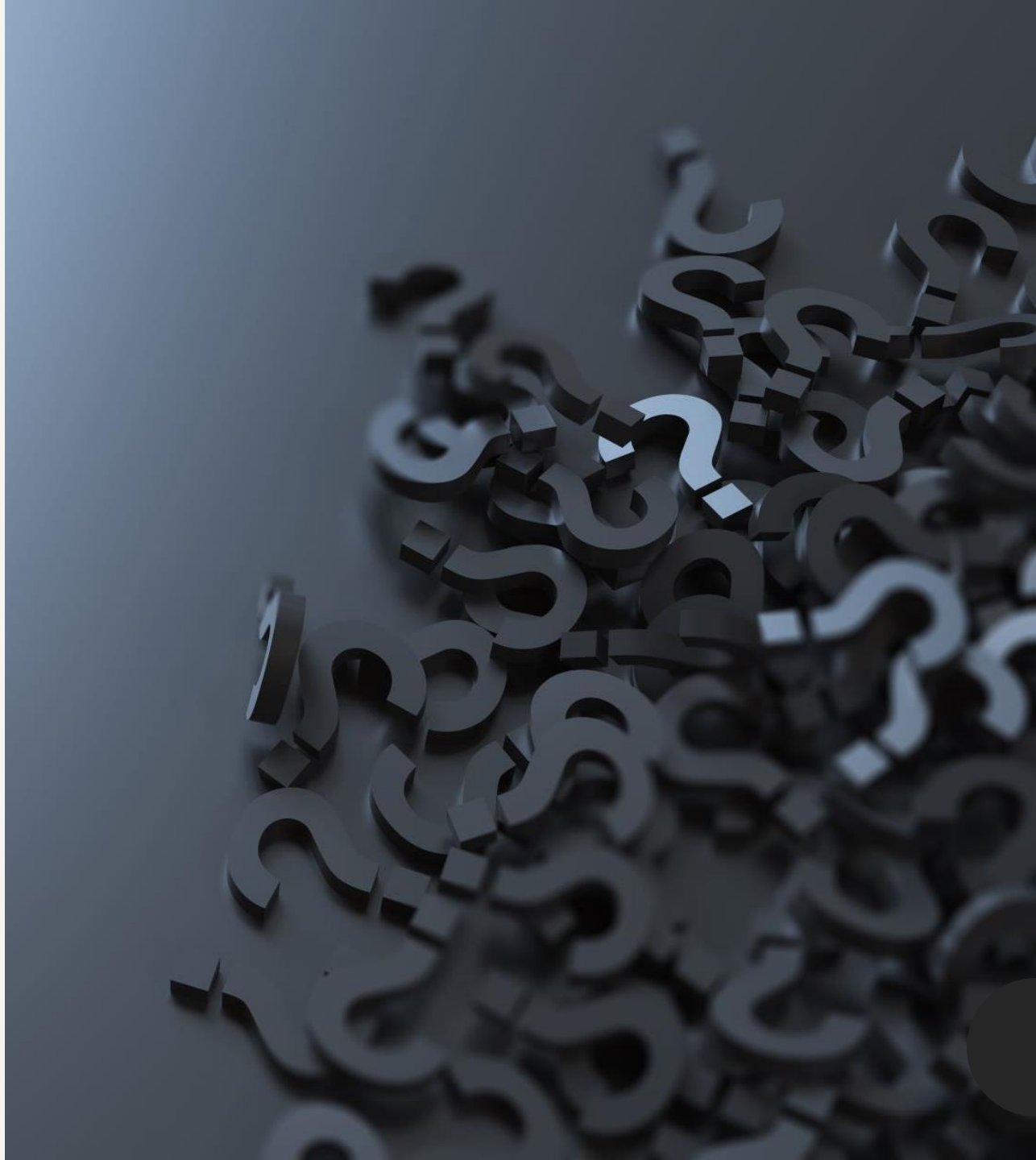
Farklı Model Askeri Araçların Derin Öğrenme Modeli ile Tespiti ve Ayrıştırılması.

Hazırlayanlar
Alper Sargın/Mehmet Akif Altınsoy
201312030/201312050

Başlangıç Olarak

- Ekibimizle beraber ilk olarak Google,Yandex gibi çeşitli arama motorlarından indirmiş olduğumuz askeri araç fotoğraflarıyla verisetimizi oluşturduk.




askeriarac (147).jpg

askeriarac (148).jpg

askeriarac (149).jpg

askeriarac (150).jpg

askeriarac (151).jpg

askeriarac (152).jpg

askeriarac (153).jpg

askeriarac (154).jpg

askeriarac (155).jpg

askeriarac (156).jpg

askeriarac (159).jpg

askeriarac (160).jpg

askeriarac (161).jpg

askeriarac (162).jpg

askeriarac (163).jpg

askeriarac (164).jpg

askeriarac (165).jpg

askeriarac (166).jpg

askeriarac (167).jpg

askeriarac (168).jpg

askeriarac (171).jpg

askeriarac (172).jpg

askeriarac (173).jpg

askeriarac (174).jpg

askeriarac (175).jpg

askeriarac (176).jpg

askeriarac (177).jpg

askeriarac (178).jpg

askeriarac (179).jpg

askeriarac (180).jpg

askeriarac (183).jpg

askeriarac (184).jpg

askeriarac (185).jpg

askeriarac (186).jpg

askeriarac (187).jpg

askeriarac (188).jpg

askeriarac (189).jpg

askeriarac (190).jpg

askeriarac (191).jpg

askeriarac (192).jpg

askeriarac (195).jpg

askeriarac (196).jpg

askeriarac (197).jpg

askeriarac (198).jpg

askeriarac (199).jpg

askeriarac (200).jpg

askeriarac (201).jpg

askeriarac (202).jpg

askeriarac (203).jpg

askeriarac (204).jpg

askeriarac (207).jpg

askeriarac (208).jpg

askeriarac (209).jpg

askeriarac (210).jpg

askeriarac (211).jpg

askeriarac (212).jpg

askeriarac (213).jpg

askeriarac (214).jpg

askeriarac (215).jpg

askeriarac (216).jpg

askeriarac (219).jpg

askeriarac (220).jpg

askeriarac (221).jpg

askeriarac (222).jpg

askeriarac (223).jpg

askeriarac (224).jpg

askeriarac (225).jpg

askeriarac (226).jpg

askeriarac (227).jpg

askeriarac (228).jpg

Etiket

- Fotoğrafları MAKESENSE AI (https://www.makesense.ai) sitesinden her fotoğrafa ayrı ayrı etiket attık.


Images
Rect
ALMAN TANKI
Labels
x: 549, y: 324
tank (132).jpg
Point
Line
Polygon

Eğitim

Eğitim

```
!python train.py --img 640 --batch 16 --epochs 50 --data coco.yaml --weights yolov5s.pt --cache --optimizer Adam --hyp data/hyps/hyp.scratch-med.yaml
```

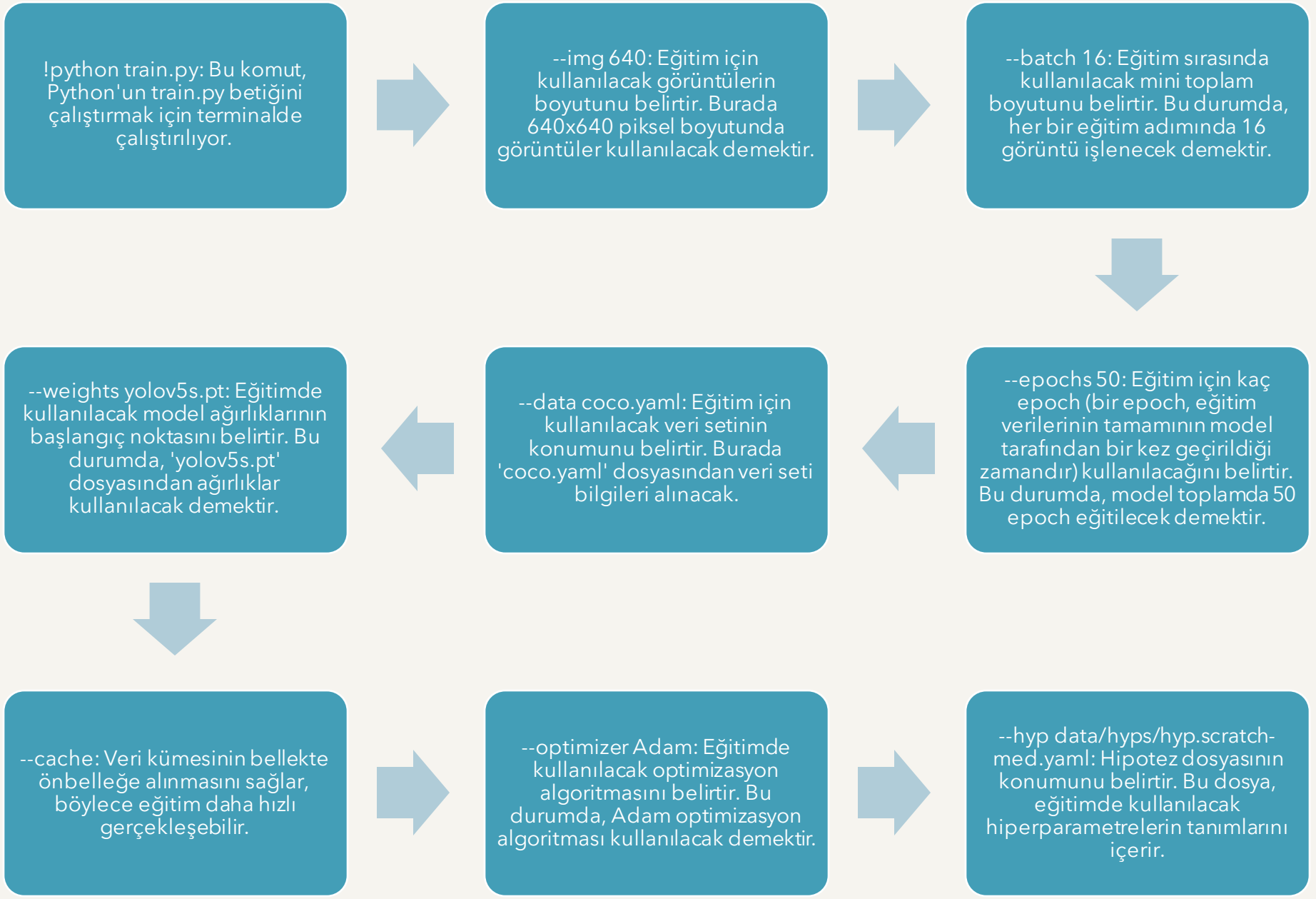
Python

```
2024-05-04 16:56:36.549220: E external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:9261] Unable to register cuDNN factory: Attempting to register factory for p
2024-05-04 16:56:36.549276: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:607] Unable to register cuFFT factory: Attempting to register factory for pl
2024-05-04 16:56:36.550852: E external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1515] Unable to register cuBLAS factory: Attempting to register factory for
train: weights=yolov5s.pt, cfg=, data=coco.yaml, hyp=data/hyps/hyp.scratch-med.yaml, epochs=50, batch_size=16, imgsz=640, rect=False, resume=False, nosave=False, n
github: up to date with https://github.com/ultralytics/yolov5 ✅
YOLOv5 🚀 v7.0-306-gb599ae42 Python-3.10.12 torch-2.2.1+cu121 CUDA:0 (Tesla T4, 15102MiB)

hyperparameters: lr0=0.01, lrf=0.1, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=0.05, cls=0.3, cls_pw=1.0
Comet: run 'pip install comet_ml' to automatically track and visualize YOLOv5 🚀 runs in Comet
TensorBoard: Start with 'tensorboard --logdir runs/train', view at http://localhost:6006/

Dataset not found ⚠️, missing paths ['/content/yolov5/yolov5/datasets/coco/val2017.txt']
Downloading https://github.com/ultralytics/yolov5/releases/download/v1.0/coco2017labels.zip to /content/yolov5/yolov5/datasets/coco2017labels.zip...
100% 46.4M/46.4M [00:00<00:00, 260MB/s]
Unzipping /content/yolov5/yolov5/datasets/coco2017labels.zip...
Downloading http://images.cocodataset.org/zips/train2017.zip to /content/yolov5/yolov5/datasets/coco/images/train2017.zip...
Downloading http://images.cocodataset.org/zips/val2017.zip to /content/yolov5/yolov5/datasets/coco/images/val2017.zip...
Downloading http://images.cocodataset.org/zips/test2017.zip to /content/yolov5/yolov5/datasets/coco/images/test2017.zip...
```

- Daha sonra etiketlediğimiz bu fotoğrafları YOLOV5 formatında indirip Google Colab ortamında yanda gösterilen kodlar ile eğittik.


!python train.py: Bu komut, Python'un train.py betiğini çalıştırmak için terminalde çalıştırılıyor.
--img 640: Eğitim için kullanılacak görüntülerin boyutunu belirtir. Burada 640x640 piksel boyutunda görüntüler kullanılacak demektir.
--batch 16: Eğitim sırasında kullanılacak mini toplam boyutunu belirtir. Bu durumda, her bir eğitim adımında 16 görüntü işlenecek demektir.
--epochs 50: Eğitim için kaç epoch (bir epoch, eğitim verilerinin tamamının model tarafından bir kez geçirildiği zamandır) kullanılacağını belirtir. Bu durumda, model toplamda 50 epoch eğitilecek demektir.
--data coco.yaml: Eğitim için kullanılacak veri setinin konumunu belirtir. Burada 'coco.yaml' dosyasından veri seti bilgileri alınacak.
--weights yolov5s.pt: Eğitimde kullanılacak model ağırlıklarının başlangıç noktasını belirtir. Bu durumda, 'yolov5s.pt' dosyasından ağırlıklar kullanılacak demektir.
--cache: Veri kümesinin bellekte önbelleğe alınmasını sağlar, böylece eğitim daha hızlı gerçekleşebilir.
--optimizer Adam: Eğitimde kullanılacak optimizasyon algoritmasını belirtir. Bu durumda, Adam optimizasyon algoritması kullanılacak demektir.
--hyp data/hyps/hyp.scratch-med.yaml: Hipotez dosyasının konumunu belirtir. Bu dosya, eğitimde kullanılacak hiperparametrelerin tanımlarını içerir.

- Eğitimin sonunda best.pt ağırlık dosyamızı aldık ve ilk testi gerçekleştirdik.

main.py > SecimEkrani

```python
class SecimEkrani:
    def __init__(self, master):
        self.master.title("Derin Öğrenme Projesi Seçim Ekranı")
        self.master.geometry("450x300")
        self.master.configure(background="white")
        tk.Label(self.master, text="Mehmet Akif Altınsoy", bg="white", fg="black", font=("Arial", 14)).pack()
        tk.Label(self.master, text="Alper SARGIN", bg="white", fg="black", font=("Arial", 14)).pack()
        tk.Label(self.master, text="Lütfen Seçim yapınız.", bg="white", fg="black", font=("Arial", 14, "bold")).pack(pady=10)

        tk.Button(self.master, text="Fotoğraf Tespiti", bg="blue", fg="white", height=2, width=50, font=("Arial", 12), command=self.handle_veriseti).pack(pady=5)
        tk.Button(self.master, text="Eğitim Grafiği", bg="blue", fg="white", height=2, width=50, font=("Arial", 12), command=self.handle_grafik).pack(pady=5)

    def handle_veriseti(self):
        self.master.destroy()
        Veriseti(self.restart)

    def handle_grafik(self):
        self.master.destroy()
        Grafik(self.restart)

    def restart(self):
        self.master = tk.Tk()
        self.__init__(self.master)
        self.master.mainloop()
```

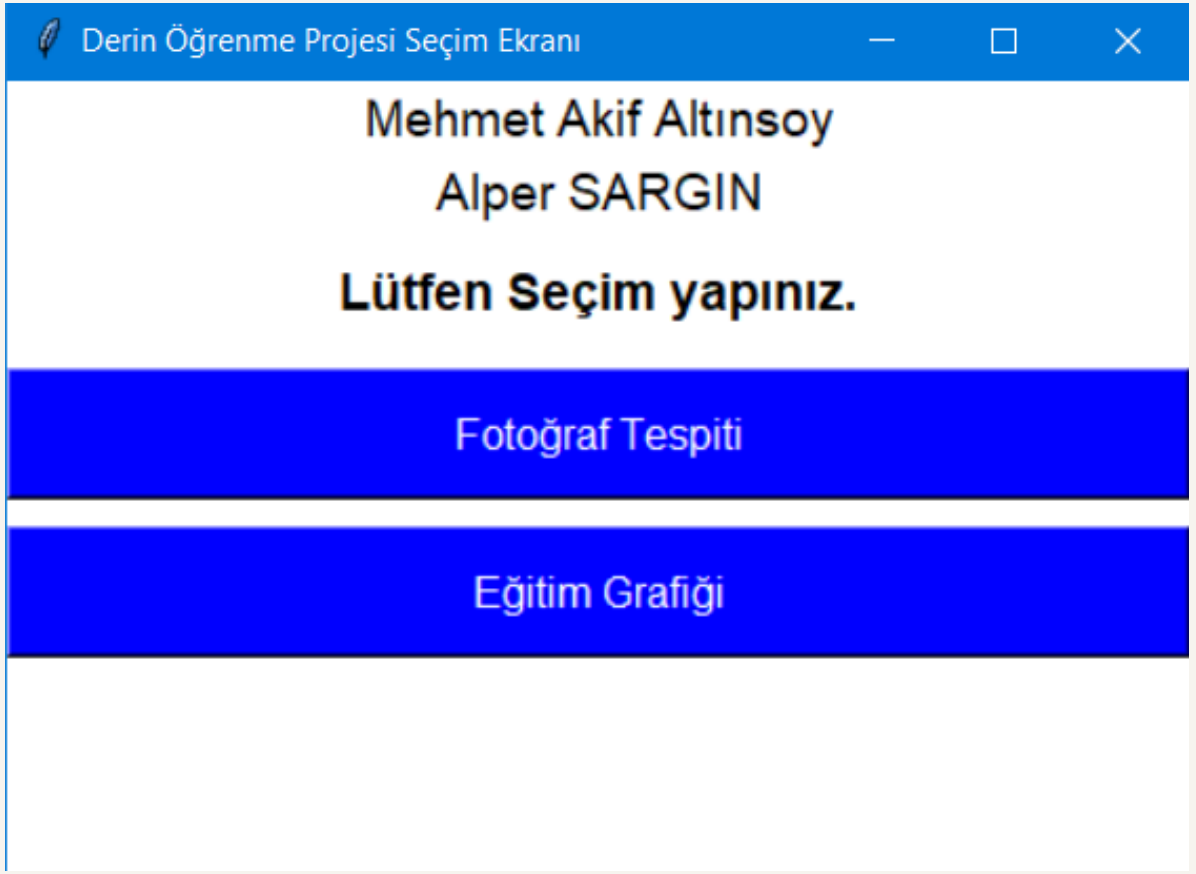
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

powershell - yolov5

```
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod> cd .\yolov5\
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5> python detect.py
detect: weights=yolov5s.pt, source=data\images, data=data\coco128.yaml, imgsz=[640, 640], conf_thres=0.25, iou_thres=0.45, max_det=1000, device=, view_img=False, save_txt=False,
 save_csv=False, save_conf=False, save_crop=False, nosave=False, classes=None, agnostic_nms=False, augment=False, visualize=False, update=False, project=runs\detect, name=exp, e
xist_ok=False, line_thickness=3, hide_labels=False, hide_conf=False, half=False, dnn=False, vid_stride=1
YOLOv5  v7.0-313-g712de55a Python-3.10.11 torch-2.3.0+cpu CPU

Fusing layers...
Model summary: 157 layers, 7018216 parameters, 0 gradients, 15.8 GFLOPs
image 1/1 C:\Users\SARGIN\Desktop\derinogr_kod\Test\1.jpg: 384x640 1 SAVAS UCAG, 127.5ms
Speed: 1.0ms pre-process, 127.5ms inference, 1.0ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs\detect\exp
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5>
```

Arayüz(TKinter)



veriseti.py > ImageViewer > __init__

```
6   class ImageViewer(tk.Tk):
46
47      def show_next_image(self):
48          if self.image_files:
49              self.current_index = (self.current_index + 1) % len(self.image_files)
50              self.show_image()
51
52      def on_closing(self):
53          self.destroy()
54          self.callback()
55
56  def Veriseti(callback):
57      folder_path = "C:/Users/SARGIN/Desktop/derinogr_kod/yolov5/runs/detect/exp"
58      app = ImageViewer(folder_path, callback)
59      app.mainloop()
60
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

powershell - yolov5

```
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod> cd .\yolov5\
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5> python detect.py
detect: weights=yolov5s.pt, source=data\images, data=data\coco128.yaml, imgsz=[640, 640], conf_thres=0.25, iou_thres=0.45, max_det=1000, device=, view_img=False, save_txt=False,
 save_csv=False, save_conf=False, save_crop=False, nosave=False, classes=None, agnostic_nms=False, augment=False, visualize=False, update=False, project=runs\detect, name=exp, e
xist_ok=False, line_thickness=3, hide_labels=False, hide_conf=False, half=False, dnn=False, vid_stride=1
YOLOv5  v7.0-313-g712de55a Python-3.10.11 torch-2.3.0+cpu CPU

Fusing layers...
Model summary: 157 layers, 7018216 parameters, 0 gradients, 15.8 GFLOPs
image 1/1 C:\Users\SARGIN\Desktop\derinogr_kod\Test\1.jpg: 384x640 1 SAVAS UCAG, 127.5ms
Speed: 1.0ms pre-process, 127.5ms inference, 1.0ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs\detect\exp
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5>
```

Fotoğraf Tespit

Grafik.py > Grafik > __init__

```
6   class Grafik:
30      class ResimGosterici:
31          def __init__(self, root, resim_listesi):
36              self.resim_etiketi = ttk.Label(self.root)
37              self.resim_etiketi.pack()
38
39              self.geri_buton = ttk.Button(self.root, text="<< Geri", command=self.onceki_resim)
40              self.geri_buton.pack(side=tk.LEFT)
41
42              self.ileri_buton = ttk.Button(self.root, text="İleri >>", command=self.sonraki_resim)
43              self.ileri_buton.pack(side=tk.RIGHT)
44
45              self.resim_referanslari = []
46              self.resmi_goster(self.gecerli_indeks)
47
48          def resmi_goster(self, indeks):
49              resim_yolu = self.resim_listesi[indeks]
50              if os.path.exists(resim_yolu):
51                  image = Image.open(resim_yolu)
52                  image = image.resize((600, 600), Image.LANCZOS)
53                  self.photo = ImageTk.PhotoImage(image)
54                  self.resim_etiketi.config(image=self.photo)
55                  self.resim_referanslari.append(self.photo)  # Referansı sakla
56              else:
57                  self.resim_etiketi.config(text=f"Resim bulunamadı: {resim_yolu}")
```

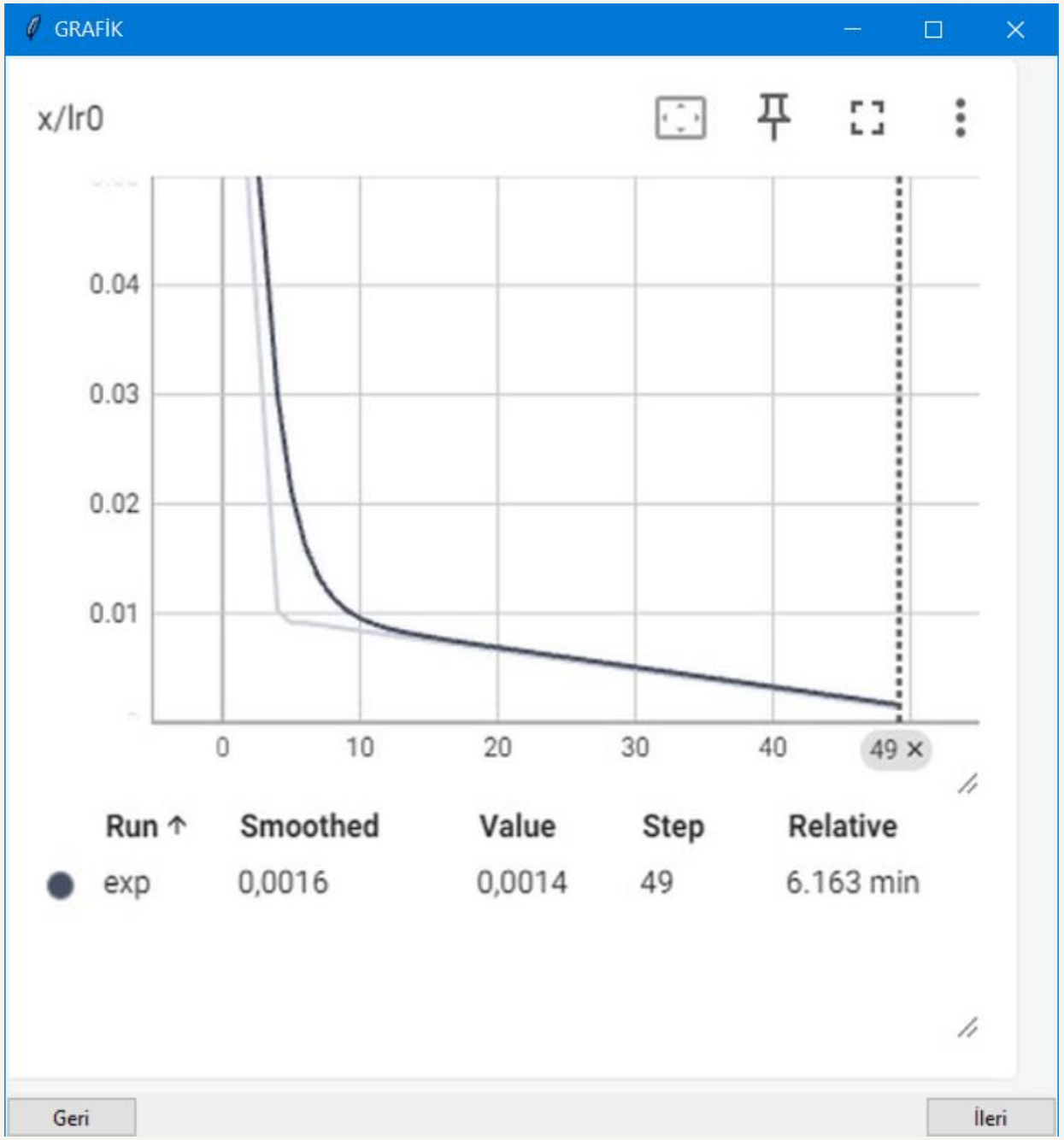
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

powershell - yolov5

```
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod> cd .\yolov5\
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5> python detect.py
detect: weights=yolov5s.pt, source=data\images, data=data\coco128.yaml, imgsz=[640, 640], conf_thres=0.25, iou_thres=0.45, max_det=1000, device=, view_img=False, save_txt=False,
 save_csv=False, save_conf=False, save_crop=False, nosave=False, classes=None, agnostic_nms=False, augment=False, visualize=False, update=False, project=runs\detect, name=exp, e
xist_ok=False, line_thickness=3, hide_labels=False, hide_conf=False, half=False, dnn=False, vid_stride=1
YOLOv5  v7.0-313-g712de55a Python-3.10.11 torch-2.3.0+cpu CPU

Fusing layers...
Model summary: 157 layers, 7018216 parameters, 0 gradients, 15.8 GFLOPs
image 1/1 C:\Users\SARGIN\Desktop\derinogr_kod\Test\1.jpg: 384x640 1 SAVAS UCAG, 127.5ms
Speed: 1.0ms pre-process, 127.5ms inference, 1.0ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs\detect\exp
(venv) PS C:\Users\SARGIN\Desktop\derinogr_kod\yolov5>
```

Learning Rate



DEMO