

UiO : Department of Informatics
University of Oslo

IN4310 Deep Learning for Image Analysis

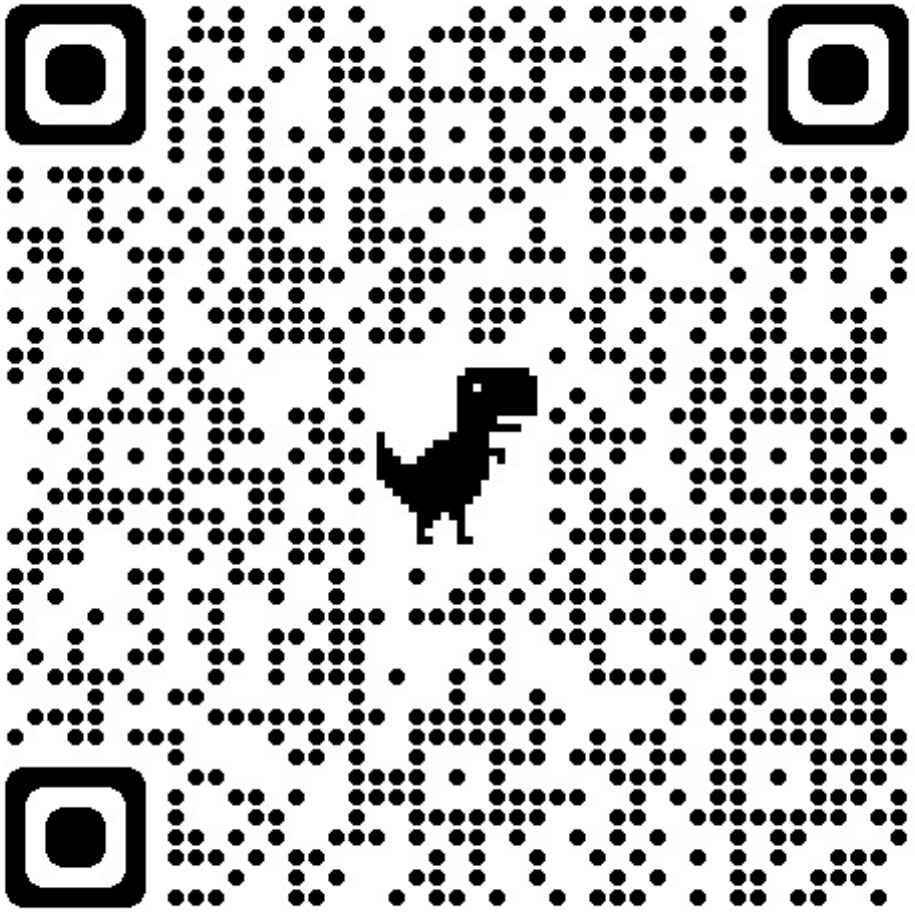
Lecture 16: Course Recap

May 25, 2023

Dhananjay Tomar

Editable Google slides

- https://docs.google.com/presentation/d/1cWly0KKGK3k2h9vZrmx3-TNAKB_ZXlgsFWCm_X5dtd0/edit?usp=sharing
- https://shorturl.at/dpuPY

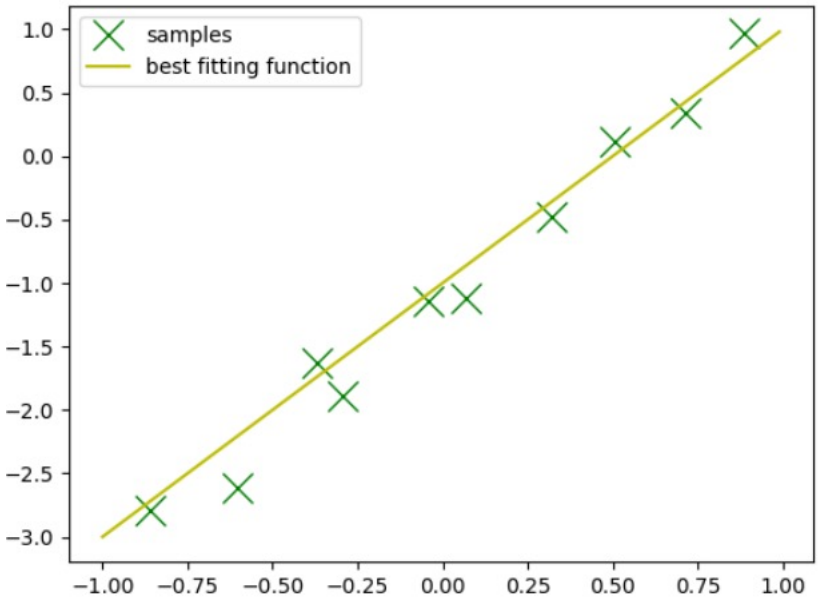


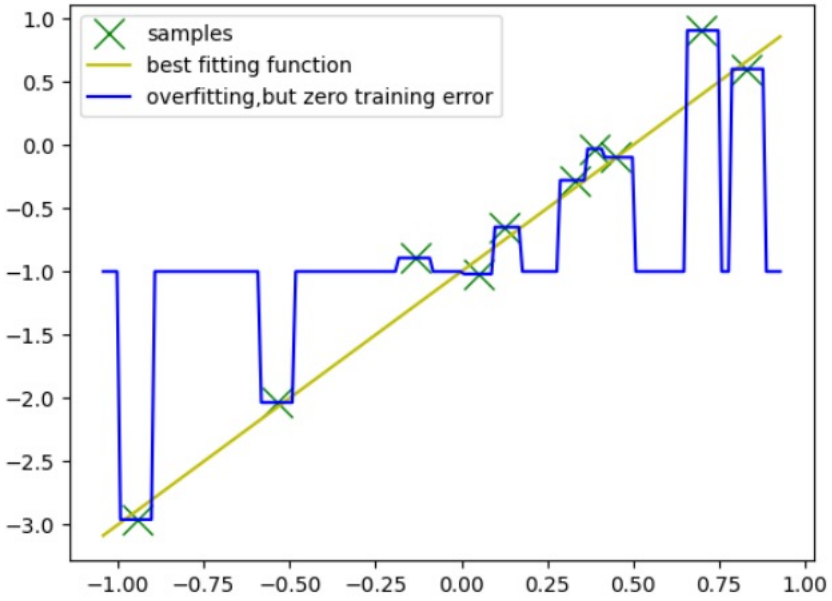
Lecture 1: Introduction

- Generalization:
  - A mapping generalizes well, if it gives low prediction errors on unseen data samples.
  - We model the uncertainty in new unseen data by modelling it as coming from a probability distribution P_{test}.
  - In practice we never know the probability distribution P_{test} which is the model for generating our data samples. We only have a finite set of data samples drawn from it.
  - But trying to generalize information from a finite-sized and noisy test dataset to P_{test} and expectations leads to overfitting which means lack of generalization.

Lecture 1: Introduction

- Overfitting:
  - Overfitting is when the loss on new unseen data points T_n (validation data, test data) is often (with high probability) much higher than the loss of the model on the training data: $L(f, T_n) \gg L(f, D_n)$
  - Overfitting always happens, the only question is: to what extent?



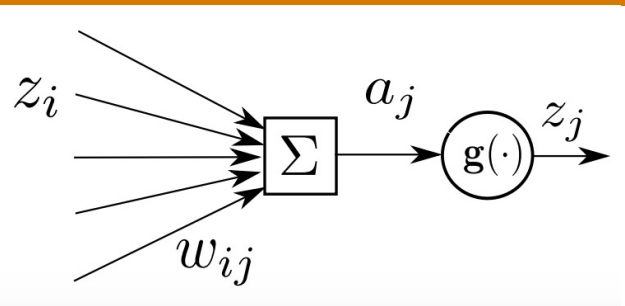


Lecture 2: Introduction to Neural Networks

- Logistic sigmoid function: $s(x) = \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}$
- Logistic regression:
  - Assume we have a linear (or affine) mapping $f_{w,b}(x) = w \cdot x + b$. Plugging it into the logistic sigmoid function s(x) provides a logistic regression model:
    - $s(f_{w,b}(x)) = \frac{e^{w.x+b}}{1+e^{w.x+b}} = \frac{1}{1+e^{-w.x-b}}$
  - For 2-class classification problems, the output $s(f_{w,b}(x)) \in [0, 1]$ is the predicted probability that sample x has class label y = 1, that is, P(Y = 1|X = x).
- Cross-entropy loss:
  - For one-hot labels $y_1, \ldots, y_n \in \{0, 1\}$, the cross-entropy loss of a data point (x_i, y_i) is the negative logarithm of the predicted probability of the ground-truth class.
    - If $y_i=1$, CE Loss = $-\log(s(f_{w,b}(x_i)))$
    - If $y_i=0$, CE Loss = $-\log(1 - s(f_{w,b}(x_i)))$

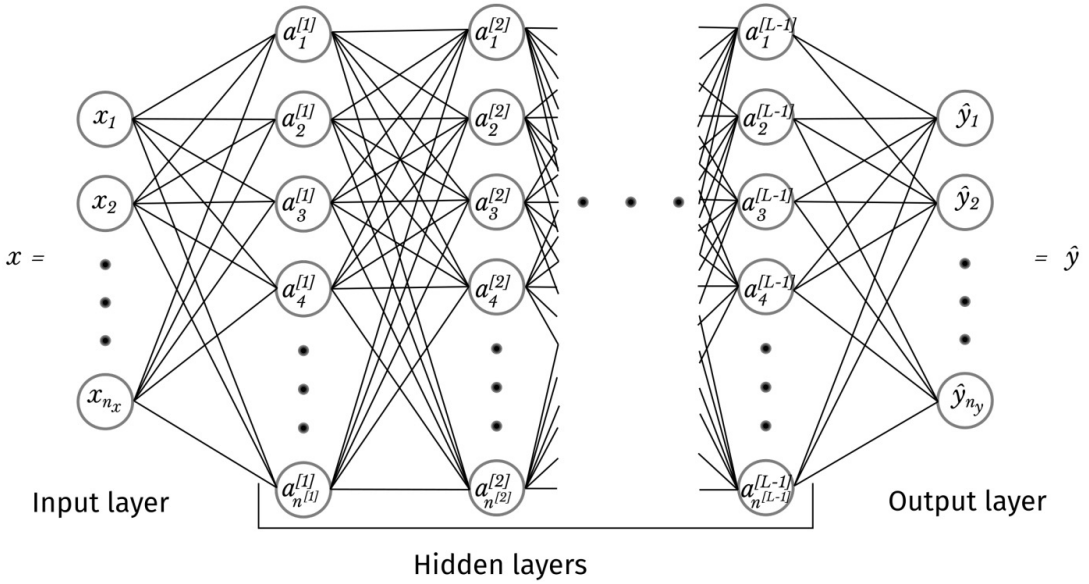
Lecture 2: Introduction to Neural Networks

- Artificial neuron
- Artificial neural network



- The output layer uses softmax function:
  - $s(x_k) = \frac{e^{x_k}}{\sum_{i=1}^{n} e^{x_i}}$

- The cross-entropy loss between two probability distribution vectors $[z_1,..., z_i,..., z_n]$ and $[y_1,..., y_i,.., y_n]$ is defined as:
  - $L(y, z) = -\sum_{i=1}^{n} y_i \log(z_i)$



- Logistic regression is an exemplary model for a single neuron network, that is, a 1-layer neural network (0 hidden layers) with one node in the output layer.

Lecture 3: Convolutional Neural networks

- Convolutional layer (1-D convolution, 2-D convolution)
- Padding
- Receptive field
- Pooling – max pooling, average pooling
- Dilated convolution (also covered in the lecture on Image Segmentation)
- Depth-wise separable convolution
  - Depthwise convolution: process each filter independently and generate 1 output filter per input filter
  - Pointwise convolution: use normal 1x1 convolution to get the required N channels as output

Lecture 4: Gradient Descent, Properties of the Gradient, Backpropagation

- How to compute gradients.
- Gradient descent.
- Learning rate schedule.
- Stochastic gradient descent.
- Backpropagation.
  - Do not confuse backpropagation with gradient descent

Lecture 5: State of the art architectures in classification for vision

- Dropout
- GoogLeNet (Inception)
  - Has multiple versions
  - Auxiliary losses
  - Parallel convolutions with different filter sizes
  - Factorise a conv with filter size nxn as a sequence of 1xn followed by nx1
- ResNets
  - Residual connections / skip connections
    - Helps with vanishing gradients by providing a direct path for gradients to flow backwards.
- Batch Normalization
  - Take note of how running mean and running variance are used
- Layer Normalization
  - Each sample in the batch is normalised independently.
  - No running mean, variance.

Lecture 5: State of the art architectures in classification for vision

- Densenets
- Finetuning
  - Take a pre-trained network and train only the last few layers for small datasets and maybe all layers for larger datasets.

Lecture 6: Optimizers

- A useful blog post to understand the basics but refer to slides for the exam: https://ruder.io/optimizing-gradient-descent/
- Learning rate schedule
- Warm-ups
- Weight decay
- Gradient descent with momentum
  - Acts as a memory
  - Can help overcome valleys and small local minimas
  - Nesterov momentum
- Exponential Moving Average, RMSProp
- Adam optimizer
  - A separate effective learning rate for each parameter
  - Couple with weight decay to get AdamW
- Squeeze-and-excitation blocks, ConvNext

Lecture 7: Performance Estimation

- Train | Validation | Test subsets (a recap)
  - Different strategies for creating these splits
- External testing, domain shift (domain shift is also covered in lecture 8)
- Facilitating generalisation (model's ability to perform well outside training data)
  - Reduce network's capacity: #trainable parameters, dropout, weight decay
  - Facilitate learning: residual connections, normalizations, etc.
  - ...
- Different performance metrics
- Estimator uncertainty (confidence intervals)

- No need to memorise the papers (with results) in the slides.
- No need to memorise the minor details in the slides.

Lecture 8: Data Augmentation

- Geometric transformations
  - Modify size, change orientation, random crop etc.
- Photometric transformations
  - Modify brightness, contrast, hue, saturation etc.
- Other transformations
  - Blur, add noise, filters etc.
- How to allow CNNs to process images of any size?
- Weight initialization strategies
  - Try to get 0 mean and same variance across all the layers
- Contrastive learning

Lecture 9&10: Recurrent Neural Networks

- Basics of RNNs
- Different input-output structures of RNNs: one-to-one, one-to-many, etc.
- Training RNNs: BPTT, truncated BPTT
- Exploding and vanishing gradients
  - Do not memorise the proof
- Preserving long range dependencies: LSTM, GRU (already covered in mandatory 2)
- Multilayer RNNs, bidirectional RNNs
- Attention mechanism

Lecture 11: Adversarial attacks

- White box attacks: The attacker has access to the model's parameters
- Targeted attacks: Trick the network to classify a sample into a fixed class which is different from the true class
- Untargeted attacks: Trick the network to misclassify the adversarial image
- Iterative gradient descent/ascent
  - Gradient ascent away from original class: untargeted
  - Gradient descent towards the least probable class: targeted
- Projected gradient method
- Defences
  - Adversarial training
  - Image transformations like quantisation
  - etc.
- Understand why do these attacks exist at all

Lecture 12: Object Detection

- Image classification + localization for one object
- One-stage vs two-stage detectors
- Faster R-CNN
- Use of anchors for detecting multiple objects
- How to handle objects of varying sizes?
- Non-maximum suppression
  - DO memorise the algorithm. It's very simple so it shouldn't be difficult.
- How to handle too many background predictions?
  - Focal loss
- Performance metrics

Lecture 13: Image Segmentation

- Semantic segmentation vs Instance segmentation vs Panoptic segmentation
- How to capture global context?
  - Downsampling
  - Dilated/atrous convolution
- How to upsample feature maps to get pixel-level labels?
  - Nearest neighbour upsampling
  - Unpooling
  - Transposed convolution
- No need to memorise the exact architectures, just know the principles behind those architectures
- From object detection to instance segmentation
- Performance metrics and loss functions

Lecture 15: Vision Transformers

- Self-attention
  - Positional encodings
  - Masked self-attention
  - Multiple heads
- Transformer encoder and decoder blocks
  - You should know the difference but no need to memorise the exact architecture
- Using transformers for images
  - Vision transformer (ViT)
  - Swin transformer
- Object detection using transformers: DETR