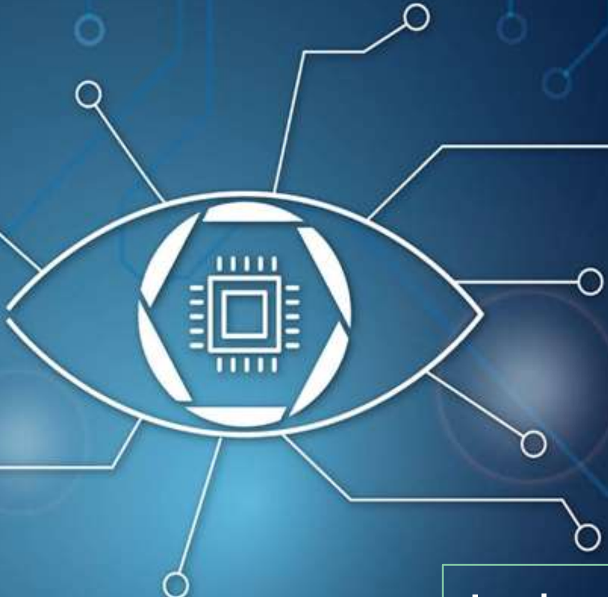
Intelligent Surveillance

A smart approach for the automatic surveillance with CCTVs

Authors:
- *Federico Matarante 2133034*
- *Serena Trovalusci 2128733*

O U T L I N E



- Introduction
- Dataset
- Proposed method
- Implementation details
- Metrics
- Final considerations

INTRODUCTION

Surveillance Systems

- Importance of CCTV cameras in maintaining security

Limitations of traditional surveillance:

- Human monitoring: fatigue, inefficiency, and delay
- Ineffective real-time crime prevention and intervention

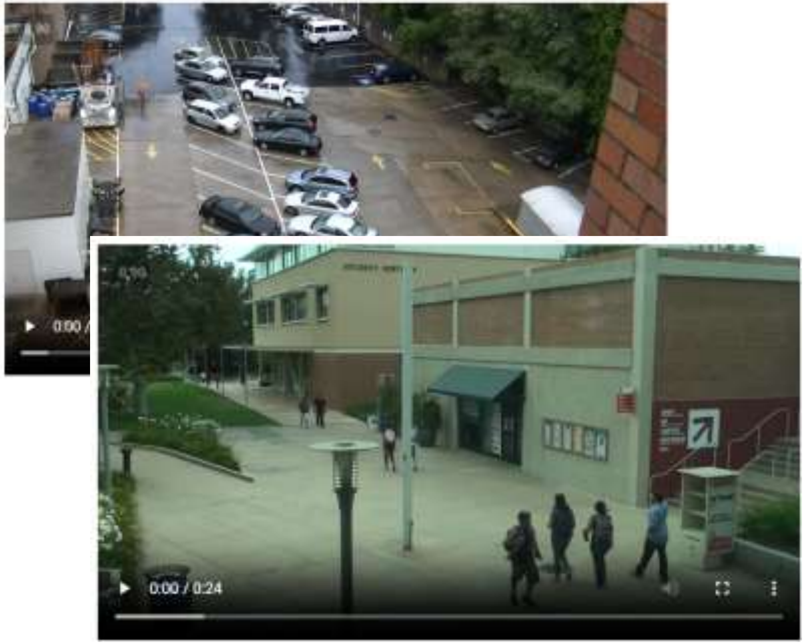
Advancements in Computer Vision:

- Automation of data processing and decision-making
- Use of image analysis and machine learning for real-time monitoring
- Potential to identify and respond to violent situations proactively

Virat Dataset

Dataset Structure and annotations

- 250 hours of ground camera videos
- 12.5 hours of annotated data of tracked objects and actions
- 13 labels of tracked objects
- 41 labels of events
- For implementation issues it’s added one more extra label of event “No event”



[VIRAT Video Data (viratdata.org)]

Virat Dataset

Dataset preprocessing for object tracking

Preprocessing steps:

1. Removed consecutive frames “too similar”. Similarity is measured by the average movement of the bounding boxes.
2. Frames sampled to images (with sampling rate) and for each a new annotation is created.
3. Single images processing: resizing, bilateral filtering and padding.

Virat Dataset

Dataset preprocessing for action recognition

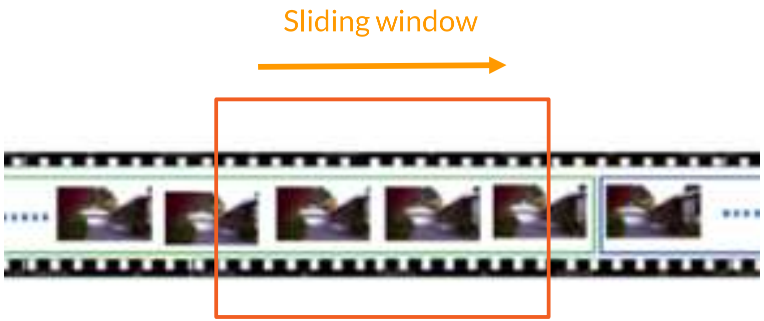
Preprocessing steps:

1. Each video of the dataset is divided in N sub-videos, one for each registered action only keeping the frames in which the action is contained (plus an offset)
2. The sub-videos are then cut only in the portions containing the event (plus an offset in every direction). Also random portions are extracted for the “No event” label.
3. Further processing is then done on the single videos to improve quality, to reduce size and to adapt it to the models. The steps are: frame sampling, resizing, bilateral filtering and padding.

Proposed Method

Detector algorithm steps:

OBJECT TRACKING → SLIDING WINDOW → CLUSTERS → ZOOM-IN → ACTION RECOGNITION



Object detection: YOLO

YOLOv8 architecture

- **BACKBONE:** CSPDarknet53
- **NECK:** FPN (Feature Pyramid Network), PAN (Path Aggregation Network)
- **HEAD:** predictions
  - **OUTPUT FORMAT->** tensor: **(N,N,Bx(5+nc))**
    - **feature map ->** NxN grid
    - **anchor boxes** x grid= B
    - **Bounding Box =** (x,y,w,h) + objectness score
    - **nc =** number of classes

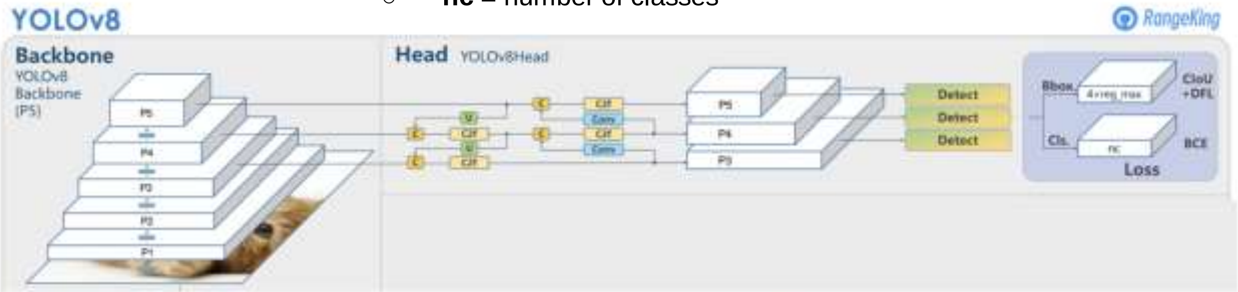


Figure 7: YOLOv8 Architecture [19]

[2305.09972] Real-Time Flying Object Detection with YOLOv8 (arxiv.org)

Object detection: YOLO

YOLOv8 Training

1. <u>Data preprocessing:</u>

- **PREPROCESSED IMAGES**
- **PREPROCESSED ANNOTATIONS:** (label, x_center, y_center, width, height)

1. <u>Fine-tuning:</u>

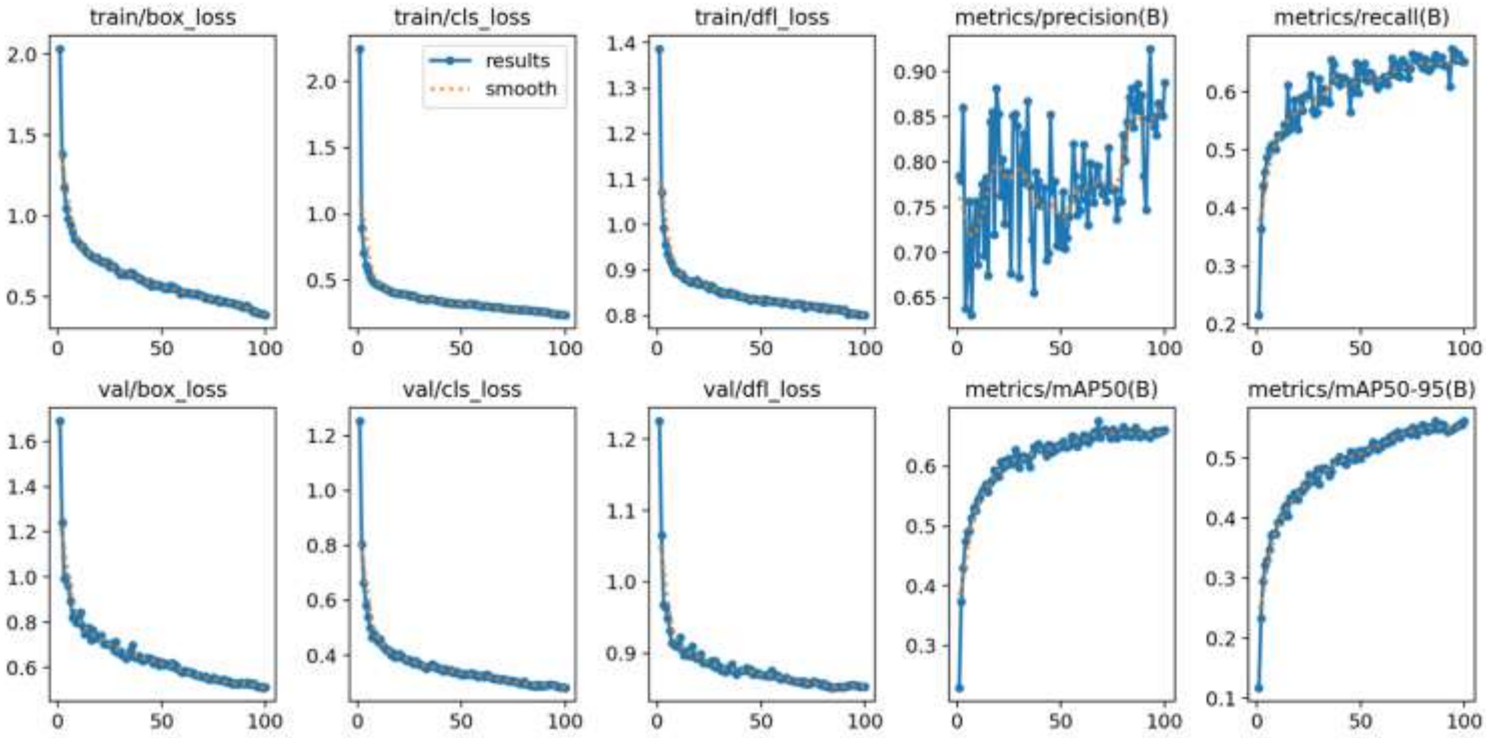
   pre-trained model by *Ultralytics*: **yolov8m.pt:** https://docs.ultralytics.com/models/yolov8/#how-do-i-train-a-yolov8-model

- 100 epochs
- No data augmentation (large dataset)

METRICS

EPOCH	PRECISION	RECALL	mAP50	mAP50-95
1OO th	0.8875	0.65155	0.66074	0.56212

METRICS



train/box_loss
train/cls_loss
train/dfl_loss
metrics/precision(B)
metrics/recall(B)
results
smooth
val/box_loss
val/cls_loss
val/dfl_loss
metrics/mAP50(B)
metrics/mAP50-95(B)

Object Tracking

- *Tracking functionality integrated within the YOLO object detection framework.*

INPUT:

video
or
sequence of frames

YOLO

OBJECT DETECTION MODEL

detects objects
in each frame

TRACKS INITIALIZATION WITH **UNIQUE IDS**

DETECTION **MATCHING**

TRACK **UPDATE**

position
+
ID

OUTPUT:

Action recognition

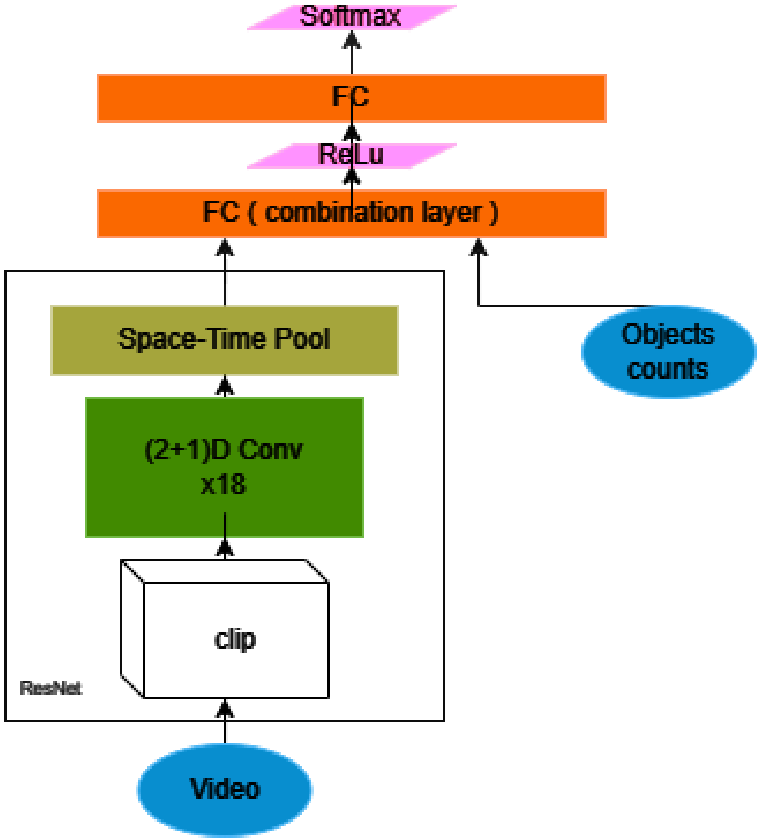
Model proposed

Architecture schema:

- Video processed by a R2Plus1-18 model.
- Objects count combined with video features
- Fully Connected layer used for classification

R2Plus1-18 key points:

- Uses (2+1)D residual convolutions - 2D convolution on space and 1D convolution on time..
- 18 layers of (2+1)D convolution
- It's pre-trained on the Kinetic dataset

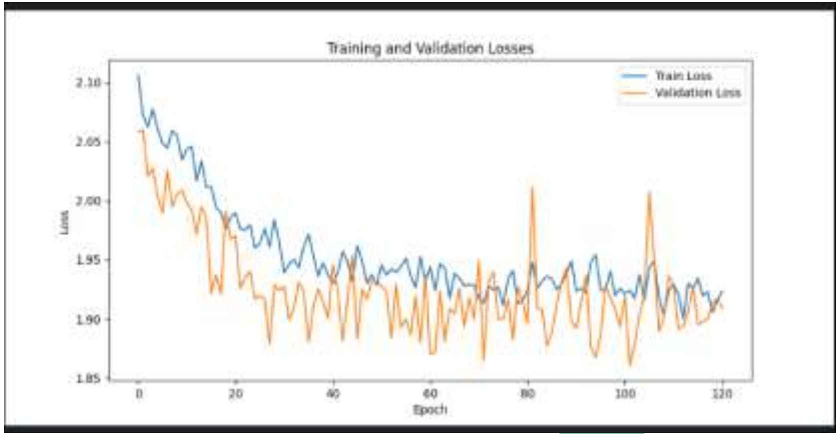


Action recognition

Model training process

Mostly for efficiency reasons:

- All the layers have been frozen but the classifier and the last (2+1)D layer
- Sub-portion of the dataset used
- Only 12/41 event classes have been trained
- Small batch size used
- No data augmentation done



Accuracy	F1 Score	Recall	Precision
40.28%	0.2926	25.37%	40.28%

Model evaluation

LR	Optimizer	Batch Size	Epochs	Dataset Size	Sub-Dataset Size
0.001	Adam	2	120	4000	2000

Training settings

Final Considerations

The data and the training process have been scaled down considerably to compensate for the low performance of the available machines, impacting on final models' accuracy.
With better hardware, some improvements could have made by:

- Downsampling less the images and videos
- Data augmentation of videos and images
- Noisy data for Action Recognition network (for example wrong object counts)
- Using more epochs, batch size and dataset size for the training process
- Possibly adding a deeper classifier at the end of the Action Recognition network
- Considering moving to a 3D convolution network for classification and comparing performances

Thanks for your attention!