TCC BI-MASTER PUC-Rio

Reconhecimento de Aeronaves a partir de Imagens de Sensoriameno Remoto usando Deep Learning

Aluno: David Fernando Castillo Zúñiga

Orientador: Leonardo Forero Mendoza

```
# Inspirado em dataset e notebook base Airbus Aircraft Detection: https://www.kaggle.com/datasets/airbusgeo/airbus-aircrafts-sample-datas
```

Importação de Datatset e de Bibliotecas Básicas

Clique duas vezes (ou pressione "Enter") para editar

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

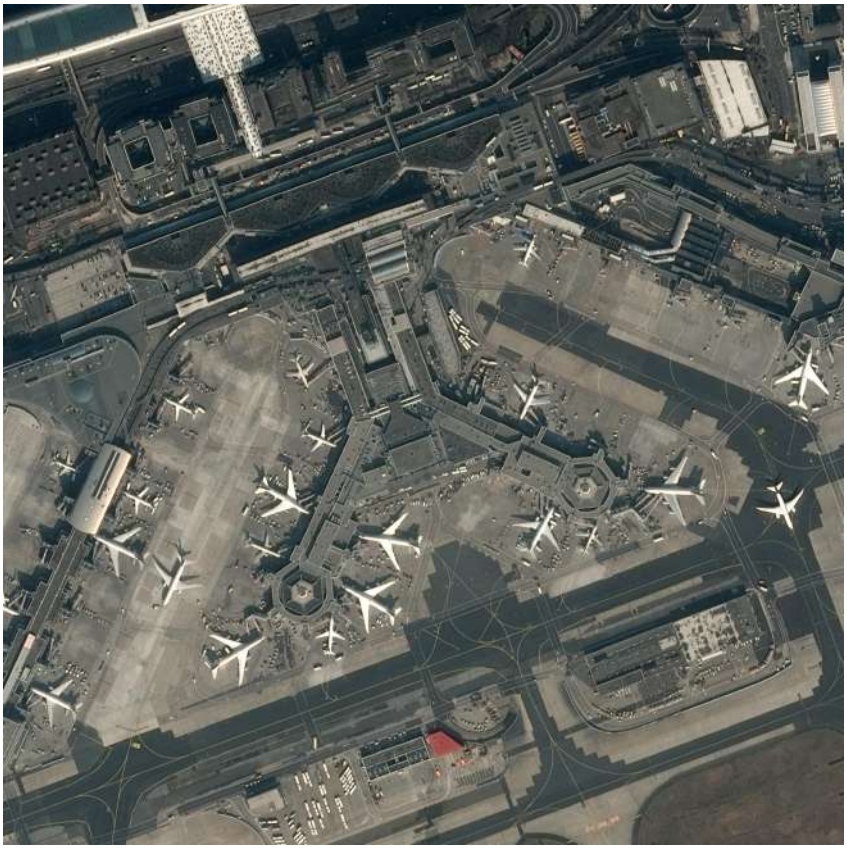
```
import os
import numpy as np
import pandas as pd
import ast
import torch
import PIL
from tqdm.auto import tqdm
import shutil as sh
from pathlib import Path
import random

from IPython.display import Image, clear_output
import matplotlib.pyplot as plt
%matplotlib inline
```

```
%%time
!git clone https://github.com/ultralytics/yolov5  # Clonar repositorio
!pip install -qr yolov5/requirements.txt  # Instale os pacotes necessários do diretório raiz do repositório clonado
!cp yolov5/requirements.txt ./
```

```
Cloning into 'yolov5'...
remote: Enumerating objects: 14918, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 14918 (delta 2), reused 6 (delta 1), pack-reused 14908
Receiving objects: 100% (14918/14918), 13.97 MiB | 8.98 MiB/s, done.
Resolving deltas: 100% (10241/10241), done.
     |████████████████████████████████| 184 kB 38.6 MB/s
     |████████████████████████████████| 62 kB 1.1 MB/s
     |████████████████████████████████| 1.6 MB 70.1 MB/s
CPU times: user 101 ms, sys: 42.9 ms, total: 144 ms
Wall time: 11.4 s
```

Análise Exploratoria Dataset

```
DATA_DIR = Path('/content/drive/My Drive/archive')
img_list = list(DATA_DIR.glob('images/*.jpg'))
pickone = random.choice(img_list)
img = PIL.Image.open(pickone)
display(img)
```



```
only_files = [DATA_DIR / f.name for f in img_list if os.path.isfile(f) and f.name[-4:] == ".jpg"]
print("Found {} images files in {}".format(len(only_files), DATA_DIR))

IMAGE_HEIGHT, IMAGE_WIDTH = img.size
num_channels = len(img.mode)
print("Image size: {}".format((IMAGE_HEIGHT, IMAGE_WIDTH)))
print("Num channels: {}".format(num_channels))
```

```
Found 103 images files in /content/drive/My Drive/archive
Image size: (2560, 2560)
Num channels: 3
```

Desejam-se adicionar as informações da caixa delimitadora ao dataframe. Uma caixa delimitadora é um retângulo ao redor do objeto detectado. O arquivo de anotação fornece as coordenadas de 2 pontos para descrever uma caixa delimitadora (cantos superior esquerdo e inferior direito). Para analisar o conjunto de dados, comutamos a informação anterior por um conjunto de 4 coordenadas límite e a largura e a altura da caixa delimitadora.

```
df = pd.read_csv(DATA_DIR / 'annotations.csv')
# converte um registro de string em um objeto python válido
def f(x):
    return ast.literal_eval(x.rstrip('\r\n'))

df = pd.read_csv(DATA_DIR / "annotations.csv",
                 converters={'geometry': f})
df.head(10)
```

	id	image_id	geometry	class
0	1	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(135, 522), (245, 522), (245, 600), (135, 600...	Airplane
1	2	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(1025, 284), (1125, 284), (1125, 384), (1025,...	Airplane
2	3	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(1058, 1503), (1130, 1503), (1130, 1568), (10...	Airplane
3	4	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(813, 1518), (885, 1518), (885, 1604), (813, ...	Airplane
4	5	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(594, 938), (657, 938), (657, 1012), (594, 10...	Airplane
5	6	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(451, 725), (524, 725), (524, 798), (451, 798...	Airplane
6	7	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(1543, 1437), (1614, 1437), (1614, 1497), (15...	Airplane

```
unique, counts = np.unique(df['class'], return_counts=True)
pd.DataFrame({'class': unique, 'count': counts})
```

	class	count
0	Airplane	3316
1	Truncated_airplane	109

```
unique, counts = np.unique(df['image_id'], return_counts=True)
#per_image = np.asarray((unique, counts)).T
print("Minimum aircrafts per image: {}".format(np.min(counts)))
print("Maximum aircrafts per image: {}".format(np.max(counts)))
```

```
Minimum aircrafts per image: 5
Maximum aircrafts per image: 92
```

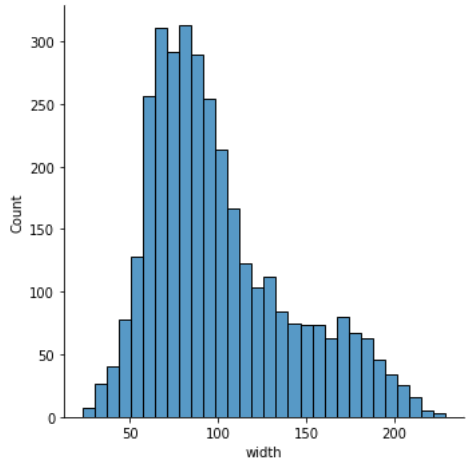
```
def getBounds(geometry):
    try:
        arr = np.array(geometry).T
        xmin = np.min(arr[0])
        ymin = np.min(arr[1])
        xmax = np.max(arr[0])
        ymax = np.max(arr[1])
        return (xmin, ymin, xmax, ymax)
    except:
        return np.nan

def getWidth(bounds):
    try:
        (xmin, ymin, xmax, ymax) = bounds
        return np.abs(xmax - xmin)
    except:
        return np.nan

def getHeight(bounds):
    try:
        (xmin, ymin, xmax, ymax) = bounds
        return np.abs(ymax - ymin)
    except:
        return np.nan

# Create bounds, width and height
df.loc[:,'bounds'] = df.loc[:,'geometry'].apply(getBounds)
df.loc[:,'width'] = df.loc[:,'bounds'].apply(getWidth)
```

```
df.loc[:,'height'] = df.loc[:,'bounds'].apply(getHeight)
df.head(10)
```

	id	image_id	geometry	class	bounds	width	height
0	1	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(135, 522), (245, 522), (245, 600), (135, 600...	Airplane	(135, 522, 245, 600)	110	78
1	2	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(1025, 284), (1125, 284), (1125, 384), (1025,...	Airplane	(1025, 284, 1125, 384)	100	100
2	3	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(1058, 1503), (1130, 1503), (1130, 1568), (10...	Airplane	(1058, 1503, 1130, 1568)	72	65
3	4	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(813, 1518), (885, 1518), (885, 1604), (813, ...	Airplane	(813, 1518, 885, 1604)	72	86
4	5	4f833867-273e-4d73-8bc3-cb2d9ceb54ef.jpg	[(594, 938), (657, 938), (657, 1012), (594, 10...	Airplane	(594, 938, 657, 1012)	63	74

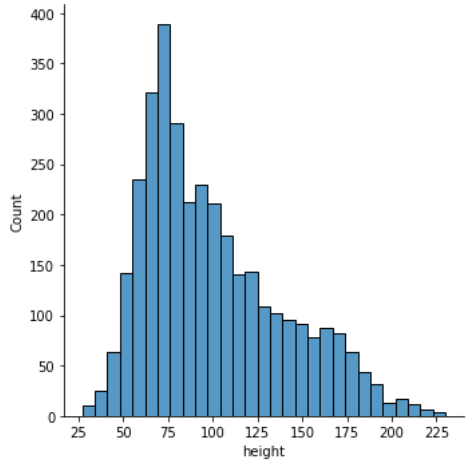
```
import seaborn as sns
sns.displot(df['width'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7ff161693be0>
```




```
sns.displot(df['height'])
```

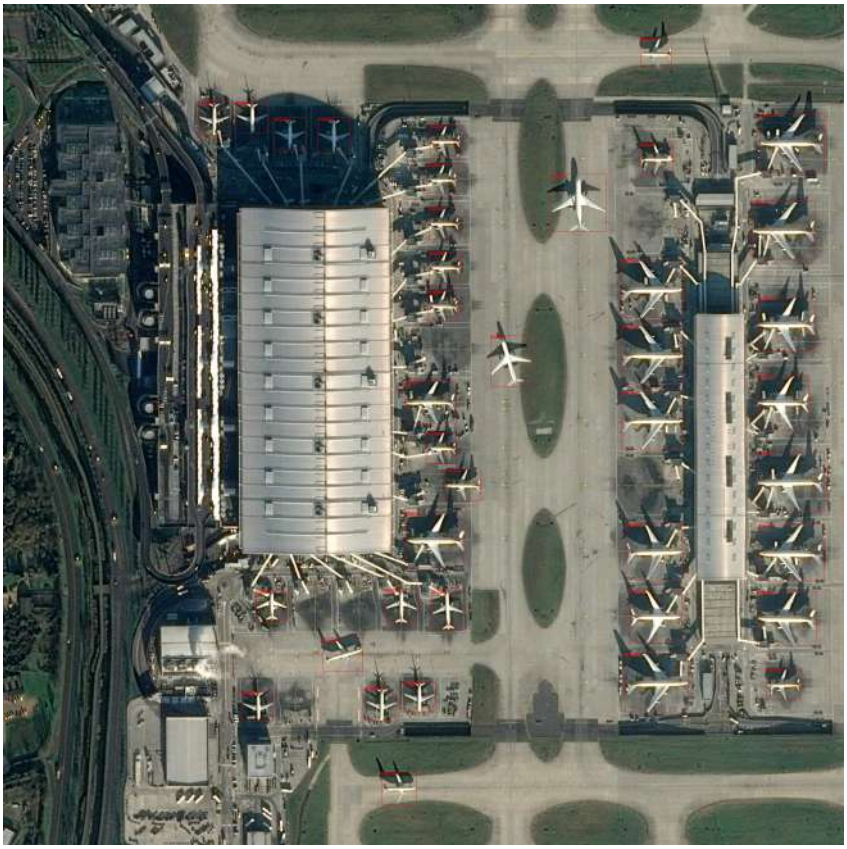
```
<seaborn.axisgrid.FacetGrid at 0x7ff15745e610>
```



Segue imagem com informação de caixa delimitadora:

```
import os.path
import PIL.ImageDraw

pickone = random.choice(img_list)
img = PIL.Image.open(pickone)
draw = PIL.ImageDraw.Draw(img)

for k, row in df[df['image_id'] == os.path.basename(pickone)].iterrows():
    draw.polygon(row['geometry'], outline=(255,0,0))
```

```
    draw.text(row['geometry'][0], row['class'], fill=(255,0,0))

display(img)
```



Criação de conjunto de images para validação

O código YOLOv5 espera uma configuração .yaml, onde o caminho para os dados e a partição de dados train/val/test são definidos.

```
# criar a lista de imagens usada para validação
fold = 1
num_fold = 5
index = df['image_id'].unique()
val_indexes = index[len(index)*fold//num_fold:len(index)*(fold+1)//num_fold]
print(val_indexes)
```

```
['78099b50-f2b6-4319-b462-f33df2966c45.jpg'
 '22291e0b-ebe2-4f3f-b53e-4e709179300a.jpg'
 'cc4f3226-c262-409e-a4b2-a576e776f7f4.jpg'
 '34ae857d-6e71-46b9-b694-d9e40fb093bc.jpg'
 '5c9e817a-dc4b-42ab-952c-3128e2de12e8.jpg'
 'af67041b-f363-47ae-8ddd-f652db3a6bab.jpg'
 '0263270b-e3ee-41dc-aeef-43ff77e66d5b.jpg'
 'd8873734-016a-4b9d-9b9e-8bc47eb13fef.jpg'
 'd0c3d270-f23e-4792-bac0-142a9cc8ccc6.jpg'
 '78400c58-1a7c-4342-a1fb-2117cb7cbc8b.jpg'
 '77f7b57f-5cf2-424d-a952-9847b3c3f35e.jpg'
 'd9399a45-6745-4e59-8903-90640b2ddf9f.jpg'
 '014de911-7810-4f7d-8967-3e5402209f4a.jpg'
 'd3d2b706-9017-41f4-b57e-469038daa634.jpg'
 '4c9d2482-788c-4d68-a3d4-478b2367abce.jpg'
 '576827bc-a94a-4611-8820-f3d56e969151.jpg'
 '4e9164aa-532e-4b76-bce4-060b090da357.jpg'
 '14436c8c-93ec-41af-9fbf-43a5f39f2b98.jpg'
 '7635d63c-6b97-4c9c-a7dc-27773d42ed4c.jpg'
 'ecfe7982-05e5-435f-824b-e24b6846316e.jpg'
 '8df07836-4606-446e-9880-6ed9e0f74543.jpg']
```

```
import os
import tqdm.notebook

# Create 512x512 tiles with 64 pix overlap in //working
TILE_WIDTH = 512
TILE_HEIGHT = 512
TILE_OVERLAP = 64
TRUNCATED_PERCENT = 0.3
_overwriteFiles = True

TILES_DIR = {'train': Path('working/train/images/'),
             'val': Path('working/val/images/')}
for _, folder in TILES_DIR.items():
    if not os.path.isdir(folder):
        os.makedirs(folder)

LABELS_DIR = {'train': Path('working/train/labels/'),
              'val': Path('working/val/labels/')}
for _, folder in LABELS_DIR.items():
    if not os.path.isdir(folder):
        os.makedirs(folder)

# Save one line in .txt file for each tag found inside the tile
def tag_is_inside_tile(bounds, x_start, y_start, width, height, truncated_percent):
    x_min, y_min, x_max, y_max = bounds
    x_min, y_min, x_max, y_max = x_min - x_start, y_min - y_start, x_max - x_start, y_max - y_start

    if (x_min > width) or (x_max < 0.0) or (y_min > height) or (y_max < 0.0):
        return None

    x_max_trunc = min(x_max, width)
    x_min_trunc = max(x_min, 0)
    if (x_max_trunc - x_min_trunc) / (x_max - x_min) < truncated_percent:
        return None

    y_max_trunc = min(y_max, width)
    y_min_trunc = max(y_min, 0)
    if (y_max_trunc - y_min_trunc) / (y_max - y_min) < truncated_percent:
        return None

    x_center = (x_min_trunc + x_max_trunc) / 2.0 / width
    y_center = (y_min_trunc + y_max_trunc) / 2.0 / height
    x_extend = (x_max_trunc - x_min_trunc) / width
    y_extend = (y_max_trunc - y_min_trunc) / height

    return (0, x_center, y_center, x_extend, y_extend)

for img_path in tqdm.notebook.tqdm(img_list):
```

```
    # Open image and related data
    pil_img = PIL.Image.open(img_path, mode='r')
    np_img = np.array(pil_img, dtype=np.uint8)

    # Get annotations for image
    img_labels = df[df["image_id"] == img_path.name]
    #print(img_labels)

    # Count number of sections to make
    X_TILES = (IMAGE_WIDTH + TILE_WIDTH + TILE_OVERLAP - 1) // TILE_WIDTH
    Y_TILES = (IMAGE_HEIGHT + TILE_HEIGHT + TILE_OVERLAP - 1) // TILE_HEIGHT

    # Cut each tile
    for x in range(X_TILES):
        for y in range(Y_TILES):

            x_end = min((x + 1) * TILE_WIDTH - TILE_OVERLAP * (x != 0), IMAGE_WIDTH)
            x_start = x_end - TILE_WIDTH
            y_end = min((y + 1) * TILE_HEIGHT - TILE_OVERLAP * (y != 0), IMAGE_HEIGHT)
            y_start = y_end - TILE_HEIGHT
            #print(x_start, y_start)

            folder = 'val' if img_path.name in val_indexes else 'train'
            save_tile_path = TILES_DIR[folder].joinpath(img_path.stem + "_" + str(x_start) + "_" + str(y_start) + ".jpg")
            save_label_path = LABELS_DIR[folder].joinpath(img_path.stem + "_" + str(x_start) + "_" + str(y_start) + ".txt")

            # Save if file doesn't exit
            if _overwriteFiles or not os.path.isfile(save_tile_path):
                cut_tile = np.zeros(shape=(TILE_WIDTH, TILE_HEIGHT, 3), dtype=np.uint8)
                cut_tile[0:TILE_HEIGHT, 0:TILE_WIDTH, :] = np_img[y_start:y_end, x_start:x_end, :]
                cut_tile_img = PIL.Image.fromarray(cut_tile)
                cut_tile_img.save(save_tile_path)

            found_tags = [
                tag_is_inside_tile(bounds, x_start, y_start, TILE_WIDTH, TILE_HEIGHT, TRUNCATED_PERCENT)
                for i, bounds in enumerate(img_labels['bounds'])]
            found_tags = [el for el in found_tags if el is not None]

            # save labels
            with open(save_label_path, 'w+') as f:
                for tags in found_tags:
                    f.write(' '.join(str(x) for x in tags) + '\n')
```

```
100%                                                  103/103 [00:43<00:00, 2.54it/s]
```

```
# Tensorboard (optional)
#%load_ext tensorboard
#%tensorboard --logdir runs/train


CONFIG = """
# train and val datasets (image directory or *.txt file with image paths)
train: /content/working/train/
val: /content/working/val/

# number of classes
nc: 1

# class names
names: ['Aircraft']
"""

with open("dataset.yaml", "w") as f:
    f.write(CONFIG)


print('Setup complete. Using torch %s %s' % (torch.__version__, torch.cuda.get_device_properties(0) if torch.cuda.is_available() else 'CP
```

```
Setup complete. Using torch 1.13.0+cu116 _CudaDeviceProperties(name='Tesla T4', major=7, minor=5, total_memory=15109MB, multi_proce
```

Criação de ambiente de métricas

```
!rm -rf working/runs
!rm -rf working/wandb
```

```
!pip install wandb
import wandb
wandb.login()
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheel
Requirement already satisfied: wandb in /usr/local/lib/python3.8/dist-packages (0.
Requirement already satisfied: requests<3,>=2.0.0 in /usr/local/lib/python3.8/dist
Requirement already satisfied: shortuuid>=0.5.0 in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: docker-pycreds>=0.4.0 in /usr/local/lib/python3.8/d
Requirement already satisfied: sentry-sdk>=1.0.0 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: setproctitle in /usr/local/lib/python3.8/dist-packa
Requirement already satisfied: pathtools in /usr/local/lib/python3.8/dist-packages
Requirement already satisfied: PyYAML in /usr/local/lib/python3.8/dist-packages (f
Requirement already satisfied: promise<3,>=2.0 in /usr/local/lib/python3.8/dist-pa
Requirement already satisfied: Click!=8.0.0,>=7.0 in /usr/local/lib/python3.8/dist
Requirement already satisfied: protobuf!=4.21.0,<5,>=3.12.0 in /usr/local/lib/pyth
Requirement already satisfied: setuptools in /usr/local/lib/python3.8/dist-package
Requirement already satisfied: psutil>=5.0.0 in /usr/local/lib/python3.8/dist-pack
Requirement already satisfied: GitPython>=1.0.0 in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: six>=1.4.0 in /usr/local/lib/python3.8/dist-package
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.8/dist-pa
Requirement already satisfied: smmap<6,>=3.0.1 in /usr/local/lib/python3.8/dist-pa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packa
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/loc
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
True
```

Treinamento e Validação

```
!python yolov5/train.py --img 512 --batch 16 --epochs 20 --data dataset.yaml --weights yolov5s.pt
```

```
wandb: Currently logged in as: davidfer. Use `wandb login --relogin` to force relogin
train: weights=yolov5s.pt, cfg=, data=dataset.yaml, hyp=yolov5/data/hyps/hyp.scratch-low.yaml, epochs=20, batch_size=16, imgsz=51
github: up to date with https://github.com/ultralytics/yolov5 ✅
YOLOv5 🚀 v7.0-55-g632bf48 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

hyperparameters: lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=
ClearML: run 'pip install clearml' to automatically track, visualize and remotely train YOLOv5 🚀 in ClearML
Comet: run 'pip install comet_ml' to automatically track and visualize YOLOv5 🚀 runs in Comet
TensorBoard: Start with 'tensorboard --logdir yolov5/runs/train', view at http://localhost:6006/
wandb: Tracking run with wandb version 0.13.7
wandb: Run data is saved locally in /content/wandb/run-20230103_153342-imyi41sg
wandb: Run `wandb offline` to turn off syncing.
wandb: Syncing run generous-voice-27
wandb: ⭐ View project at https://wandb.ai/davidfer/train
wandb: 🚀 View run at https://wandb.ai/davidfer/train/runs/imyi41sg
Downloading https://ultralytics.com/assets/Arial.ttf to /root/.config/Ultralytics/Arial.ttf...
100% 755k/755k [00:01<00:00, 492kB/s]
Downloading https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt to yolov5s.pt...
100% 14.1M/14.1M [00:00<00:00, 338MB/s]

Overriding model.yaml nc=80 with nc=1

                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     90880  models.common.C3                        [256, 128, 1, False]
 18                -1  1    147712  models.common.Conv                      [128, 128, 3, 2]
 19          [-1, 14]  1         0  models.common.Concat                    [1]
 20                -1  1    296448  models.common.C3                        [256, 256, 1, False]
 21                -1  1    590336  models.common.Conv                      [256, 256, 3, 2]
 22          [-1, 10]  1         0  models.common.Concat                    [1]
 23                -1  1   1182720  models.common.C3                        [512, 512, 1, False]
 24      [17, 20, 23]  1     16182  models.yolo.Detect                      [1, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 1
Model summary: 214 layers, 7022326 parameters, 7022326 gradients, 15.9 GFLOPs

Transferred 343/349 items from yolov5s.pt
AMP: checks passed ✅
```

```
optimizer: SGD(lr=0.01) with parameter groups 57 weight(decay=0.0), 60 weight(decay=0.0005), 60 bias
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip_limit=
train: Scanning /content/working/train/labels... 2952 images, 1419 backgrounds, 0 corrupt: 100% 2952/2952 [00:01<00:00, 1706.73it
train: New cache created: /content/working/train/labels.cache
```

Deteção de Imagens

```
!python yolov5/detect.py --source /content/drive/MyDrive/archive/extras --img-size 2560 --weights /content/yolov5/runs/train/exp/weights/
```

```
detect: weights=['/content/yolov5/runs/train/exp/weights/best.pt'], source=/content/drive/MyDrive/archive/extras, data=yolov5/data/
YOLOv5 🚀 v7.0-55-g632bf48 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs
image 1/6 /content/drive/MyDrive/archive/extras/022f91f0-1434-401f-a11b-e315b7068100.jpg: 2560x2560 26 Aircrafts, 117.1ms
image 2/6 /content/drive/MyDrive/archive/extras/08a8132a-a6c7-4cab-adee-7e2976fd2822.jpg: 2560x2560 28 Aircrafts, 119.4ms
image 3/6 /content/drive/MyDrive/archive/extras/22bc9d20-02c4-4554-8fed-2c127d54b5ed.jpg: 2560x2560 31 Aircrafts, 114.7ms
image 4/6 /content/drive/MyDrive/archive/extras/55aa185a-01c8-4668-ae87-1f1d67d15a08.jpg: 2560x2560 28 Aircrafts, 101.3ms
image 5/6 /content/drive/MyDrive/archive/extras/65825eef-f8a1-41b3-ac87-4a0a7d482a0e.jpg: 2560x2560 20 Aircrafts, 93.3ms
image 6/6 /content/drive/MyDrive/archive/extras/defbf838-828b-4427-9bb7-9af33563ea9c.jpg: 2560x2560 67 Aircrafts, 120.7ms
Speed: 5.7ms pre-process, 111.1ms inference, 1.3ms NMS per image at shape (1, 3, 2560, 2560)
Results saved to yolov5/runs/detect/exp
```

```
DATA_DIR = Path('yolov5/runs/detect/exp/')
img_list = list(DATA_DIR.glob('*.jpg'))
random.seed(3)
pickone = random.choice(img_list)
img = PIL.Image.open(pickone)
display(img)
```

✓ 57s conclusão: 12:57