Pedestrian Detection Final Presentation

Purdue University - Vertically Integrated Projects

Image Processing and Analysis - Fall 2024

Advisors: Prof. Carla Zoltowski, Prof. Edward Delp

Vertically Integrated Projects

Members

Donny Weintz
Computer Engineering

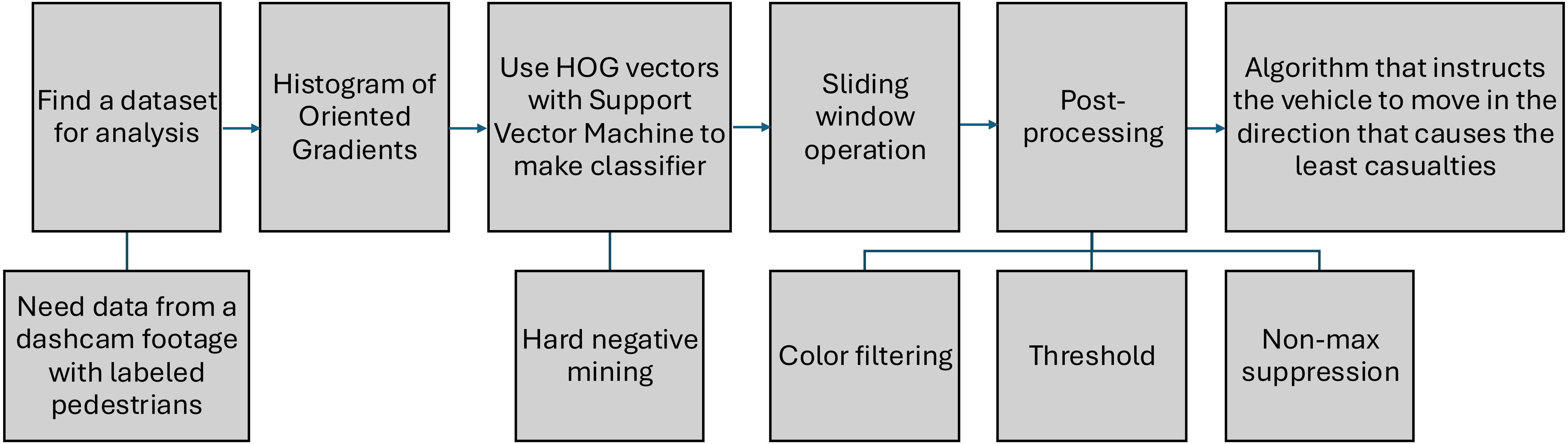
Aditya Mallepalli
Computer Science

Jianing Wang
Computer Science

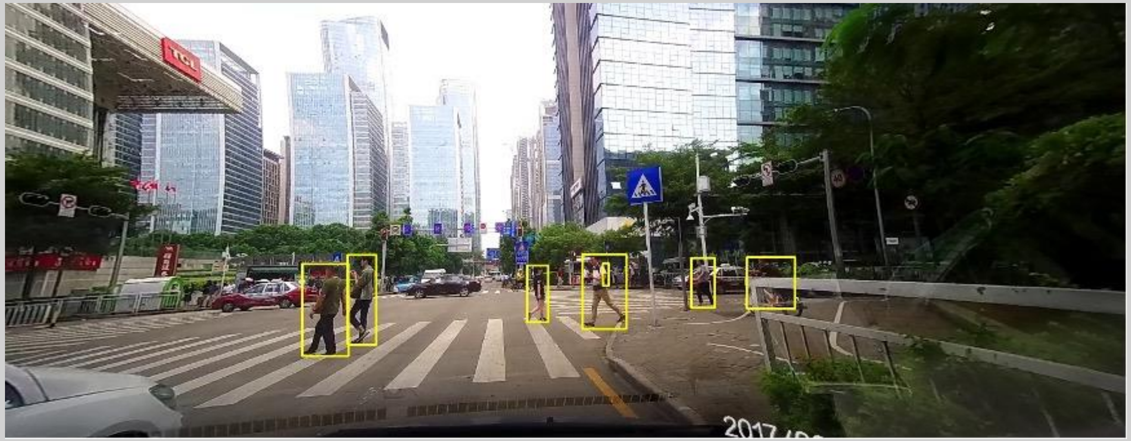
Preetham Yerragudi
Computer Science

Aakarsh Rai
CS DS double major

Project Flow Diagram



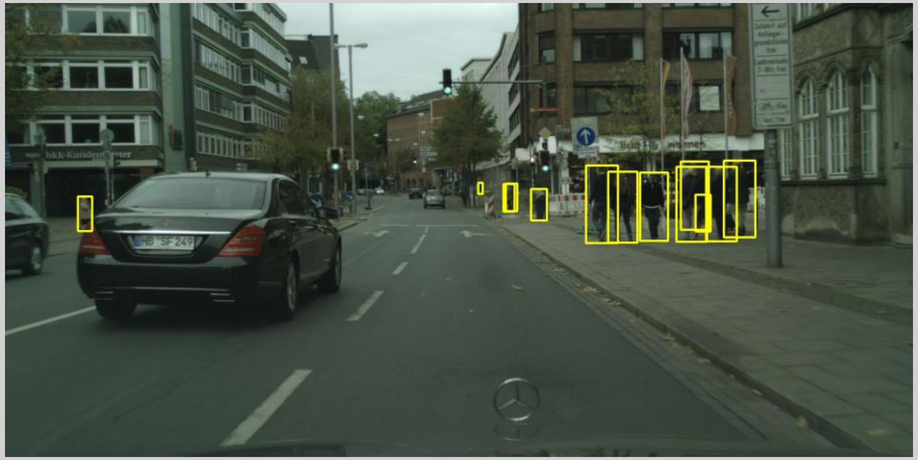
Dataset



WIDER Dataset

- 88,260 images
- Images taken from wide angle camera
- Lighting conditions vary across images

Switched to



Cityscapes Dataset

- 3,475 images
- Images taken without wide angle lens
- Lighting conditions are similar across images

Histograms of Oriented Gradients

Preprocessing
- Grayscale image
- Resize to 1:2 width to height ratio

Gradients
- Sobel operator

x-direction

-1	0	1
-2	0	2
-1	0	1

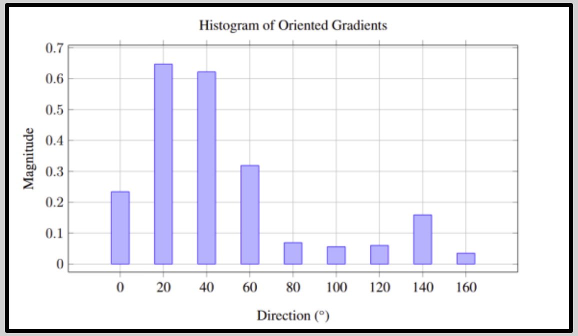
y-direction

-1	-2	-1
0	0	0
1	2	1

Magnitude and Direction
- Gradient Magnitude $g = \sqrt{g_x^2 + g_y^2}$
- Gradient Direction $\theta = \arctan\left(\frac{g_y}{g_x}\right)$

Break into cells
- We used 8x8 pixel cells

Make Histogram for Each Cell
- Direction on x-axis (0-160 degrees)
- Place magnitude values into appropriate bins



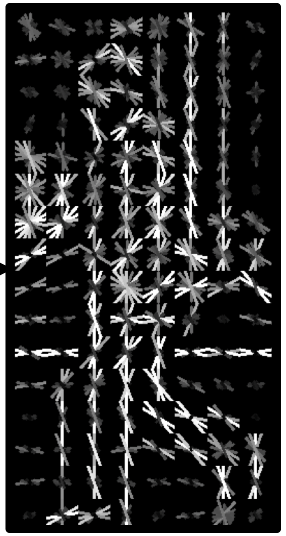
Block Normalization
- Create 2x2 blocks of adjacent cells
- Normalize feature vectors in each block

$$\vec{v}_{norm} = \frac{\vec{v}}{\sqrt{\sum_i (x_i^2)}}$$

$\vec{v} = 36x1\ feature\ vector$
$x_i = ith\ element\ in\ \vec{v}$

Original Image

HOG Features



Support Vector Machine (SVM)

- An algorithm that finds the optimal hyperplane to correctly classify data.

$$w = weight\ vector$$

$$x = input\ vector$$

$$b = intercept$$

Equation of hyperplane in n-dimensional space: $\boxed{w \cdot x + b = 0}$

$w \cdot x + b = 0$ → Value is on hyperplane

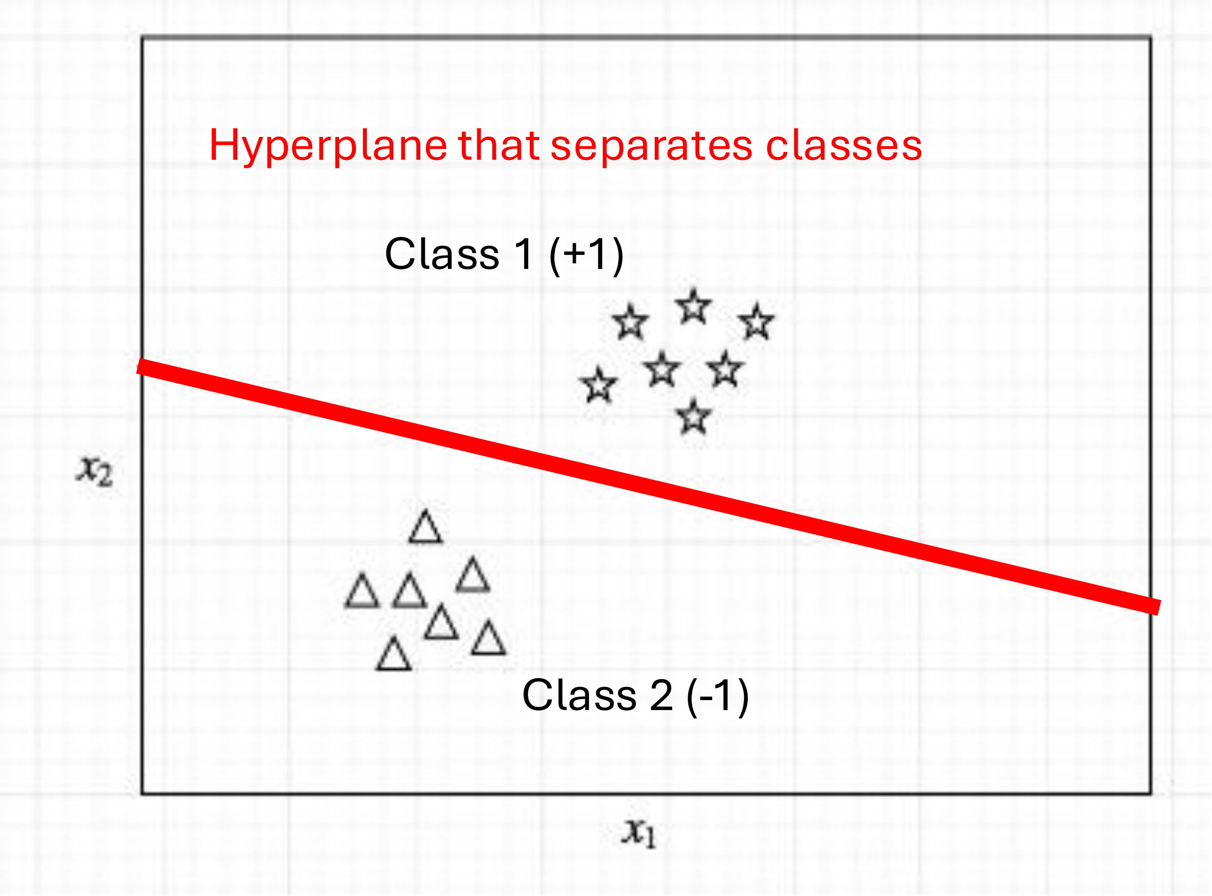
$w \cdot x + b \geq 0$ → Value is positive class

$w \cdot x + b \leq 0$ → Value is negative class

Classify based on location with respect to hyperplane

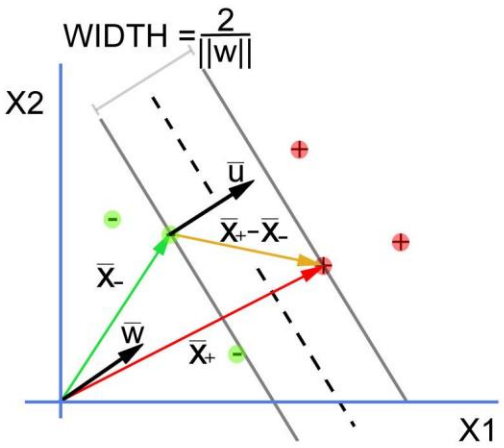
$$h(x_i) = \begin{cases} +1 & if\ w \cdot x + b \geq 0 \\ -1 & if\ w \cdot x + b \leq 0 \end{cases}$$

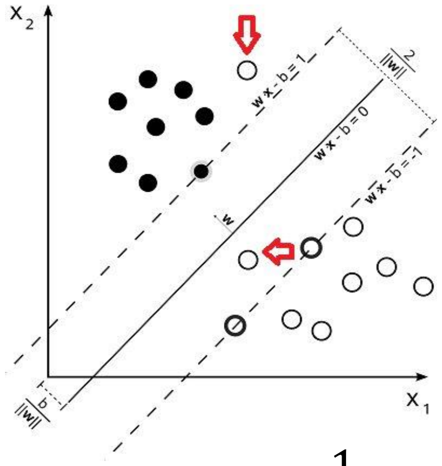
Hyperplane in 2 dimensions



Soft Margin

Hard Margin





$$Maximize \ \frac{2}{||w||} \rightarrow Minimize \ \frac{1}{2}||w||^2$$

$$Minimize \ \frac{1}{2}||w||^2$$

$Minimize\ Hinge\ Loss$
$= \max(0, 1 - y_i(w \cdot x_i + b))$

The SVM classifier assigns a confidence score $c(x)$ to each point x.
The confidence score is given by:

$$c(x) = \frac{(w \cdot x + b)}{\| w \|}$$

Parameter Update for Hinge Loss

$$\min_{w,b}\left(\frac{1}{n}\sum_{i=1}^{n}\max\big(0, 1-y_i(w\cdot x_i+b)\big)+\frac{\lambda}{2}\|w\|^2\right)$$

Objective (Soft Margin SVM Equation)

$$\max\big(0, 1-y_i(w\cdot x_i+b)\big)$$

Hinge loss term at point *i*

$\eta = step\ size$

$\lambda = positive\ constant$

$y_i = classification\ at\ point$

Optimize objective to minimize hinge loss. Two Scenarios:

- **Scenario 1:** point is incorrectly classified or inside margin.
- **Scenario 2:** point is correctly classified.

We compute new values for w and b. We repeat this process until convergence.

$$y_i(w\cdot x_i+b) < 1 \longrightarrow \nabla_w = -y_i x_i,\quad \nabla_b = -y_i$$

$$y_i(w\cdot x_i+b) \geq 1 \longrightarrow \nabla_w = 0,\quad \nabla_b = 0$$

$$w_{new} = w - \eta\left(\frac{1}{n}\sum_{i=1}^{n} -y_i x_i + \lambda w\right)$$

$$b_{new} = b - \eta\left(\frac{1}{n}\sum_{i=1}^{n} -y_i\right)$$

$$w_{new} = w - \eta\lambda w$$

$$b_{new} = b$$

Sliding window

Given an input image of dimensions $H \times W$, the sliding window is applied over a grid of coordinates (x, y) with a fixed window size $w \times h$.

Let the set of all possible window coordinates be:

$$W = \{(x, y) \mid x \in [0, W - w], y \in [0, H - h]\}$$

Given step size s:

$$x_i = x_{i-1} + s,$$
$$y_i = y_{i-1} + s$$

$$f(x, y) = HOG\big(w(x, y)\big)$$

Window contains pedestrian if:

$$c\big(f(x, y)\big) > T$$

where $T = threshold$

8 x 8 pixel cells

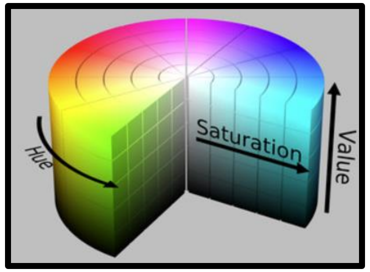
16 x 16 pixel cells

Color Filtering

- Want to flag the colors typically associated with trees (green/brown)

$P_C = Percentage\ of\ pixels\ in\ hue\ range$
$T = threshold$

$$\text{Classification} = \begin{cases} \text{Not a pedestrian} & \text{if } P_C \geq T, \\ \text{Potential pedestrian} & \text{if } P_C < T. \end{cases}$$



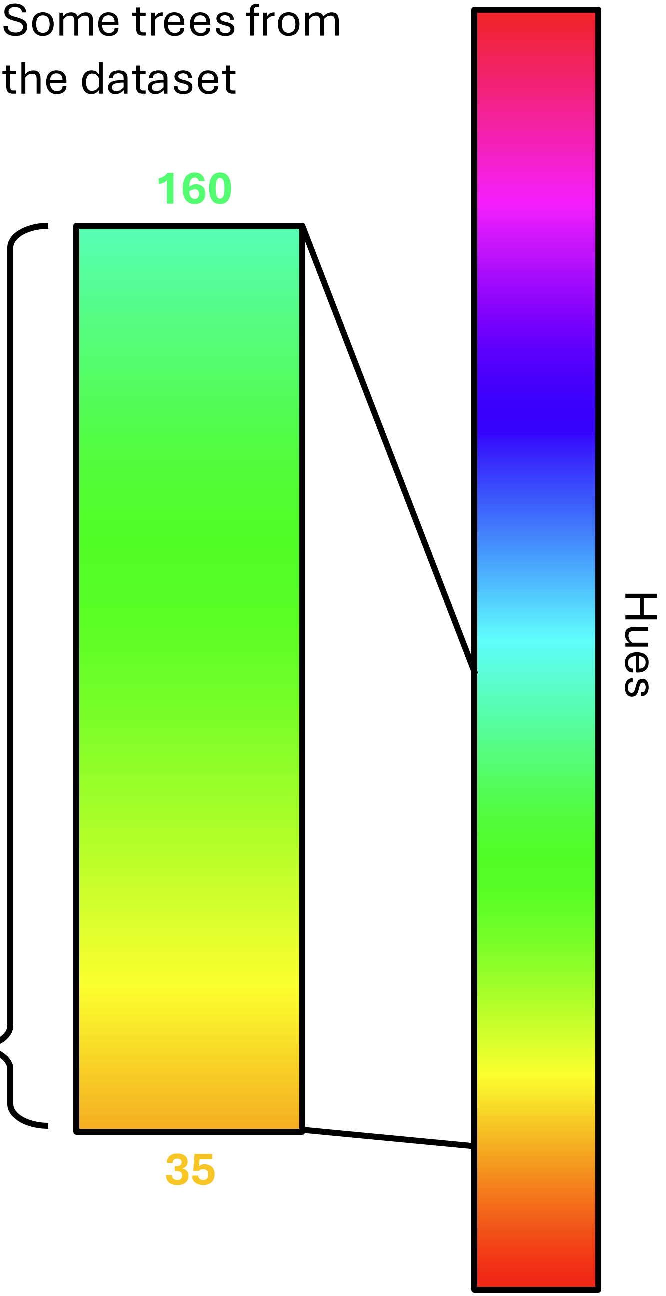
SharkDderivative work: SharkD [CC BY-SA 3.0 or GFDL], via Wikimedia Commons

Some trees from the dataset

Filtering Parameters

- **Score Threshold:** 1.2
- **Green/Brown Threshold:** 30%
- **HSV Color Range:**

  (35, 30, 0) to (160, 255, 255)



Chose Hues between 35 and 160 to correspond to green/brown

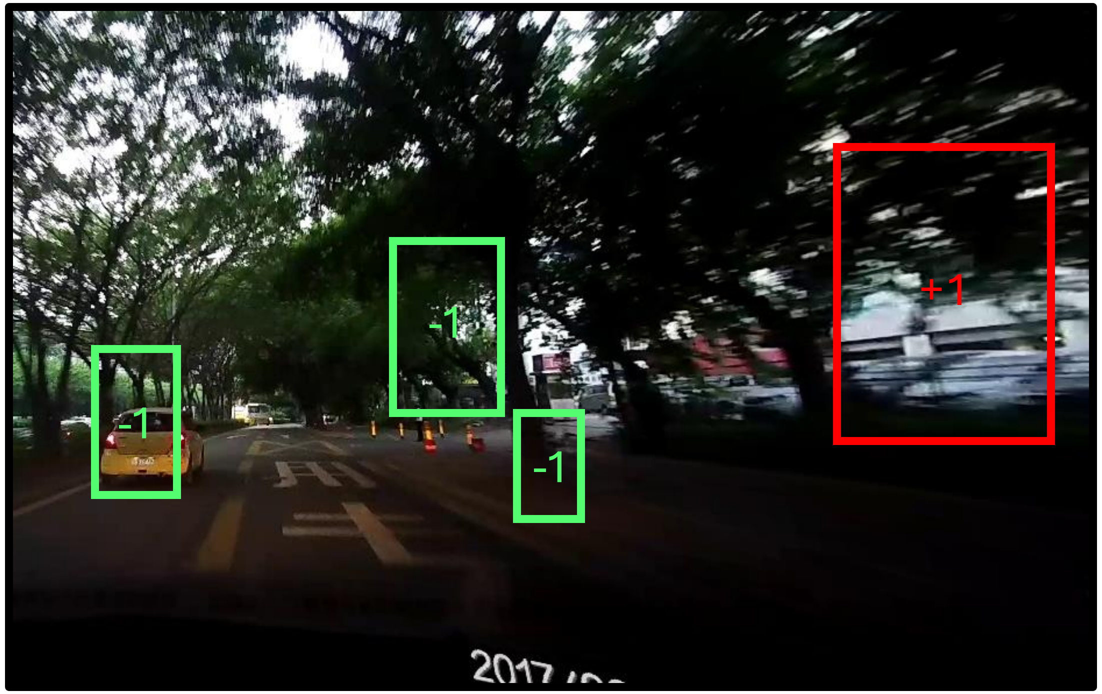
Hard Negative Mining

Train an initial model with part of the training dataset

Test it on negative samples from more of the training dataset

Retrain the model with all the additional false positive misclassifications

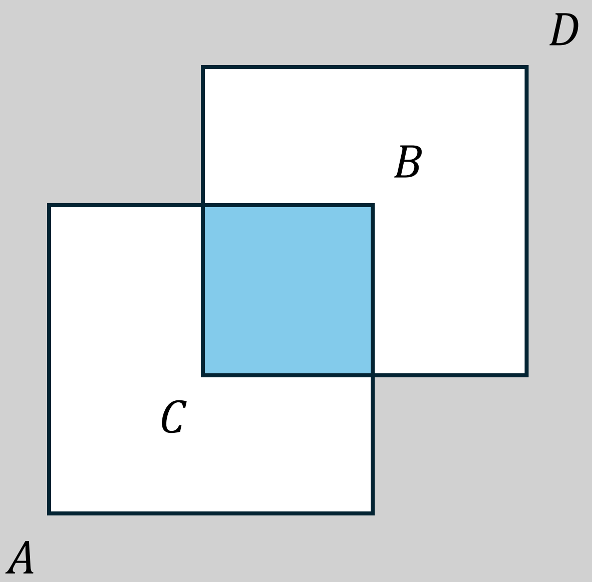
This forces the model to target specific features that distinguish between pedestrians and non pedestrians



One false positive classification

Intersection Over Union/Jaccard Index

Intersection

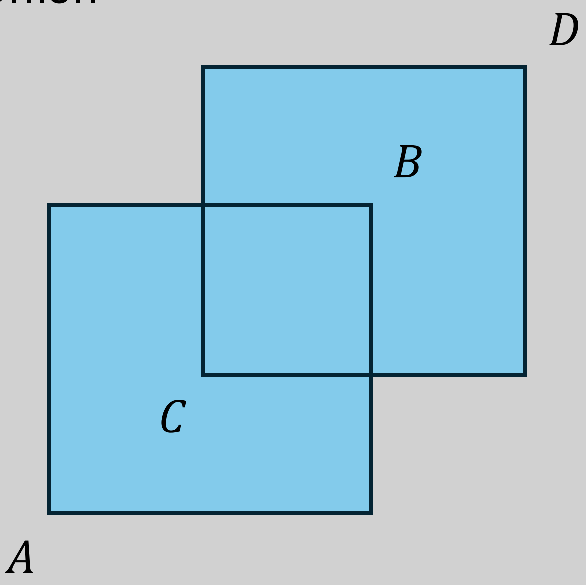


$$width = \min(B_x, D_x) - \max(A_x, C_x)$$

$$height = \min\left(B_y, D_y\right) - \max(A_y, C_y)$$

$$intersection = width \cdot height$$

Union



$$area_{AB} = (B_x - A_x)(B_y - A_y)$$

$$area_{CD} = (D_x - C_x)(D_y - C_y)$$

$$union = area_{AB} + area_{CD} - intersection$$

$$IOU = \frac{intersection}{union}$$

Non-Maximum Suppression

$s = decision\ score\ (distance\ from\ hyperplane)$

$$\begin{bmatrix} A_x & A_y & B_x & B_y & s_{AB} \\ C_x & C_y & D_x & D_y & s_{CD} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

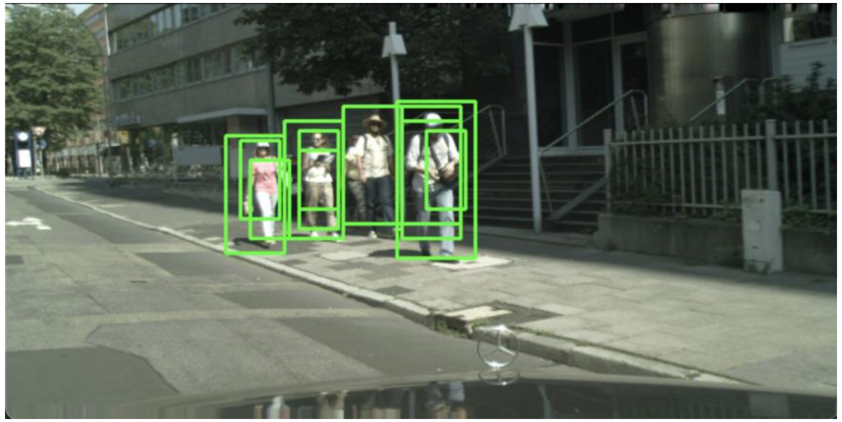
Sorted based on s in descending order

Traverse down the list, computing each box's IOU with the boxes further down the list

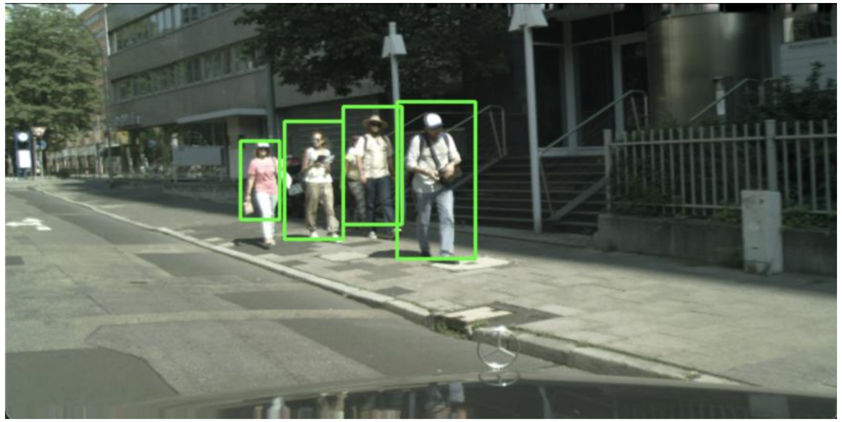
If the IOU is greater than a threshold, remove the box with the lower s

$threshold = 0.3$

Before NMS



After NMS



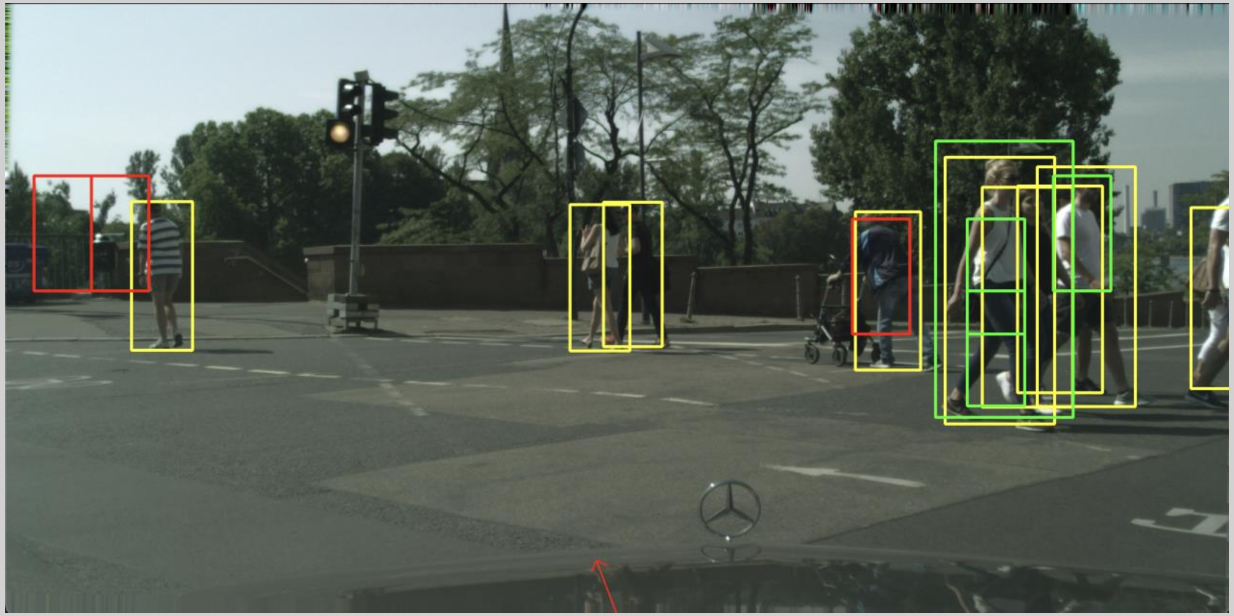
F1 Score

Measures the predictive performance of binary classification models

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{TP} + \text{False Positives (FP)}}$$

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{TP} + \text{False Negatives (FN)}}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$



$IOU\ Threshold = 0.3$

Ground Truths

False Positives

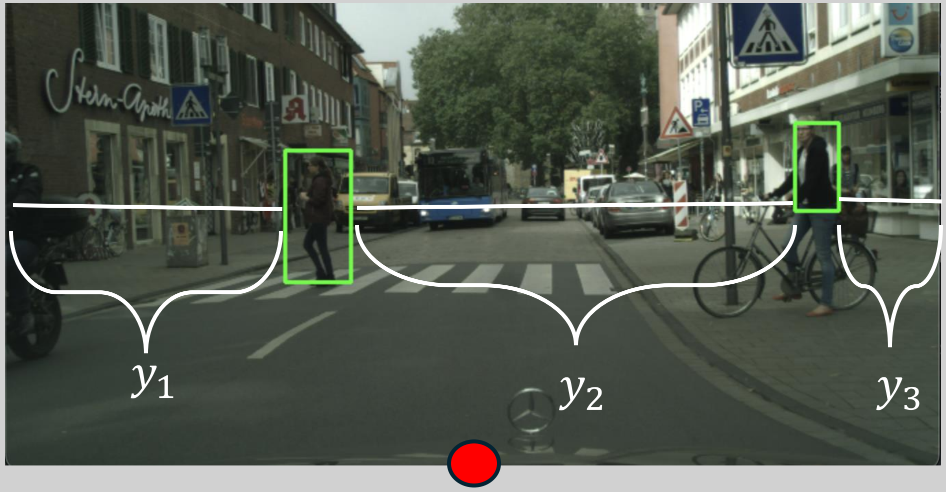
True Positives

$Precision = 0.57$

$Recall = 0.44$

$F1\ Score = 0.50$

Final Output

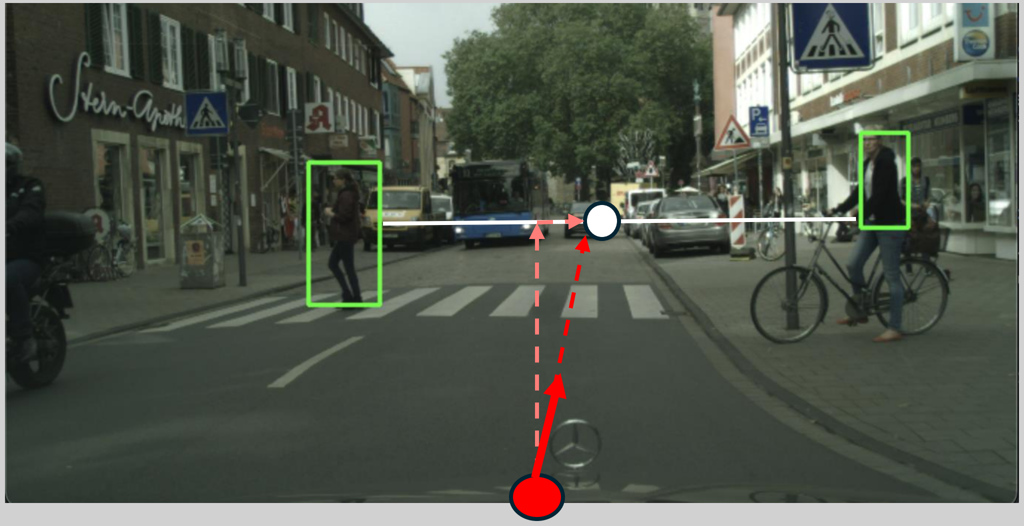


C

$$n = \#\ detected\ pedestrians$$
$$y_i = image\ sections\ with\ no\ pedestrians$$
$$i = n + 1$$
$$L(y_1) = length\ of\ section\ in\ pixels$$

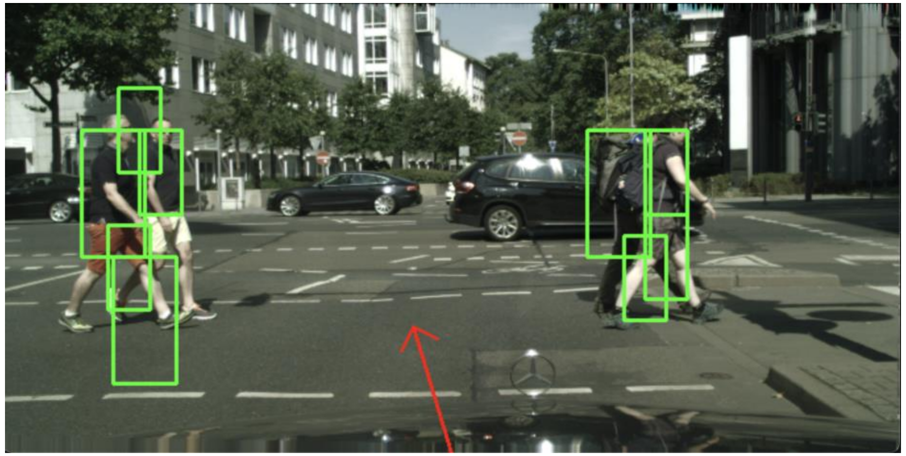
$$y_{max} = \max_i L(y_i)$$
$$w = image\ width$$
$$C = \left(\frac{w}{2}\ px, 0\ px\right)$$
$$Mid(y_i) = Midpoint\ of\ y_i$$
$$d = direction\ vector$$
$$d_u = direction\ unit\ vector$$

$$y_{max} = y_2$$
$$\Delta x = Mid(y_{max})_x - C_x$$
$$\Delta y = Mid(y_{max})_y - C_y$$
$$d = (\Delta x, \Delta y)$$
$$d_u = \frac{d}{|d|} = \frac{d}{\sqrt{(\Delta x^2 + \Delta y^2)}}$$

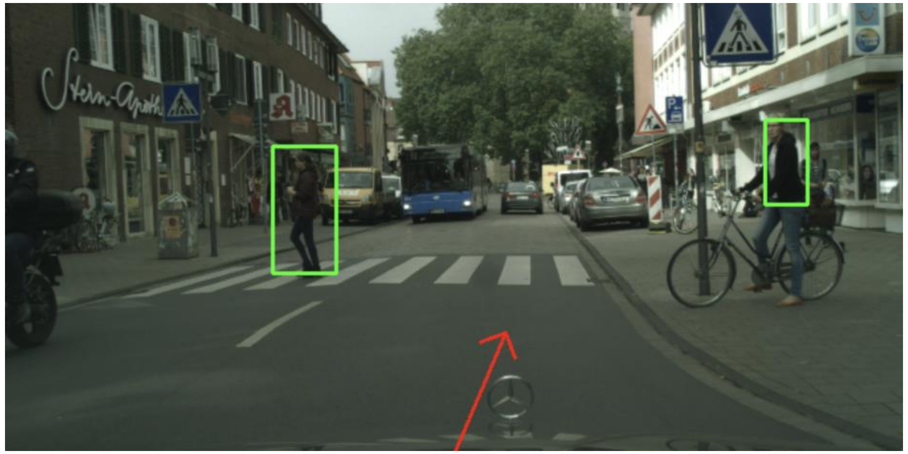


C

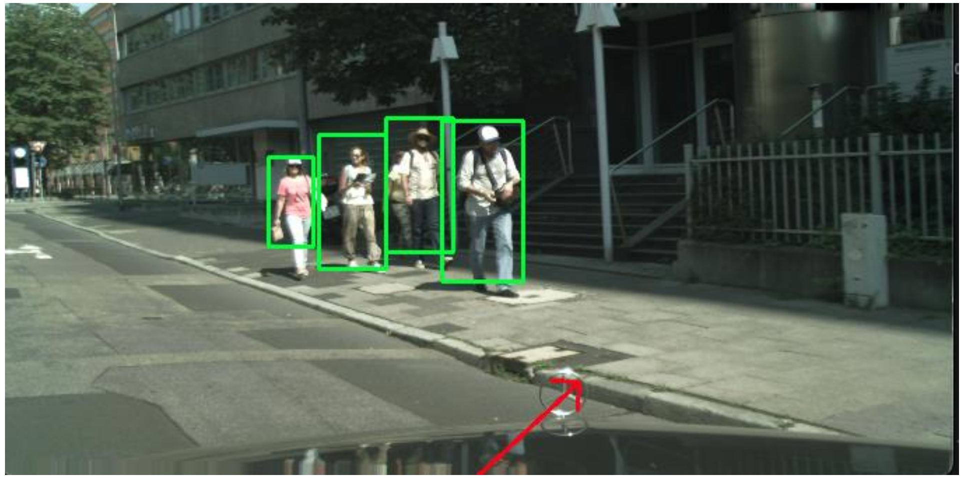
Final Output



$d_u = (-0.29, -0.96)$



$d_u = (0.39, -0.91)$



$d_u = (0.74, -0.67)$

References

- https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1467360
- https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/resources/lecture-16-learning-support-vector-machines/
- https://people.csail.mit.edu/dsontag/courses/ml16/slides/lecture5.pdf
- https://www.cityscapes-dataset.com/downloads/
- https://competitions.codalab.org/competitions/20132
- https://ieeexplore.ieee.org/document/1699659
- https://pmc.ncbi.nlm.nih.gov/articles/PMC9455026/
- https://arxiv.org/abs/2410.02212