SAPIENZA
UNIVERSITÀ DI ROMA

F1 racing cars
detection and tracking in videos

Andrea Gasparini 1813486

Edoardo Di Paolo 1728334

A.Y. 2021-2022

- **LaSOT dataset**
- **Addressed Task**
- **Video detection**

LaSOT Dataset

- LaSOT: **La**rge-scale **S**ingle **O**bject **T**racking
  - largest densely annotated tracking benchmark
- **85** object classes
- More than **3,87** million frames
- Each frame manually annotated with a bounding box

LaSOT Dataset – Example

Each “video” comes with:

- one image per frame
- a ground truth file with the bounding box coordinates of each frame
- (108, 147): bounding box top-left pixel
- (323, 128): bounding box width and height

LaSOT Dataset – Racing subsets

- Selected "**racing**" category datasets:
  - racing-1: **1605** frames
  - racing-9: **2897** frames
  - racing-11: **1264** frames
  - racing-12: **1137** frames
- Totaling **6903** images with different dimensions

Addressed task

Image detection

- Transfer learning
- Bounding box prediction on images

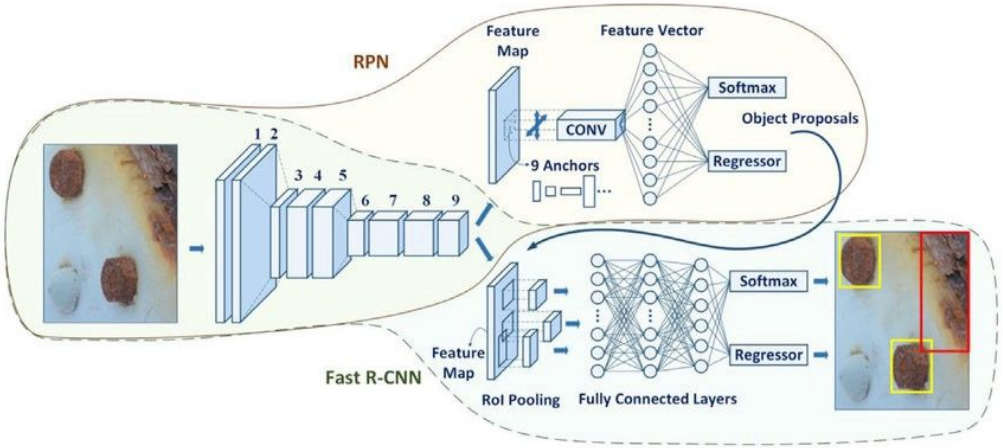
Video tracking

- Video frames splitting
- Predictions and bb drawing for tracking
- Re-create the video

Tracking enhancement through histograms

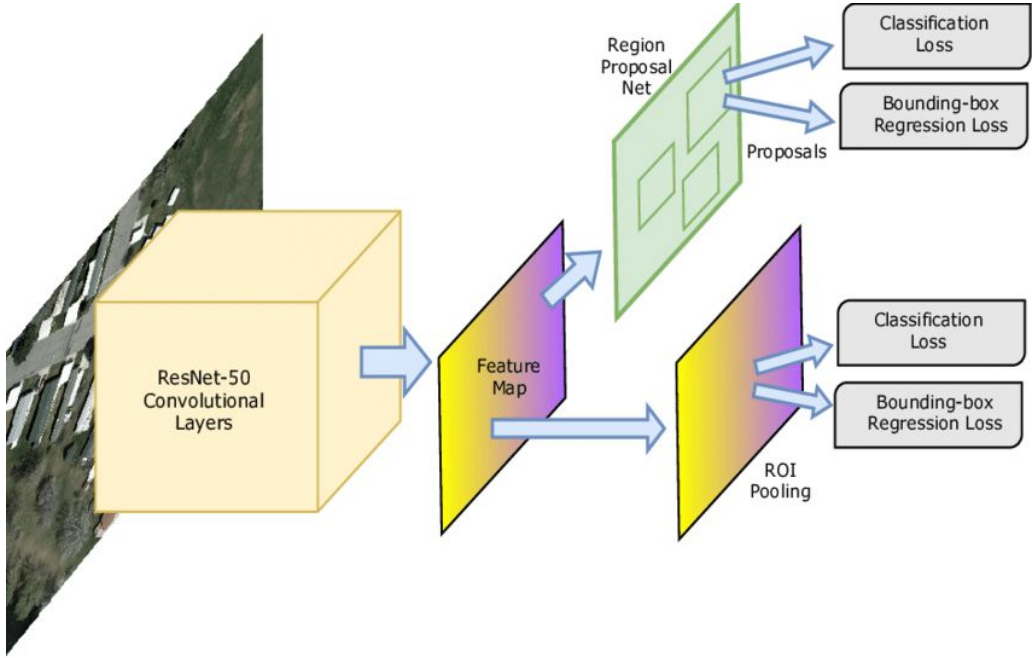
- Discrepancies adaption between consecutive frames

Task – Image detection



- Convolutional layers:
  - extract the appropriate features for the images
  - generally composed of convolution layers, pooling layers and a fully connected layer
- Region Proposal Network:
  - small neural network sliding on the last feature map of the convolution layers and predict whether there is an object or not and also predict the bounding box
- Classes and bounding boxes prediction

Task – Image detection



- Our model:
  - Faster R-CNN architecture
    - ResNet-50-FPN backbone
  - regression head
    - 2 Fully-connected layers
- expected input tensors:
  - boxes: the ground-truth boxes in [x1, y1, x2, y2] format, with $0 \leq x1 < x2 \leq W$ and $0 \leq y1 < y2 \leq H$
  - labels: the class label for each ground-truth box, in our case always 1
- model output:
  - boxes: the predicted boxes
  - labels: the predicted labels
  - scores: the scores of each detection

Task – Image detection

- PyTorch Lightning:
  - a lightweight PyTorch wrapper for high-performance AI research
- RacingF1Detector: a LightningModule wrapping our model
- RacingF1Dataset: class handling our datasets
  - random_split
  - preprocess_sample
- RacingF1DataModule: a LightningDataModule loading the datasets and wrapping them in DataLoaders
- CustomRescale: class for re-scaling samples to a given dimension

Task – Image detection

- Google Colab:
  - training and testing steps
  - compute model's outputs and draw the bounding boxes
  - training time **> 6 hours**
  - training stopped after **5 epochs**
  - very likely to timeout

Task – Video tracking

- OpenCV has been used to implement:
  - split_video_to_frames: given a video, this function splits a video into frames
  - generate_bounding_boxes: given a PyTorch model, a list of images, a score threshold and an IoU threshold, it predicts the bounding box and it draws the red rectangle in the image
  - merge_frames_to_video: given a path of images, the FPS, the function merges the frames into a video

Task – Tracking enhancement with histograms

- Exploit histograms distance to enhance the predictions
  - HISTCMP_INTERSECT: the distance method
  - calculate_histogram_from_coordinates: given an image and a list of coordinates, this function crops the image and then it calculates the histogram in the given coordinates
  - check_bounding_boxes: this function checks two frames at a time and it uses the histograms to understand if an object is in the same position and, eventually, it draws the bounding box

Tracking example – 1

Tracking example – 2

Tracking example – 3

SAPIENZA
UNIVERSITÀ DI ROMA

Thank you for your attention!

References

- LaSOT: A High-quality Large-scale Single Object Tracking Benchmark: arxiv.org/pdf/2009.03465.pdf
- Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks: arxiv.org/abs/1506.01497
- Multiple Objects Tracking Algorithms: github.com/huanglianghua/mot-papers
- Google Colab: colab.research.google.com
- PyTorch Lightning: pytorchlightning.ai
- OpenCV: opencv.org