Exercise 1

About the Exercise Session

- 2 weeks for each exercise + Office hours (OH) for questions in between

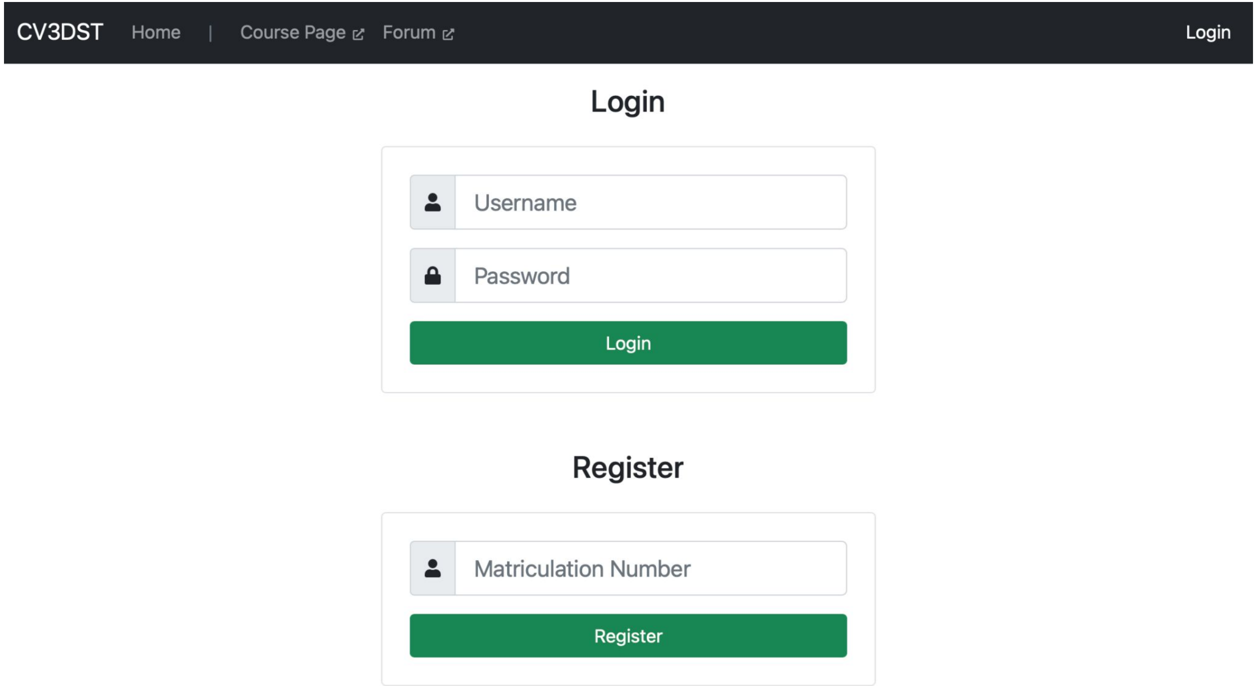
~~Exercise 00~~	~~OH~~	Exercise 01	OH	Exercise 02	OH	Exercise 03	OH
19.10	26.10	02.11	09.11	16.11	23.11	30.11	14.12

Holidays and New Year

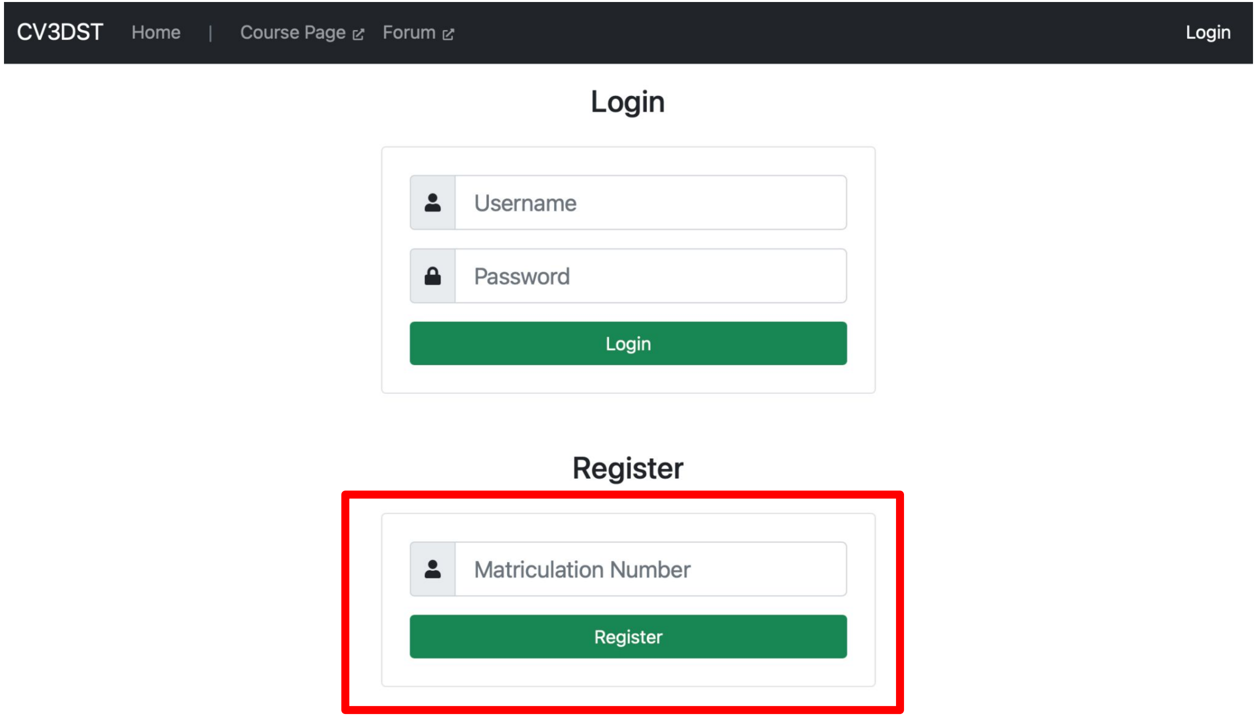
Exercise 04	OH	Exercise 05	OH
11.01	18.01	25.01	01.02

Deadline always 23:59 CET on due date

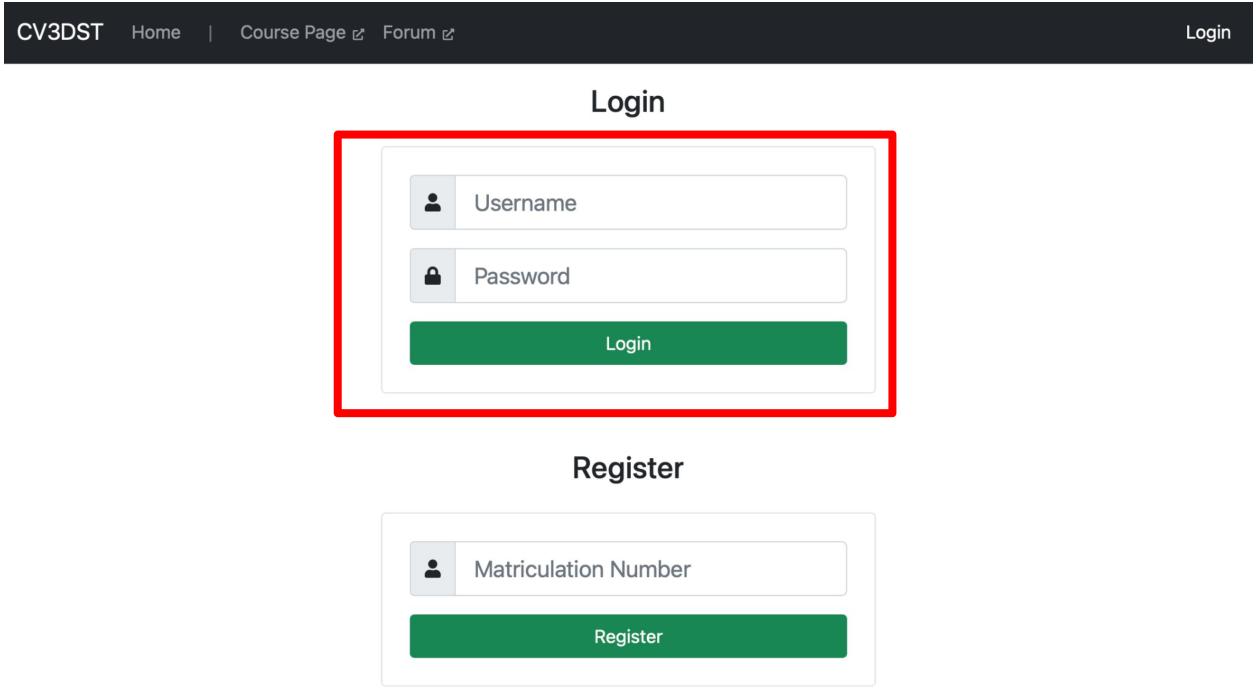
Submission Page



Submission Page



Submission Page



Exercise 1

Exercise submission

Exercise 00 – Introduction [Optional]

Exercise 01 – Object Detection

Info

- **Description:** Object Detection
- **Start:** 2023-11-02 10:00:00
- **Deadline:** 2023-11-15 23:59:59
- **Requirement:** 50.0
- **Max Score:** 100.0
- **Submission:** Within the working period you can submit solutions as often as you want
- **Bonus:** The best score counts for the bonus
- **Evaluation Time:** The maximum evaluation time is 30min.
- **Issues:** In rare cases it can still happen that your submission will get stuck in "queued". If your submission is there for more than 10min, please submit again. - Sorry for the inconveniences.

Exercise 02 – Hungarian Tracking

Exercise 03 – ReID with GNN

Exercise 04 – Semantic Segmentation

Exercise 05 – Unsupervised Segmentation Segmentation

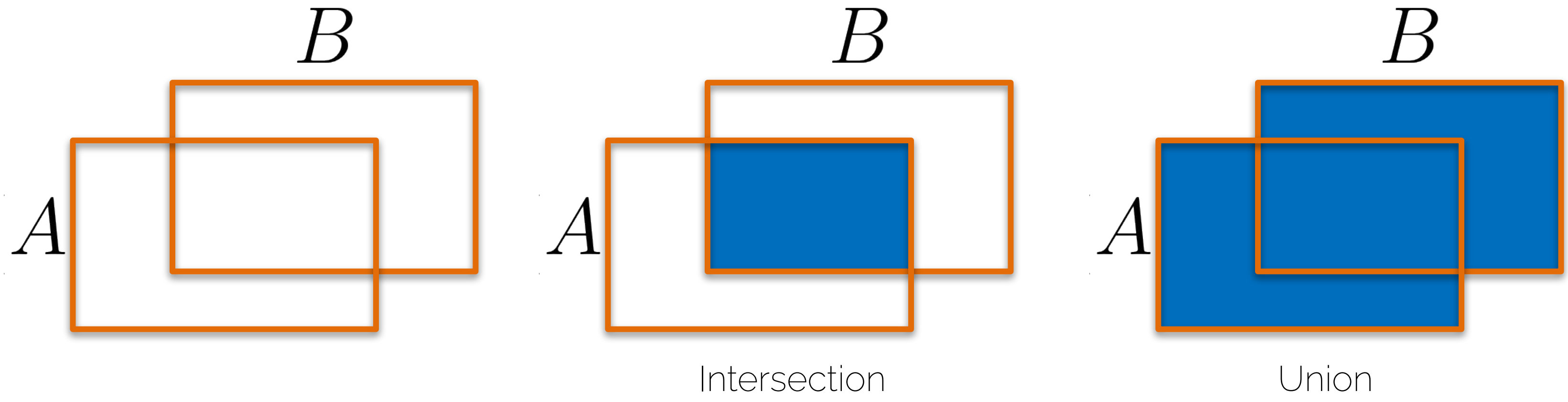
Exercise 1

- Object detection with HOG classifiers
  - computing the IoU
  - creating a histogram of oriented gradients
  - sliding window detection
  - non-maximum suppression
- Optional: Comparison with a state of the art object detector (FasterRCNN)

Task 1 - IoU

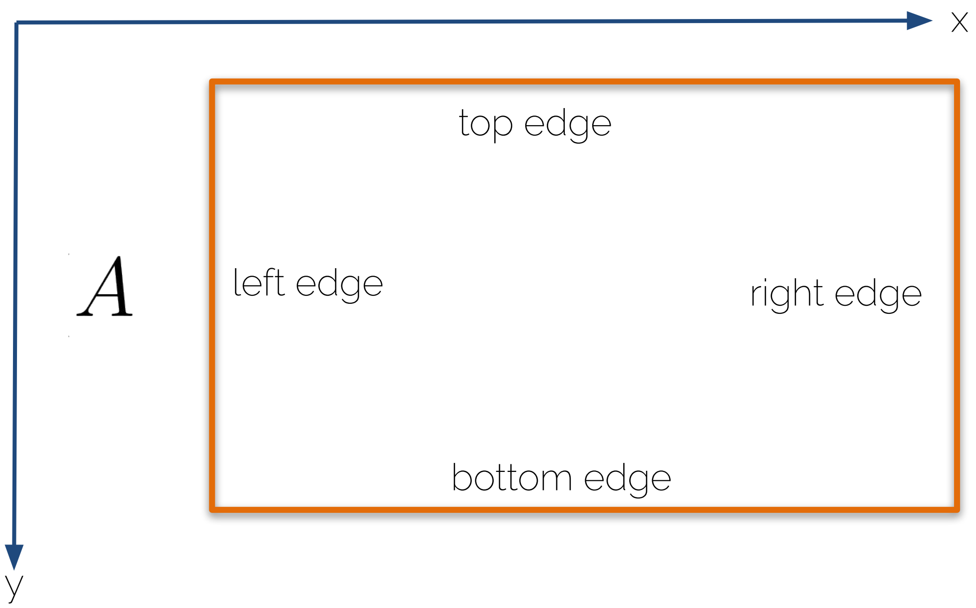
- also called the Jaccard Index
- compute the overlap of two regions

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$



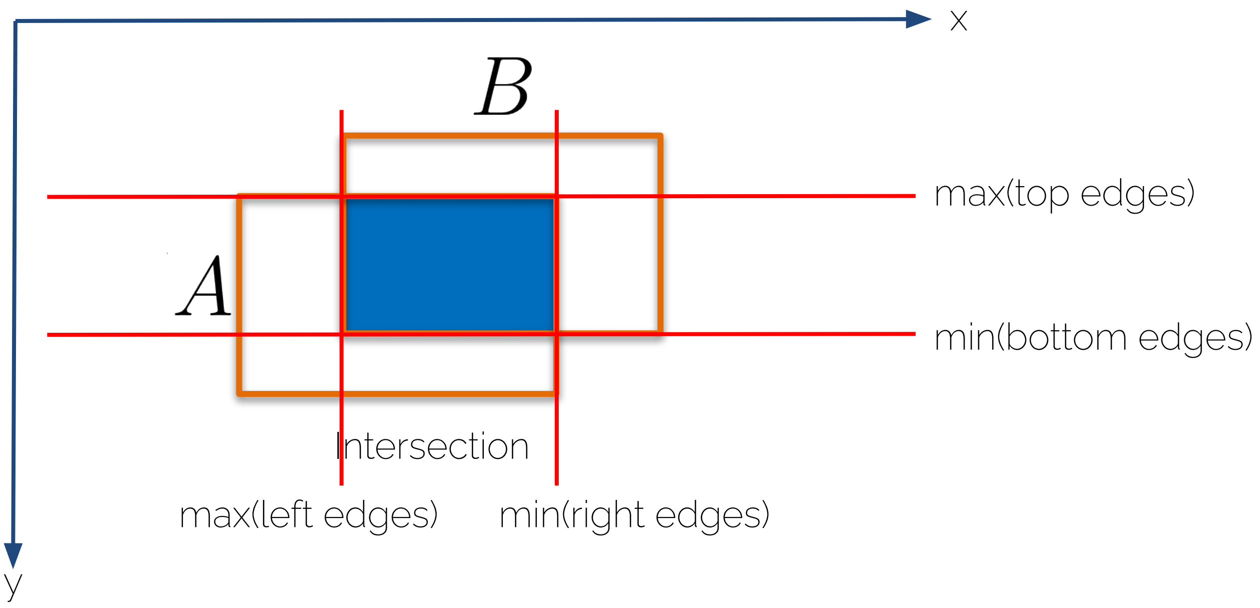
Task 1 - IoU

conventions



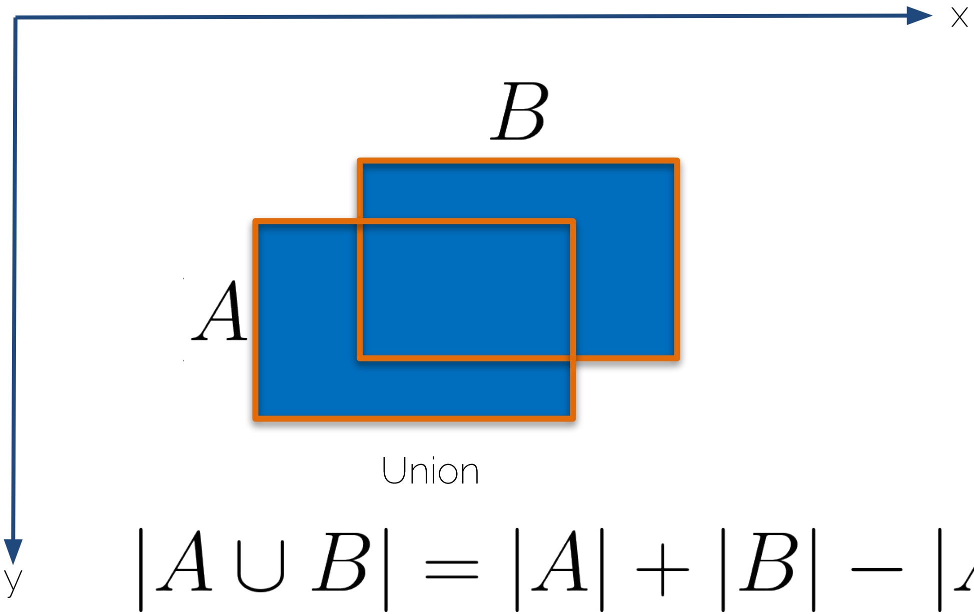
Task 1 - IoU

- intersection



Task 1 - IoU

- union



$$|A \cup B| = |A| + |B| - |A \cap B|$$

Task 2 - Compute Image Gradient

- computed with central difference
- first compute the gradients in x and y direction
- be mindful of the orientation of the coordinate system of the image and the order of row/column of pytorch tensors
- add padding (reflective) to get the correct output size

$$\nabla_x I(x,y) = I(x+1,y) - I(x-1,y)$$
$$\nabla_y I(x,y) = I(x,y+1) - I(x,y-1)$$

Task 2 - Compute Image Gradient

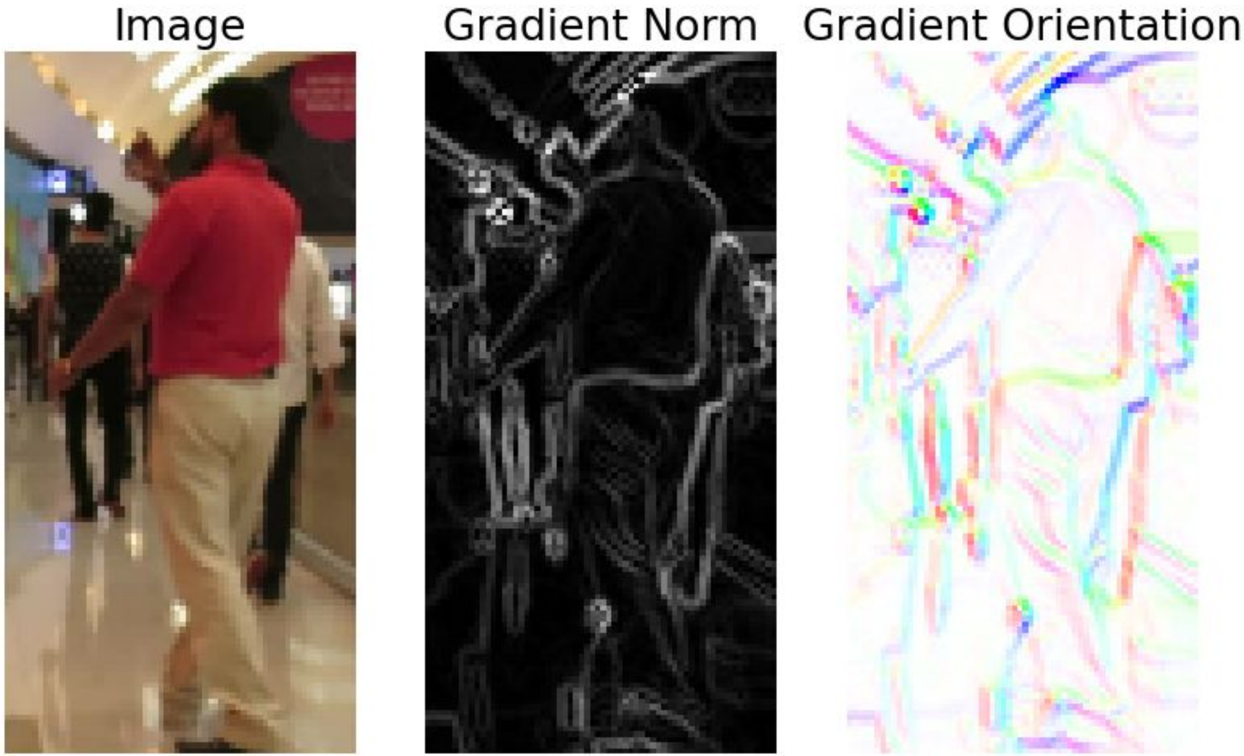
- compute the norm and angle

$$\nabla_{\mathrm{n}} I(x,y) = \left\| \begin{pmatrix} \nabla_x I(x,y) \\ \nabla_y I(x,y) \end{pmatrix} \right\|_2$$

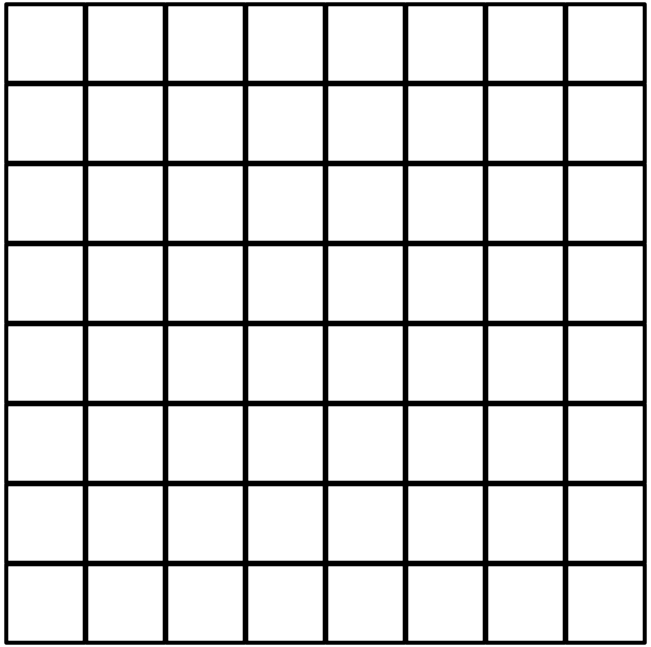
$$\nabla_{\mathrm{a}} I(x,y) = \arctan \frac{\nabla_x I(x,y)}{\nabla_y I(x,y)} \quad *$$

* Be mindful of dividing by zero (there might be a nice default implementation)

Task 2 - Compute Image Gradient



Task 3 - Filling the HOG bins

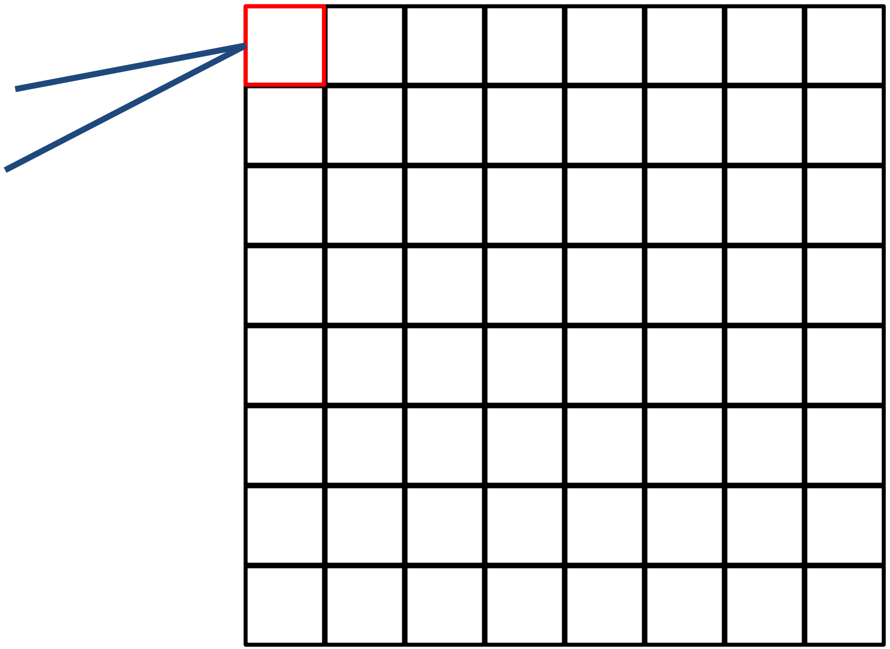


0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

0° 180°

Task 3 - Filling the HOG bins

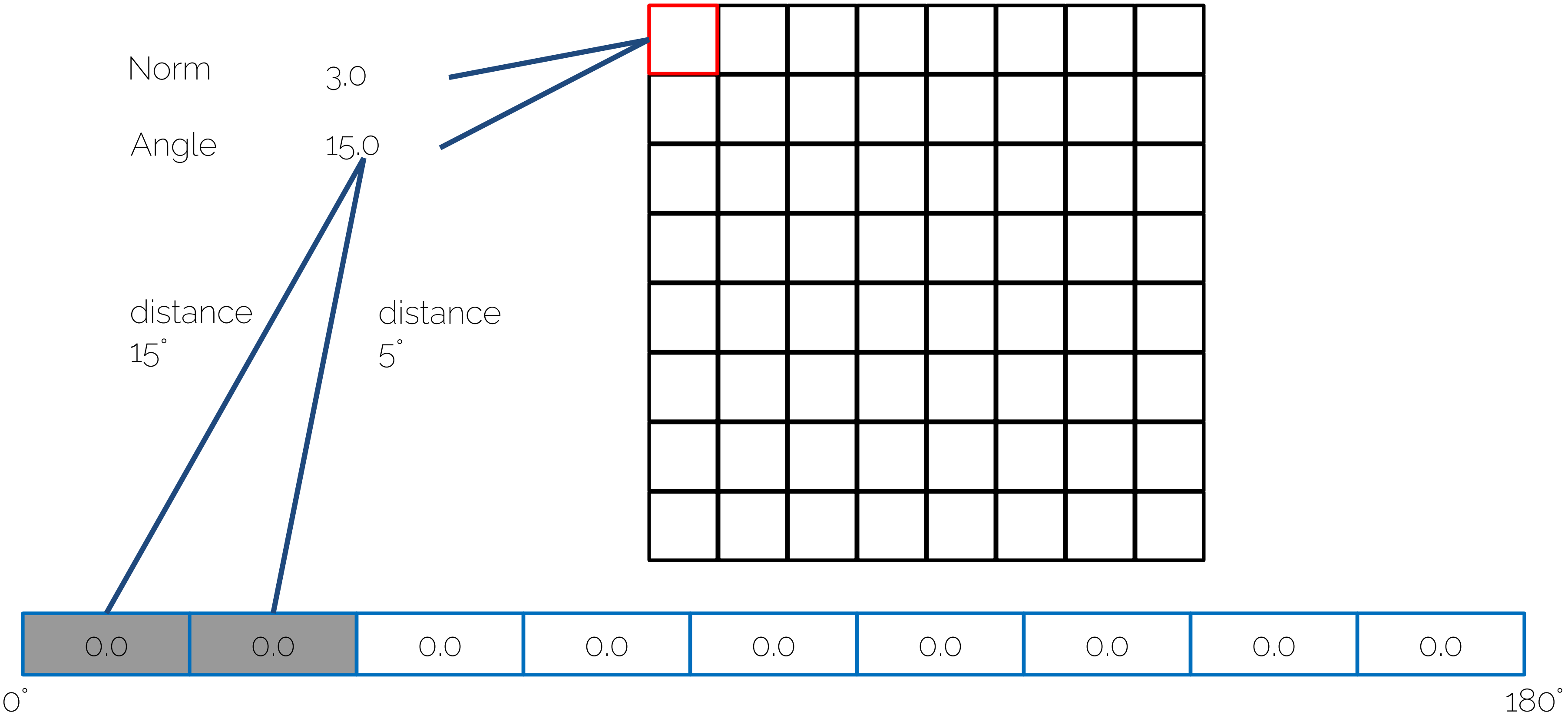
Norm	3.0
Angle	15.0



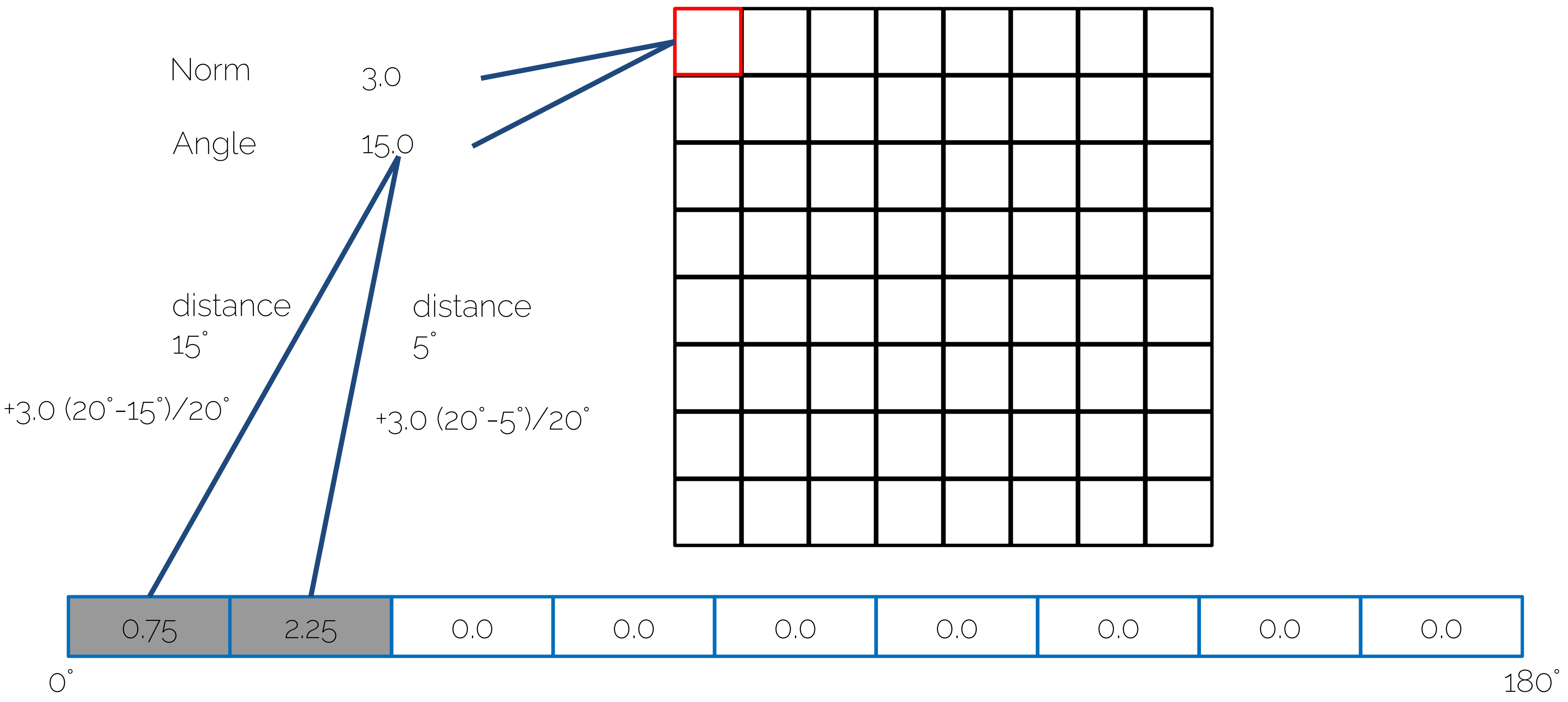
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

0° 180°

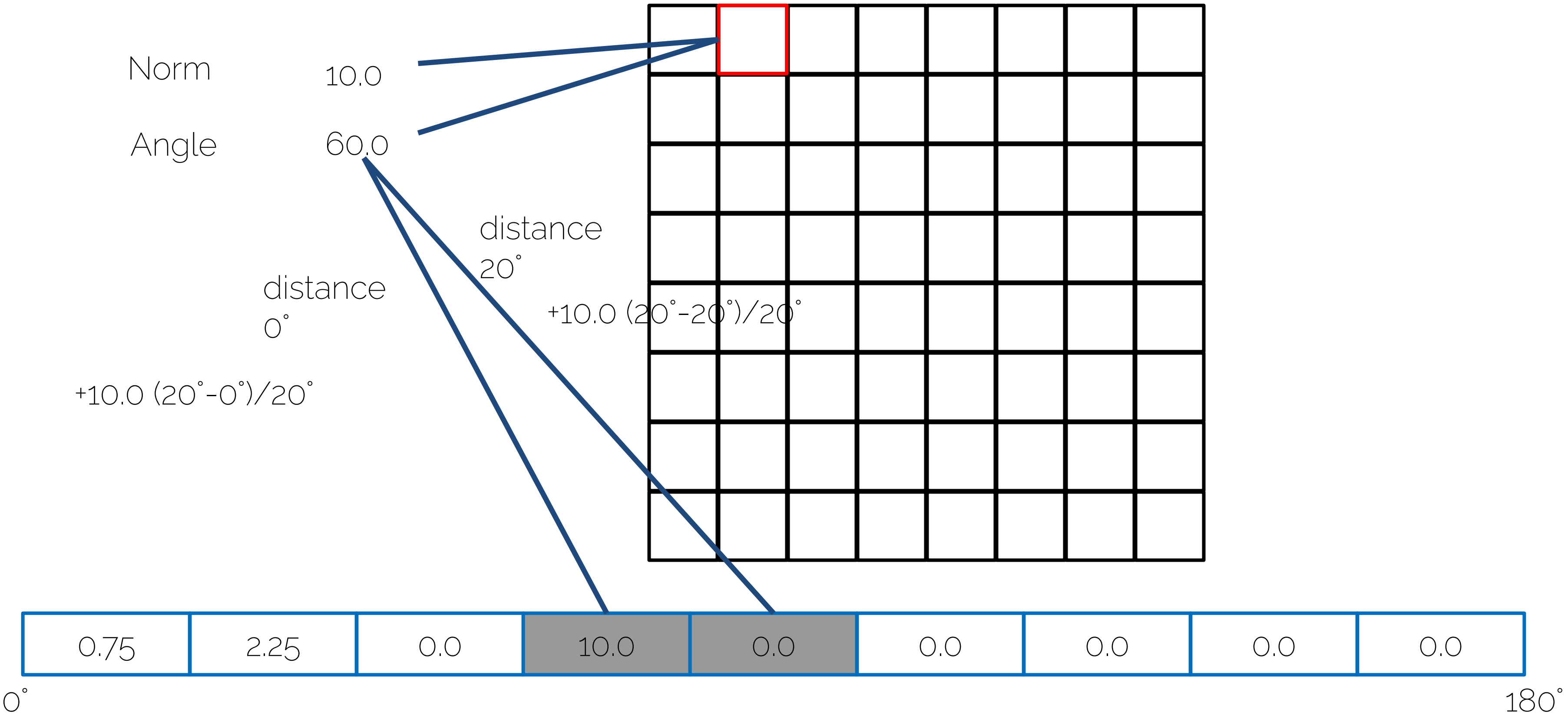
Task 3 - Filling the HOG bins



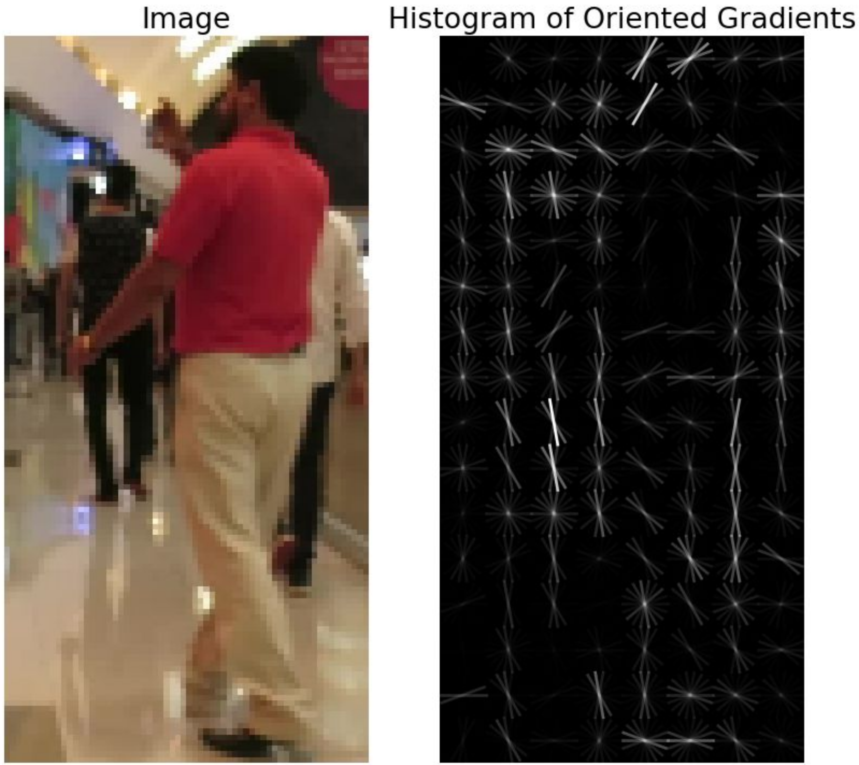
Task 3 - Filling the HOG bins



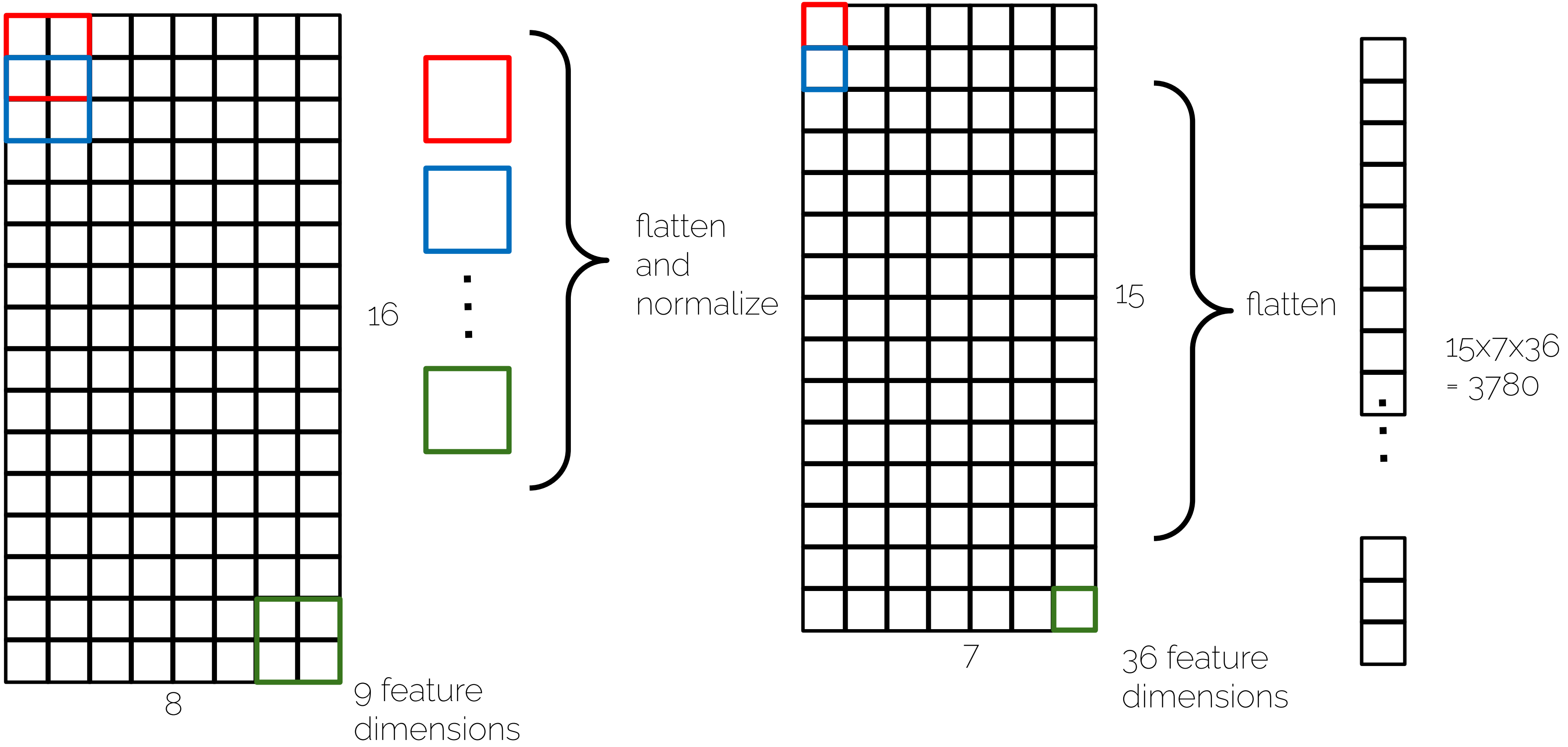
Task 3 - Filling the HOG bins



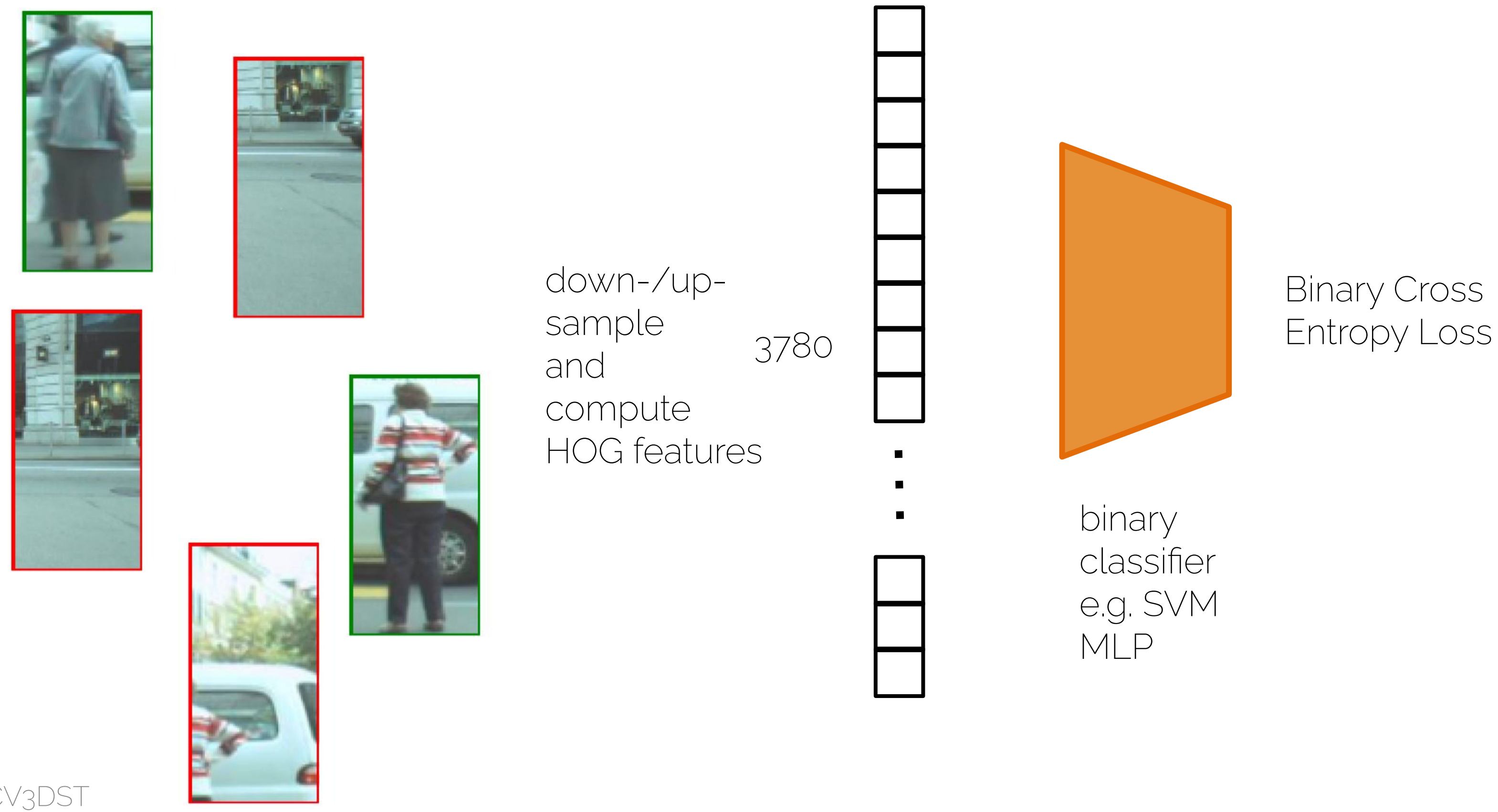
Task 3 - Filling the HOG bins



HOG Features - Additional Steps



HOG Features - Additional Steps

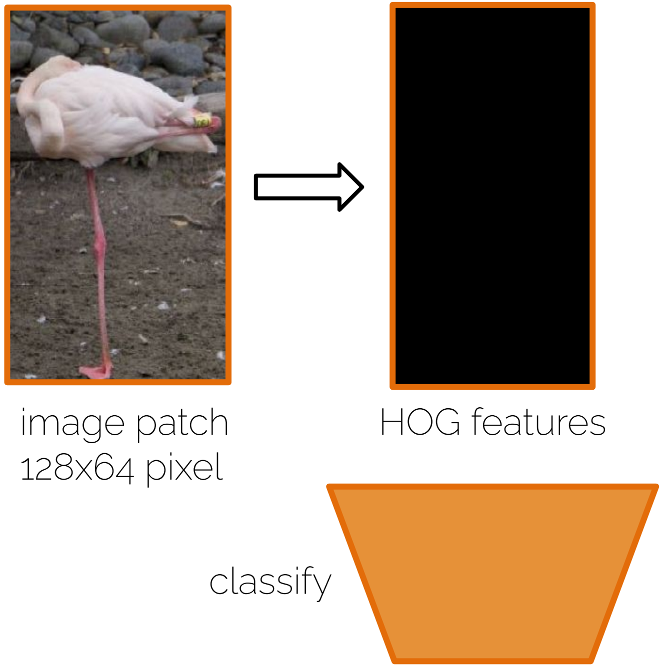


Task 4 - Sliding Window Detection

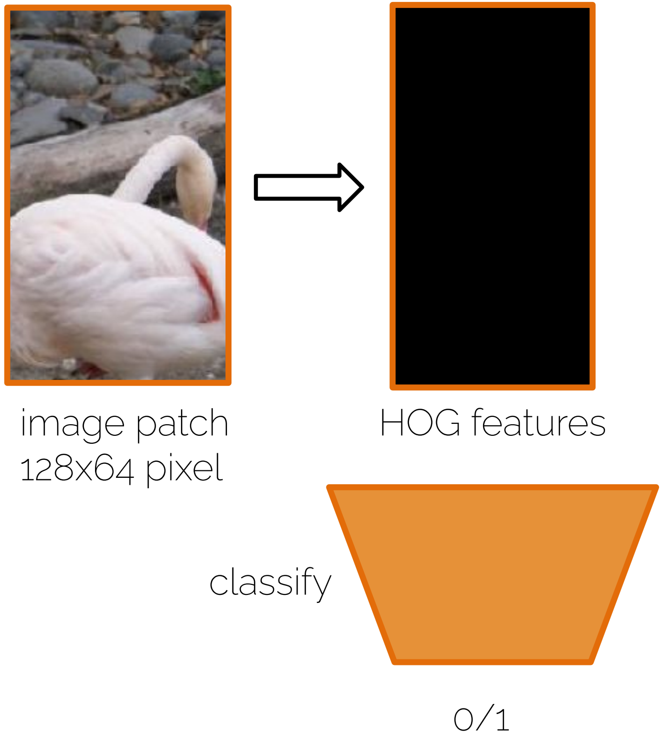
- naive approach to object detection
- exhaustive search across all of the image
- fixed detection ratio
- need for image pyramid to capture different scales

- implementation for different strides
  - less detections
  - speed up detection time

Task 4 - Sliding Window Detection



Task 4 - Sliding Window Detection



Task 5 - NMS

- algorithm to select the “best” prediction from a bunch of predictions
- it is a post-processing step, not a machine learning component
- a deterministic process that does not require any learning
- it is used in both classical approaches and deep learning methods

Task 5 - NMS

Task 5 - NMS

Algorithm 1 Non-Max Suppression

procedure NMS(B, c)
    $B_{nms} \leftarrow \emptyset$
    for $b_i \in B$ **do**
        $discard \leftarrow$ False
        for $b_j \in B$ **do**
            if $\text{same}(b_i, b_j) > \lambda_{nms}$ **then**
                if $\text{score}(c, b_j) > \text{score}(c, b_i)$ **then**
                    $discard \leftarrow$ True
        if not $discard$ **then**
            $B_{nms} \leftarrow B_{nms} \cup b_i$
    return B_{nms}

possible vectorization

B: Bounding Boxes
c: Classification Probability
$\text{same}(b_i, b_j)$: $\text{IoU}(b_i, b_j)$
$\text{score}(c, b_j)$: $c\,[b_i]$

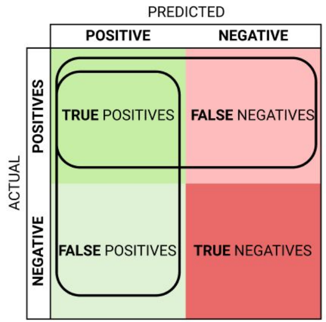
How does precision and recall change with the choice of λ?

Task 5 - NMS

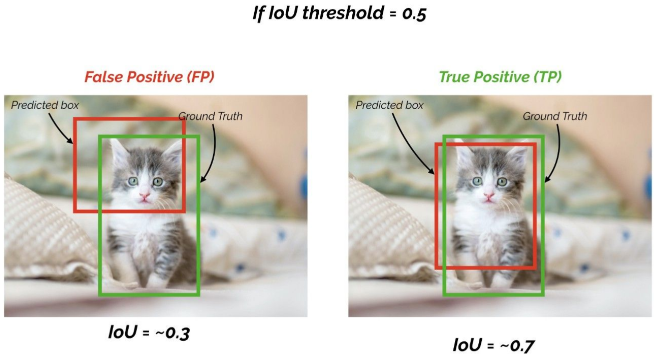
- high λ:
  - less bounding boxes that overlap enough
  - more predicted bounding boxes
  - more false positives/less false negatives
  - lower precision/higher recall
- low λ:
  - more bounding boxes that overlap enough
  - less predicted bounding boxes
  - less false positives/more false negatives
  - higher precision/lower recall

Task 6 - Evaluation (optional)

- Confusion Matrix
  - TP: model predicted object, actually object
  - FP: model predicted object, no object
  - FN: model did not predict, actually object



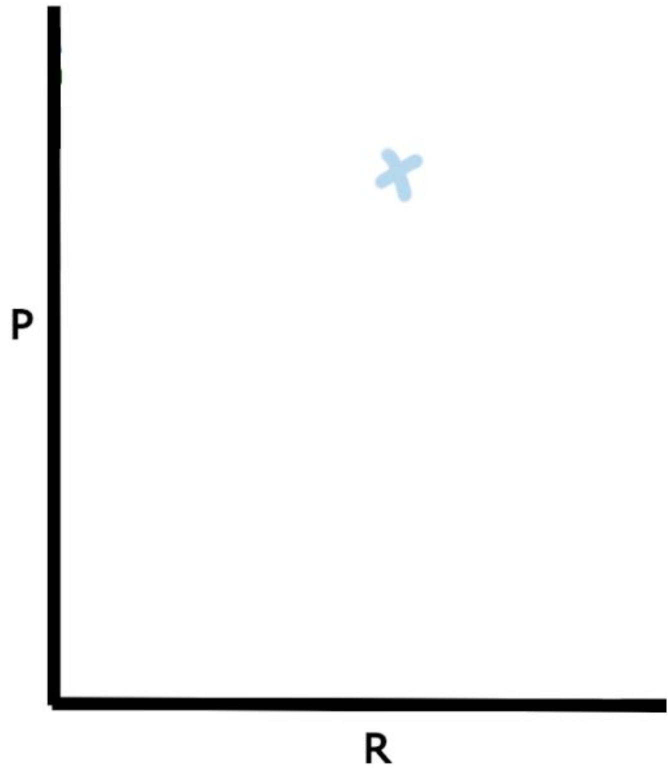
- Getting TP, FP, FN:
  - Set of positive predictions P_pred
  - Set of positive annotations P



Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: How can we build a Precision-Recall-Curve?

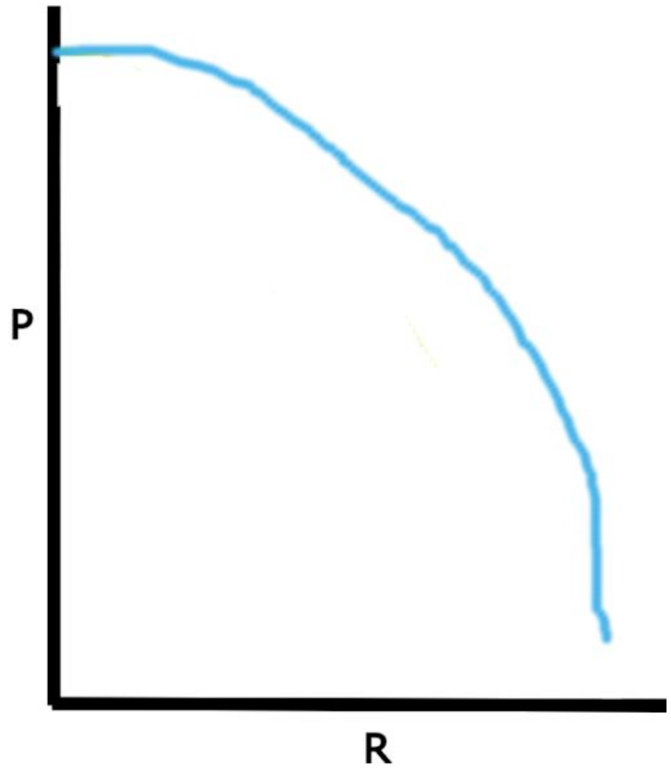


Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: How can we build a Precision-Recall-Curve?

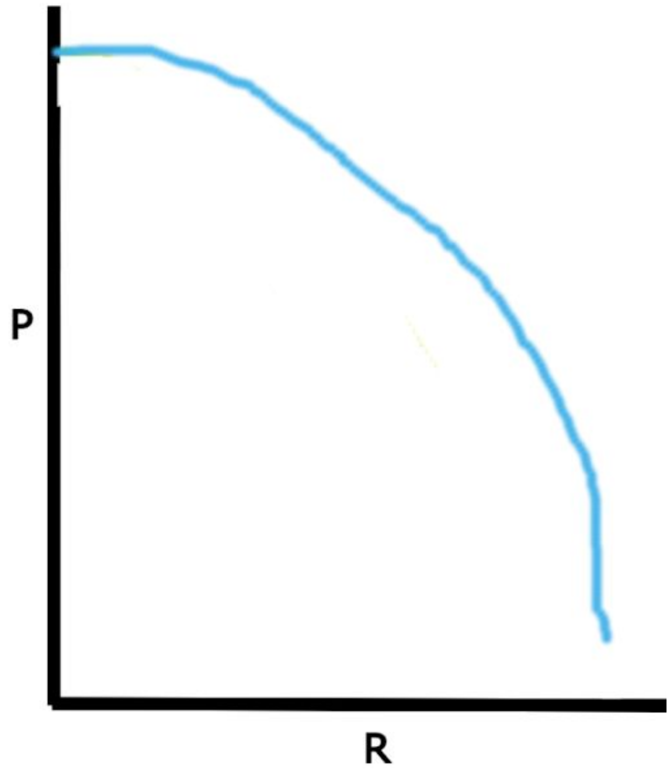
-> Vary confidence threshold



Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: What is the Area under Curve and Average Precision?



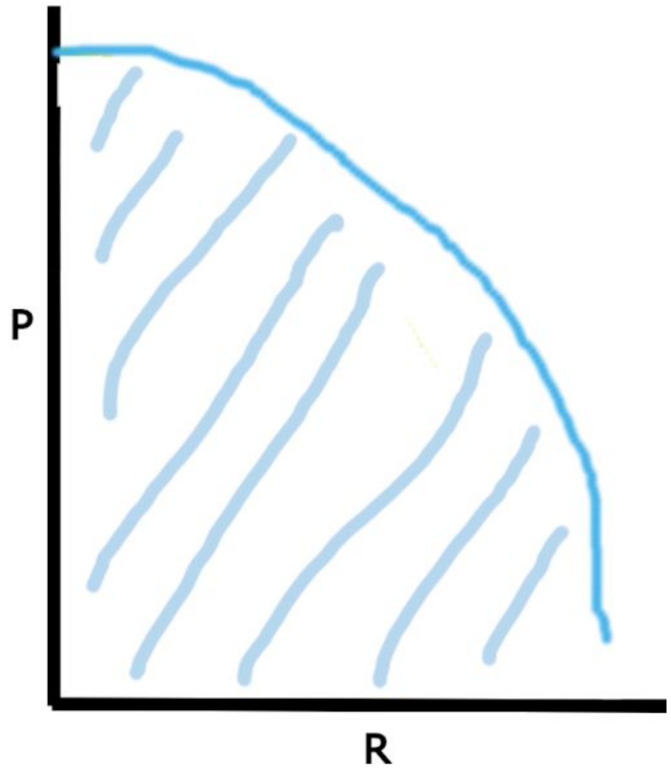
Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: What is the Area under Curve and Average Precision?

$$\int_0^1 P(R)dR$$

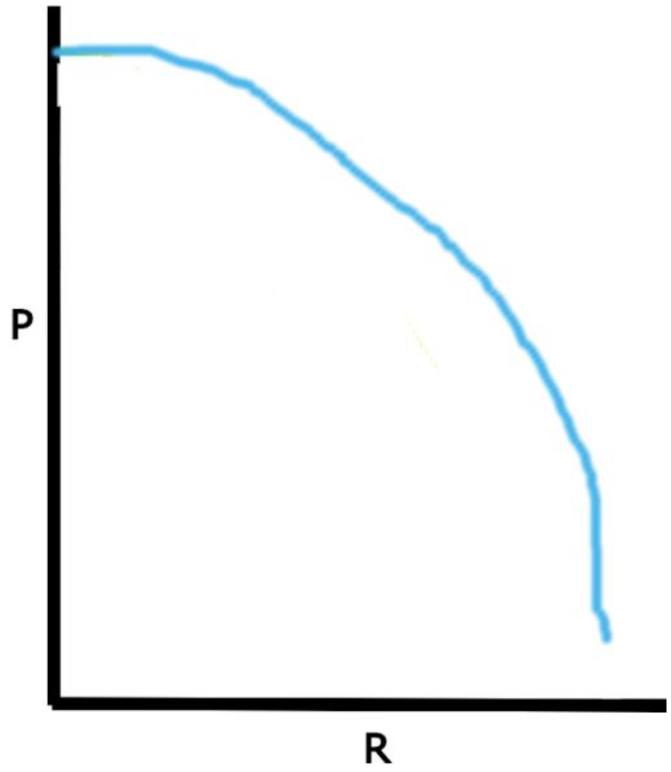
$$\sum_k P(R_k)\Delta R$$



Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: How does the IoU influence the curve?

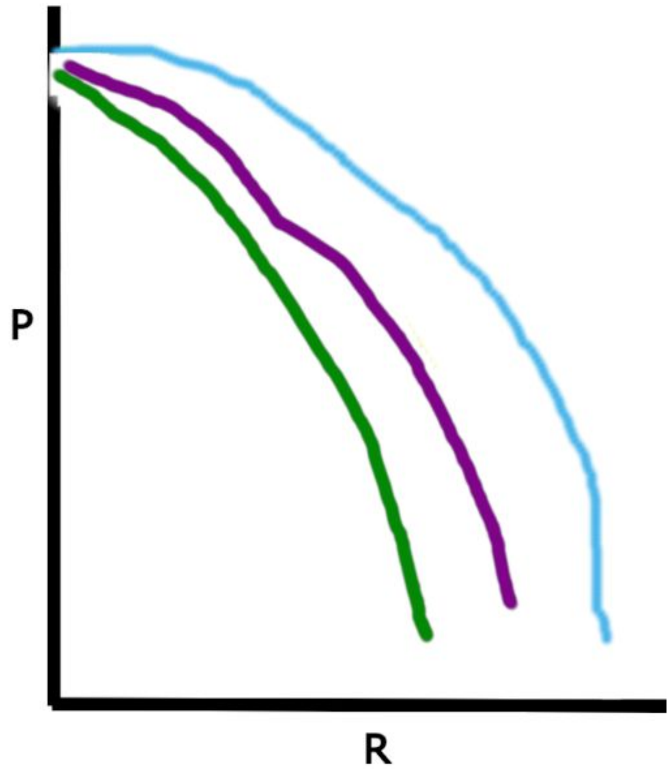


Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: How does the IoU influence the curve?

Task: Attach IoU_thresholds 0.5, 0.6, 0.7 to the curves.



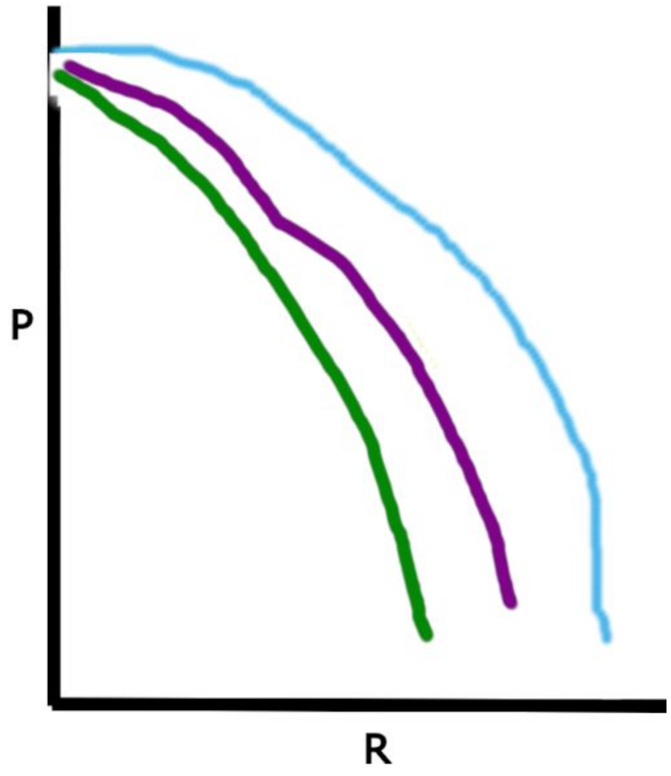
Task 6 - Evaluation (optional)

- Precision = TP/ P_pred
- Recall = TP/P

Question: How does the IoU influence the curve?

Task: Attach IoU_thresholds 0.5, 0.6, 0.7 to the curves.

Higher IoU thresholds result in (more/less) TP and therefore (higher/lower) recall.



General Tips

- read through the notebooks carefully
  Yes. Even the introductory parts. They might contain hints on how to solve the problems
- look for pytorch implementations before you try to do everything by your own
- try to vectorize your operations. Vectorizations speed up your computation time significantly

Links

- Test server: https://cv3dst.cvai.cit.tum.de/login
- If you have trouble registering https://forms.gle/yZkZiDiyHxWuNqQG7
- Data for Exercise 01: https://vision.in.tum.de/webshare/g/cv3dst/exercise_01.zip

Links for the individual datasets

- MOT
  https://vision.in.tum.de/webshare/g/cv3dst/datasets/MOT16.zip
- market
  https://vision.in.tum.de/webshare/g/cv3dst/datasets/market.zip
- obj_seg
  https://vision.in.tum.de/webshare/g/cv3dst/datasets/obj_seg.zip
- reid_gnn
  https://vision.in.tum.de/webshare/g/cv3dst/datasets/reid_gnn.zip