ImageNet Classification with Deep Convolutional Neural Networks

Authors: Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton
University of Toronto

Presenter: Yuanzhe Li
10/4/2016

Outline

- Introduction
- Network Architecture
  - ReLU Nonlinearity
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing Overfitting
  - Data Augmentation
  - Dropout
- Learning
- Results and Discussion

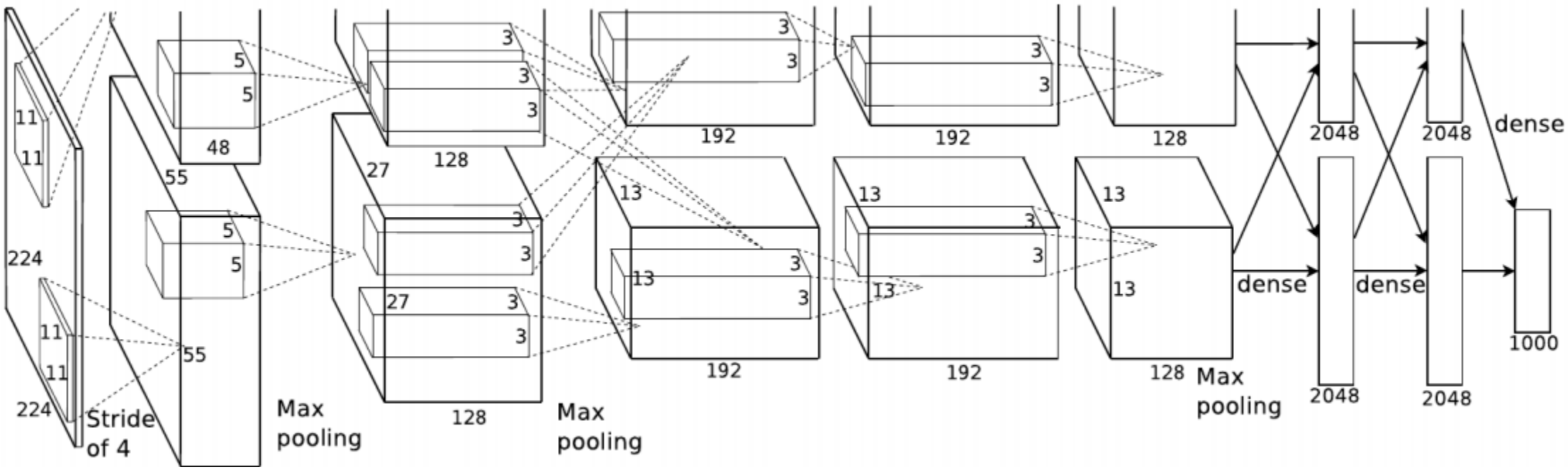
Introduction

- ImageNet
  - Over 15 million labeled high-resolution images
  - Over 22,000 categories
- ILSVRC
  - (In 2010 contest) 1.2 million images depicting 1000 object categories
  - 50,000 validation images, 150,000 testing images
  - Customary to report top-1 and top-5 error rates

Fei-Fei Li and Olga Russakovsky

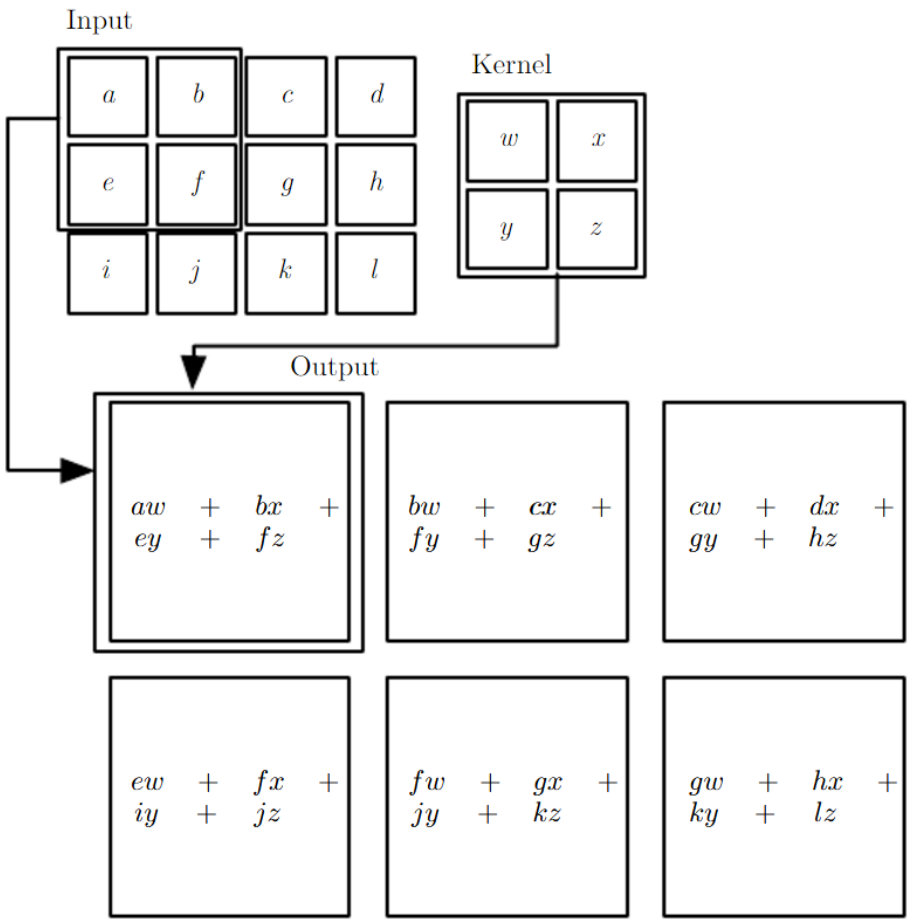
Network Architecture

- Overall Architecture



Network Architecture

- Convolution operation

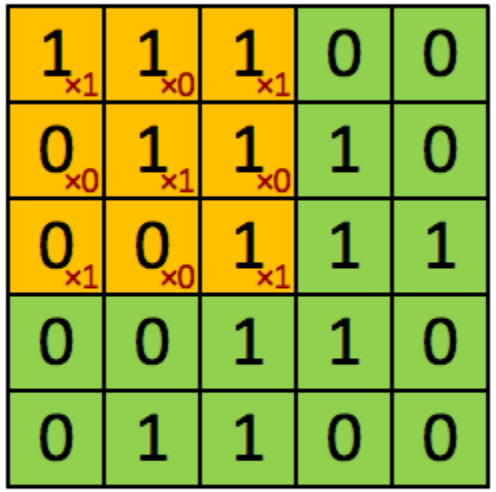


Goodfellow et al., 2016

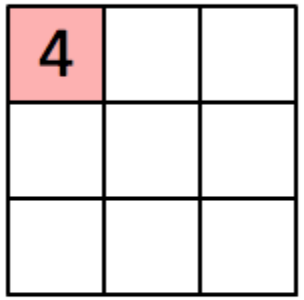
$$x_{ij}^{\ell} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \omega_{ab} y_{(i+a)(j+b)}^{\ell-1}$$

Assume that we use *m* by *m* kernels in the *l*-th layer

Andrew Gibiansky



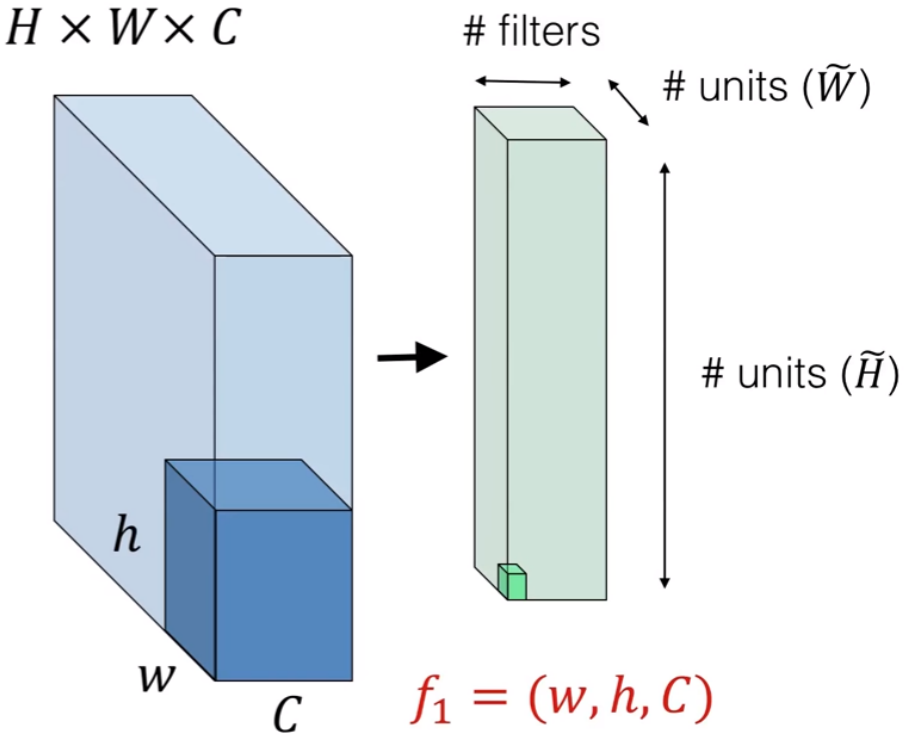
Image

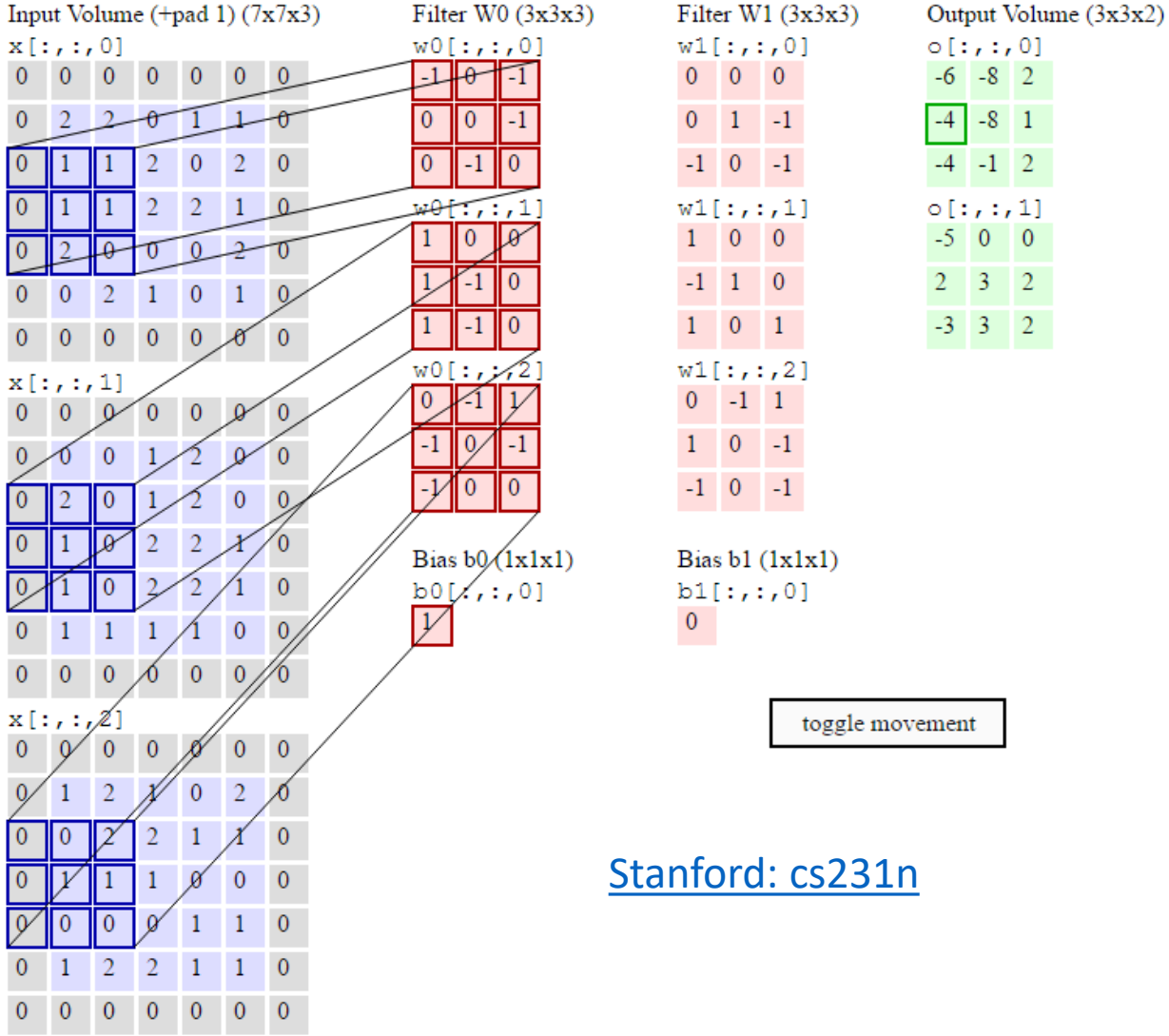


Convolved Feature

Network Architecture

- 2-D Convolution



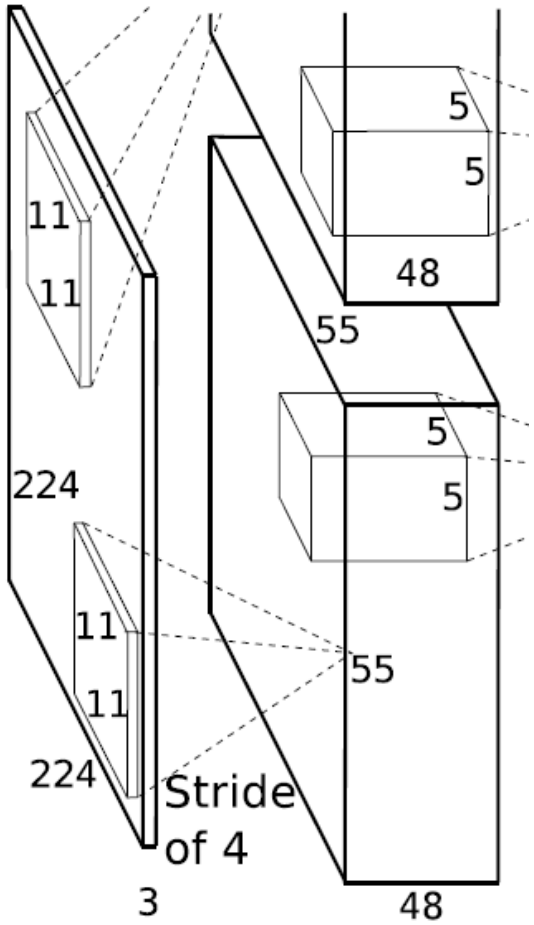


Stanford: cs231n

Network Architecture

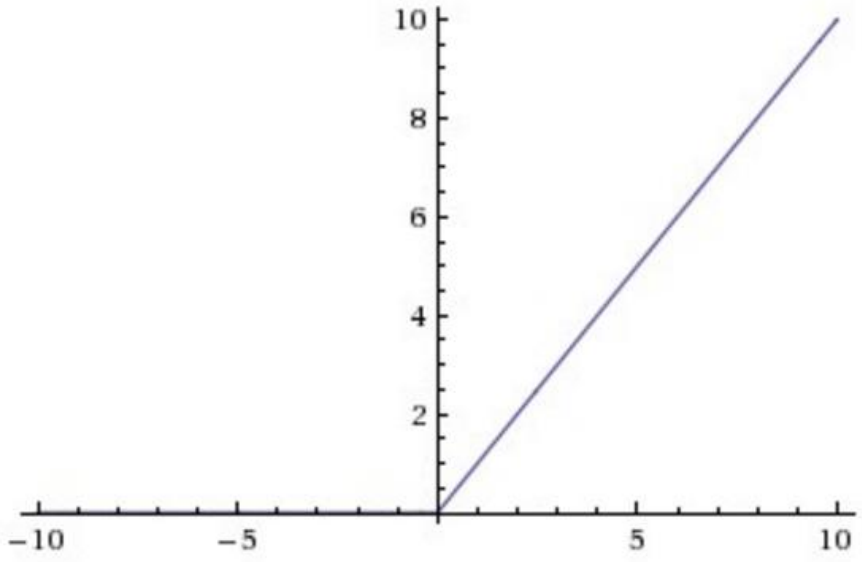
- Convolutional Layer (1st ConvLayer)
  - Images: 224 x 224 x 3
  - Kernels: 11 x 11
  - Stride = 4 x 4
  - Depth: 48
  - Output: 55 x 55 x 48

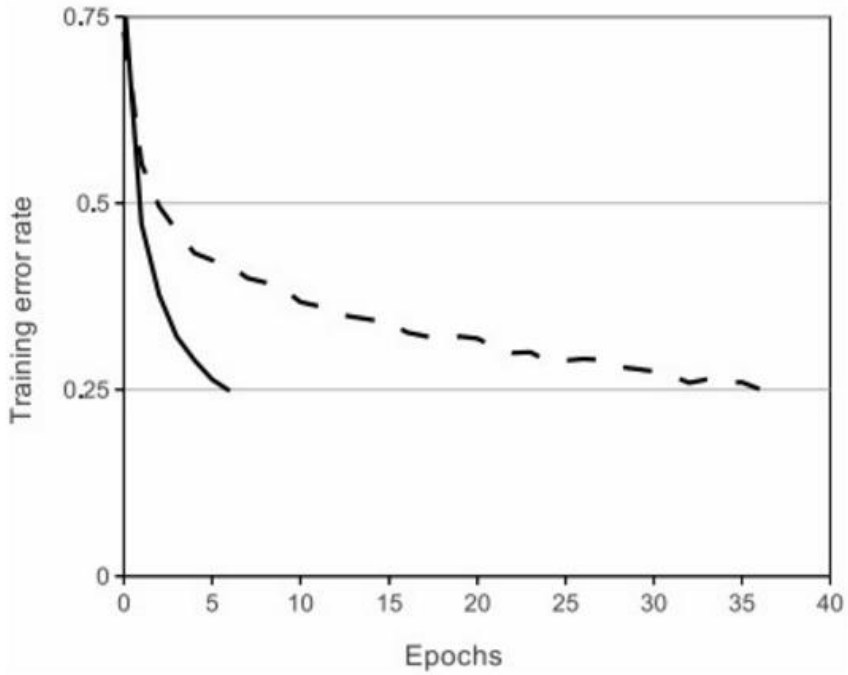
For more about filter size, depth, stride, zero-padding, and the convolutional operation, check [Stanford - cs231n]



Network Architecture

- ReLU (Rectified Linear Units) Nonlinearity
  - f(x) = max(0,x)





Stanford: cs231n

Network Architecture

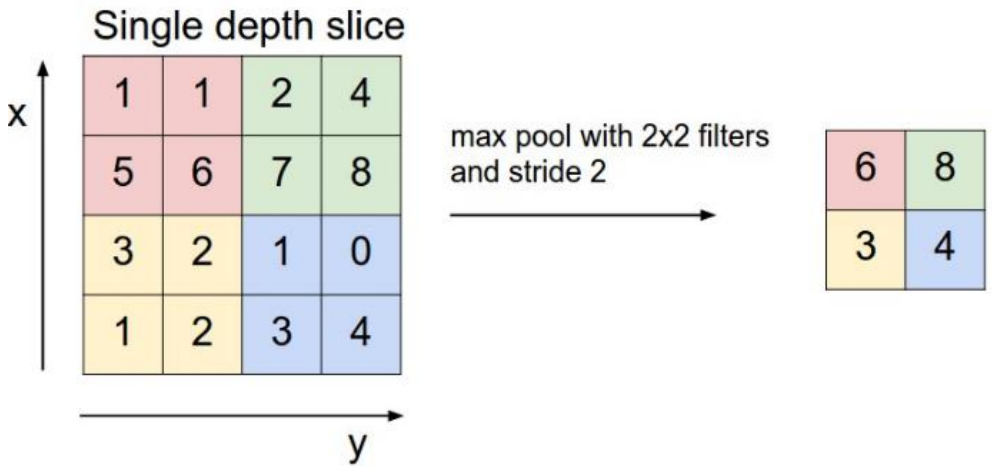
- Local Response Normalization (LRN)

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

  - b: Response-normalized activity
  - a: Activity computed by applying kernel i at position (x, y) and the ReLU
  - Hyper-parameters: $\mathrm{k} = 2, \mathrm{n} = 5, \alpha = 10^{-4}, \beta = 0.75$
  - Reduces the top-1 and top-5 error rate by 1.4% and 1.2%, respectively

Network Architecture

- Local Pooling
  - Max Pooling Layer
  - Grid of pooling units spaced s pixels apart, summarizing neighborhoods of size $z \times z$
  - $s = z$ means no overlapping

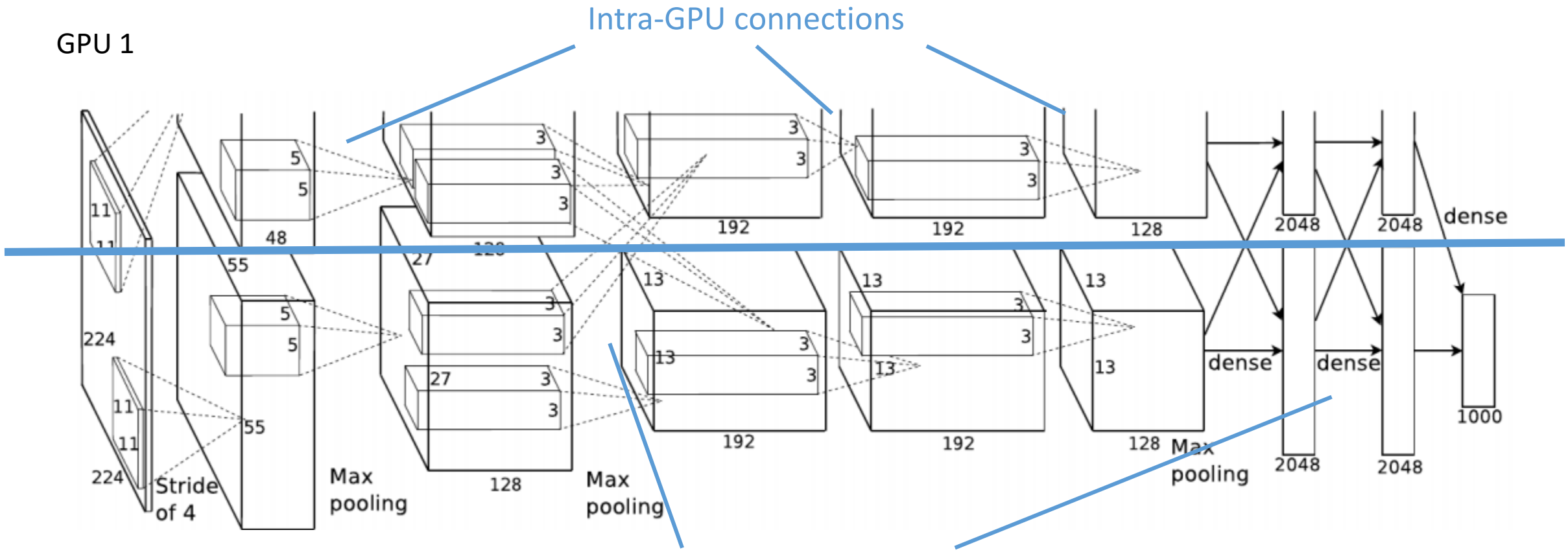


Traditional local max pooling (cs231n)

- $s < z$ Overlapping Pooling, same unit can be pooled more than once
- $s = 2, z = 3$
- Reduction on top-1 and top-5 error rates by 0.4% and 0.3% respectively, compared with traditional max pooling

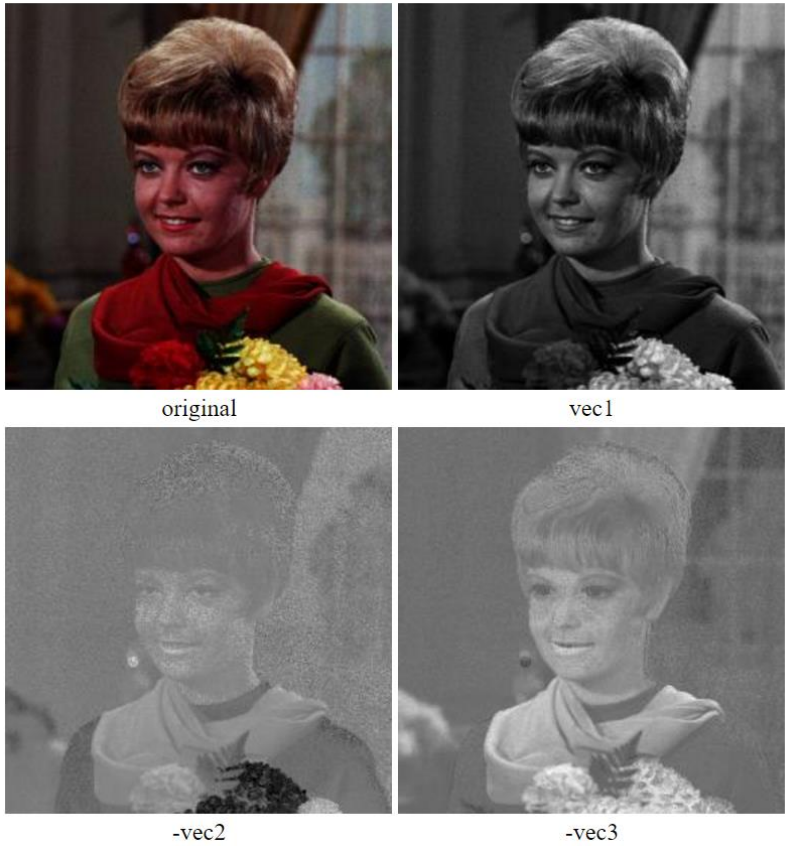
Network Architecture

- Training on Multiple GPUs



Reducing Overfitting

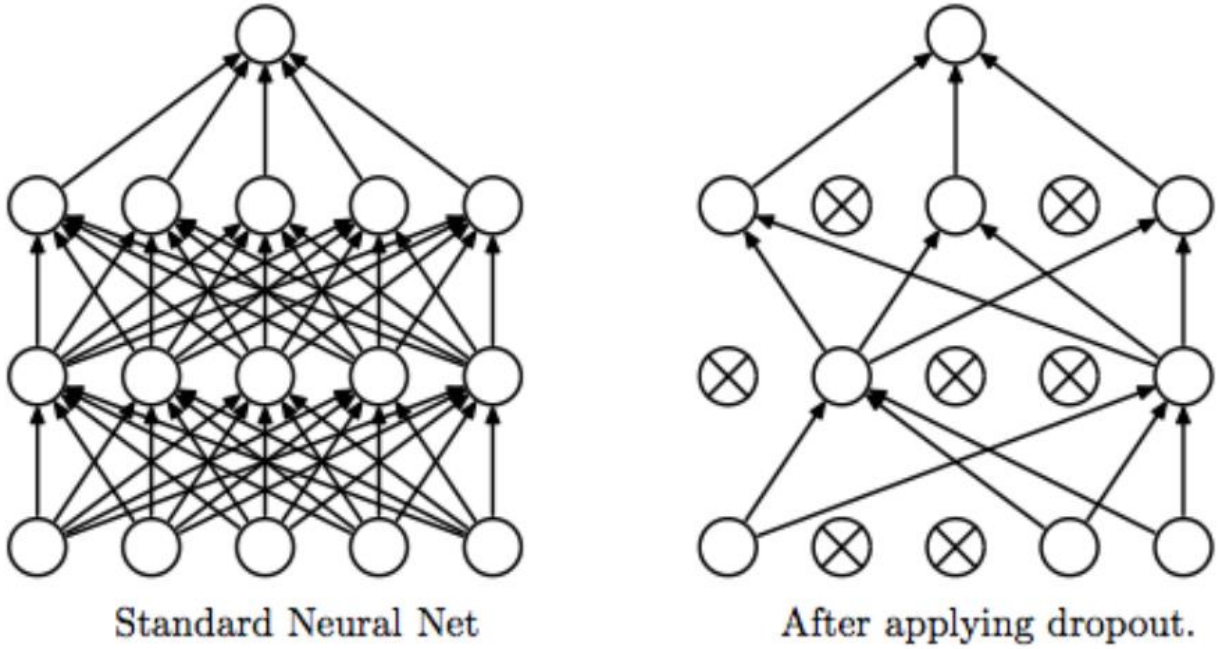
- Data Augmentation
  - 60 million parameters and 650,000 neurons $\Rightarrow$ Overfitting
  - Image translation
    - Extracting 224×224 patches from 256×256 images, and horizontal reflections
    - Use the average of 10 patches to predict for testing set
  - Altering the intensities of the RGB channels
    - Image pixels $I_{xy} = \left[I_{xy}^R, I_{xy}^G, I_{xy}^B\right]^T$
    - Add $[\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$
    - Reduces top-1 error rate by over 1%



John Loomis

Reducing Overfitting

- Dropout
  - During training, set to zero the output of randomly selected hidden neurons with probability 0.5
  - At test time, use all the neurons but multiply their outputs by 0.5



Tasci and Kim

Details of Learning

- Stochastic Gradient Descent

momentum(damping parameter)

Learning rate (initialized at 0.01)

divide by 10 when validation error rate stopped improving with the current rate

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

$$w_{i+1} := w_i + v_{i+1}$$

weight decay

Gradient of Loss
w.r.t weight
Averaged over batch

Tasci and Kim

- Batch size: 128
- 90 epochs over 1.2 million images (5-6 days on two NVIDIA GTX 580 3GB GPUs)

Results

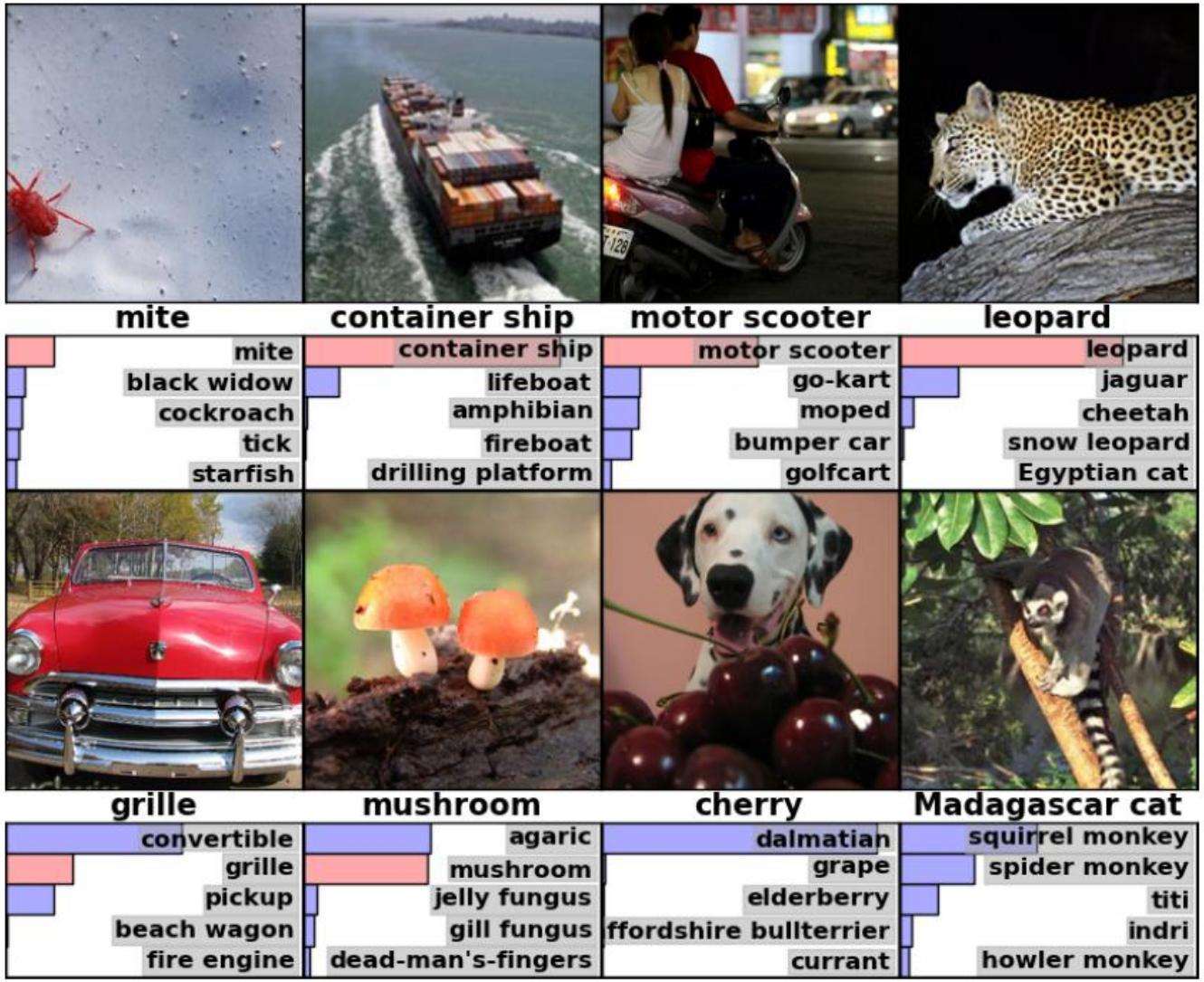
- Results on ILSVRC-2010

Model	Top-1	Top-5
Sparse coding	47.1%	28.2%
SIFT + FVs	45.7%	25.7%
CNN	37.5%	17.0%

- Results on ILSVRC-2012

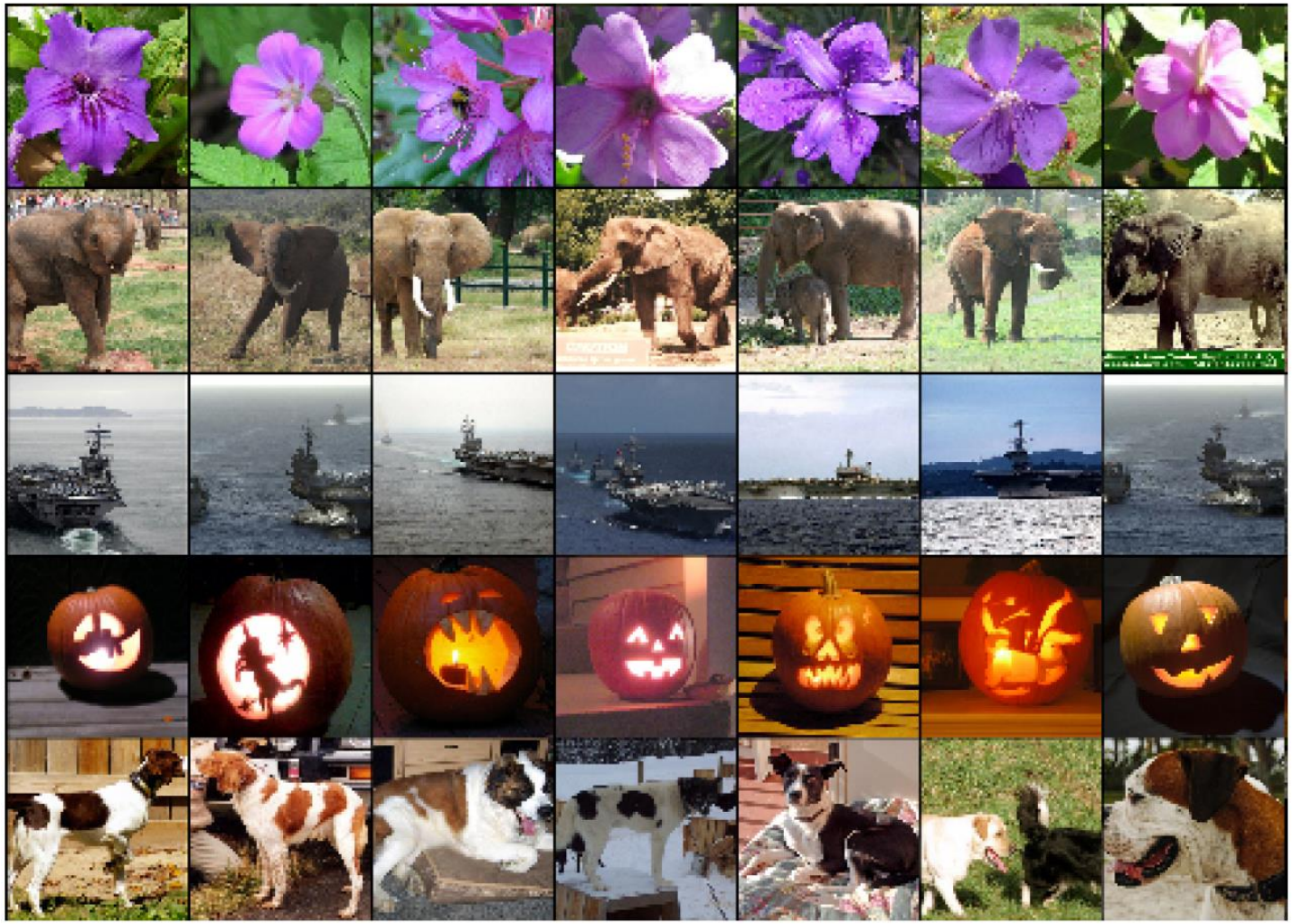
Model	Top-5
SIFT + FVs	26.2%
1 CNN	16.4%
5 CNNs	15.3%

Results


mite
mite
black widow
cockroach
tick
starfish
container ship
container ship
lifeboat
amphibian
fireboat
drilling platform
motor scooter
motor scooter
go-kart
moped
bumper car
golfcart
leopard
leopard
jaguar
cheetah
snow leopard
Egyptian cat
grille
convertible
grille
pickup
beach wagon
fire engine
mushroom
agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers
cherry
dalmatian
grape
elderberry
ffordshire bullterrier
currant
Madagascar cat
squirrel monkey
spider monkey
titi
indri
howler monkey

Results

- 4096 dimensional features from the last fully-connected layers



Discussion

- To go deep is really important for achieving good results
  - Removing one convolutional layer degrades the result by about 2% for top-1 performance
- Ultimately to use very large and deep convolutional nets
- To explore temporal structure from video sequences, which is missing in static images

Questions?