ImageNet Classification with Deep Convolutional Neural Networks

Xiao'ao Song
Kevin Steijn

Outline

- Introduction
- Network Architecture
- Learning
- Reduce Overfitting
- Results
- Discussion
- References

University of Zurich[UZH]

Department of Informatics

Introduction

Statistics (as of 11-March-2020)

Paper published in the year 2012

It has been cited 58,432 times

Authors:

Alex Krizhevsky - 92,686 citations

Ilya Sutskever - 157,906 citations

Geoffrey Hinton - 354,732 citations

“AlexNet” refers to this paper

Dataset - ImageNet

14,197,122 images

21,841 synsets indexed

https://karpathy.github.io/assets/cnntsne.jpeg

“ *Each meaningful concept in WordNet, possibly described by multiple words or word phrases, is called a "synonym set" or "synset"* ”

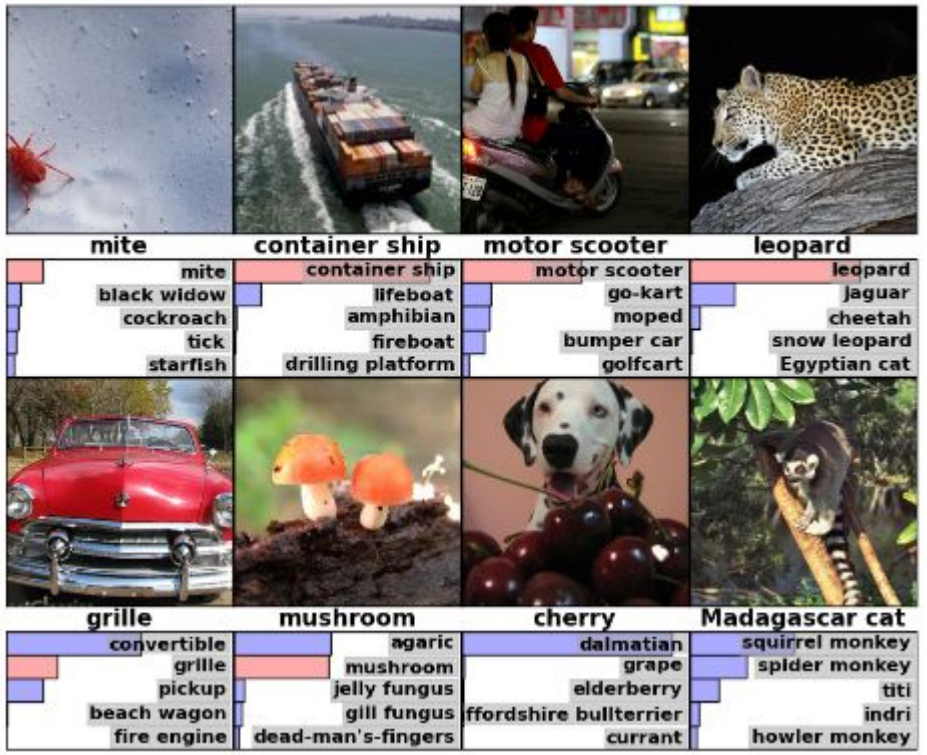
http://image-net.org/about-overview

“ *WordNet® is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.* ”

https://wordnet.princeton.edu/

Downsampled to 256x256 fixed resolution

Goal - Classification



ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)

~ 1.2 million training images

50,000 validation images

150,000 testing images

Classification - Make 5 guesses about the image label

ILSVRC-2010 was used to perform the experiments

This model won the ILSVRC-2012 competition by a large margin (~10.8%)

http://www.image-net.org/challenges/LSVRC/2012/results.html

Error Metrics

Top-5

Top-5 error rate is the fraction of test images for which the correct label is not among the five labels considered most probable by the model.

Top-1

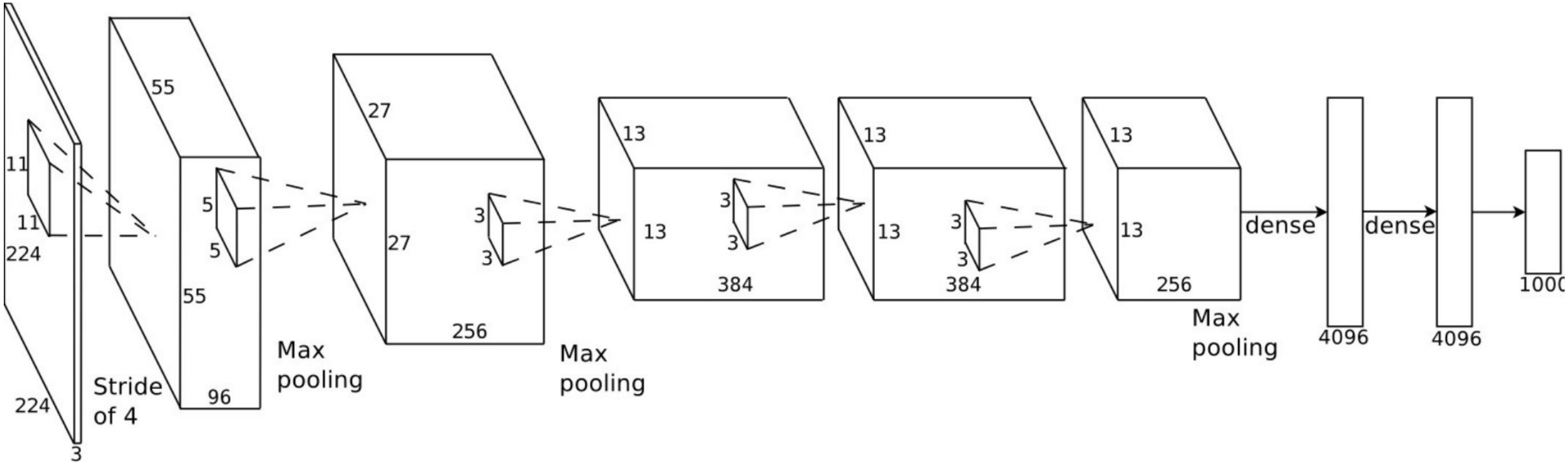
Top-1 error rate is the fraction of test images for which the correct label is not the label considered most probable by the model.

University of Zurich[UZH]

Department of Informatics

Network Architecture

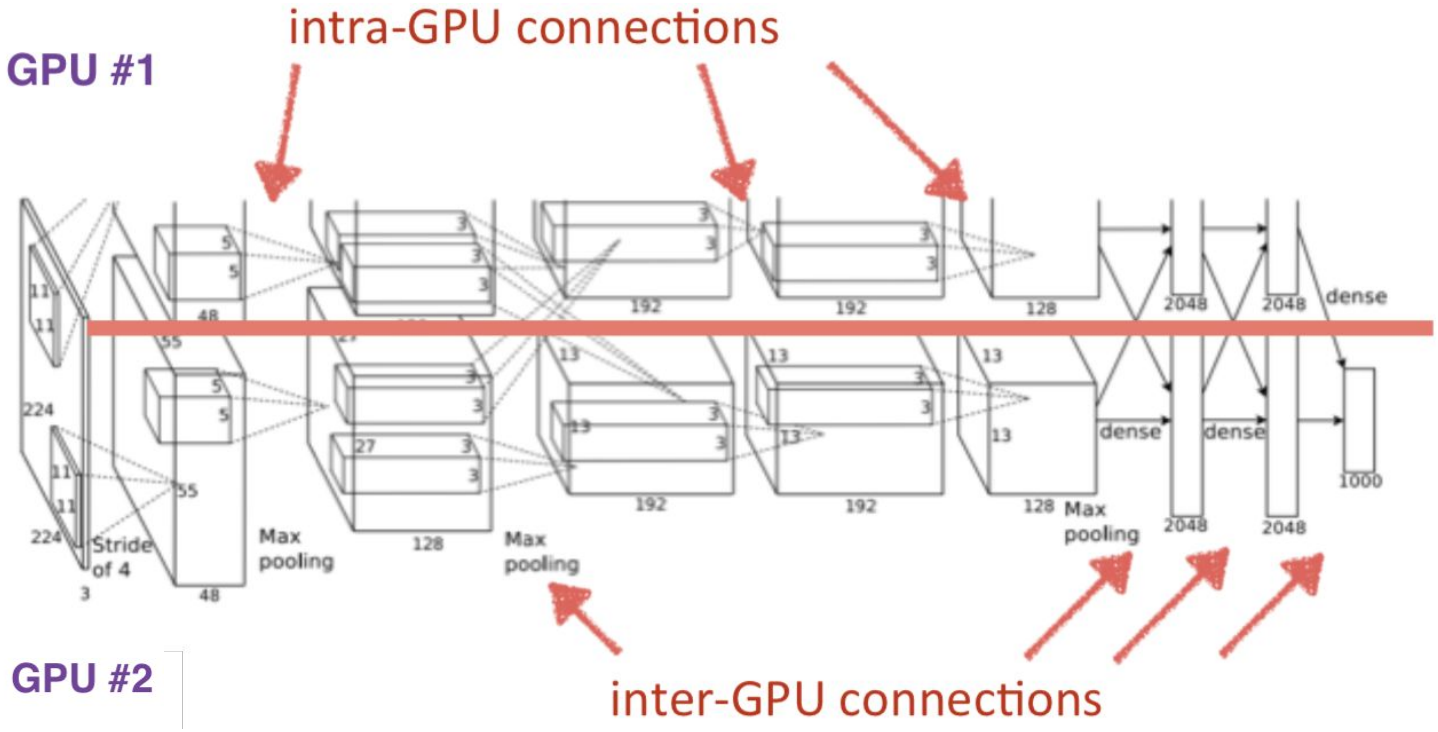
Architecture



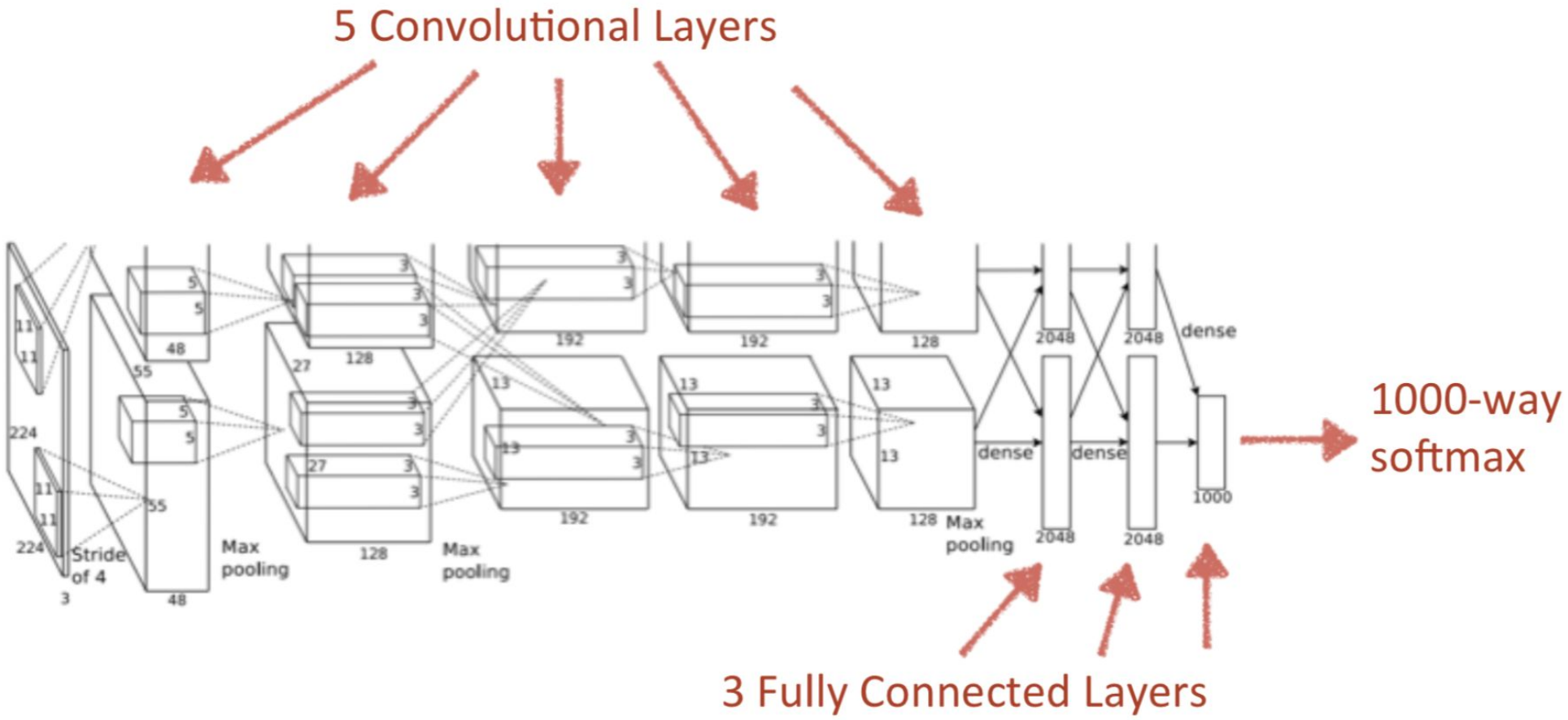
http://image-net.org/challenges/LSVRC/2012/supervision.pdf

Training on Multiple GPUs

Top-1 and Top-5 error rates decreases by 1.7% & 1.2% respectively, comparing to the net trained with one GPU and half neurons!!

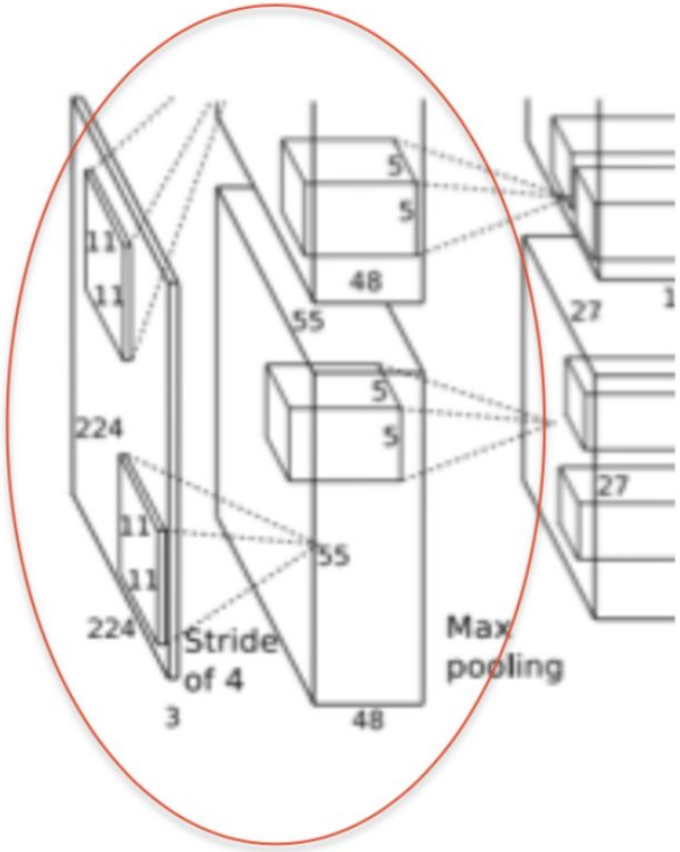


Architecture



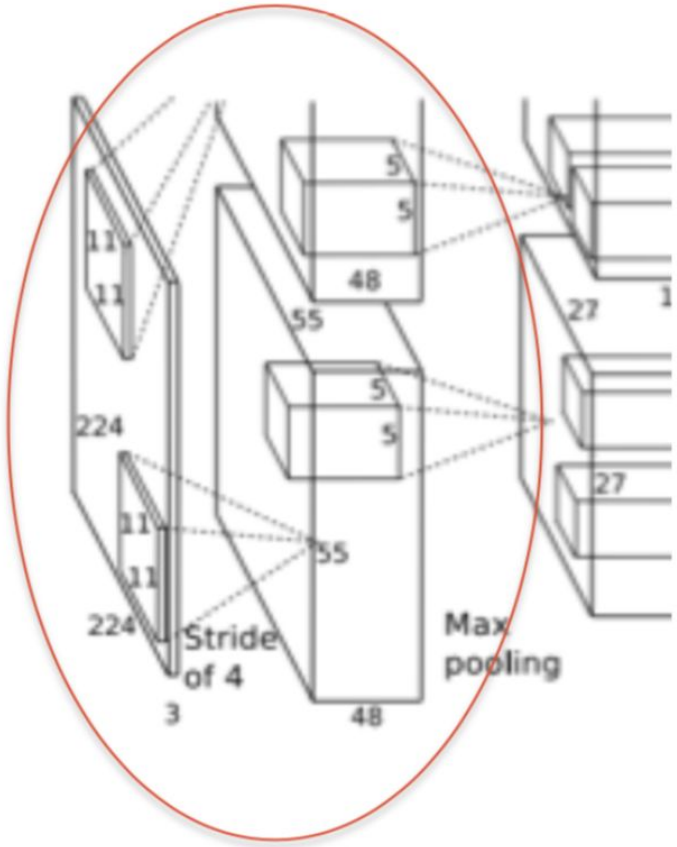
http://image-net.org/challenges/LSVRC/2012/supervision.pdf

Layer 1 (Convolutional)



- Images: 224x224x3
- F (receptive field size): 11
- S (stride) = 4
- Conv layer output: 55x55x96

Layer 1 (Convolutional)



- 55*55*96 = 290,400 neurons
- each has 11*11*3 = 363 weights and 1 bias
- 290400 * 364 = 105,705,600 paramaters on the first layer of the AlexNet alone!

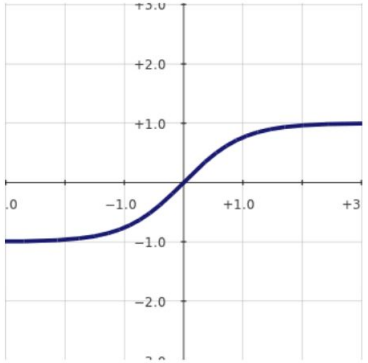
Rectified Linear Unit (ReLU) Nonlinearity

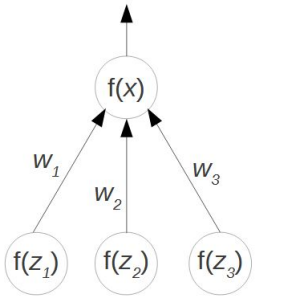
- **Standard way to model a neuron**

$$f(x) = tanh(x) \quad \text{or} \quad f(x) = (1 + e^{-x})^{-1}$$

- **Non-saturating nonlinearity (ReLU)**

f(x) = tanh(x)

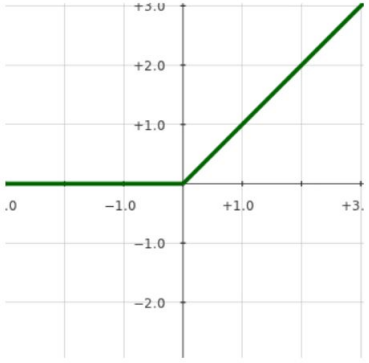




$x = w_1 f(z_1) + w_2 f(z_2) + w_3 f(z_3)$

x is called the total input to the neuron, and f(*x*) is its output

f(x) = max(0, x)



Very bad (slow to train)

Very good (quick to train)

http://image-net.org/challenges/LSVRC/2012/supervision.pdf

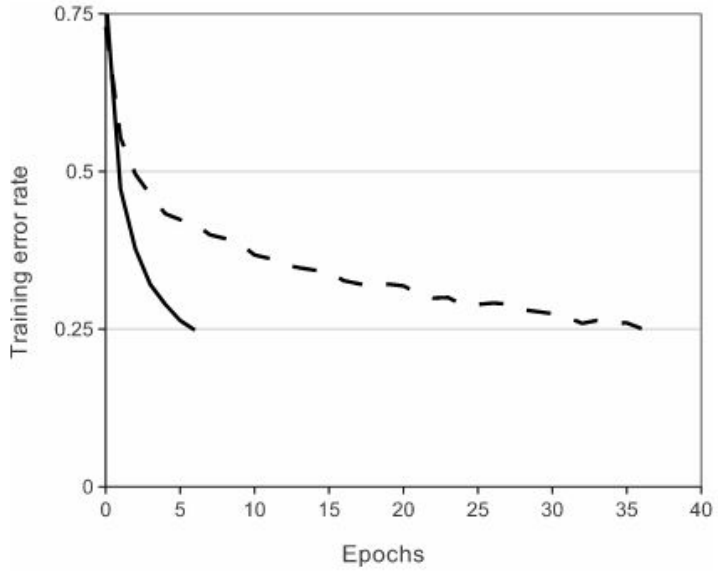


Figure 1: A four-layer convolutional neural network with ReLUs (**solid line**) reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons (**dashed line**). The learning rates for each net-

Local Response Normalization

- **No need to input normalization with ReLUs.**
- **But still the following local normalization scheme helps generalization.**

$$b^i_{x,y} = a^i_{x,y} \Big/ \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$

Response-normalized activity

Activity of a neuron computed by applying kernel I at position (x,y) and then applying the ReLU nonlinearity

- **Response normalization reduces top-1 and top-5 error rates by 1.4% and 1.2% , respectively.**

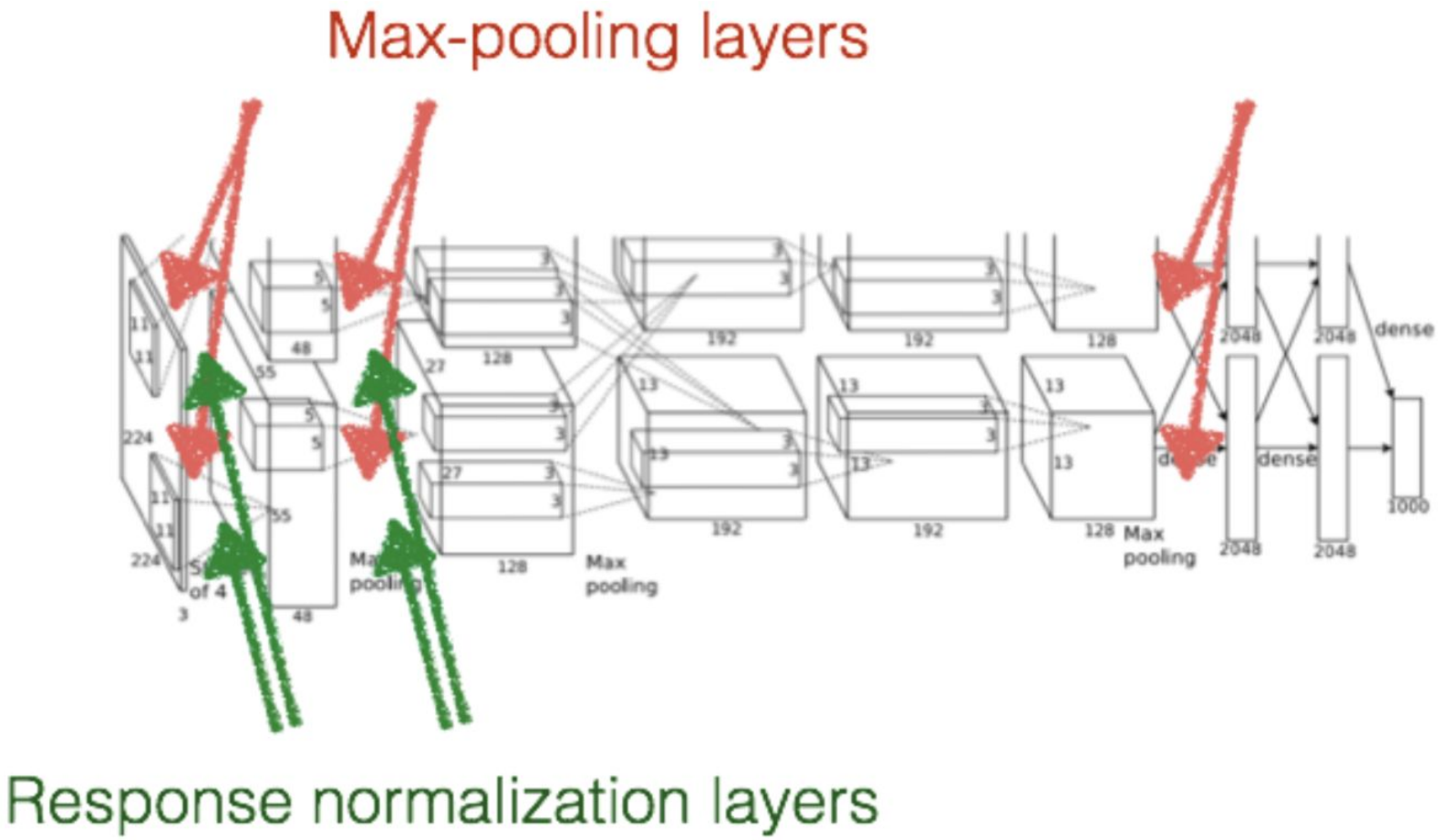
3.3 Local Response Normalization

ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. However, we still find that the following local normalization scheme aids generalization. Denoting by $a^i_{x,y}$ the activity of a neuron computed by applying kernel i at position (x, y) and then applying the ReLU nonlinearity, the response-normalized activity $b^i_{x,y}$ is given by the expression

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$

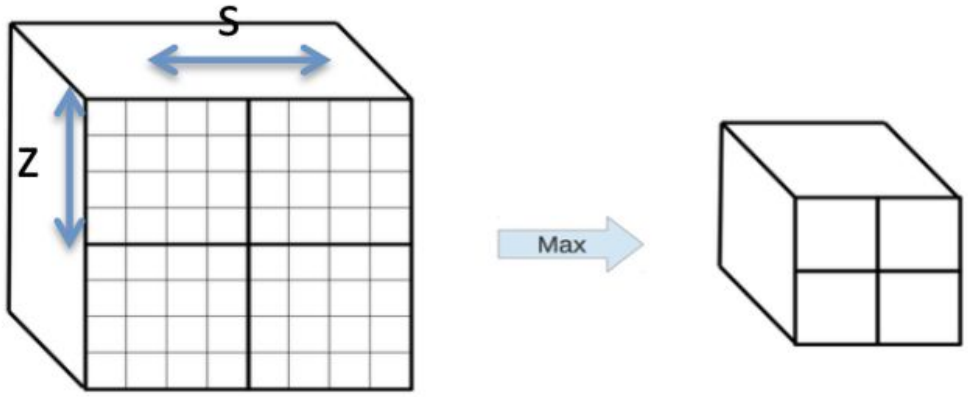
where the sum runs over n "adjacent" kernel maps at the same spatial position, and N is the total number of kernels in the layer. The ordering of the kernel maps is of course arbitrary and determined before training begins. This sort of response normalization implements a form of lateral inhibition inspired by the type found in real neurons, creating competition for big activities amongst neuron outputs computed using different kernels. The constants k, n, α, and β are hyper-parameters whose values are determined using a validation set; we used $k = 2$, $n = 5$, $\alpha = 10^{-4}$, and $\beta = 0.75$. We applied this normalization after applying the ReLU nonlinearity in certain layers (see Section 3.5).

Overlapping Pooling



Overlapping Pooling

- **Traditional pooling (s = z)**

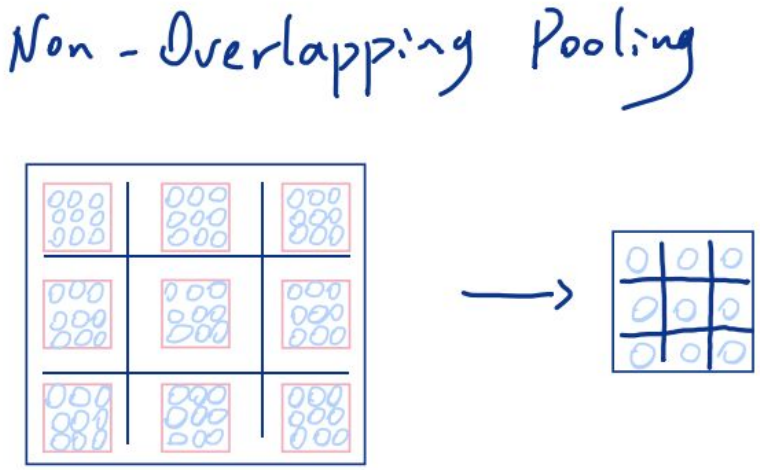


- **s < z ➡ overlapping pooling**
  - ➢ *In paper, s = 2, z = 3*

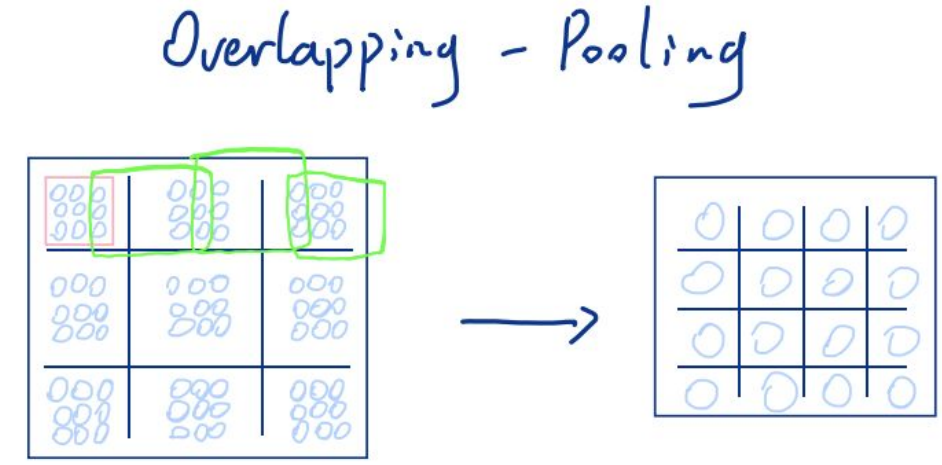
- **top-1 and top-5 error rates decrease by 0.4% and 0.3%, respectively, compared to the non-overlapping scheme s = 2, z = 2**

Overlapping Pooling

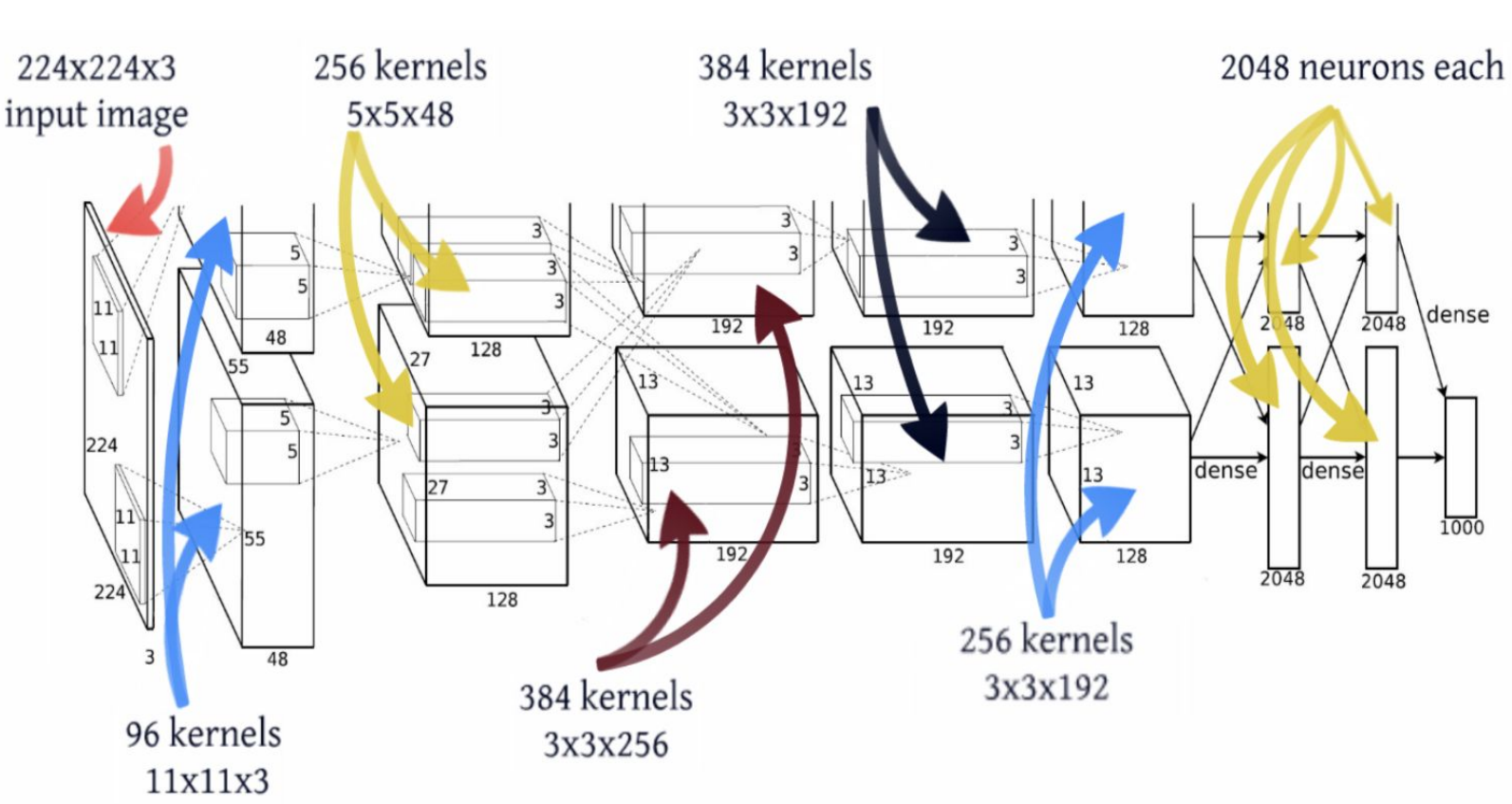
- **Traditional pooling (s = z)**



- **s < z: overlapping pooling**
  - ➢ *In paper, s = 2, z = 3*



Architecture Overview



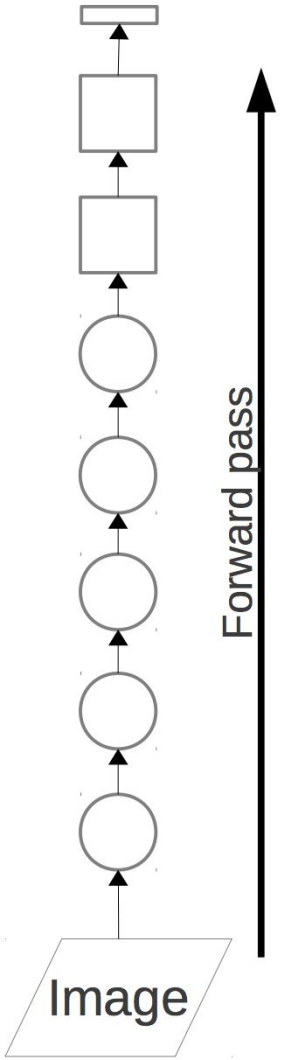
4M	FULL CONNECT	4Mflop
16M	FULL 4096/ReLU	16M
37M	FULL 4096/ReLU	37M
	MAX POOLING	
442K	CONV 3x3/ReLU 256fm	74M
1.3M	CONV 3x3ReLU 384fm	224M
884K	CONV 3x3/ReLU 384fm	149M
	MAX POOLING 2x2sub	
	LOCAL CONTRAST NORM	
307K	CONV 5x5 /ReLU 256fm	223M
	MAX POOL 2x2sub	
	LOCAL CONTRAST NORM	
35K	CONV 11x11/ReLU 96fm	105M

University of Zurich[UZH]

Department of Informatics

Learning

Training



Forward pass

Using stochastic gradient descent and the *backpropagation algorithm* (just repeated application of the chain rule)

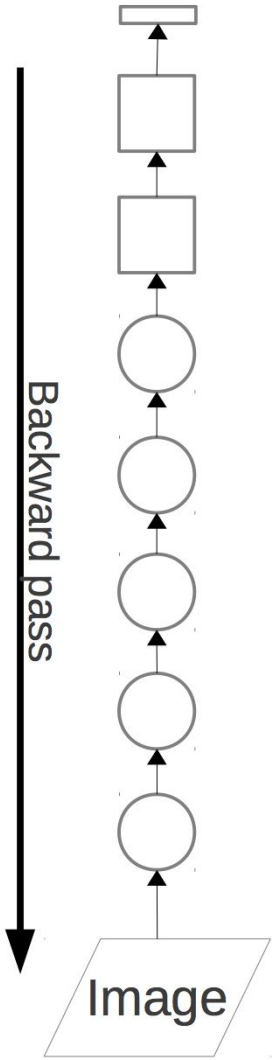
One output unit per class

x_i = total input to output unit i

$$f(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{1000} \exp(x_j)}$$

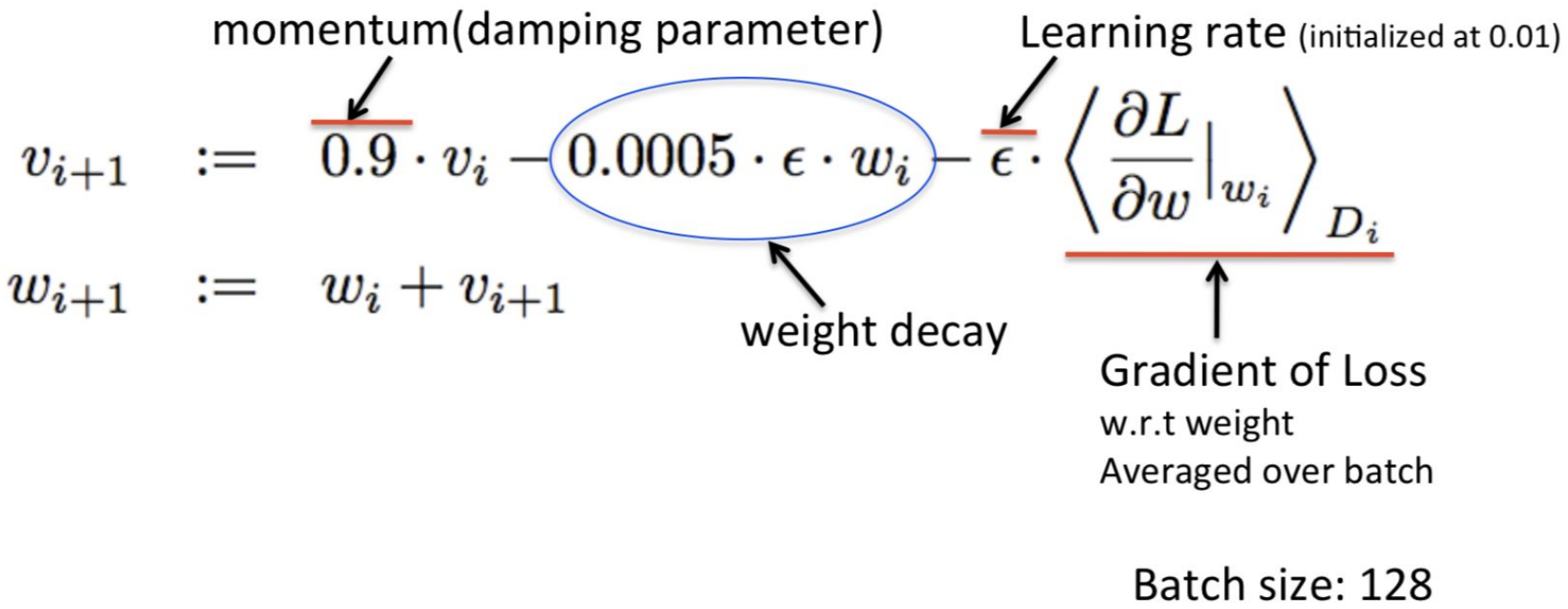
We maximize the log-probability of the correct label, $\log f(x_t)$

Backward pass



Stochastic Gradient Descent Learning

- **Momentum Update**



- **The training took 5 to 6 days on two NVIDIA GTX 580 3GB GPUs.**

University of Zurich[UZH]

Department of Informatics

Reducing Overfitting

Data Augmentation

Image translations and horizontal reflections

Sample 224x224 patches from their 256x256 images for this step

Increase in size of training set of factor 2048

http://image-net.org/challenges/LSVRC/2012/supervision.pdf

Changing intensity of the RGB channels

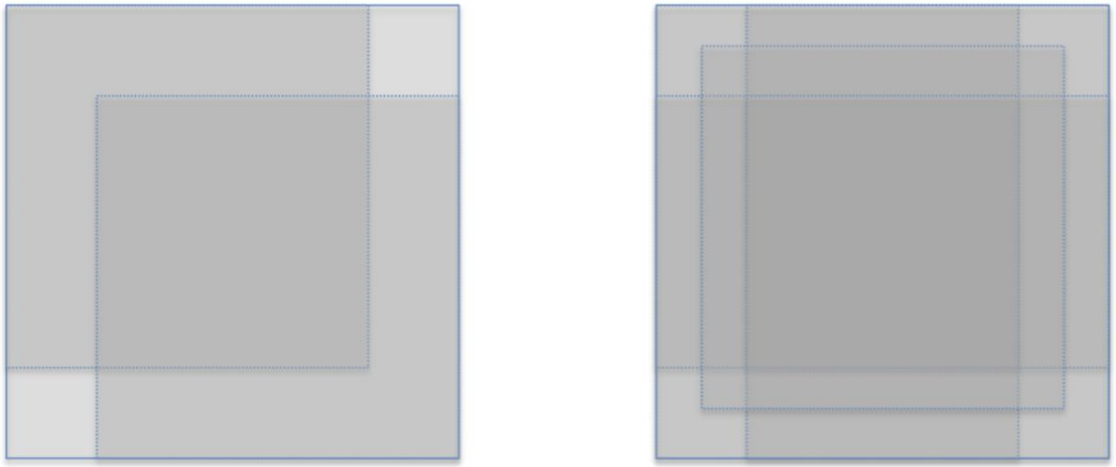
PCA on RGB pixel values

Add multiples of the found principal components to each training image

Reduces the top-1 error rate by more than 1%

Reducing Overfitting

- **Data Augmentation**
  - 60 million parameters, 650,000 neurons
    - Overfits a lot.
  - Crop 224x224 patches (and their horizontal reflections.)
  - At test time, average the predictions on the 10 patches



Reducing Overfitting

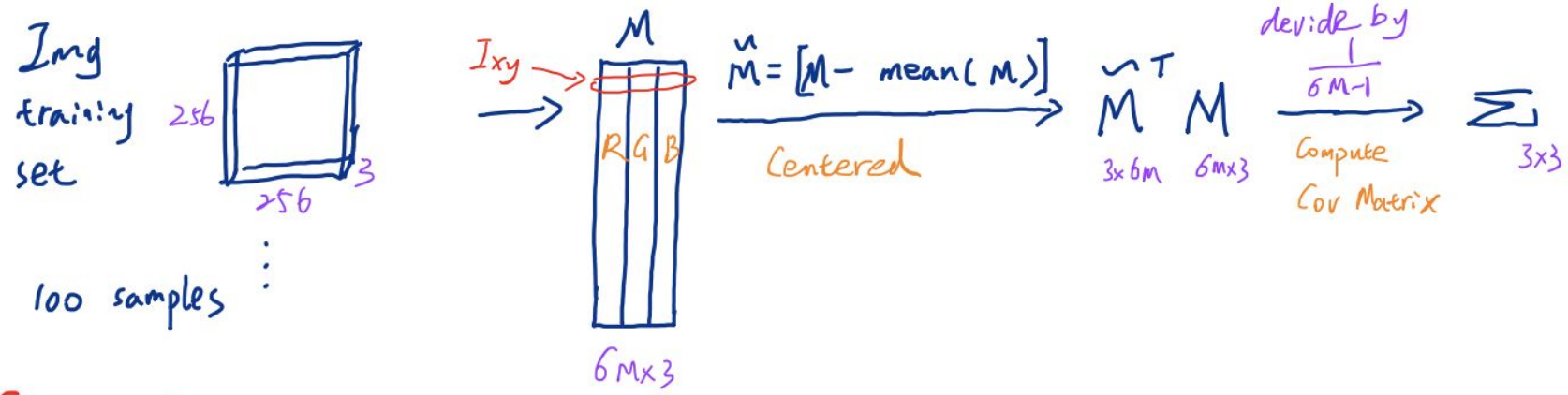
- **Data Augmentation**
  - **Change the intensity of RGB channels**

$$I_{xy} = [I_{xy}^R, \ I_{xy}^G, \ I_{xy}^B]^T$$

  - **Add multiples of principle components**

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$

$$\alpha_i \sim N(0, \ 0.1)$$



In paper

calculate Eigen-value and Eigen-vetor :

$$\Sigma p = \lambda p$$

$$(\Sigma - \lambda I) p = 0$$

find $\det|\Sigma - \lambda I| = 0$ →

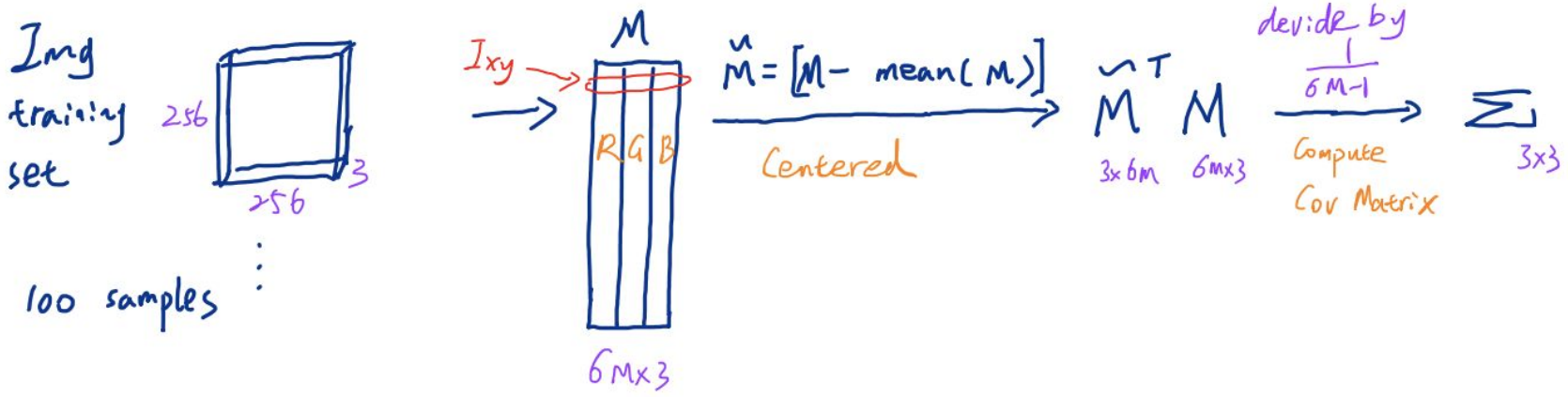
λ_1, P_1

λ_2, P_2

λ_3, P_3

$$I_{xy}' = I_{xy} + [P_1, P_2, P_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T, \text{ where } \alpha_i \sim N(0, 0.1)$$

$$\underset{3\times1}{\begin{bmatrix} I_R \\ I_G \\ I_B \end{bmatrix}} + \underset{3\times3}{\begin{bmatrix} P_{1R} & P_{2R} & P_{3R} \\ P_{1G} & P_{2G} & P_{3G} \\ P_{1B} & P_{2B} & P_{3B} \end{bmatrix}} \underset{3\times1}{\begin{bmatrix} \alpha_1\lambda_1 \\ \alpha_2\lambda_2 \\ \alpha_3\lambda_3 \end{bmatrix}}$$



Alternatively: drawn r from $N(0, \Sigma_{3x3})$

$$I_{xy}' = I_{xy} + r$$

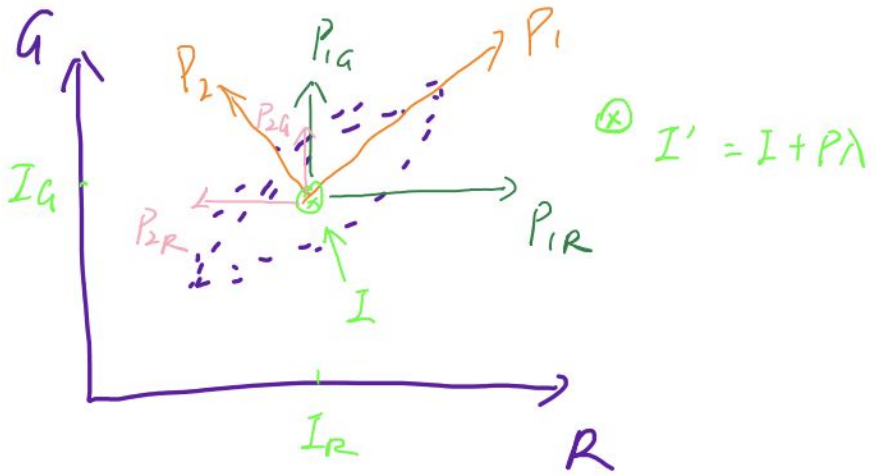
* simpler but similar performance

2D example

Interpretation of PCA

$$\vec{p}_1 = \begin{pmatrix} p_{1R} \\ p_{1G} \end{pmatrix}, \quad \vec{p}_2 = \begin{pmatrix} p_{2R} \\ p_{2G} \end{pmatrix}$$

$$I + P\lambda = I + \begin{pmatrix} p_{1R} & p_{2R} \\ p_{1G} & p_{2G} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} I_R \\ I_G \end{pmatrix} + \begin{pmatrix} p_{1R}\lambda_1 + p_{2R}\lambda_1 \\ p_{1G}\lambda_1 + p_{2G}\lambda_2 \end{pmatrix}$$

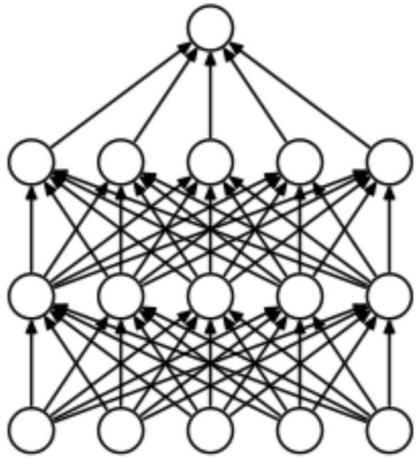


Reducing Overfitting

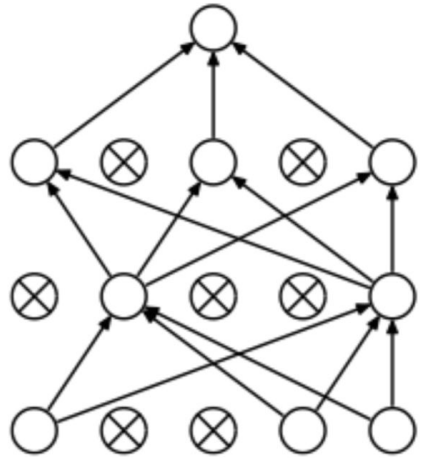
Dropout

Each training stage all nodes are included with probability $p = 0.5$

Cost is a factor of 2 during training



Standard Neural Net



After applying dropout.

- With probability 0.5
- last two 4096 fully-connected layers.

Figure credit from Srivastava et al.

Reducing Overfitting

- **Softmax**

$$L = \frac{1}{N}\sum_i -\log\left(\frac{e^{f_{y_j}}}{\sum_j e^{f_j}}\right) + \lambda \sum_k \sum_l W_{k,l}^2$$

$j = 1...1000$

$P(y_i \mid x_i; W)$ Likelihood

University of Zurich[UZH]

Department of Informatics

Results

Error Metrics

Top-5

Top-5 error rate is the fraction of test images for which the correct label is not among the five labels considered most probable by the model.

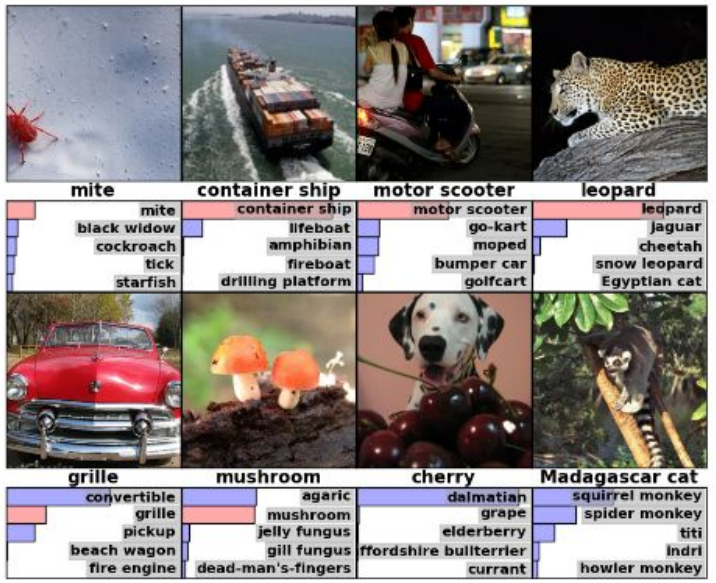
Top-1

Top-1 error rate is the fraction of test images for which the correct label is not the label considered most probable by the model.

ILSVRC-2010

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.



[2] A. Berg, J. Deng, and L. Fei-Fei.Large scale visual recognition challenge 2010. www.image-net.org/challenges. 2010.

[24] . Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification.InComputer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pages 1665–1672. IEEE,2011.

ILSVRC-2012 Results

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

ILSVRC-2012

SuperVision corresponds to the model being presented

- 7 hidden “weight” layers
- 650K neurons
- 60M parameters
- 630M connections

- Rectified Linear Units
- Overlapping pooling
- Dropout trick
- Randomly extracted 224x224 patches for more data

Task 1

Team name	Filename	Error (5 guesses)	Description
SuperVision	test-preds-141-146.2009-131-137-145-146.2011-145f.	0.15315	Using extra training data from ImageNet Fall 2011 release
SuperVision	test-preds-131-137-145-135-145f.txt	0.16422	Using only supplied training data
ISI	pred_FVs_wLACs_weighted.txt	0.26172	Weighted sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively.
ISI	pred_FVs_weighted.txt	0.26602	Weighted sum of scores from classifiers using each FV.
ISI	pred_FVs_summed.txt	0.26646	Naive sum of scores from classifiers using each FV.

Task 2

Team name	Filename	Error (5 guesses)	Description
SuperVision	test-rect-preds-144-cloc-141-146.2009-131-137-145-	0.335463	Using extra training data for classification from ImageNet Fall 2011 release
SuperVision	test-rect-preds-144-cloc-131-137-145-135-145f.txt	0.341905	Using only supplied training data
OXFORD_VGG	test_adhocmix_detection.txt	0.500342	Re-ranked DPM detection over Mixed selection from High-Level SVM scores and Baseline Scores, decision is performed by looking at the validation performance
OXFORD_VGG	test_finecls_detection_bestbbox.txt	0.50139	Re-ranked DPM detection over High-Level SVM Scores
OXFORD_VGG	test_finecls_detection_firstbbox.txt	0.522189	Re-ranked DPM detection over High-Level SVM Scores - First bbox selection heuristic

University of Zurich[UZH]

Department of Informatics

Discussion

Depth is really important

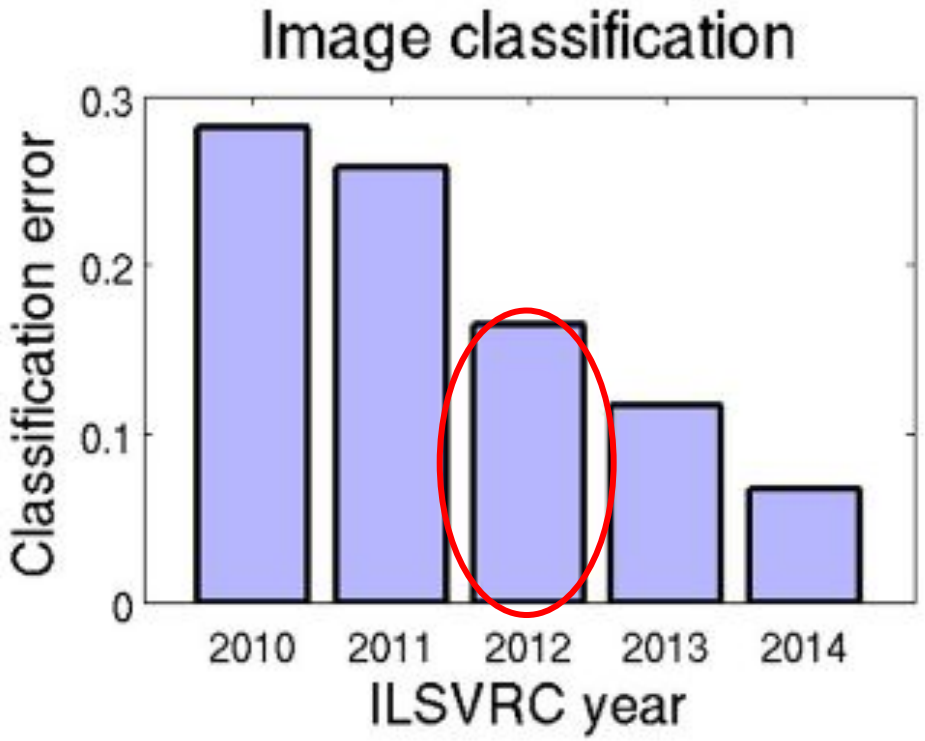
- **Removing a single convolutional layer degrades the performance.**

- **In later research:**

K. Simonyan, A. Zisserman.
Very Deep Convolutional Networks for Large-Scale Image Recognition. Technical report, 2014.

→ 16-layer model, 19-layer model. 7.3% top-5 test error on ILSVRC-2012

Impact



Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115.3 (2015): 211-252.

2013

Convolutional Neural Network

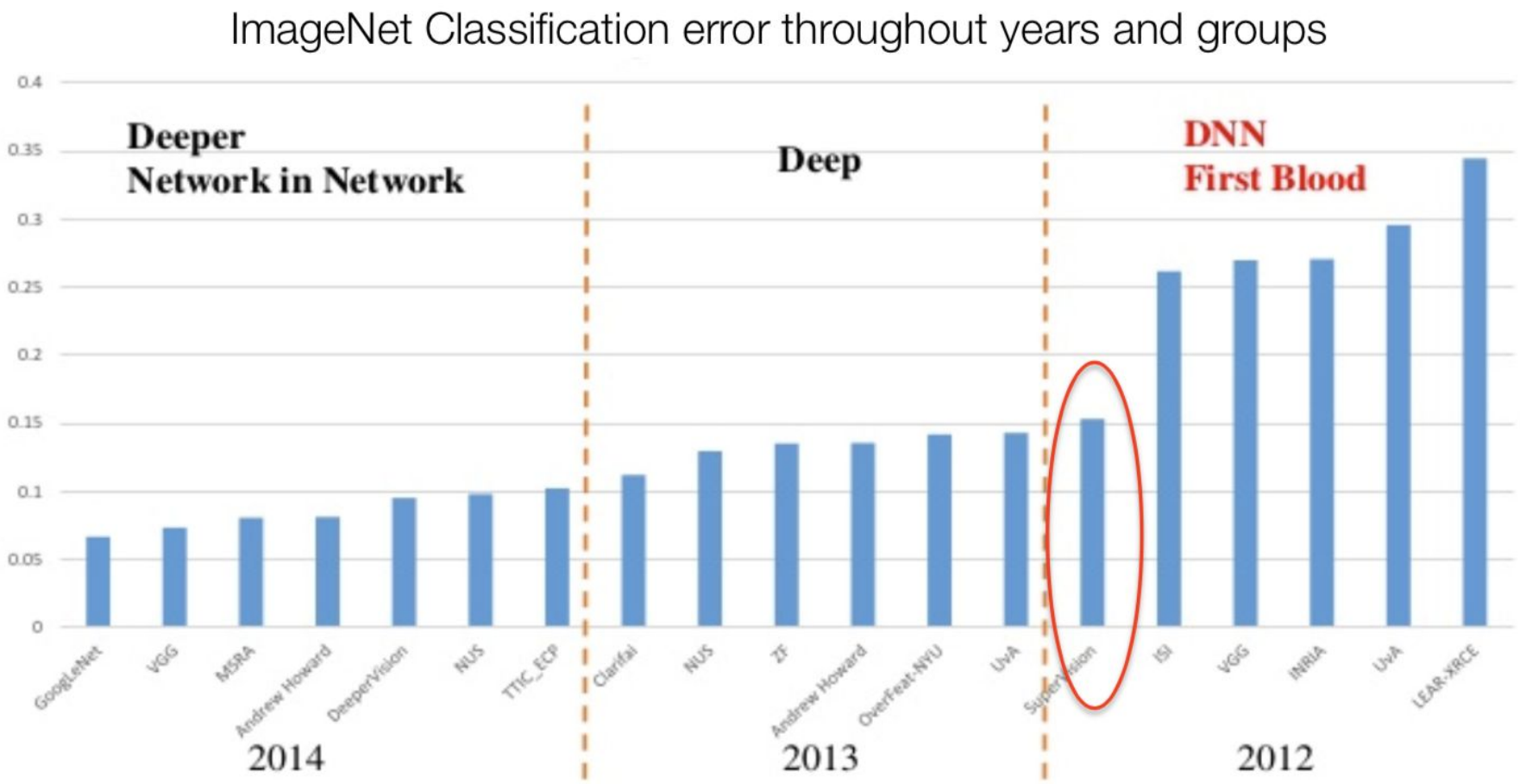
Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.

2014

Convolutional Neural Network

Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Impact


ImageNet Classification error throughout years and groups
Deeper
Network in Network
Deep
DNN
First Blood
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
GoogLeNet
VGG
MSRA
Andrew Howard
DeepVision
NUS
TTIC_ECP
Clarifai
NUS
ZF
Andrew Howard
OverFeat-NYU
UvA
Supervision
ISI
VGG
INRIA
UvA
LEAR-XRCE
2014
2013
2012

Adding Layers

Removing a layer from the middle of the CNN results in a loss of the top-1 error rate of 2%

Adding more layers improves performance?

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

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**
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**
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**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Adding Layers

Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

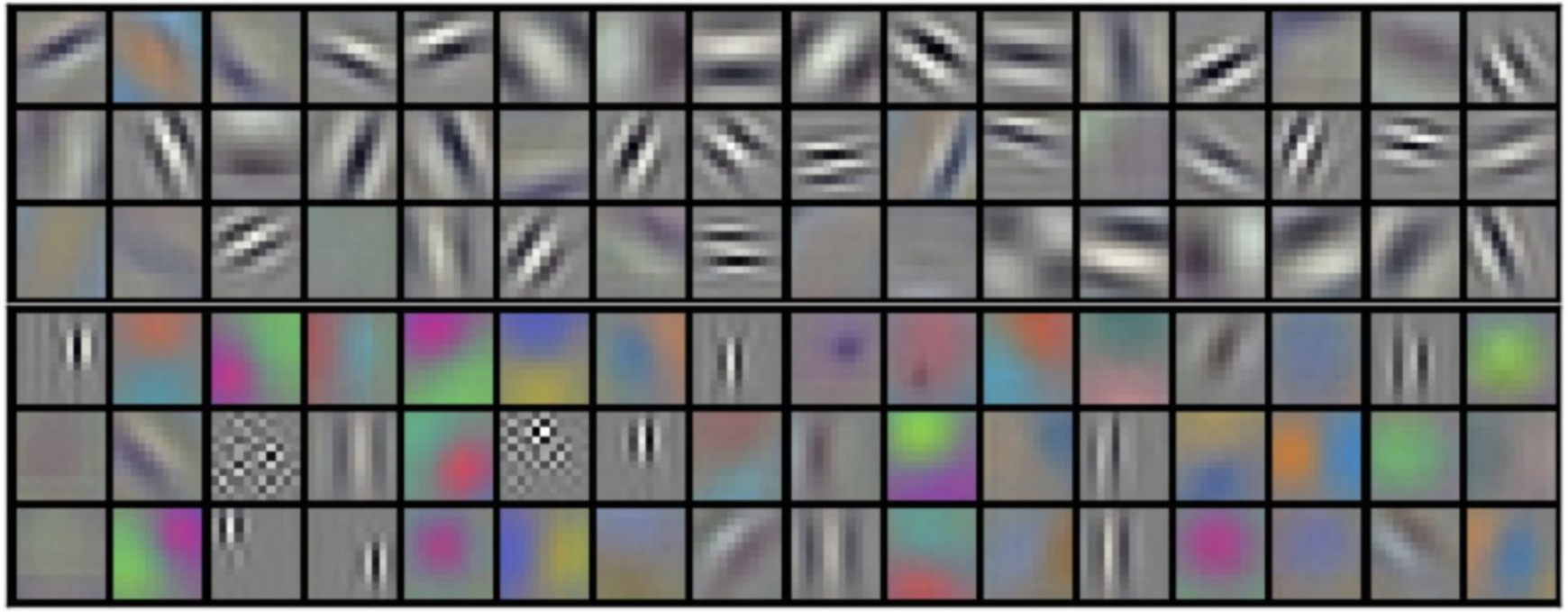
Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Going Deeper

Picture from Kaiming He

ICCV15

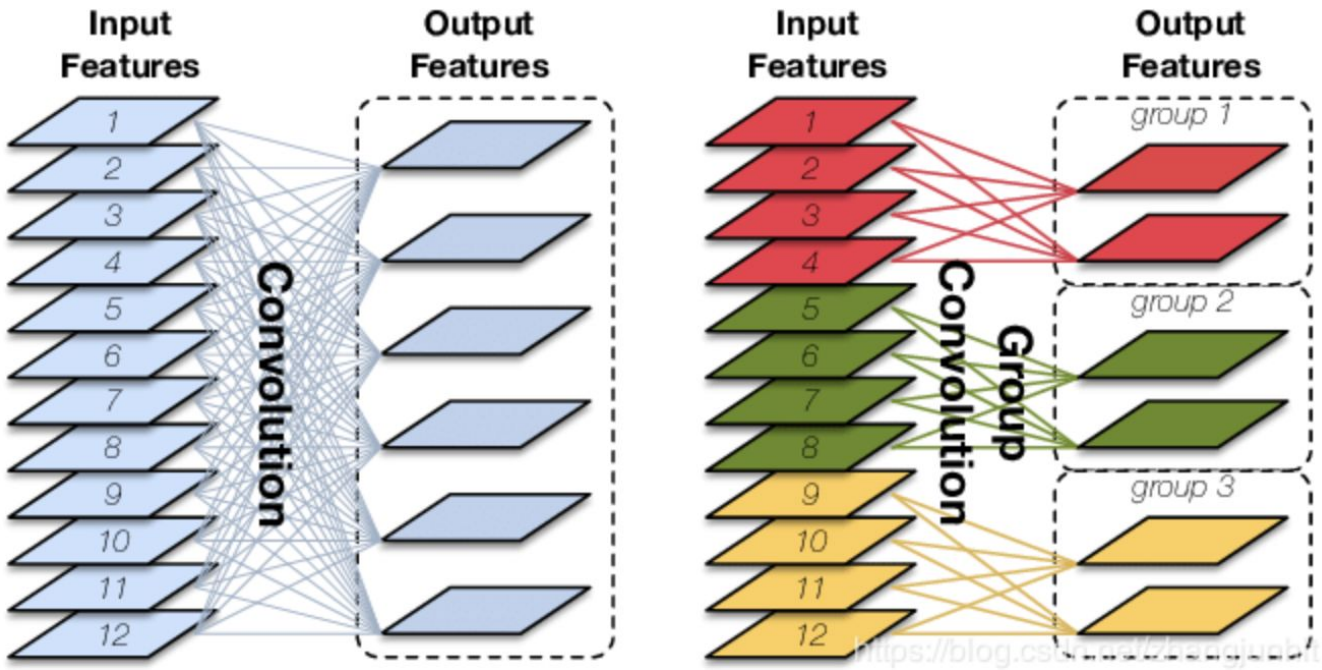
Visualize 96 Convolutional Kernels



- 11 x 11 x 3 size kernels.
- top 48 kernels on GPU 1 : color-agnostic
- bottom 48 kernels on GPU 2 : color-specific.

Why?

Group Convolution



- Grouping filters
- Train and convolve independently
- Better performance

Xijun Wang, Meina Kan, Shiguang Shan, Xilin Chen, "Fully Learnable Group Convolution for Acceleration of Deep Neural Networks", CVPR 2019

References

References

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15.1 (2014): 1929-1958.

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115.3 (2015): 211-252.

Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

University of Zurich[UZH]

Department of Informatics

Questions?

主成份分析: 算法描述

$\mathbf{Y} = n \times l \quad \mathbf{X} = n \times d \quad \mathbf{W} = d \times l$

- 输入： n个d维样本数据所构成的矩阵$\mathbf{X}$，降维后的维数l
- 输出：映射矩阵$\mathbf{W} = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_l}\}$
- 算法步骤：

1：对于每个样本数据$\boldsymbol{x}_i$进行中心化处理： $\boldsymbol{x}_i = \boldsymbol{x}_i - \mu,\ \mu = \frac{1}{n}\sum_{j=1}^{n} x_j$

2: 计算原始样本数据的协方差矩阵：$\boldsymbol{\Sigma} = \frac{1}{n-1}\mathbf{X}^{\mathbf{T}}\boldsymbol{X}$

3: 对协方差矩阵$\boldsymbol{\Sigma}$进行特征值分解，对所得特征根按其值大到小排序$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$

4: 取前l个最大特征根所对应特征向量$\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_l}$组成映射矩阵$\mathbf{W}$

5：将每个样本数据$\boldsymbol{x}_i$按照如下方法降维：$(\boldsymbol{x}_i)_{1\times d}(\mathbf{W})_{d\times l} = 1 \times l$