Lab. – Classification

Convolutional Neural Networks (CNNs)

Yuan-Fu Liao

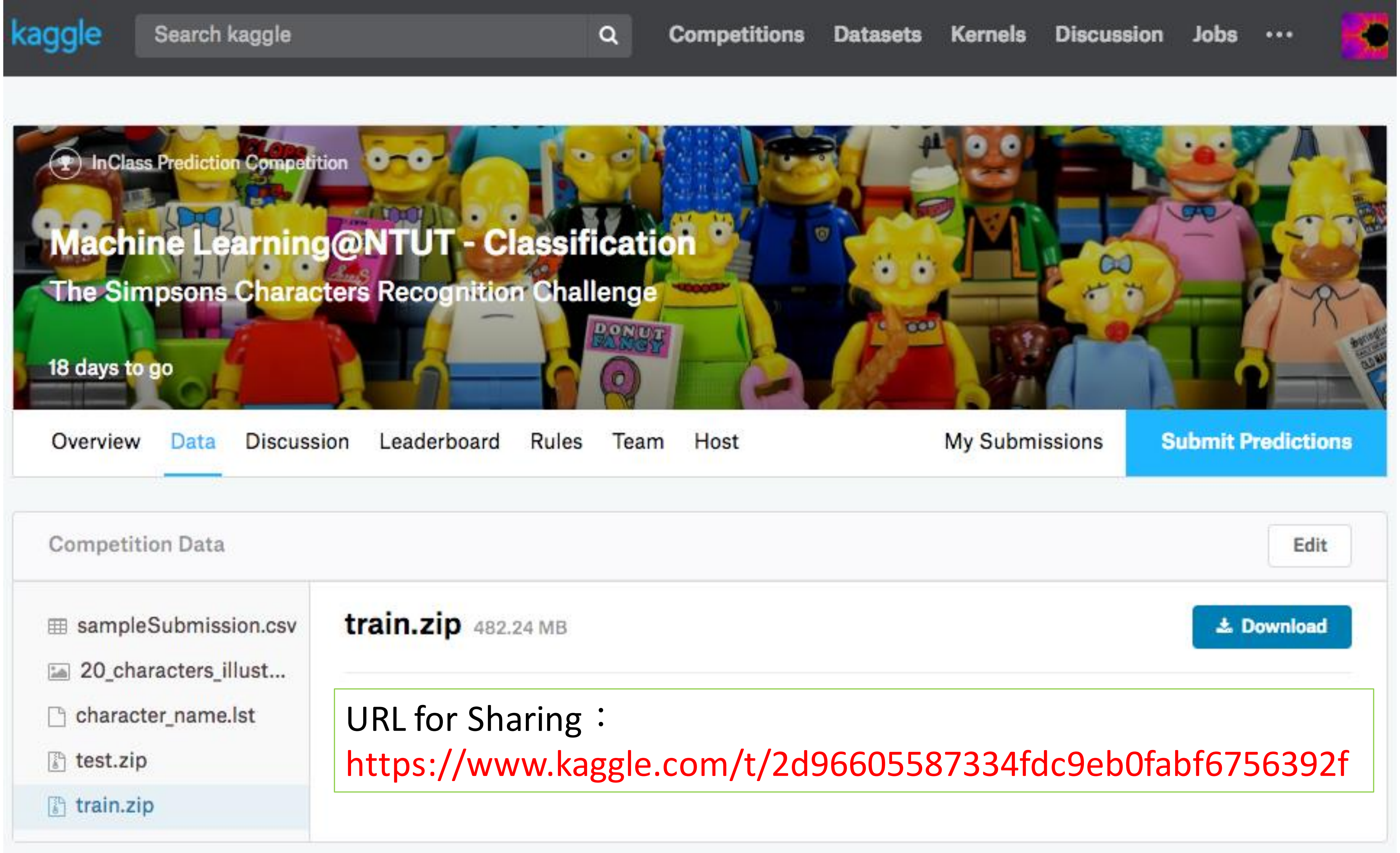
National Yang Ming Chiao Tung University

yfliao@nycu.edu.tw

Kaggle Competition - The Simpsons Characters Recognition Challenge

https://www.kaggle.com/t/2d96605587334fdc9eb0fabf6756392f

- Images of 20 characters extracted from The Simpsons episodes
  - ✓ About 1000 images per character
  - ✓ Pictures are under various size, scenes
  - ✓ not necessarily centered in each image and could sometimes be with or cropped from other characters



Machine Learning@NTUT - Classification

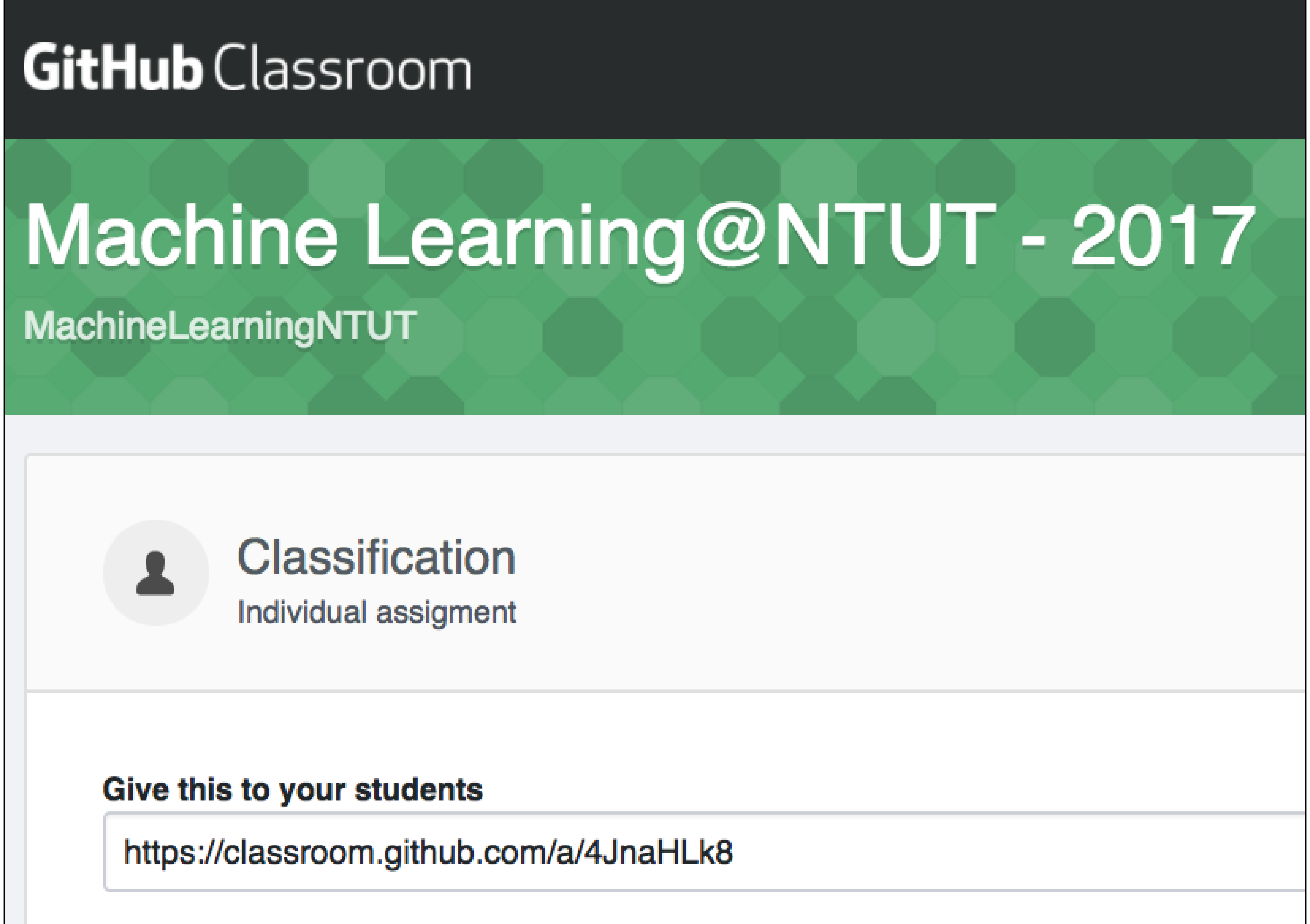
The Simpsons Characters Recognition Challenge

18 days to go

sampleSubmission.csv
20_characters_illust...
character_name.lst
test.zip
train.zip

train.zip 482.24 MB

URL for Sharing：
https://www.kaggle.com/t/2d96605587334fdc9eb0fabf6756392f


GitHub Classroom

Machine Learning@NTUT - 2017

MachineLearningNTUT

Classification

Individual assigment

Give this to your students

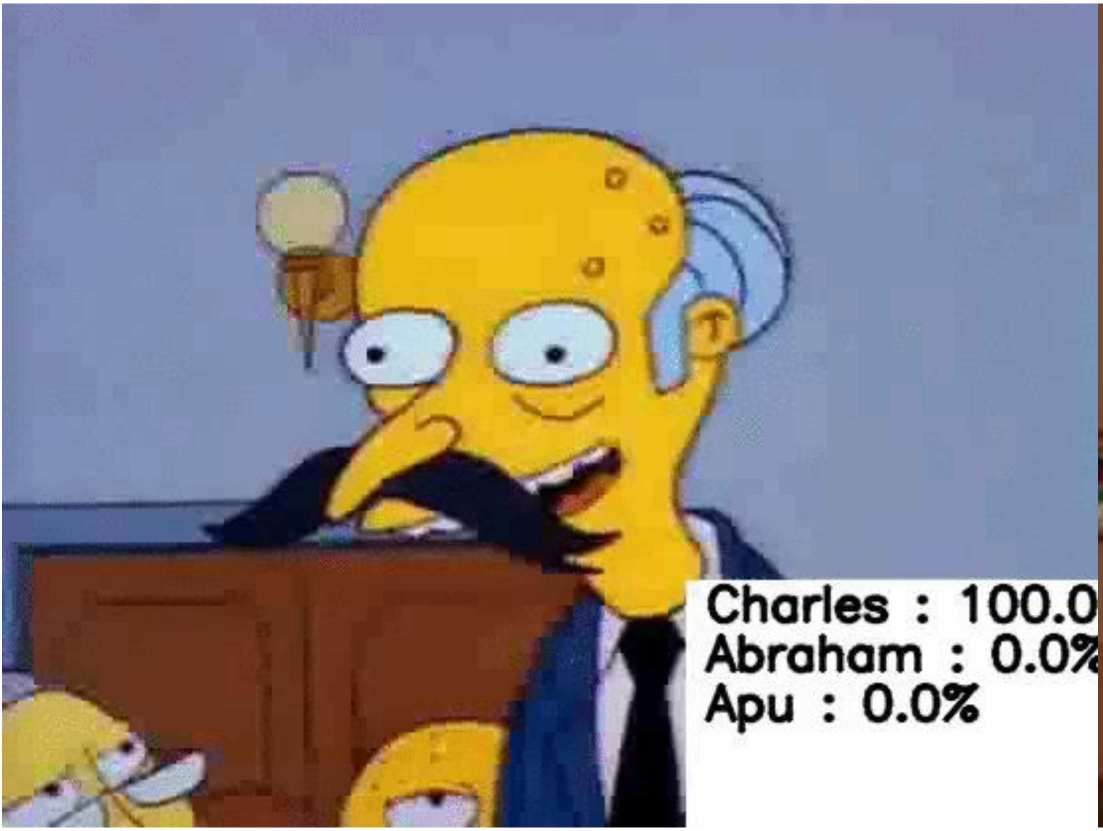
https://classroom.github.com/a/4JnaHLk8

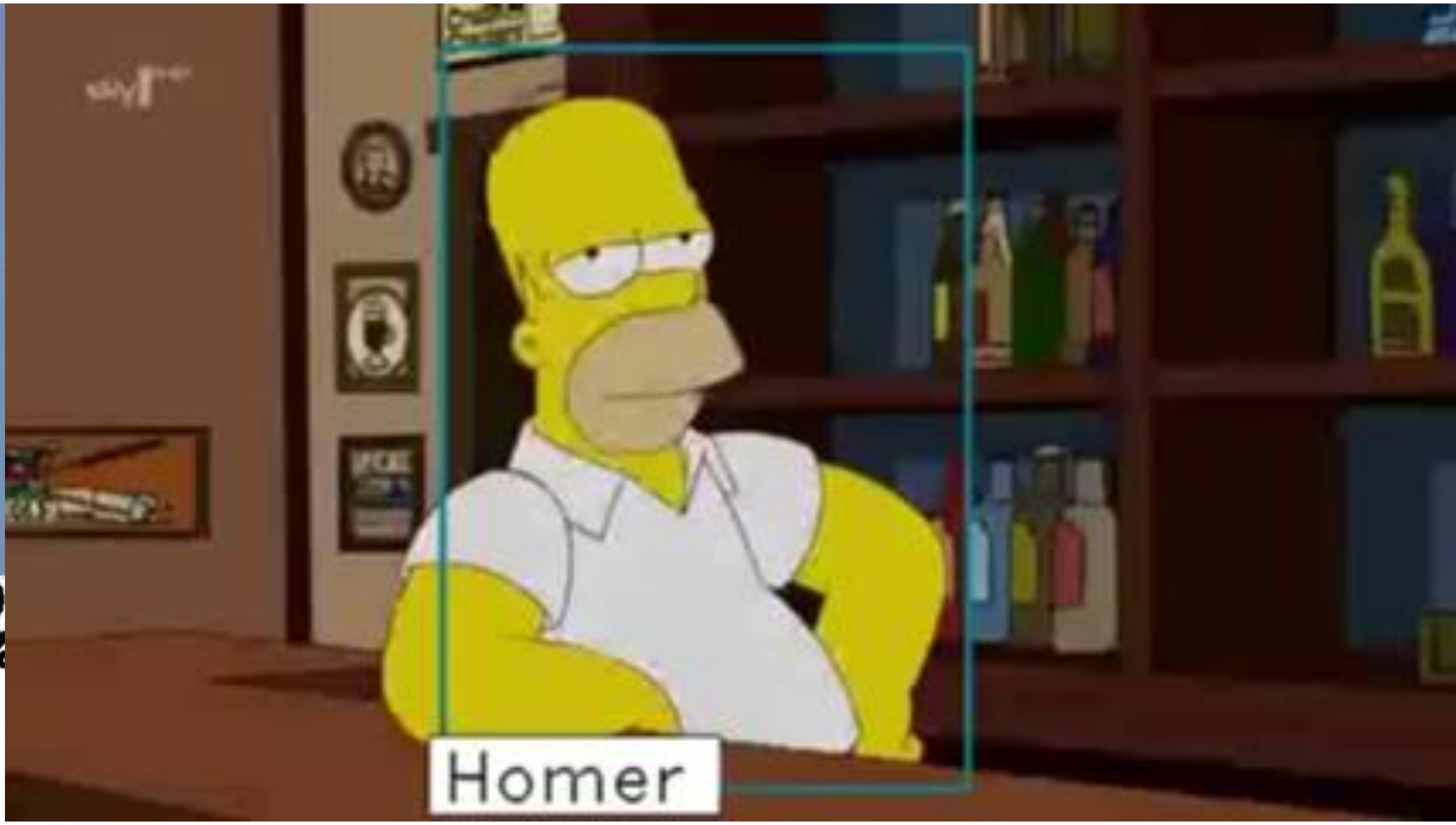
20 Characters in The Simposons

abraham_grampa_simpson apu_nahasapeemapetilon bart_simpson charles_montgomery_burns chief_wiggum
comic_book_guy edna_krabappel homer_simpson kent_brockman krusty_the_clown

lenny_leonard lisa_simpson marge_simpson mayor_quimby milhouse_van_houten
moe_szyslakned_flanders nelson_muntz principal_skinner sideshow_bob

Demo

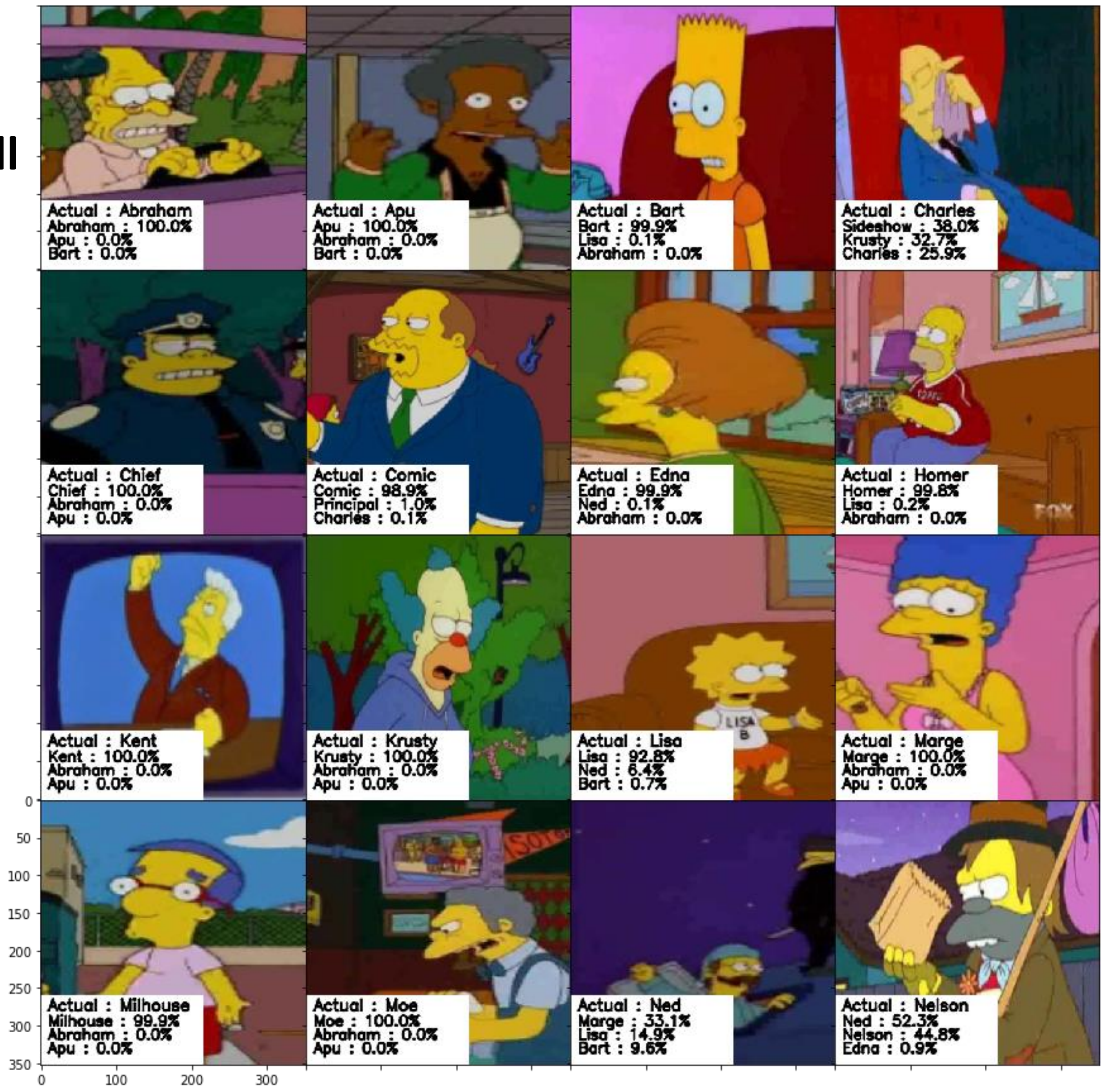




https://thumbs.gfycat.com/HarshCaringIndianpalmsquirrel-mobile.mp4

https://youtu.be/ta_hCXKZ_DM

- # Task1
  - **Predict Simpsons Characters in all pictures**



Actual : Abraham
Abraham : 100.0%
Apu : 0.0%
Bart : 0.0%

Actual : Apu
Apu : 100.0%
Abraham : 0.0%
Bart : 0.0%

Actual : Bart
Bart : 99.9%
Lisa : 0.1%
Abraham : 0.0%

Actual : Charles
Sideshow : 38.0%
Krusty : 32.7%
Charles : 25.9%

Actual : Chief
Chief : 100.0%
Abraham : 0.0%
Apu : 0.0%

Actual : Comic
Comic : 98.9%
Principal : 1.0%
Charles : 0.1%

Actual : Edna
Edna : 99.9%
Ned : 0.1%
Abraham : 0.0%

Actual : Homer
Homer : 99.8%
Lisa : 0.2%
Abraham : 0.0%

Actual : Kent
Kent : 100.0%
Abraham : 0.0%
Apu : 0.0%

Actual : Krusty
Krusty : 100.0%
Abraham : 0.0%
Apu : 0.0%

Actual : Lisa
Lisa : 92.8%
Ned : 6.4%
Bart : 0.7%

Actual : Marge
Marge : 100.0%
Abraham : 0.0%
Apu : 0.0%

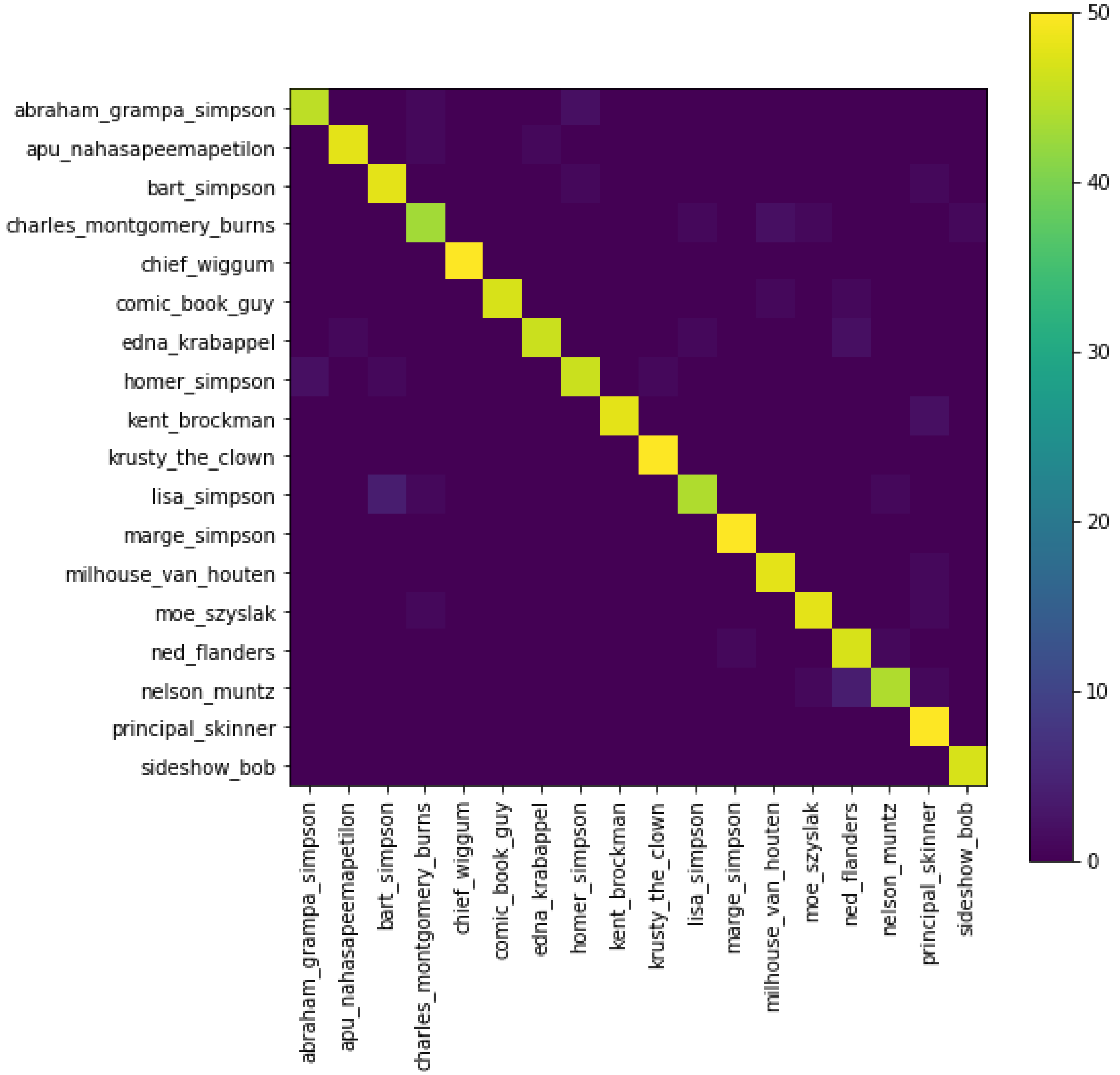
Actual : Milhouse
Milhouse : 99.9%
Abraham : 0.0%
Apu : 0.0%

Actual : Moe
Moe : 100.0%
Abraham : 0.0%
Apu : 0.0%

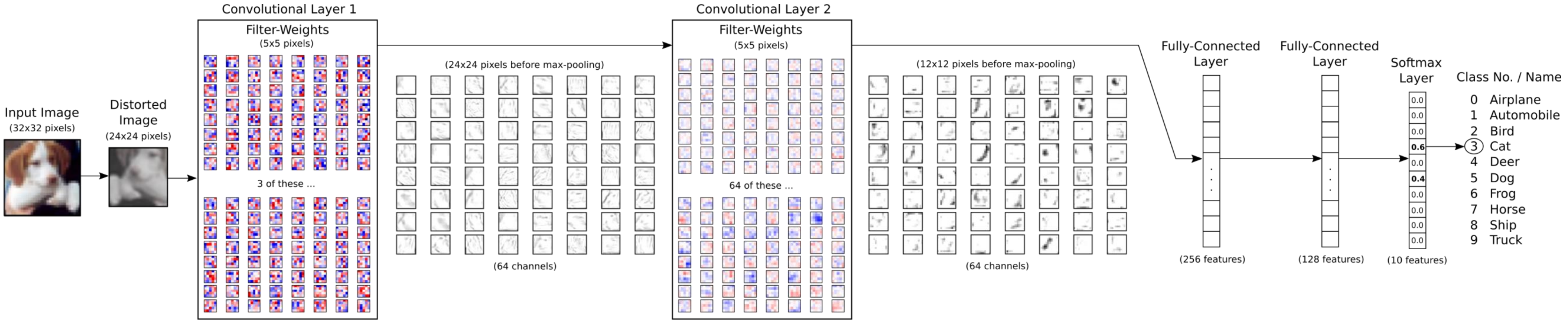
Actual : Ned
Marge : 33.1%
Lisa : 14.9%
Bart : 9.6%

Actual : Nelson
Ned : 52.3%
Nelson : 44.8%
Edna : 0.9%

- # Task2
  - **Compute the Confusion Matrix**



- # Task3
  - ## Visualization and Understanding Convolutional Neural Networks
    - 畫出每一層filter的權重 只需要抓第一層的權重，tensor轉image
    - ~~畫出每一層的feature map~~

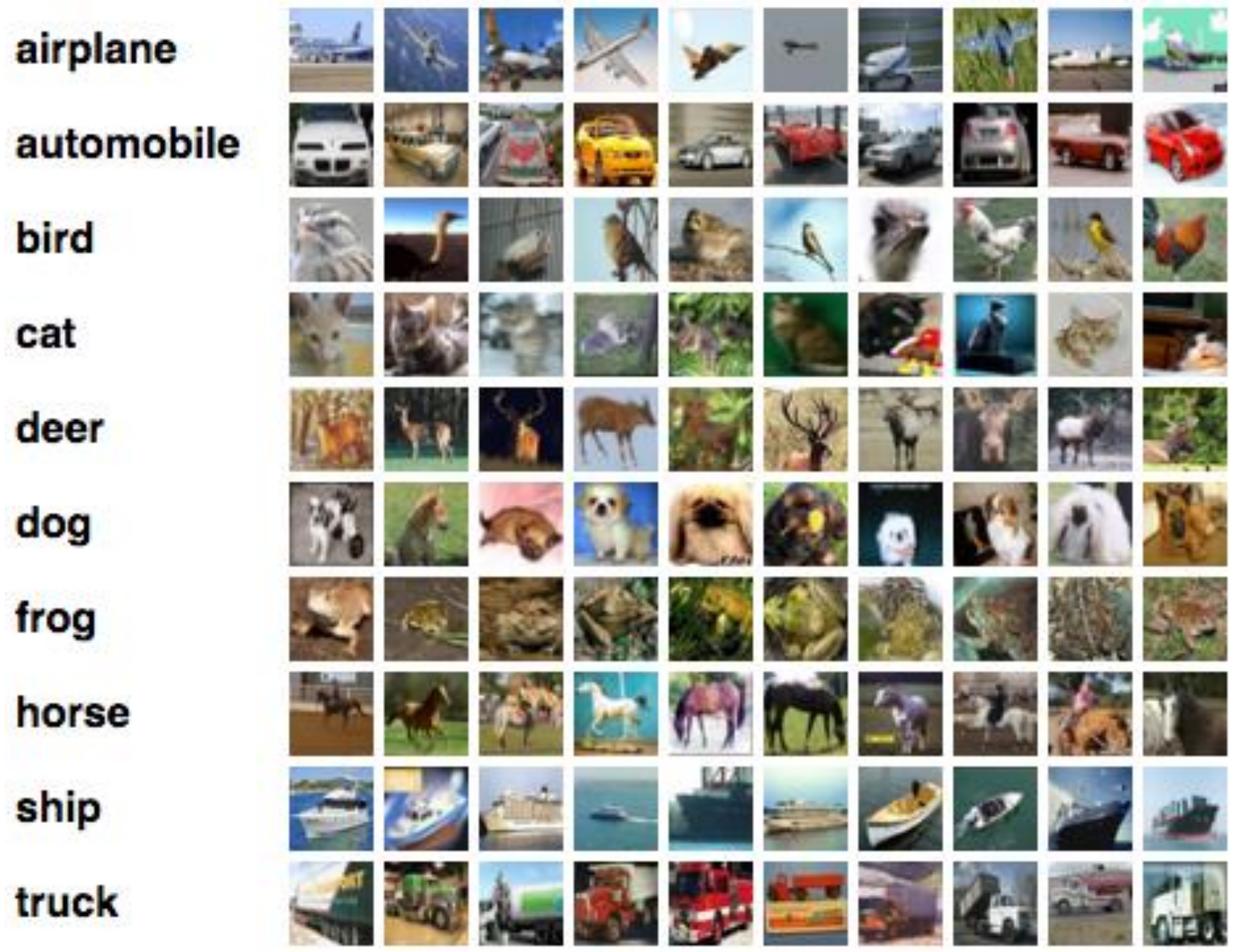


Example

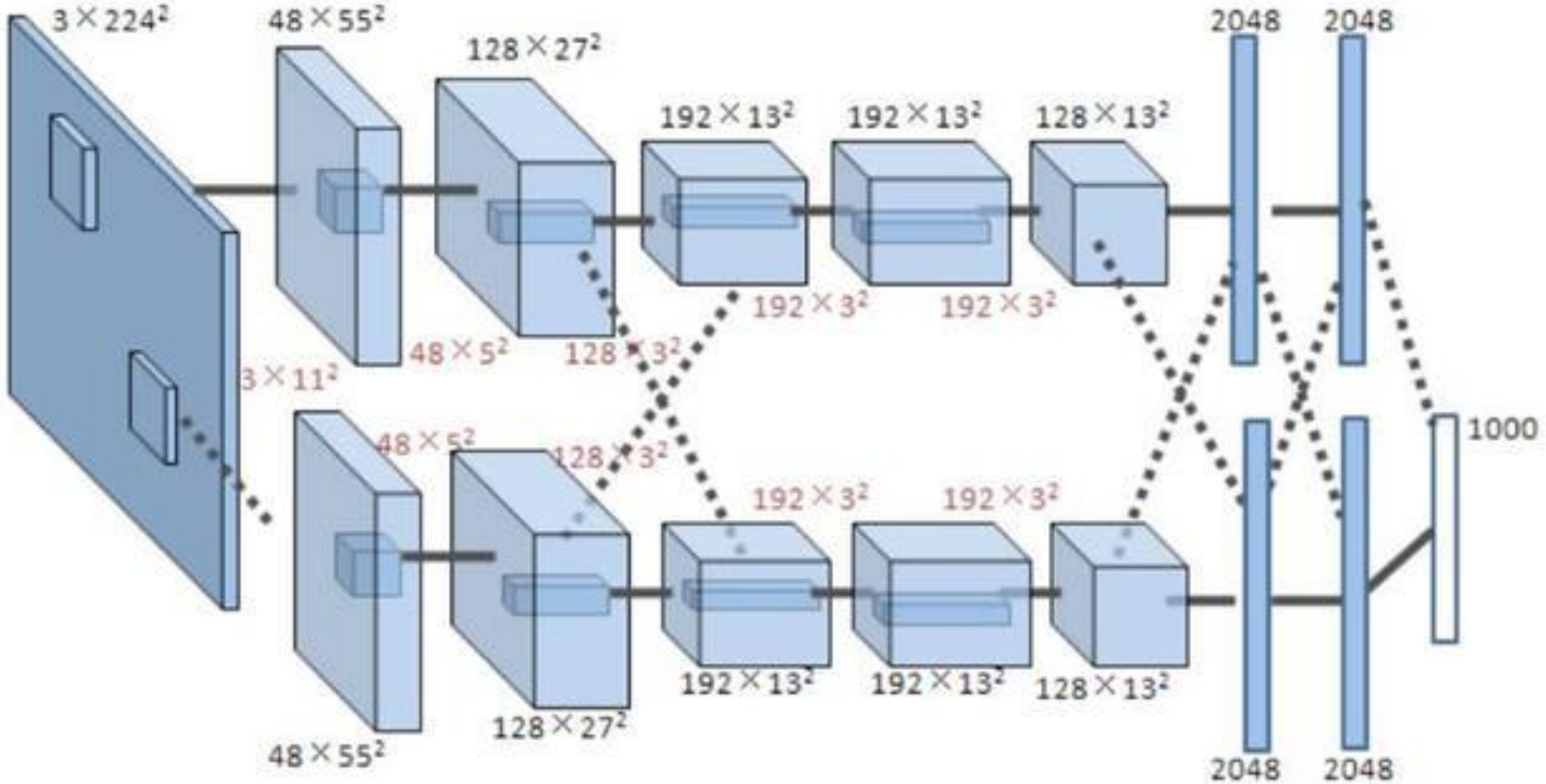
- Tensorflow Tutorial - Convolutional Neural Networks
  - https://www.tensorflow.org/versions/r0.11/tutorials/deep_cnn/index.html
  - The code for this tutorial resides in tensorflow/models/image/cifar10/
  - Accuracy on Test-Set: 79.3% (7932 / 10000)

CIFAR-10

- CIFAR-10 classification is a common benchmark problem in machine learning.
- The problem is to classify RGB 32x32 pixel images across 10 categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck



AlexNet



3×224^2
48×55^2
128×27^2
192×13^2
192×13^2
128×13^2
2048
2048
3×11^2
48×5^2
128×3^2
192×3^2
192×3^2
48×5^2
128×3^2
192×3^2
192×3^2
1000
192×13^2
192×13^2
128×13^2
128×27^2
48×55^2
2048
2048

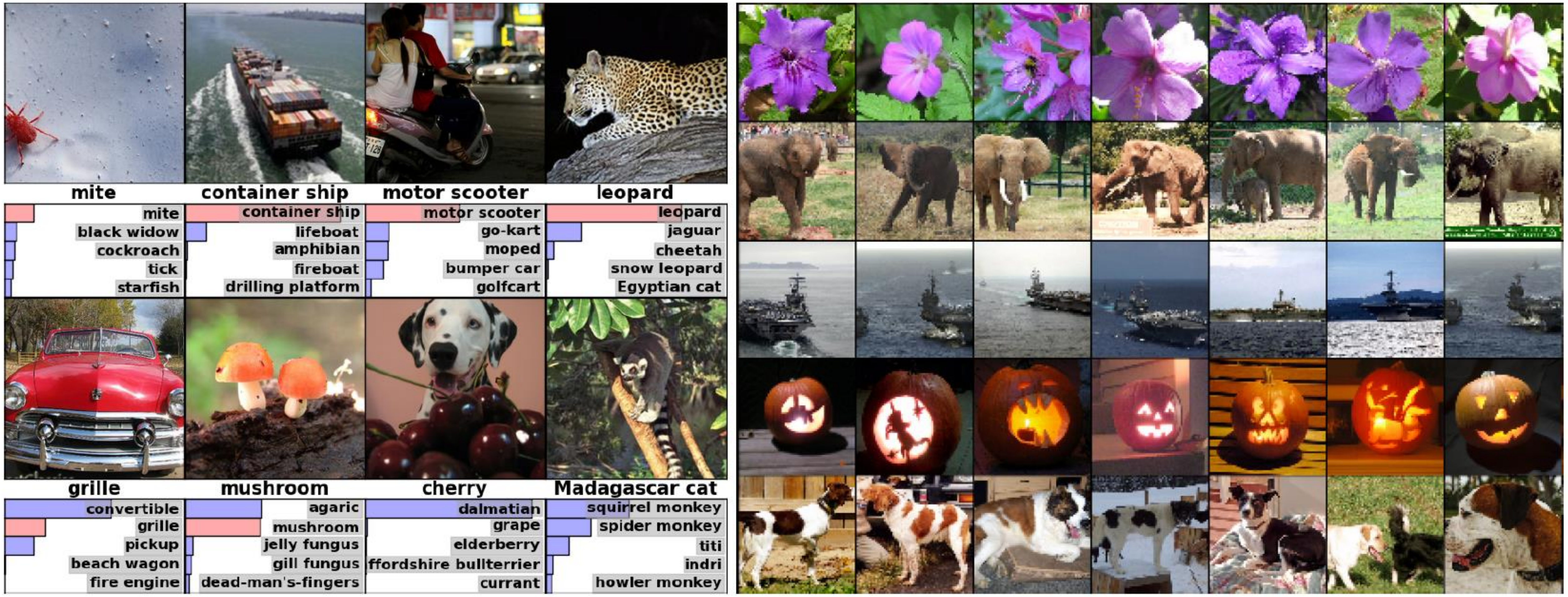


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 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.

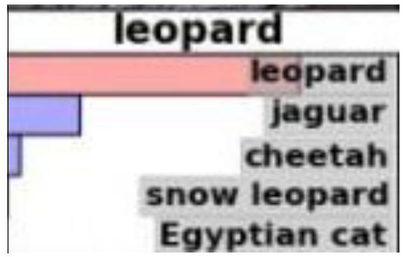
References

- TensorFlow Tutorial CIFAR-10
  - https://www.youtube.com/watch?v=3BXfw_1_TF4
- TensorFlow CIFAR-10 tutorial, detailed step-by-step review
  - Part 1: http://www.aimechanic.com/2016/10/13/d242-tensorflow-cifar-10-tutorial-detailed-step-by-step-review-part-1/
  - Part 2: http://www.aimechanic.com/2016/10/17/d246-tensorflow-cifar-10-tutorial-detailed-review-part-2/
- TensorFlow-Tutorials/06_CIFAR-10.ipynb
  - https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/06_CIFAR-10.ipynb

ImageNet Classification with Deep Convolutional Neural Networks

Goa
l

Classifica(on



ImageNe
t

- Over 15M labeled high resolution images
- Roughly 22K categories
- Collected from web and labeled by Amazon Mechanical Turk

h-p://image--net.org/

ILSVRC

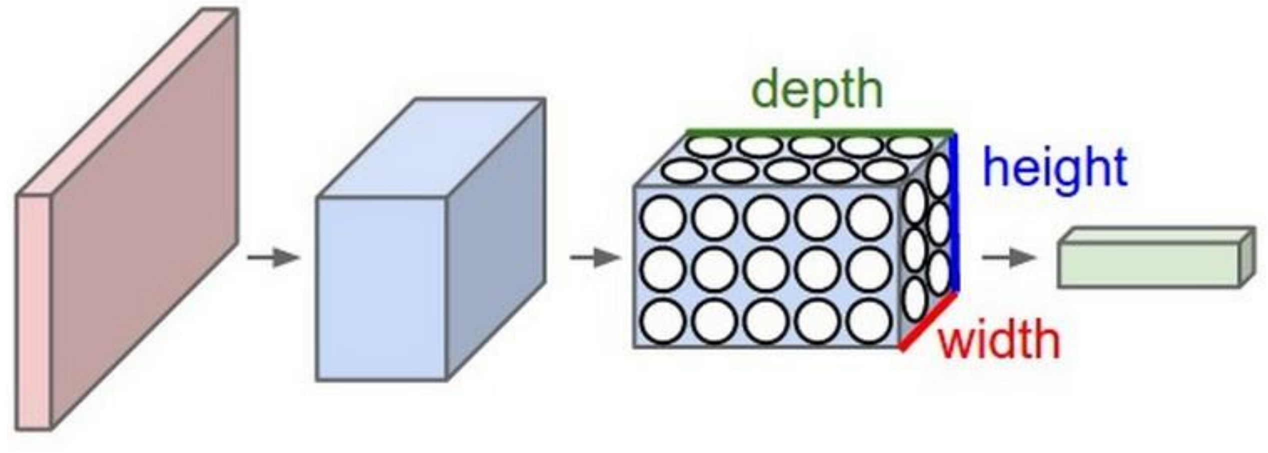
- Annual competition of image classification at large scale
- 1.2M images in 1K categories
- Classification: make 5 guesses about the image label

EntleBucher

Appenzeller

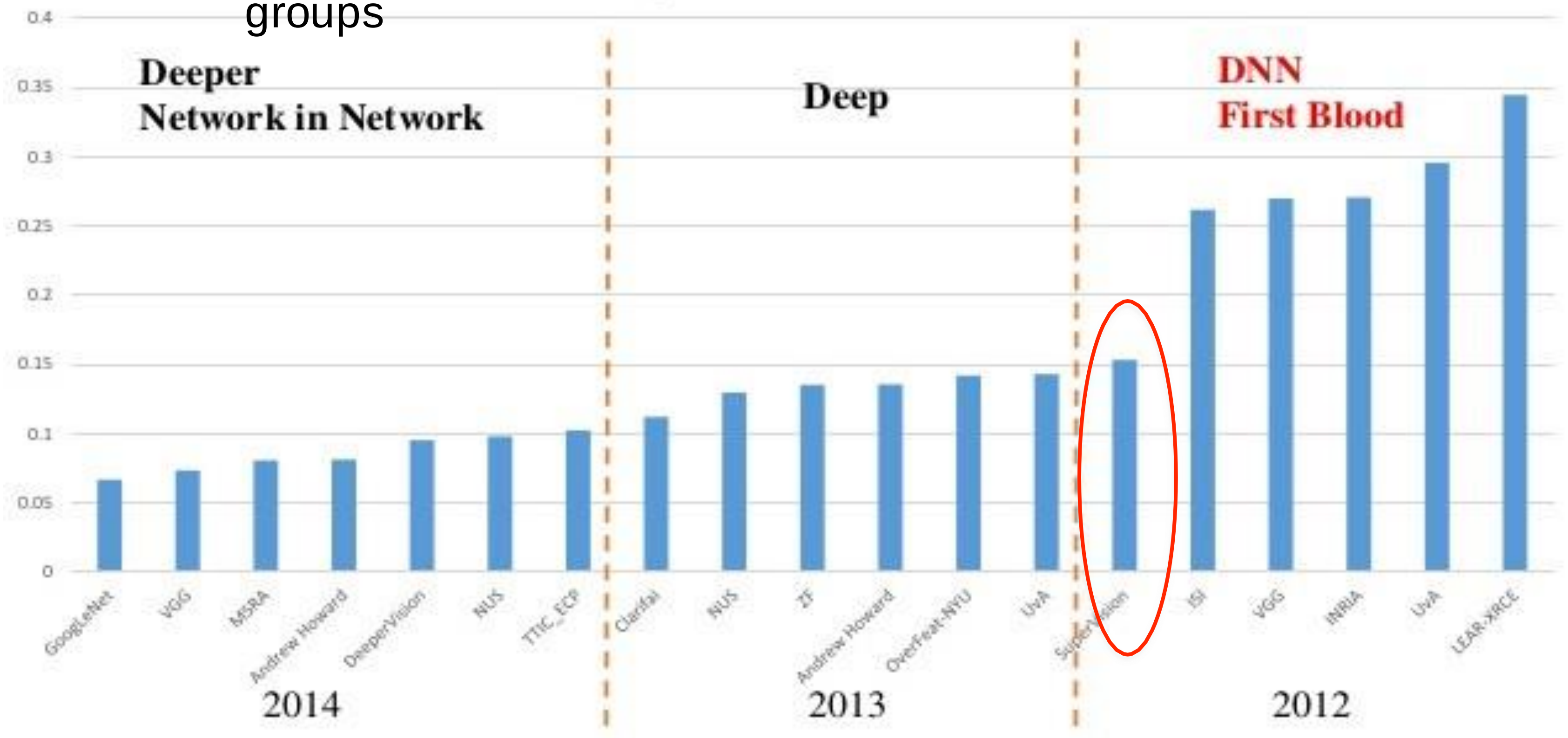
ConvolutionalNeural Networks

- Model with a large learning capacity
- Prior knowledge to compensate all data we do not have

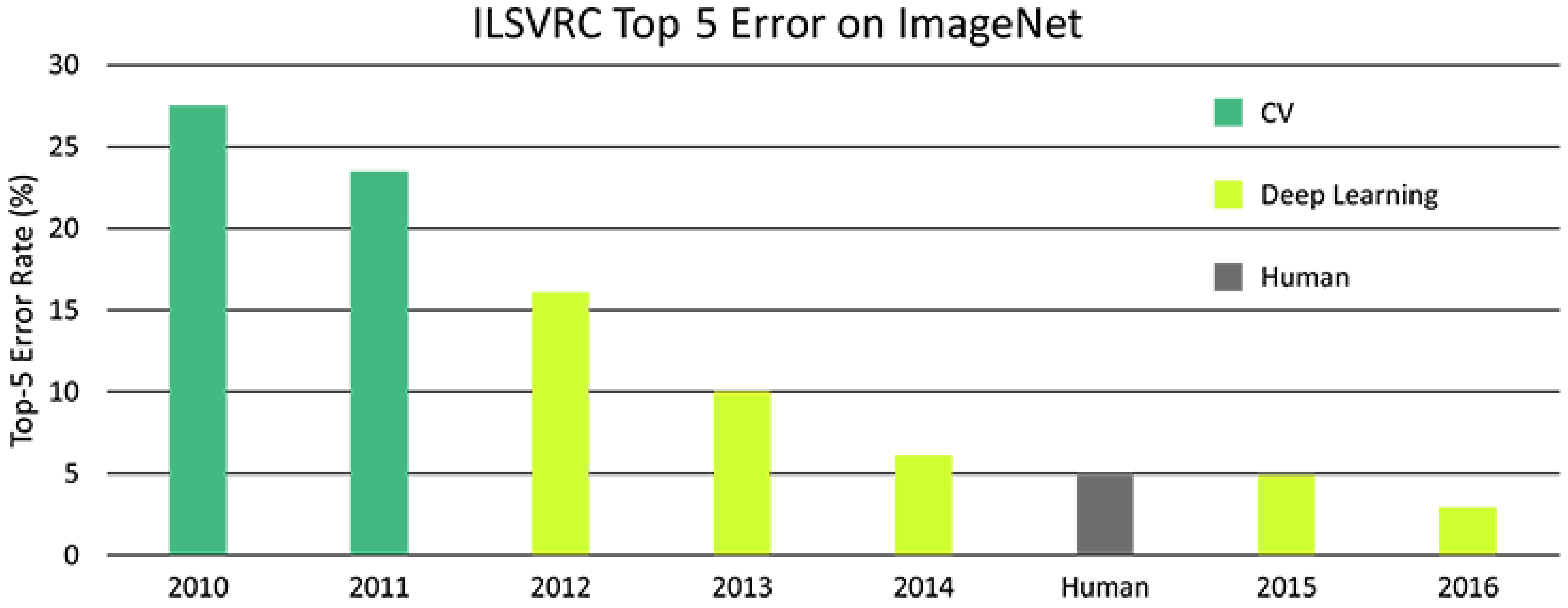


ILSVRC

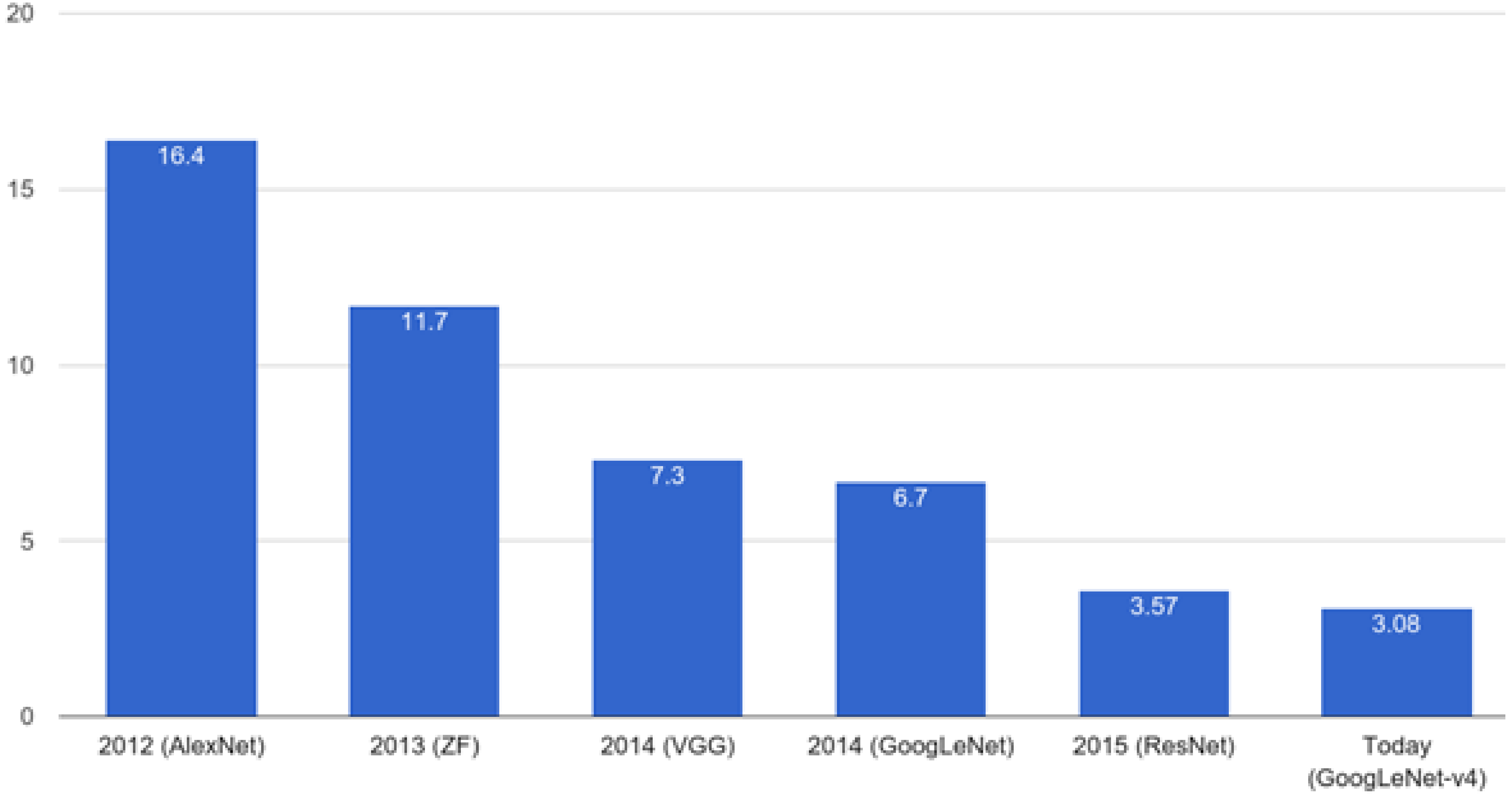
ImageNet Classification error throughout years and groups



Li Fei-Fei: ImageNet Large Scale Visual Recognition Challenge, 2014 http://image-net.org/


ILSVRC Top 5 Error on ImageNet
Top-5 Error Rate (%)
30
25
20
15
10
5
0
2010
2011
2012
2013
2014
Human
2015
2016
CV
Deep Learning
Human

ImageNet Classification Error (Top 5)



SuperVision (SV)

Image classification with deep convolutional neural networks

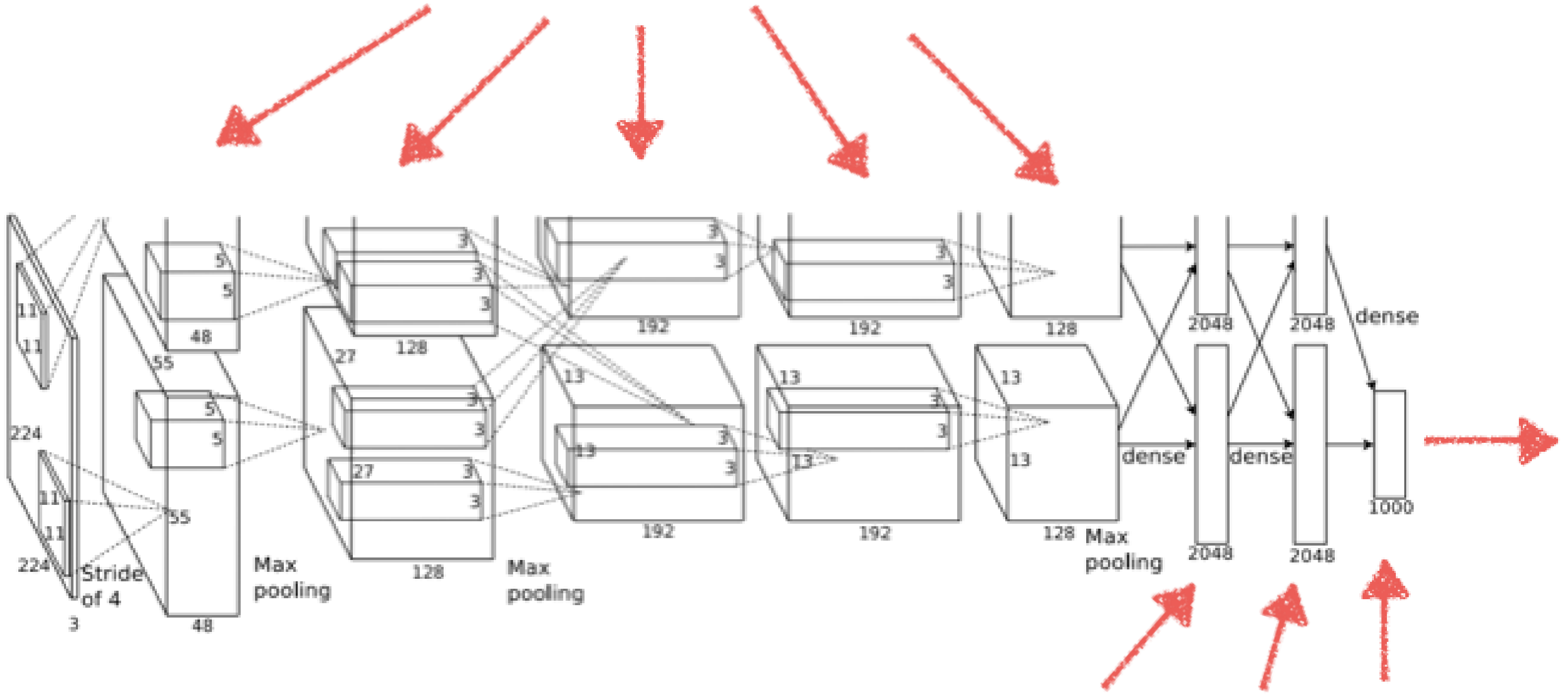
- 7 hidden “weight” layers
- 650K neurons
- 60M parameters
- 630M connections

- Rectified Linear Units, overlapping pooling, dropout trick
- Randomly extracted 224x224 patches for more data

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

Architectur e

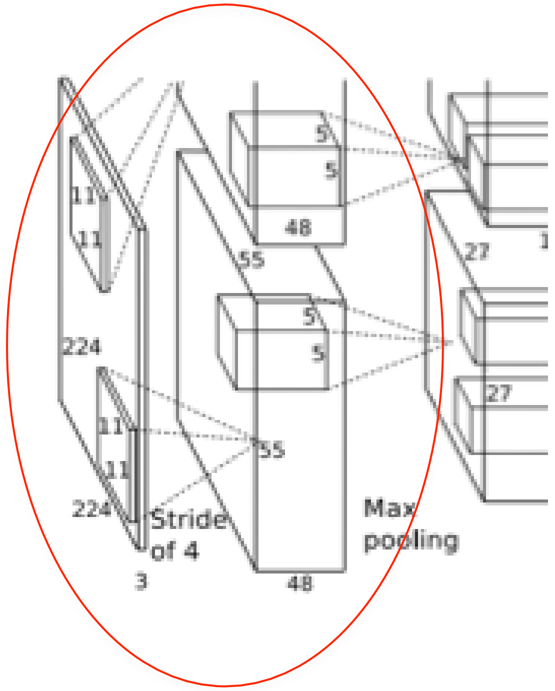
5 Convolu(onal Layers



1000-way soLmax

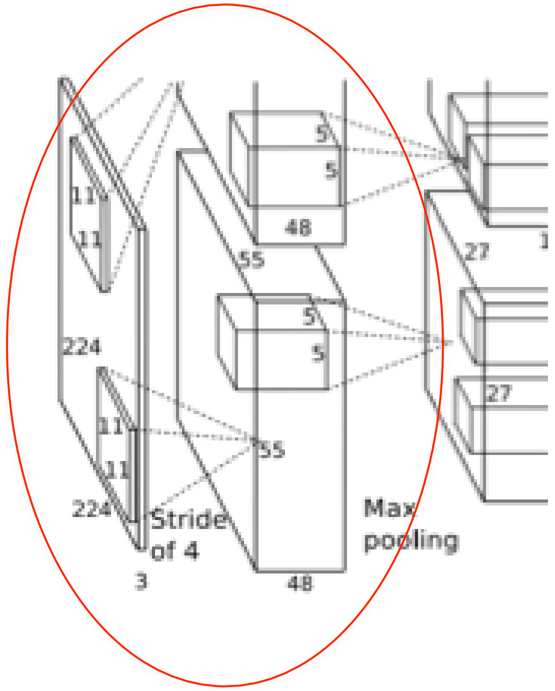
3 Fully Connected Layers

Layer 1 (Convolutional)



- Images: 227x227x3
- 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!

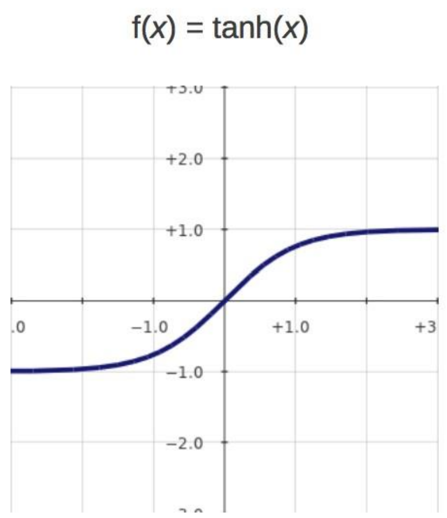
Architecture

RELU Nonlinearity

- Standard way to model a neuron

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

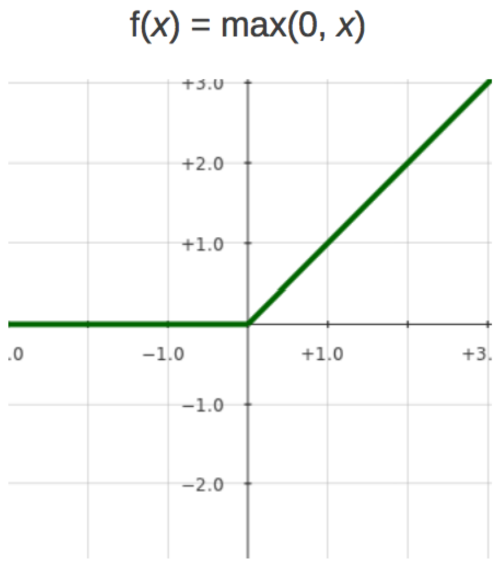
Very slow to train



- Non-saturating nonlinearity (RELU)

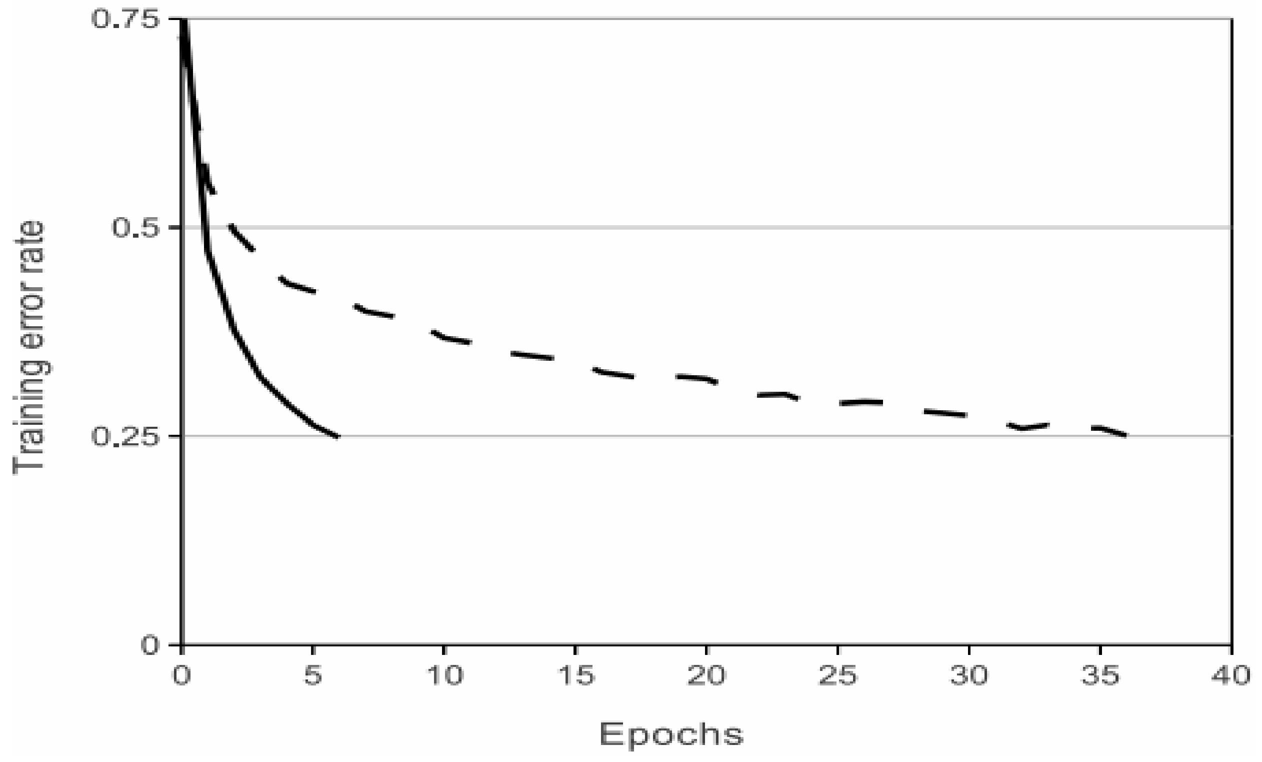
$$f(x) = \max(0, x)$$

Quick to train



Architectur e

RELU Nonlinearity



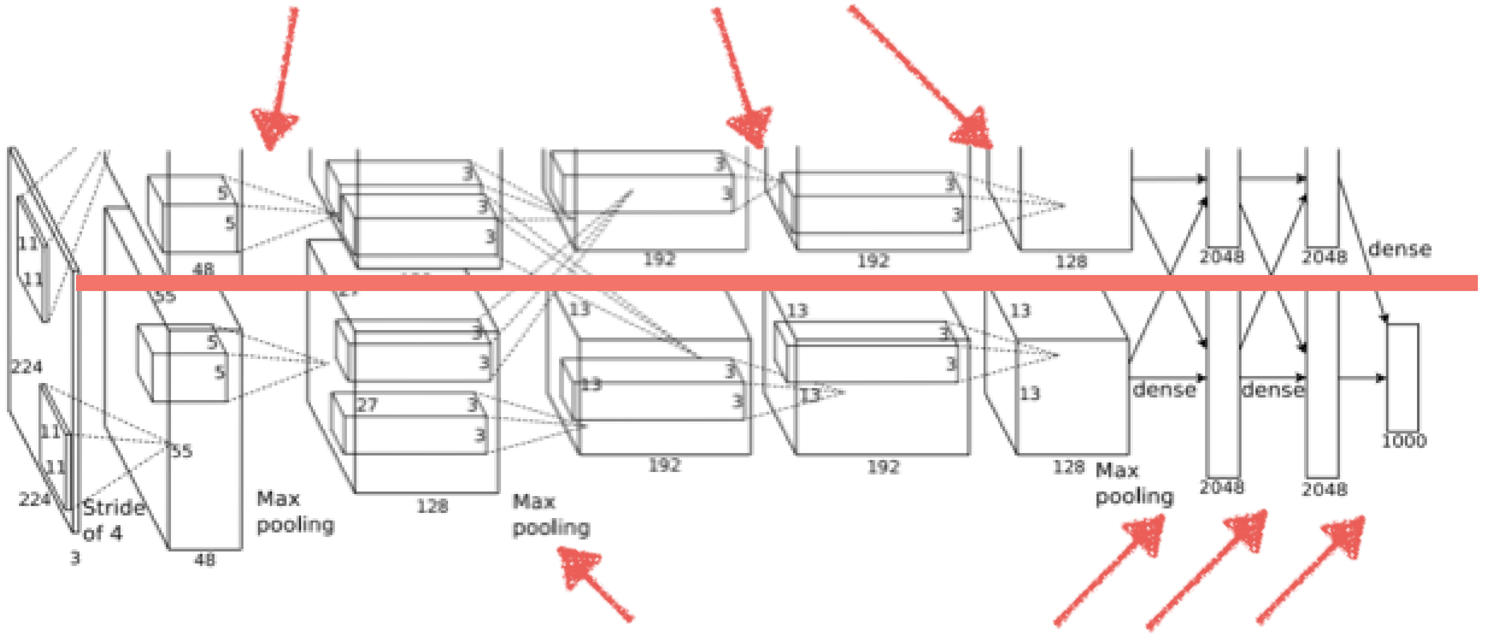
A 4 layer CNN with ReLUs (solid line) converges six times faster than an equivalent network with tanh neurons (dashed line) on CIFAR-10 dataset

Architecture

Training on Multiple GPUs

GPU #1

intra--GPU connec(ons



GPU #2

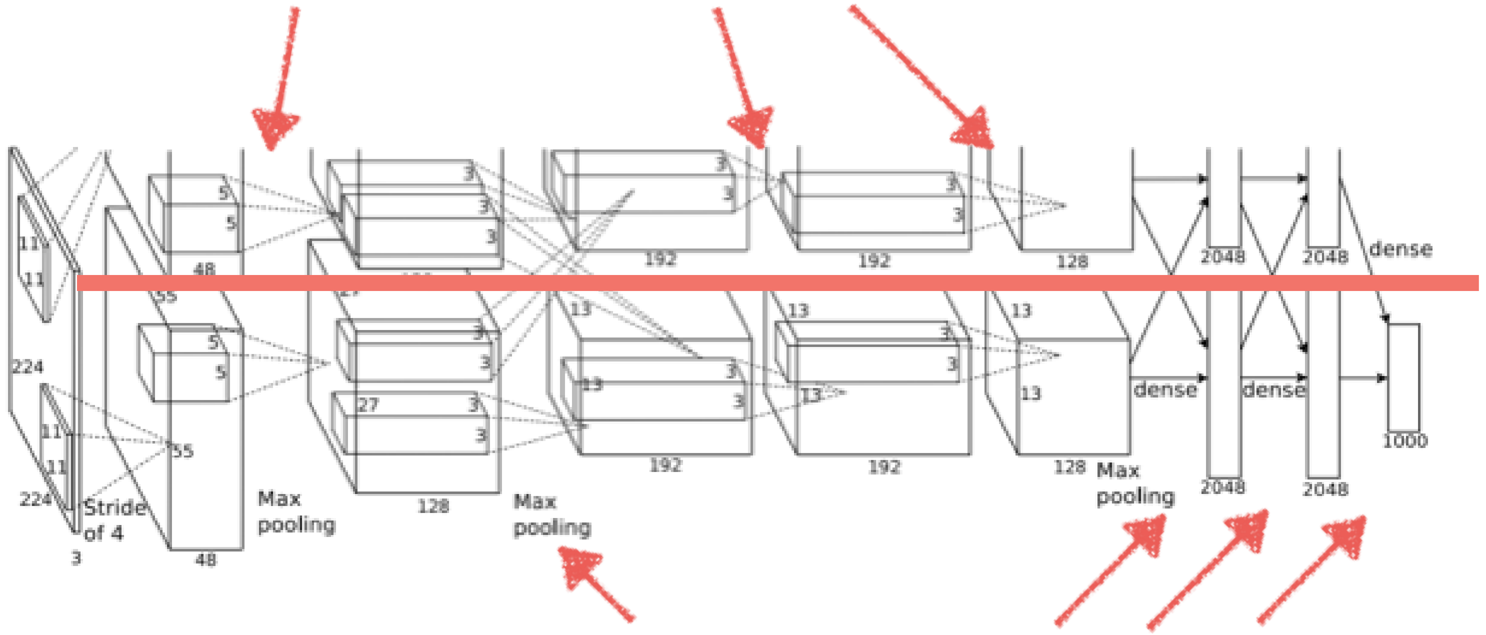
inter--GPU connec(ons

Architecture

Training on Multiple GPUs

GPU #1

intra--GPU connec(ons



GPU #2

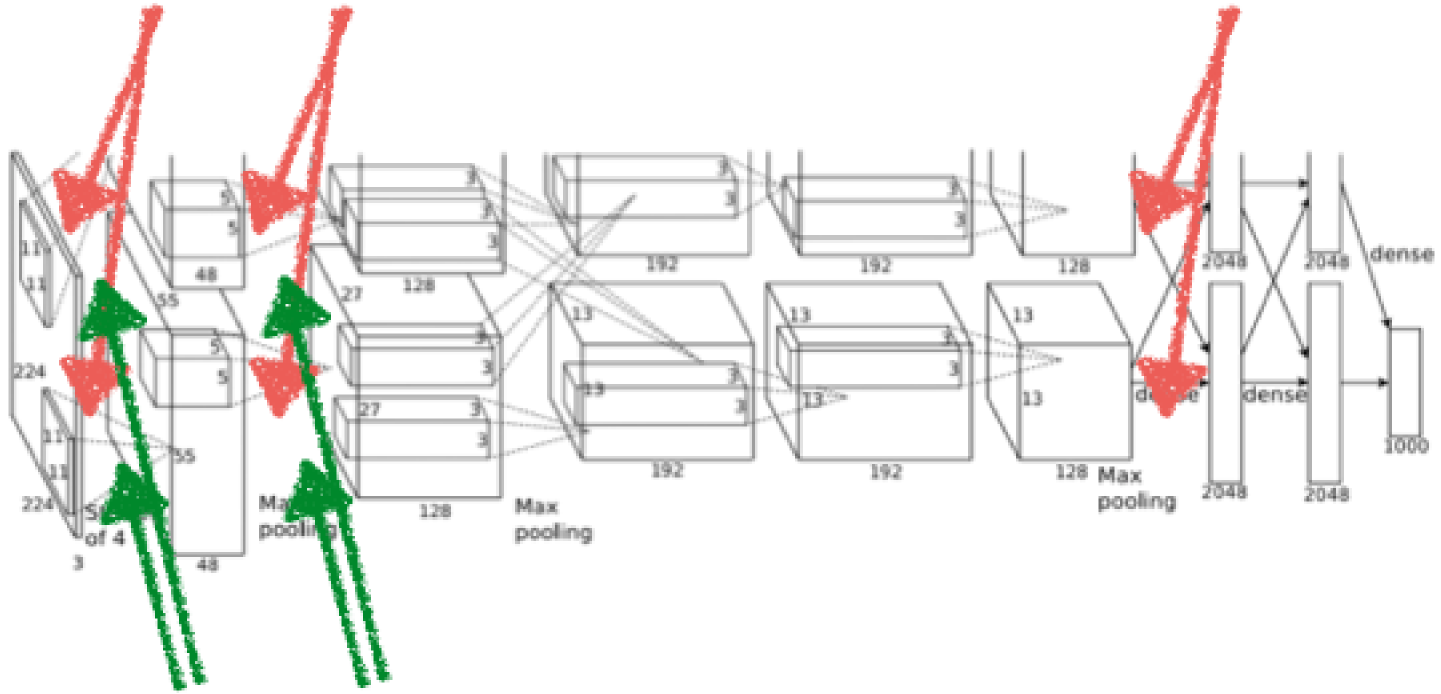
inter--GPU connec(ons

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

Overlaping Pooling

Max-pooling layers



Response normalization layers

Architecture

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 ac(vity

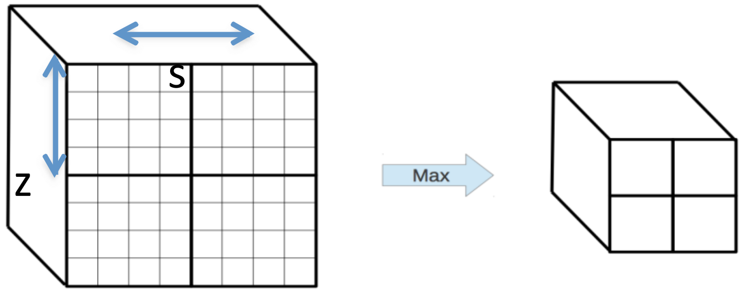
Ac(vity of a neuron computed by applying kernel I at posi(on (x,y) and then applying the ReLU nonlinearity

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

Architecture

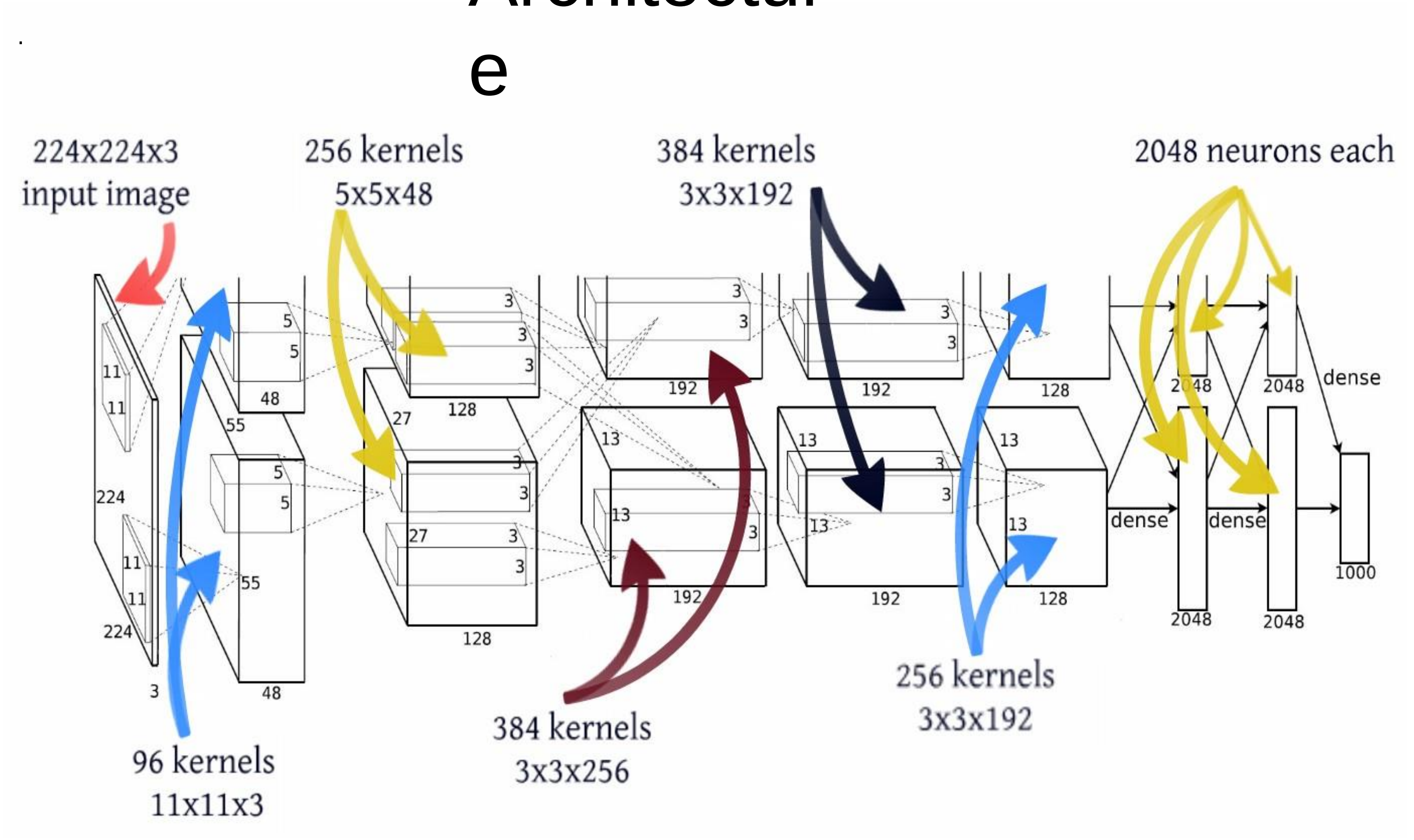
Overlaping Pooling

- Traditional pooling (s = z)

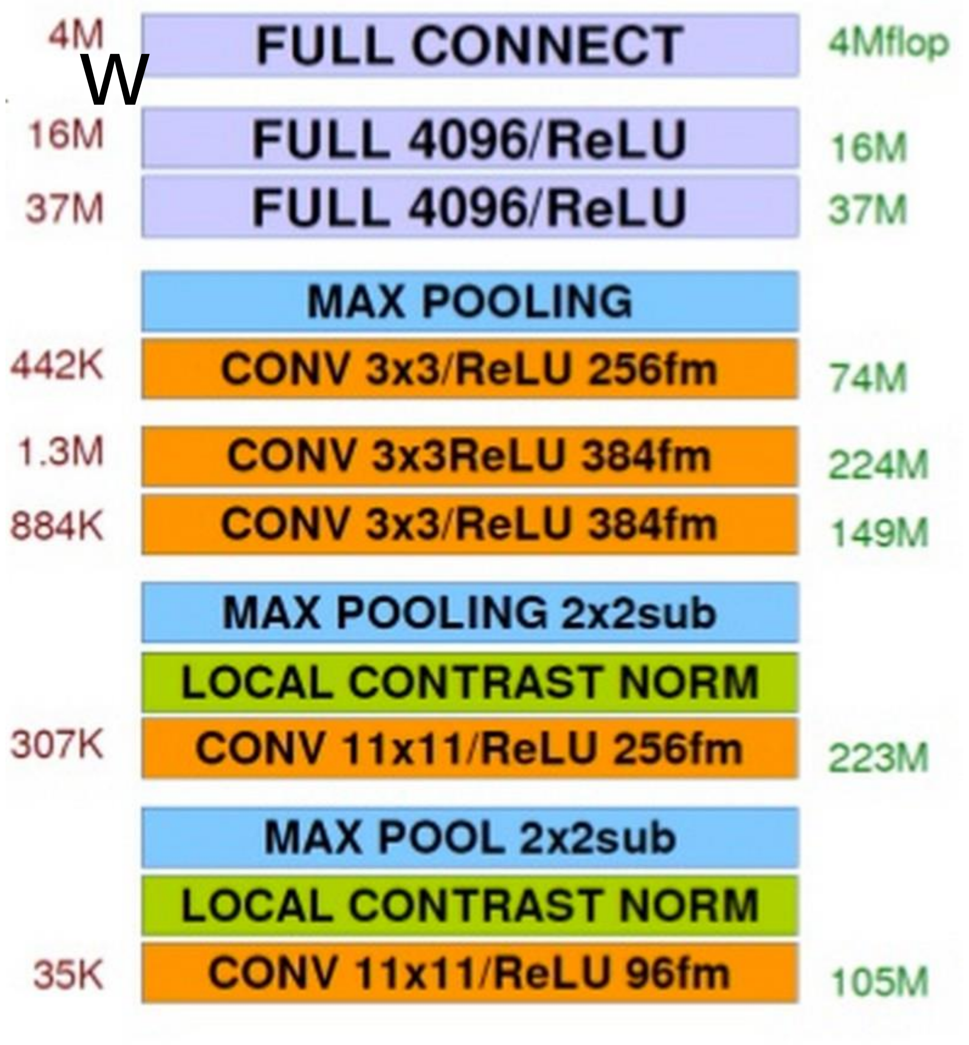


- s < z ➔ overlapping pooling
- top-1 and top-5 error rates decrease by 0.4% and 0.3%, respectively, compared to the non-overlapping scheme s = 2, z = 2

Architectur e



Architecture Overview



Reducing Overfitting

Data Augmentation

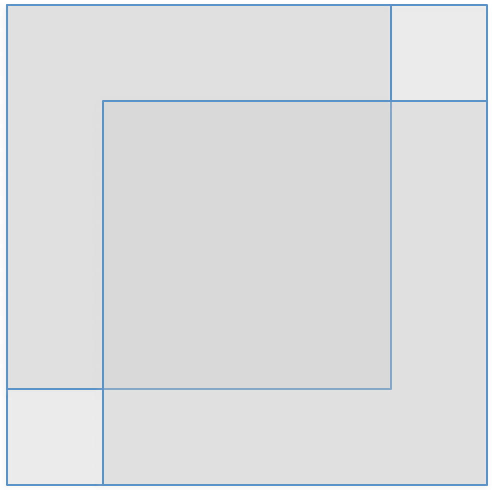
- •• 60 million parameters, 650,000 neurons
  → Overfits a lot.

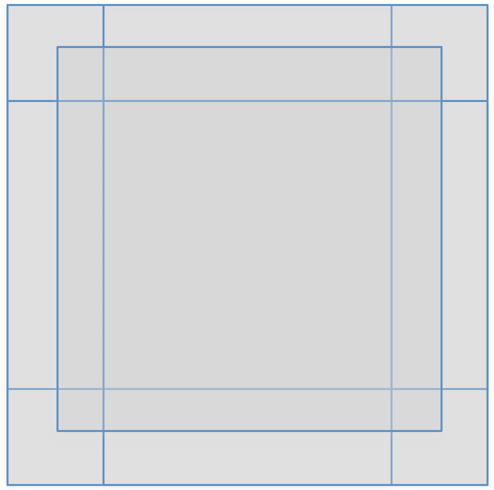
- ••Crop 224x224 patches (and their horizontal reflections.)

Reducing Overfitting

Data Augmentation

- •At test time, average the predictions on the 10 patches.





Reducing

- Softmax

$$L = \frac{1}{N}\sum_i -\log\left(\underbrace{\boxed{\frac{e^{f_{y_j}}}{\sum_j e^{f_j}}}}_{\boxed{P(y_i \mid x_i; W)}}\right) + \lambda \sum_k \sum_l W_{k,l}^2$$

Likelihood

j = 1…1000

- No need to calibrate to average the predictions over 10 patches.

cf. SVM

$$L = \frac{1}{N}\sum_i \sum_{j \neq y_i}\left[\max(0, f(x_i;W)_j - f(x_i;W)_{y_i} + \Delta) + \lambda \sum_k \sum_l W_{k,l}^2\right]$$

Slide credit from Stanford CS231N Lecture 3

Reducing Overfitting

Data Augmentation

- Change the intensity of RGB channels
-

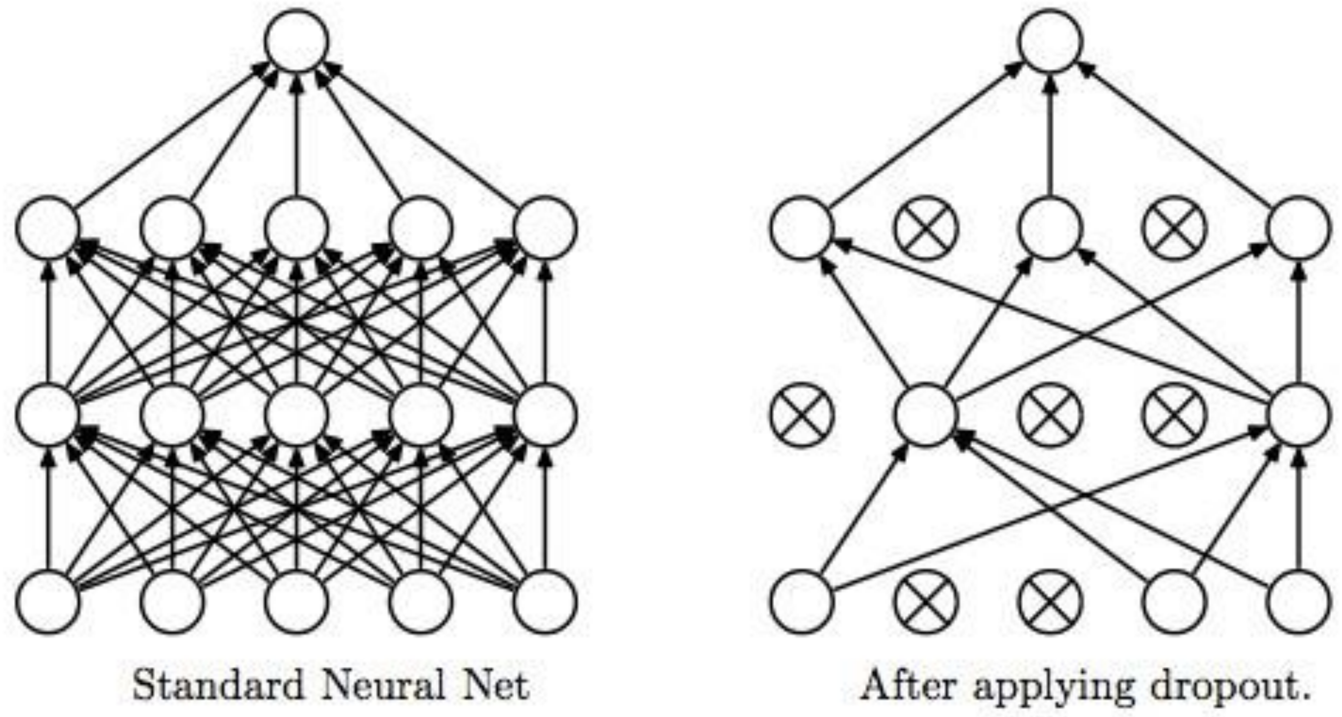
$$I_{xy} = [I_{xy}, \quad I_{xy}^{G}, \quad I_{xy}^{BR}]^{T}$$

add multiples of principle components

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

$$\langle_{i} \sim N(0, \quad 0.1)$$

Reducing Overfitting Dropou



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

Figure credit from Srivastava et al.

Stochastic Gradient Descent Learning

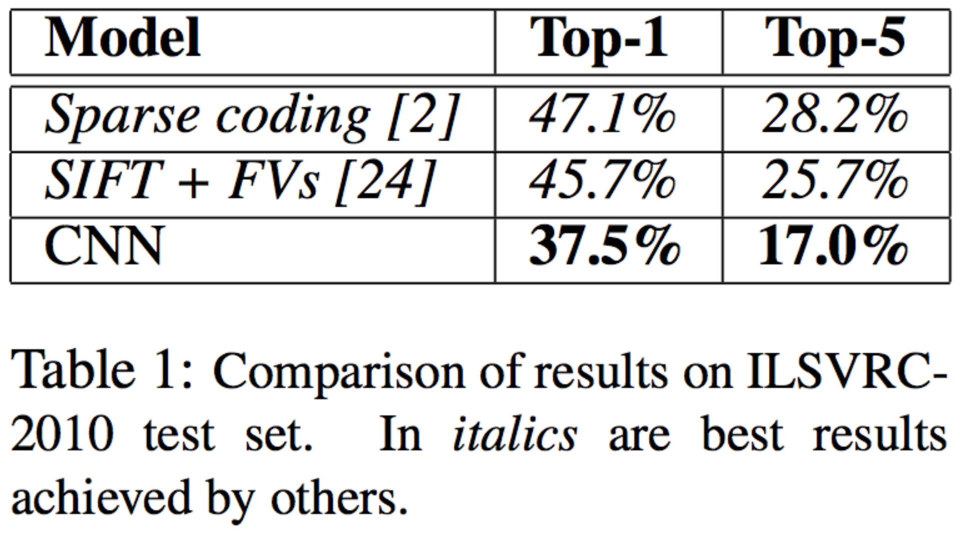
Momentum Update

$$
\begin{aligned}
v_{i+1} &:= \overbrace{0.9}^{\text{momentum(damping parameter)}} \cdot v_i - \underbrace{0.0005 \cdot \epsilon \cdot w_i}_{\text{weight decay}} - \overbrace{\epsilon}^{\text{Learning rate (ini(alized at 0.01)}} \cdot \underbrace{\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}}_{\text{Gradient of Loss w.r.t weight Averaged over batch}} \\
w_{i+1} &:= w_i + v_{i+1}
\end{aligned}
$$

Batch size: 128

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

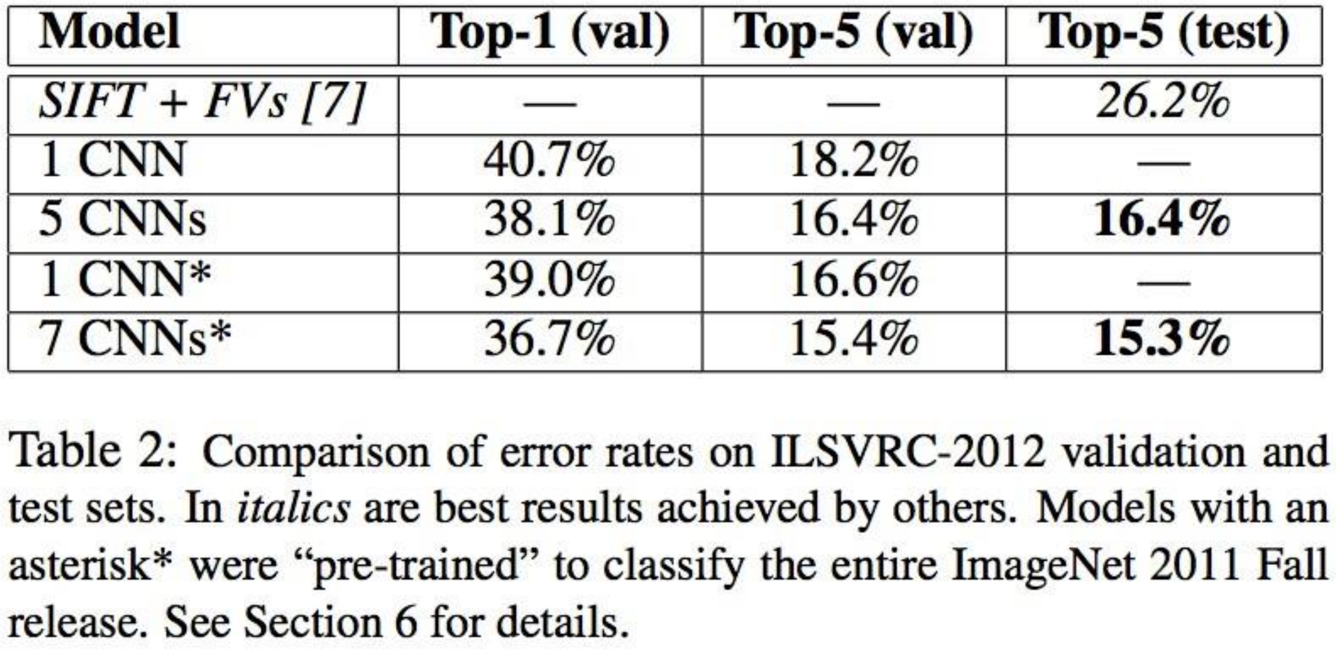
Results: ILSVRC-2010



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.

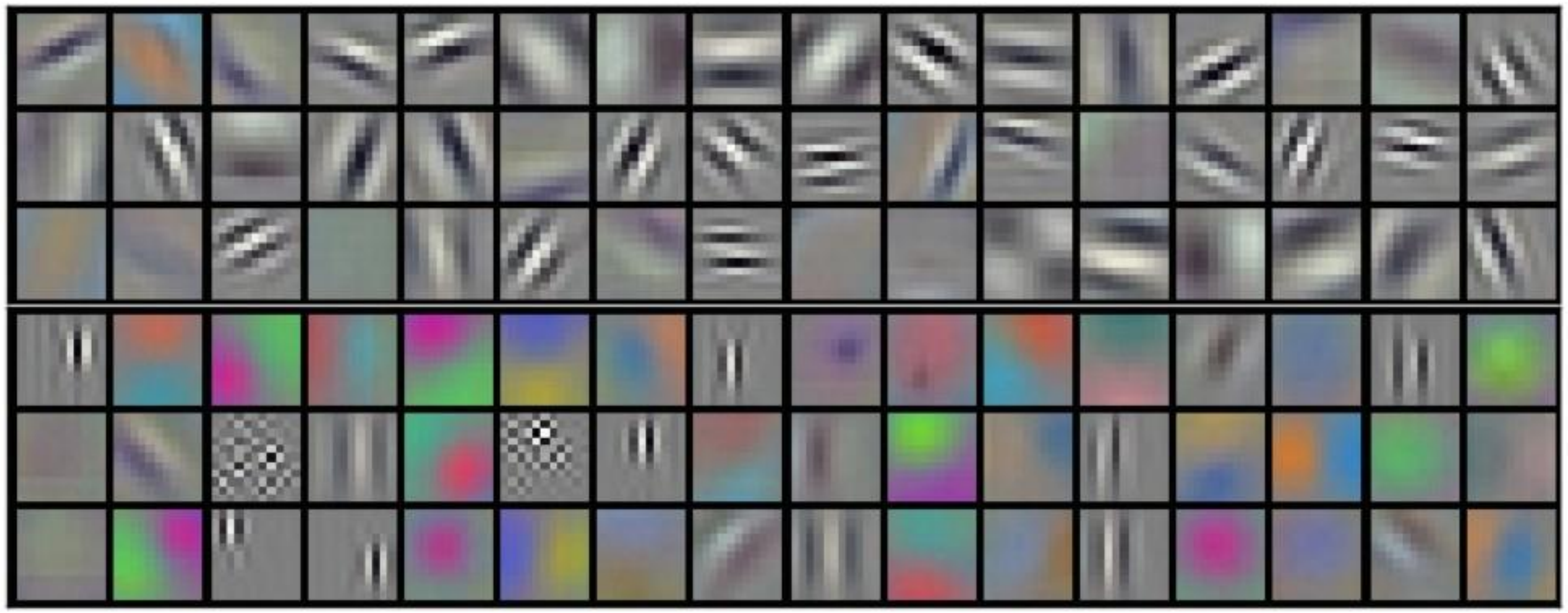
Results: ILSVRC-2012



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.

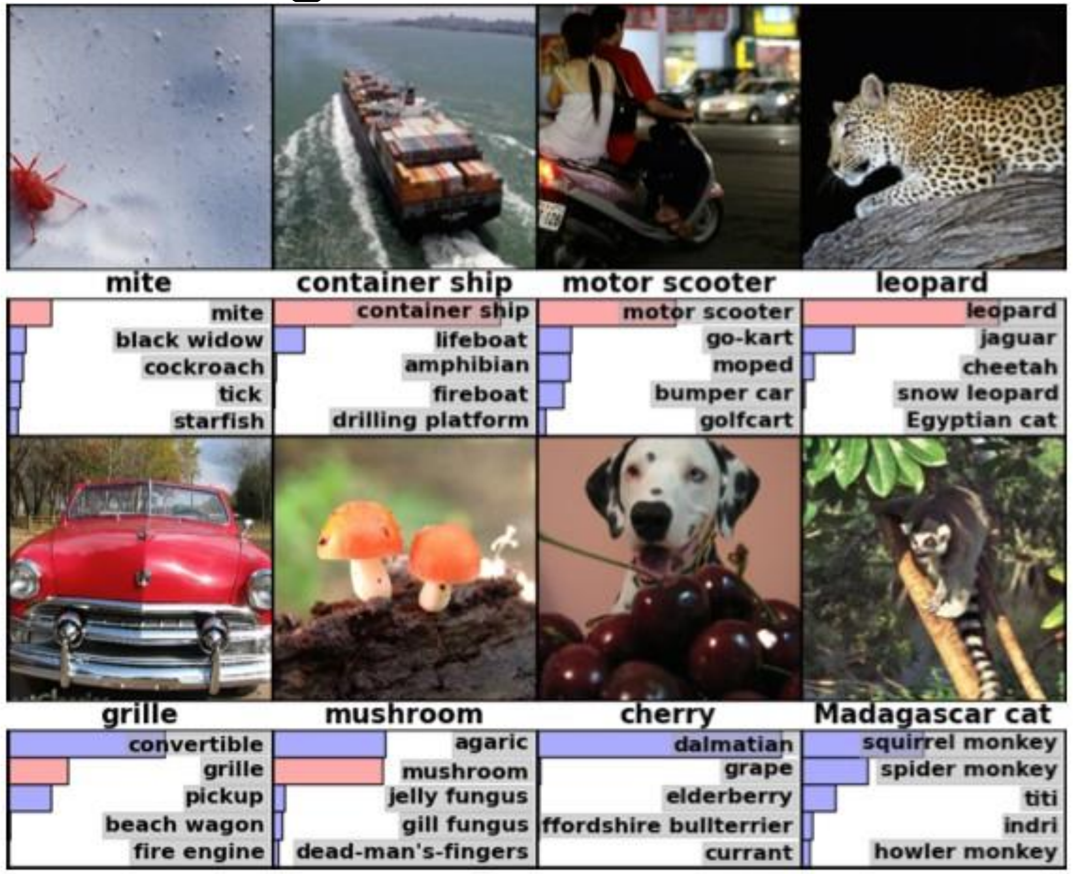
96 Convolutional



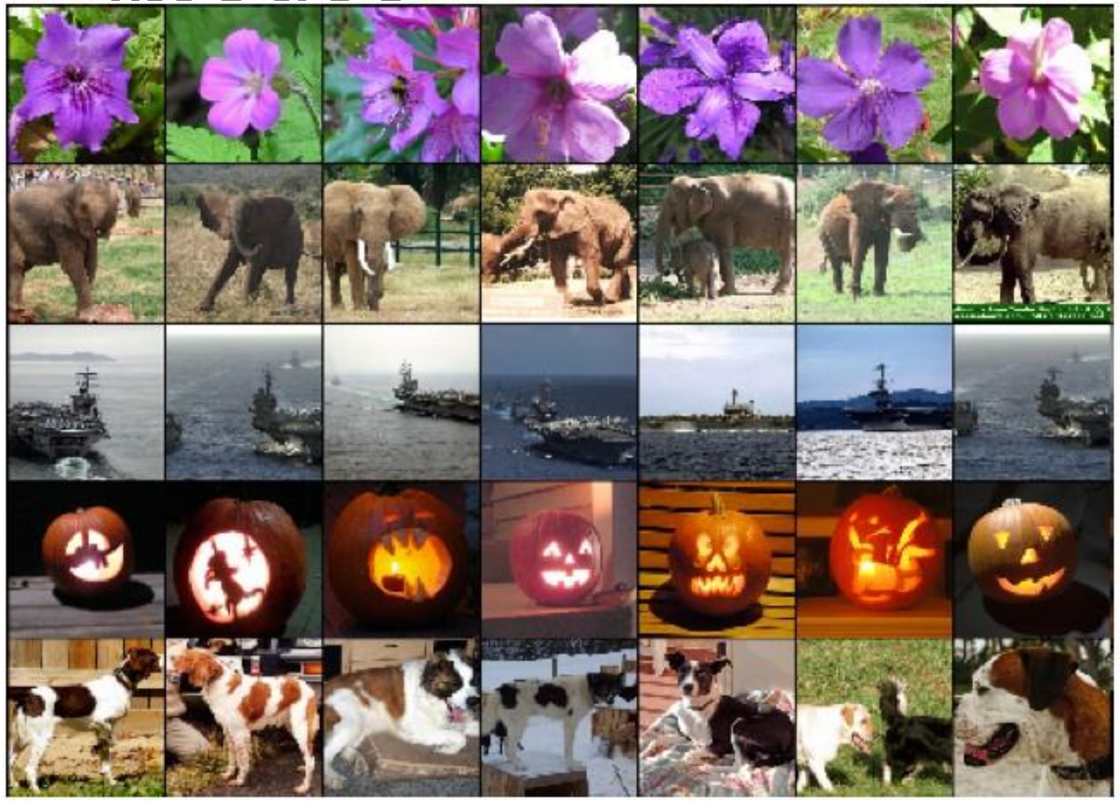
- 11 x 11 x 3 size kernels.
- top 48 kernels on GPU 1 : color-agnostic
- bottom 48 kernels on GPU 2 : color-specific.

Why?

Eight ILSVRC-2010test images



Five ILSVRC-2010 test



The output from the last 4096 fully-connected layer

: 4096 dimensional feature.

Discussion

- • Depth is really important.

  removing a single convolutional layer degrades the performance.

  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