Introduction

IN 5400 — Machine Learning for Image Analysis

Anne Solberg, Alexander Binder, Andreas Kleppe, Are Jensen

13 Jan 2021

University of Oslo

Plan for today

- Introduction about the course
- Motivation: what deep learning can do
- Short background on image classification
- Short introduction to image convolution
- Information about exercises and jupyter/python

Introduction to the course

Lecturers

Alexander Binder
alexabin@ifi.uio.no

Andreas Kleppe
andrekle@ifi.uio.no

Are Charles Jensen
arej@ifi.uio.no

Overview

Lectures

- Machine learning
- Deep learning and image analysis
- Getting deep learning to work in practice
- Applications

Groups

- Jupyter notebooks, numpy and image analysis
- Exercises on image classification, image captioning, image segmentation
- Pytorch for more complex deep learning
- Students get time-limited access to GPU-servers at USIT for selected exercises (from March)

Schedule

Lectures

- Wednesday 14.15-16.00, 20th Jan, 27th Jan: zoom only, as of today from 3rd of Feb.: (Ole Johan Dahls hus) Aud Simula (Check room on course web page)

Mandatory exercises

- 2 Mandatory exercises, in PyTorch

Groups

- Wednesdays 12.15-14, (OJD) Datastue Modula Room Modula
- Group lecturer
  Gabriel Balaban gabrib@ifi.uio.no
  Lecturers and others will join in weeks with high workload
- Larger team for correcting mandatory exercises

Zoom rules

Questions:

preferred to raise hand or write in chat.

Course resources

- Course web page: https://www.uio.no/studier/emner/matnat/ifi/IN5400/v21/index.html
- Curriculum: lecture notes, weekly exercises, mandatory exercises
- No book closely followed, but links to relevant literature/notes given, see `http://d2l.ai` for a free online book and for the first 3 lectures as soft intro into the basic topics: `http://neuralnetworksanddeeplearning.com/`.
- Links to relevant online lectures given for each week
- Devilry for submitting mandatory exercises
- Digital exam, focus on understanding the topics

Who is this course for?

- Core focus: Master and PhD students working on image analysis problems using deep learning
- You should feel comfortable to code in python. Not recommended if one has no knowledge of python and is not a quick self-learner for python.
- Some knowledge in machine learning assumed.
- Will be useful for others with a good background in Python and linear algebra wanting to learn deep learning
- Demanding programming exercises: highly suggested to do all of them!
- Learning is about loss functions and derivation - you are expected to do calculations and compute derivatives by hand
- Expertize wanted for many companies - good job prospects, but your motivation must be high!

Prelimary schedule

- 13.1 Introduction and PyTorch
- 20.1 Linear Models, Recap on Gradient Based learning
- 27.1 Densely connected Feedforward nets
- 3.2 Convolutional nets (CNNs)
- 10.2 Chain rule, Back propagation, initialization
- 17.2 CNN Architectures
- 24.2 Optimizers beyond SGD, Guidelines for training, finetuning
- 3.3 Segmentation
- 10.3 Object Detection
- 17.3 Adversarial Attacks on deep learning and Defenses
- 24.3 Recurrent nets
- 31.3 no lecture (Easter holiday)
- 7.4 Explaining neural network decisions (gradient, guided backprop, LRP)
- 14.4 GANs I (classic, Wasserstein GANs)
- 21.4 GANs II, + bonus topic
- 5.5 attention models + bonus topic
- 12.5 Performance estimation, Cohort overfitting and Generalizability
- 19.5 Repetition
- 4.6 Exam

Web page

- https://www.uio.no/studier/emner/matnat/ifi/IN5400/v21/index.html
- Lectures with links to lecture foils, curriculum, weekly exercises, and mandatory exercises.
- Messages on the web page
- Urgent messages emailed to studenter.in5400@ifi.uio.no and studenter.in9400@ifi.uio.no. Links to your official uio-email - read this!

Mandatory exercises

- Required: use Python and PyTorch
- Exercise 1: implement training with finetuning for a medium size prediction problem (a few thousand images, not millions), available around March 1.
- Exercise 2: (tentative) Image captioning in PyTorch using a pretrained CNN and a Recurrent network
- PhD-students: Mandatory essay in addition to mandatory exercises
- Programming exercises are individual work, and the solution should be your own code.
- Plagiarism on mandatory exercises can be sometimes an issue at IFI, and the consequenses are severe.
- You are required to read the regulations at https://www.uio.no/english/studies/examinations/compulsory-activities/mn-ifi-mandatory.html

Groups first week

- First group Monday 20.1 12.15-14
- exercise in pytorch beyond mere matrix multiplication
- Download zip-file with exercises

Using Jupyter notebooks

- Install python3, pytorch, matplotlib on your own computer (Windows: anaconda, Unixes: many options)
- Second option: Use the linux computer in room Modula (/opt/ifi/anaconda3/bin/jupyter-notebook)
- Later in the course: access GPUs on servers

Curriculum

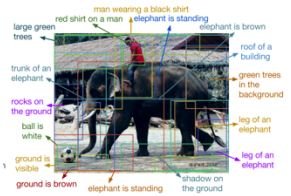
- Lecture notes, weekly exercises, and mandatory exercises define the curriculum
- The field is rapidly changing so that wont stick to a book
- Links to useful notes and online lectures will be given.

Required background

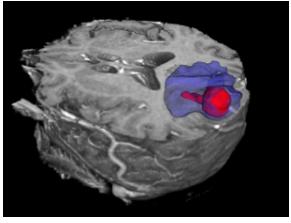
- Good knowledge in Python programming
- Good knowledge in mathematics (as a tool)
  - Linear algebra: vectors, matrix operations, dot products
  - Derivation of loss functions
  - probability: expectations, conditional probabilities of n variables
  - Simple optimization
- Experience in image analysis desired

Motivation

Many tasks require information from images



badminton



Object detection

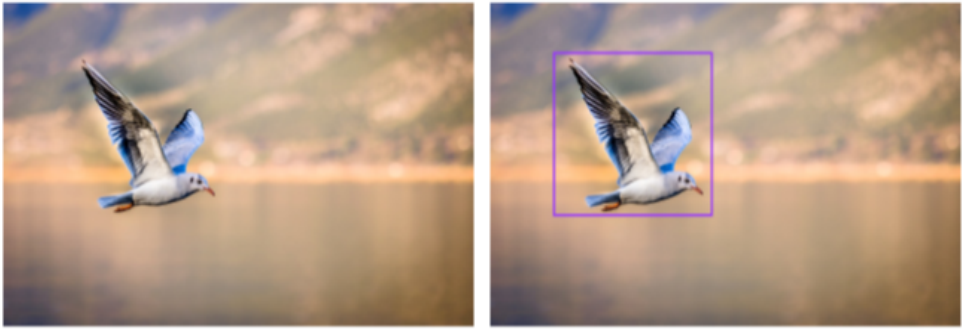
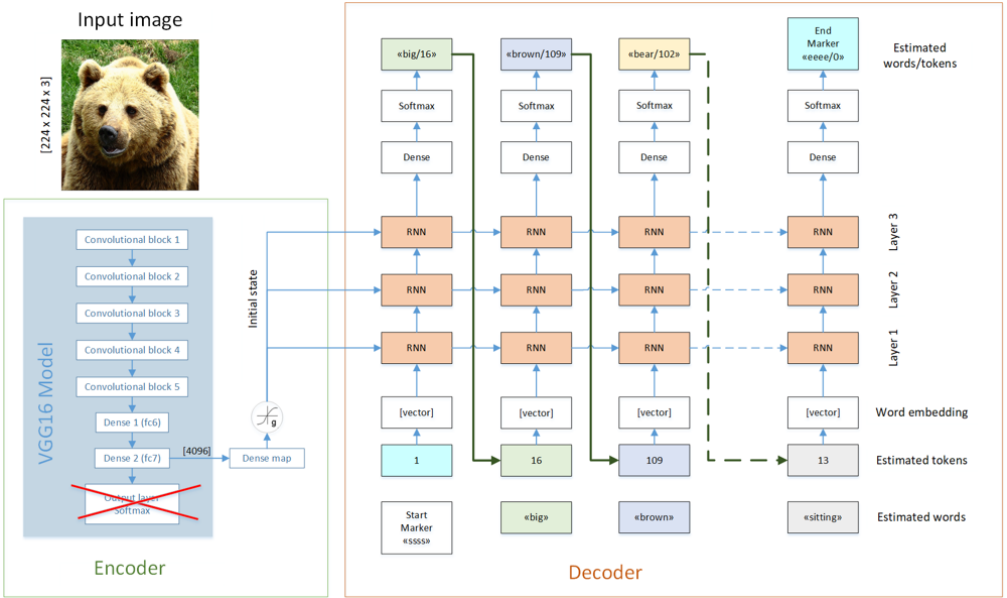


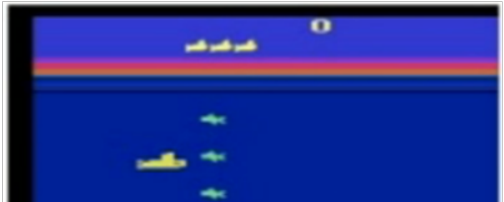
Figure 1: Seagull. Image source: https://www.pixabay.com

Instance segmentation

Combining recurrent and convolutional networks for image captioning



Reinforcement learning for games



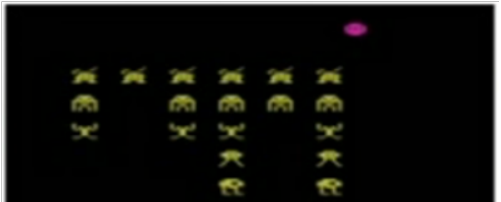


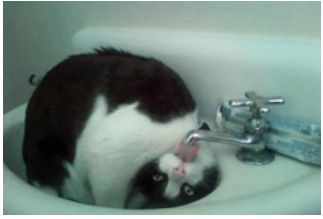
Image Generation

source: T. Karras et al. https://arxiv.org/pdf/1912.04958

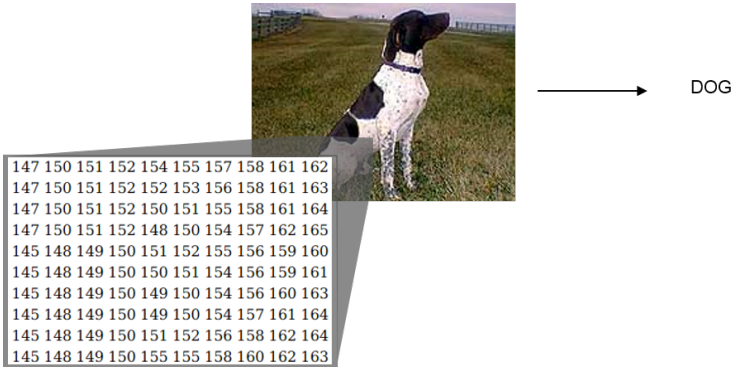
https://www.youtube.com/watch?v=BIZg_PPuj_0

A prelim view on challenges in image analysis

- The human visual system is so good in recognizing objects
- It is almost impossible to specify all variations of objects that can occurr
- Writing new code for each variation is not feasible



Example case: Image classification

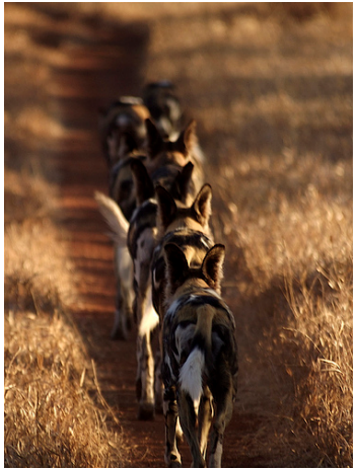


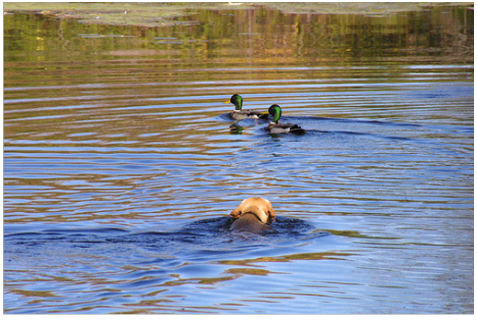
Task: use the entire image to classify the image into one of a set of known classes.

Which object is present in the image?

Challenges: Illumination

Challenges: Occlusion



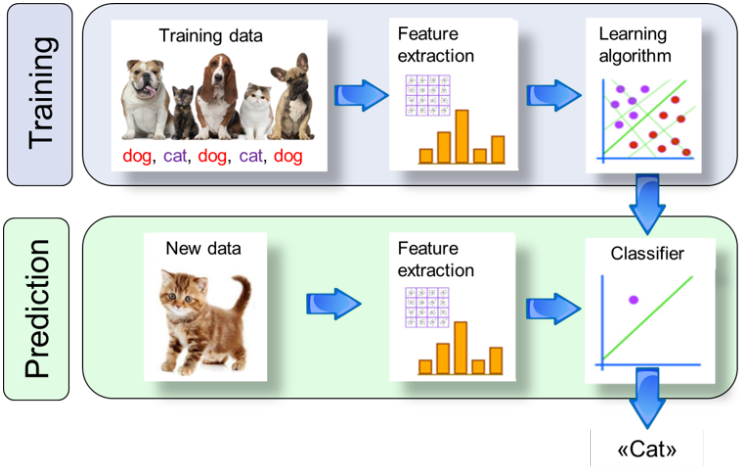


Challenges: Deformation

Challenges: Background clutter

Challenges: Intraclass variation

Classical machine learning



next goal: show why too simple methods may not work well.

Towards a classification baseline: Measuring distances between two images

- Image F, value $F(k,i,j)$ for pixel (i,j) in channel $k \in \{r,g,b\}$.
- **L1-distance:**

$$d_1(F_1,F_2) = \sum_k \sum_i \sum_j |F1(k,i,j) - F2(k,i,j)|$$

- **L2-distance:**

$$d_2(F_1,F_2) = \sqrt{\sum_k \sum_i \sum_j (F1(k,i,j) - F2(k,i,j))^2}$$

L2 is called Euclidean distance[1]

[1] 1. How it is related to a similarity measure? 2. which non-math limitation does that have for image data?

A baseline: K-Nearest-Neighbor classification

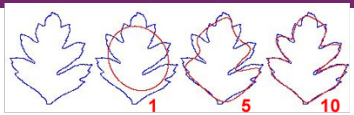
- Given a set of m training images $F_{(i)}$ with known class labels $y_{(i)}, i \in \{1, m\}$
- Classification of a new sample $F_{(new)}$ is done as follows:
  - Out of the m training images, find the k images with smallest distance (measured by L1 or L2) to the new sample $F_{(new)}$
  - Out of these k samples, identify the most frequent class label
  - Assign the class label for the sample $F_{(new)}$ as the most frequent class label among the m. Denote the predicted class as $\hat{y}_{(i)}$.
- Take a test set: measure Classification error on test set by counting the number of samples where the predicted class is unequal to the true class $\hat{y}_{(i)} \neq y_{(i)}$
- k must be selected a priori (try different values of k and choose the one with the lowest classification error using crossvalidation) (for 2-class problems k to be odd)

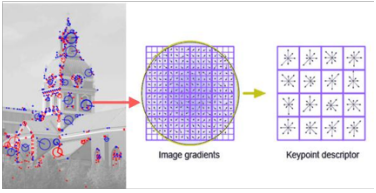
About KNN-classification

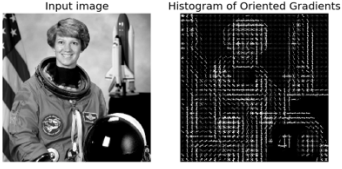
- If $k = 1$ (1NN-classification), each sample is assigned to the class of the closest sample in the training data set.
- In the limit of near infinite training data, this is theoretically a very good classifier for k very large,.
- This classifier involves no training time, but the time needed to classify one pattern will depend on the number of training samples, as the distance to all points in the training set is computed (naive, non-hierarchical case).
- **Problem:** For image classification using the original pixel values as feature values, the results is not invariant to object position, scale, contrast!
- $\Rightarrow$ dont use raw pixels. Extract information, compute some features from the image

Good features are essential

- Features = Data representation
- Good features capture posterior belief about explanatory causes and underlying factors of variation
- Multitude of hand-designed features currently in use







Architectures: image recognition systems

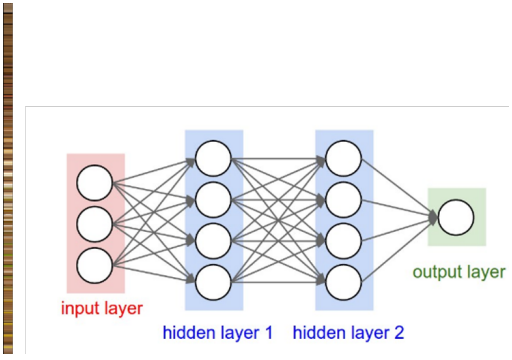
Tradional: Handcrafted features + supervised classifier

Mainstream approach until recently for image (and speech) recognition:

Deep learning: Train multiple stages end-to-end

Deep Learning: Why going beyond a fully connected net? Good Features?

Unwrap 2D image a 1D vector. One input node per pixel. Neurons in each layer are connected to every output from the previous layer.

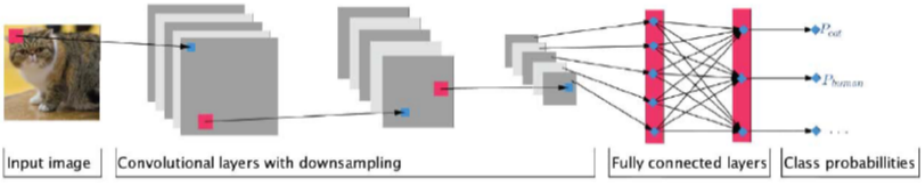


Good Features? Fully connected neural network on images

- Most image applications are absolute position invariant
- A fully connected network will have too many parameters and not be able to scale to normal size images and generalize

Good Features! Convolutional networks enforce similar response to neighboring pixels

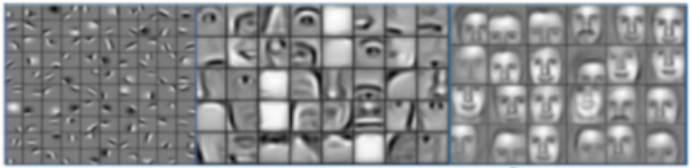
Apply same detection step in sliding windows.

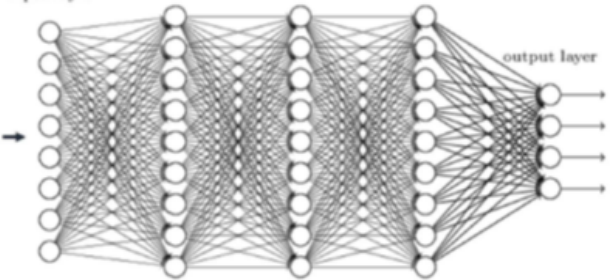


Able to detect same pattern in many locations $\rightarrow$ translation invariance

How do deep networks create invariance?

Deep neural networks learn hierarchical feature representations





What do the layers learn?

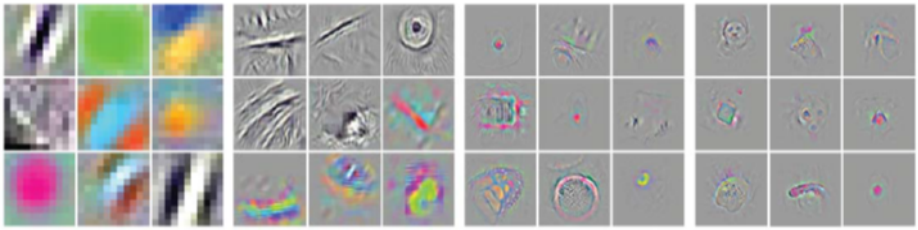
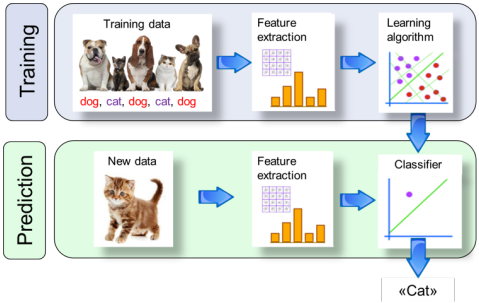


Figure 2: Visualized filter responses for some of the nodes of the first four layers of Alex-net (Krizhevsky et al. (2012)). The layers 1-4 are displayed from left to right. Since the filter responses cannot be directly related to the input image (due to the non-linearity in the network) the visualization technique by Zeiler and Fergus (2014) was used. The filter responses in this figure originally appeared in Zeiler and Fergus (2014) and is printed with permission from the authors of the original article.

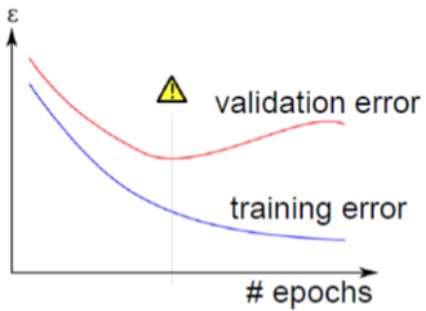
Challenges with deep learning

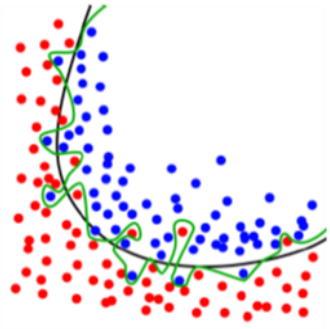
Training a machine learning system



- Training data: data with known prediction labels used to find the parameters of the classifier
- Validation data: data with known prediction labels used to select 1. hyperparameters and 2. architectures
- Test data: data with known prediction labels used once to estimate the error rate of the system

The danger of overfitting to training data





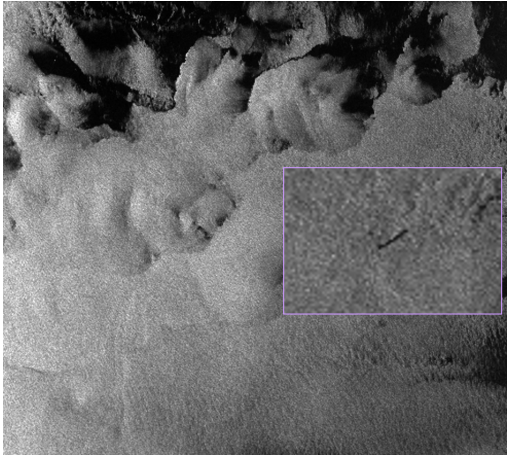
High variance

Working with skewed data

- Challenge: mostly data from “normal” classes, few samples from other classes (e.g. cancer, pollution etc.)
- Problem: How can we avoid that the networks fits the normal class and ignores the small but important class.

Working with skewed data - cont.

- Challenge: mostly data from “normal” classes, few samples from other classes (e.g. cancer, pollution etc.)
- Problem: How can we avoid that the networks fits the normal class and ignores the small but important class.

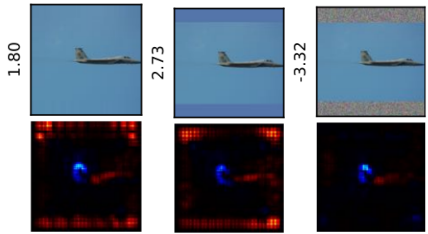


Domain shift

Ohen ghoor!

With power comes responsibility!

- Models with millions of parameters can fit to almost anything!
- Risk: make prediction based on artefacts/undesired queues which correlate with training labels

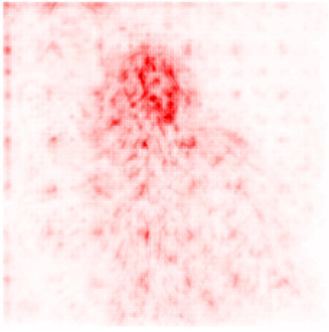


https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G?

- Seeing inside the black box: why did the computer decide the given class?
- How do we evaluate the results? How to find out what we have not learned yet?
- When not to make a prediction ? – outliers
- What are the ethical sides of the problem and the decisions?

Explaining the results

Afghan hound, Afghan



Limitations of deep learning

- overfitting risk with small sample sizes (ideas: fine-tuning, cross-validation instead of train-val split)
- Complex tasks can be difficult to learn, particularly memory-intensive tasks
- Often easy to fool the network: images which to humans look like the right thing.
- You have only learnt what is covered by your training data
- Deep learning requires good knowledge of technical details, and how to train efficiently e.g. `https://arxiv.org/abs/1905.08233`

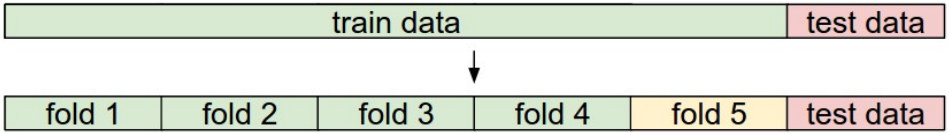
Recap: Hyperparameter selection by Cross-validation

Selecting K for k-NN using crossvalidation

For each value of k:

- Cross-validation: split the training data into d subset/folds
- Train on data from d-1 folds ,
- Estimate the accuracy/compute the number of correctly classified images on the last fold , store the accuracy.
- Repeat this nfold times and compute for the average of the accuracies.
- Repeat with different values of k, select the value that got the highest accuary.
- Using the best value of K, classify the test data set ONCE to get the accuracy of the classifier.

Crossvalidation example - 5-folds



- Example: 10000 training images, 5000 test images.
- Split training images into 5 folds of 2000 images.
- For each fold:
  - Hold fold f out, Train on the other 4 folds (8000 images), compute accuracy on the last 1000 images during cross-valiation.
  - Repeat so each fold is held out once.
- After finding k: train on all 10000, and estimate the reported accuracy on the 5000 test images.

Questions?