ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton (University of Toronto, 2012)

Jihyeon Song
2021.04.30

목차
INDEX

1 Introduction

2 The Dataset

3 The Architecture

4 Reducing Overfitting

5 Details of learning

6 Results

7 Discussion

Abstract

- Trained a large, deep convolutional neural network to classify the 1.2M high-resolution images in ImageNet LSVRC-2010
- Non-saturating neurons and GPU implementation
- Employed a regularization method called “dropout”
- We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry

1 Introduction

- NYU Object Recognition Benchmark (NORB)
  - 97,200 image pairs of 50 toys in 5 categories

- Caltech
  - Caltech-101: About 40 to 800 images in 101 categories
  - Caltech-256: 30,607 images in 257 categories

- Canadian Institute for Advanced Research (CIFAR)
  - CIFAR-10: 60,000 images in 10 classes
  - CIFAR-100: 600 images in 100 classes

1 Introduction: Contributions

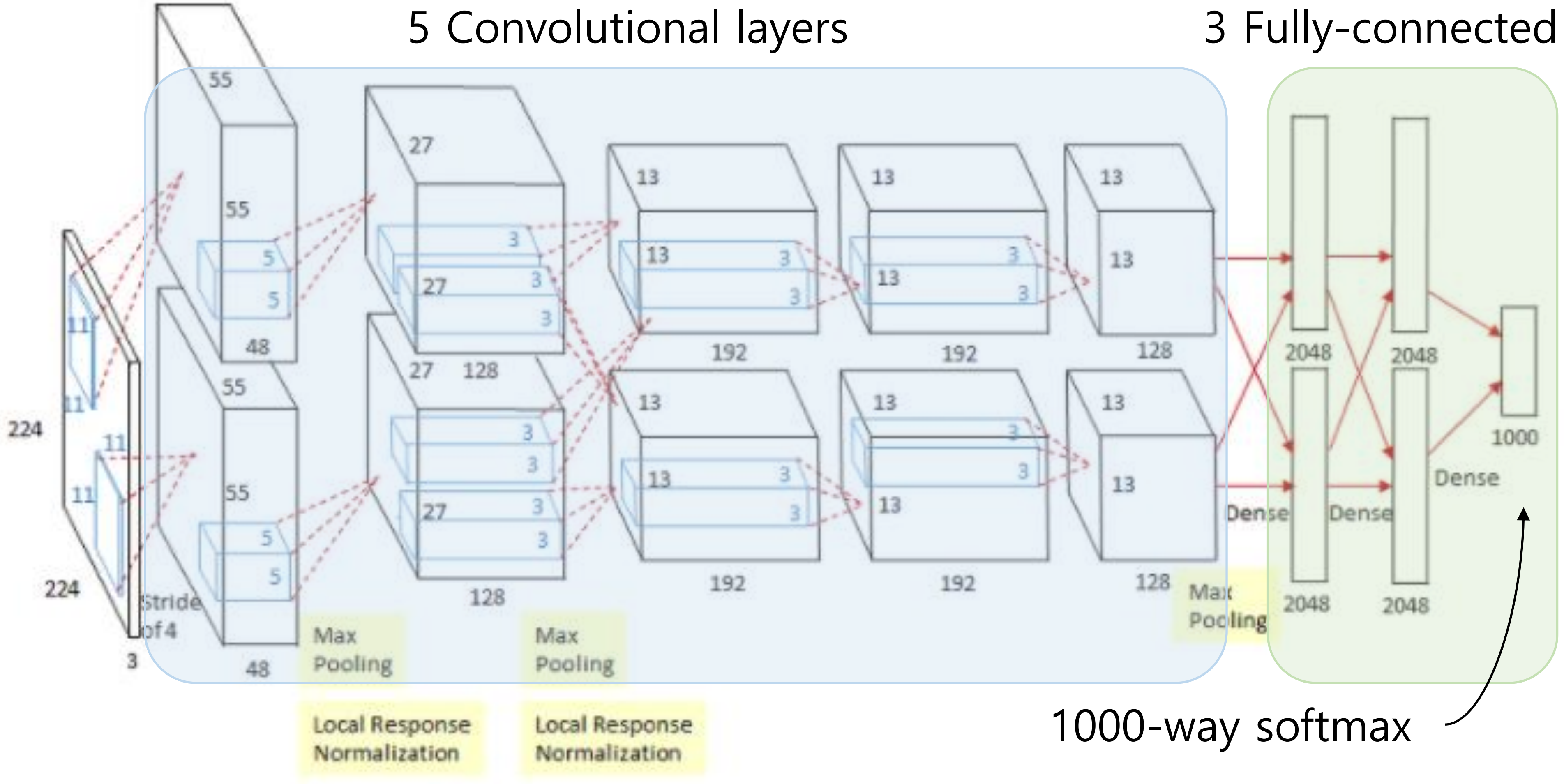
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)-2010/2012
  - Using the subsets of ImageNet

- We trained one of the largest Convolutional Neural Network (CNN)
  - The best results ever reported on these datasets
  - Highly-optimized GPU implementation
  - Improve performance and reduce training time
  - Preventing overfitting
  - Called AlexNet after the first author of the paper

2 The Dataset

- The ImageNet project is a large visual database
  - Over 15M labeled high-resolution images belonging to roughly 22,000 categories
- ILSVRC
  - 1.2M training images, 50K validation images, 150K testing images in 1000 categories
  - ILSVRC-2012: test set labels are unavilable
  - Down-sampled the images to a fixed resolution of 256x256
  - Subtracting the mean activity over the training set from each pixel
- Top-1 and Top-5 error rates
  - The metric used to rank the methods in the classification challenge

3 The Architecture



5 Convolutional layers

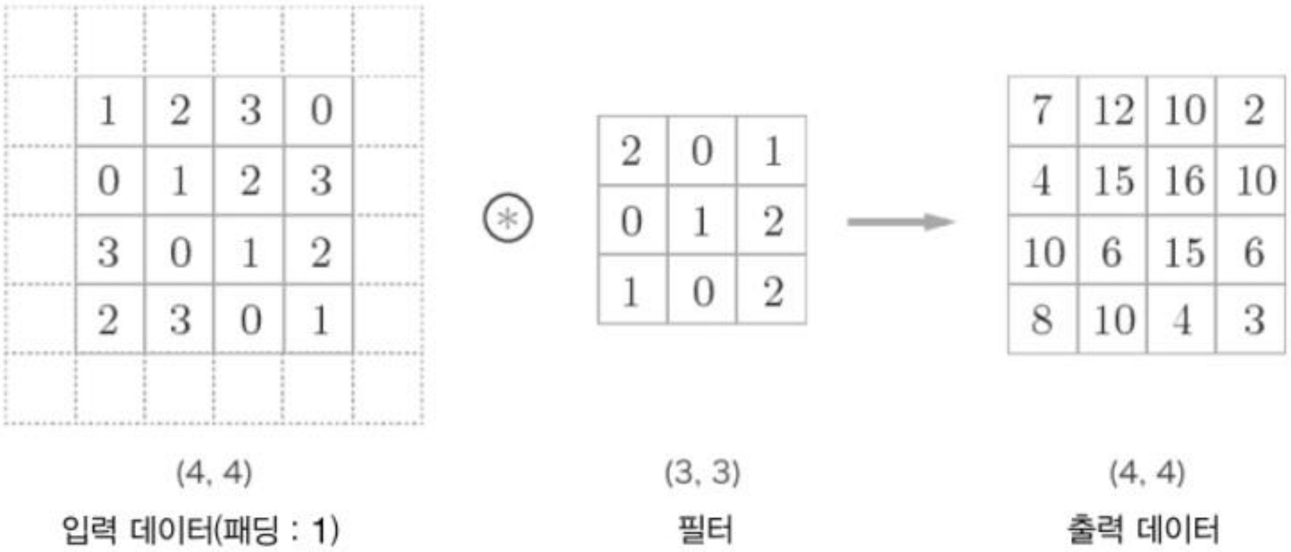
3 Fully-connected layers

1000-way softmax

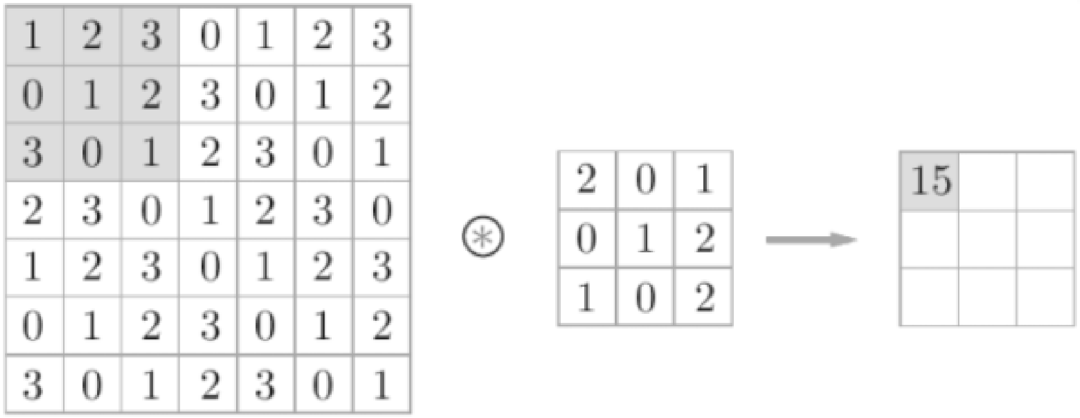
AlexNet Architecture

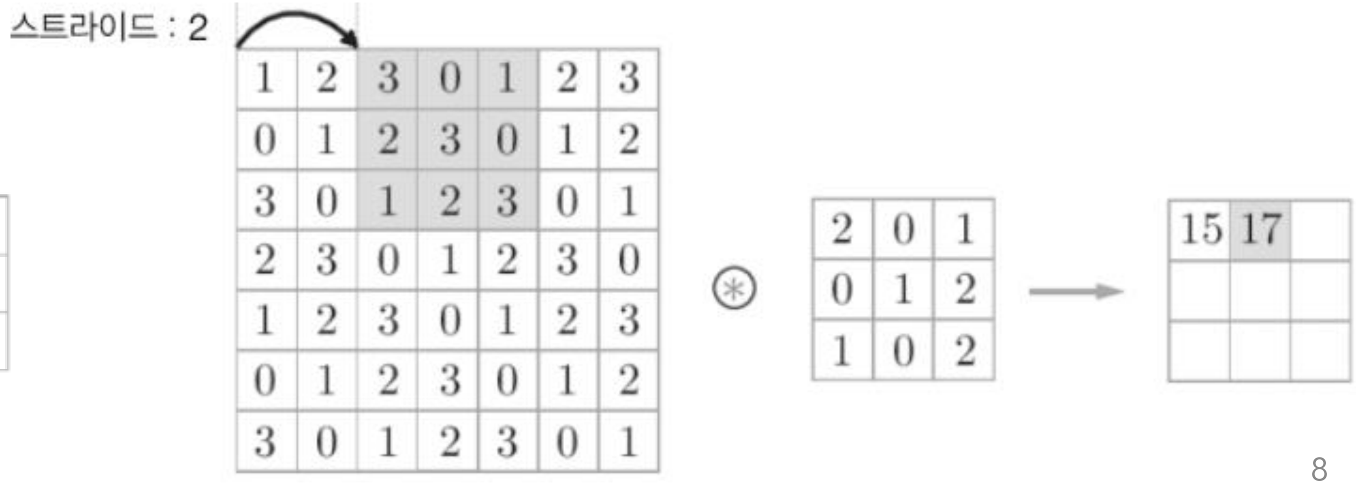
3 The Architecture

- Filter = kernel



- (H, W): Input size, (FH, FW): filter,
  (OH, OW): feature map output size, P: padding, S: stride
  - (OH, OW) = (((H+2P-FH) / S)+1, ((W+2P-FW) / S)+1)





3 The Architecture: ReLU

- Rectified Linear Units (ReLUs)

Sigmoid

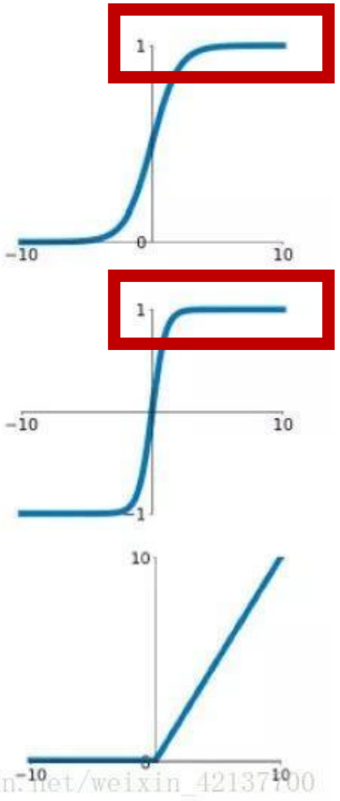
$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh

$\tanh(x)$

ReLU

$\max(0, x)$



Saturating

Gradient vanishing

Non-saturating

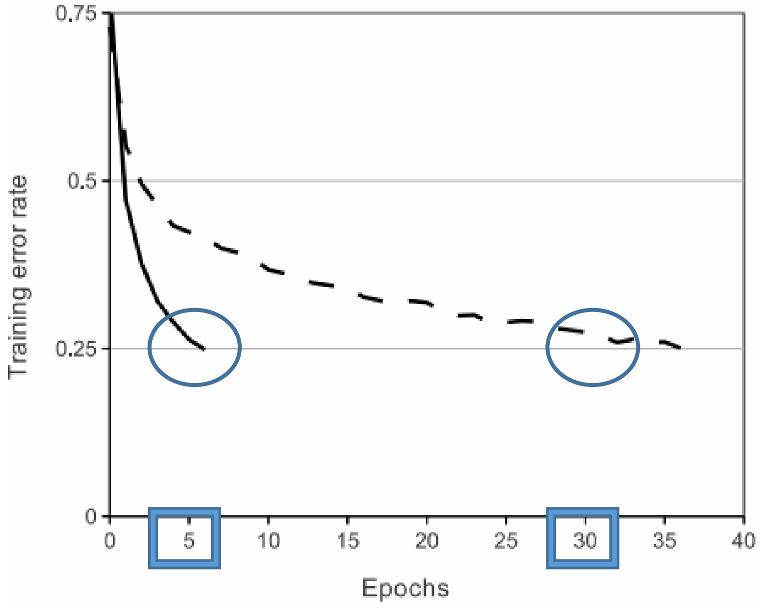


Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**. The learning rates for each network were chosen independently to make training as fast as possible. No regularization of any kind was employed. The magnitude of the effect demonstrated here varies with network architecture, but networks with ReLUs consistently learn several times faster than equivalents with saturating neurons.

3 The Architecture: Multiple GPUs

Model Overview

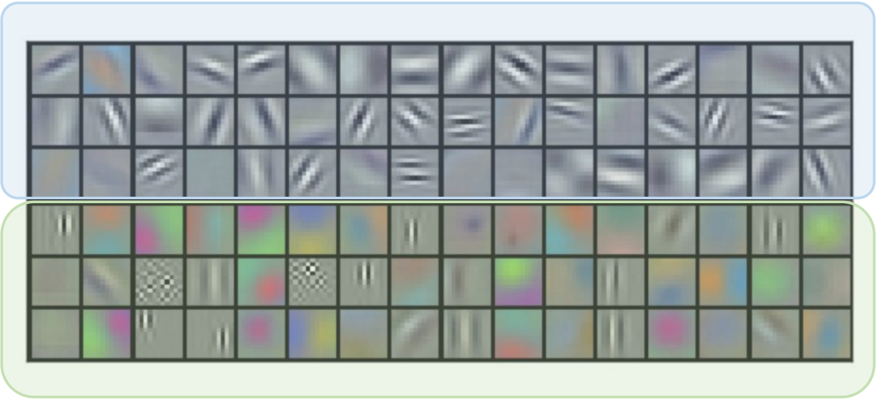
X 5 layers

X 3 layers

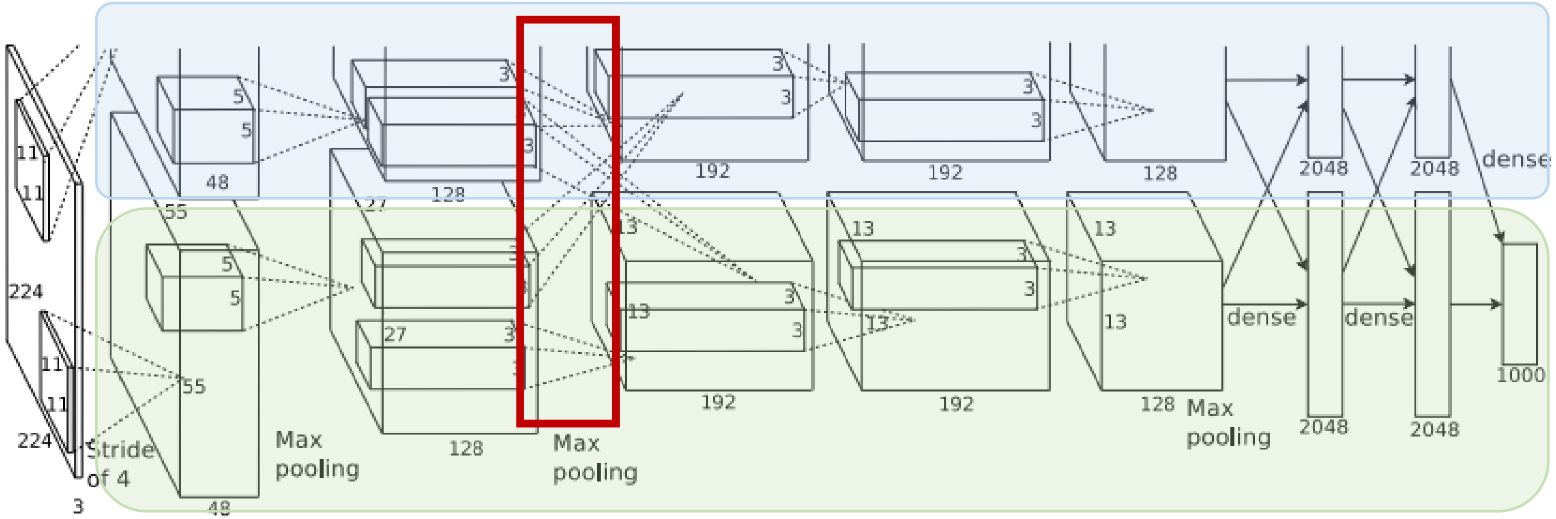
X 60M

X 650,000

- GTX 580 GPUs with 3GB memory X 2
  - 1.2M training images are too big to fit on one GPU
  - Puts half of the kernels on each GPU, the GPUs communicate only in certain layers
  - This scheme reduces our top-1 and top-5 error rates by 1.7% and 1.2%

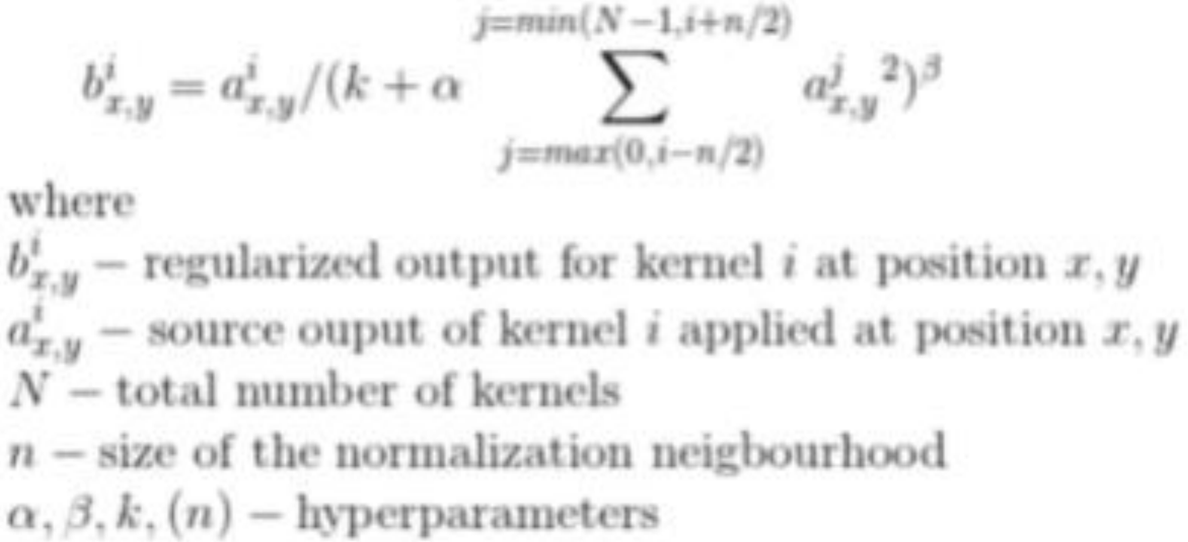


GPU-1, GPU-2



3 The Architecture: Local Response Normalization (LRN)

- If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron.
  - Hyper-parameters: k = 2, n = 5, $\alpha = 10^{-4}$, $\beta = 0.75$
  - LRN layer implements the lateral inhibition
  - Response normalization reduces our top-1 and top-5 error rates by 1.4% and 1.2%



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

where

$b^i_{x,y}$ – regularized output for kernel i at position x, y

$a^i_{x,y}$ – source ouput of kernel i applied at position x, y

N – total number of kernels

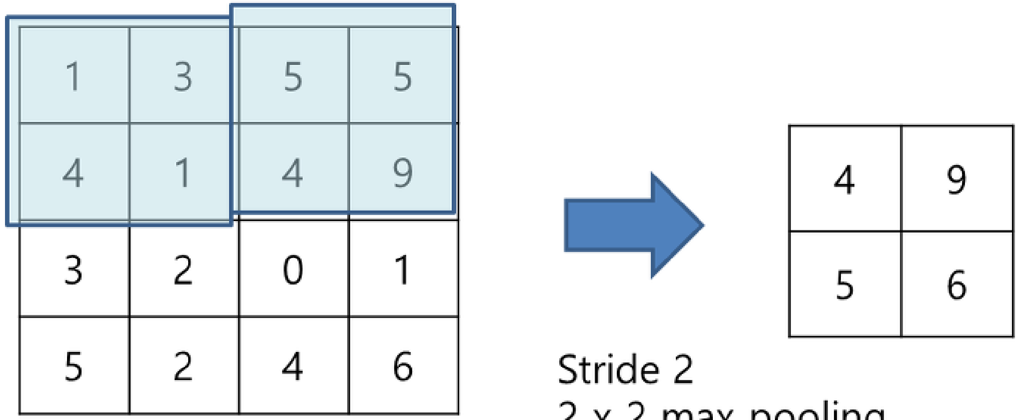
n – size of the normalization neigbourhood

$\alpha, \beta, k, (n)$ – hyperparameters

3 The Architecture: Overlapping Pooling

- Stride < Kernel size
  - Stride = 2, kernel size = 3
  - This scheme reduces the top-1 and top-5 error rates by 0.4% and 0.3%
- We observe during training that models with overlapping pooling find it difficult to overfit

Non-overlapping pooling

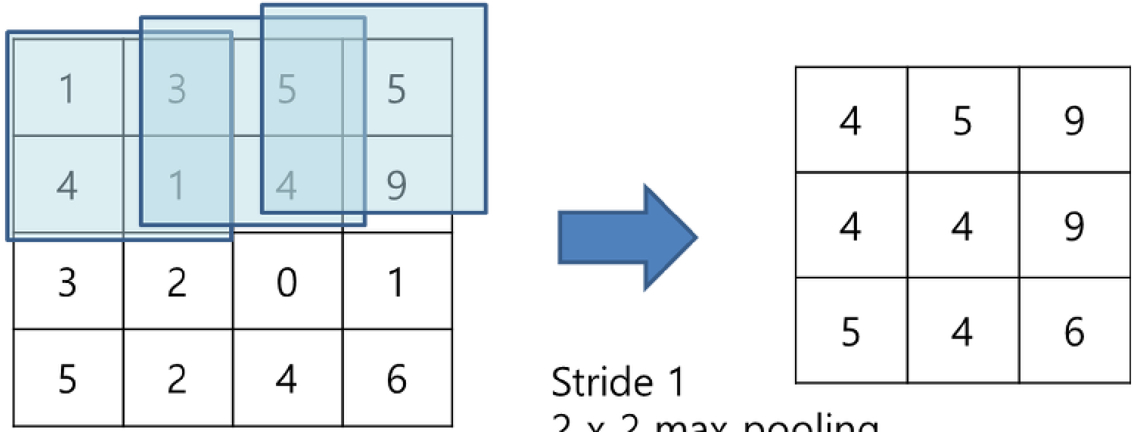


1	3	5	5
4	1	4	9
3	2	0	1
5	2	4	6

4	9
5	6

Stride 2
2 x 2 max pooling

Overlapping pooling



1	3	5	5
4	1	4	9
3	2	0	1
5	2	4	6

4	5	9
4	4	9
5	4	6

Stride 1
2 x 2 max pooling

3 The Architecture: Overall Architecture

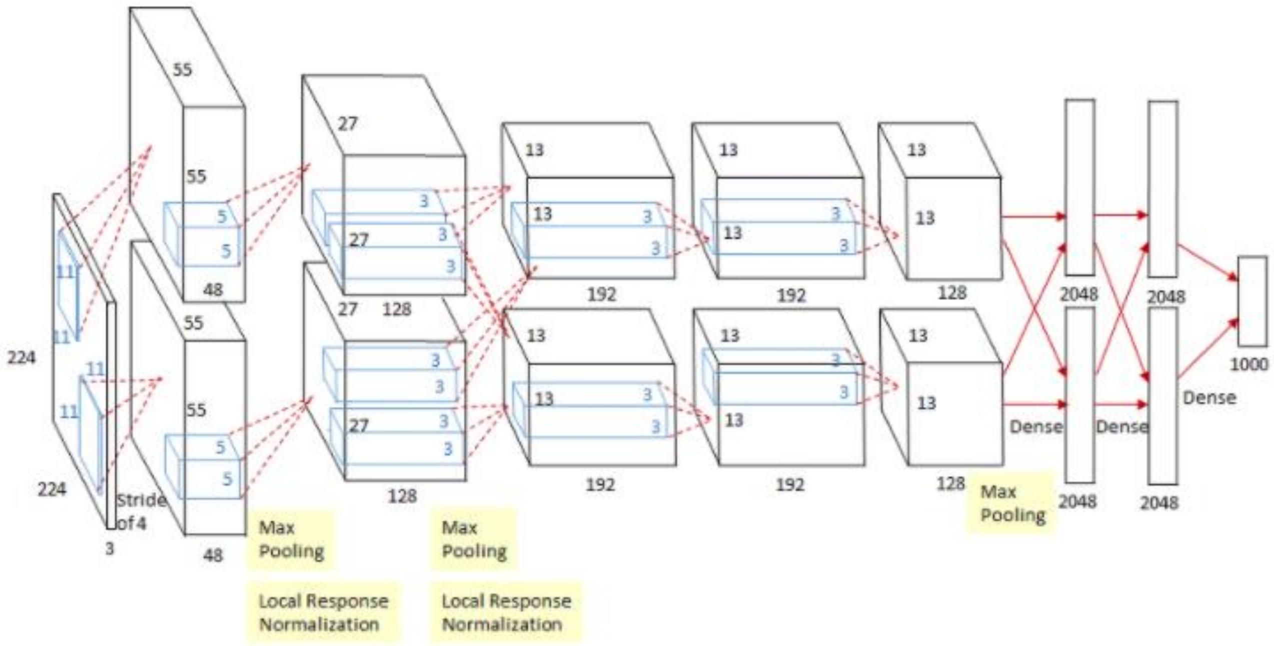
Input: 227×227×3 input images (224×224×3 sizes is mentioned in the paper and also in the figure, however, it is later pointed out that it should be 227, or 224×224×3 is padded during the 1st convolution.)

1st: Convolutional Layer: 2 groups of 48 kernels, size 11×11×3 (stride: 4, pad: 0)
Outputs 55×55 ×48 feature maps ×2 groups
Then **3×3 Overlapping Max Pooling (stride: 2)**
Outputs 27×27 ×48 feature maps ×2 groups
Then **Local Response Normalization**
Outputs 27×27 ×48 feature maps ×2 groups

2nd: Convolutional Layer: 2 groups of 128 kernels of size 5×5×48 (stride: 1, pad: 2)
Outputs 27×27 ×128 feature maps ×2 groups
Then **3×3 Overlapping Max Pooling (stride: 2)**
Outputs 13×13 ×128 feature maps ×2 groups
Then **Local Response Normalization**
Outputs 13×13 ×128 feature maps ×2 groups

(OH, OW) = (((H+2P-FH) / S)+1, ((W+2P-FW) / S)+1)

1st: (((227+2*0-11) / 4)+1, ((227+2*0-11) / 4)+1)
= (((227-11)/4)+1, ((227-11)/4)+1) = (55,55)



AlexNet Architecture

3 The Architecture: Overall Architecture

3rd: Convolutional Layer: 2 groups of 192 kernels of size 3×3×256 (stride: 1, pad: 1)
Outputs 13×13 ×192 feature maps ×2 groups

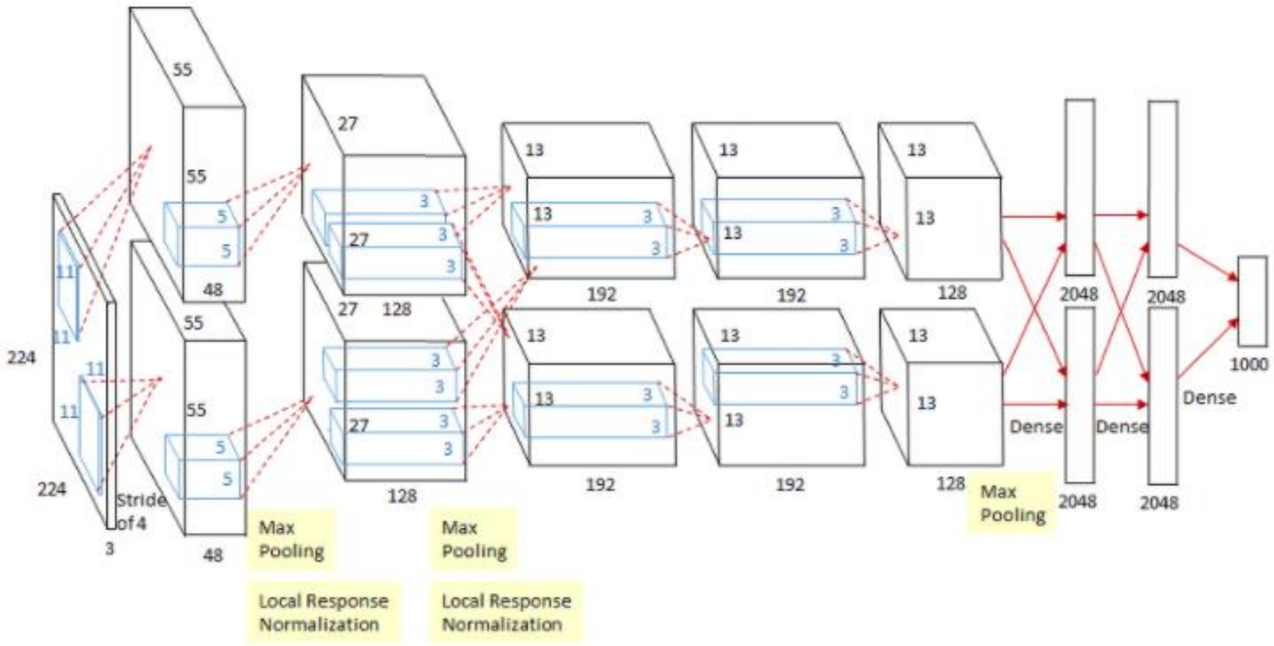
4th: Convolutional Layer: 2 groups of 192 kernels of size 3×3×192 (stride: 1, pad: 1)
Outputs 13×13 ×192 feature maps ×2 groups

5th: Convolutional Layer: 256 kernels of size 3×3×192 (stride: 1, pad: 1)
Outputs 13×13 ×128 feature maps ×2 groups
Then **3×3 Overlapping Max Pooling (stride: 2)**
Outputs 6×6 ×128 feature maps ×2 groups

6th: Fully Connected (Dense) Layer of
4096 neurons

7th: Fully Connected (Dense) Layer of
4096 neurons

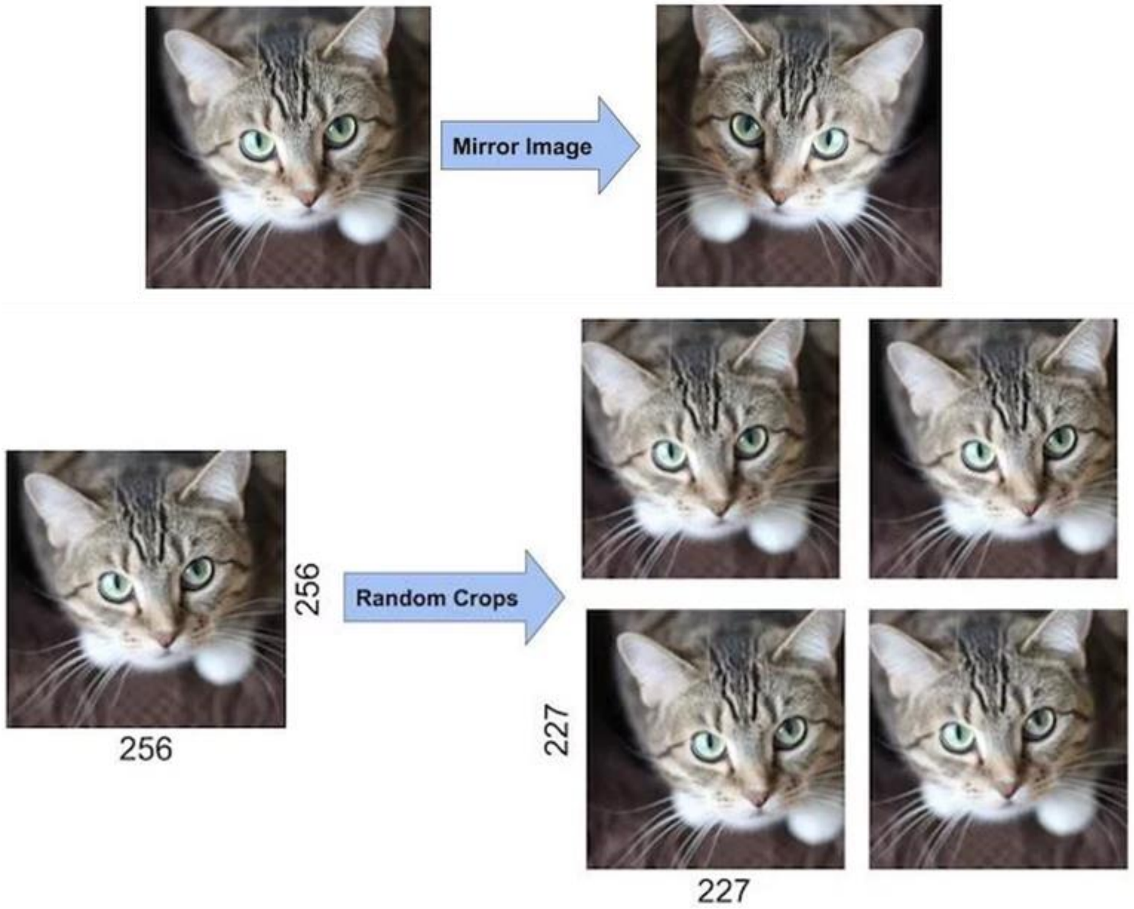
8th: Fully Connected (Dense) Layer of
Outputs 1000 neurons (since there are 1000 classes)
Softmax is used for calculating the loss.



AlexNet Architecture

4 Reducing Overfitting: Data augmentation

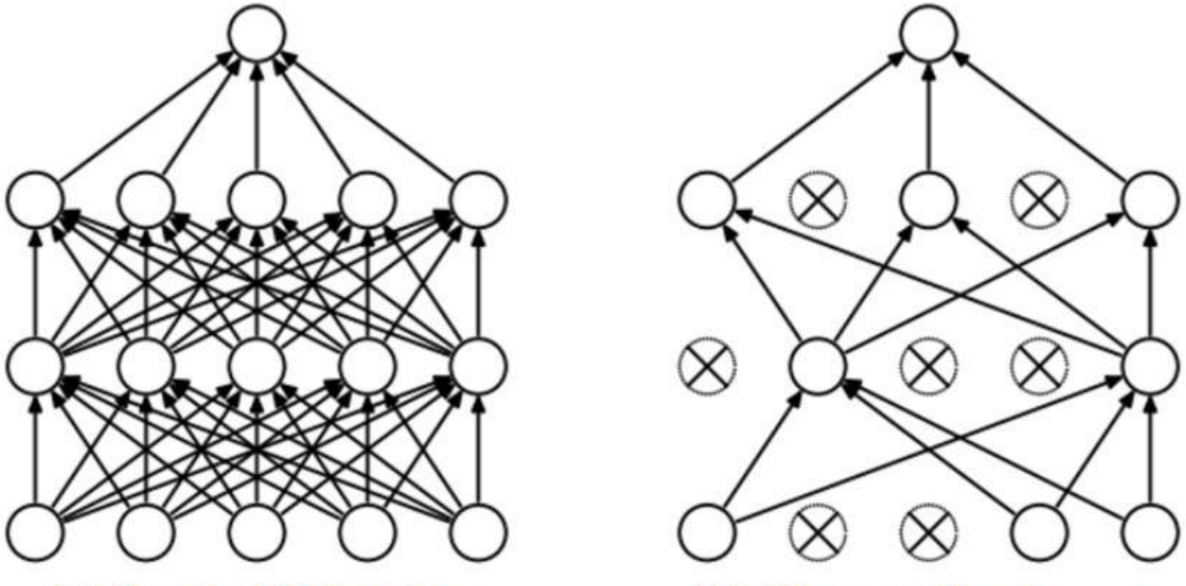
1. Generating image translations and horizontal reflections
2. Altering the intensities of the RGB channels in training images



- This scheme reduces the top-1 error rate by over 1%

4 Reducing Overfitting: Dropout

- Setting to zero the output of each hidden neuron with probability 0.5
- We use dropout in the first two fully-connected layers
- Reduces complex co-adaptations of neurons



(a) Standard Neural Net (b) After applying dropout.

5 Details of learning

- Trained models using stochastic gradient descent (SGD) with
  - Batch size: 128
  - Momentum: 0.9
  - Weight Decay: 0.0005

i: iteration index
v: momentum variable
ϵ: learning rate

- The update rule for weight w

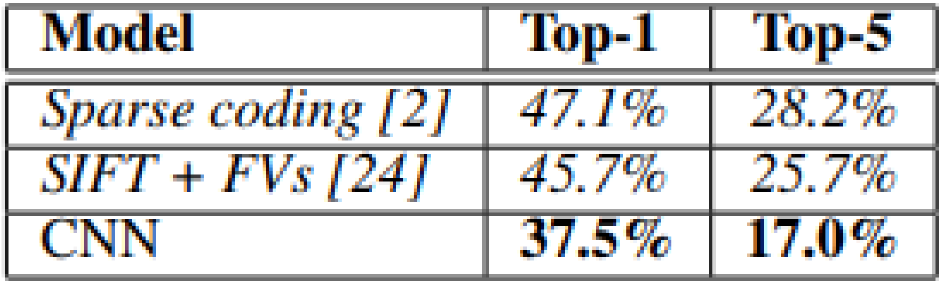
$$
\begin{aligned}
v_{i+1} &:= 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i} \\
w_{i+1} &:= w_i + v_{i+1}
\end{aligned}
$$

  - Initialized the weights in each layer from a zero-mean Gaussian distribution with standard deviation 0.01
  - Bias 1: Conv 2, 4, 5, FC layer / Bias 0: remaining layers
  - Learning rate: all layers 0.01 and reduced three times prior to termination

- Network is trained for roughly 90 cycles, 5-6 days on two GPUs

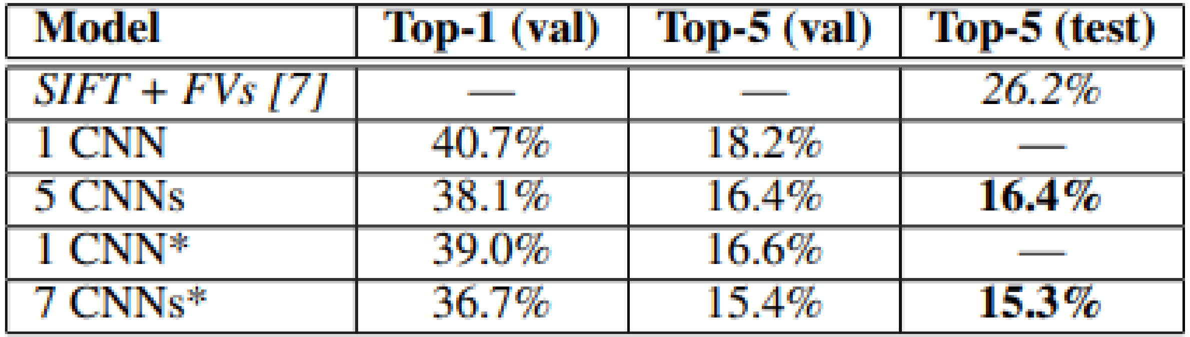
6 Results

- We also report our error rates on the Fall 2009 version of ImageNet with 10,184 categories and 8.9 million images
- Our top-1 and top-5 error rates on this dataset are 67.4% and 40.9%
    - The best published results on this dataset are 78.1% and 60.9%



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.



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.

6 Results: Qualitative Evaluations

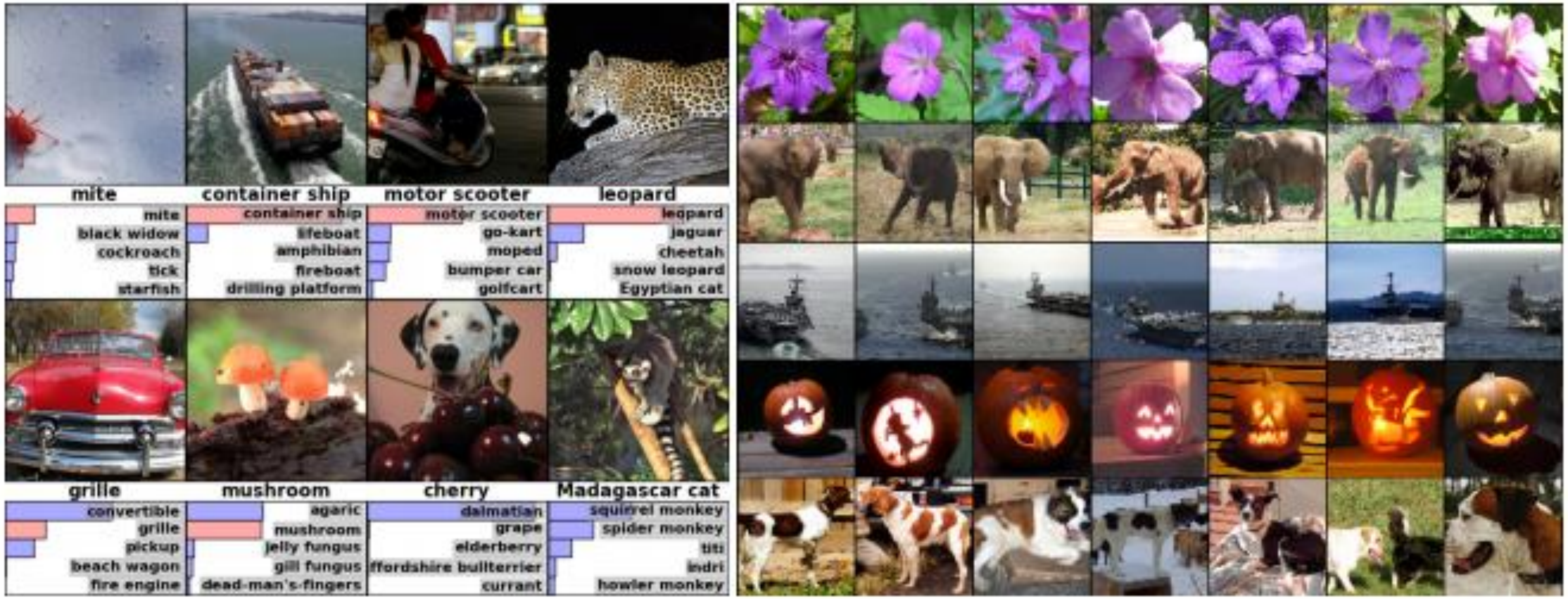
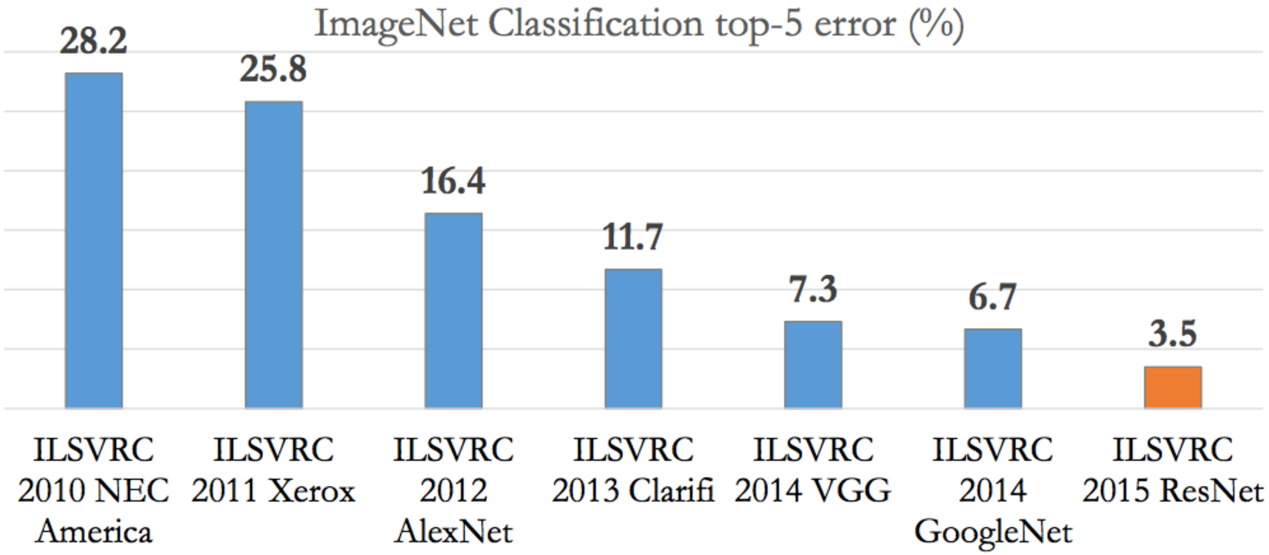


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

7 Discussion

- Large and deep CNNs can recognize large datasets using purely supervised learning
- It is notable that our network's performance degrades if a single convolutional layer is removed
- AlexNet, GoogleNet, VGGNet, ResNet



Reference

- https://medium.com/coinmonks/paper-review-of-alexnet-caffenet-winner-in-ilsvrc-2012-image-classification-b93598314160
- https://bskyvision.com/421
- https://hydragon-cv.info/entry/VERY-DEEP-CONVOLUTIONAL-NETWORKS-FOR-LARGE-SCALE-IMAGE-RECOGNITION
- https://soobarkbar.tistory.com/4
- https://ratsgo.github.io/deep%20learning/2017/10/09/CNNs/
- https://www.datamaker.io/posts/34/
- https://m.blog.naver.com/laonple/220654387455

Thank you