A Shallow Introduction into the Deep Machine Learning

Jan Čech

What is the “Deep Learning” ?

- Deep learning (by G. Hinton, DL pioneer, Turing Award 2018 holder)
  = both the classifiers and the features are learned automatically

image $\xrightarrow{\text{classifier}}$ label

- Typically not feasible, due to high dimensionality

image $\xrightarrow{\text{hand-engineering}}$ features $\xrightarrow{\text{classifier}}$ label

(e.g. SIFT, SURF, HOG, or MFCC in audio)

- Suboptimal, requires expert knowledge, works in specific domain only

image $\xrightarrow{\text{learning}}$ features $\xrightarrow{\text{classifier}}$ label

(feature hierarchies)

Deep neural network

What is the “Deep Learning” ? Other definitions…

- Andrew Ng (founder of Google Brain, chief of Baidu AI research)
  - “**Very large neural networks** we can now have and … huge amounts of data that we have access to.”
- Jeff Dean (head of Google AI)
  - “When you hear the term deep learning, just think of a **large deep neural net**. **Deep refers to the number of layers** typically and so this kind of the popular term that’s been adopted in the press. I think of them as deep neural networks generally.”
- Yoshua Bengio (DL pioneer, Turing Award Holder 2018)
  - “Deep learning algorithms seek to exploit the unknown **structure** in the input distribution in order to **discover good representations**, **often at multiple levels**, with higher-level learned features defined in terms of lower-level features.”
- Yann LeCun (DL pioneer, Turing Award Holder 2018)
  - “Deep learning [is] … **a pipeline of modules all of which are trainable**. … deep because [has] multiple stages in the process of recognizing an object and all of those stages are part of the training.”

Deep Learning omnipresent

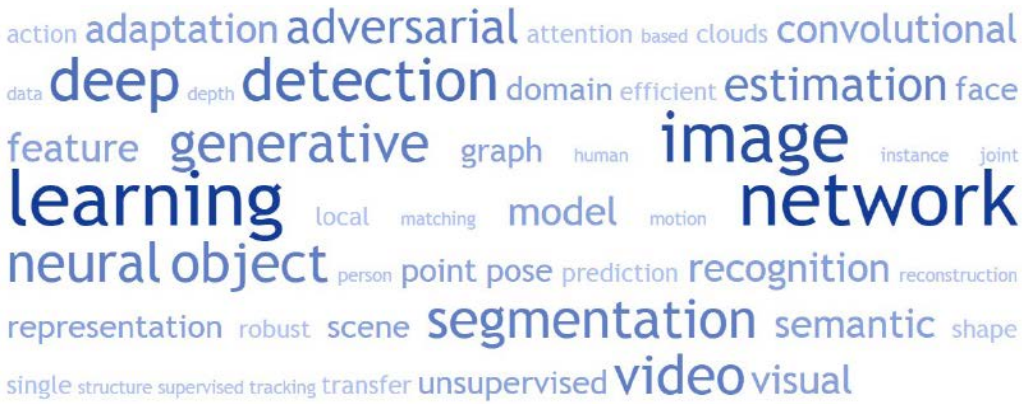
- Besides the Computer Vision DL is extremely successful in, e.g.
  - Automatic Speech Recognition
    - Speech to text, Speaker recognition
  - Natural Language Processing (LLMs)
    - Machine translation, Question answering, Chatbots (**ChatGPT**)
  - Robotics / Autonomous driving
    - Reinforcement learning
  - Data Science / Bioinformatics (e.g., Alphafold)

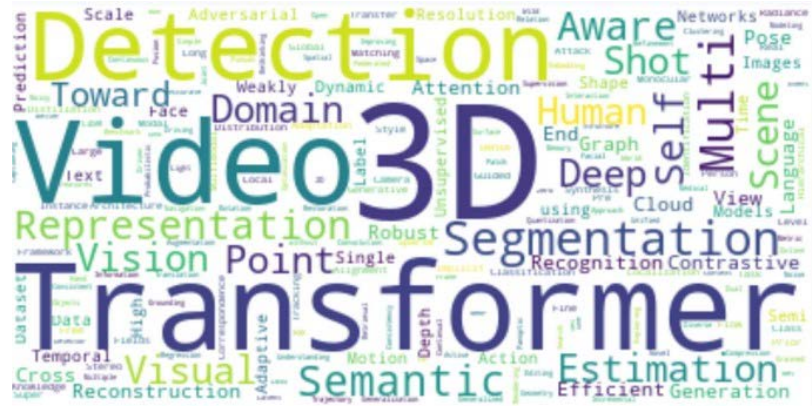
- Shift of paradigm in Computer Vision
  - Large-scale image category recognition (ILSVRC' 2012 challenge)

INRIA/Xerox	33%,
Uni Amsterdam	30%,
Uni Oxford	27%,
Uni Tokyo	26%,
Uni Toronto	**16% (deep neural network)** [Krizhevsky-NIPS-2012]

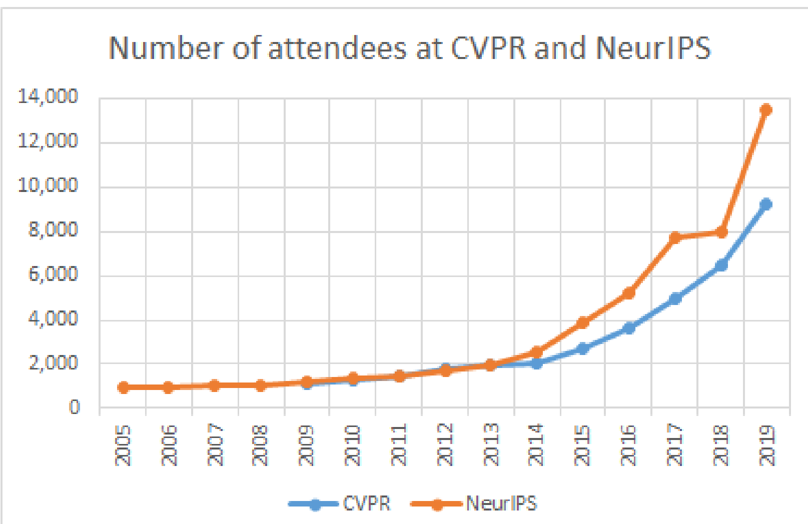
Explosion of interest in “Deep Learning” after 2012

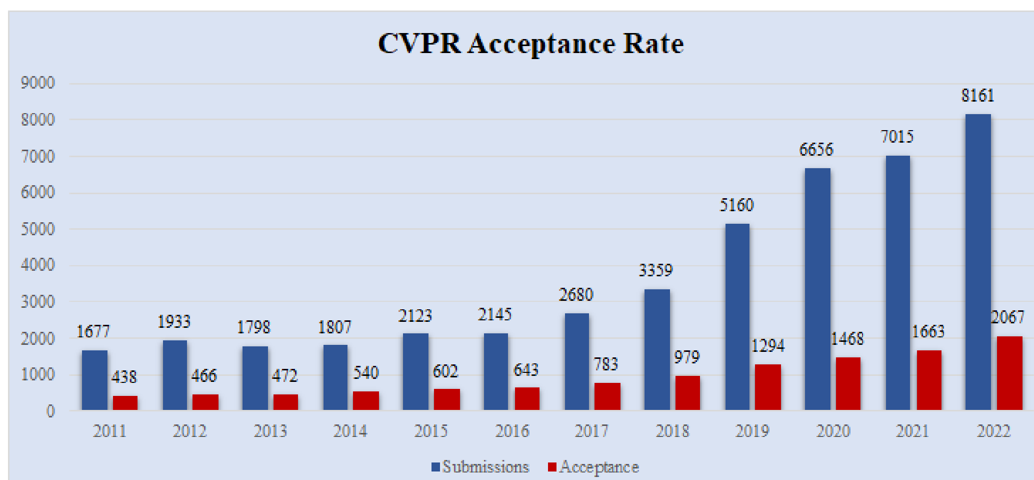
- Paper title keywords, CVPR 2019/2022





- Number of attendees/submissions in major Computer Vision and Machine Learning grows exponentially





Data Source: https://hai.stanford.edu/, https://github.com/BIGBALLON/CVPR2022-Paper-Statistics

Examples of Deep learning in Computer Vision

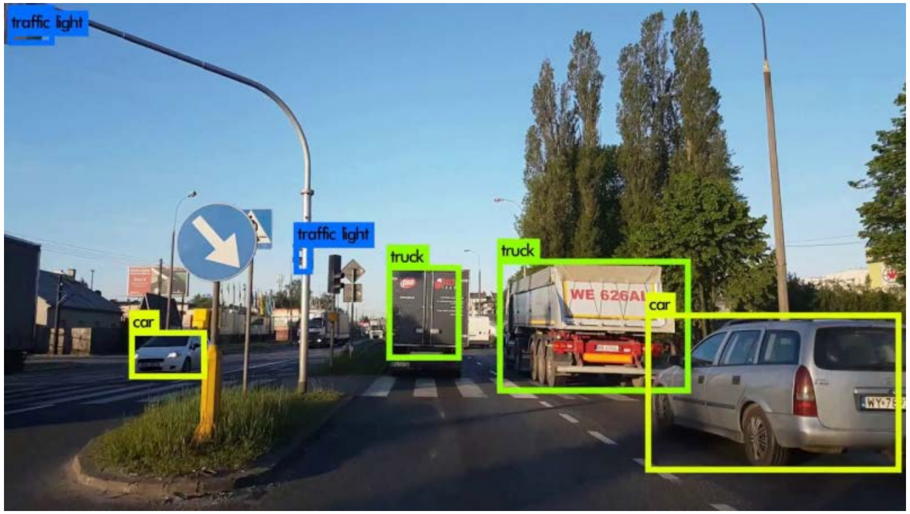
- Image classification [Krizhevsky-NIPS-2012]
  - Input: RGB-image
  - Output: Single label (Probability Distribution over Classes)

IMAGENET

  - ImageNet dataset (14M images, 21k classes, Labels by Amazon Mechanical Turk)
  - ImageNet Benchmark (1000 classes, 1M training images)

Examples of Deep learning in Computer Vision

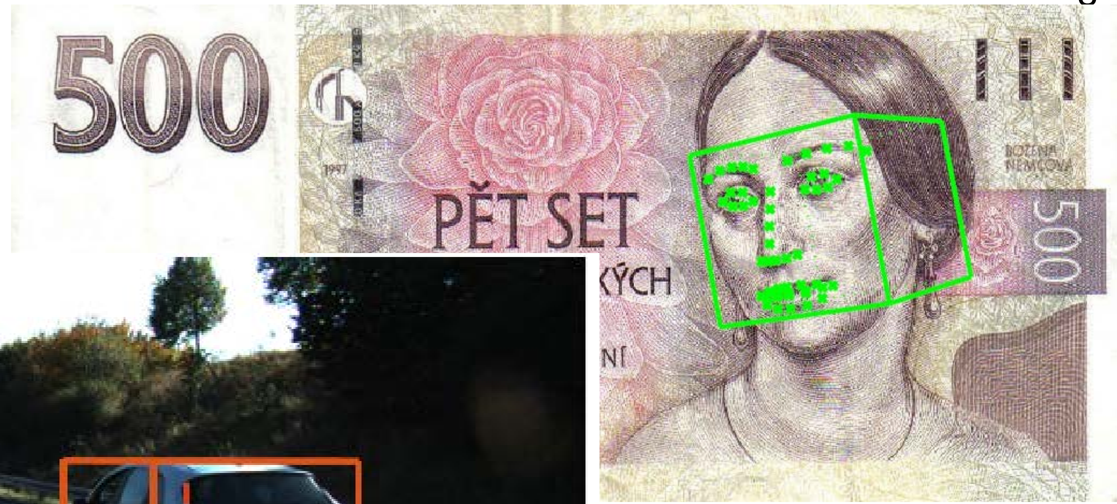
- Object Detection
  - Multiple objects in the image [RCNN, YOLO, …]

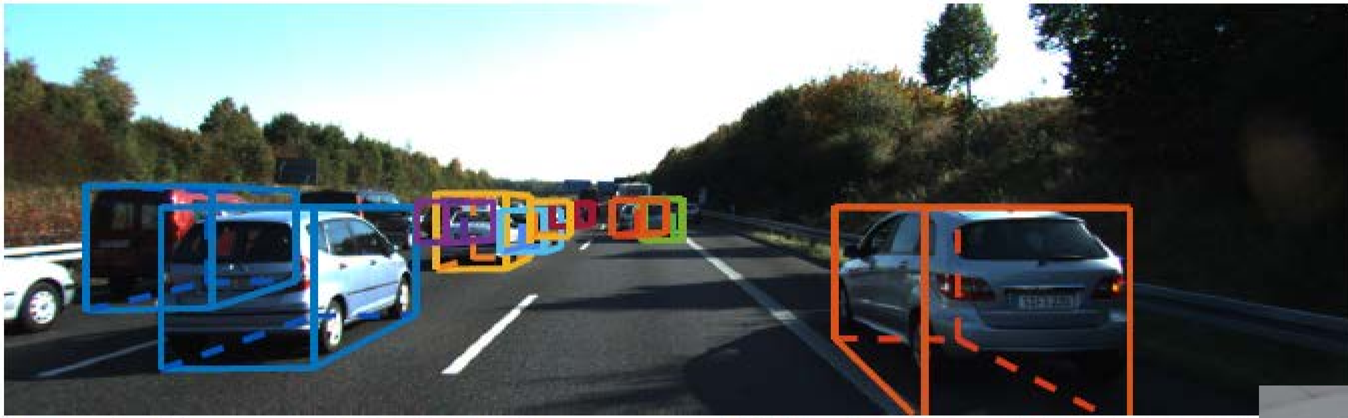


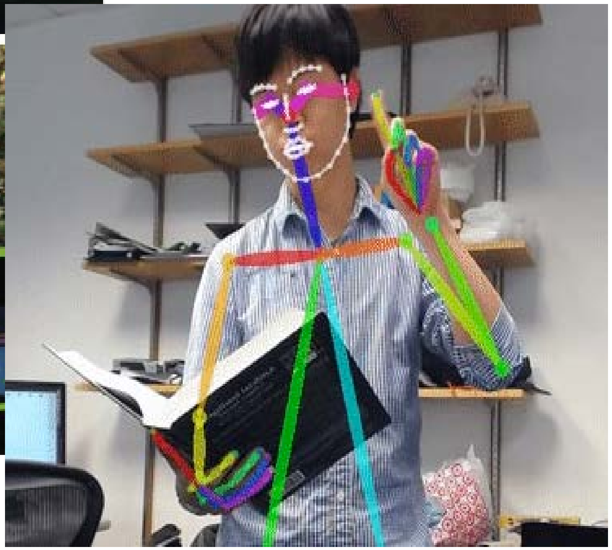
  - E.g. Face [Hu-Ramanan-2017], Text localization [Busta-2017]

Examples of Deep learning in Computer Vision

- (3D) Pose estimation
  - [Hu-2018], [OpenPose]

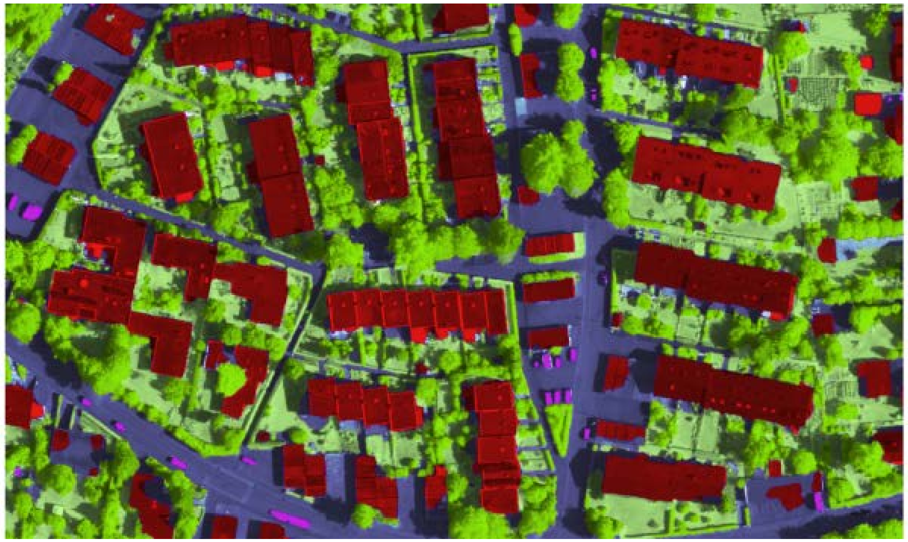






Examples of Deep learning in Computer Vision

- Image Segmentation (Semantic/Instance Segmentation)
  - Each pixel has a label [Long-2015], [Mask-RCNN-2017]

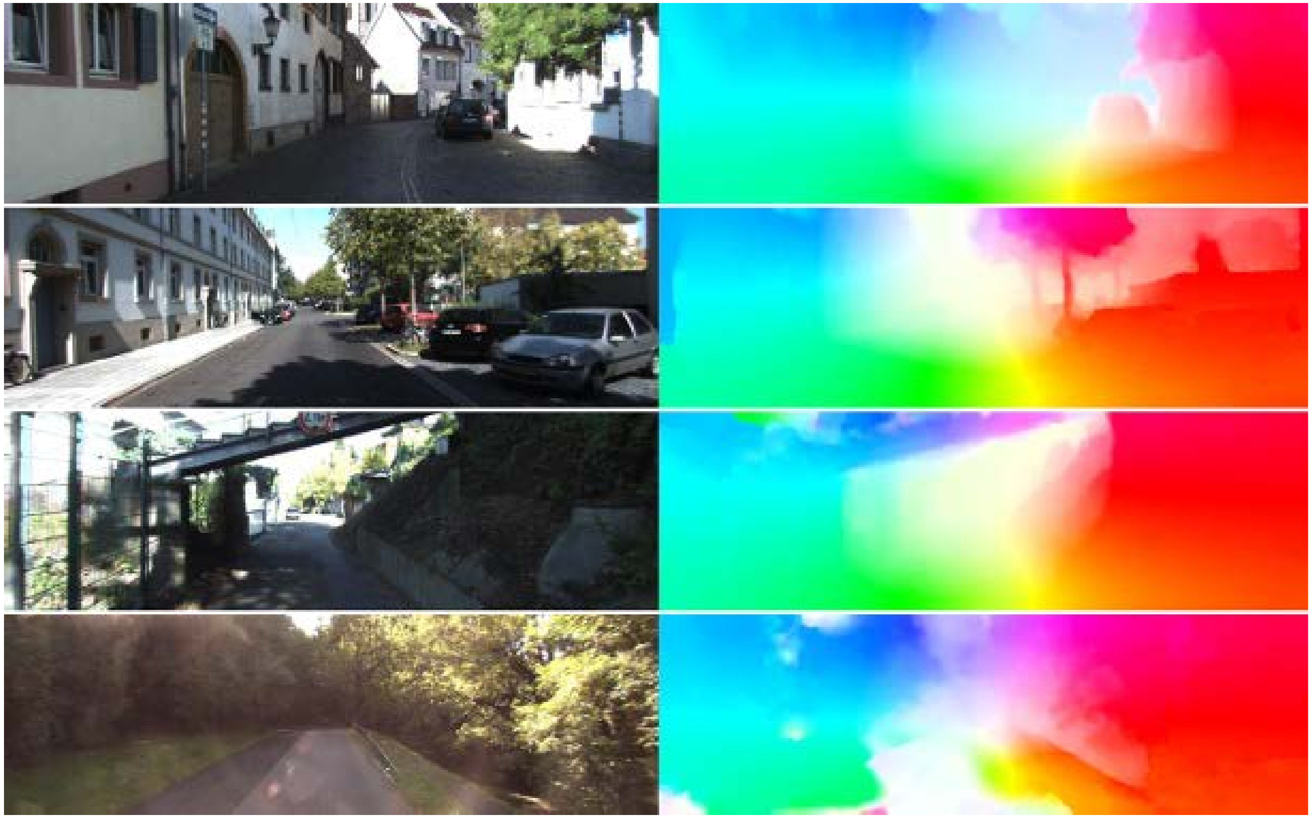


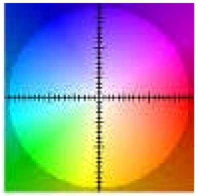
Semantic segmentation

Instance segmentation

Examples of Deep learning in Computer Vision

- Motion
  - Tracking
  - Optical Flow [Neoral-2018]
    - Predict pixel level displacements between consecutive frames





Examples of Deep learning in Computer Vision

- Stereo (depth from two images)
- Depth from a single (monocular) image [Godard-2017]

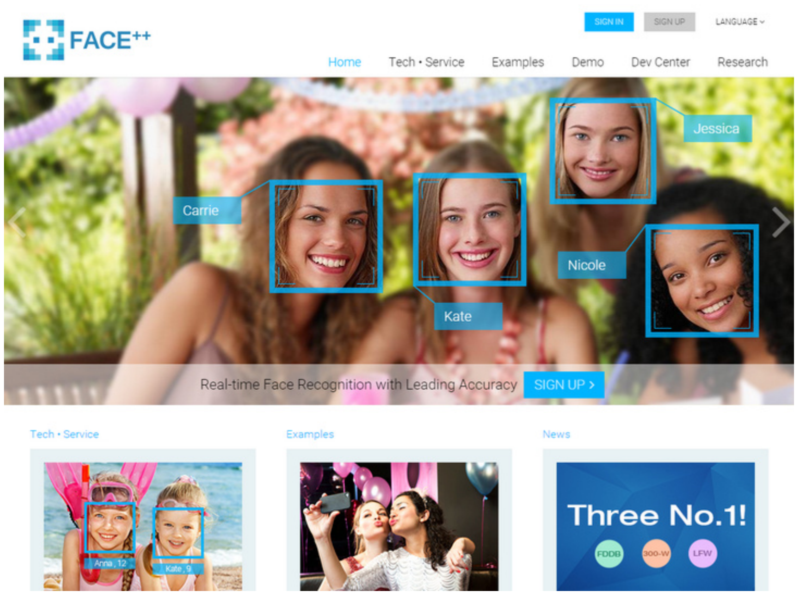
Examples of Deep learning in Computer Vision

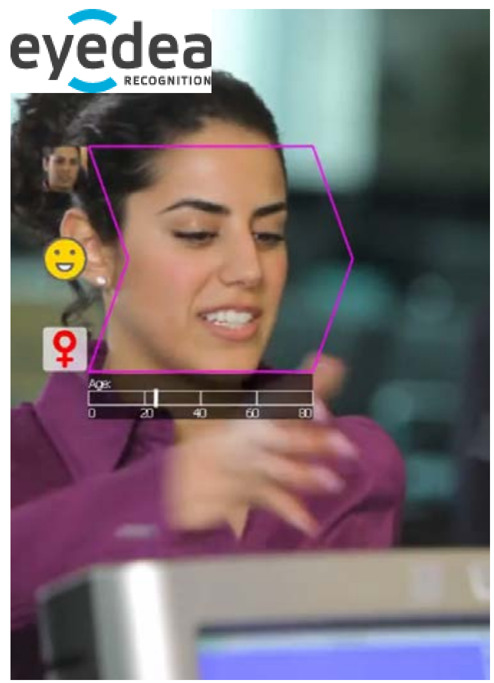
- Image based novel view synthesis
  - Given: a set of sparse images => arbitrary view (smooth camera path)
  - NeRF (Neural Radiance Field for View Synthesis), [Mildenhall-2020]

Examples of Deep learning in Computer Vision

- Faces
  - Recognition / Verification
  - Gender/Age
  - Landmarks, pose
  - Expression, emotions

…already in commerce





Examples of Deep learning in Computer Vision

- Lip reading [Chung-2017]

[YouTube]

Examples of Deep learning in Computer Vision

- Image-to-Image translation [Isola-2017]

Day to Night

input output

BW to Color

input output

- Deblurring, Super-resolution [Šubrtová-2018]

16x16 256x256 (predicted) 256x256 (ground-truth)

Examples of Deep learning in Computer Vision

- Generative models
  - Generating photo-realistic samples from image distributions
  - Variational Autoencoders, GANs [Nvidia-GAN]

(Images synthetized by a random sampling)

Examples of Deep learning in Computer Vision

- Generative models (cont.)
  - Large text2image models, 2022+ (DALL-E2, Imagen, Midjourney, Stable Diffusion – open source, model available)

panda mad scientist mixing sparkling chemicals, artstation

a propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese

Examples of Deep learning in Computer Vision

- Real image manipulation / editing
  - Instruct Pix2Pix (textual image manipulation) [Brooks-2023]

Input | "Apply face paint" | "What would she look like as a bearded man?" | "Put on a pair of sunglasses" | "She should look 100 years old"

  - Hairstyle Transfer [Šubrtová-2021]

Examples of Deep learning in Computer Vision

- Action/Activity recognition
- Neural Style Transfer
- Image Captioning/Visual Question Answering
- and many more…

[deepart.io]

[BLIP]

a brown dog wearing glasses while sitting at a desk

User What is unusual about this image?

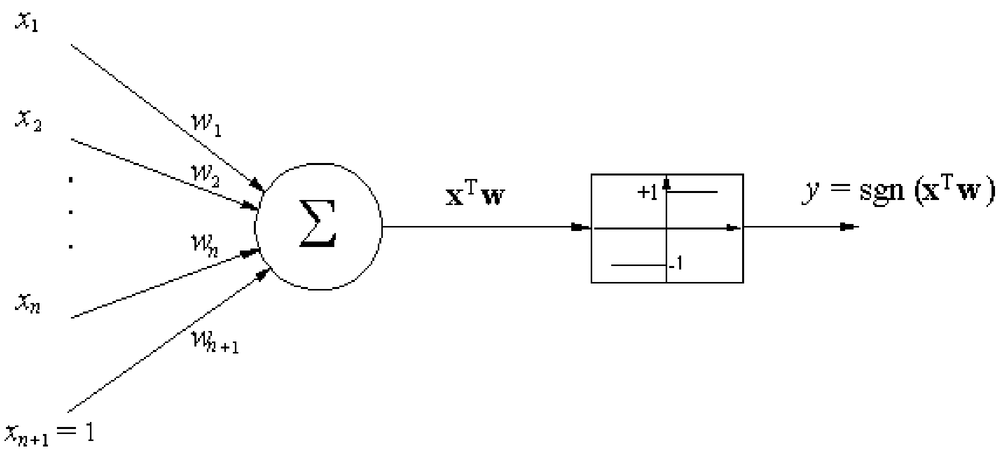
[GPT-4]

Source: https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg

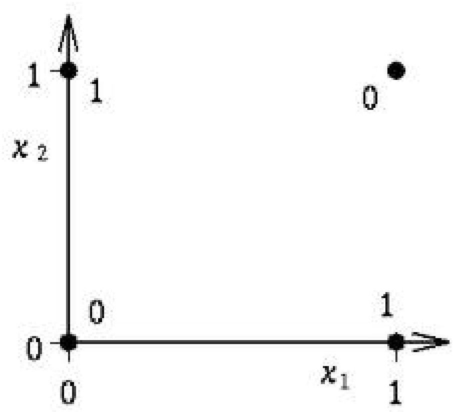
GPT-4 The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

History: (Artificial) Neural Networks

- Neural networks are here for more than 50 years
  - Rosenblatt-1956 (perceptron)



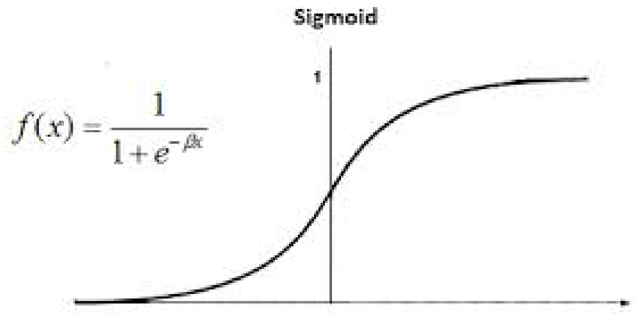
  - Minsky-1969 (xor issue, => skepticism)



History: Neural Networks

Rumelhart and McClelland – 1986:

– Multi-layer perceptron,
– Back-propagation (supervised training)
  • Differentiable activation function
  • Stochastic gradient descent

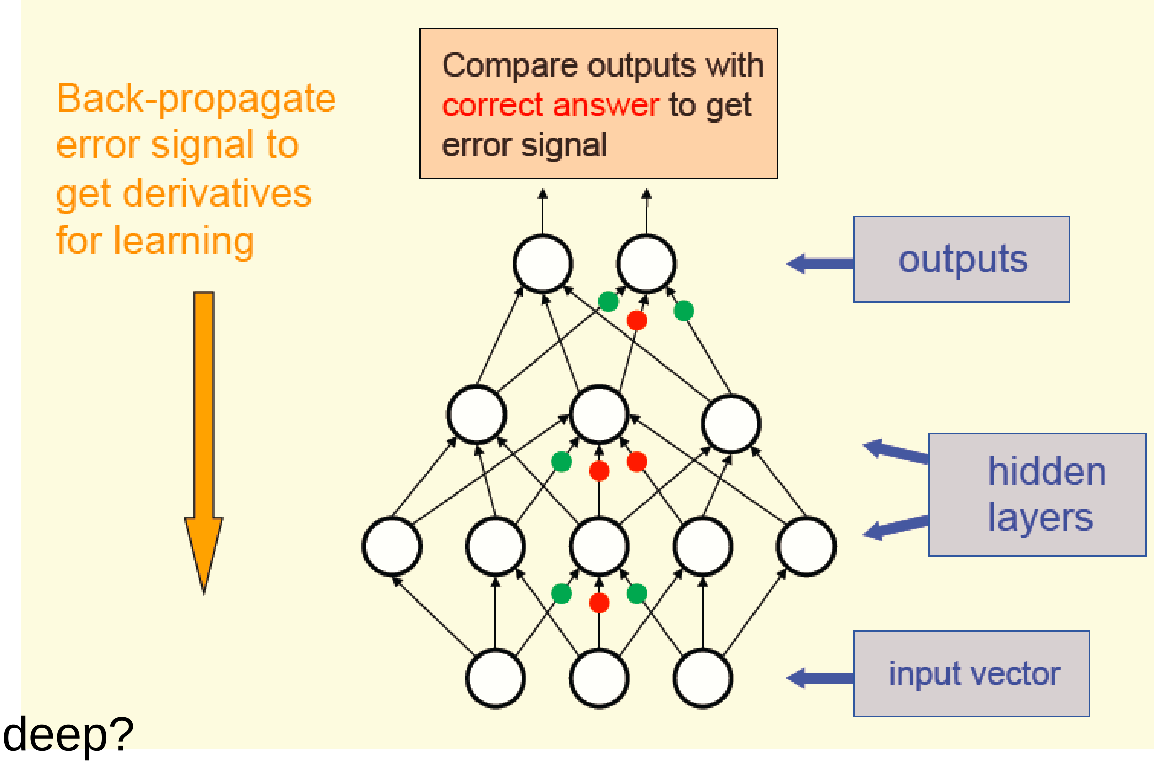


Empirical risk

$$Q(w) = \sum_{i=1}^{n} Q_i(w),$$

Update weights:

$$w := w - \alpha \nabla Q_i(w).$$



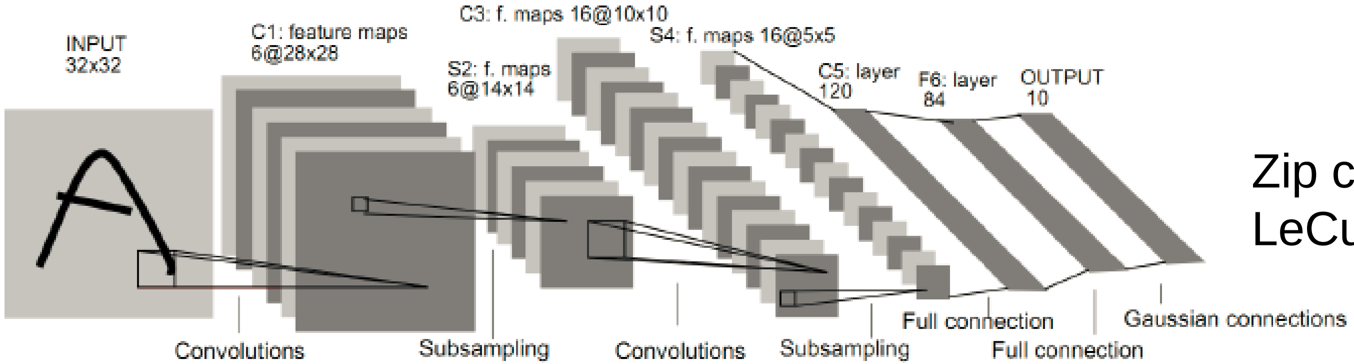
What happens if a network is deep?
(it has many layers)

What was wrong with back propagation?

- Local optimization only (needs a good initialization, or re-initialization)
- Prone to over-fitting
  - too many parameters to estimate
  - too few labeled examples
- Computationally intensive

=> Skepticism: A deep network often performed worse than a shallow one

Why does it work now?

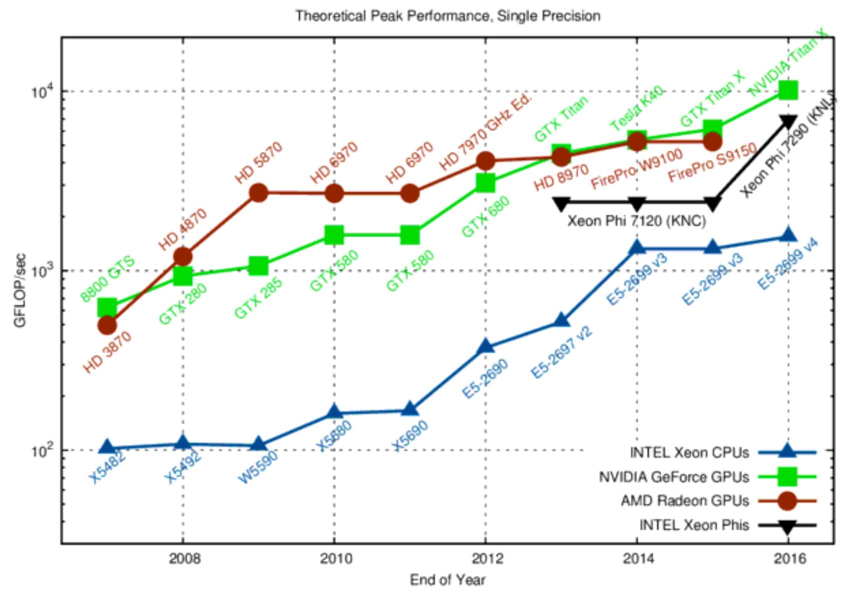


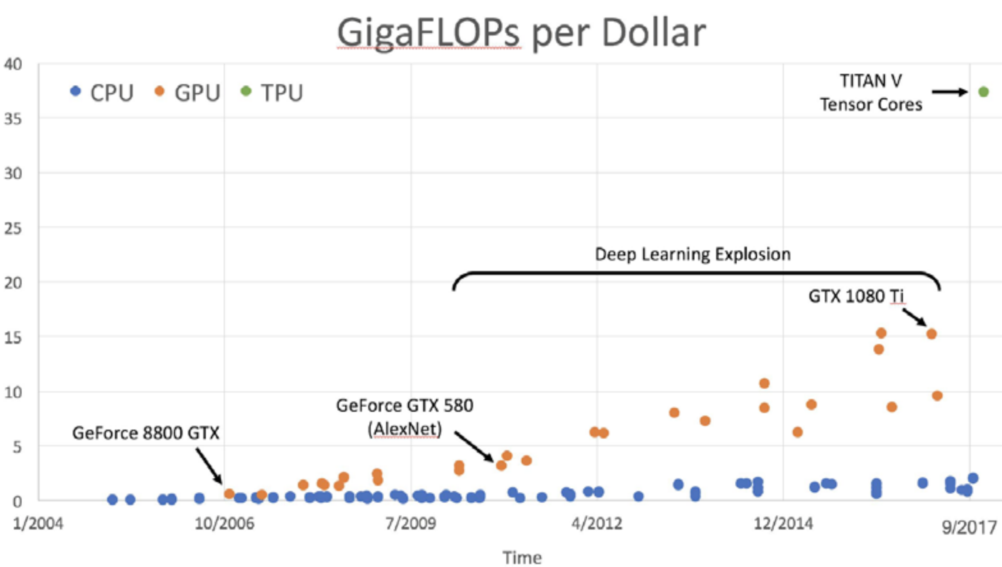
Zip codes recognition,
LeCun 1989

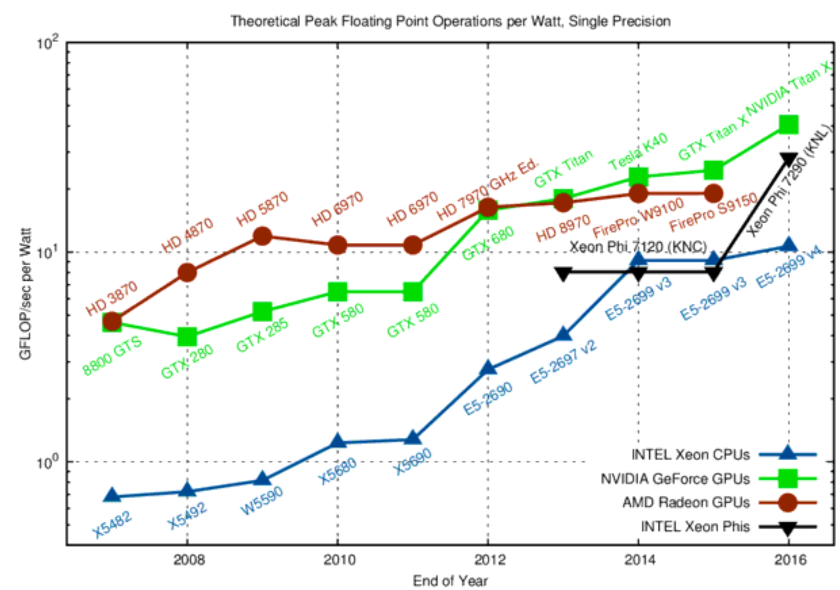
- However nowadays:
  - Large collections of labeled data available
    - ImageNet (14M images, 21k classes, hand-labeled)
  - Reducing the number of parameters by weight sharing
    - **Convolutional** layers – [LeCun-1989]
  - Novel tricks to prevent overfitting of deep nets
  - Fast enough computers (parallel hardware, GPU)

=> Optimism: It works!

Computational power

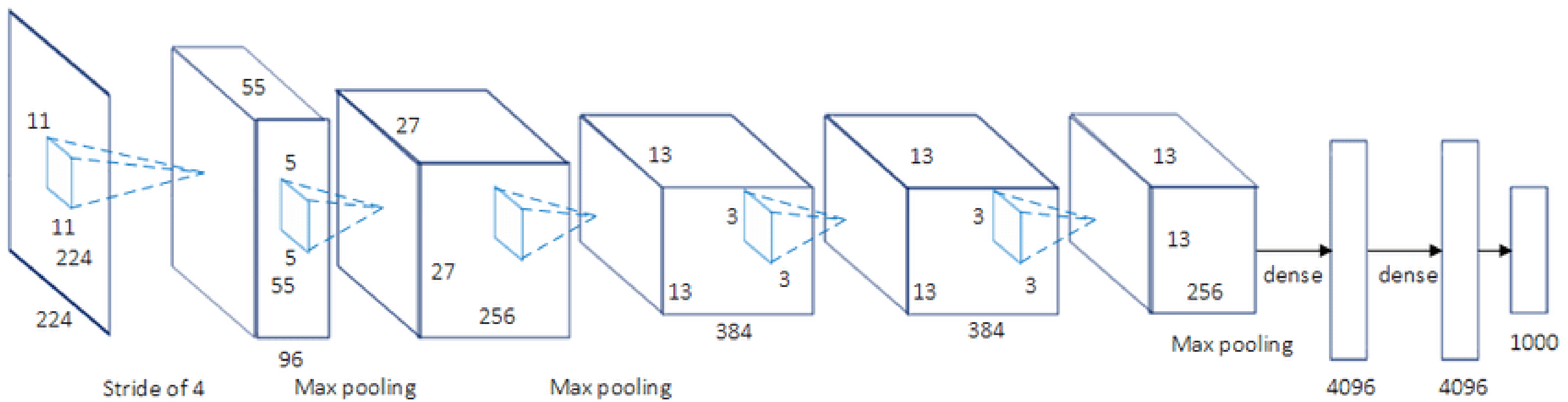






Deep convolutional neural networks

- An example for Large Scale Classification Problem:
  - Krizhevsky, Sutskever, Hinton: *ImageNet classification with deep convolutional neural networks*. NIPS, 2012.
    - Recognizes 1000 categories from ImageNet
    - Outperforms state-of-the-art by significant margin (ILSVRC 2012)

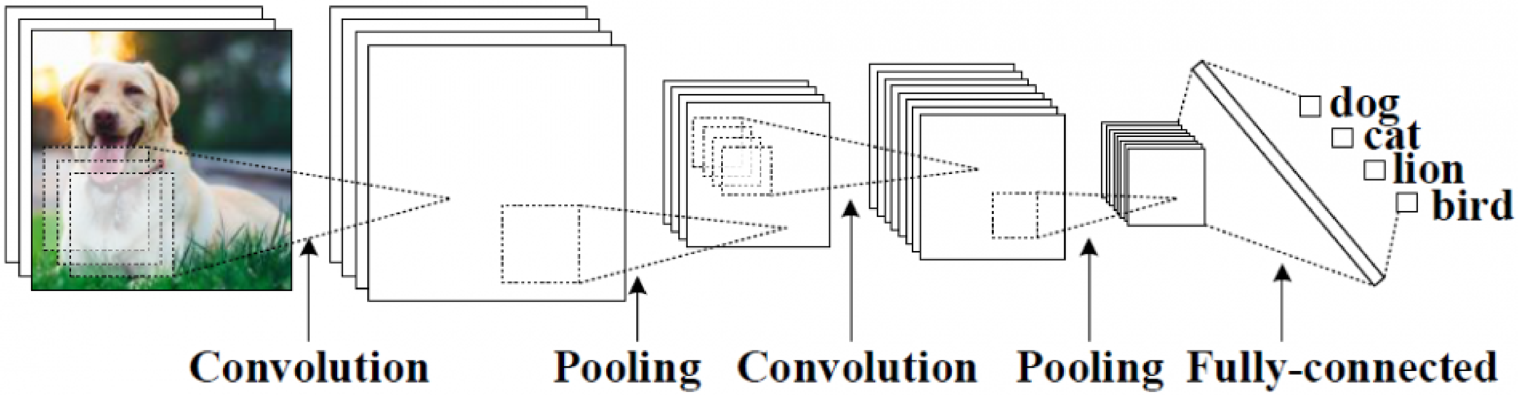


"Alex-Net"

- 5 convolutional layers, 3 fully connected layers
- 60M parameters, trained on 1.2M images (~1000 examples for each category)
- Cross-Entropy loss (softmax log-loss)

Deep CNNs – basic building blogs

- A computational graph (chain/directed acyclic graph) connecting layers
  - Each layer has: Forward pass, Backward pass
  - The graph is end-to-end differentiable

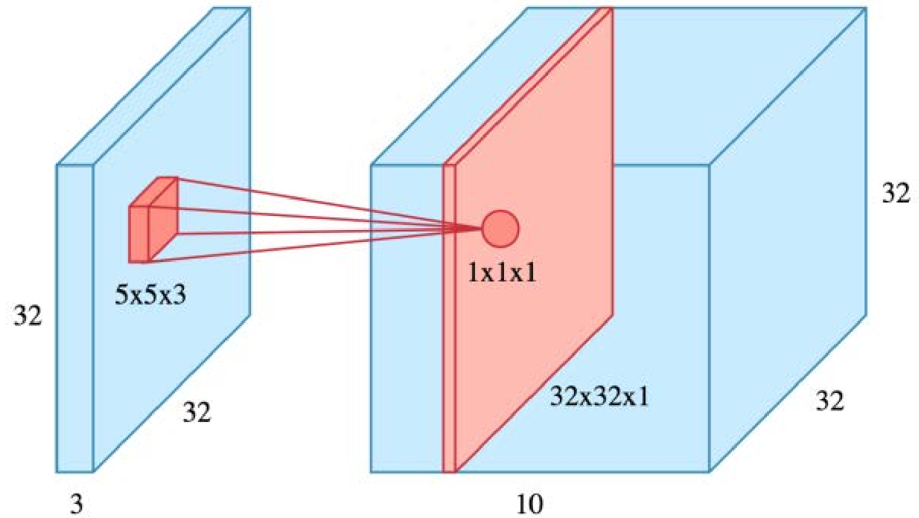


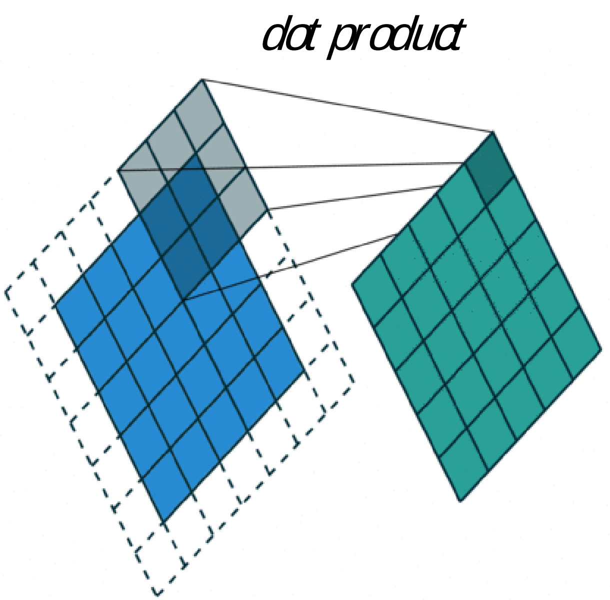
1. Input Layer
2. Intermediate Layers
   1. Convolutions
   2. Max-pooling
   3. Activations
3. Output Layer
4. Loss function over the output layer for training

Convolutional layer

- **Input**: tensor ($W \times H \times D$)
  - "image" of size W×H with D channels
- **Output**: tensor ($W' \times H' \times D'$)

- A bank of D' filters of size ($K \times K \times D$) is convolved with the input to produce the output tensor
  - Zero Padding (P), extends the input by zeros
  - Stride (S), mask shifts by more than 1 pixel
  - $K \times K \times D \times D'$ parameters to be learned

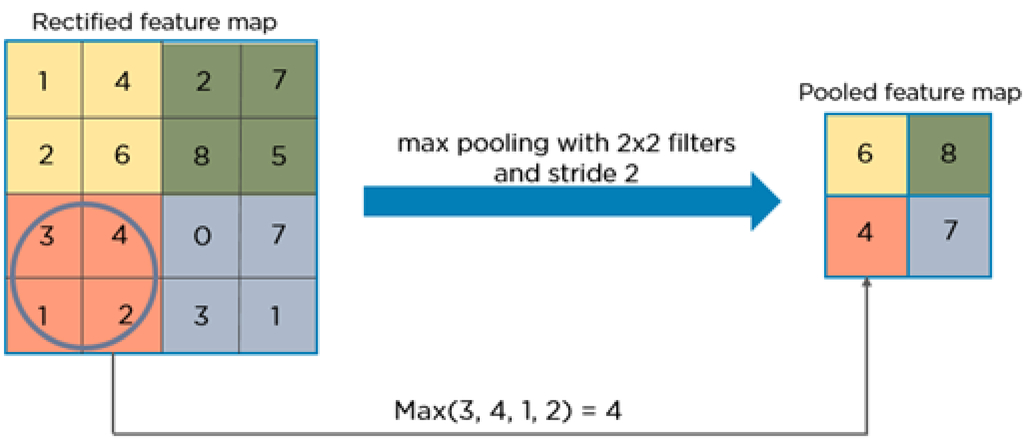


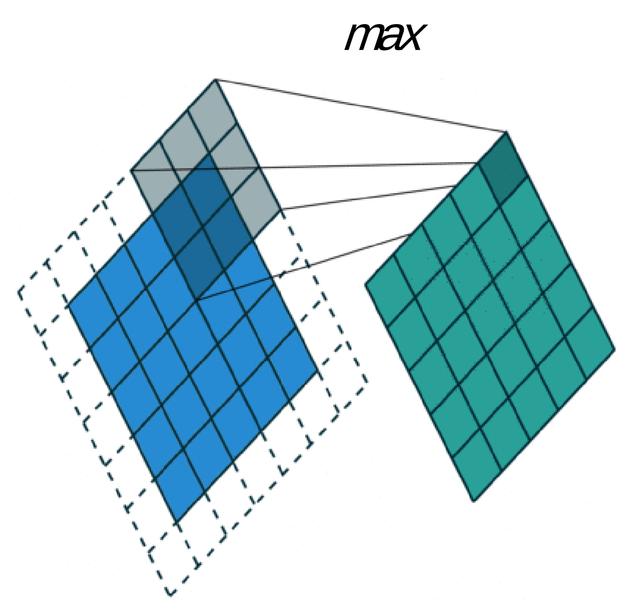


Max-pooling layer

- Same inputs (*W*×*H*×*D*) and outputs (*W'*×*H'*×*D*) as convolutional layer
- Same parameters: Mask Size (*K*), Padding (*P*), Stride (*S*)

- Same sliding window as in convolution, but instead of the dot product, pick maximum
- Non-linear operation
- No parameters to be learned



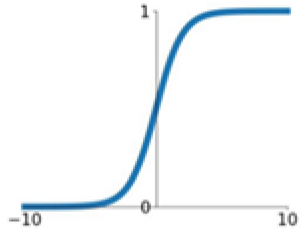


Activation functions

- Non-linearity, applied to every singe cell of the tensor
- Input tensor and output tensor of the same size

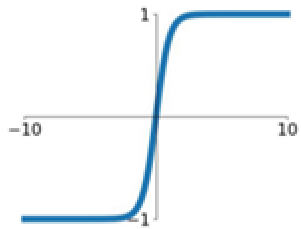
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



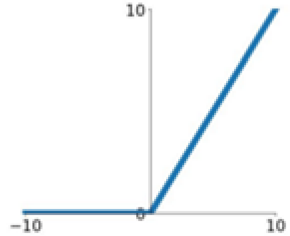
tanh

$\tanh(x)$



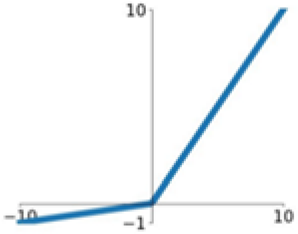
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

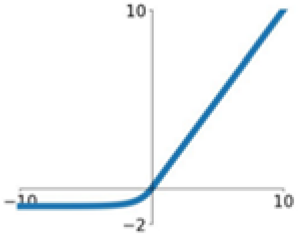


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

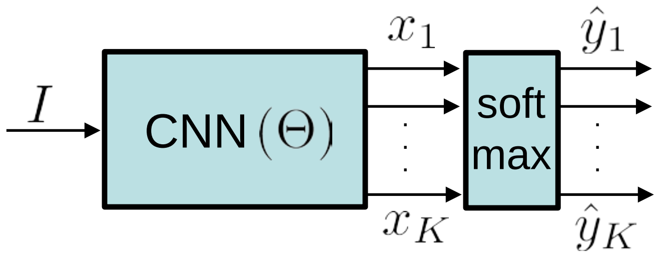
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$



- ReLU is the simplest (used in the AlexNet, good baseline)
- Saturating non-linearity (sigmoid, tanh) causes “vanishing” gradient

Multiclass Classification loss

- Cross-Entropy loss (softmax log loss)



$$\hat{y}_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$$

  – Softmax output as discrete PDF over classes

    e.g., (0.1, 0.05, 0.7, 0.05, 0.1)

$$\hat{y}_i \geq 0, \sum_{i=1}^{K} \hat{y}_i = 1$$

  – Ground-truth classes “one-hot encoding”

    e.g., (0, 0, 1, 0, 0)

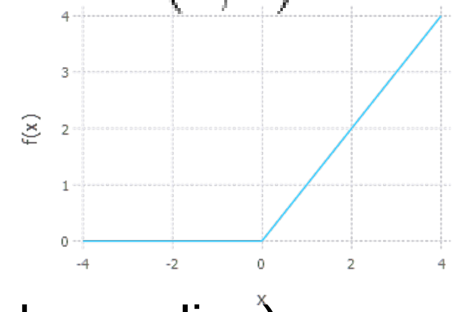
$$y_i = \begin{cases} 1 & i \text{ is the truth class} \\ 0 & \text{otherwise} \end{cases}$$

$$\boxed{L(\mathbf{y}, \hat{\mathbf{y}}(\Theta)) = -\sum_{i=1}^{K} y_i \log(\hat{y}_i) = -\log(\hat{y}_{i^*})}$$

i^* is index of the truth class

Deep convolutional neural networks

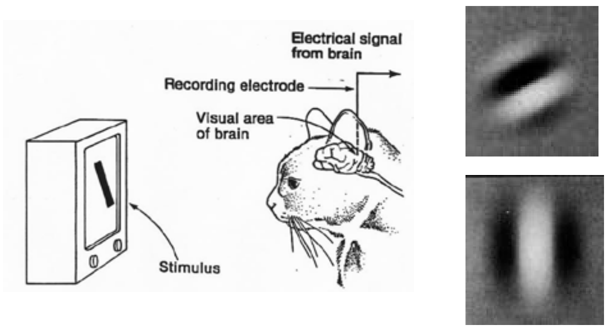
$$f(x) = \max(0, x)$$



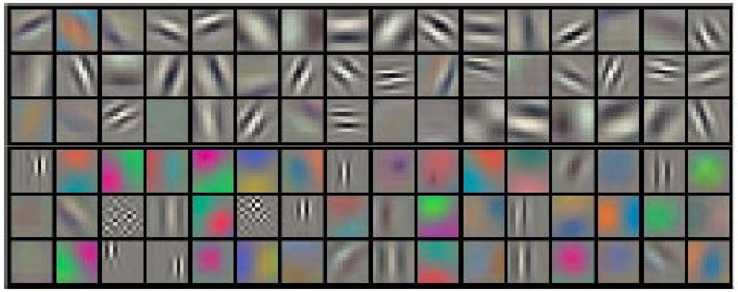
- Additional tricks: "Devil is in the details"
  - Rectified linear units instead of standard sigmoid
    => Mitigate vanishing gradient problem
  - Convolutional layers followed by max-pooling
    - Local maxima selection in overlapping windows (subsampling)
    => dimensionality reduction, shift insensitivity
  - Dropout
    - 50% of hidden units are randomly omitted during the training, but weights are shared in test time
    - Averaging results of many independent models (similar idea as in Random forests)
    => Probably very significant to reduce overfitting
  - Data augmentation
    - Images are artificially shifted and mirrored (10 times more images)
    => transformation invariance, reduce overfitting

Deep convolutional neural networks

- Supervised training
  - The training is done by a standard back-propagation
  - enough labeled data: 1.2M labeled training images for 1k categories
  - Learned filters in the first layer
    - Resemble cells in primary visual cortex



[Hubel-Wiesel-1959]

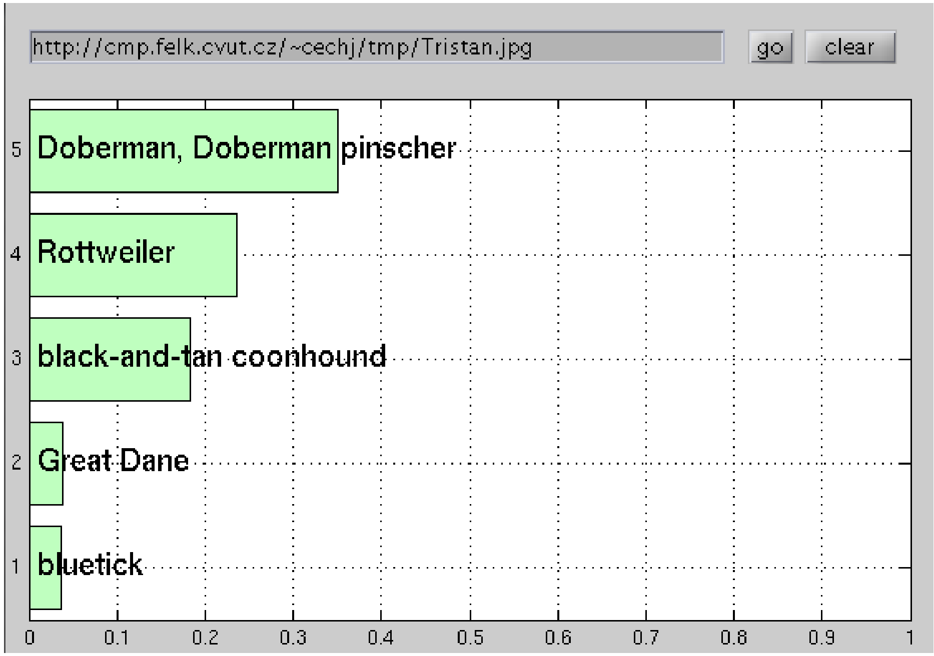


Learned first-layer filters

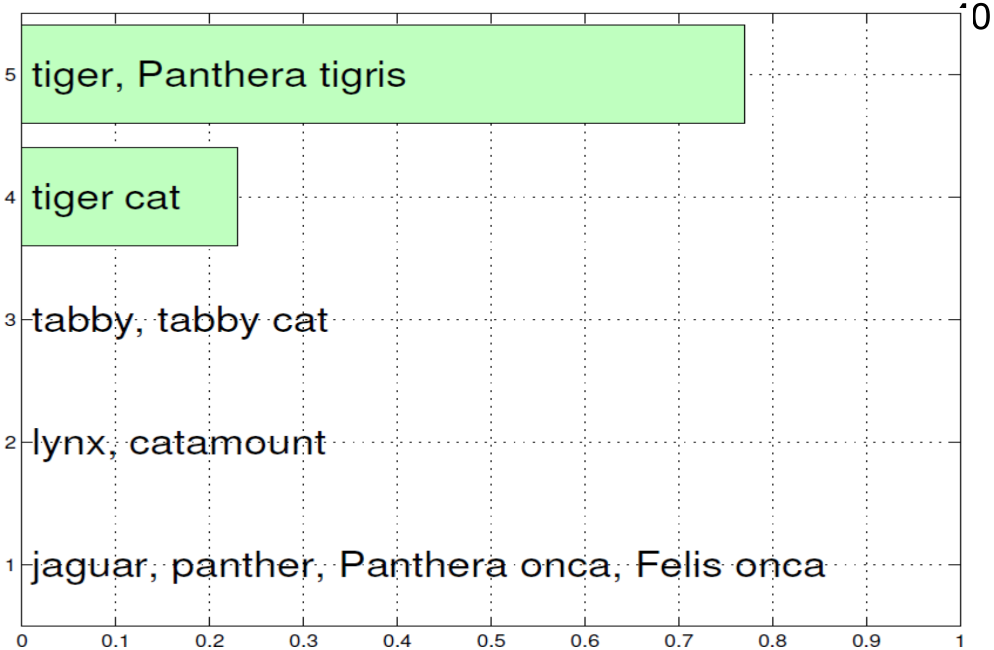
- Training time:
  - 5 days on NVIDIA GTX 580, 3GB memory (Krizhevsky, today faster)
  - 90 cycles through the training set
- Test time (forward step) on GPU
  - Implementation by Yangqing Jia, http://caffe.berkeleyvision.org/
  - 5 ms/image in a batch mode

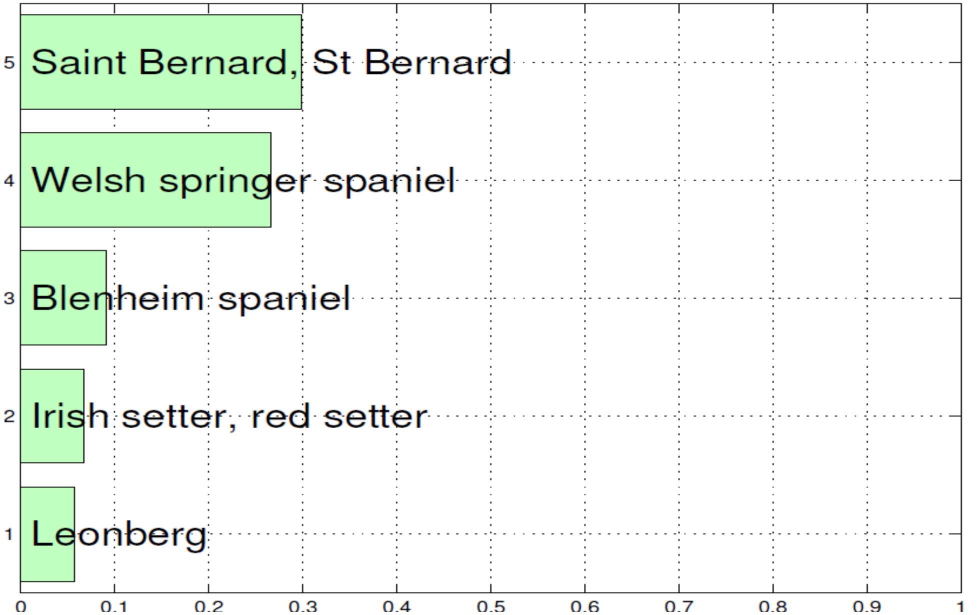
Early experiments 1: Category recognition

- Implementation by Yangqing Jia, 2013, http://caffe.berkeleyvision.org/
  - network pre-trained for 1000 categories provided
- Which categories are pre-trained?
  - 1000 "most popular" (probably mostly populated)
  - Typically very fine categories (dog breeds, plants, vehicles…)
  - Category "person" (or derived) is missing
  - Recognition accuracy subjectively surprisingly good…



It is not a texture only...





Early experiments 2: Category retrieval

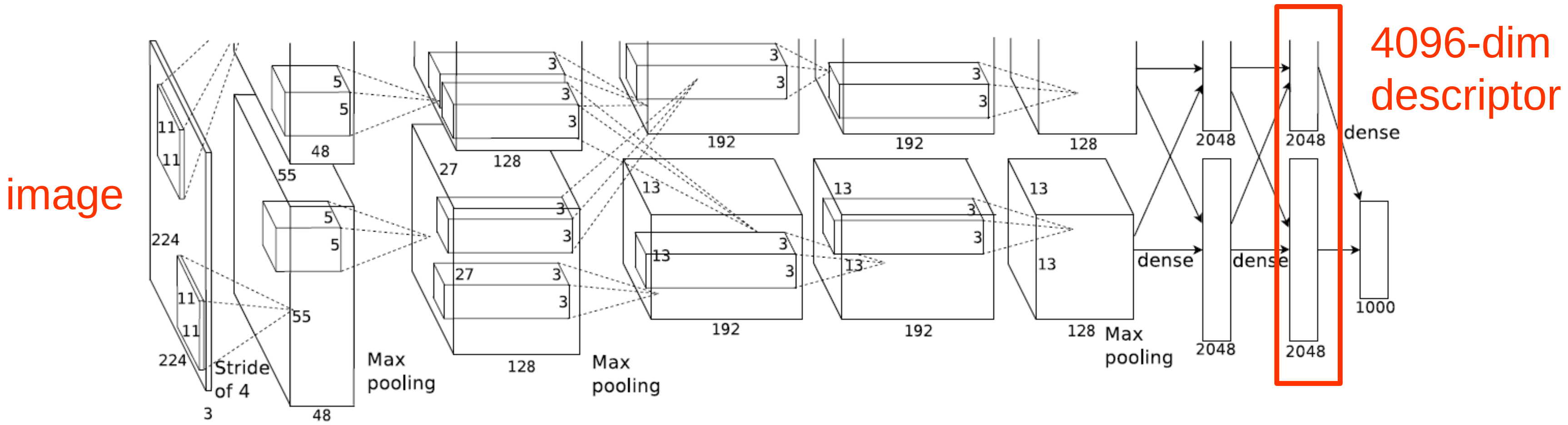
- 50k randomly selected images from Profimedia dataset
- Category: Restaurant (results out of 50k-random-Profiset)

Early experiments 2: Category retrieval

- Category: stethoscope (results out of 50k-random-Profiset)

Early experiments 3: Similarity search

- Indications in the literature that the last hidden layer carry semantics
  - Last hidden layer (4096-dim vector), final layer category responses (1000-dim vector)
  - New (unseen) categories can be learned by training (a linear) classifier on top of the last hidden layer
    - Oquab, Bottou, Laptev, Sivic, CVPR, 2014
    - Girshick, Dphanue, Darell, Malik, CVPR, 2014
  - **Responses of the last hidden layer can be used as a compact global image descriptor**
    - Semantically similar images should have small Euclidean distance



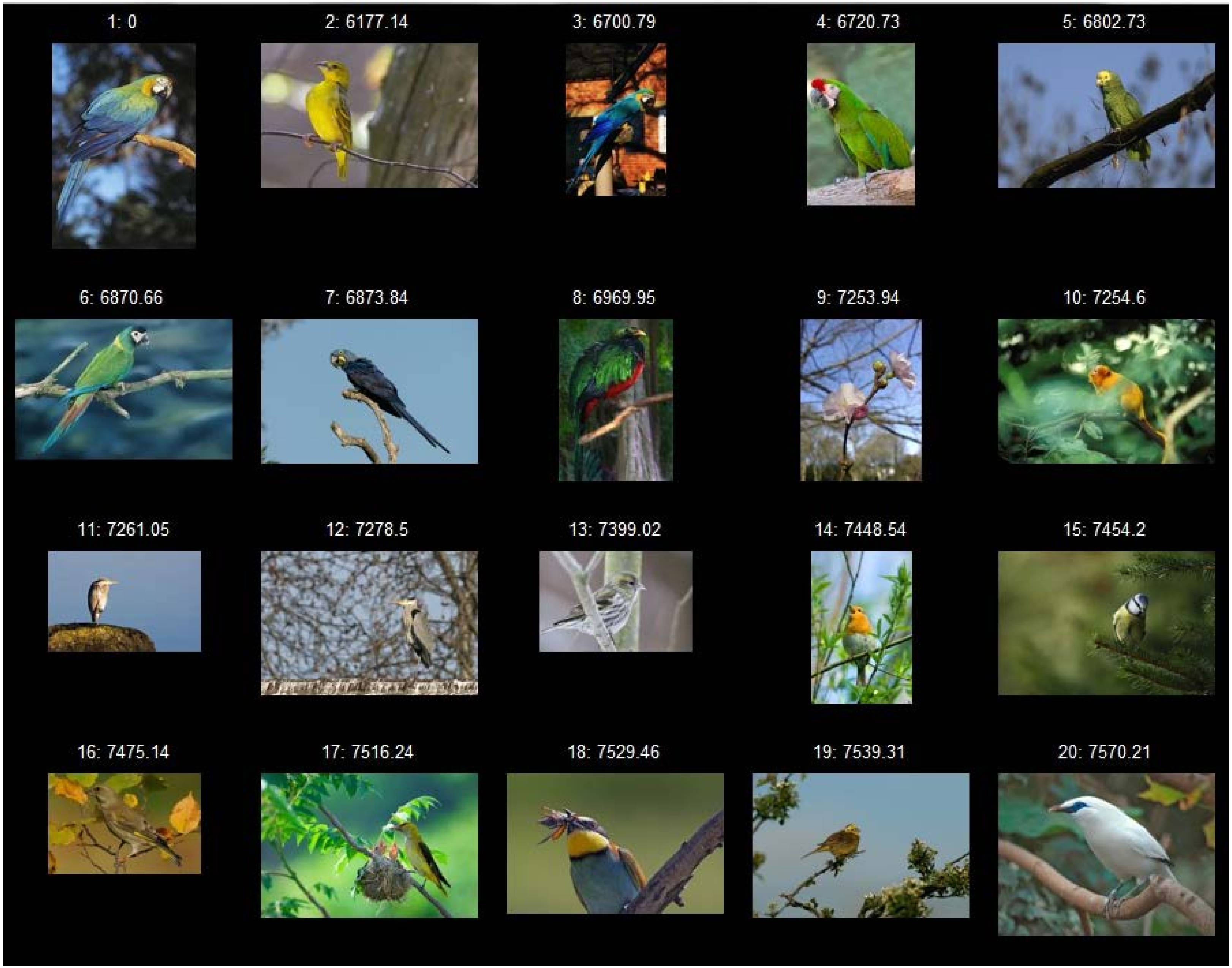
Early experiments 3: Similarity search

- Qualitative comparison: (20 most similar images to a query image)
  1. MUFIN annotation (web demo), http://mufin.fi.muni.cz/annotation/, [Zezula et al., *Similarity Search: The Metric Space Approach.* 2005.]
     - Nearest neighbour search in **20M** images of Profimedia
     - Standard global image statistics (e.g. color histograms, gradient histograms, etc.)
  2. Caffe NN (last hidden layer response + Euclidean distance),
     - Nearest neighbour search in **50k** images of Profimedia
     - **400 times smaller dataset** !

MUFIN results

Early experiments 3: Similarity search

Caffe NN results



Early experiments 3: Similarity search

MUFIN results

Early experiments 3: Similarity search

Caffe NN results

Early experiments 3: Similarity search

MUFIN results

Early experiments 3: Similarity search

Caffe NN results

Early experiments 3: Similarity search

MUFIN results

Early experiments 3: Similarity search

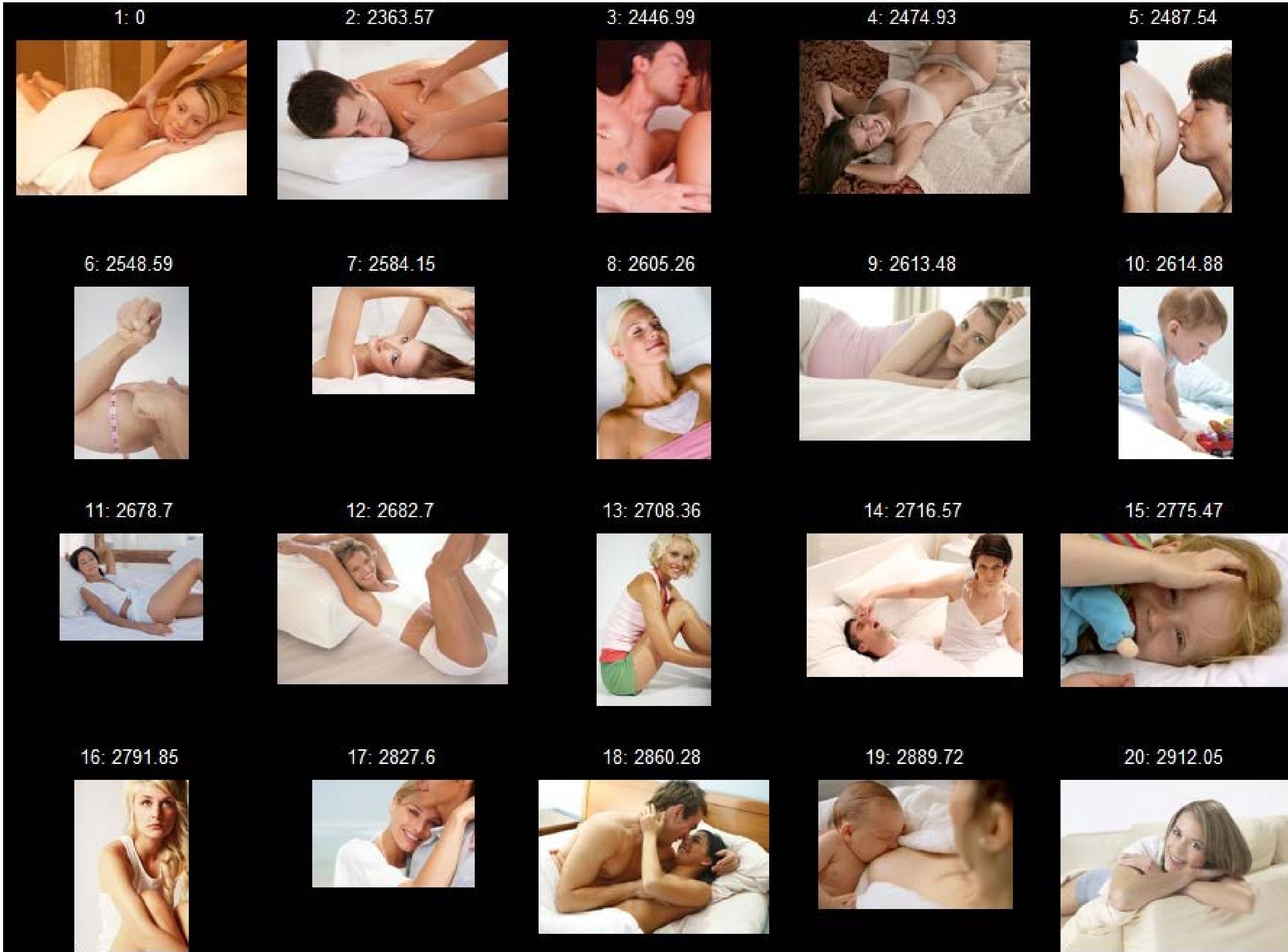
Caffe NN results

Early experiments 3: Similarity search

MUFIN results

Early experiments 3: Similarity search

Caffe NN results



Novel tricks

- Network initialization
  - Mishkin, Matas. *All you need is a good init*. ICLR 2016
  - Weights initialization: zero mean, unit variance, orthogonality

- Batch normalization
  - Iosse, Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. NIPS 2015
  - Zero mean and unit variance weights are “supported” during training to avoid vanishing gradient

⇒ Small sensitivity to learning rate setting (can be higher, faster training – 10 times fewer epochs needed)

⇒ Regularizer (dropout can be excluded/smaller) (better optimum found)

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

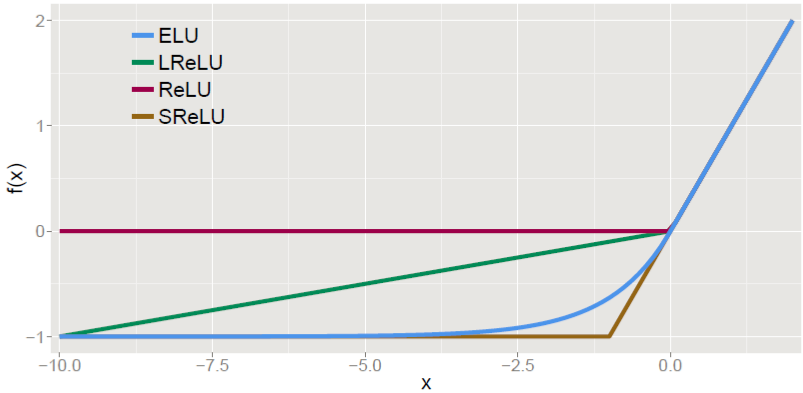
$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Novel tricks II.

- Exponential Linear Units (ELU) [Clevert et al., ICLR 2016]

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \le 0 \end{cases}$$

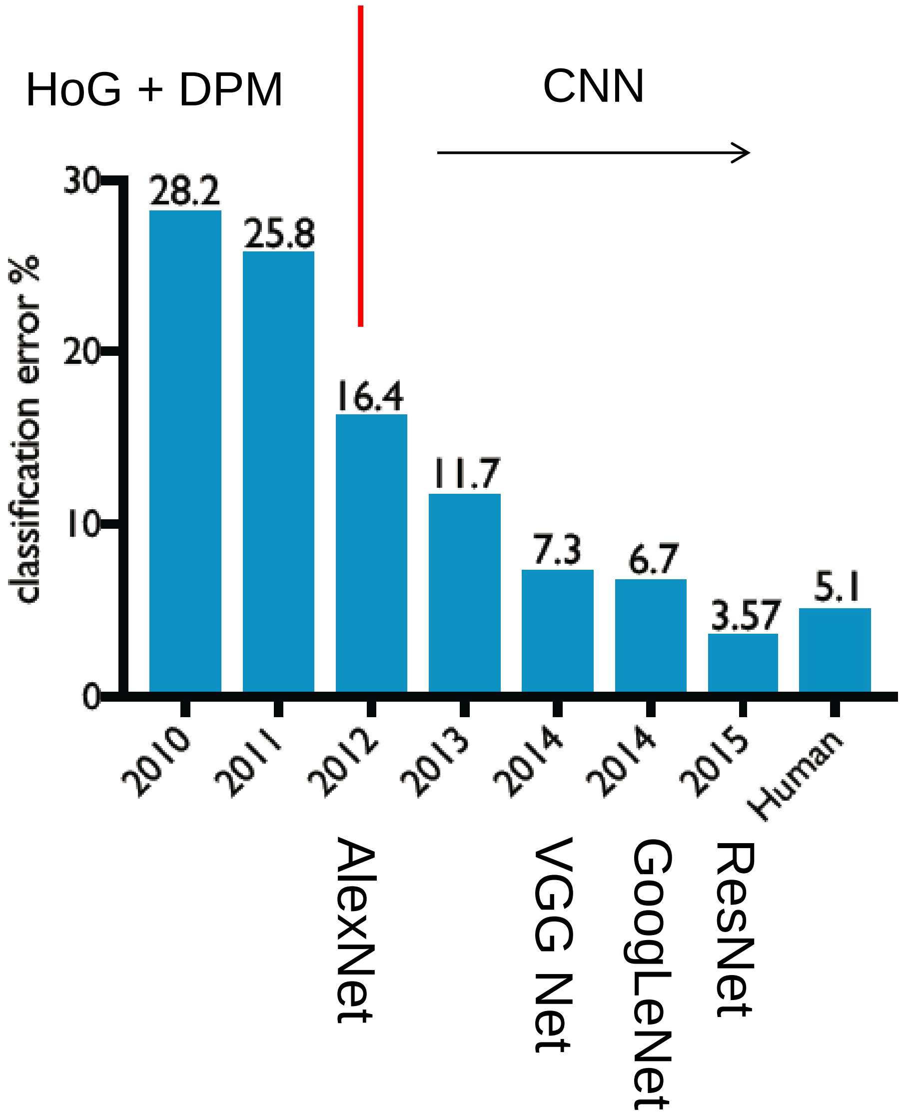


  – Self normalizing properties, batch normalization unnecessary
  – Faster training reported

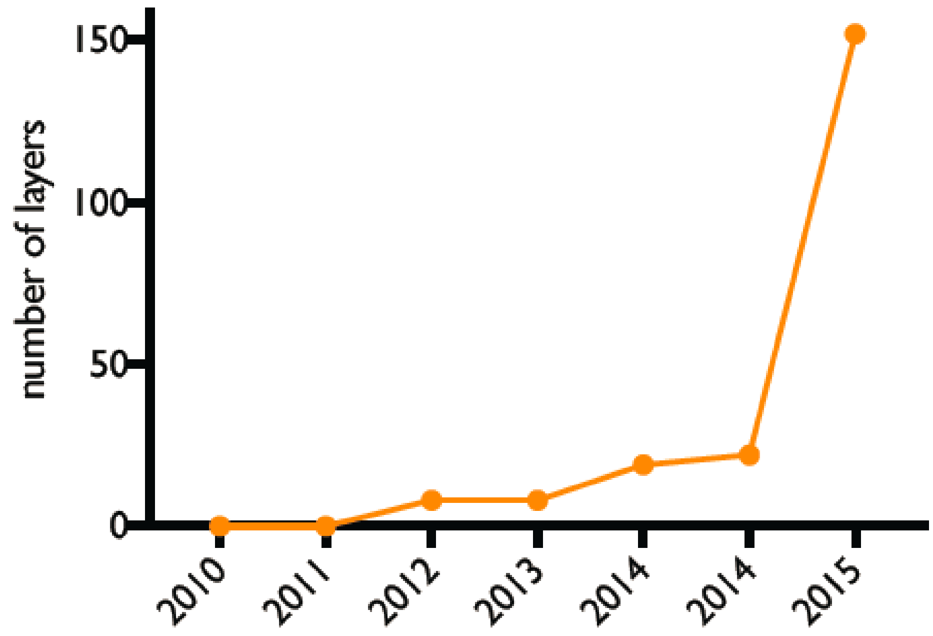
- ADAM optimizer [Kingma and Ba, ICLR 2015]
  = (ADAptive Moments)
  – Often improves over SGD (with momentum),
  – Low sensitivity on learning rate setting

Novel architectures

- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)



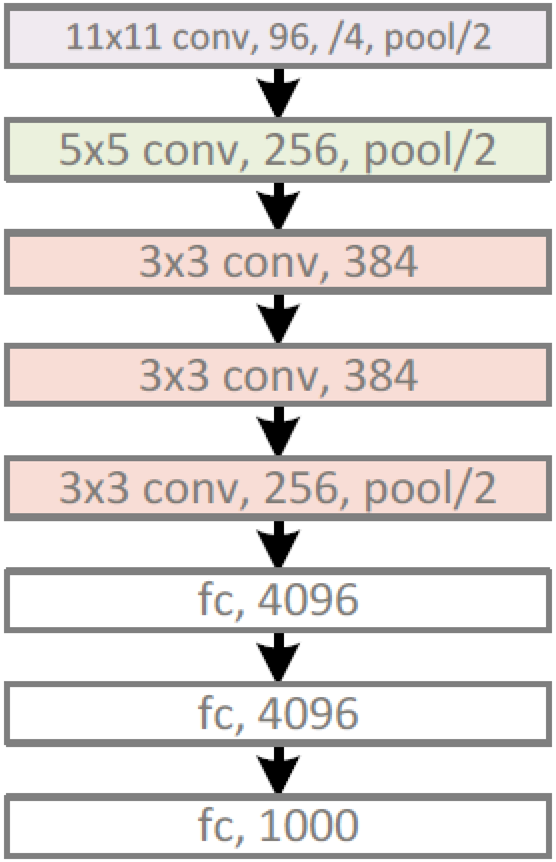
IMAGENET

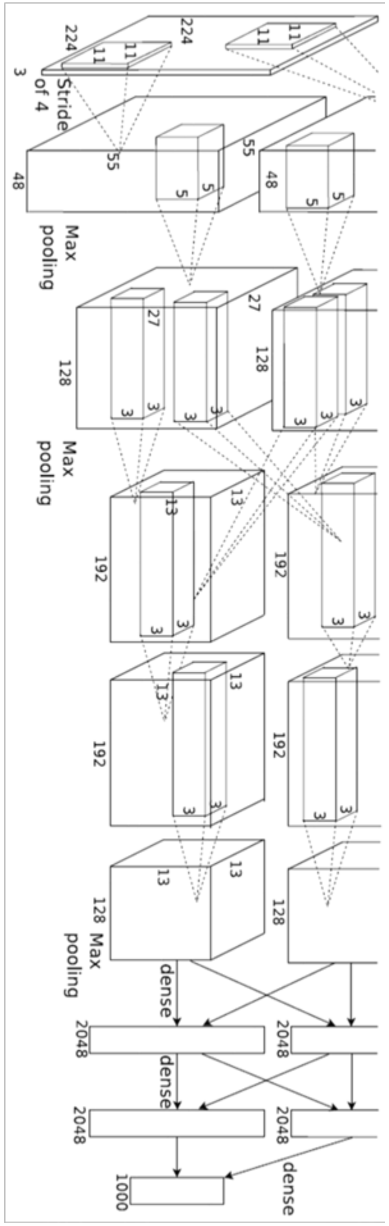


=> “Go deeper”

CNN architectures

- AlexNet
  - [Krishevsky et al., NIPS 2012]



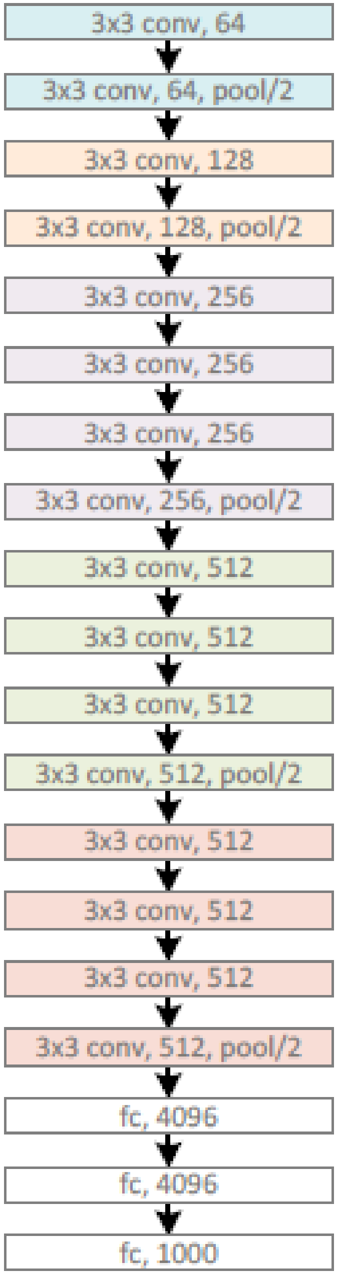


CNN architectures

- VGG Net: VGG-16, VGG-19
  - [Simonyan and Zisserman, ICLR 2015]
  - Deeper than AlexNet
  - Smaller filters (3x3 convolutions), more layers

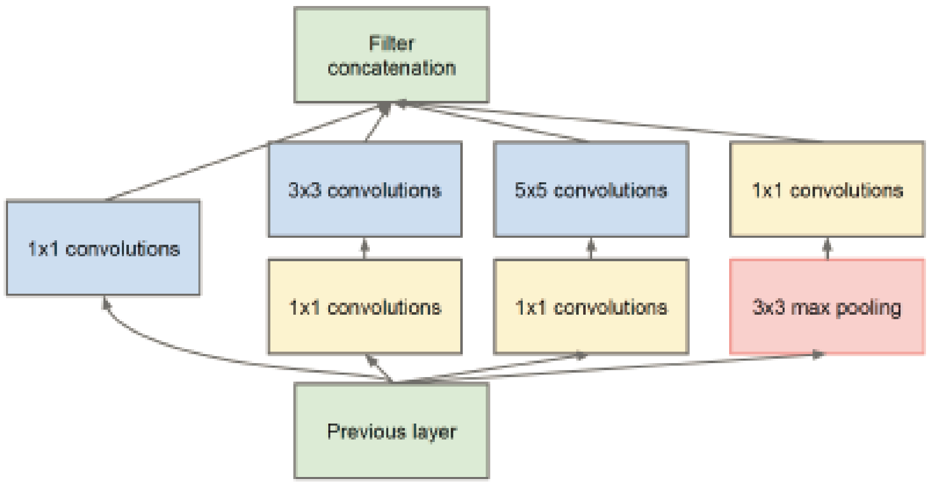
    => Same effective receptive field,

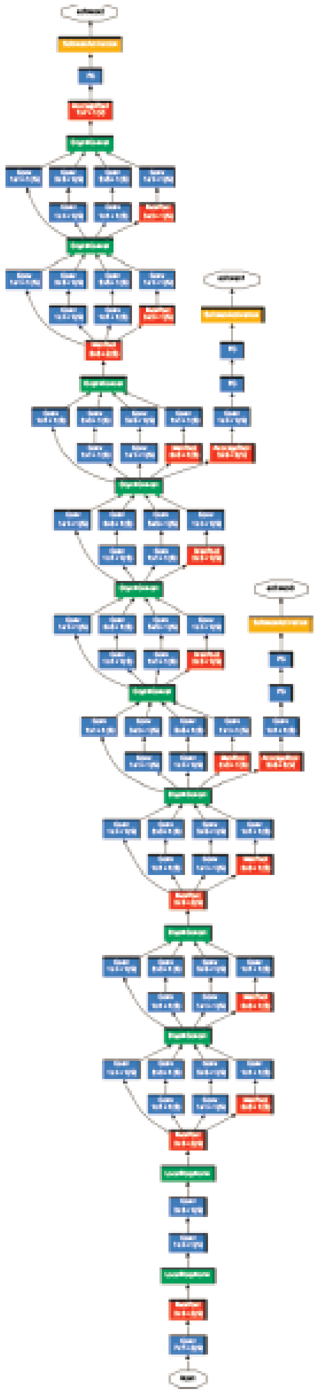
    but more "non-linearity"



CNN architectures

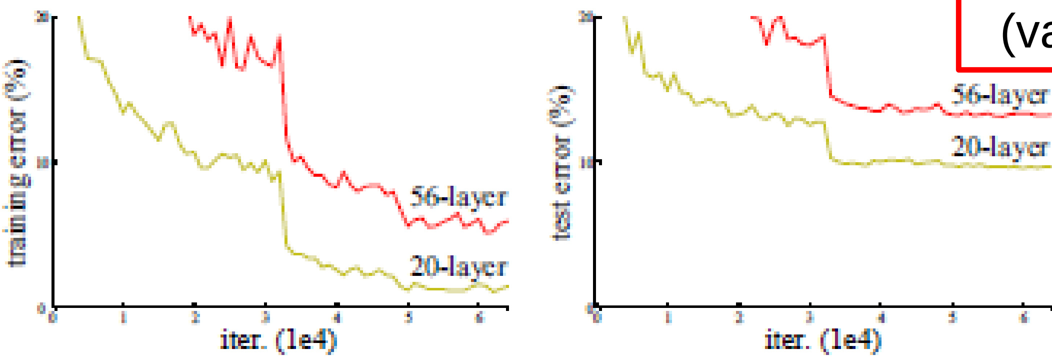
- GoogLeNet
  - [Szegedy et al., CVPR 2015]
  - 22 layers, No Fully-Connected layers
  - Accurate, much less parameters
  - “Inception” module (Net in net)



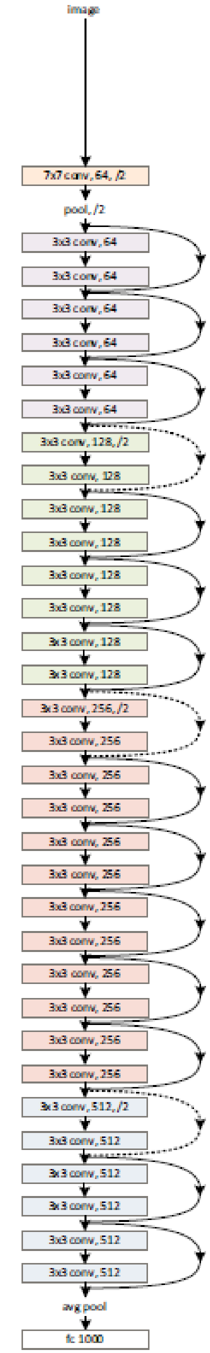


CNN architectures

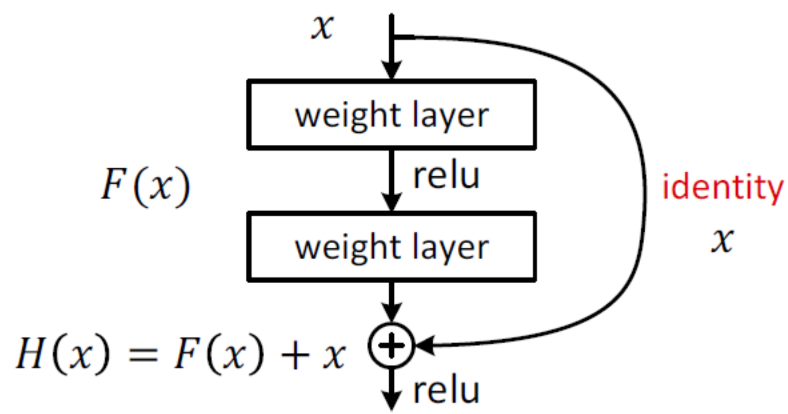
- ResNet
  - [He et al., CVPR 2016]



=> Plain deeper models are not better (vanishing gradient)

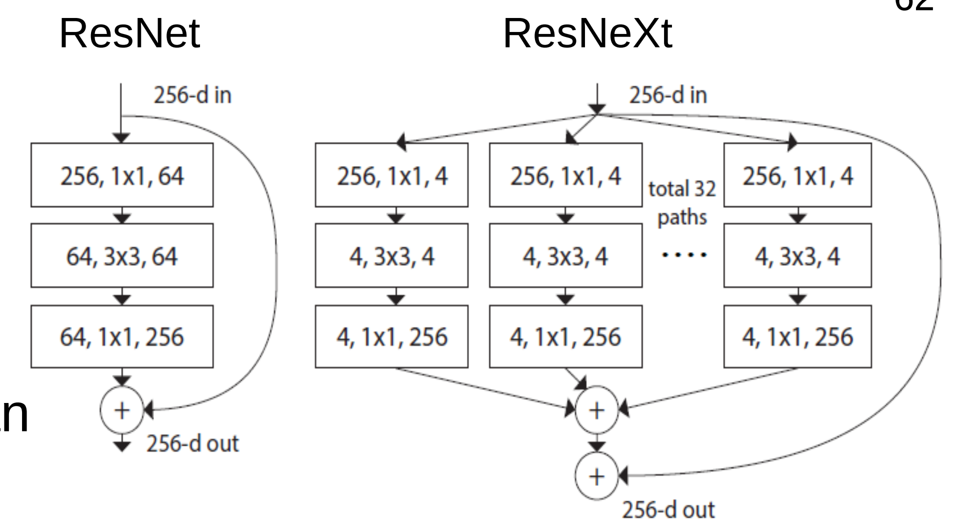


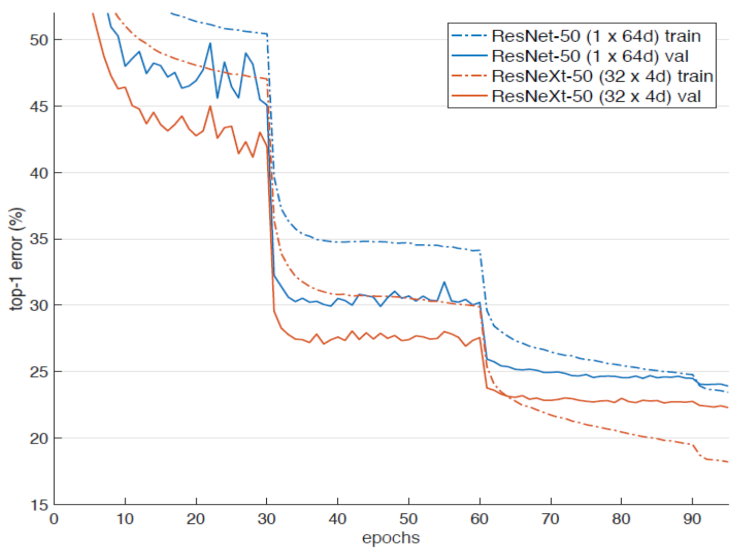
  - Residual modules, 152 layers

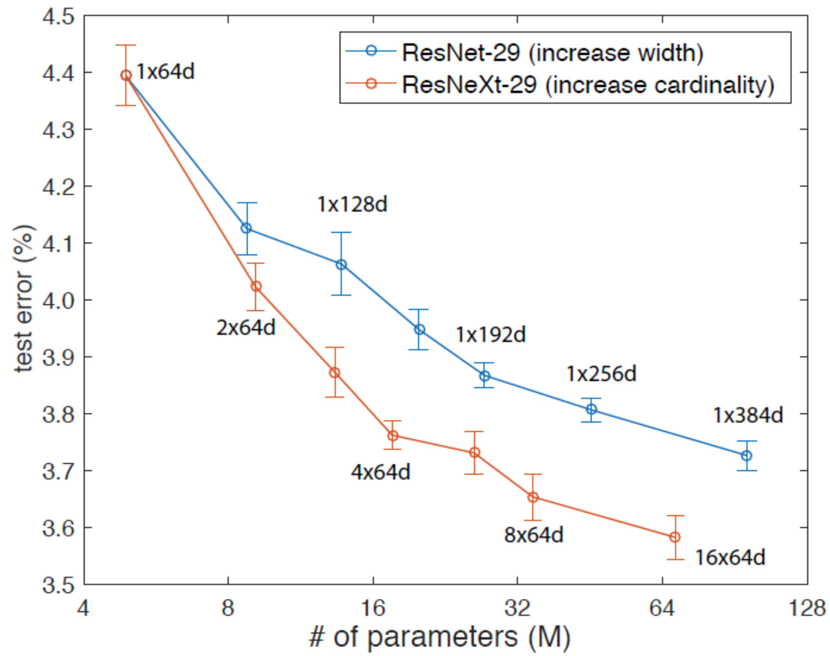


CNN architectures

- ResNeXt
  - [Xie-CVPR-2017]
  - Improvement of ResNet
  - Cardinality
    - number of branches in a block
  - “Increasing cardinality, better than going wider or deeper"



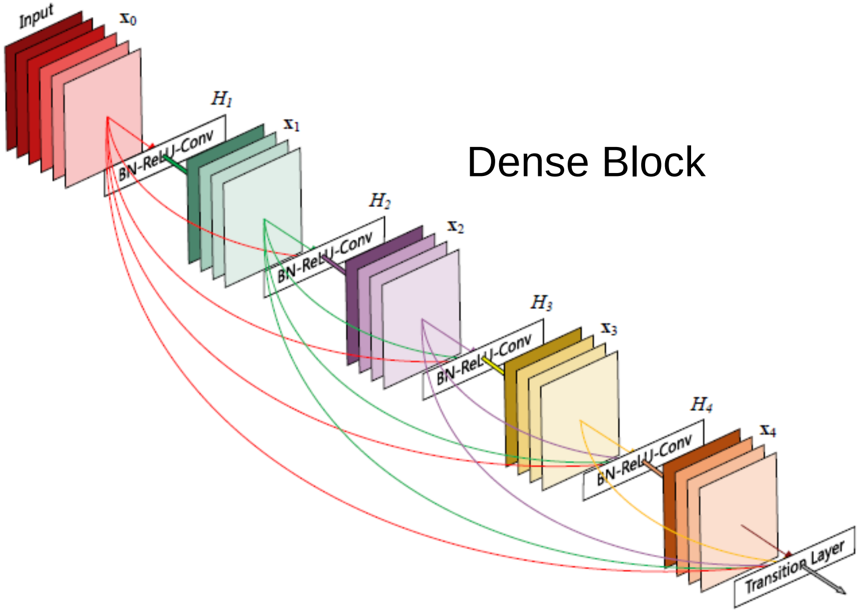


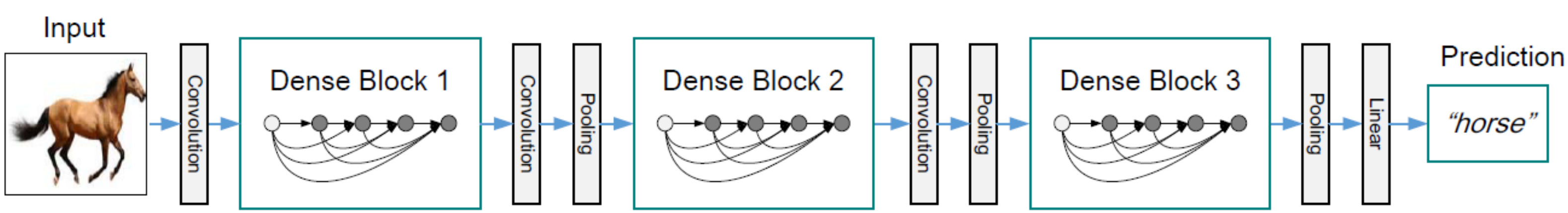


CNN architectures

- DenseNet
  - [Huang-CVPR-2017]
  - Densifying Skip connections
  - Chain of several “dense blocks”
  - Argument: Features are reused
  - Higher accuracy with fewer parameters over ResNet reported
  - Best paper award @ CVPR

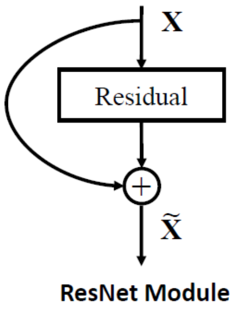
Dense Block

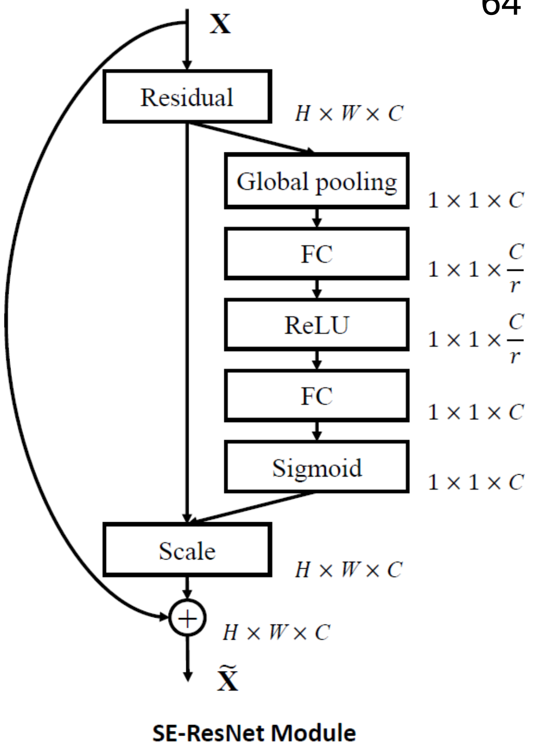


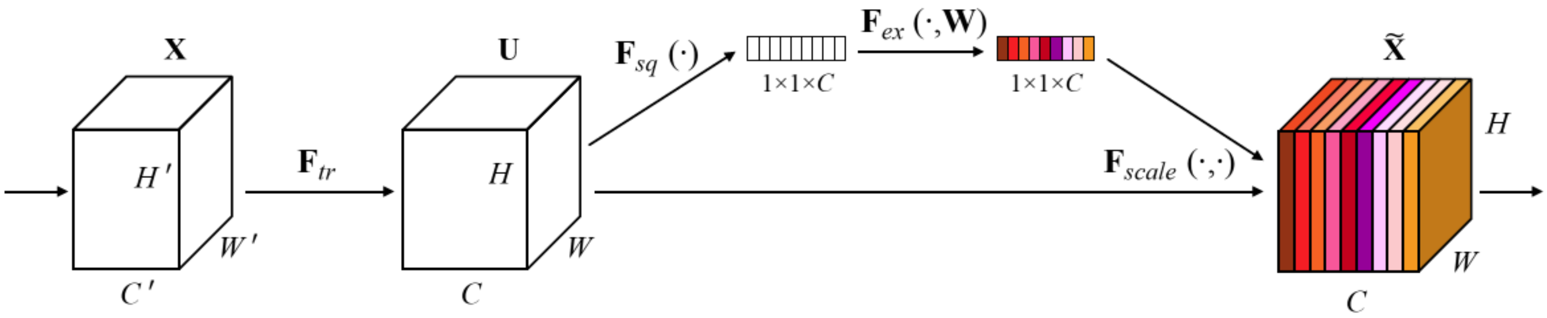


CNN architectures

- Squeeze-and-Excitation Network (SE-Net)
  - [Hu-CVPR-2018, Hu-TPAMI-2019]
  - Chain of SE-blocks
  - Squeeze:
    - Channel descriptor by aggregating over spatial dimension
  - Excitation
    - Small bottleneck fully connected net producing scale of each channel
  - Capture channel interdependences
  - Winner of ILSVRC 2017 (Top-5 err 2.25%)
  - Negligible extra computational cost

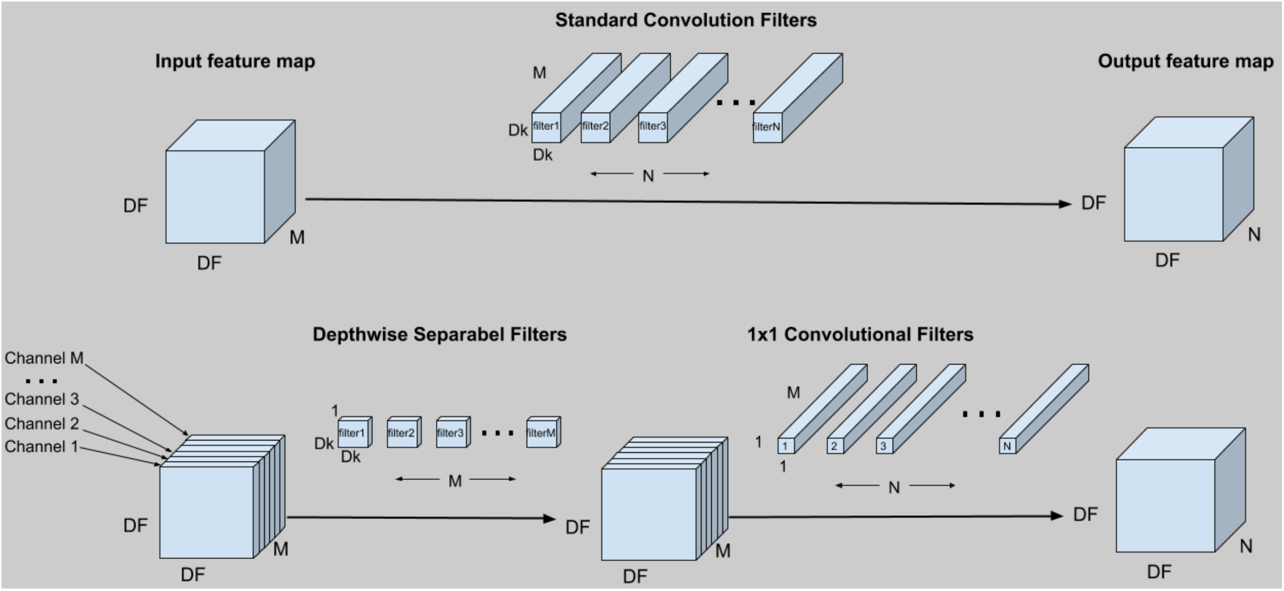




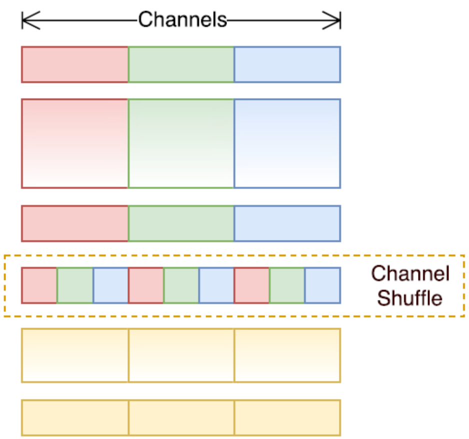


CNN architectures

- Computationally efficient architectures
  - **MobileNet** [Howard-2017, Google Inc.]
    - depth wise separable convolutions

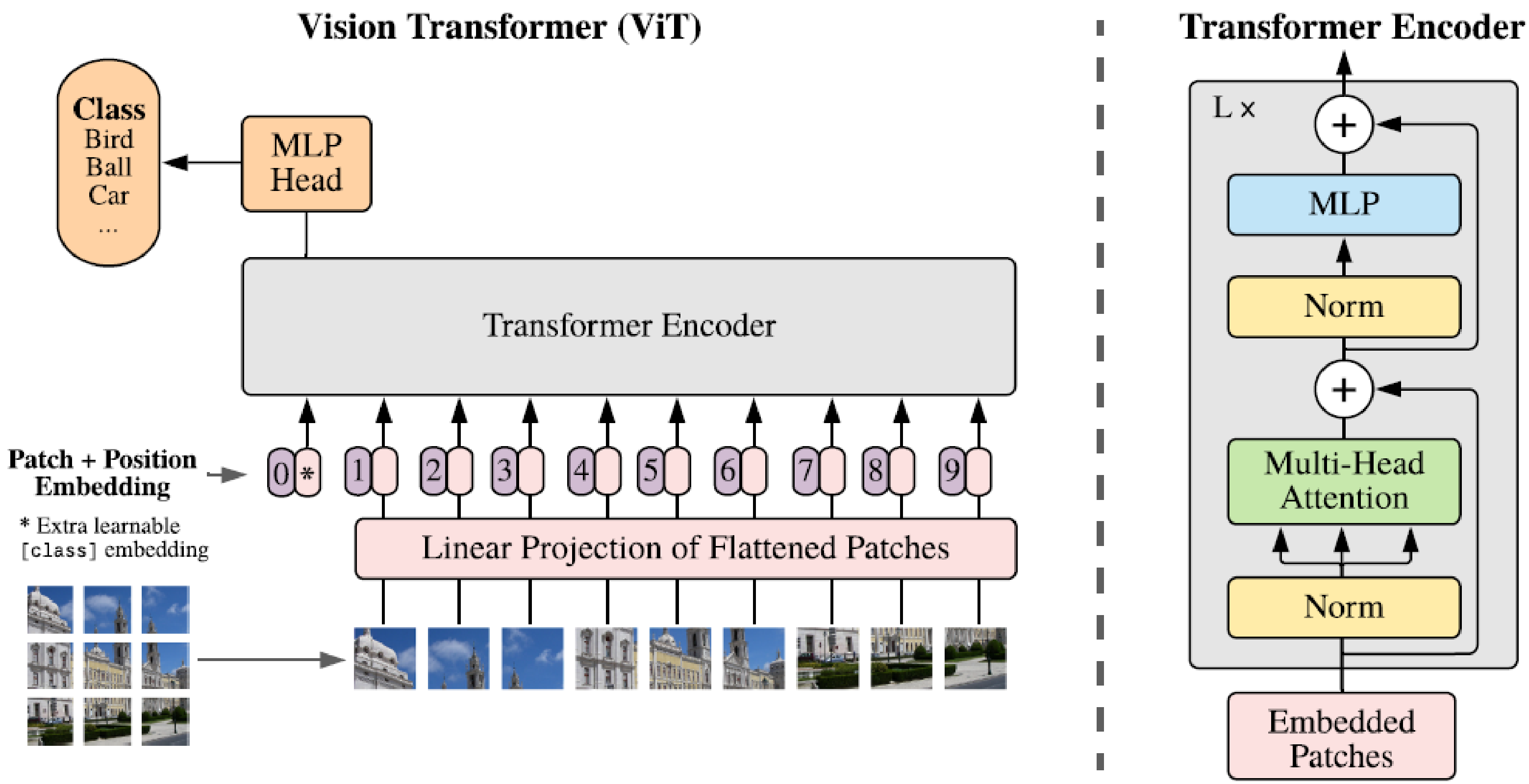


  - **ShuffleNet** [Zhang-CVPR-2018, Face++]
    - Comparable accuracy with AlexNet, 13x speed up



DNN architectures - Transformers

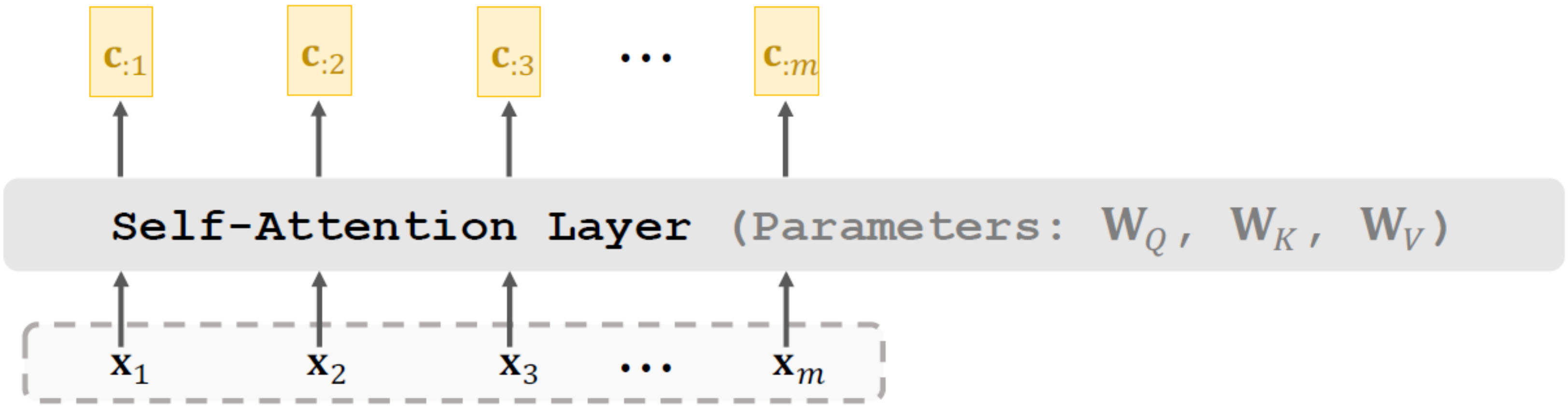
- Vision Transformers [Dosovitskiy-2021]
  - Taken from Natural Language Processing [Wasvani-2017]
  - No Convolutions
  - Image is cut into fixed-size patches and the sequence of vectorized patches (tokens/words) is fed into the transformer



  - Outperforms ResNET on ImageNet, but needs 100M image pretraining

DNN architectures - Transformers

- (Vision) Transformer
  - Main idea: **Self-Attention Mechanism**
    - Inputs (vectors $\mathbf{x_1}, \ldots, \mathbf{x_m}$)
    - Parameters (matrices $\mathbf{W_Q}, \mathbf{W_K}, \mathbf{W_V}$)



Query: $\mathbf{q}_{:i} = \mathbf{W}_Q \mathbf{x}_i$, Key: $\mathbf{k}_{:i} = \mathbf{W}_K \mathbf{x}_i$, Value: $\mathbf{v}_{:i} = \mathbf{W}_V \mathbf{x}_i$.

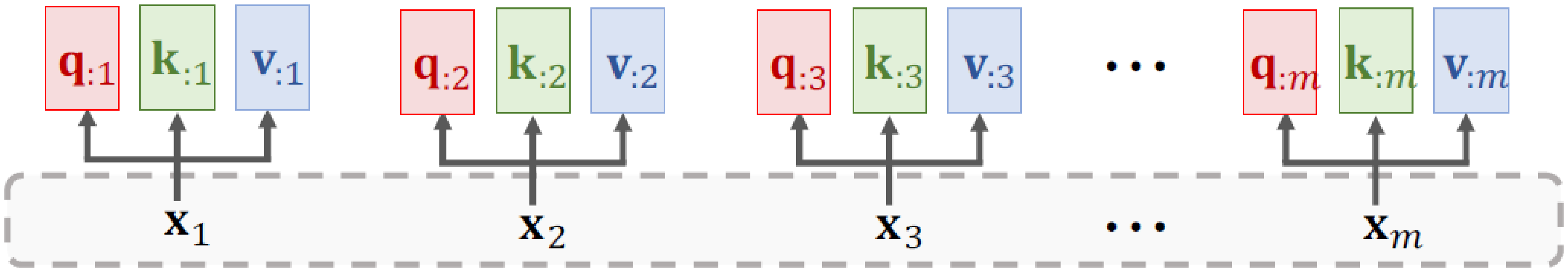
$$\mathbf{c}_{:j} = \mathbf{V} \cdot \mathrm{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right).$$

Courtesy of Shusen Wang

DNN architectures - Transformers

- (Vision) Transformer
  - Main idea: **Self-Attention Mechanism**
    - Inputs (vectors $\mathbf{x_1}$, …, $\mathbf{x_m}$)
    - Parameters (matrices $\mathbf{W_Q}, \mathbf{W_K}, \mathbf{W_V}$)

Query: $\mathbf{q}_{:i} = \mathbf{W}_Q \mathbf{x}_i,$ Key: $\mathbf{k}_{:i} = \mathbf{W}_K \mathbf{x}_i,$ Value: $\mathbf{v}_{:i} = \mathbf{W}_V \mathbf{x}_i.$

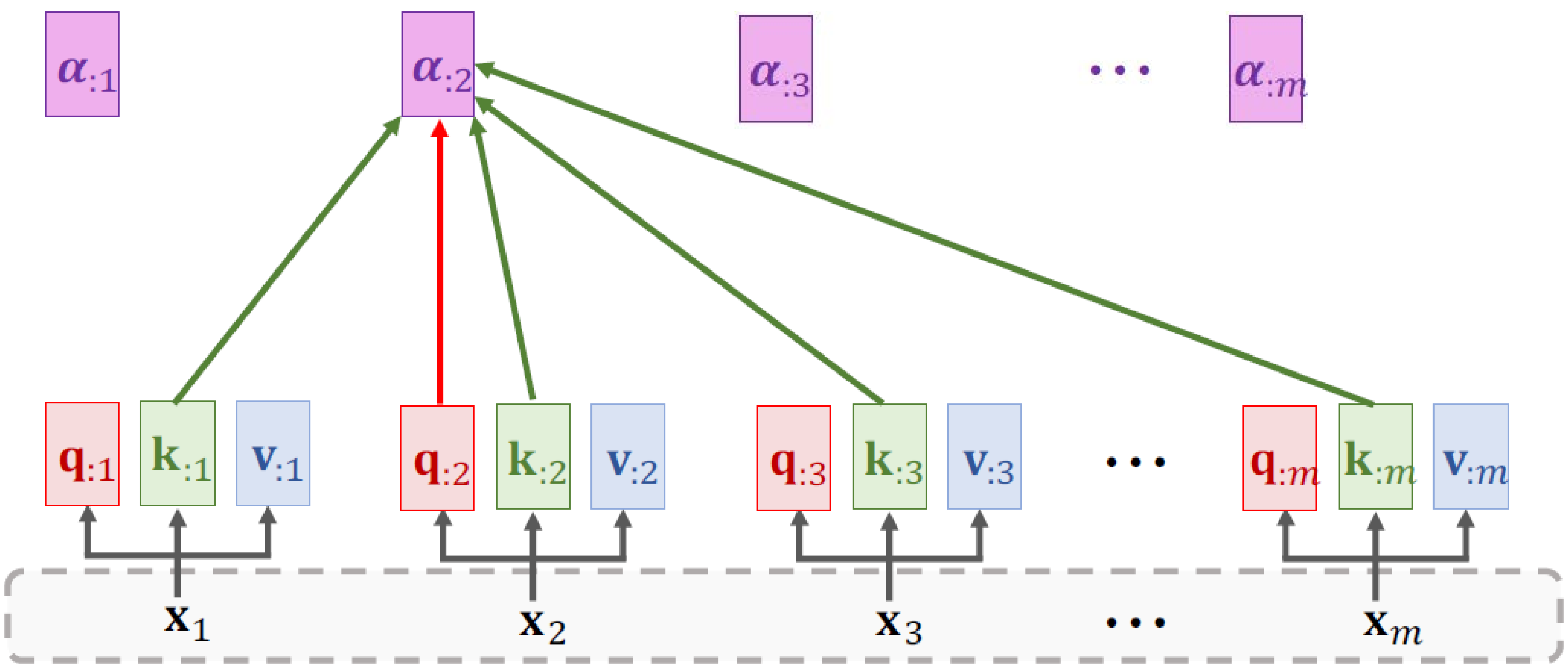


Courtesy of Shusen Wang

DNN architectures - Transformers

- (Vision) Transformer
  - Main idea: **Self-Attention Mechanism**
    - Inputs (vectors $\mathbf{x}_1$, …, $\mathbf{x}_m$)
    - Parameters (matrices $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$)

Weights: $\alpha_{:j} = \mathrm{Softmax}\left(\mathbf{K}^T \mathbf{q}_{:j}\right) \in \mathbb{R}^m.$

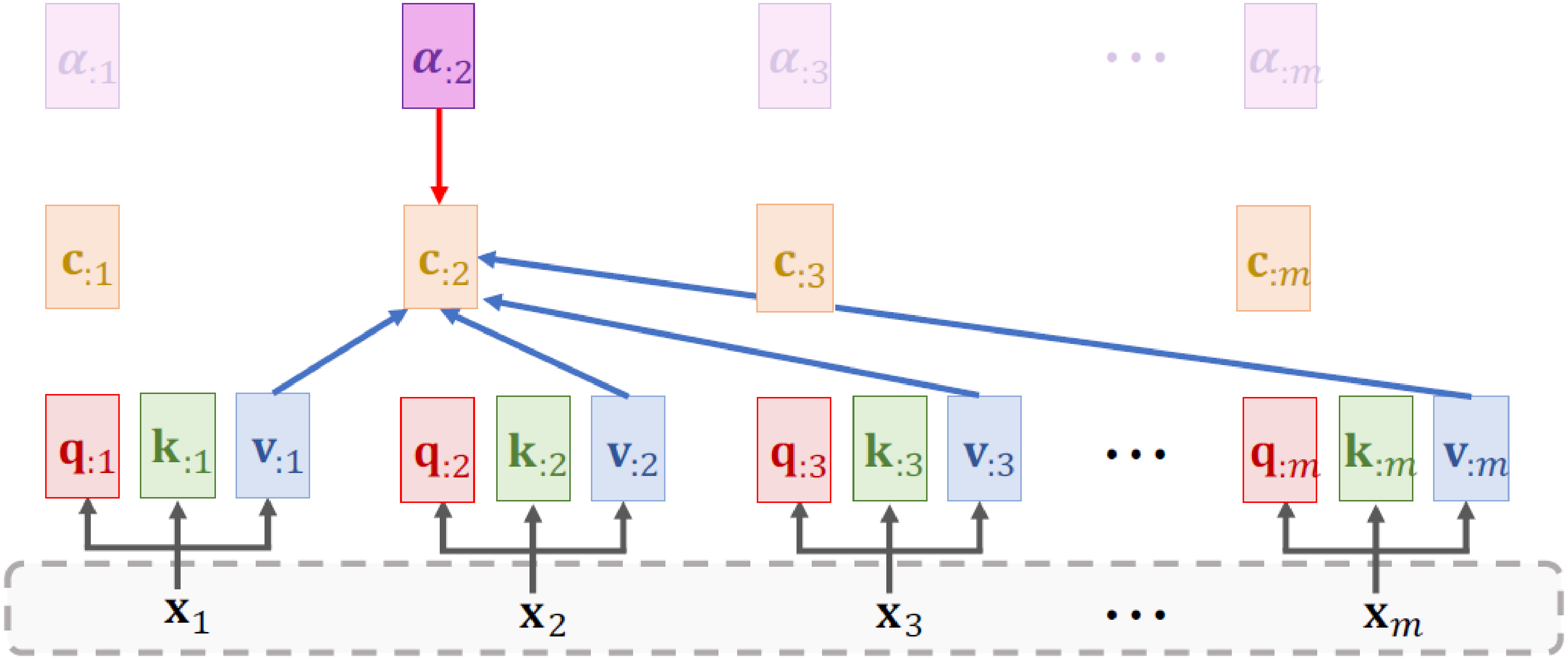


Courtesy of Shusen Wang

DNN architectures - Transformers

- (Vision) Transformer
  - Main idea: **Self-Attention Mechanism**
    - Inputs (vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$)
    - Parameters (matrices $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$)

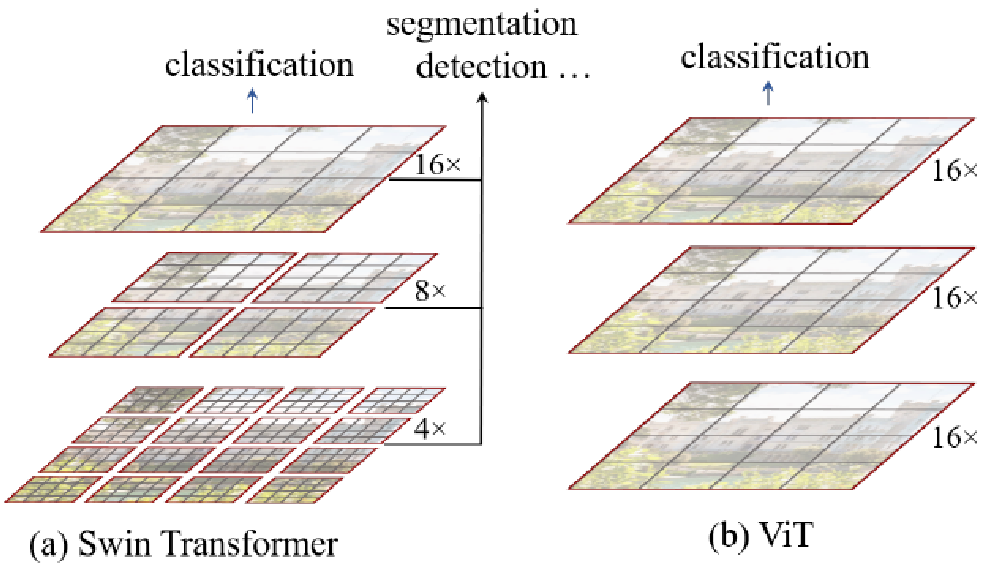
Output vectors: $\mathbf{c}_{:j} = \alpha_{1j}\mathbf{v}_{:1} + \cdots + \alpha_{mj}\mathbf{v}_{:m} = \mathbf{V}\boldsymbol{\alpha}_{:j}.$

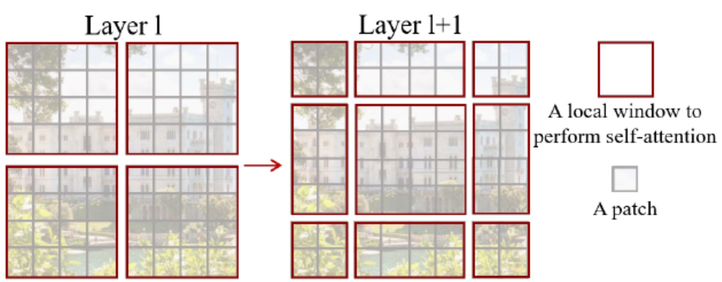


Courtesy of Shusen Wang

DNN Architectures- Transformers

- SWIN Transformer [Liu-2021] ("Shifted Windows")
  - Improvement of ViT transformer
    - data hungry (needs large set pretraining)
    - Image tokens to large – unsuitable for object detection, semantic segmentation
  - Hierarchical features
    - Self attention within windows (linear complexity w.r.t. image size)
    - Cross-window connection (cyclic window shifting in subsequent layers)

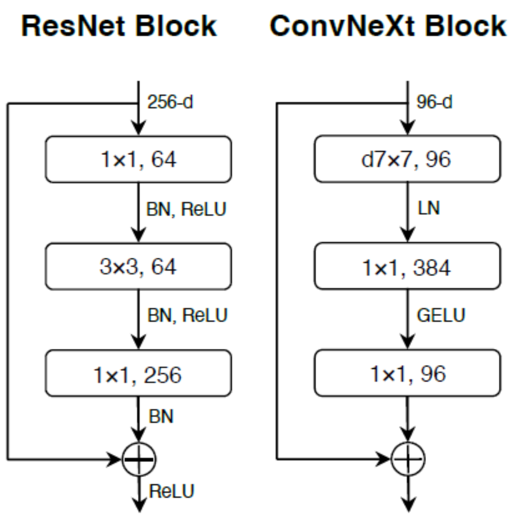


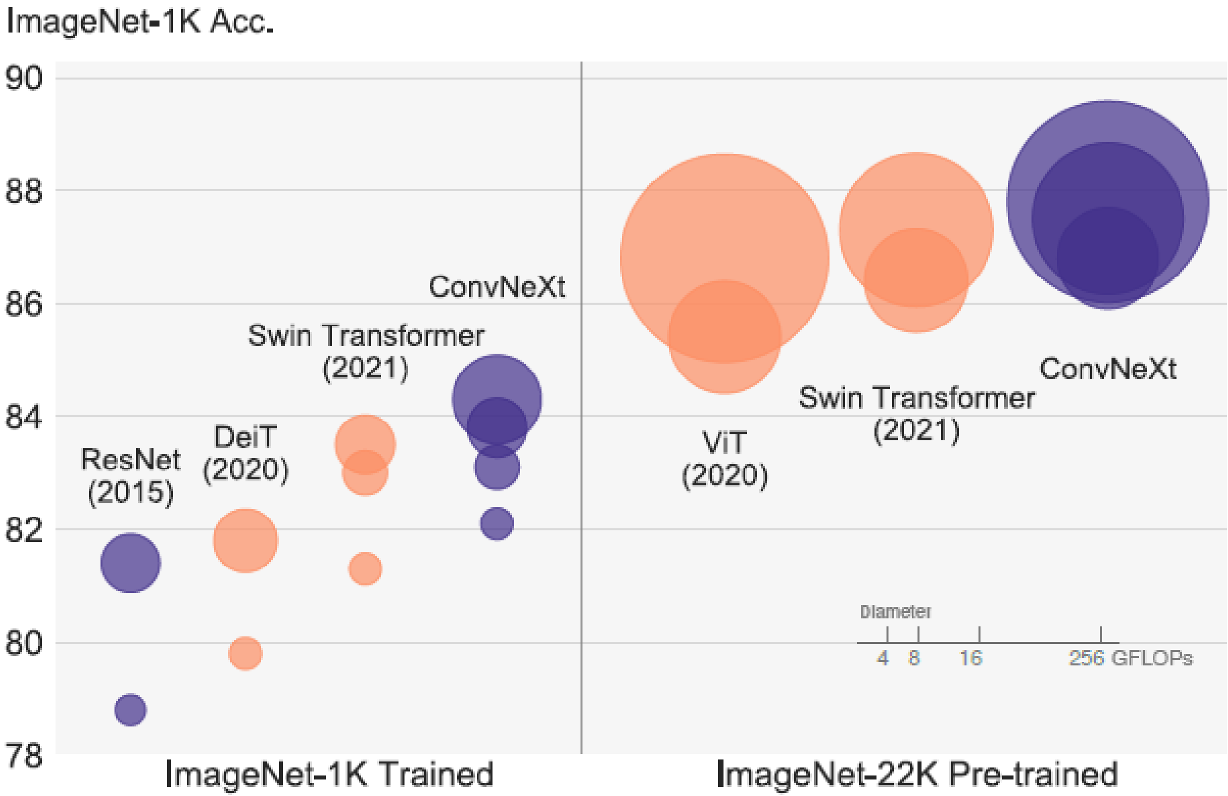


  - State-of-the-art general purpose backbone (recognition, detection, segmentation, ….)

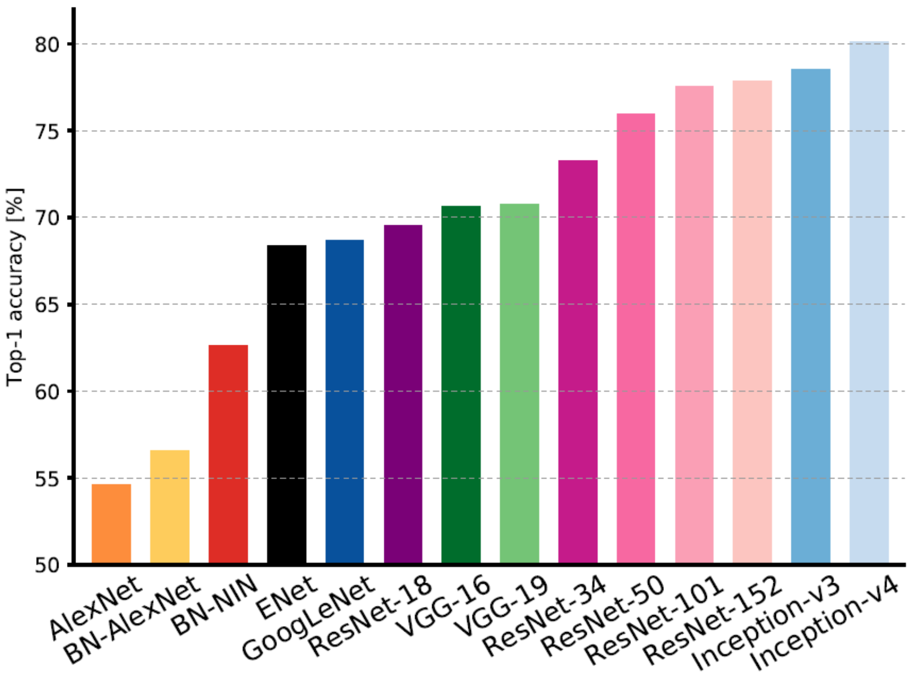
DNN Architectures – ConvNext

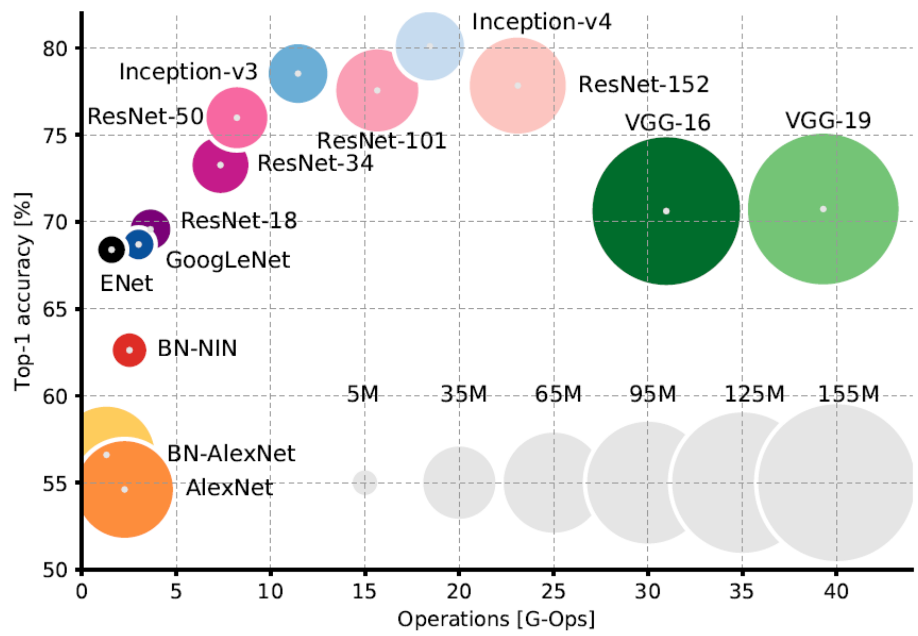
- ConvNeXt [Liu-2022]
  - Pure Convolutional Neural Network (again)
  - Similar to ResNet, but tweaked
  - Larger kernel size, BatchNorm -> LayerNorm
  - ReLU -> GeLU (smoother)





CNN models (comparison)





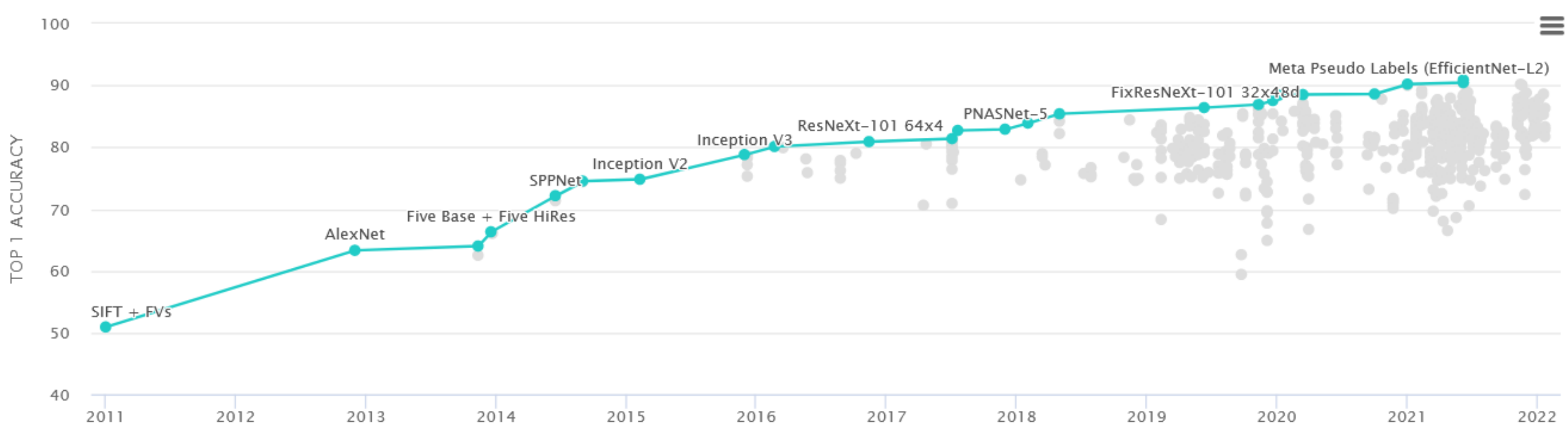
- [Canziani et al., [An Analysis of Deep Neural Network Models for Practical Applications](), 2017. arXiv:1605.07678v4]

CNN models (comparison)

- ImageNet leaderboard (Top-1 accuracy)

Leaderboard | Dataset

View Top 1 Accuracy by Date for All models



Rank	Model	Top 1 Accuracy	Top 5 Accuracy	Number of params	Extra Training Data	Paper	Code	Result	Year	Tags
1	**CoAtNet-7**	90.88%		2440M	✓	CoAtNet: Marrying Convolution and Attention for All Data Sizes			2021	Conv+Transformer JFT-3B
2	**ViT-G/14**	90.45%		1843M	✓	Scaling Vision Transformers			2021	Transformer JFT-3B
3	**CoAtNet-6**	90.45%		1470M	✓	CoAtNet: Marrying Convolution and Attention for All Data			2021	Conv+Transformer

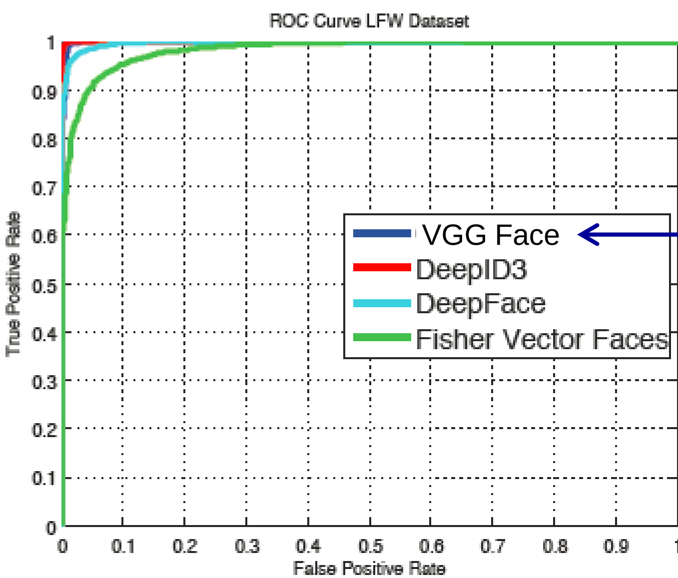
Face interpretation problems

- Face recognition, face verification
  - Architecture similar to AlexNet - very deep CNN (softmax at the last layer)

  [Taigman-ECVV-2014] DeepFace: Closing the Gap to Human-Level Performance in Face Verification (authors from Facebook)

  [Parkhi-BMVC-2015] Deep Face recognition (authors from Oxford Uni)

  - 2.6M images of 2.6k celebrities, trained net available

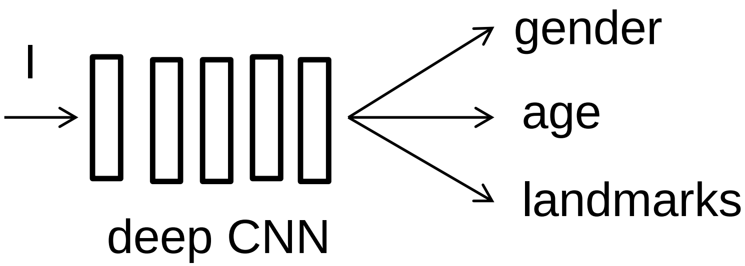


No.	Method	# Training Images	# Networks	Accuracy
1	Fisher Vector Faces	-	-	93.10
2	DeepFace (Facebook)	4 M	3	97.35
3	DeepFace Fusion (Facebook)	500 M	5	98.37
4	DeepID-2,3	Full	200	99.47
5	FaceNet (Google)	200 M	1	98.87
6	FaceNet+ Alignment (Google)	200 M	1	99.63
7	(VGG Face)	2.6 M	1	98.78

- Face represented by penultimate layer response, similarity search, large scale indexing

Face interpretation problems

- Facial landmarks, Age / Gender estimation
  - Multitask network
    - Shared representation
    - Combination of both classification and regression problems



Age estimation – How good the network is?

- Our survey

  ~20 human subjects , ~100 images of 2 datasets

MORPH dataset

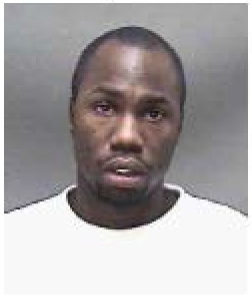
True: 22, MAE: 18.8

True: 36, MAE: 17.8

True: 33, MAE: 16.3

True: 22, MAE: 16.1

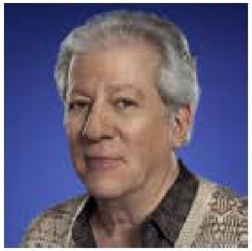
True: 25, MAE: 16.0



IMDB dataset

True: 25, MAE: 0.5

True: 66, MAE: 1.0



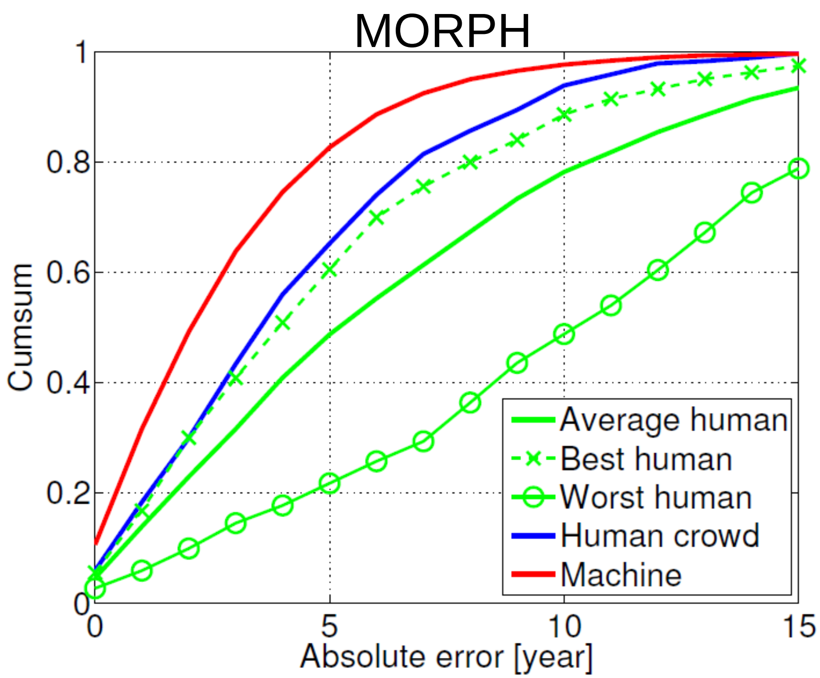
True: 29, MAE: 1.0

True: 19, MAE: 1.0

True: 43, MAE: 1.0

Age estimation – How good the network is?

- Better than average human…



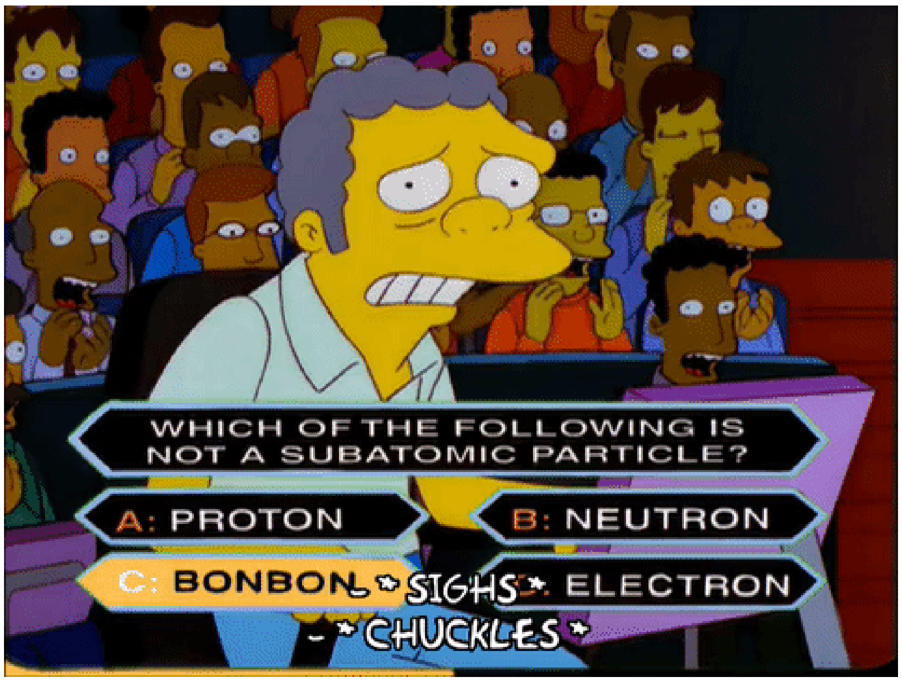
	MAE	CS5	MaxAE
Average human :	6.8	48.6	24.1
Human crowd :	4.7	65.1	19.0
Machine :	3.2	82.6	26.0

	MAE	CS5	MaxAE
Average human :	8.2	41.7	31.5
Human crowd :	5.7	59.0	21.0
Machine :	5.1	62.5	42.7

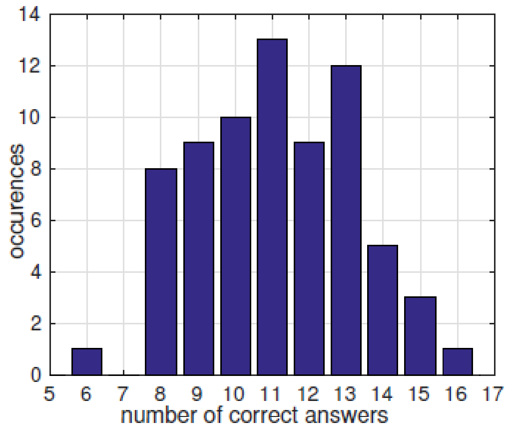
- [Franc-Cech-IVC-2018]
- Network runs real-time on CPU

Predicting Decision Uncertainty from Faces

- [Jahoda, Vobecky, Cech, Matas. *Detecting Decision Ambiguity from Facial Images*. In Face and Gestures, 2018]
- Can we train a classifier to detect uncertainty?



Training set: 1,628 sequences
Test set: 90 sequences

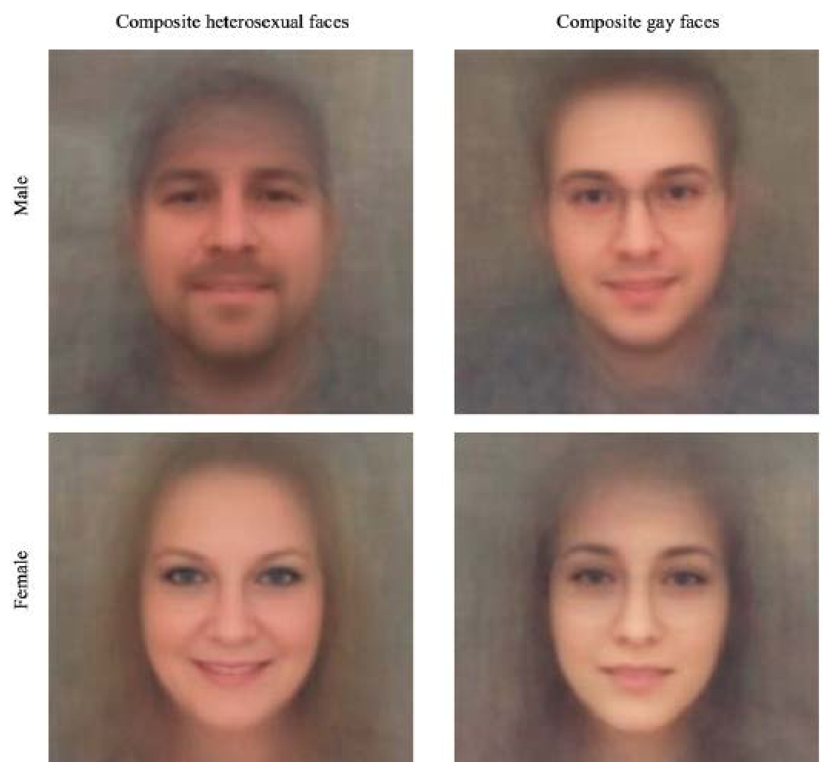


=> YES, we can…

- CNN 25% error rate, while human volunteers 45%

Sexual Orientation from Face Images

- [Wang and Kosinki. *Deep neural networks are more accurate than humans at detecting sexual orientation from facial images*. Journal of Personality and Social Psychology, 2018]
- Better accuracy than human in (gay vs. heterosexual)
  - 81% accuracy (for men), average human accuracy (61%)
  - 71% accuracy (for women) average human accuracy (54%)
  - Accuracy further improved if 5 images provided (91%, 83%)



General recipe to use deep neural networks

- Recipe to use deep neural network to “solve any problem” (G. Hinton 2013)
  - Have a deep net
  - If you do not have enough labeled data, pre-train it by unlabeled data; otherwise do not bother with pre-initialization
  - Use rectified linear units instead of standard neurons (sigmoid)
  - Use dropout to regularize it (you can have many more parameters than training data)
  - If there is a spatial structure in your data, use convolutional layers

- Novel:
  - Use Batch Normalization [Ioffe-Szegedy-NIPS-2015]
  - ReLU => ELU, GELU
  - Adaptive Optimizers (ADAM)
  - Various architectures (AlexNet, VGG, GoogLeNet, ResNet, ResNeXt, DenseNet, SE-Net, MobileNet, ShuffleNet, Transformers, Swin, ConvNext)

- Experience:
  - Data matters (the more data the better), transfer learning, data augmentation

Conclusions

- DNNs efficiently learns the abstract representation
- Low computational demands for running, Training needs GPU
- Many “deep” toolboxes: Caffe (Berkeley), MatconvNet (Oxford), TensorFlow (Google), Theano (Montreal), **PyTorch** (Facebook), …
- NNs are (again) in the “Golden Age” (or witnessing a bubble), as many practical problems seem solvable in near future
- Explosion of interest of DNN in literature, graduates get incredible offers, start-ups appear all the time

- Do we understand enough what is going on?

http://www.youtube.com/watch?v=LVLoc6FrLi0

Human_Abducted_by_UFO.mp4

Acknowledgement: I borrowed some images from slides and blogs of G. Hinton, A. Ng, Y. Le Cun, Fei-Fei Li, K. He, J. Brownlee, K. Rupp, Shusen W

Further Resources

- Deep Learning Textbook
  - Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press, 2016
  - Available on-line for free.

- Lectures / video-lectures
  - Stanford University course on Deep Learning (cs231n)
  - MIT lectures on Introduction in Deep Learning (MIT 6.S191)

- Various blogs and on-line journals
  - Google AI blog (https://ai.googleblog.com/)
  - OpenAI blog (https://openai.com/blog)
  - MetaAI blog (https://ai.facebook.com/blog/)
  - Andrej Karpathy (blog)
  - …