Summary

5LSM0: Convolutional neural networks for computer vision

Fons van der Sommen

Electrical Engineering / VCA research group

Administrative

FINAL ASSIGNMENT

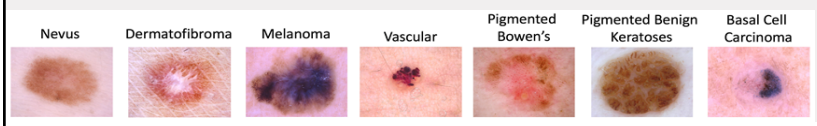
Final assignment on canvas!

Automotive: Cityscapes segmentation challene

- https://www.cityscapes-dataset.com/benchmarks/#scene-labeling-task

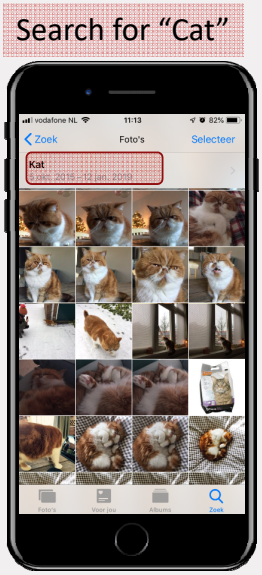
Healthcare: ISIC 2018 Skin lesion classification

- https://challenge2018.isic-archive.com/task3/



Introduction

Q: What key two developments have enabled this revolution?



Computer vision

- Can we make a computer see?
- Earliest attemts during the 60ies
- Image processing methods

Deep learning

- Revolutionized machine learning/computer vision
- Convolutional neural network

Computer vision subdomains

- Classification, Segmentation, Detection, Tracking, Re-identification

Data-driven image classification

Computer vision is hard

- Variation in viewpoint, illumination etc

➢ Changes almost all pixel values!

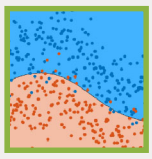
Linear classification

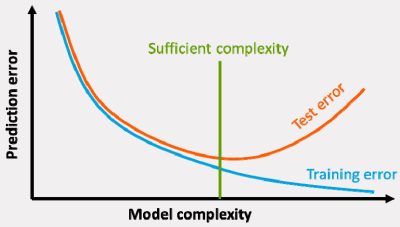
$$f(\mathbf{x}, \mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Q: To what mathematical operation is this equivalent?

Image ▷ Function ▷ Prediction

Feature extraction + machine learning





Model performance evaluation

- Almost all models contain hyperparameters
- Use appropriate metrics / validation

Data-driven image classification

Questions

A. How does KNN work? What are two major downsides of this classification method?

B. For what cases could this approach potentially be useful?

C. Using raw pixels a good idea? Why not? Better idea?

D. In a linear classification function $f(\mathbf{x}, \mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$,
if $\mathbf{x}$ represents a 10 by 10 pixel image what's the size of $\mathbf{W}$, $\mathbf{b}$?

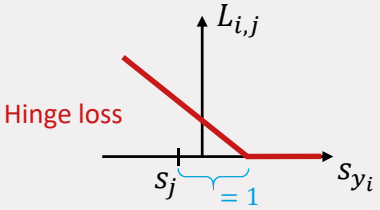
E. How could you view the vectors of $\mathbf{W}$?

F. Why never evaluate your model on the training set? Why are these numbers still useful?

G. Why is the vanilla Accuracy a poor metric for a lot of problems? What better to use?

H. How can we estimate good hyperparameters for our model?

Loss functions and optimization



Loss functions

- Way to quantify the "badness" of a certain (set of) prediction(s)
- Highe loss, softmax (cross-entropy) loss
- Regularization term to quantify how much we like the solution

Optimization

- Follow the slope to go downwards in the loss-landscape
- Compute gradient information to find the slope
  - *Finite differences is super slow!*
  - *Derive analytical expression to compute gradient at a given point*
- Data sets can be large, we don't want to wait until we've seen all the data
  - *Mini-batch gradient descent!*

$$W \leftarrow W - \eta \nabla_W L$$

Loss functions and optimization



Car	**2.1**	2.1	-0.5
Boat	1.4	**1.3**	1.5
Tree	-0.1	-2.2	**3.1**

Questions

A. Hinge-loss for the example on the right?
B. Downside of the Hinge-loss? Min/max possible loss?
C. What is the Hinge-loss at initialization (with small weights)?
D. Difference between L_1 regularization and L_2 regularization? (in terms of resulting weights)
E. Cross-entropy loss is –log of the softmax function, why use softmax? Why take –log?
F. Min/max cross-entropy loss? Loss at initialization? For what score vector is the loss zero?
G. How does gradient descent minimize the loss function?
H. Crucial parameter in this procedure?
I. Effect of taking mini-batches?

$$L_i = -\log\left(\frac{e^{s_k}}{\sum_j e^{s_j}}\right)$$

Neural networks and backpropagation

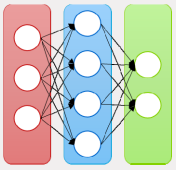
Aritificial neural networks

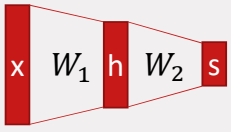
- Biologically inspired
- Linear classification followed by nonlinearity
- Output can be computed efficiently

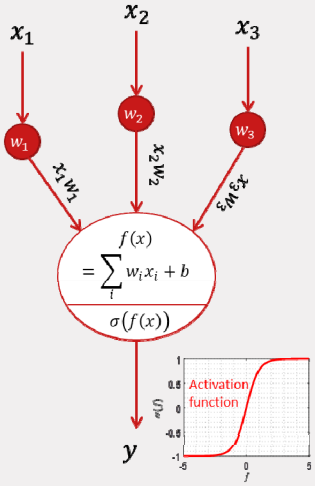
Computational graphs

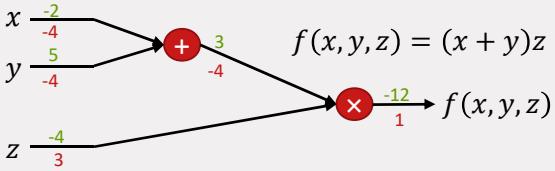
- Write function as computational graph
- Find the gradient of the loss function
  (at any given location)

Q: What is this "location" exactly?









Neural networks and backpropagation

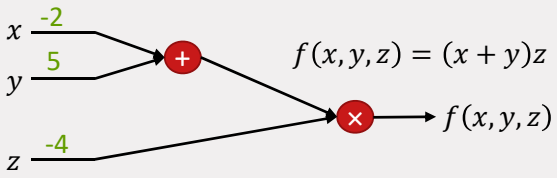
Questions

A. Why do backprop for neural networks?

B. In the example to the right:

- $\partial f/\partial x, \partial f/\partial y, \partial f/\partial z?$

C. Goal of forward and backward pass?

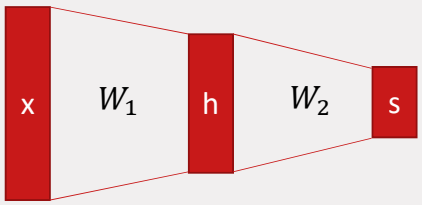


D. What can we already compute and store for each node in the forward pass?

E. How many layers does the network on the right have?

F. What crucial operation allows us to do more interesting stuff than just linear classification?

G. How can we efficiently compute the inputs to a layer?



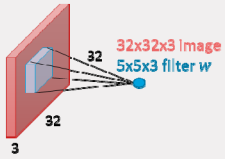
Convolutional Neural Networks (CNNs)

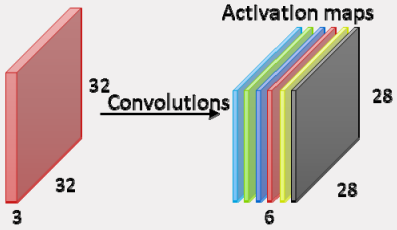
Hierarchical filtering structure combined with neural network

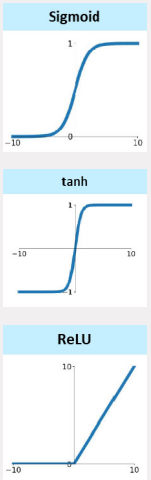
- In each layer, apply several filters to the (intermediate) representation
- Filter kernels can be learned from the data by means of backpropagation!

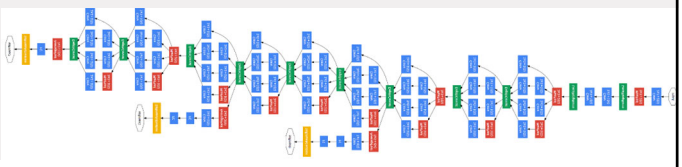
Many different architectures

- AlexNet, ResNet, VGG, …
- Mostly reconfiguration of a set of based building blocks:
  - *Norm, convolution, activation, pooling, FC, …*





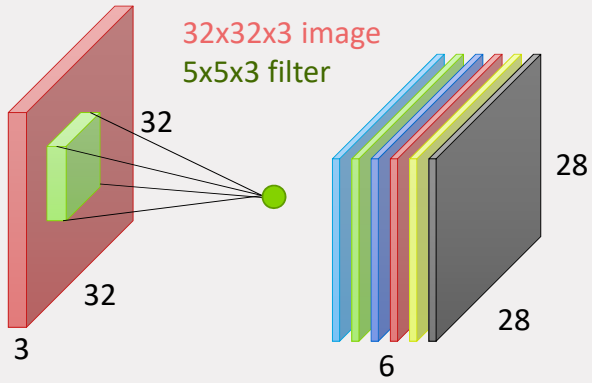




Convolutional Neural Networks (CNNs)

Questions

A. Difference between conventional computer vision / machine learning and deep learning (CNNs)?
B. Number of parameters in convolution step? ->
C. Purpose of pooling layers?
D. What structures captured in the first layer?
E. And in the later layers?
F. Difference between conv layers and FC layers?
G. Problem with sigmoid activation?
H. Solved by tanh! Problem with tanh?
I. Advantage of ResNet?



Training neural networks

Before you start

- Normalize your data
- Initilialize your weights (smartly!)
- Pick a proper optimizer (e.g. ADAM)

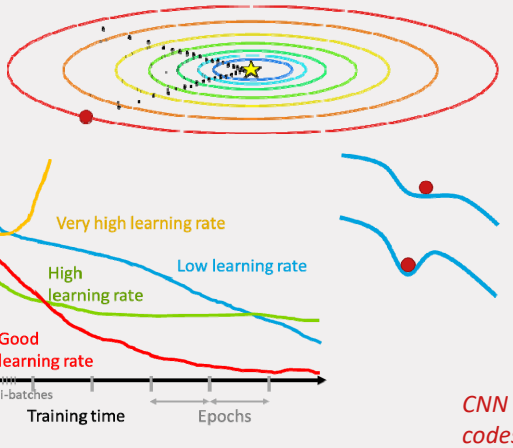
While training

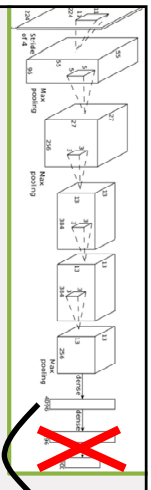
- Learning rate decay
- Observe loss function

Regularization

- Drop-out, data augmentation

Transfer learning





CNN codes

Conventional classifier

Training neural networks

Questions

A. Why normalize to zero-mean? Why is unity std dev less of an issue with images?

B. Problem with weight initialization using small Gaussian distributed numbers?
   - *What happens during forward pass? Backward pass?*

C. Problem with initializing with big weights?

D. How does BatchNorm address this problem?

E. How should we find our hyper parameters? How to sample the parameter space? Why?

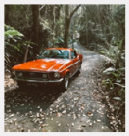
F. How can we identify good training from a loss curve? How can we see it goes wrong?

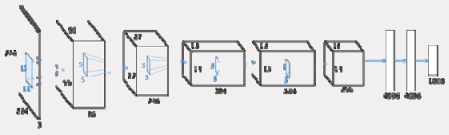
G. Why can loss curves also increase during training?

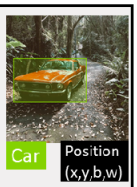
H. What is meant by a loss function with a bad condition number? Why is this a problem? How can you solve it?

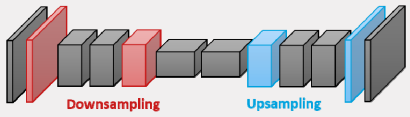
Supervised learning

Data + "ground truth" / reference









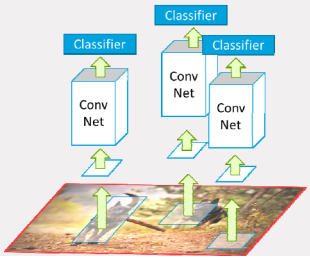
- Pooling
- Strided convolution
- Unpooling
- Strided transpose convolution

Classification (+localization)
- CNN (+ regression loss)

Object detection
- R-CNN, YOLO, Single-Shot Detection (SSD)

Semantic segmentation
- Fully Convolutional Network (FCN)

Instance segmentation
- Mask-RNN



Supervised learning

Questions

A. Difference between semantic segmentation and instance segmentation?
B. What is unpooling? Two different varieties?
C. Fundamental difference with transpose convolution?
D. If we want to do object detection, why don't we run a sliding window over an image with a pre-trained CNN attached to it?
E. In Classification + Localization we use two losses, which ones? For what purposes?
F. Why can't we just add these losses?
G. Major problem with R-CNN?
H. How can we improve this?
I. Why can't we push this through the FC layers?

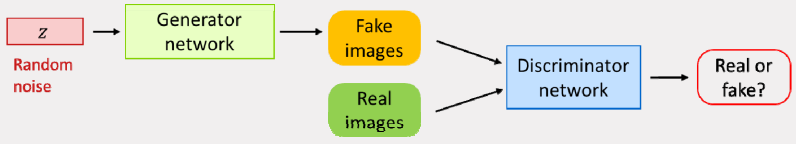
Unsupervised learning

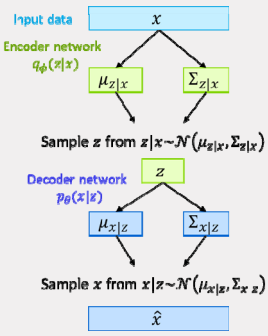
Variational Autoencoders (VAEs)

- Optimize variational lower bound on likelihood. Useful latent representation, inference queries.
- Current sample quality not the best.

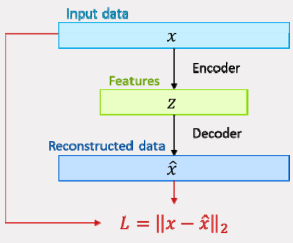
Generative Adversarial Networks (GANs)

- Game-theoretic approach, best samples!
- Can be tricky and unstable to train, no inference queries (e.g. $p(x)$, $p(x|z)$)

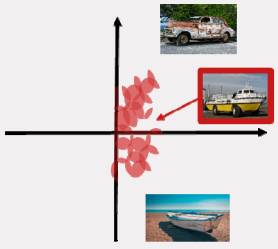




Variational Auto-encoders



Auto-encoders



Unsupervised learning

Questions

A. How does PixelRNN work and what is a major problem with this approach?

B. What's the main idea of an auto-encoder?

C. Why can't we just sample from the feature space of an auto-encoder and expect to obtain visually similar examples? (Or output that even makes sense for that matter…)

D. How does a variational auto-encoder solve this?

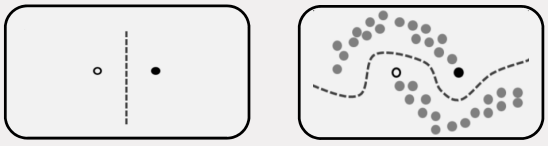
E. How do the two terms of the VEA loss function relate to this solution?

$$\mathcal{L}(x^{(i)}, \theta, \phi) = \mathbf{E}_z[\log p_\theta(x^{(i)}|z)] - D_{KL}\left(q_\phi(z|x^{(i)})||p_\theta(z)\right)$$

F. What can we do with a VAE what we cannot do with a GAN?

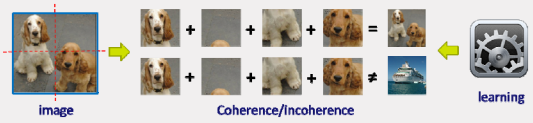
G. In a GAN, What does this $D(G(z)) \rightarrow 0$ imply?

Beyond supervised learning



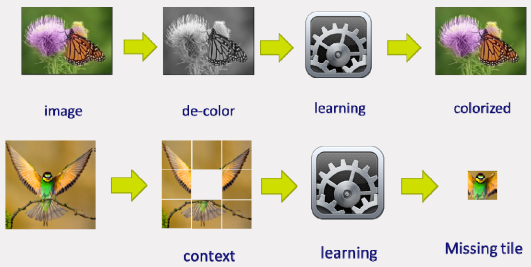
Semi-supervised learning

- Distribution of unlabeled data may help finding an accurate model for the labeled data
- Self-training: use model to predict labels of unlabeled data and retrain



Self-supervised learning

- Create your own labels to capture data structure
- Use a transform and learn to undo it
  - *E.g. inpainting, rotation, counting, colorization, …*
- Beware of potential pitfalls!



Beyond supervised learning

Questions

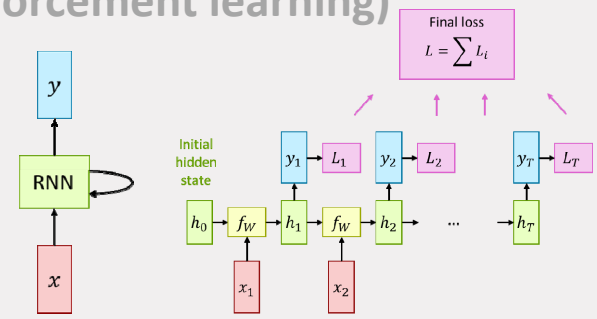
A. When to use self-supervised learning?
B. How can we evaluate if the self-supervised learning has done a good job?
C. How do you proceed after self-supervised learning?
D. Potential pitfalls of self-supervised learning?
E. How can you identify these pitfalls?

Not in exam!

Sequence modeling (and reinforcement learning)

Recurrent Neural Networks (RNNs)

- Model sequential behavior with a CNN
- Hidden state feeds back into itself
- Unroll network over sequential axes (e.g. time)



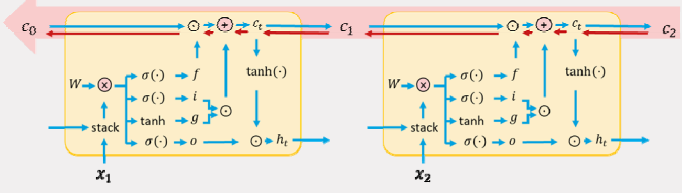
Long Short-Term Memory (LSTM)

- Additional "cell-state"

$$h_t = o \odot \tanh(c_t) \qquad c_t = f \odot c_{t-1} + i \odot g$$

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma(\cdot) \\ \sigma(\cdot) \\ \sigma(\cdot) \\ \tanh(\cdot) \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

i	Input gate
f	Forget gate
o	Output gate
g	... Gate

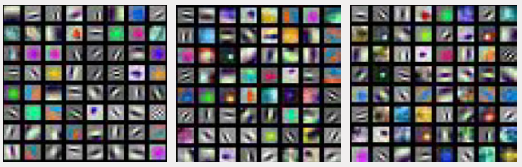


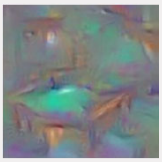
Sequence modeling and reinforcement learning

Questions

A. What is truncated backpropagation through time? And why use it?
B. To what other training approach is this similar?
C. Main benefit of LSTM vs vanilla RNN?
D. What problems can occur during training of a vanilla RNN?
E. To what other CNN architecture does a RNN with LSTMs look similar?
F. Why does the g-gate use a tanh activation, while the other gates use a sigmoid?

Visualization and understanding



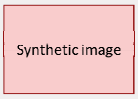


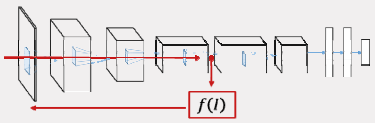
Visualize the effect of certain layers

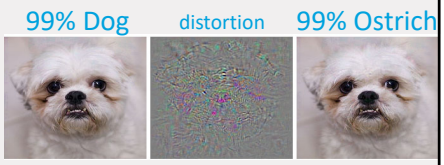
- Check out filter kernels or last layer

Investigate specific neurons

- Store the patches for which a neuron fires
- Synthesize images that cause a neuron to fire







Visualize the gradient

- Compute the gradient on the pixels to see the effect of individual pixels on the loss

Beyond visualization

- Exploit the gradient to fool the network!
- Texture synthesis and style transfer

Visualization and understanding

Questions

A. What the **** is happening over there?! ->>
B. Why can't we straightforwardly visualize the filter kernels of all layers after the first layer?
C. How can we obtain information on what the last layers are looking for?
D. When we look at maximally activating patches, the later layers react to larger patches, why?
E. How can we create images that have a similar texture as a certain reference image?

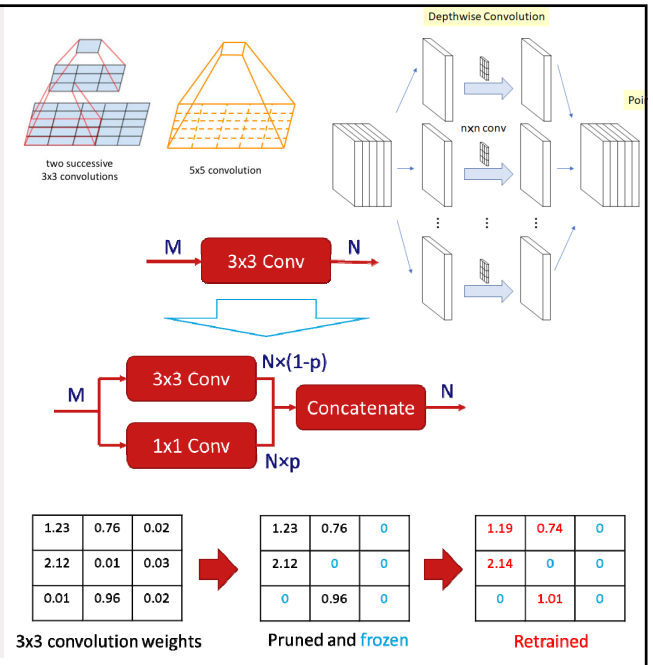
Efficient deep learning

Designing memory-efficient deep convolutional architectures

- Replacing fully connected layers by convolutions
- Depth-wise convolutions
- Stacking filters and spatial kernel reduction
- Evenly spaced down-sampling

Channel reduction

- Deep compression: compressing conv weights
- Pruning + Huffman coding
- Weight quantization



Concluding remarks

Thanks for your attention!

Follow-up course:

- **5AUA0** Advanced Sensing using Deep Learning – Quartile 4

Internships / graduation projects available! Contact:

- Me (fvdsommen@tue.nl), Gijs (G.Dubbelman@tue.nl) or Egor(E.Bondarev@tue.nl)
- *For, respectively, Healthcare, Automotive or Surveillance*

Rate my course please!☺