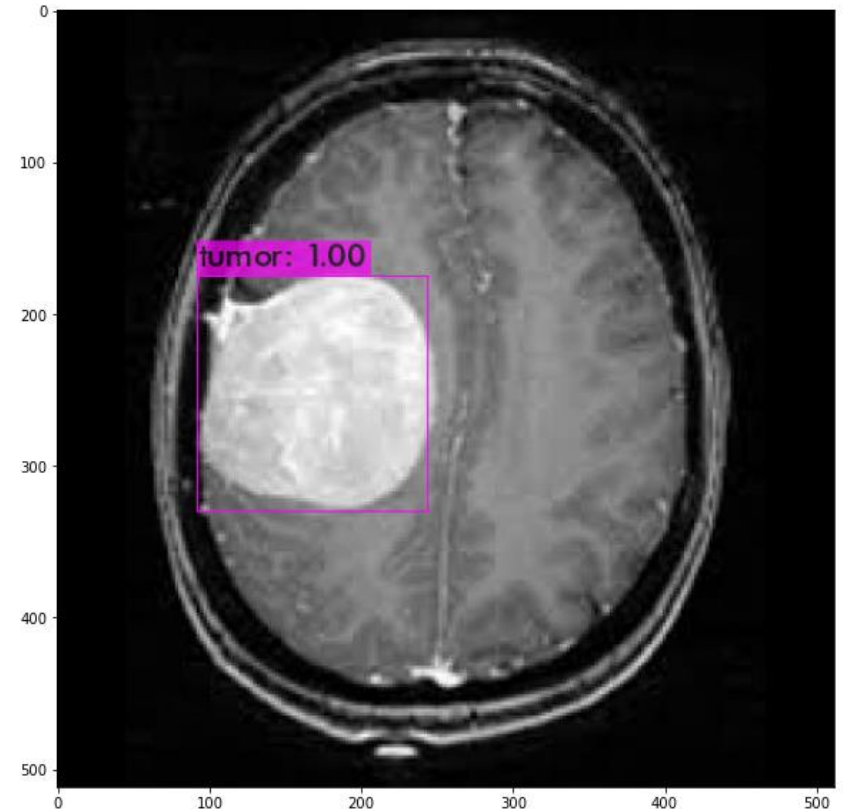
Infer a single image.

You can choose GPU or CPU mode according to your own hardware.

```
[6]: import matplotlib.pyplot as plt  
plt.rcParams["figure.figsize"] = (10, 10)
```

```
[7]: src = Image.open('predictions.jpg')  
plt.figure("src")  
plt.imshow(src)
```

```
[7]: <matplotlib.image.AxesImage at 0x28bbfa33bb0>
```

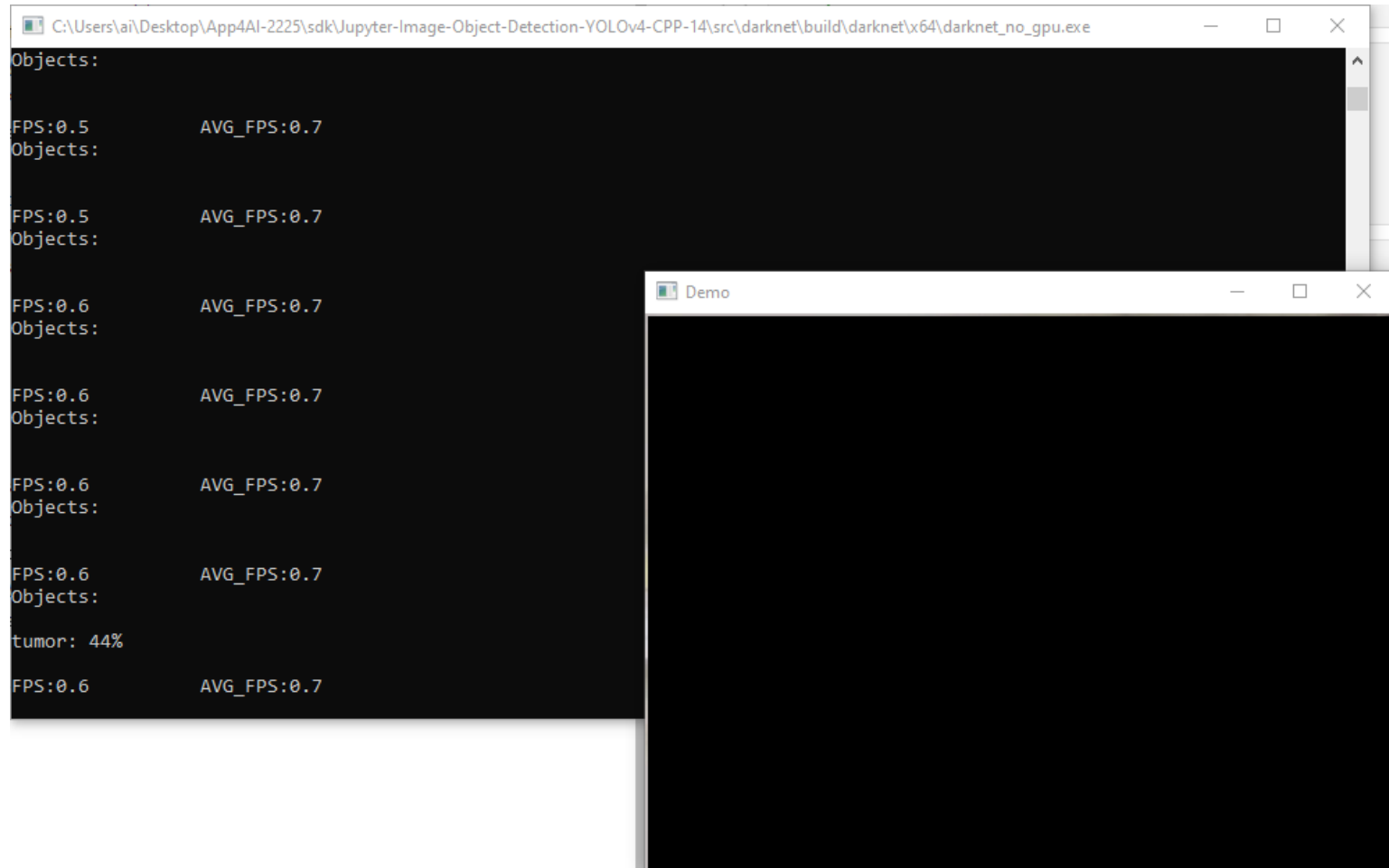


9_inference_webcam_GPU.ipynb

Infer the image of the webcam.

You can choose GPU or CPU mode according to your own hardware.

Press the “Esc” key on the display to turn the webcam off.

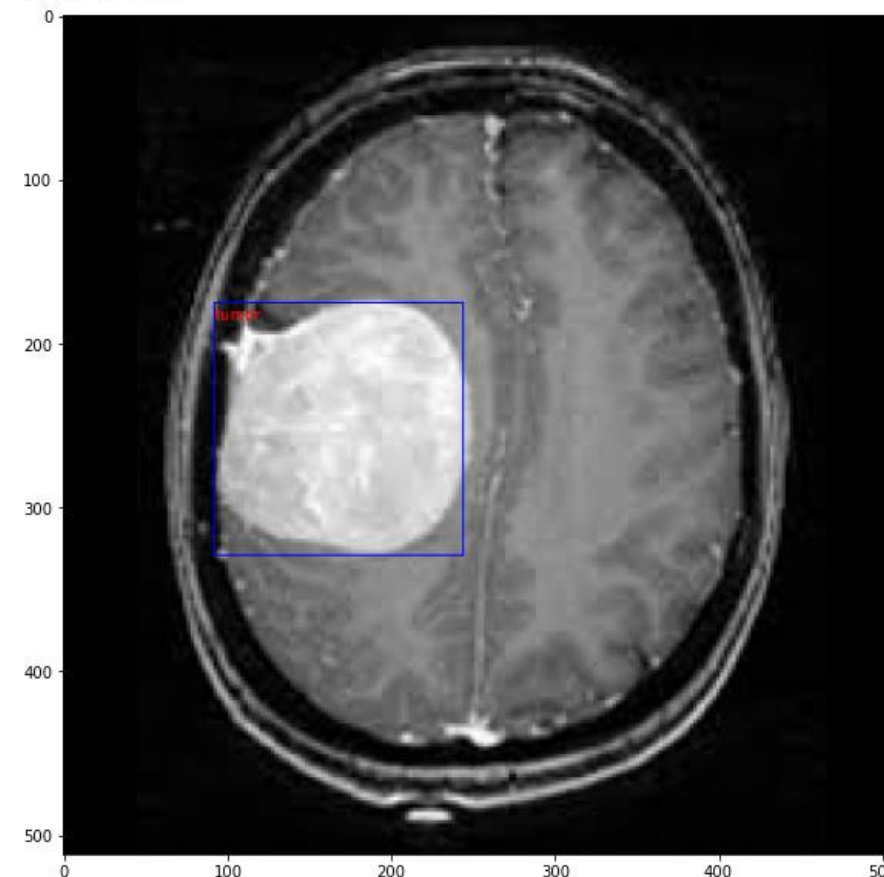


10_inference_folder_1_GPU.ipynb

Infer all images in the folder.

You can choose GPU or CPU mode according to your own hardware.

```
tumor-1.jpg  
tumor  
168 252  
92 175 152 154
```



```
Underkill Rate: 0.00%, Overkill Rate: 0.00%, Right Rate: 100.00%, Total: 1
```

11_inference_folder_demo_GPU.ipynb

Continue to infer all images in the folder

You can choose GPU or CPU mode according to your own hardware.

Press the "Esc" key on the display to turn off continuous inference.



12_YOLOv4_auto_labeling_GPU.ipynb

According to the trained model, the images in the test folder in the dataset are automatically labeled, and the labeled images and labeled voc files are stored in the auto_labeling folder of the dataset.

Reference

- Please refer to the readme.txt in the SDK folder.
- LEADERG AppForAI: <https://www.leaderg.com/appforai-windows>
- Copyright © LEADERG INC. All rights reserved.