Statistisches Data Mining (StDM)
Woche 14

Oliver Dürr
Institut für Datenanalyse und Prozessdesign
Zürcher Hochschule für Angewandte Wissenschaften

oliver.duerr@zhaw.ch
Winterthur, 20 Dezember 2016

No laptops, no phones, no problems

Multitasking senkt Lerneffizienz:

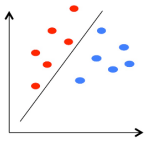
- **Keine Laptops im Theorie-Unterricht Deckel zu oder fast zu (Sleep modus)**

Orga

- Prüfung
  - Openbook oder nicht?

Overview of classification (until the end to the semester)

Classifiers

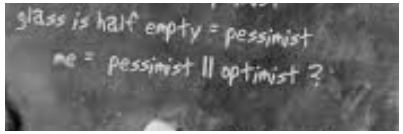


K-Nearest-Neighbors (KNN)
Logistic Regression
Linear discriminant analysis
Support Vector Machine (SVM)
Classification Trees
Neural networks NN
Deep Neural Networks (e.g. CNN, RNN)
...

Evaluation

Cross validation
Performance measures
ROC Analysis / Lift Charts

Theoretical Guidance / General Ideas

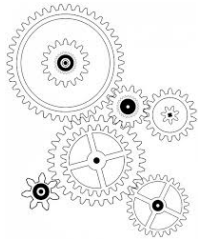


Bayes Classifier
Bias Variance Trade off (Overfitting)

Combining classifiers

Bagging
Random Forest
Boosting

Feature Engineering

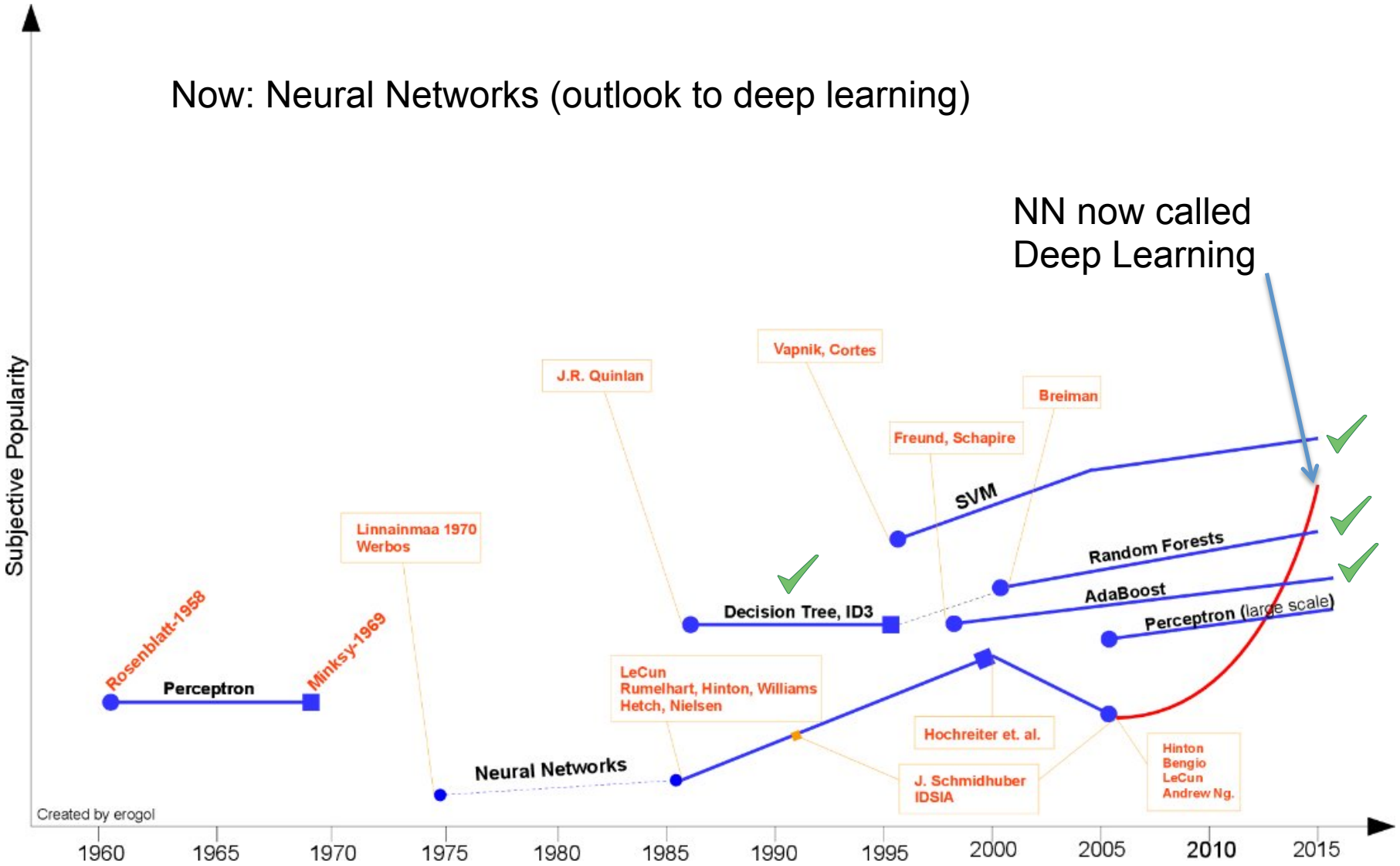


Feature Extraction
Feature Selection

Neural Networks

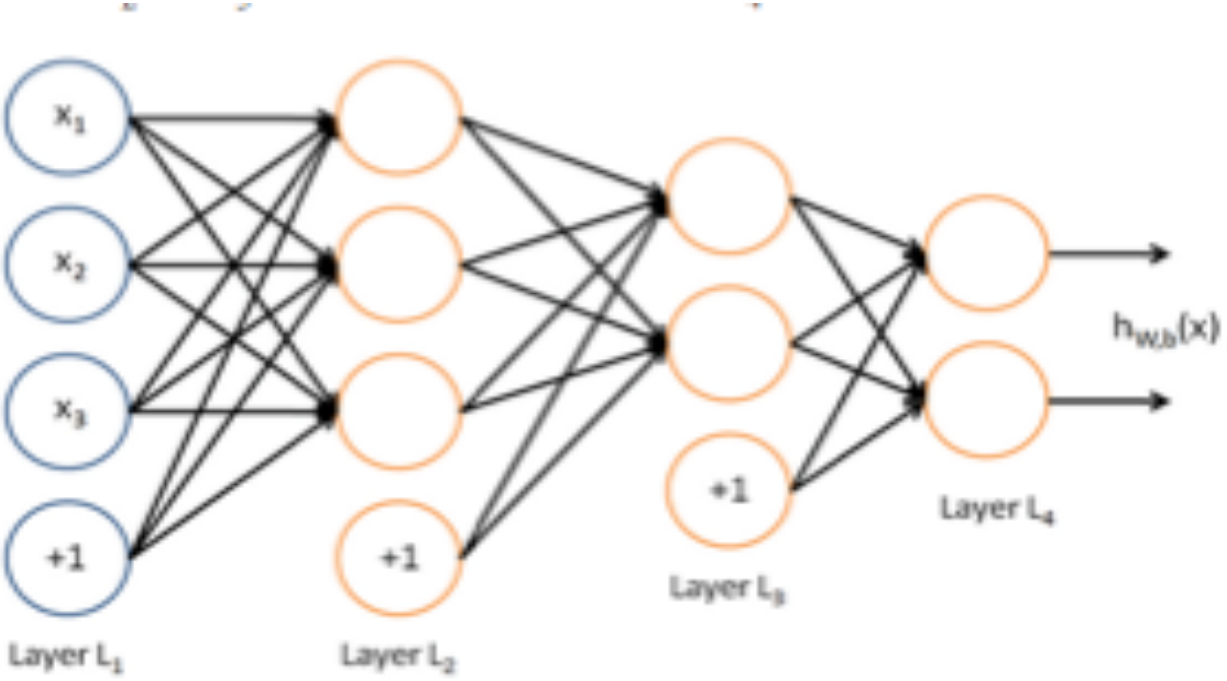
Brief History of Machine Learning (supervised learning)

Now: Neural Networks (outlook to deep learning)

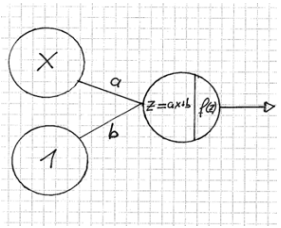


Taken from http://www.erogol.com/brief-history-machine-learning/

Architecture of a NN



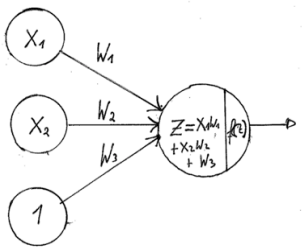
We start with....



1-D Logistic Regression

Cost function for multinomial regression

Training Examples Y=1
or Y=0

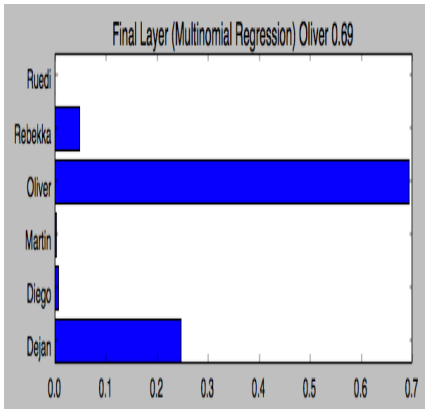


$$-nJ(\theta) = L(\theta) = L(a,b) = \sum_{i \in All\ ones} \log(p_1(x^{(i)})) \quad + \sum_{i \in All\ zeros} \log(p_0(x^{(i)}))$$

N Training Examples classes (1,2,3,…,K)

Output of last layer

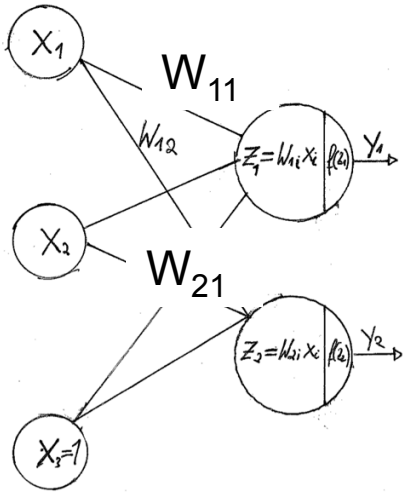
p_i



$$-nJ(\theta) = L(\theta) = L(a,b) = \sum_{i \in y_j=1} \log(p_1(x^{(i)})) \quad + \sum_{i \in y_j=2} \log(p_2(x^{(i)})) + \ldots + \sum_{i \in y_j=K} \log(p_K(x^{(i)}))$$

Example: Look at class of single training example. Say it's Dejan, if classified correctly p_dejan = 1 ➔ Loss = 0. Real bad classifier put's p_dejan=0 ➔ Loss = Inf.

Loss Function for multinomial regression



Function to maximize prob. to a certain class in last layer

$$-nJ(\theta) = L(\theta) = L(a,b) = \sum_{i \in y_j = 1} \log(p_1(x^{(i)})) \quad + \sum_{i \in y_j = 2} \log(p_2(x^{(i)})) + \ldots + \sum_{i \in y_j = K} \log(p_K(x^{(i)}))$$

Sum is over all training data with belong to class 1.

- Optimisation with gradient descent

$$\theta_i'' \leftarrow \theta_i' - \alpha \left. \frac{\partial J(\theta)}{\partial \theta_i} \right|_{\theta_i = \theta_i'}$$

Multinomial Logistic Regression in R

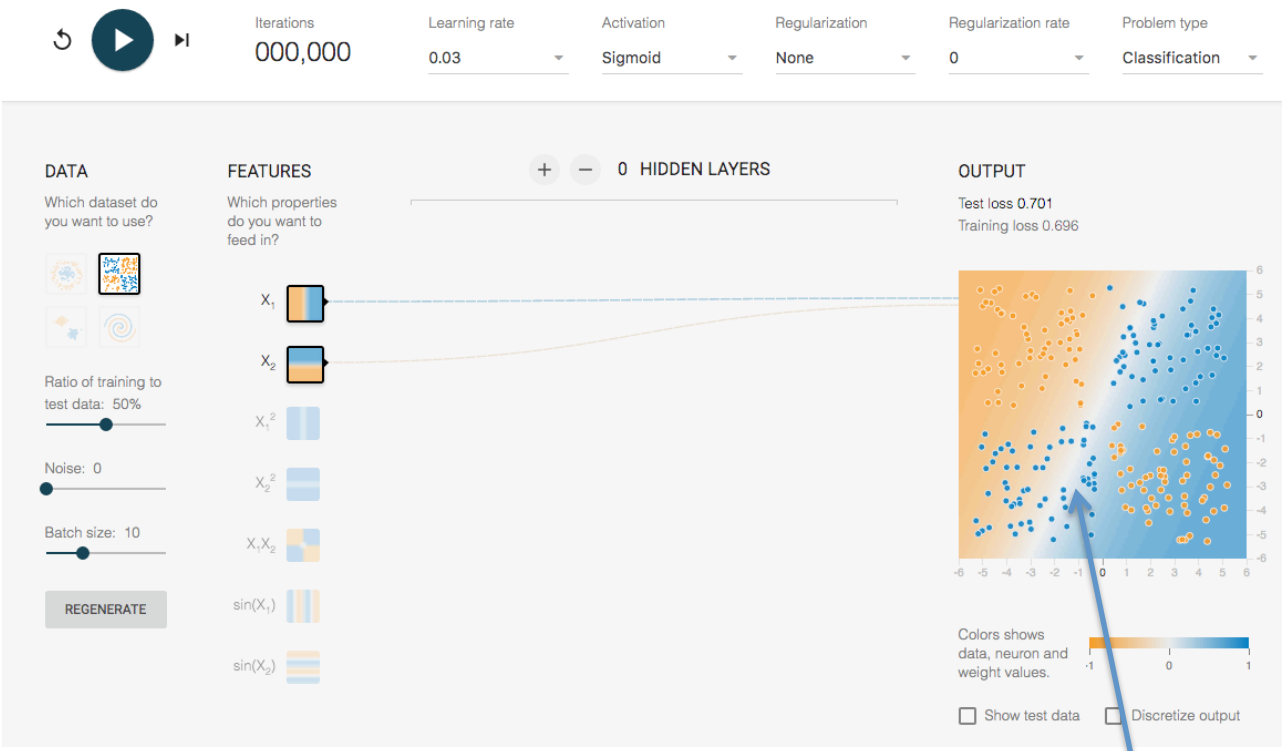
```
library(nnet)
#An iterative procedure minimizing the loss function
fit  = multinom(Species ~ ., data = iris)
predict(fit, iris, type='class')[1:5]
preds = predict(fit, iris, type='prob')
true_column = as.integer(iris$Species)

# Check if we really optimized the loss function
p_class = c(preds[1:50,1], preds[51:100,2],
preds[101:150,3]) #These are sorted
-sum(log(p_class)) # The cost function
```

Ende Wiederholung

Limitations of (multinomial) logistics regression

Linear regression in NN speak: “on hidden layer”



Network taken from

Linear Boundary!

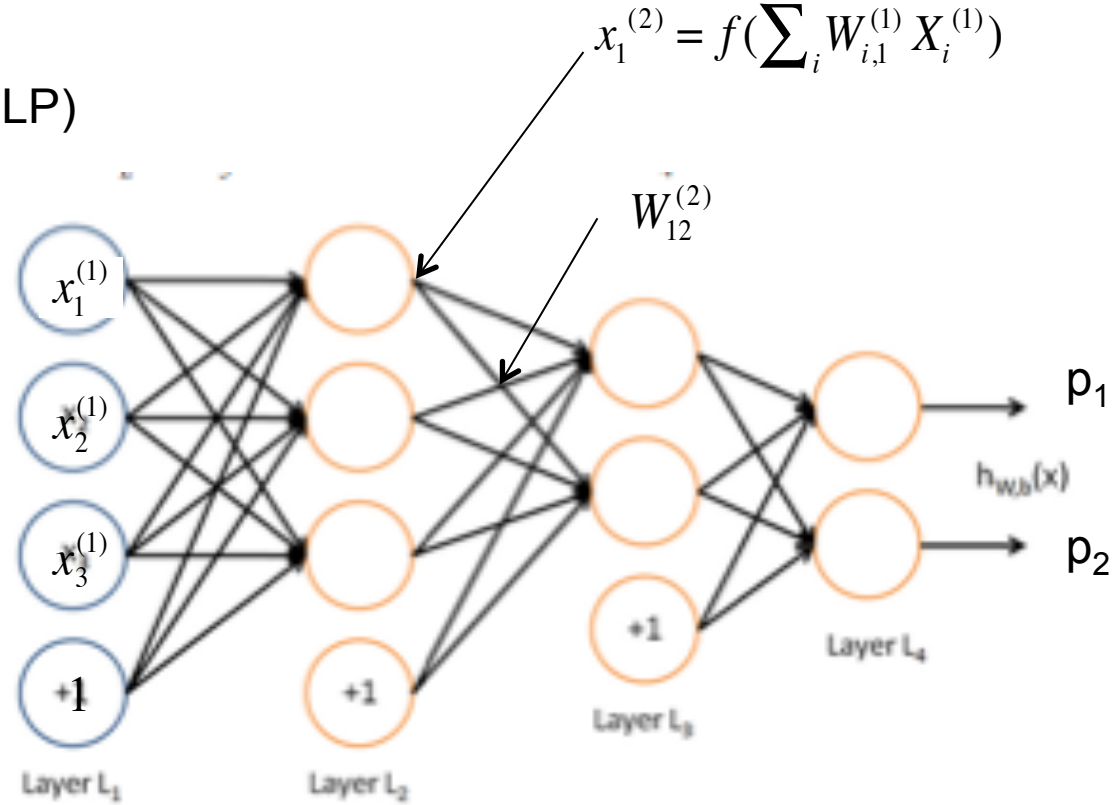
More than one layer

We have all the building blocks

- Use outputs as new inputs
- At the end use multin. logistic regression
- Names:
  - Fully connected network
  - Multi Layer Perceptron (MLP)

- Training via gradient descent

$$\theta_i'' \leftarrow \theta_i' - \alpha \left. \frac{\partial J(\theta)}{\partial \theta_i} \right|_{\theta_i = \theta_i'}$$



Simple chaining for afficinados (Einstein sum convention):

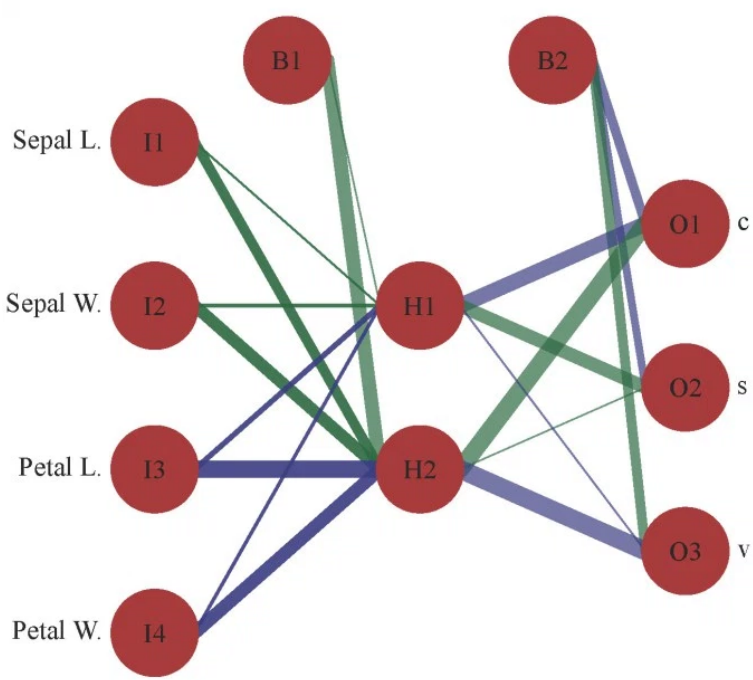
$$x_j^{(l)} = f(W_{j'j}^{(l-1)} f(W_{j''j'}^{(l-2)} f(W_{j'''j''}^{(l-3)}) \cdots f(W_{j''''j'''}^{(1)}))))$$

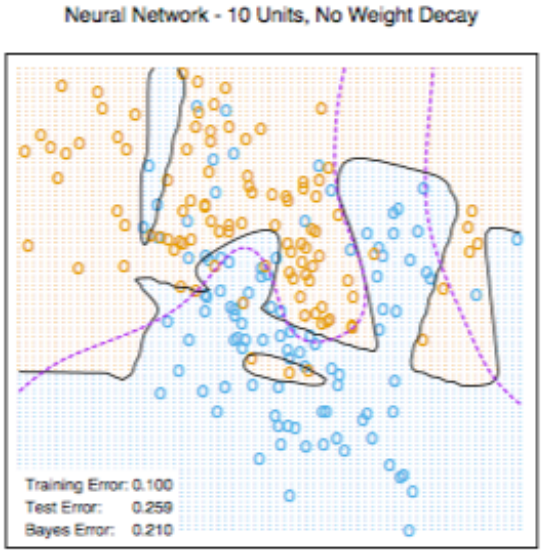
Neural Network with hidden units

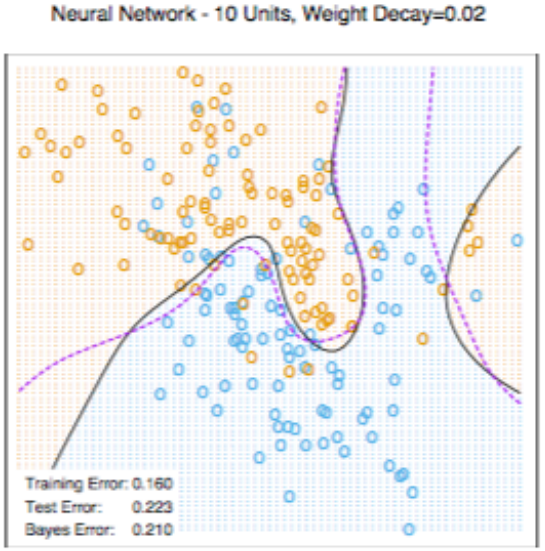
- Go to http://playground.tensorflow.org and train a neural network for the data:

- Use sigmoid activation and start with 0 hidden layers. Increase the number of hidden layers.

One hidden Layer







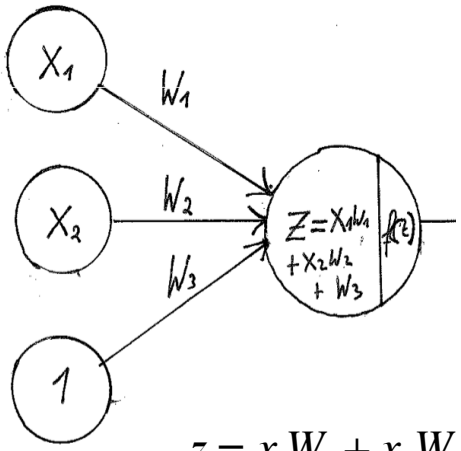
A network with one hidden layer is a universal function approximator

NN with one hidden layer in R

```
#### Fit a neural net with one hidden layer (size 12)
spam.fit = nnet(spam ~ ., data = spam[-testIdx,],
size=12, maxit=500)
spam.pred <- predict(spam.fit, spam[testIdx,],
type='class')
# Result
table(spam$spam[testIdx], spam.pred)
```

Different Activations

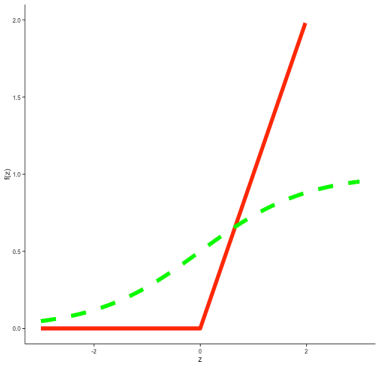
N-D log regression



$$f(z) = \begin{cases} \dfrac{\exp(z)}{1+\exp(z)} \\ \max(0,z) \end{cases}$$

$$z = x_1 W_1 + x_2 W_2 + W_3 = \theta^T x$$

Activation function a.k.a. Nonlinearity f(z)



Motivation:

Green:
logistic regression.

Red:
ReLU faster convergence

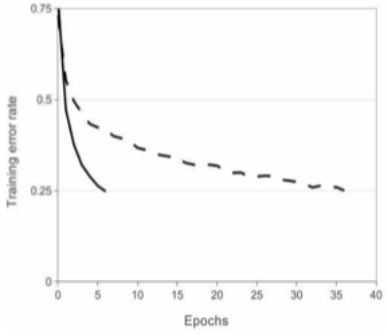
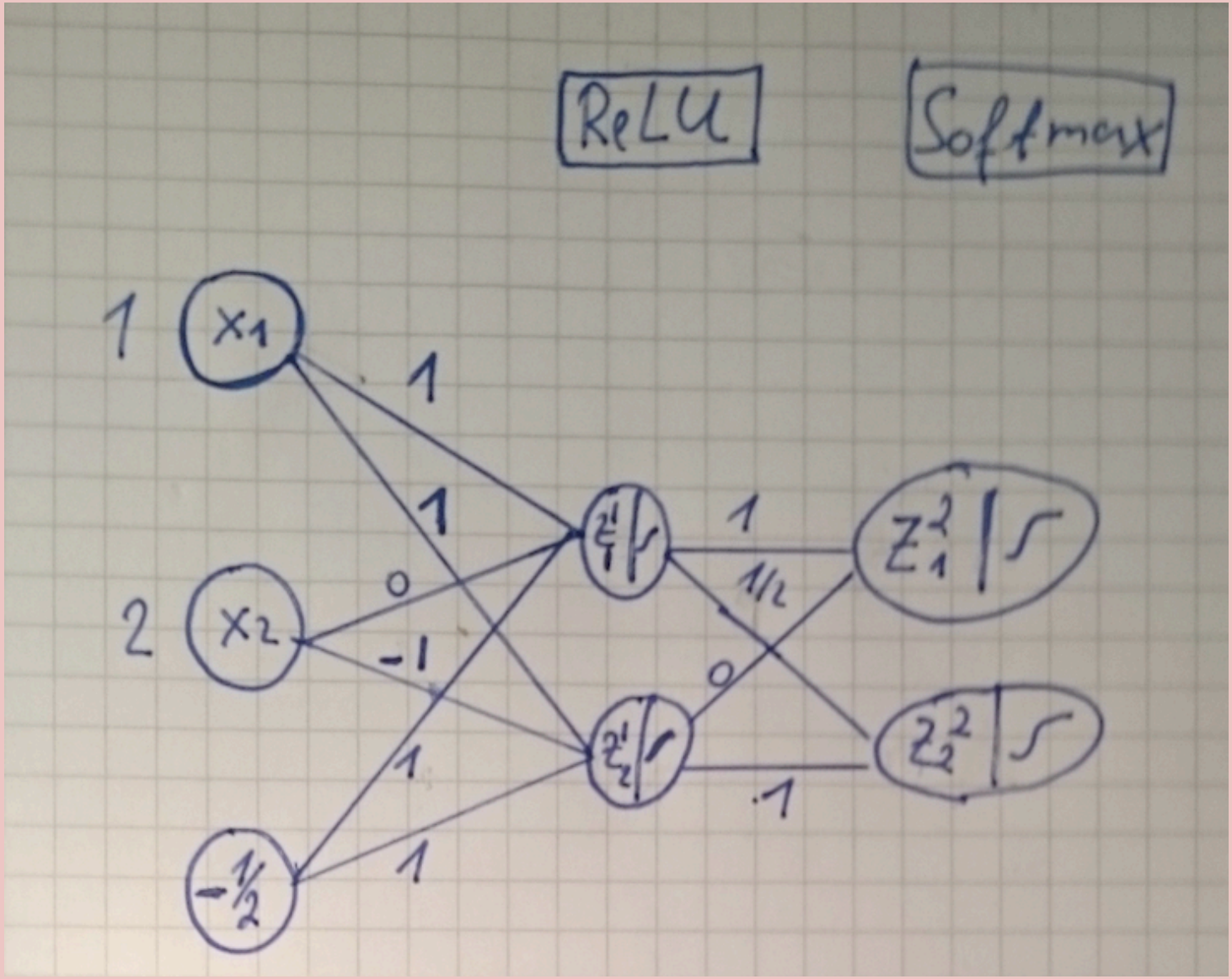


Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons

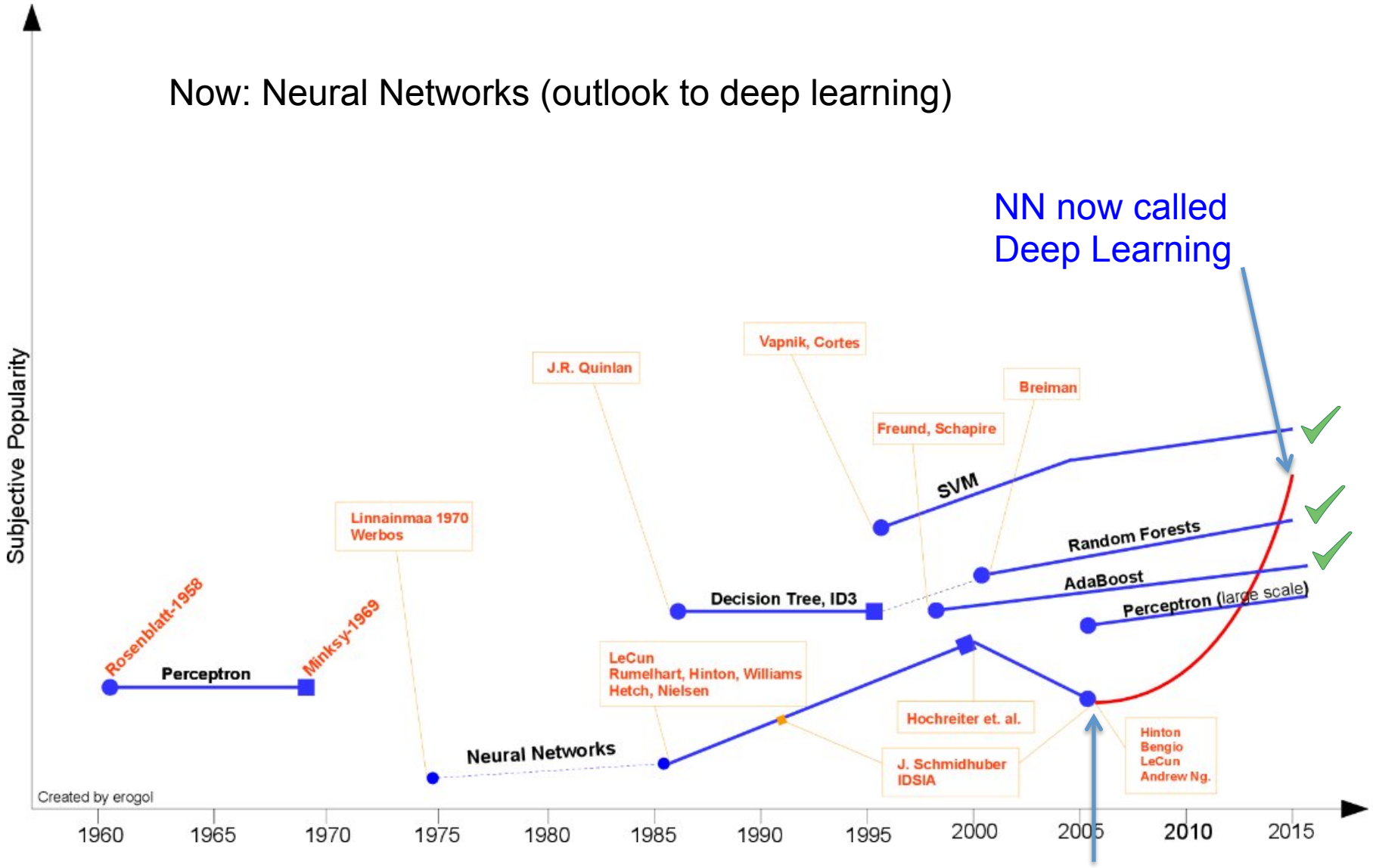
Source:
Alexnet
Krizhevsky et al 2012

Aufgabe: Rechnen von Hand



Brief History of Machine Learning (supervised learning)

Now: Neural Networks (outlook to deep learning)



Taken from http://www.erogol.com/brief-history-machine-learning/

With R 1 hidden layer ☹


WE NEED TO GO
DEEPER
memegene

What others say (NVidea Deep Learning Course)

DEEP LEARNING & AI

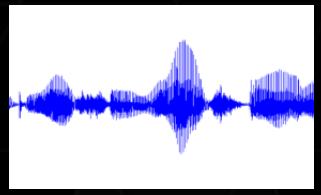
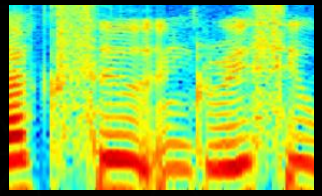
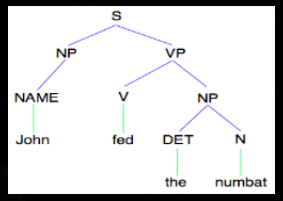
Deep Learning has become the most popular approach to developing Artificial Intelligence (AI) – machines that perceive and understand the world

The focus is currently on specific perceptual tasks, and there are many successes.

Today, some of the world's largest internet companies, as well as the foremost research institutions, are using GPUs for deep learning in research and production

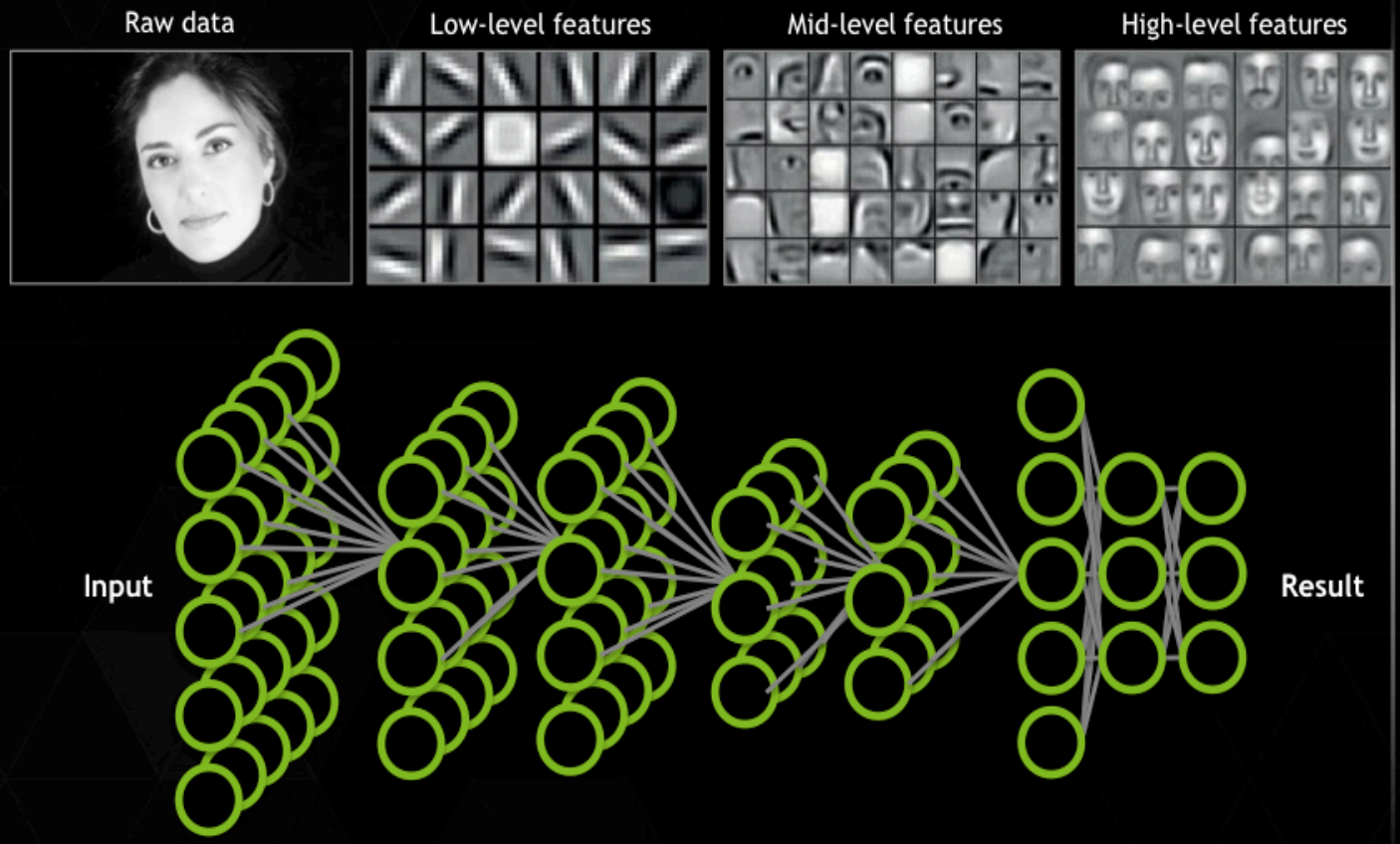
What others say (NVidea Deep Learning Course)

TRADITIONAL MACHINE PERCEPTION - HAND TUNED FEATURES

Raw data	Feature extraction	Classifier/ detector	Result
		SVM, shallow neural net, …	
		HMM, shallow neural net, …	Speaker ID, speech transcription, …
		Clustering, HMM, LDA, LSA …	Topic classification, machine translation, sentiment analysis…

What others say (NVidea Deep Learning Course)

DEEP NEURAL NETWORK (DNN)



Application components:

Task objective
- e.g. Identify face

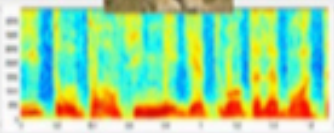
Training data
- 10-100M images

Network architecture
- ~10 layers
- 1B parameters

Learning algorithm
- ~30 Exaflops
- ~30 GPU days

Features are learned by the network!

Use cases of deep learning

Input	Output
Pixels: 	"lion"
Audio: 	"see at tuhl res taur aun ts"
<query, doc>	P(click on doc)
"Hello, how are you?"	"Bonjour, comment allez-vous?"
Pixels: 	"A close up of a small child holding a stuffed animal"

Source: NIdea course

Currently two architectures

- Recursive Neural Networks
  - Basically for Sequences (Text)

- **Convolutional Neural Networks**
  - For Images
  - (But also for text)

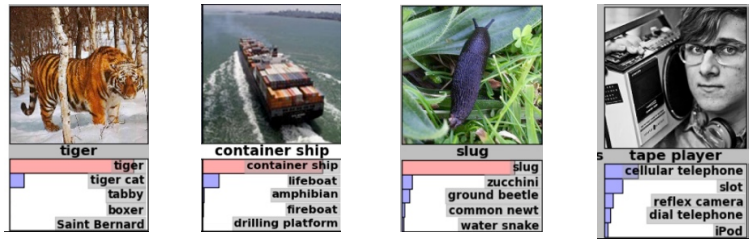
The state of the art 2012

Kaggle dog vs cat competition

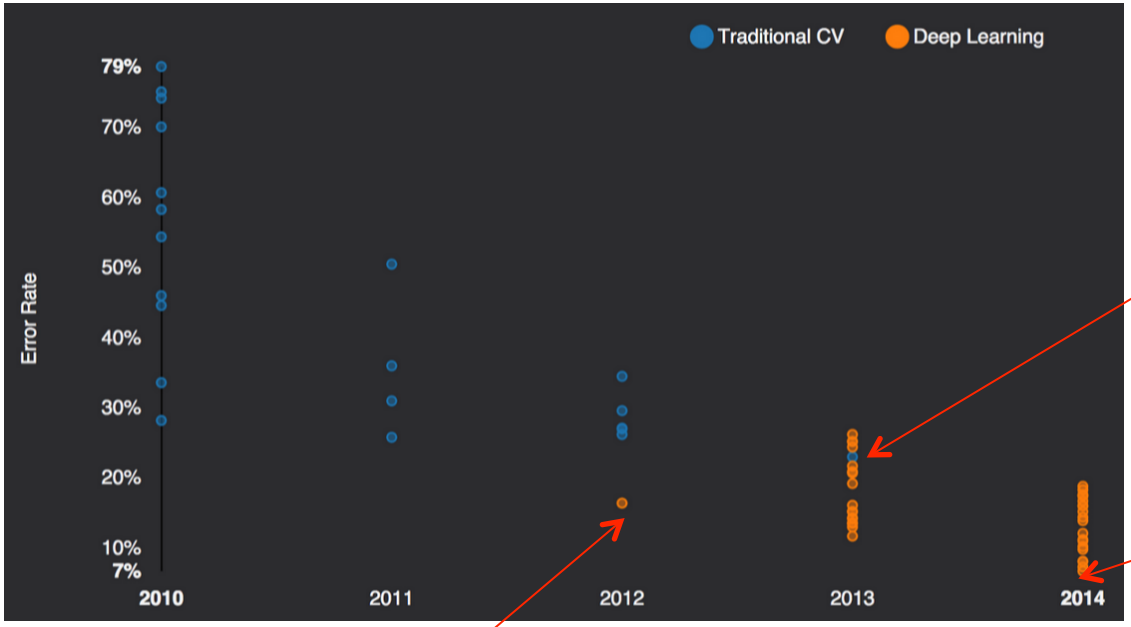
Deep Blue beat Kasparov at chess in 1997.
Watson beat the brightest trivia minds at Jeopardy in 2011.
Can you tell Fido from Mittens in 2013?

Why DL: Imagenet 2012, 2013, 2014, 2015

1000 classes
1 Mio samples



...



Human: 5% misclassification

Only one non-CNN approach in 2013

GoogLeNet 6.7%

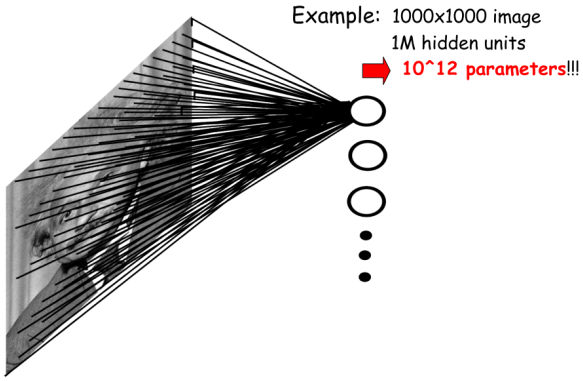
A. Krizhevsky first CNN in 2012

2015: It gets tougher
4.95% Microsoft (Feb 6 surpassing human performance 5.1%)
4.8% Google (Feb 11) -> further improved to 3.6 (Dec)?
4.58% Baidu (May 11 banned due too many submissions)
3.57% Microsoft (Resnet winner 2015)

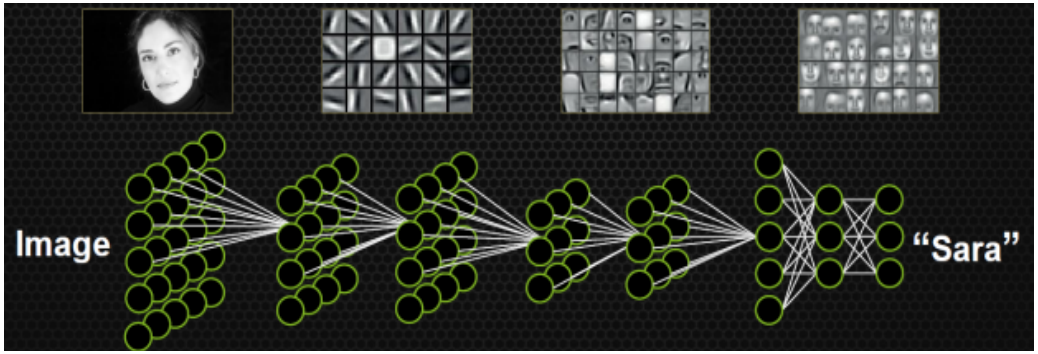
Figure: https://medium.com/global-silicon-valley/machine-learning-yesterday-today-tomorrow-3d3023c7b519

Issues MLP for Images

Have



Want



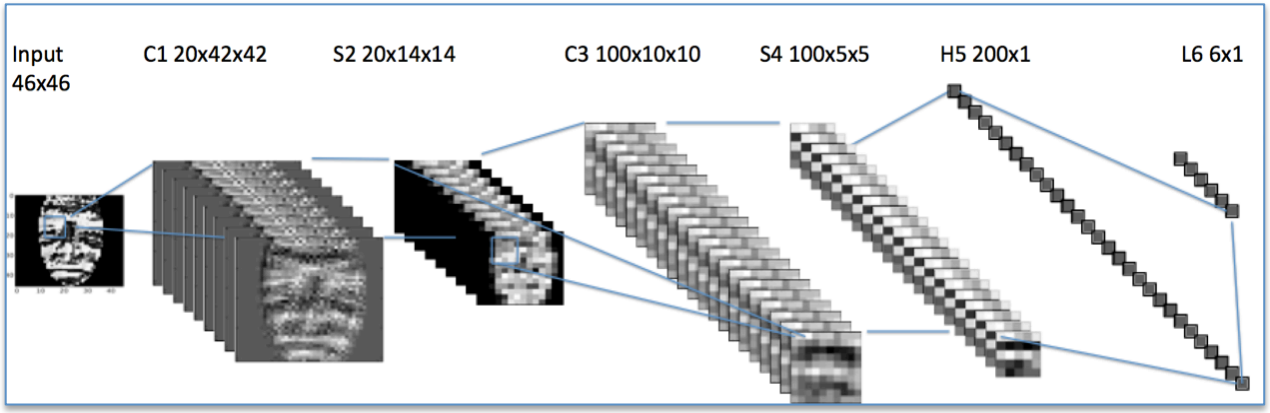
- Many weights already for one layer, but we want many layers.
- Locality of images not taken into account
- No feature learning

Remedy:

- Weight sharing → Convolution (less weights)
- Hierachically reducing size → Pooling

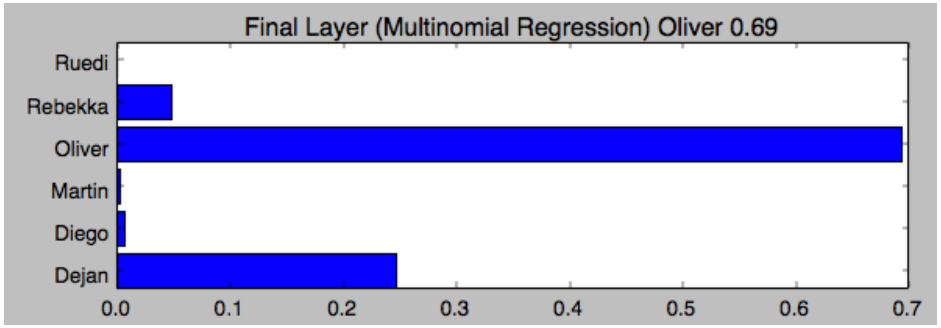
Example on a CNN

Conv Max Pool Conv Max Pool Full Con 1 Multinomial Log. Reg



Convolution & Max Pooling

'standard' neural net

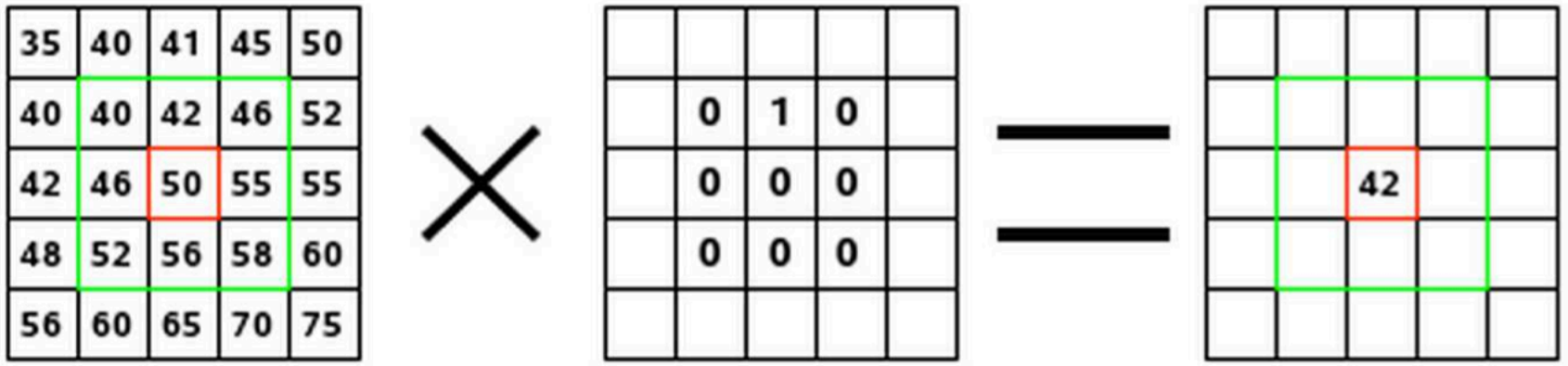


Lets look at the building blocks...

The convolutional part

Convolutional networks
Ingredient I: Convolution

What is convolution?

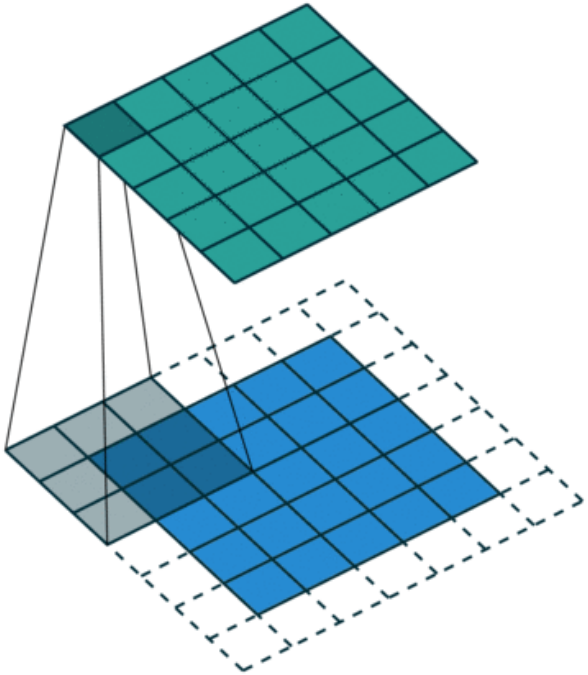


The 9 weights W_{ij} are called kernel (or filter)

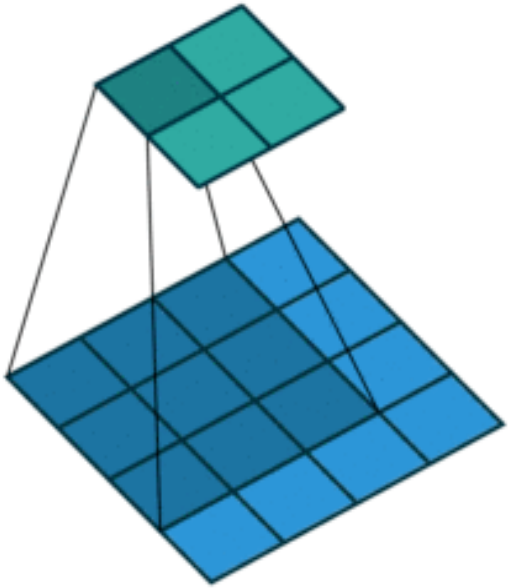
The weights are not fixed, they are learned!

Gimp documentation: http://docs.gimp.org/en/plug-in-convmatrix.html

Convolutional networks
Ingredient I: Convolution



padding



no padding

The *same* weights are slid over the image

Illustration: https://github.com/vdumoulin/conv_arithmetic

Example of designed Kernel / Filter

Edge enhance
Filter

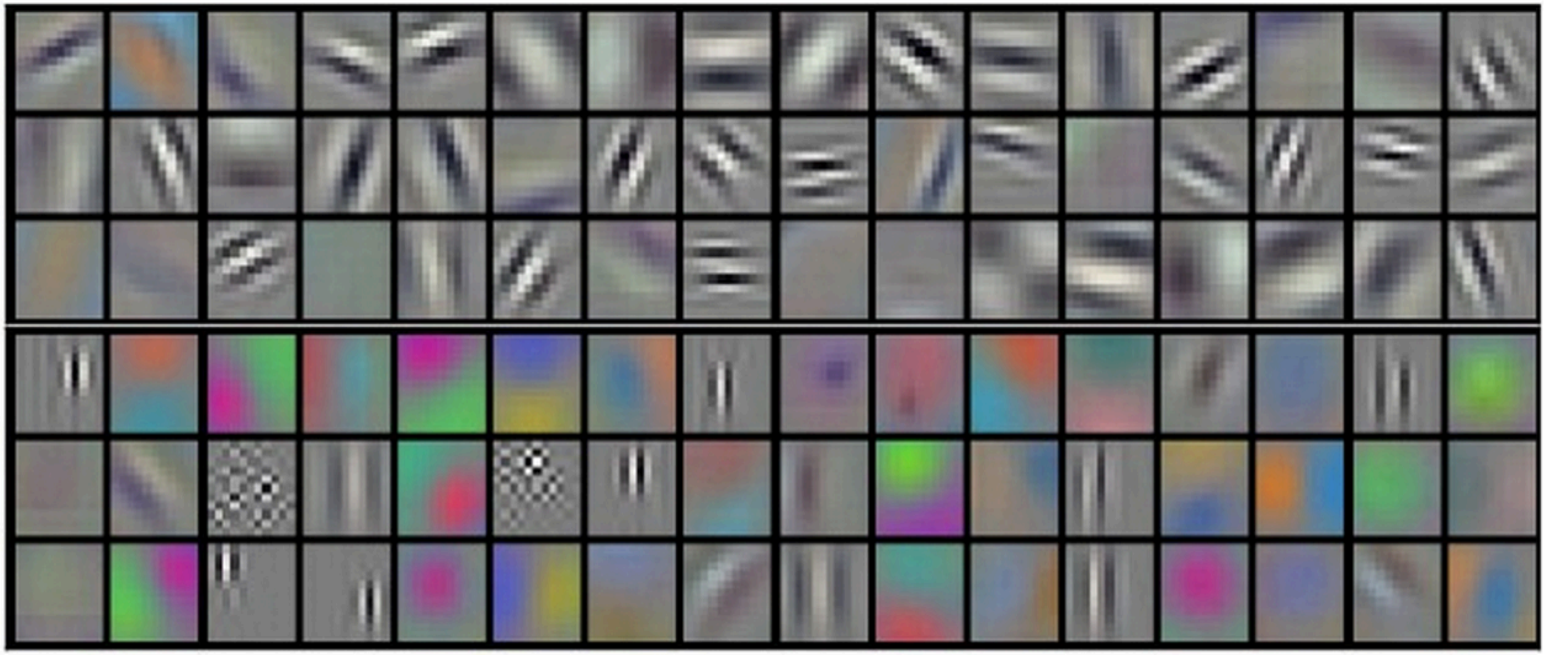
	0	0	0	
	-1	1	0	
	0	0	0	

But again!
The weights are not fixed. They are learned!

If time: http://setosa.io/ev/image-kernels/

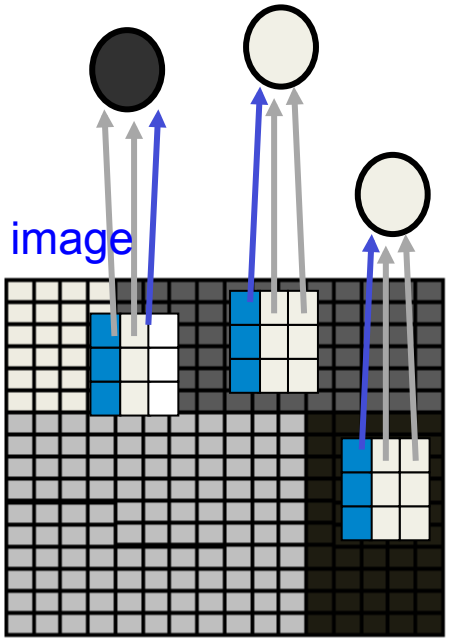
Gimp documentation: http://docs.gimp.org/en/plug-in-convmatrix.html

Example of learning filters



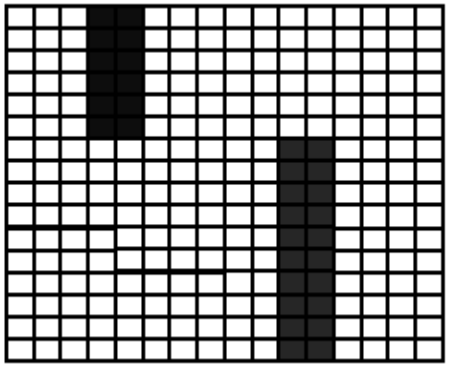
Examples of the 11x11x3 filters learned (Taken from Krizhevsky et al. 2012). Looks pretty much like old fashioned filters.

Filters are feature detectors



image

The filters are active at the position, where feature is found



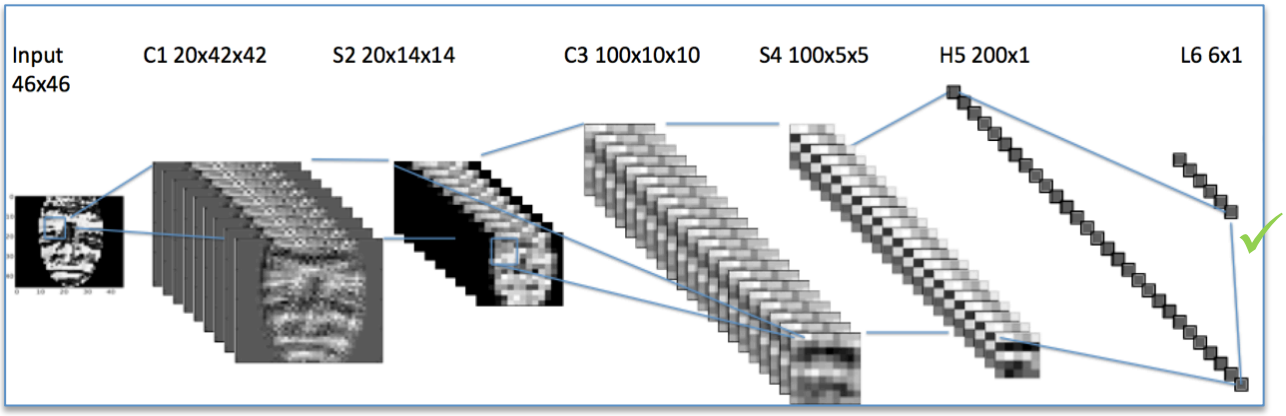
feature map

Filters are ‘active’ at a certain position of the image/

The found features a.k.a. feature map. They are kind of a new image. We can use them as a starting point to look for features again.
But first we want to reduce the image size…

Slides thanks to Beate

Maxpooling Building Block



4x4

12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

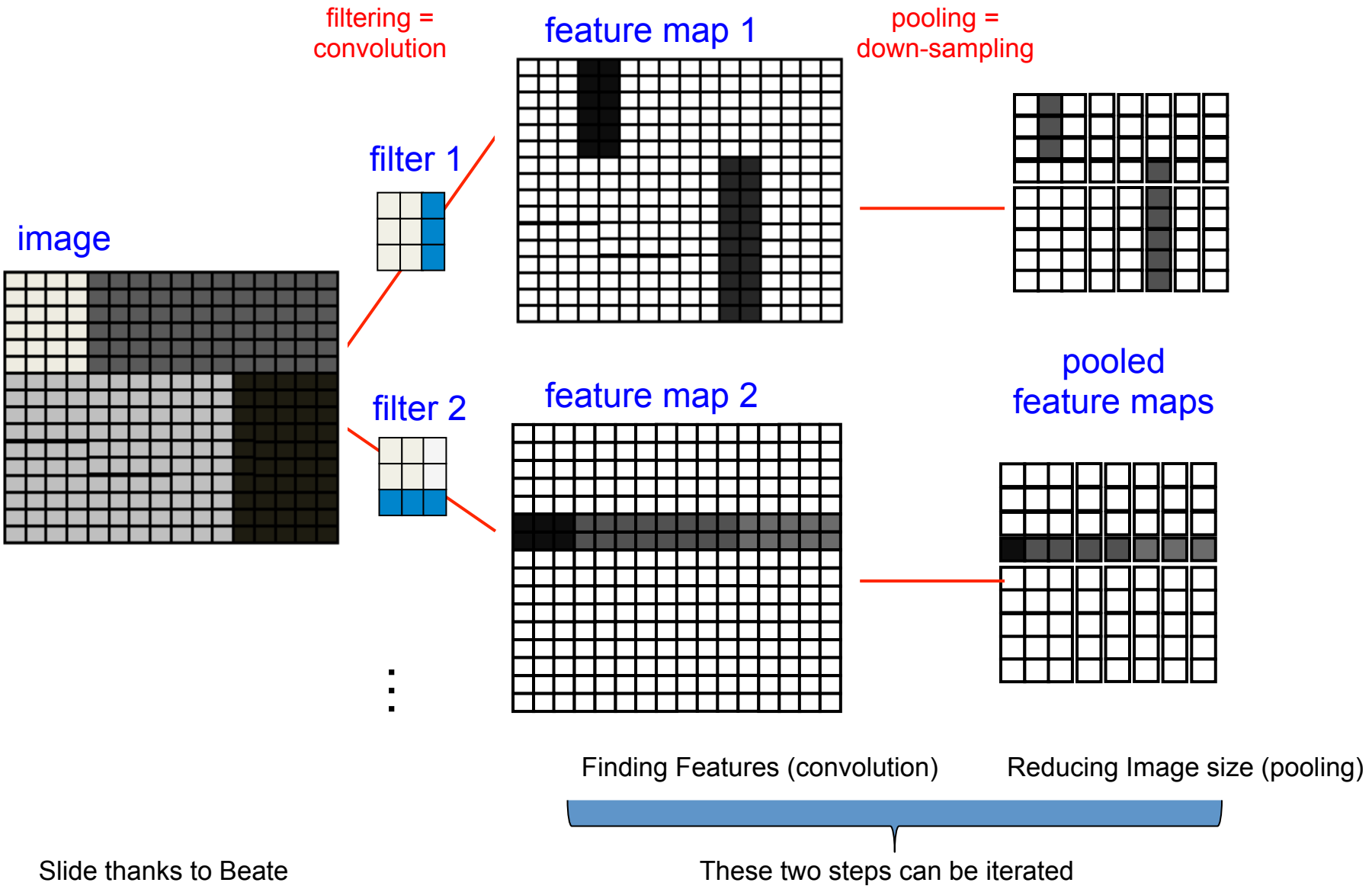
2x2

20	30
112	37

Simply join e.g. 2x2 adjacent pixels in one.

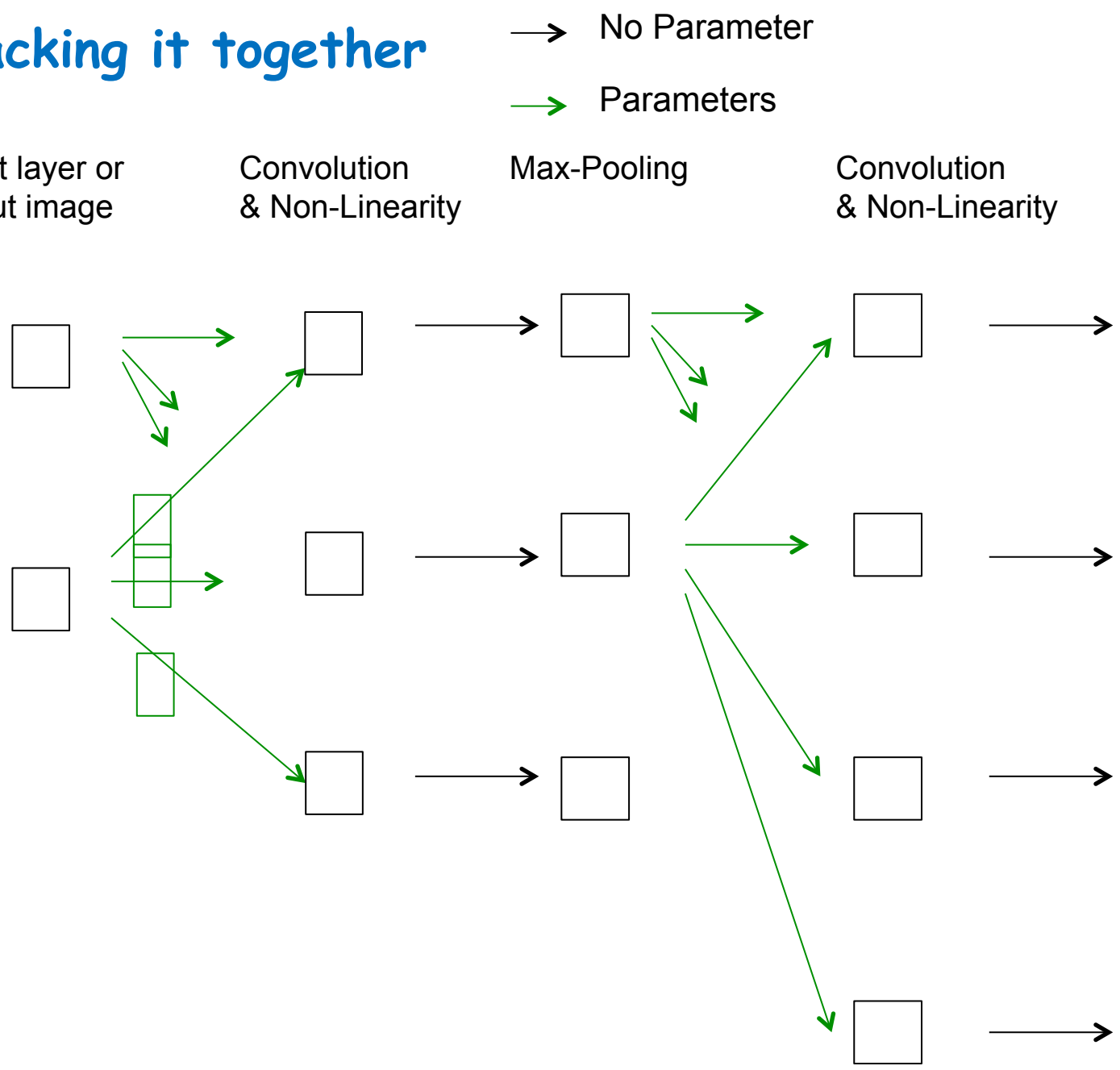
Hinton: „The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster“

Propagating the features down



Slide thanks to Beate

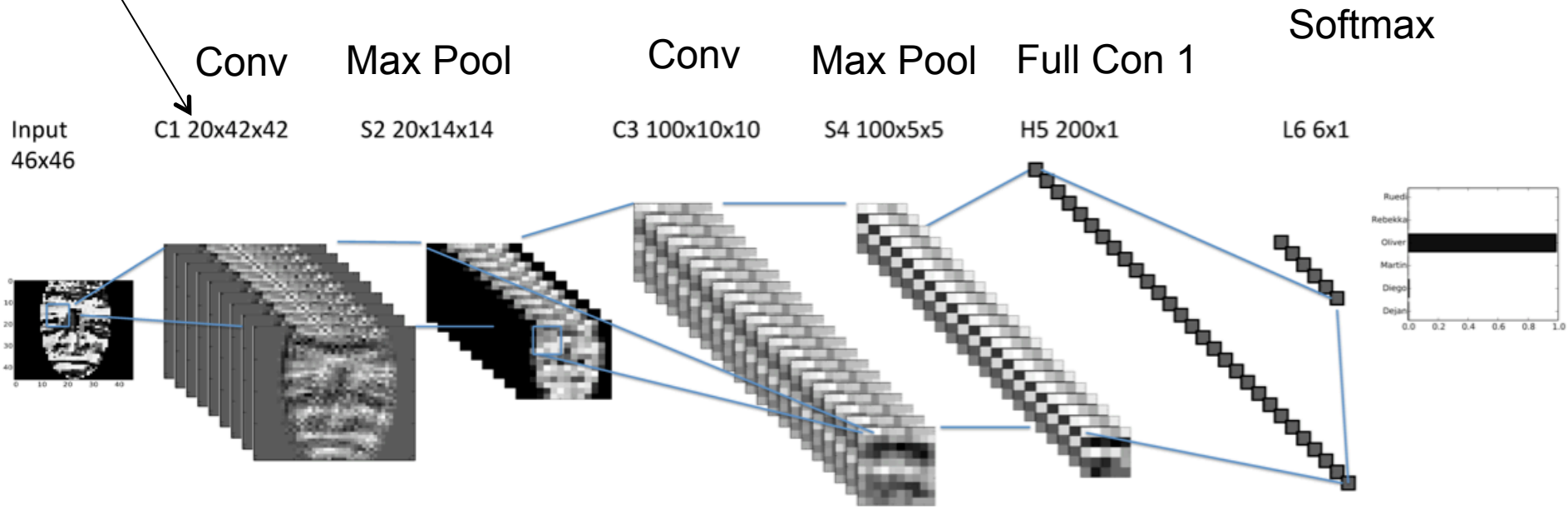
Stacking it together



A simple version of the CNN (LeNet5 Architecture)

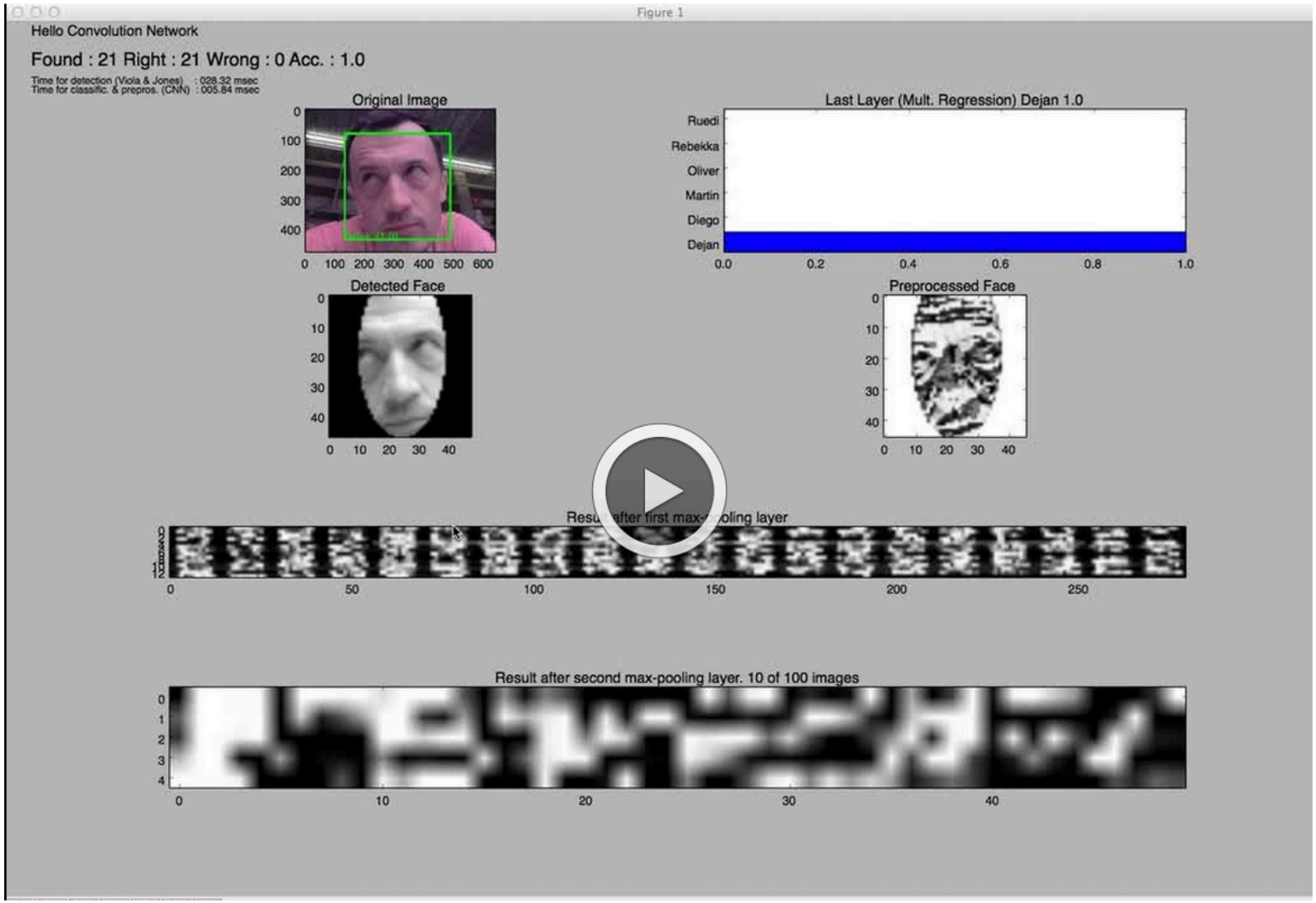
20 Kernels a 5x5 weights produce a depth of 20

100 Kernels each applied to



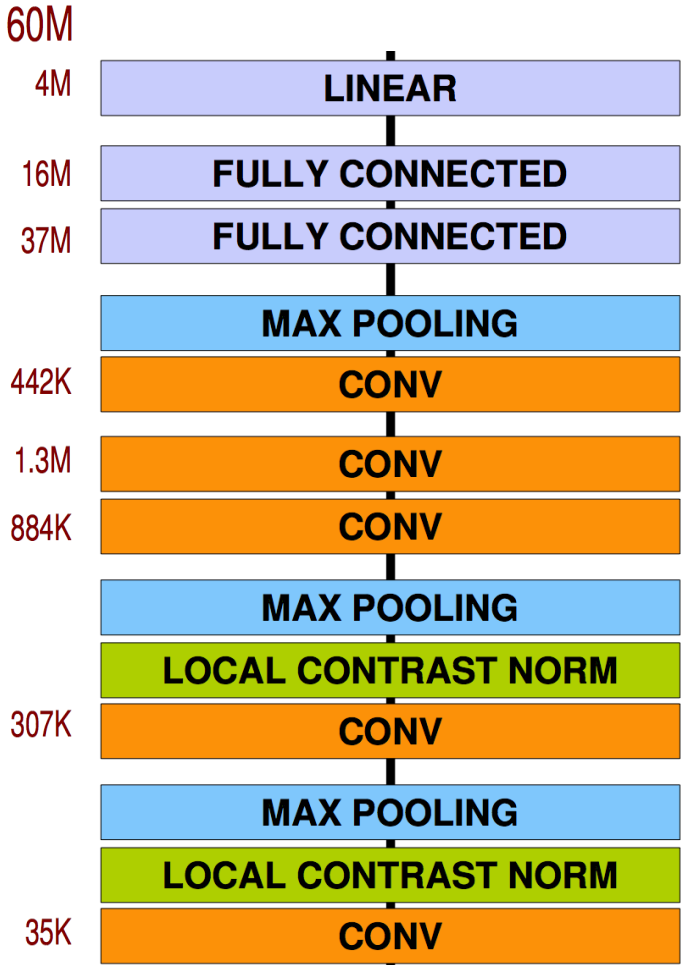
O. Dürr, R. Axthelm, M. Loeser et al., "Deep Learning Algorithms for Efficient Face Recognition on a Raspberry Pi", *Eurographics Conference 2015*

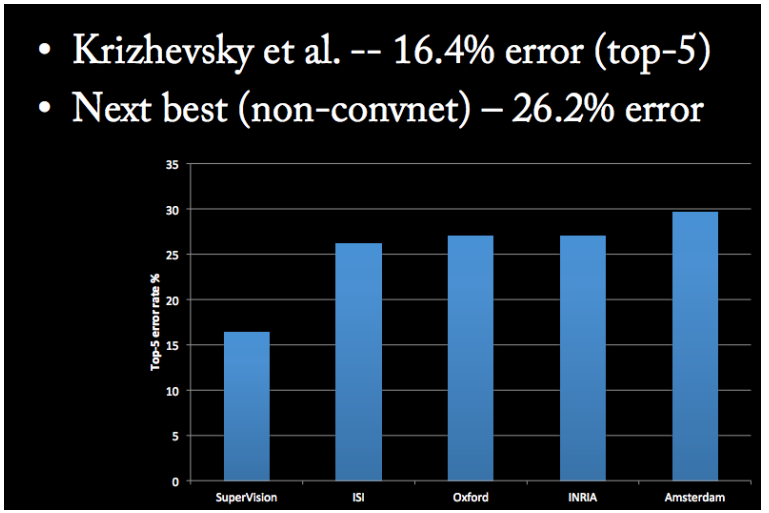
Let's watch a movie



https://www.youtube.com/watch?v=ol1eJa-UWNU&feature=youtu.be (please turn off sou

A typical recent architecture (AlexNet, 2012)



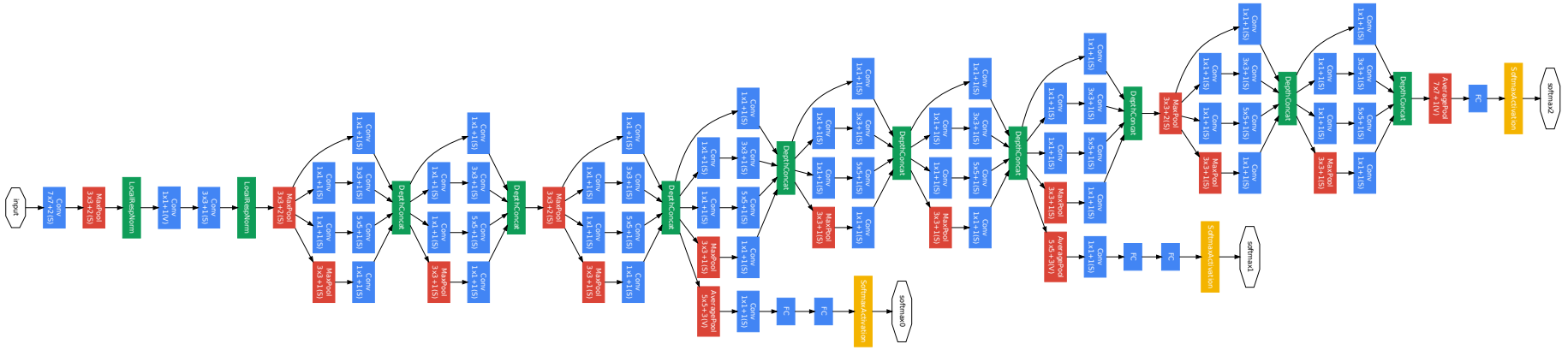


Seminal paper. 26.2% error ➔16.5%

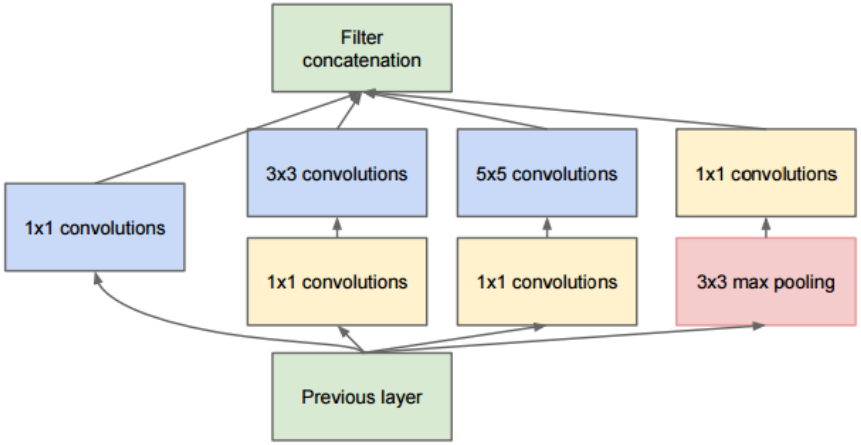
- Dropout (see below)
- ReLU instead of sigmoid
- Parallelisation on many GPUs
- Local Response Normalization (not used anymore)

A bit of a simplification, since Alex Net is build for 2 GPUs and normalization. Caffe code from here

Winning architecture (GoogLeNet, 2014)



The inception module (convolutions and maxpooling)



Few parameters, hard to train.
Comments see here

Going deeper with convolution http://arxiv.org/abs/1409.4842

A typical very recent architecture („Oxford Net"(s), 2014)

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

- More traditional, easier to train
- More weights than GoogLeNet
- Small pooling
- More than 1 conv. before maxpooling.
- No strides (stride 1)
- ReLU after conv. and FC
- Caffe implementation and pre-trainined model

http://arxiv.org/abs/1409.1556

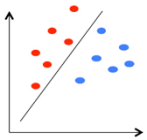
ImageNet Challenge 2014, 2nd classificatio

Further links to deep learning

- Overview Paper on DL: http://www.nature.com/nature/journal/v521/n7553/full/nature14539.html also part of the material
- Course with focus on vision http://cs231n.stanford.edu/
- Slides from the CAS: https://dl.dropboxusercontent.com/u/9154523/talks/Deep_Learning_CAS.pdf

Overview of classification

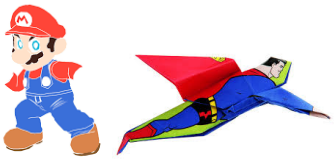
Classifiers



K-Nearest-Neighbors (KNN)
Classification Trees,
Logistic Regression
Linear discriminant analysis
Support Vector Machine (SVM)
Neural networks NN
[Convolution Neural Networks]
…

Evaluation

Performance measures
Crossvalidation
ROC Analysis
Lift Charts

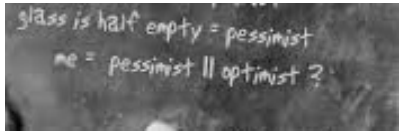


Combining classifiers

Bagging
Random Forest
Boosting

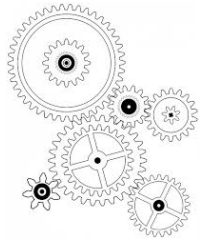
Theoretical Guidance / General Ideas

Bayes Classifier
Bias Variance Trade off (Overfitting)



Feature Engineering

Feature Transformations
Feature Selection



Comparison of different classifiers

- K-Nearest-Neighbors (KNN)
  - Robust
- Classification Trees
  - Robust can, handle categorical data, not so performant
- Logistic Regression
  - Need scaling
- Linear discriminant analysis
  - Need Scaling, Good for many classes
- Support Vector Machine (SVM)
  - Need scaling, often quite perfomant
- Neural networks NN
  - Need scaling
- Convolution Neural Networks
  - Not in R, long training times
- Bagging
  - Seldom used for non trees, for trees: better used RF
- Random Forest
  - Robust hard to implement
- Boosting
  - Sometimes quite performat, hard to understand