



max planck institut
informatik

High-Level Computer Vision SS 2024

Tutorial 1

Sukrut Rao, Haoran Wang, Amin Parchami

Max Planck Institute for Informatics, Saarland Informatics Campus

April 29, 2024



Outline

- Course Organization
- Assignment 1 Discussion
- Python Tutorial



Outline

- **Course Organization**
- Assignment 1 Discussion
- Python Tutorial

TAs



Sukrut Rao



Haoran Wang



Amin Parchami

Contacting TAs

Online:

- For questions of general interest (e.g. about assignments): Forum
 - <https://cms.sic.saarland/hlcvss24/forum/>
 - Ask in the appropriate topic forum
- For other questions: E-mail
 - hlcv-ss24@mpi-inf.mpg.de
 - Visible to all TAs
 - **Please do not email us individually!**

In-Person:


- During Tutorials: Mondays 10:15 AM - 12:00 PM, **E1.5 002**
- In Office Hours: Wednesdays 9:00 AM - 10:00 AM, **E1.4 024**

Tutorials

- On Mondays 10:15 AM - 12:00 PM, at **E1.5 002**
- For:
 - Discussing assignment problems
 - Introductions to tools useful for the course (e.g. PyTorch)
 - Answering questions
 - Help with assignments
- Agenda for each tutorial will be announced in the CMS
- Not held every week (will be announced in the CMS each week)

Course Page: CMS

[Courses](#) [Main Page](#) [Information](#) [Registration](#) [Personal Status](#) [Login](#)



High-Level Computer Vision

[Bernt Schiele](#)
Advanced Lecture (6 CP), Summer Semester 2024







Registration for this course is open until **Monday, 06.05.2024 23:59**.

- Register at <https://cms.sic.saarland/hlcvss24> by **May 6, 2024**
- For:
 - Announcements
 - Course Materials (e.g. slides, project instructions, old lecture videos)
 - Some submissions
 - User-specific information (e.g. access tokens for servers, exam time slots)

For Questions and Discussion: Forum

[Courses](#)[Main Page](#)[Information ▾](#)[Registration](#)[Personal Status](#)[Forum](#)

- Ask questions in the appropriate forum
- Separate subforum for each assignment

FORUM	TOPICS	POSTS
 High-Level Computer Vision: General	1	2
 Finding teammates	2	2
 Questions about Lectures	0	0
 Questions about Assignments Subforum: Assignment 1	0	0
 Questions about Project	0	0
 Off-Topic	1	2

- If you want notifications (optional):








[Board index](#) ☒ [Subscribe forum](#)[Contact us](#) [The team](#) [Members](#) [Delete cookies](#) All times are UTC+02:00 [≡](#)

Overview of Evaluation

- Four assignments (25%):
 - First: basic feature extraction, object identification, and image classification
- Project (25%)
- Exam (50%): need to pass separately

Assignments

- Usually published on Mondays
- Deadline usually in 2 weeks
- To be submitted in teams of three
- Can use the forum to find teammates:

FORUM		TOPICS	POSTS
 High-Level Computer Vision: General		1	2
 Finding teammates		2	2
 Questions about Lectures		0	0
 Questions about Assignments Subforum:  Assignment 1		0	0
 Questions about Project		0	0
 Off-Topic		1	2

Registering teams in CMS

Deadline: May 6, 2024

News
Teams
Anonymous Comment
Settings
Unregister

Teams

Assignments and Project *Not in a team*
No Team You can change teams until 24.04.2023 23:59.

Create team **Join team**

1. Creating a team:

Teams

Assignments and Project You're in **Team #8** together with Sukrut Sridhar Rao (1 members).
Team #8 Your team has 2 free slots. Invite code: **PHhGhTVvwpr0AnR**
You can change teams until 24.04.2023 23:59.

Leave team

2. Joining a team:

Join a team

Invite Code ?

Join Team **Close**

Projects

- Structure:
 - Proposal
 - Interim report
 - Final presentation: 15-min talk
 - Final report: 4-5 pages
- Time: ~1-1.5 months
- After the assignments
- Topic can be as per your choice

Sample Projects

- Age Recognition
- Other recognition tasks....
- Image retrieval for 3D objects
- Object retrieval in videos, on a mobile phone
- Person re-identification
- Image and video segmentation
- Detection and segmentation
- Domain Adaptation
- Explainability and Interpretability
- Image generation tasks (GANs, conditional generation, style transfer)

Computing Resources (GPUs)

- Access to university GPU servers will be provided
 - One account per team
 - Likely starting from Assignment 3

Additional options:

- Google Colab (<https://colab.research.google.com/>)
- Your own resources (e.g. personal laptop)



Outline

- Course Organization
- **Assignment 1 Discussion**
- Python Tutorial

Assignment 1

Goal: getting familiar with Python & doing small and easy image processing tasks

Part 1

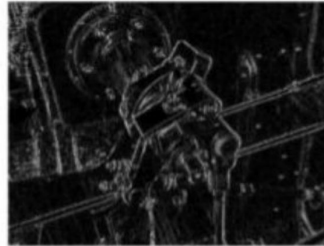
- Q1: Image filtering
- Q2: Image representations and histogram distances
- Q3: Object identification
- Q4: Performance evaluation

Part 2

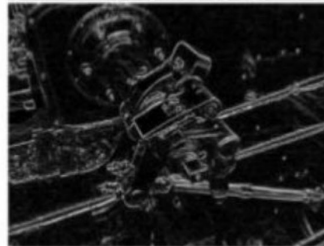
- Image classification with SVM

Q1: Image filtering

Goal: implementing an edge detector using the derivative of the Gaussian filter

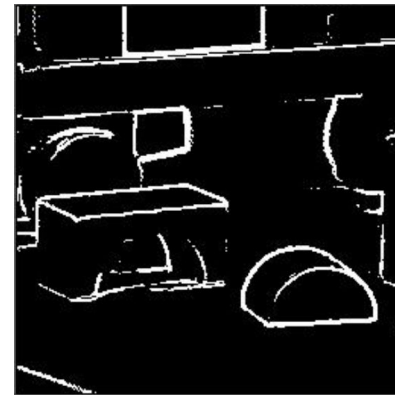
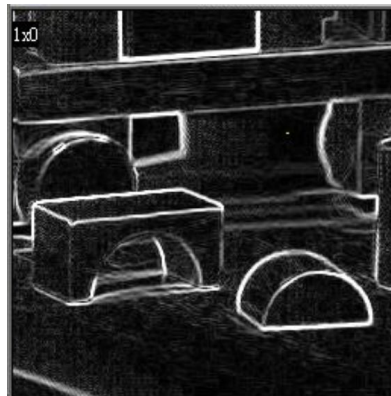
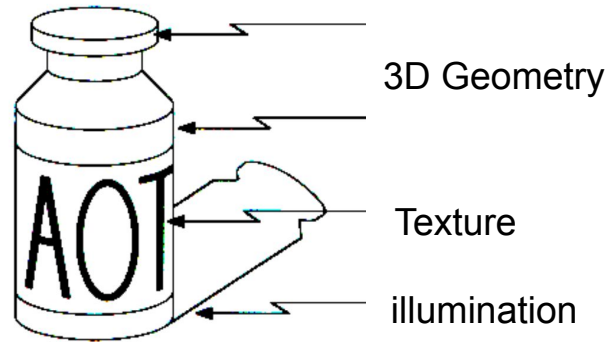


Edges along the x axis



Edges along the y axis

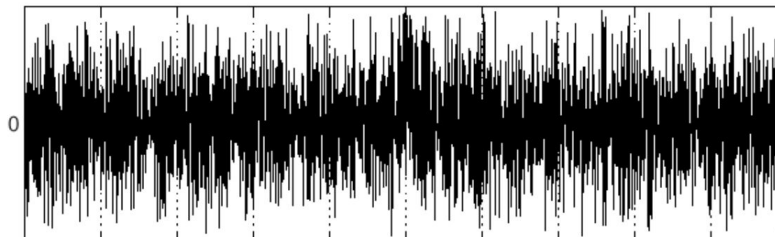
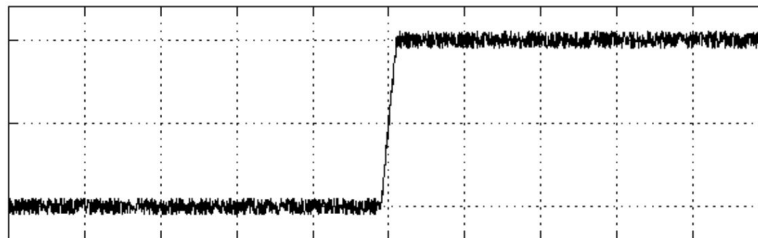
Q1-Detour: Why Edges?



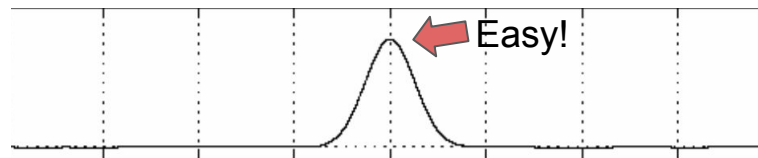
Seem to cover *interesting* locations

Images taken from [Udacity](#) slides.

Q1-Detour: One Problem: Noise!



Where's the edge !?



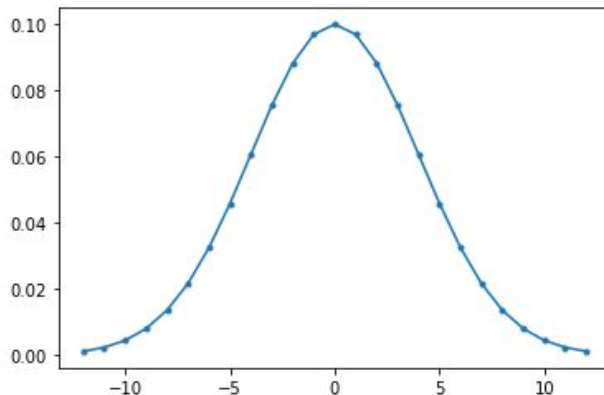
Wish it was like this!

Q1.1: 1D Gaussian distribution

Q1.1 Implement a method that computes the values of a 1-D Gaussian for a given variance σ^2 . The method should also give a vector of values on which the Gaussian filter is defined: integer values on the interval $[-3\sigma, 3\sigma]$.

$$G = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right). \quad \text{mean} = 0$$

```
def gauss(sigma):  
    x = np.arange(math.ceil(-3*sigma), math.floor(3*sigma) + 1)  
    Gx = 1/(sigma*math.sqrt(2*math.pi)) * np.exp(-np.square(x)/(2*sigma**2))  
    return Gx, x
```



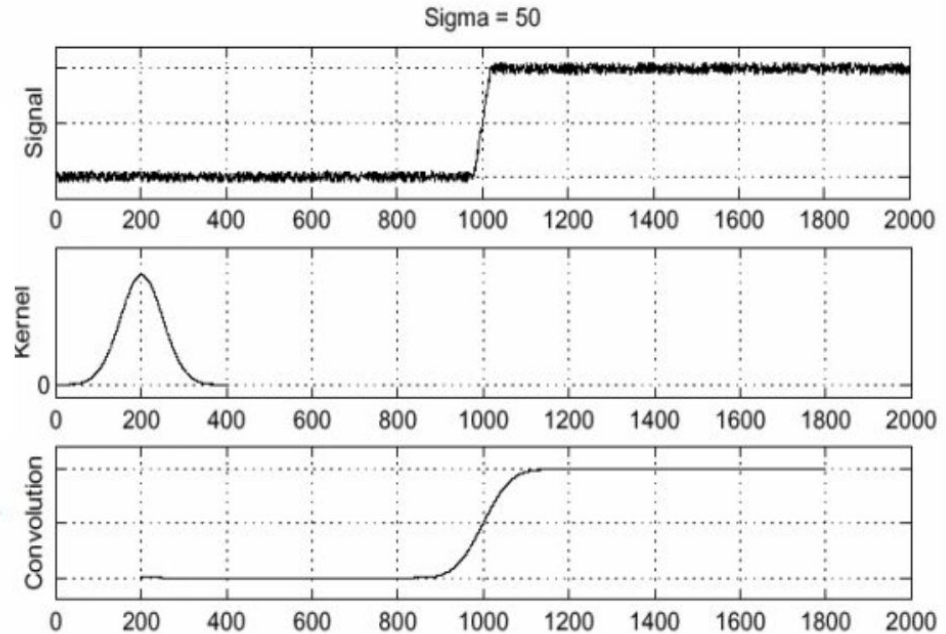
Q1.2: 2D Gaussian filtering

Convolution operation (1D)

Convolution: $G = H * F$

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v] F[i - u, j - v]$$

Kernel



Q1.2: 2D Gaussian filtering

Separability of the Gaussian filter

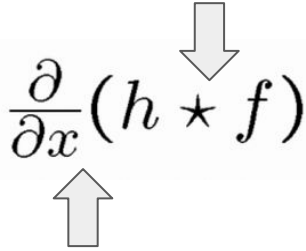
$$\begin{aligned} G_{\sigma}(x, y) &= \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2 + y^2}{2\sigma^2}} \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}} \right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}} \right) \end{aligned}$$

Q1.2: 2D Gaussian filtering

```
def gaussianfilter(img, sigma):  
    # Kernel should have the same number of dimensions as the image  
    kernel = np.expand_dims(gauss(sigma)[0], 0)  
    # Horizontal gaussian convolution  
    outimage = signal.convolve2d(img, kernel, boundary='symm', mode='same')  
    # Vertical gaussian convolution  
    outimage = signal.convolve2d(outimage, kernel.T, boundary='symm', mode='same')  
    return outimage
```

So we've got a pipeline!

What we just implemented!



The diagram shows the mathematical expression $\frac{\partial}{\partial x}(h \star f)$ centered on a light gray background. A downward-pointing arrow is positioned above the expression, pointing towards the convolution symbol \star . An upward-pointing arrow is positioned below the expression, pointing towards the derivative operator $\frac{\partial}{\partial x}$.

$$\frac{\partial}{\partial x}(h \star f)$$

Also convolution by a filter.



The filter is represented as a 1x3 grid of black squares with white text. The values are -1/2, 0, and +1/2.

-1/2	0	+1/2
------	---	------

So we've got a pipeline!

What we just implemented!

$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x} h) \star f$$

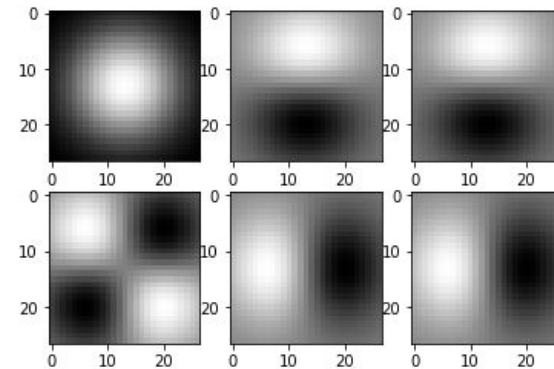
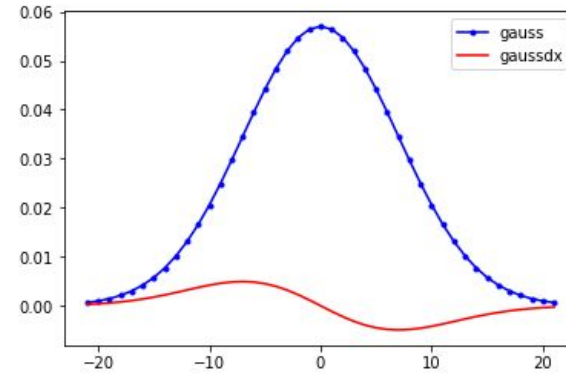
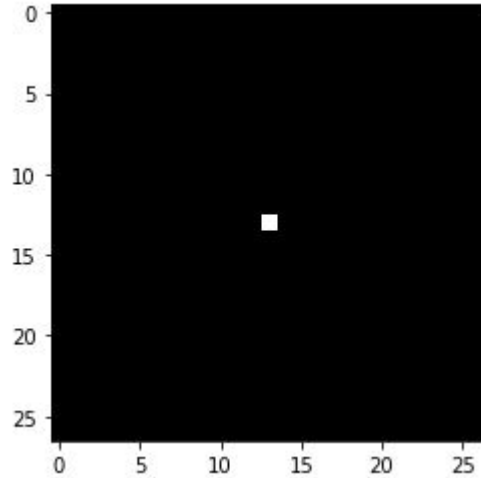
Also convolution by a filter.

-1/2	0	+1/2
------	---	------

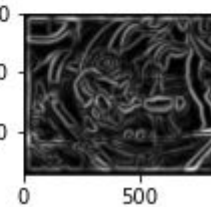
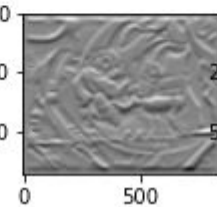
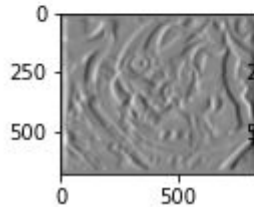
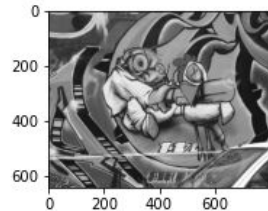
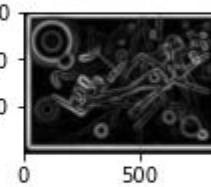
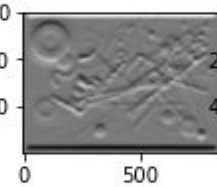
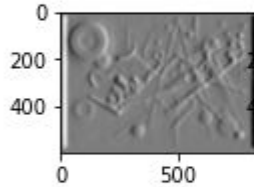
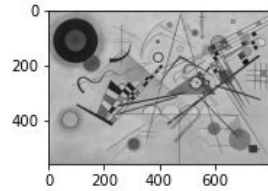
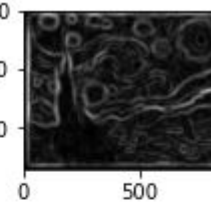
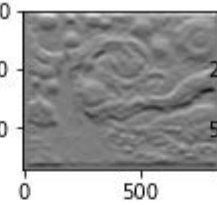
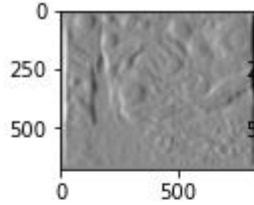
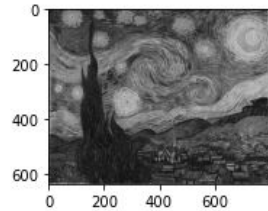
Due to **associativity of convolution**, we can take derivative of the Gaussian kernel and apply convolution on image (time consuming) only once with the final kernel.

Hence we need, ***derivative of the Gaussian***.

Q1.3: 1D Gaussian derivative



Q1.4: 2D Gaussian derivative



Q2: Image representations and histogram distances

- Normalized grayscale histogram
 - Convert image into grey scale. Normalize the image into range $[0,1]$. Compute histogram
- RGB color histogram.
 - 3D joint histogram of RGB values
- RG chromaticity space
 - 2D Joint histogram of rg v
- Histogram of gradients dx and dy
 - Compute the gradient of the image along x and y directions.
 - Compute the histogram of the gradients and concatenate into one long vector.

RGB histogram

- Compute 3D histogram for RGB color combinations.
- For input n bin, discretize each channel into these bin indexes.
- Map bin indexes into n^3 number of bins. Illustration of the process in 2D

		red			
		0-63	64-127	128-191	192-255
blue	0-63	43	78	18	0
	64-127	45	67	33	2
	128-191	127	58	25	8
	192-255	140	47	47	13

RG histogram- 2D histogram on chromaticity image

$$r = \frac{R}{R + G + B}$$

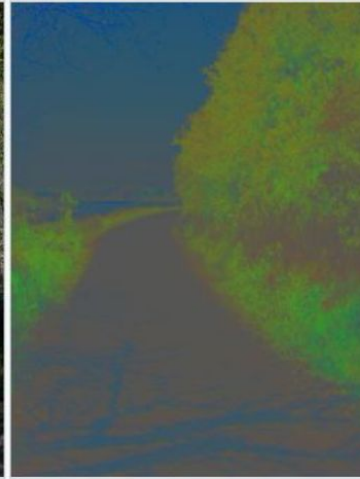
$$g = \frac{G}{R + G + B}$$

$$b = \frac{B}{R + G + B}$$

$$r + g + b = 1$$



RGB image

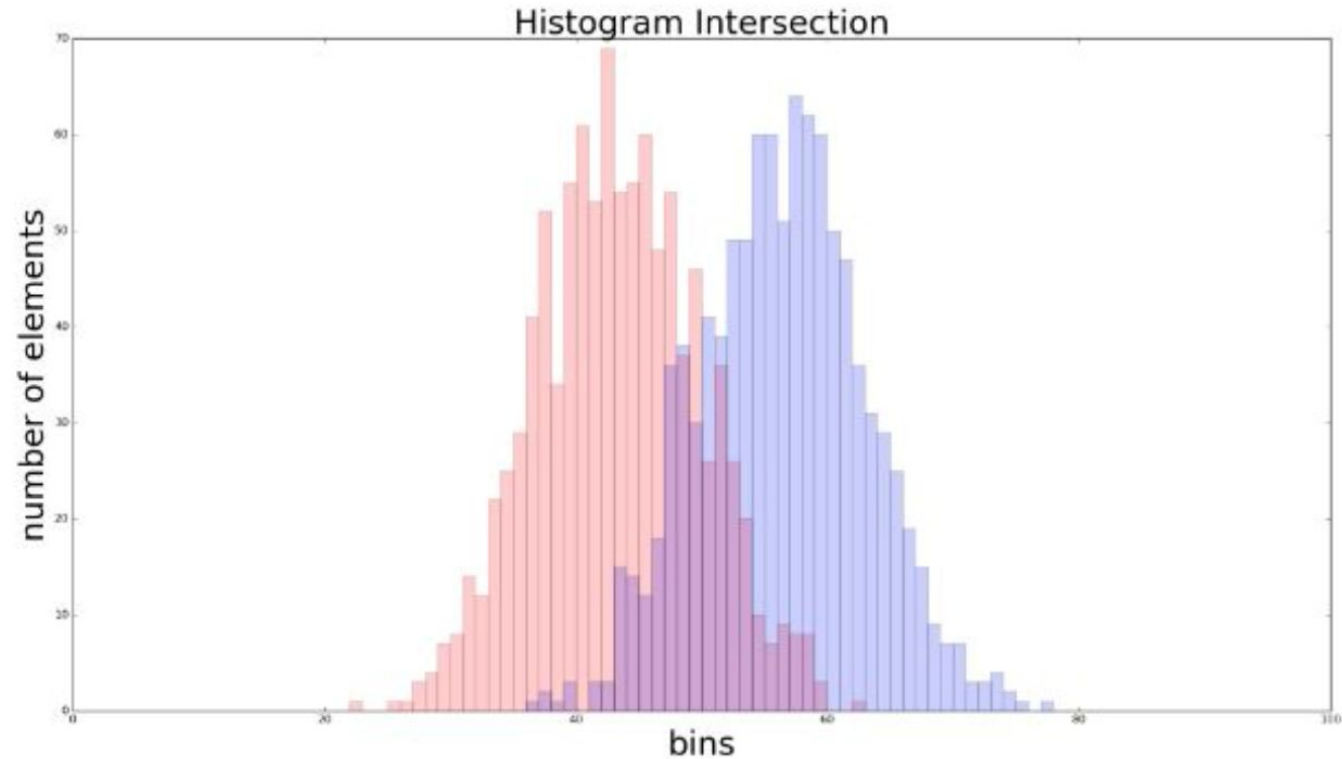


rg - chromaticity



greyscale image

Histogram distances



Histogram distances

- City-block or L1 distance

$$D_1(A, B) = \sum_{i=0}^{b-1} |H_i(A) - H_i(B)|.$$

- Intersection distance

$$D_3(A, B) = n - \sum_{i=0}^{b-1} \min(H_i(A), H_i(B)).$$

- Euclidean or L2 distance

$$D_2(A, B) = \sqrt{\sum_{i=0}^{b-1} (H_i(A) - H_i(B))^2}.$$

- Chi -squared distance

$$D_4(A, B) = \sum_{i=0}^{b-1} \frac{(H_i(A) - H_i(B))^2}{H_i(A) + H_i(B)}$$

Intersection distance

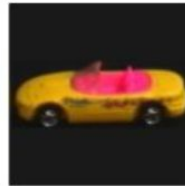
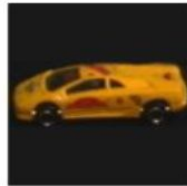
```
def dist_intersect(x,y):  
    # your code here  
    Dist = 0  
    for i in range(len(x)):  
        Dist = min(x[i], y[i]) + Dist  
    Dist = 1 - Dist  
    return Dist
```

Q3: Object identification

- Using this space (in this case the histogram space), one can perform several recognition tasks - e.g. identification.
- In this question you will use various histogram features and metrics from to find the best matching image from the database for a given query image.



(a)



(b)



Q4: Performance evaluation

How to compare models?

1. Compare a single number
 - a. E.g. Accuracy, top-k Accuracy
 - b. **Easy** to Compare, but **might be misleading** in some cases (e.g ?)

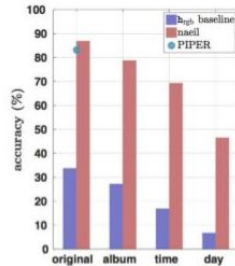


Figure 4. Recognition accuracy across different experimental setups on the test data.

Q4: Performance evaluation

How to compare models?

1. Compare a single number
 - a. E.g. Accuracy, top-k Accuracy
 - b. **Easy** to Compare, but **might be misleading** in some cases (e.g imbalanced data)
2. Compare Curves
 - a. Precision-Recall curve, ROC curve

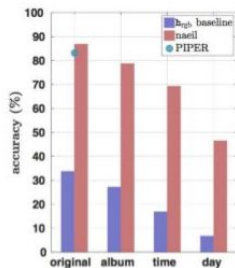
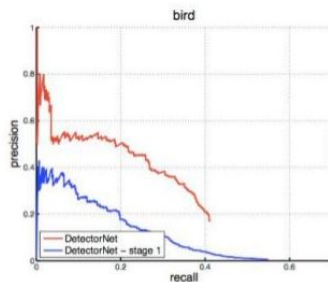


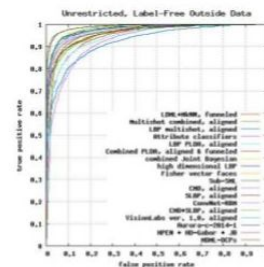
Figure 4. Recognition accuracy across different experimental setups on the test data.

Accuracy (Oh, ICCV'15)



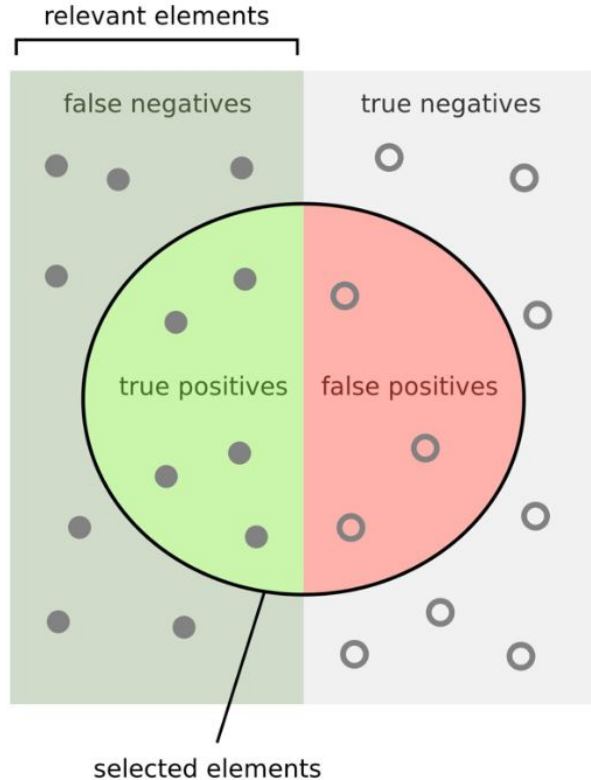
Precision-Recall (Szegedy,

NIPS'13)



ROC (LFW Face verification)

Precision and recall



How many selected
items are relevant?

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

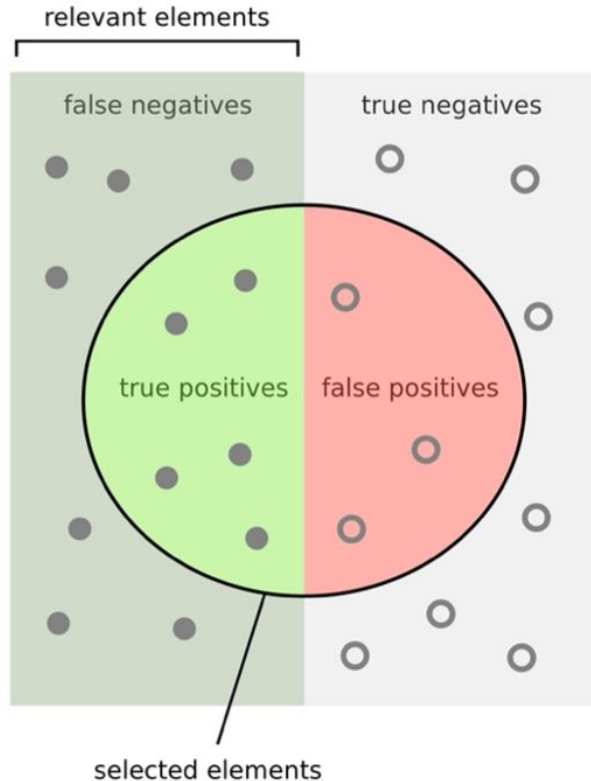
How many relevant
items are selected?

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Accuracy

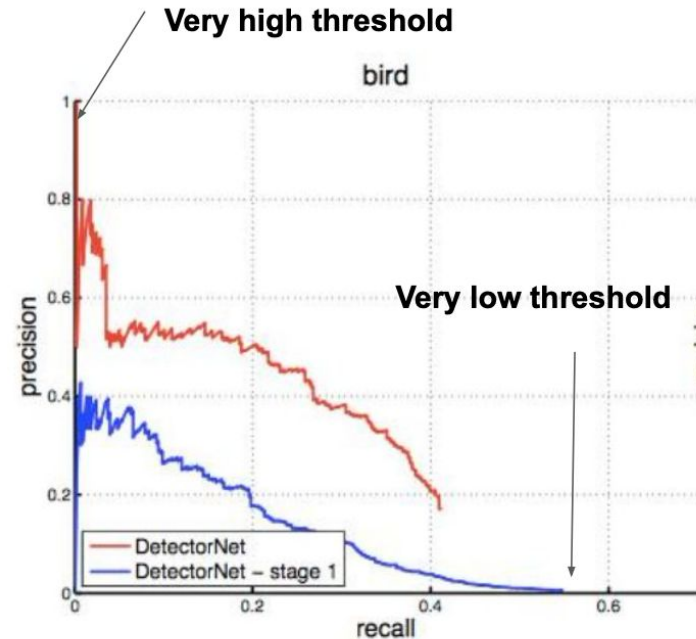


$$\text{Accuracy} = \frac{\text{\#Correct Predictions}}{\text{\#Total Examples}}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision-Recall curve

- Usually model output is continuous value. Binary decision is made by thresholding this value
 $P(\text{match}|x,y) > \text{threshold}$
- Different thresholds lead to different operating points.
- Precision recall curve allows us to compare models independent of the threshold



One could also try to sum up the plot by computing the Area Under the Curve (AUC), named **Average Precision** for the PR curve.

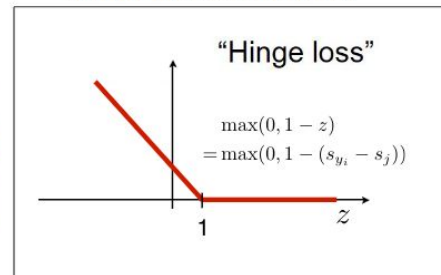
Part 2 - Image Classification

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	3.2	1.3	2.2
car	5.1	4.9	2.5
frog	-1.7	2.0	-3.1

Multiclass SVM loss:



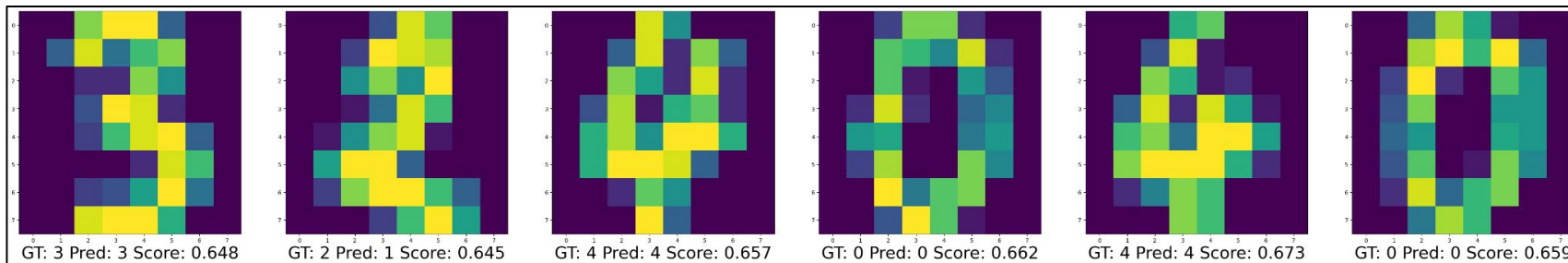
$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Part 2 - Image Classification

Idea #3: Split data into **train**, **val**, and **test**; choose hyperparameters on val and evaluate on test

Better!





Submission

- Deadline: 13.05.2024, 23:59 CEST
- Please make sure your notebooks runs out-of-the-box.



Outline

- Course Organization
- Assignment 1 Discussion
- **Python Tutorial**



Install Conda

Installing on Linux

1. Download the installer:

- [Miniconda installer for Linux.](#)
- [Anaconda installer for Linux.](#)

2. Verify your installer hashes.

3. In your terminal window, run:

- Miniconda:

```
bash Miniconda3-latest-Linux-x86_64.sh
```

- Anaconda:

```
bash Anaconda3-latest-Linux-x86_64.sh
```

4. Follow the prompts on the installer screens.

If you are unsure about any setting, accept the defaults. You can change them later.

5. To make the changes take effect, close and then re-open your terminal window.

6. Test your installation. In your terminal window or Anaconda Prompt, run the command

`conda list`. A list of installed packages appears if it has been installed correctly.

Installing on Windows

1. Download the installer:

- [Miniconda installer for Windows.](#)
- [Anaconda installer for Windows.](#)

2. Verify your installer hashes.

3. Double-click the `.exe` file.

4. Follow the instructions on the screen.

If you are unsure about any setting, accept the defaults. You can change them later.

When installation is finished, from the **Start** menu, open the Anaconda Prompt.

5. Test your installation. In your terminal window or Anaconda Prompt, run the command

`conda list`. A list of installed packages appears if it has been installed correctly.

[here](#)

Installing on macOS

1. Download the installer:

- [Miniconda installer for macOS.](#)
- [Anaconda installer for macOS.](#)

2. Verify your installer hashes.

3. Install:

- Miniconda---In your terminal window, run:

```
bash Miniconda3-latest-MacOSX-x86_64.sh
```


- Anaconda---Double-click the `.pkg` file.

4. Follow the prompts on the installer screens.

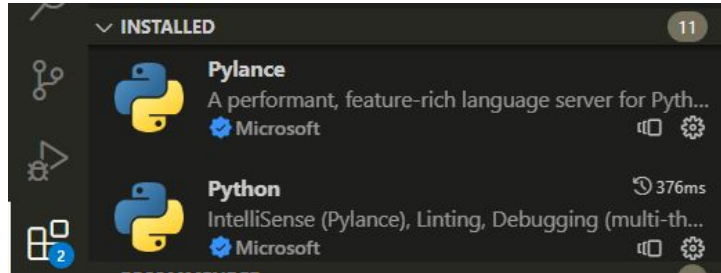
[here](#)

[here](#)

First Step: Set up the environment

- Create your environment
 - `conda create --name MyEnvName`
- View the available environments
 - `conda info --envs`
- Activate your environment
 - `conda activate MyEnvName`
- Install Required Packages
 - `conda install jupyter numpy pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c nvidia`
Only if you have GPU
 - `conda install matplotlib -c conda-forge`
 - Sometimes you might have to install some packages with pip instead
 - `pip install ...`
- Now you need to select this created environment for your Python interpreter.

Install these two extensions if using VSCode



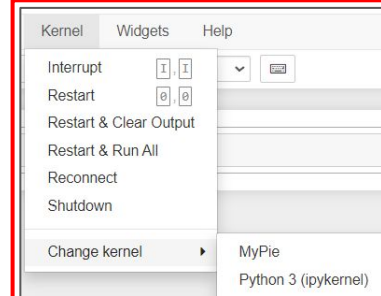
Hit Ctrl/CMD + Shift + P and search for “Python: select interpreter”
Select the conda environment that you just created!

You need ipykernel (in your conda env) to run Notebooks.

With your conda environment activated, type in:

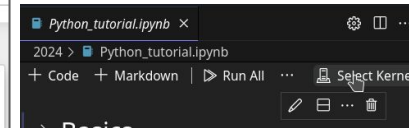
```
conda install ipykernel
```

```
python -m ipykernel install --user --name MyEnvName \  
--display-name "MyEnvName"
```



browser

Select the kernel
in the notebook.



VSCode



Python Tutorial