Convolutional Neural Networks

(Application in Object and Scene Recognition)

Harsh Agrawal

(Sept 8th, 2015)

ECE: 6504, Deep Learning For Perception

Contents

A bit of history:

- Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998]
- Three key ideas: Local Receptive Fields, Shared Weights, Sub-sampling.

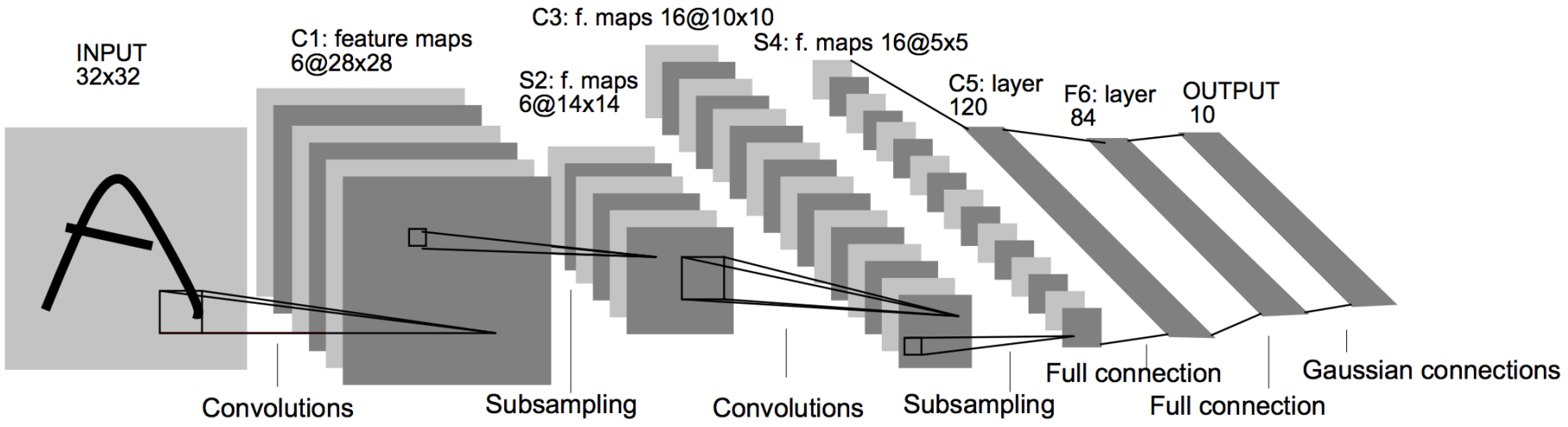


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

LeNet 5, Overview

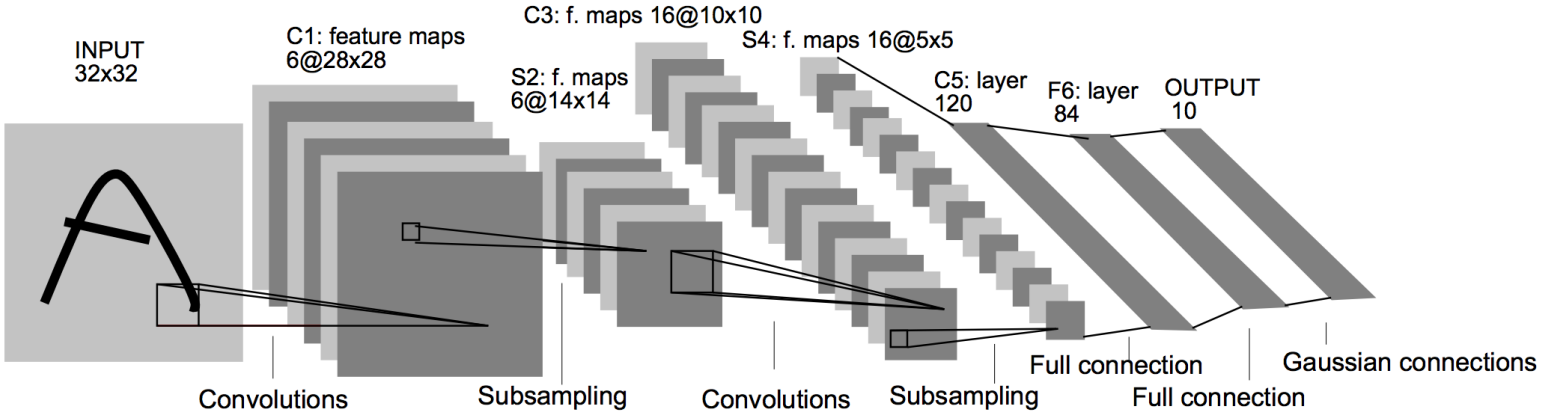


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- Input: 32x32 pixel image.
- Largest character is 20x20 (All important info should be in the center of the receptive field of the highest level feature detectors)
- Black and White pixel values are normalized: E.g. White = -0.1, Black =1.175 (Mean of pixels = 0, Std of pixels =1)

LeNet 5, Layer C1

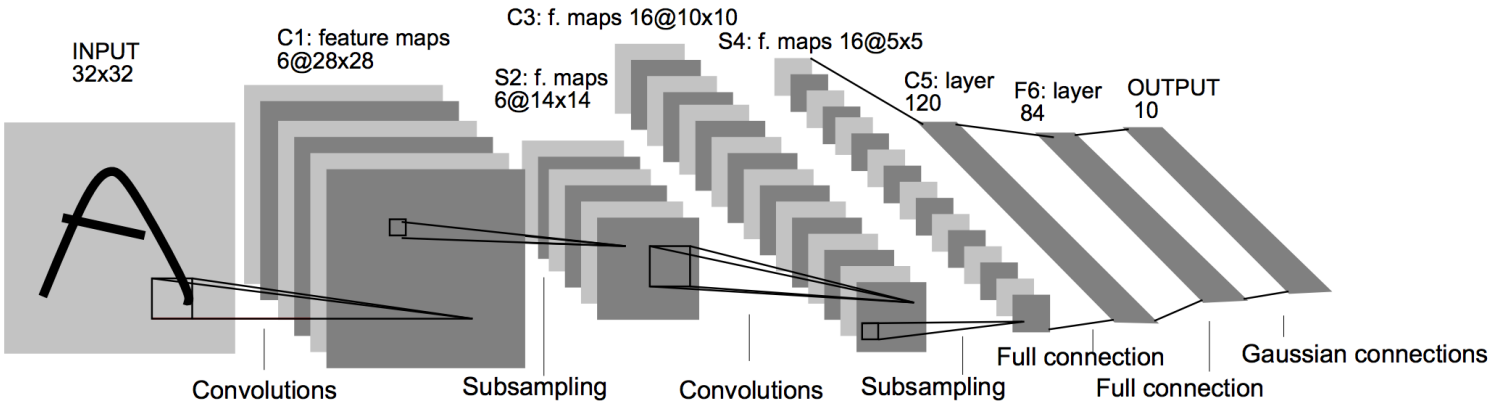


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- C1: Convolutional layer with 6 feature maps of size 28x28. $C1_k$ (k=1…6)
- Each unit of C1 has a 5x5 receptive field in the input layer.
  - Topological structure
  - Sparse connections
  - Shared weights
- (5*5+1)*6=156 parameters to learn
- Connections: 28*28*(5*5+1)*6=122304
- If it was fully connected we had (32*32+1)*(28*28)*6 parameters

LeNet 5, Layer S2

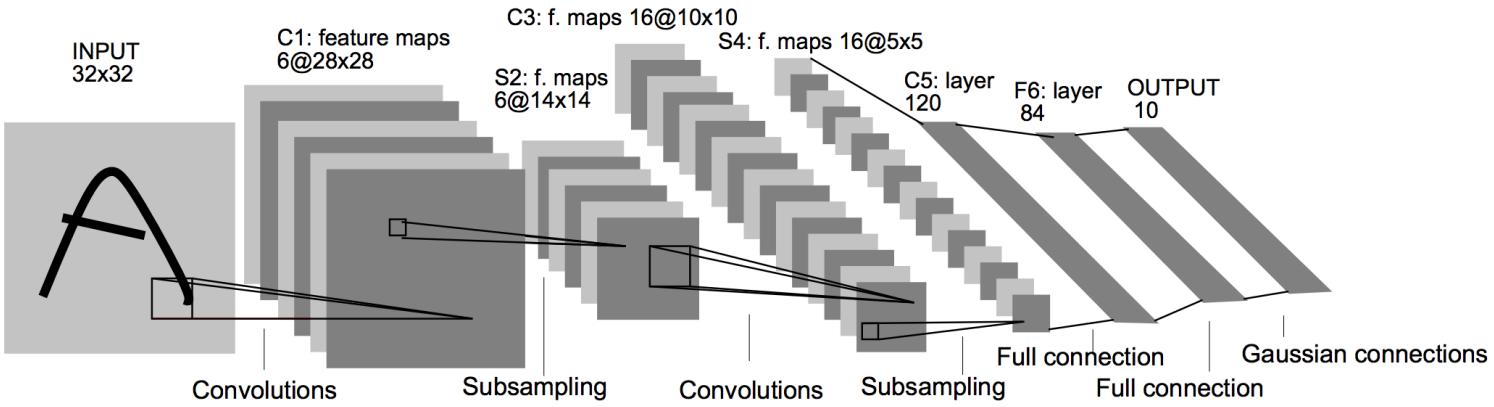


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- S2: Subsampling layer with 6 feature maps of size 14x14 2x2 non overlapping receptive fields in C1 Layer
- S2: 6*2=12 trainable parameters.
- Connections: 14*14*(2*2+1)*6=5880

LeNet 5, Layer C3

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

- C3: Convolutional layer with 16 feature maps of size 10x10
- Each unit in C3 is connected to several! 5x5 receptive fields at identical locations in S2
- Layer C3: 1516 trainable parameters. Connections: 151600

LeNet 5, Layer S4

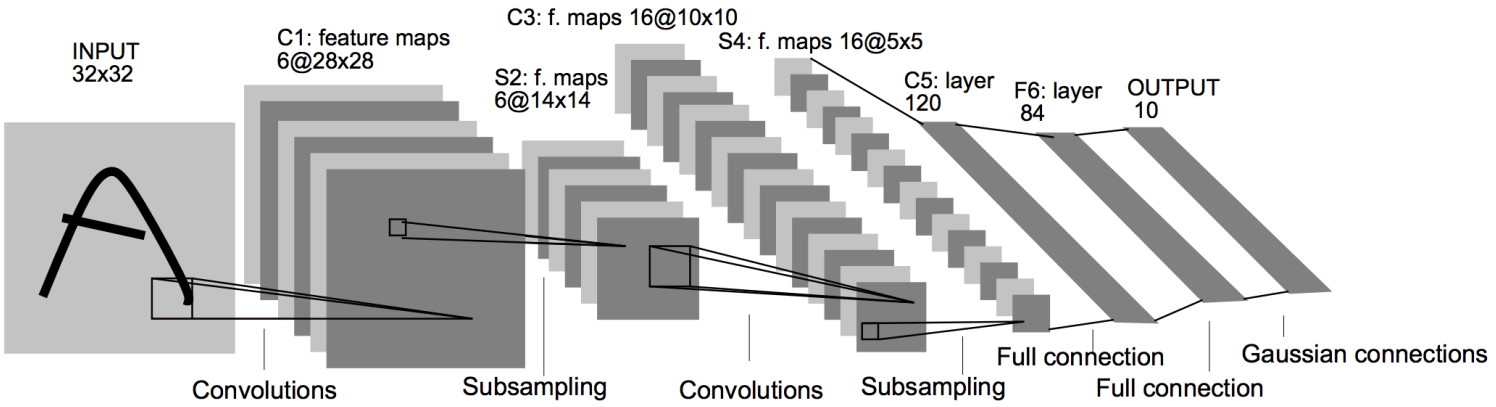


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- S4: Subsampling layer with 16 feature maps of size 5x5
- Each unit in S4 is connected to the corresponding 2x2 receptive field at C3
- Layer S4: 16*2=32 trainable parameters.
- Connections: 5*5*(2*2+1)*16=2000

LeNet 5, Layer C5

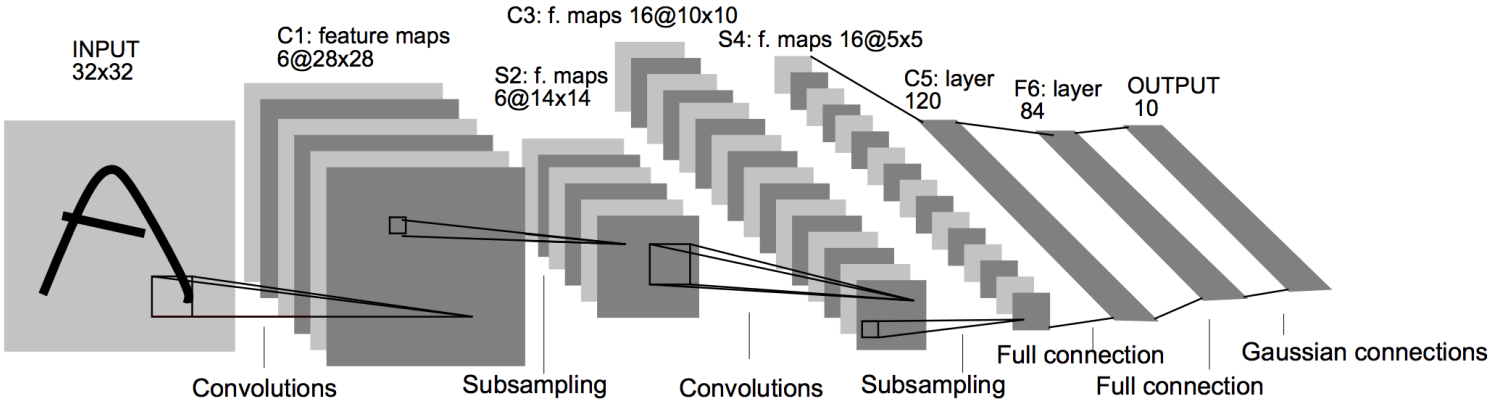


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- C5: Convolutional layer with 120 feature maps of size 1x1
- Each unit in C5 is connected to all 16 5x5 receptive fields in S4
- Layer C5: 120*(16*25+1) = 48120 trainable parameters and connections (Fully connected)

LeNet 5, Layer F6

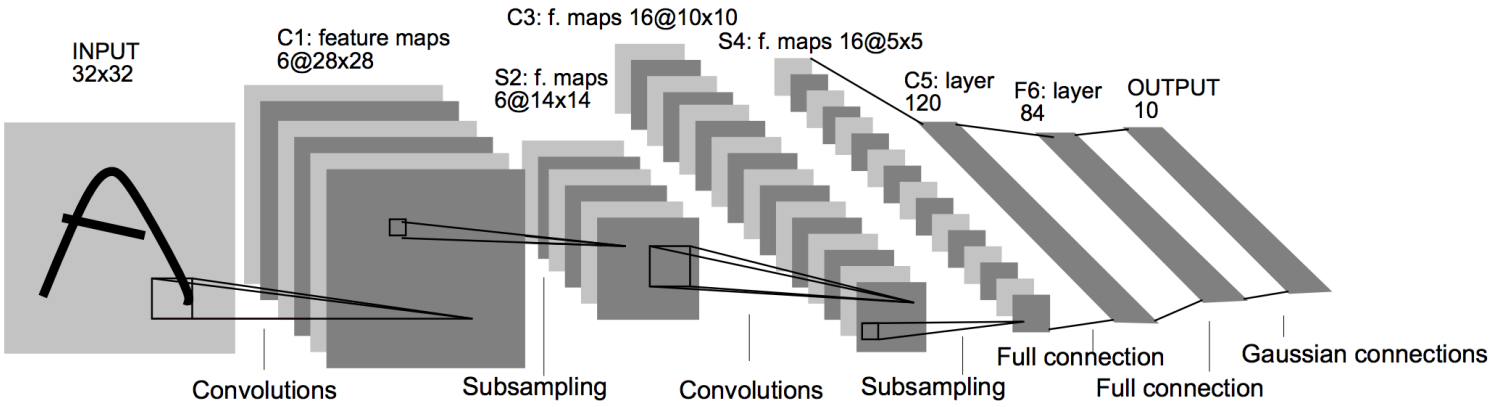


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

- Layer F6: 84 fully connected units. 84*(120+1)=10164 trainable parameters and connections.
- Output layer: 10RBF (One for each digit) 84=7x12, stylized image
- Weight update: Backpropagation

Classification Task

- The goal is to recognize objects present in an image.

Tags	Confidence
racer, race car, racing car	0.7962
sports car, sport car	0.1341
cab, hack, taxi, taxicab	0.0278
convertible	0.0165
car wheel	0.0080

ImageNet

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

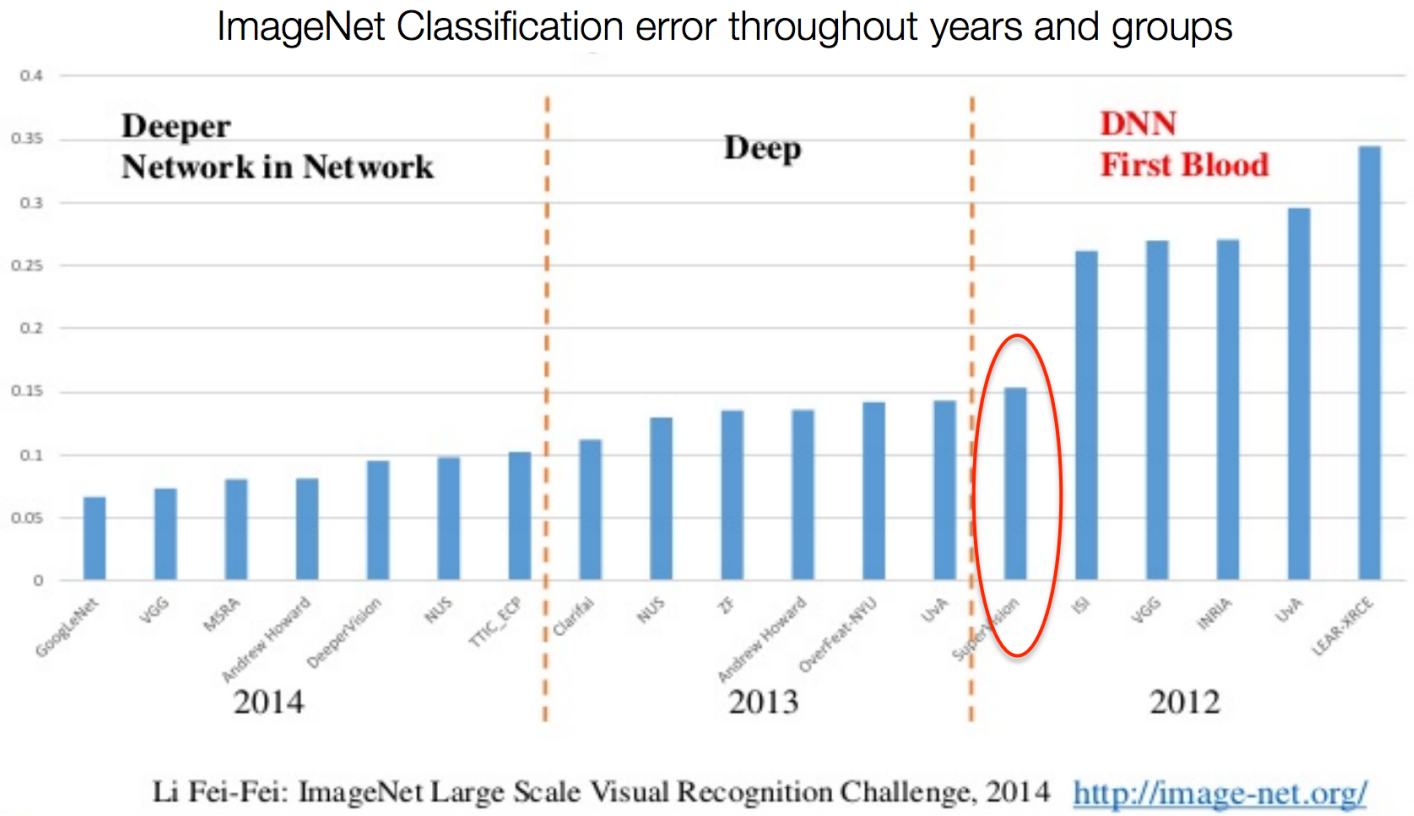
http://image-net.org

Picture Credits: Andrej Karpathy

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- Annual competition of image classification at large scale.
- 1.2M training images in 1K categories.
- 50K validation images, 150K testing images.
- Classification: make 1 (Top-1 error) /5 (Top-5 error) guesses about the image. label.

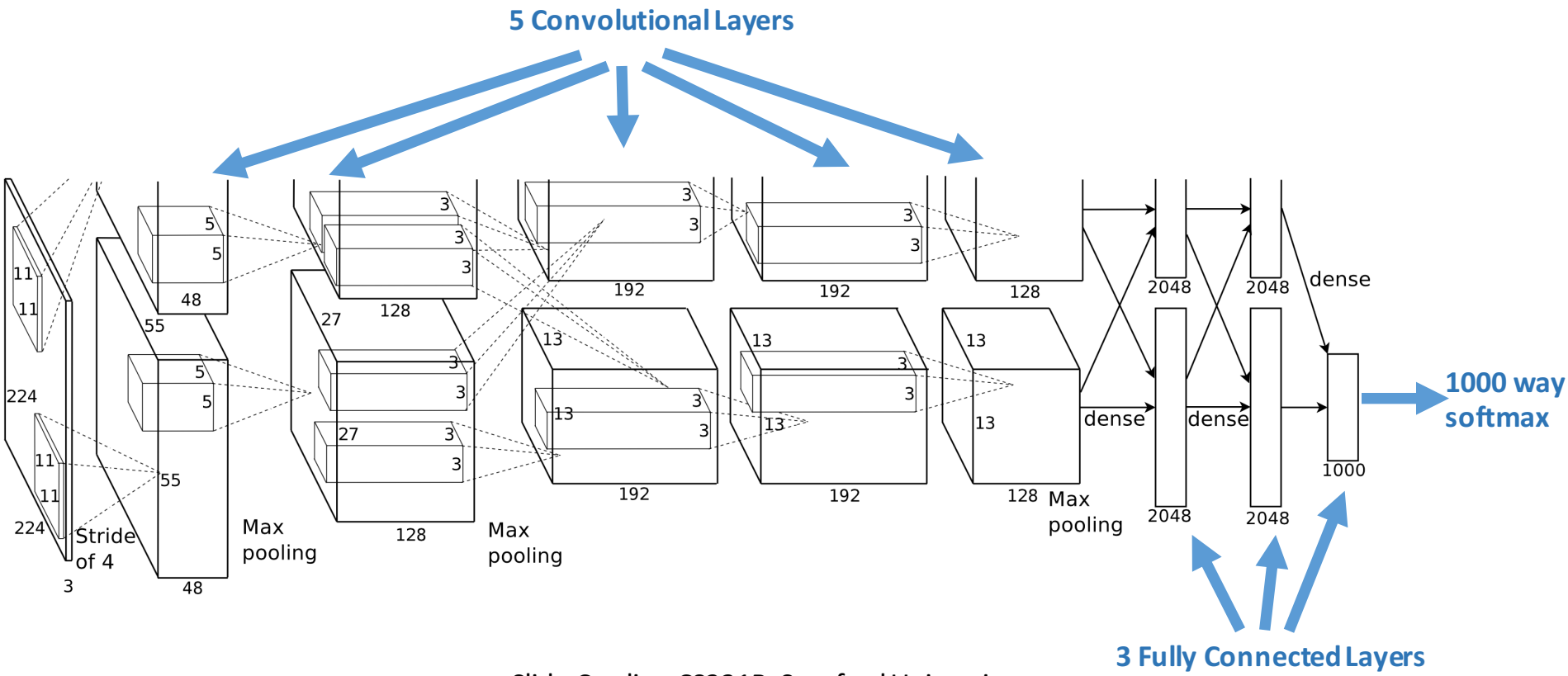
ILSVRC


ImageNet Classification error throughout years and groups
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Deeper
Network in Network
Deep
DNN
First Blood
GoogLeNet
VGG
MSRA
Andrew Howard
DeeperVision
NUS
TTIC_ECP
Clarifai
NUS
ZF
Andrew Howard
OverFeat-NYU
UvA
SuperVision
ISI
VGG
INRIA
UvA
LEAR-XRCE
2014
2013
2012
Li Fei-Fei: ImageNet Large Scale Visual Recognition Challenge, 2014
http://image-net.org/

AlexNet (Supervision)

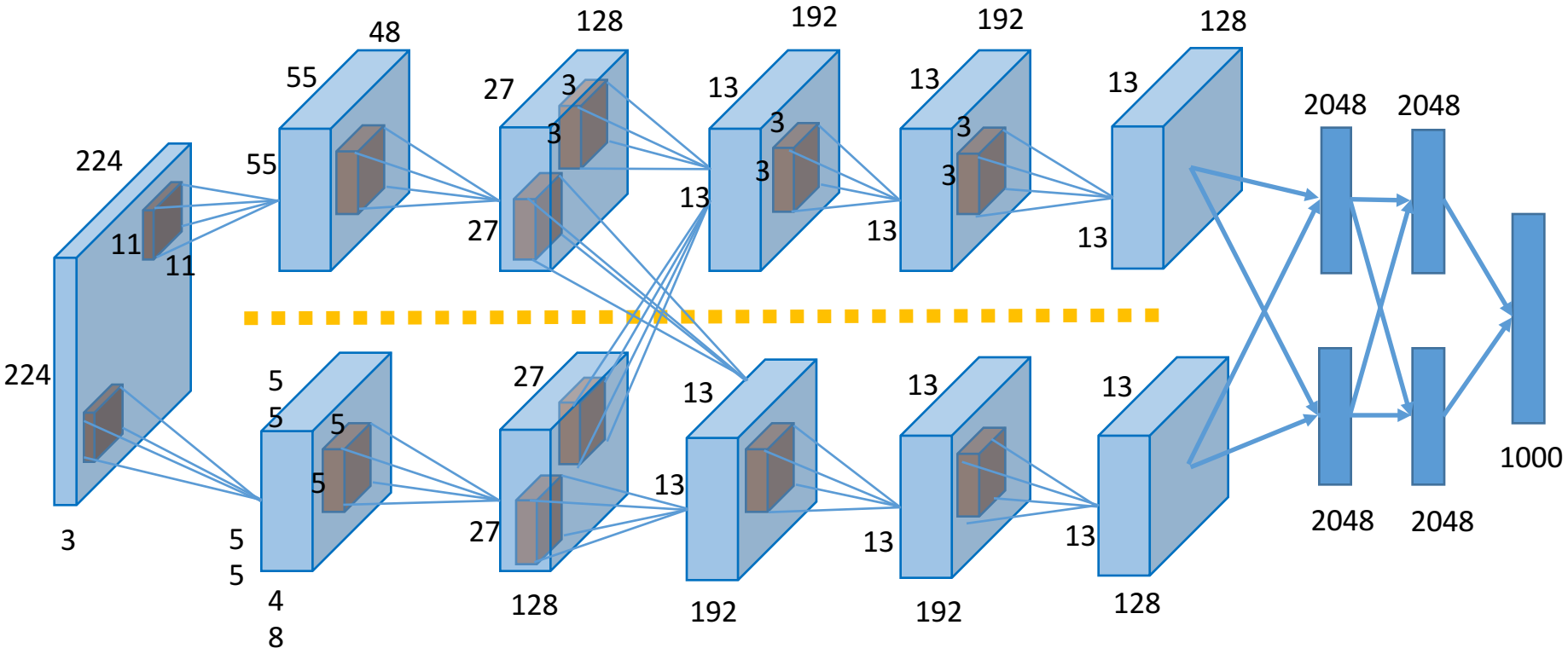
- Similar framework to LeCun'98 but,
- Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
- More data (10 6 vs. 10 3 images)
- GPU implementation (50x speedup over CPU)
- Trained on two GPUs for a week
- Better regularization for training (DropOut)

Architecture – Overview



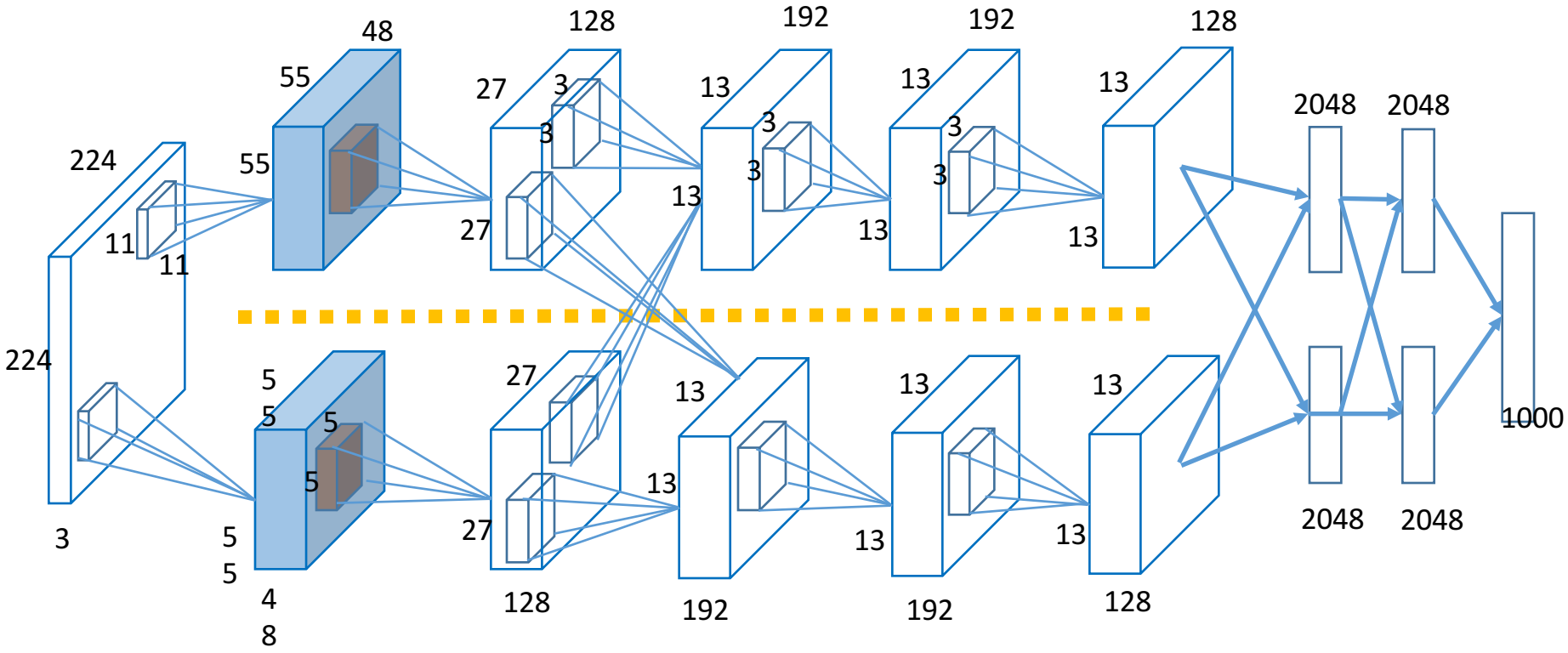
Slide Credits: CS231B, Stanford University

Architecture - Overview

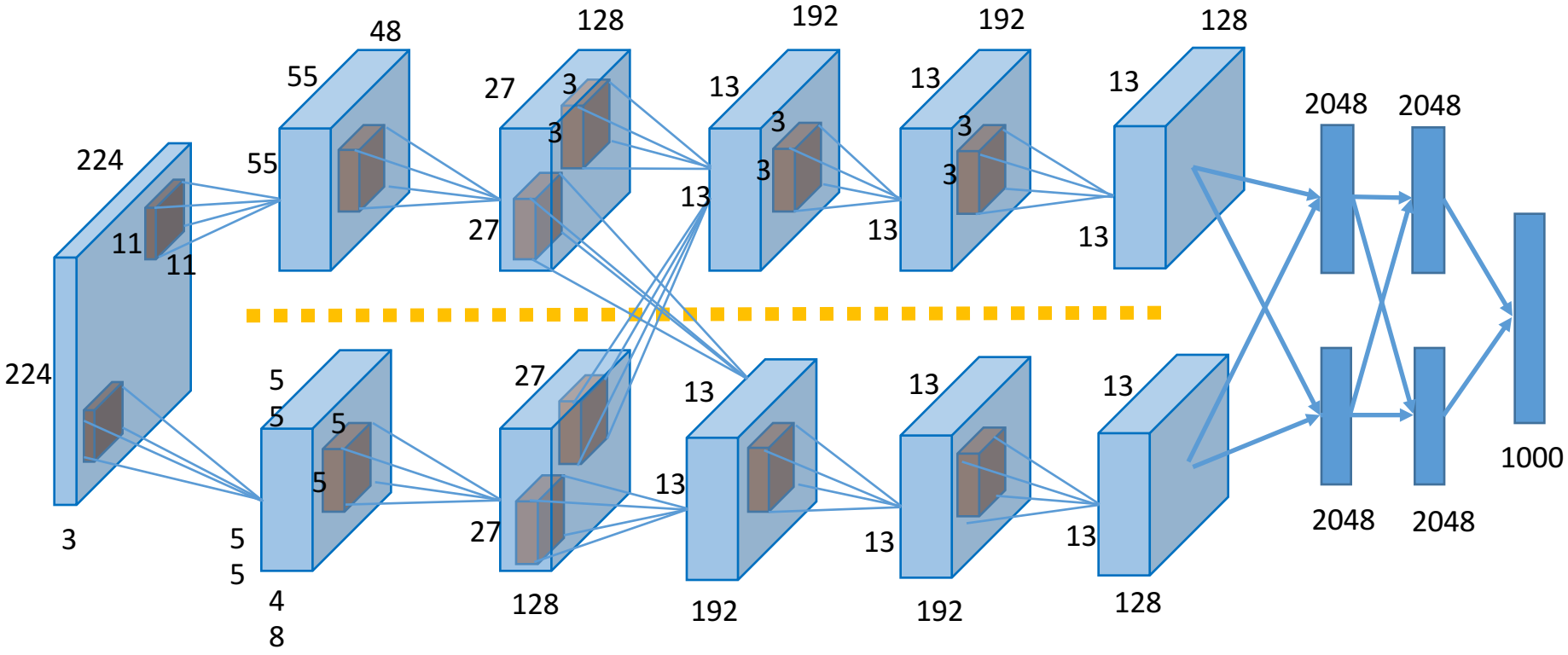


Architecture - Overview

- 55*55*96 = 290,400 neurons, each having 11*11*3=363 weights + 1 bias
- 290400 * 364 = 105,705,600 parameters in first layer alone.
- Total 60M real-valued parameters and 650,000 neurons

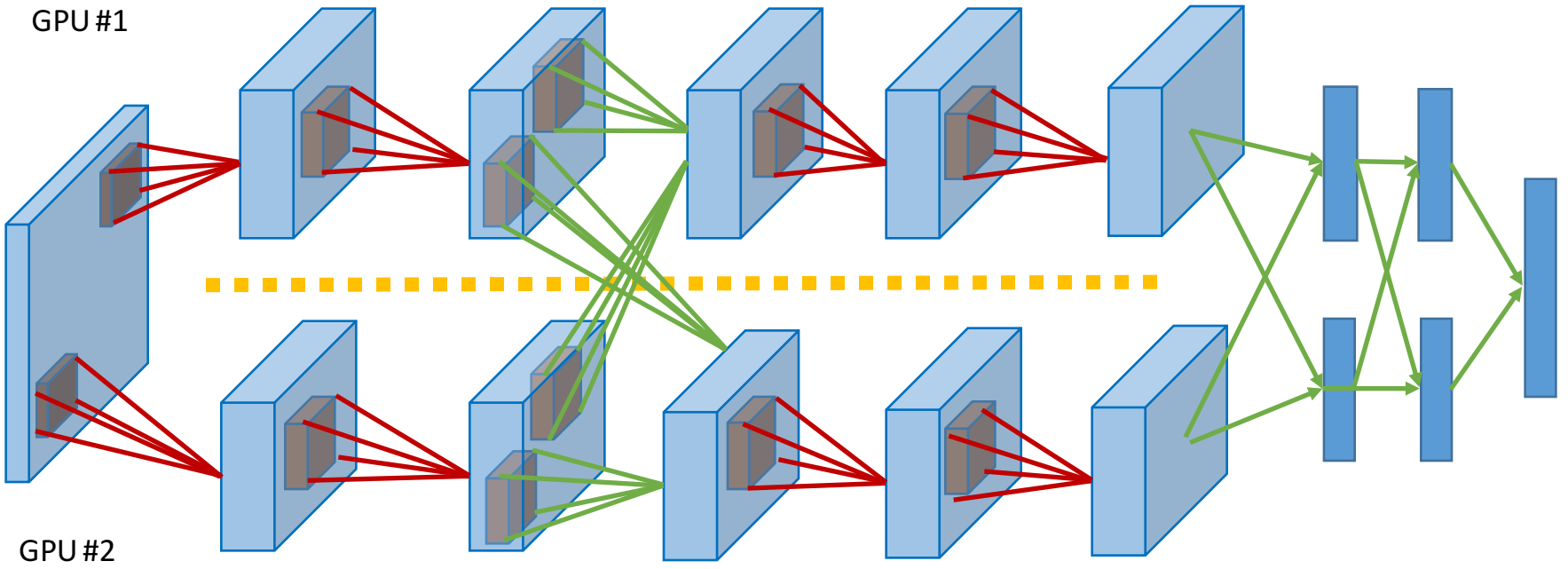


Architecture - Overview

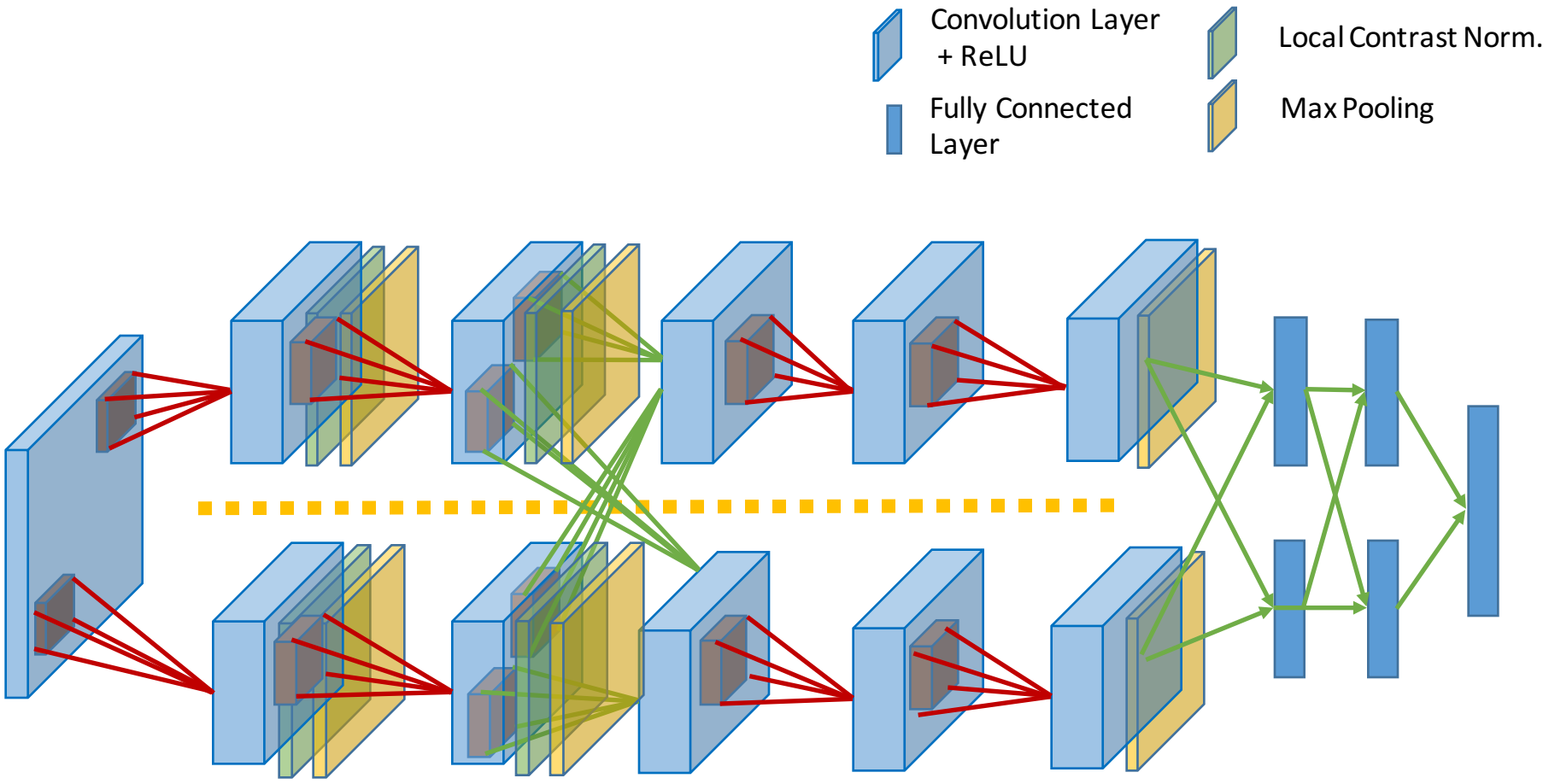


Architecture - Overview

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

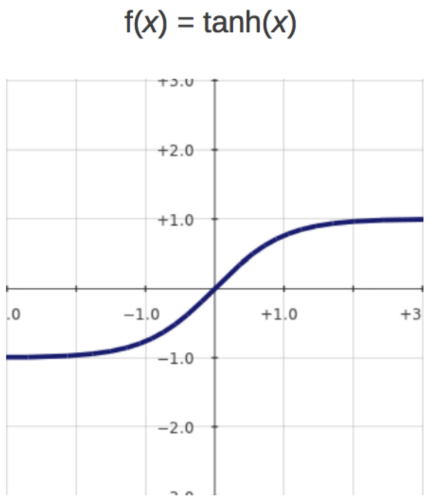


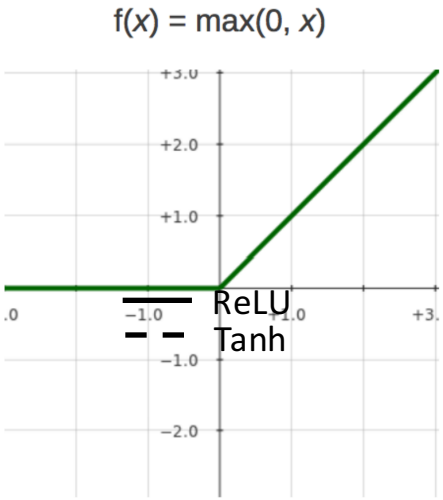
Architecture - Overview

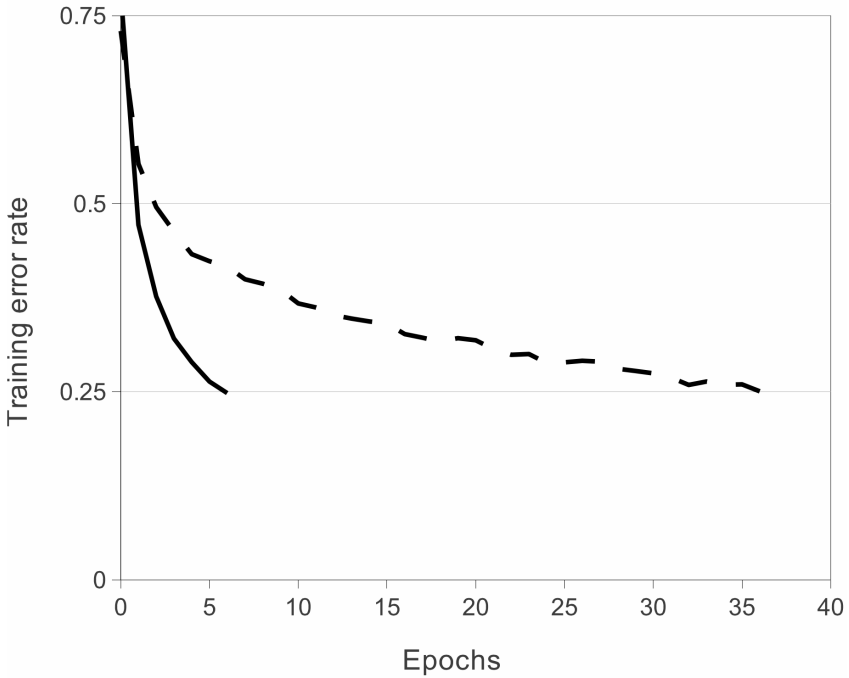


ReLU Nonlinearity

- Standard way to model a neuron
  - $f(x) = \tanh(x)$
  - $f(x) = (1 + e^{-x})^{-1}$
  - Very slow to train.
- Non-saturating nonlinerity: Rectified Linear Units (ReLU)
  - $f(x) = \max(0, x)$
  - Quick to train.







With a four layer CNN, ReLU reaches 25% error rate six times faster than Tanh on CIFAR-10

Local Response Normalization

- ReLUs don't need input normalization.
- Following normalization scheme helps generalization

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

Response normalized activity

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

k, n, α, β are hyper-parameters which are determined using validation set.

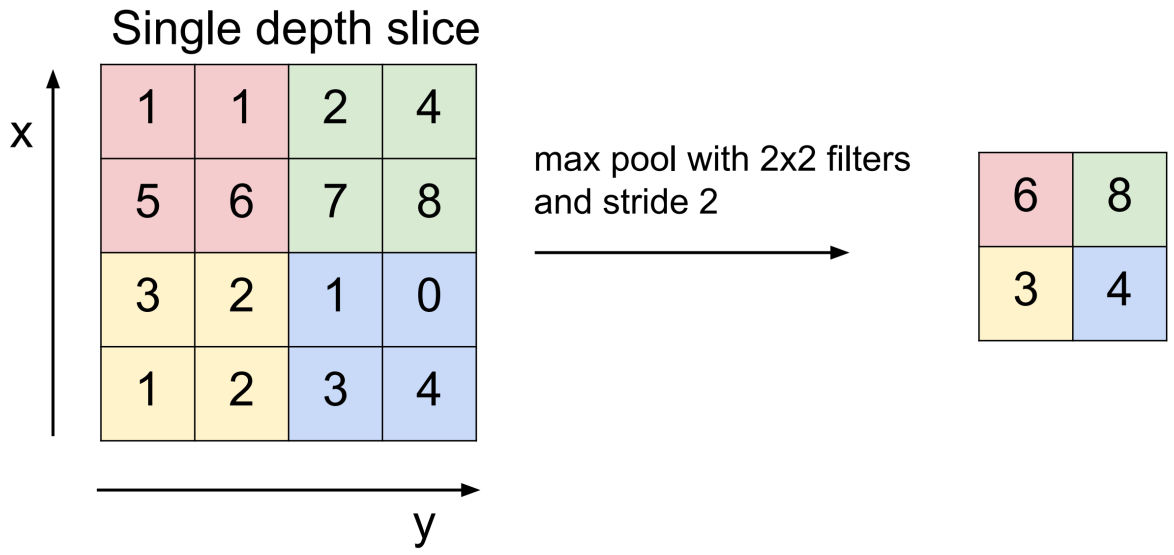
The paper had: k=2, n=5, $\alpha=10^{-4}$, $\beta=-.75$

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

Slide Credits: CS231B, Stanford University

Max Pooling

- Convenience Layer: Makes the representation smaller and more manageable without loosing too much information.
- Input Volume of size [W1 * H1 * D1], receptive fields F*F, and stride S
- Output Volume [W2 * H2 * D1]
- W2 = (W1 – F) /S + 1, H2 = (H1 – F) /S +1

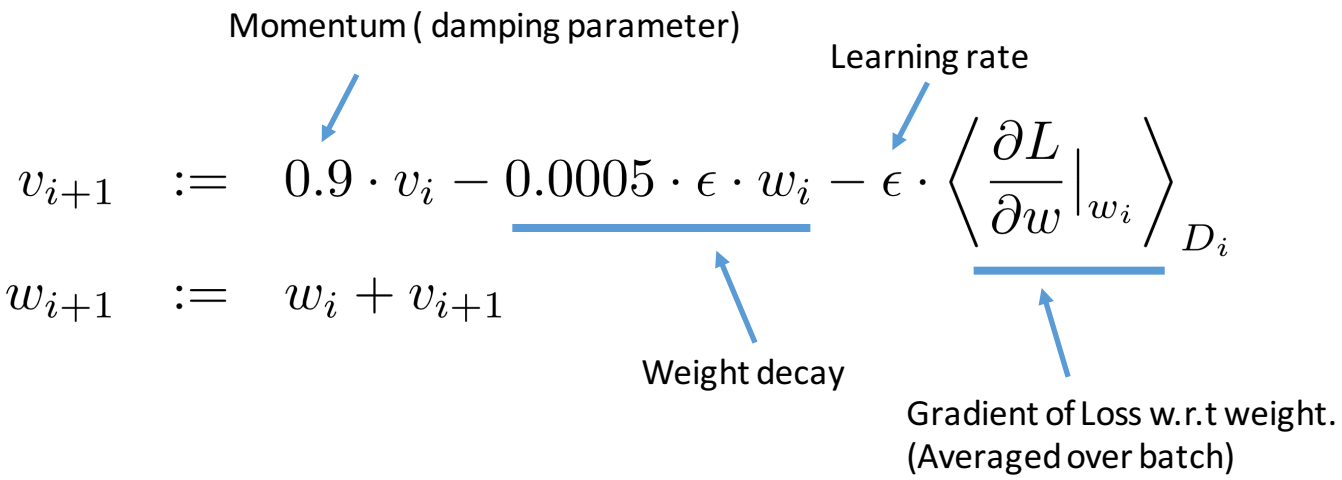


Overlapping Pooling

- If we have set stride less than f (field) then we obtain overlapping pooling.
- Specifically in AlexNet: s=2; z=3
- Reduces the top-1 and top-5 error rates by 0.4% and 0.3% respectively.

Stochastic Gradient Descent Learning

- Batch Size: 128
- The training took 5 to 6 days on two NVIDIA GTX 580 3GB GPUs



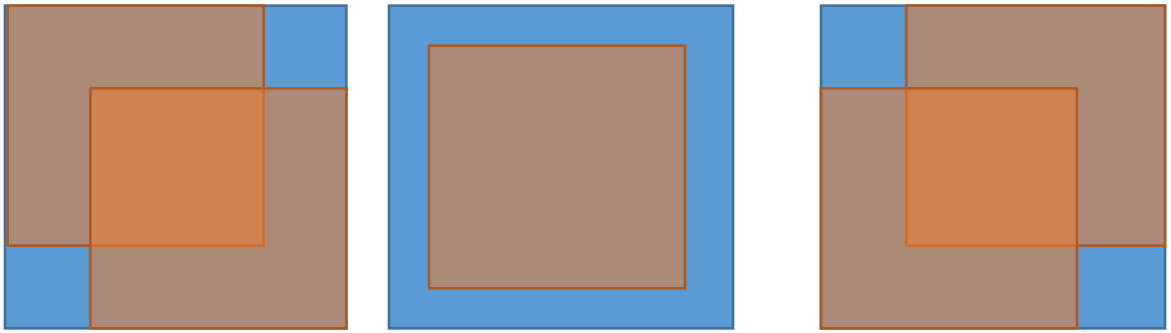
Data Augmentation

- Easiest way to reduce overfitting on image data is to artificially enlarge the dataset using label-preserving transformations
- Two forms of data augmentation:
  - Image Translation and Horizontal Reflection.
  - Changing RGB intensities.

Data Augmentation: Type #1

- Image translation by randomly extracting 224 x 224 patches from the 256 X 256 images.
- Horizontal reflections of these 224 x 224 patches.
- Dataset increased by factor of 2048 though the resulting training examples are highly inter-dependent.

- At test time: 5 224 x 224 patches (four corners and one center patch) and their horizontal patches are used.
- Averaging the predictions made by the softmax layer.



Slide Credits: CS231B, Stanford University

Data Augmentation: Type #2

- Changing RGB intensities:
  - Perform PCA on the set of RGB values throughout the training set.

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

  - Add multiples of principle components

$$[p_1, p_2, p_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$

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

$\mathbf{P}_i$ and $\boldsymbol{\lambda}_i$ are the ith eigenvector and eigenvalue of the 3x3 covariance matrix.

$\boldsymbol{\alpha}_i$ is the random variable.

Slide Credits: CS231B, Stanford University

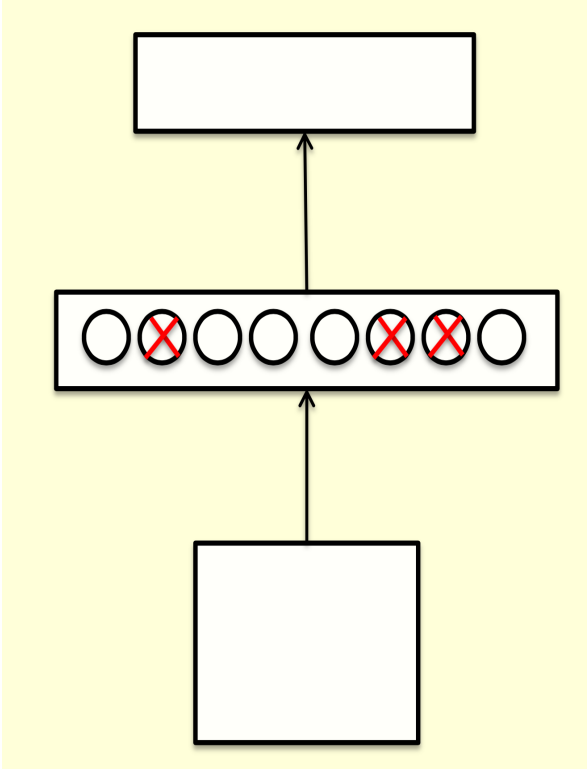
Averaging big deep neural nets is hard

- Each net takes a long time to learn
- At test time, we don't want to run lots of different large neural nets.
- Everyone who wins competitions does it by averaging/boosting models.
  - Random Forest
  - AdaBoost

Credits: Geoffrey E. Hinton, NIPS 2012

Dropouts : An efficient way to average many large neural nets

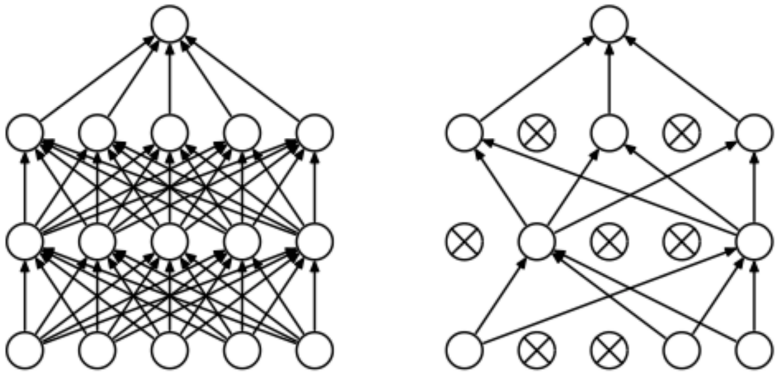
- Consider a neural net with one hidden layer.
- Each time we present a training example, we randomly omit each hidden unit with probability 0.5.
- Equivalent to randomly sampling from 2^h different units.
- All architectures share weights.



Credits: Geoffrey E. Hinton, NIPS 2012

Dropouts as form of model averaging

- We sample from 2^h models i.e. only a few of the models ever get trained. They only get one training example
- Due to sharing of weights, the model is strongly regularized.
  - Pulls the weights towards what other models want.
  - Better than L2 and L1 that pull weights towards zero.



Standard Neural Net

After applying dropout.

Credits: Geoffrey E. Hinton, NIPS 2012

Figure Credit: Srivastava et al.

What do we do at test time?

- We could sample many different architectures and take the geometric mean of their output distributions.
- Faster way would be to use all the hidden units but after halving their outgoing weights.
  - In case of single hidden layer , this is equivalent to the geometric mean of the predictions of all models.
  - For multiple layers, it's a pretty good approximation and its fast.

Credits: Geoffrey E. Hinton, NIPS 2012

How well does dropout work?

- If your deep neural net is significantly overfitting, it will reduce the number of errors by a lot.
- If not overfitting, use a bigger one
  - # of parameters >> # training examples.
  - Synapses are cheaper than experiences.

Credits: Geoffrey E. Hinton, NIPS 2012

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.

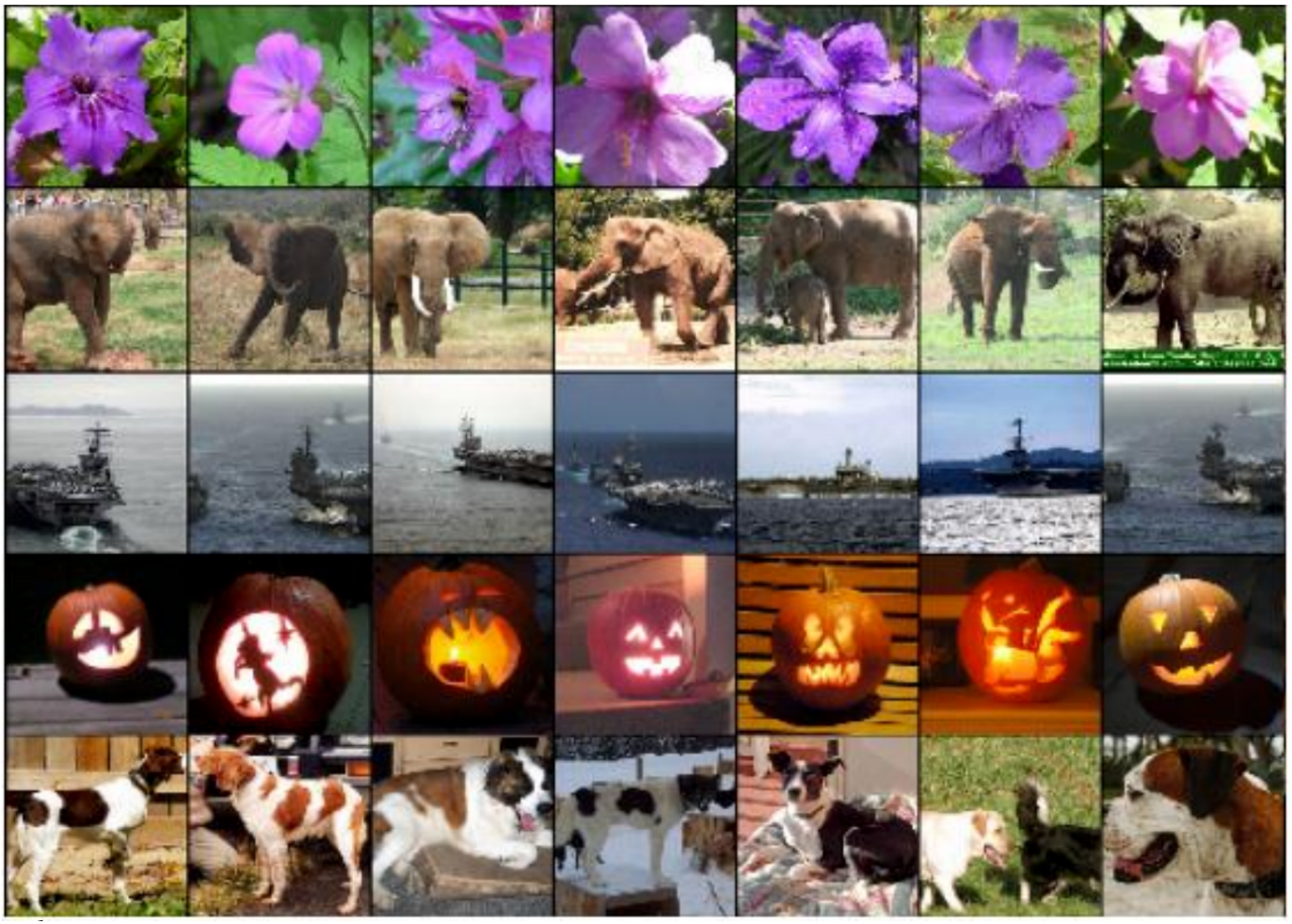
Results: ILSVRC - 2012

- 1CNN*: Training 1 CNN, with an extra sixth convolutional layer over the last pooling layer to classify the entire ImageNet Fall 2011 release and then fine-tuning it on ILSVRC-2012.
- 7CNN*: Averaging 5CNN + two CNNs pretrained on Fall 2011 release.

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%**

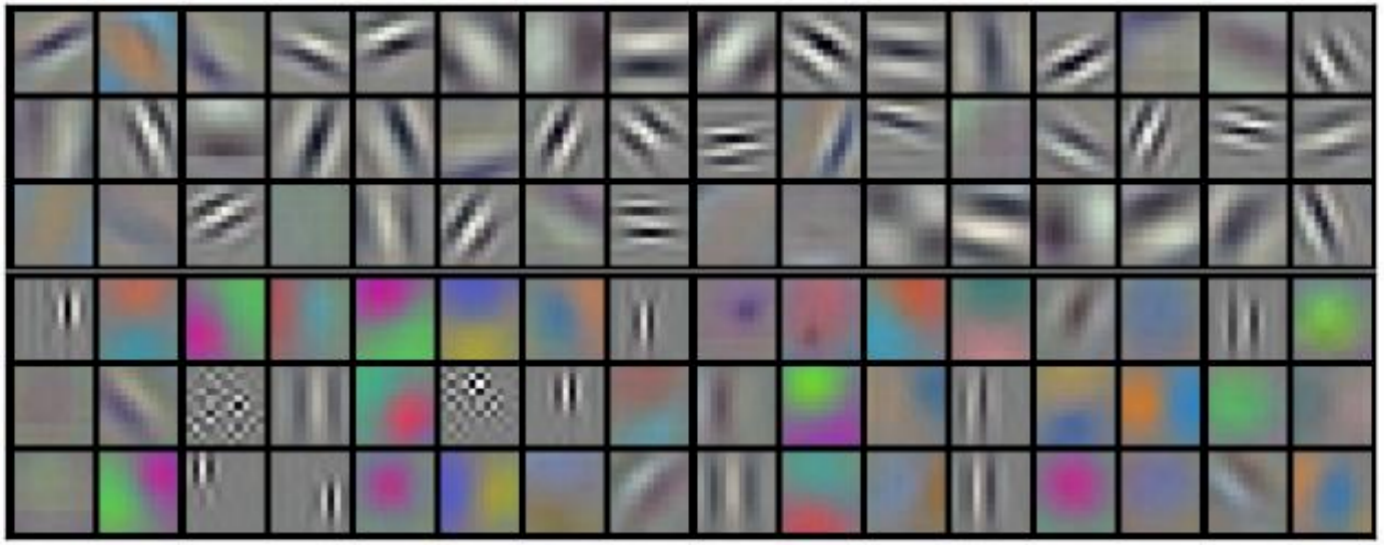
Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

Result: Image Similarity



- 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.

First Convolutional Layer



- 96 convolutional kernels of size 11×11×3 learned by the first convolutional layer on the 224×224×3 input images.
- The top 48 kernels were learned on GPU1 while the bottom 48 kernels were learned on GPU2

Scene Recognition – Another hallmark task for computer Vision

- Given an image, predict which place we are in


Bedroom


Harbor

Places205 – Scene Centric Database

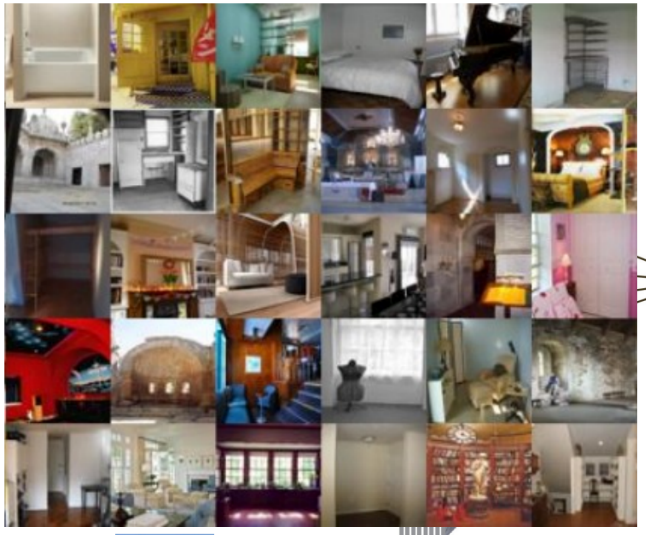
- To learn deep features for scene recognition, we require a scene-centric database as big as ImageNet.
- Places205 contains 205 categories, 2,448,873 images
- Each category contains atleast 5000 images.

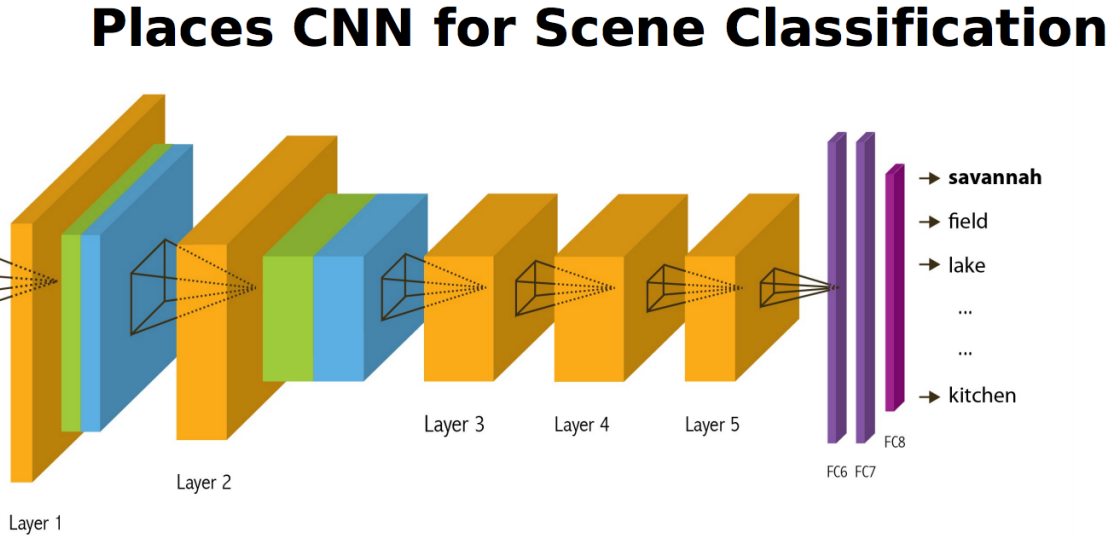
Past Datasets:

- SUN397: 397 scene categories, atleast 100 images each, total 108,754 images.
- Indoor67: 67 categories, 15620 images.

Training CNN for Scene Recognition

- Training Set: Places205, 2.5M images for 205 categories
- Validation Set: 100 images for every category. 20,500 total.
- Test Set: 200 images for every category. 41,000 total.
- Places CNN: Similar to Caffe Reference Net.
- Took about 6 days to finish 30,000 iterations on a single Tesla K40.





Places

Experiments and Results

Table 1: Classification accuracy on the test set of Places 205 and the test set of SUN 205.

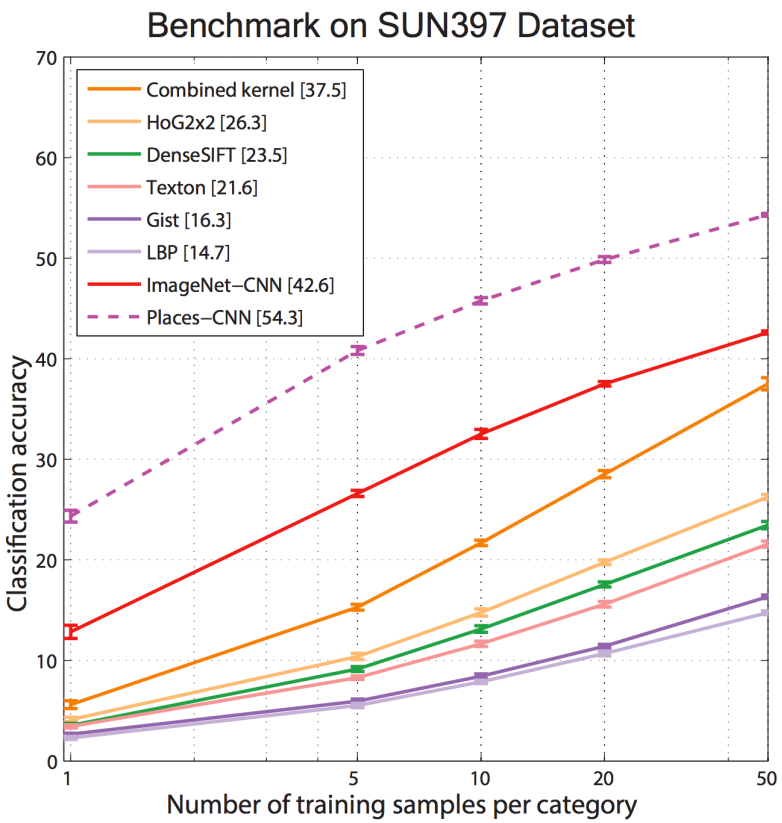
	Places 205	SUN 205
Places-CNN	**50.0%**	**66.2%**
ImageNet CNN feature+SVM	40.8%	49.6%

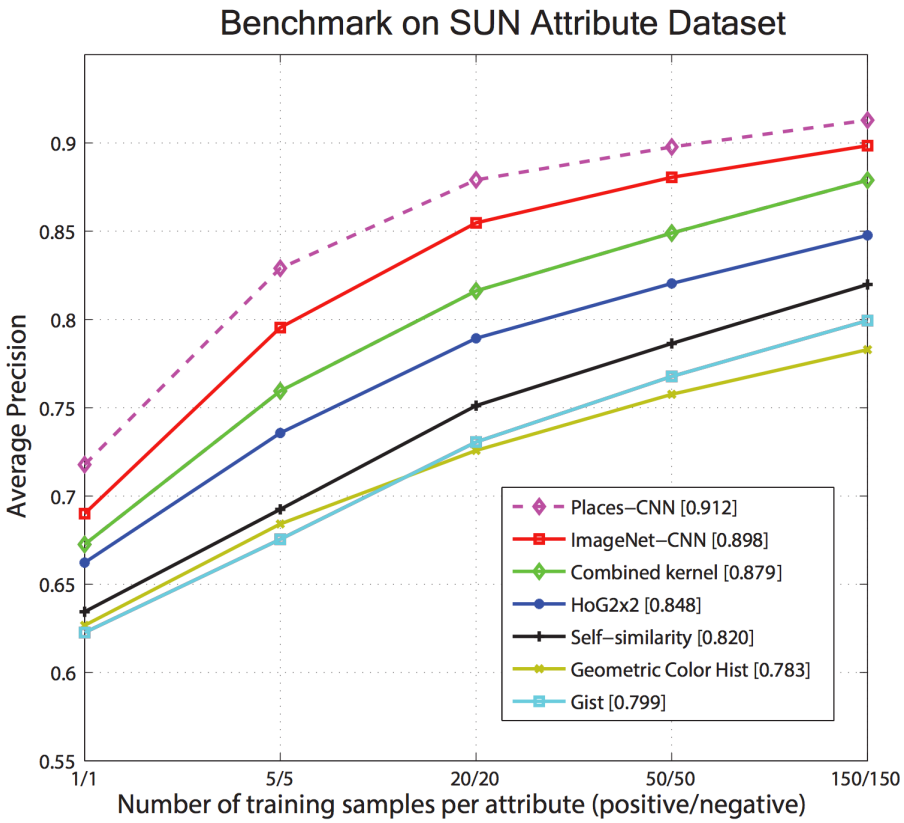
- Top-5 error for Places 205 and SUN 205 test set is 18.9% and 8.1% respectively

Table 2: Classification accuracy/precision on scene-centric databases and object-centric databases for the Places-CNN feature and ImageNet-CNN feature. The classifier in all the experiments is a linear SVM with the same parameters for the two features.

	SUN397	MIT Indoor67	Scene15	SUN Attribute
Places-CNN feature	**54.32±0.14**	**68.24**	**90.19±0.34**	**91.29**
ImageNet-CNN feature	42.61±0.16	56.79	84.23±0.37	89.85
	Caltech101	Caltech256	Action40	Event8
Places-CNN feature	65.18±0.88	45.59±0.31	42.86±0.25	94.12±0.99
ImageNet-CNN feature	**87.22±0.92**	**67.23±0.27**	**54.92±0.33**	**94.42±0.76**

Experiments and Results





Object Detectors emerge in CNN trained for Scenes

This paper shows that object detectors emerge inside a CNN trained for scene classification, without any object supervision

Object detectors for free!

Uncovering the CNN representation

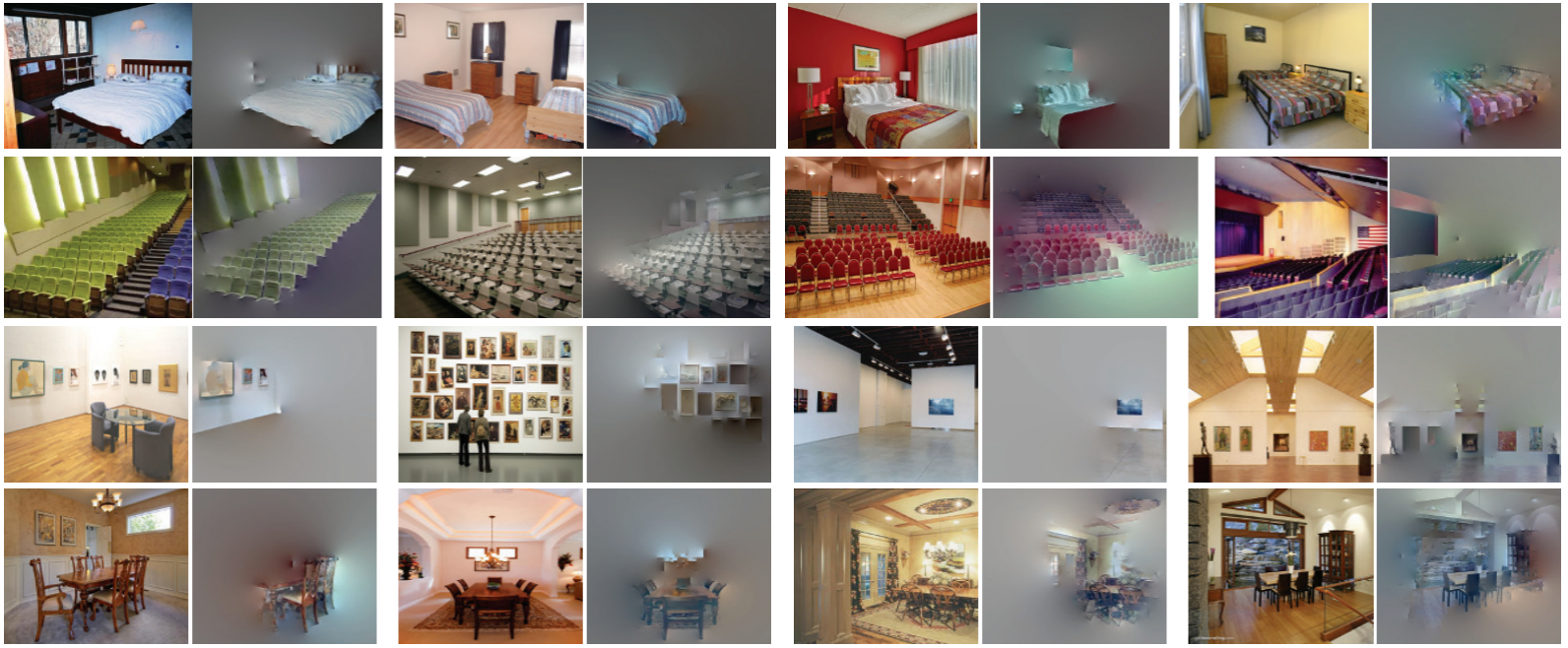
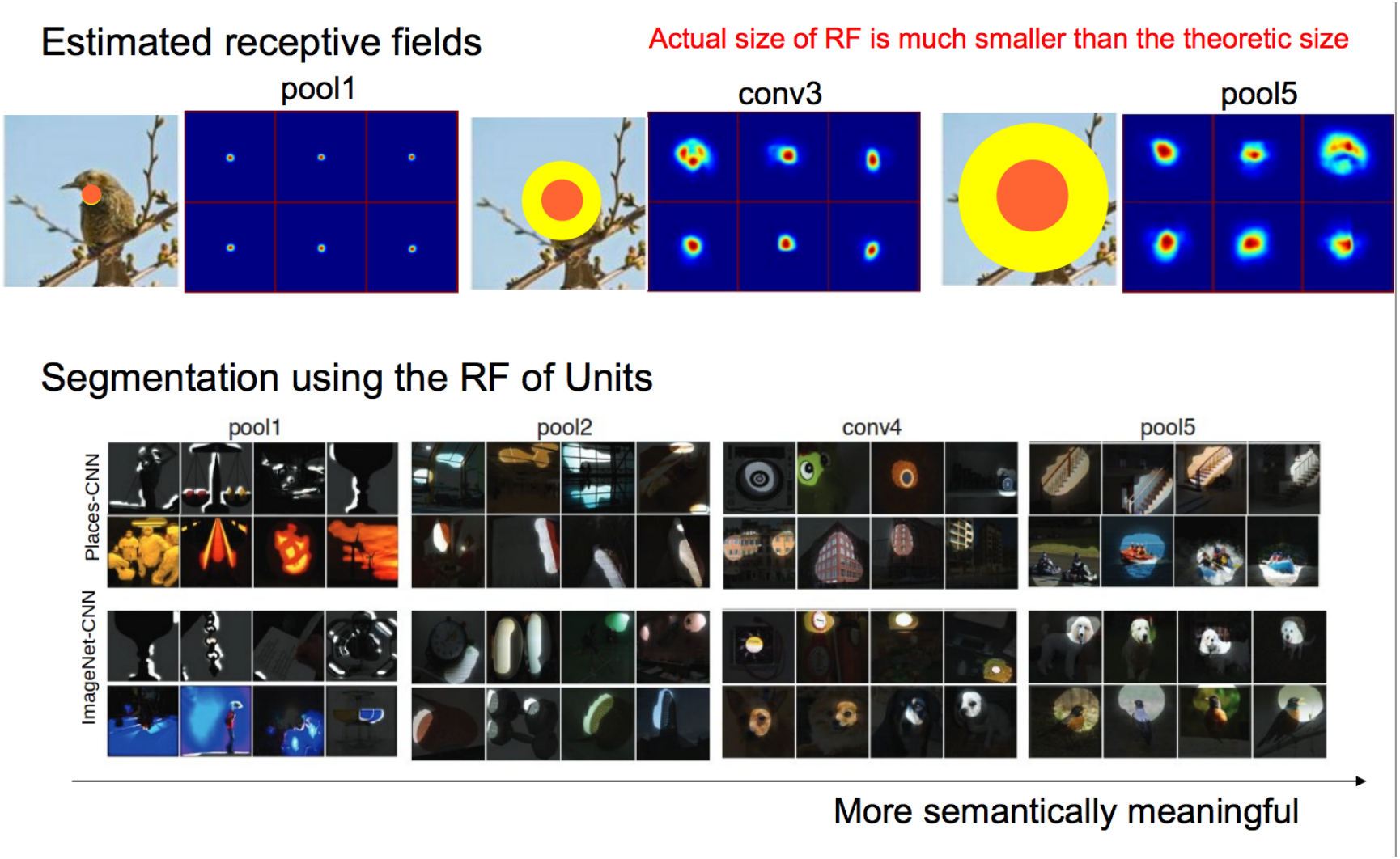


Figure 2: Each pair of images shows the original image (left) and a simplified image (right) that gets classified by the Places-CNN as the same scene category as the original image. From top to bottom, the four rows show different scene categories: bedroom, auditorium, art gallery, and dining room.

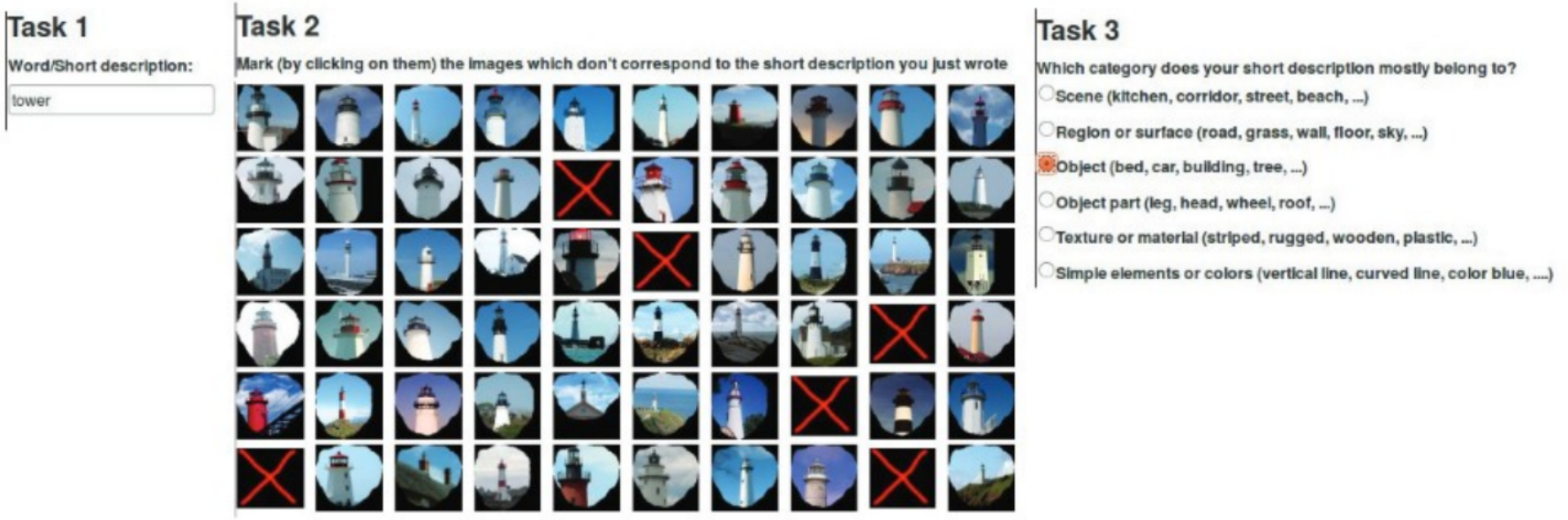
- Remove segments iteratively, until misclassification.
- Some objects are crucial for recognizing scenes.

Estimating the receptive field



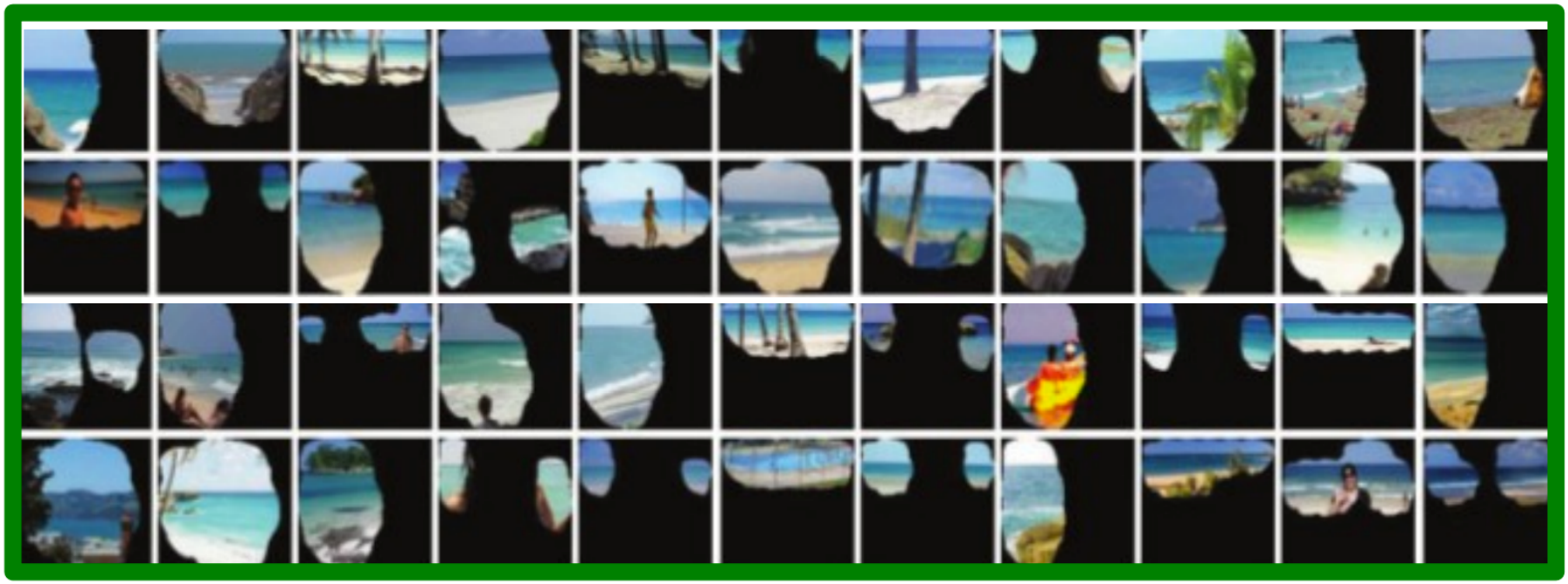
Annotating the semantics of the units.

Top ranked segmented images are cropped and sent to Amazon Turk for annotation.



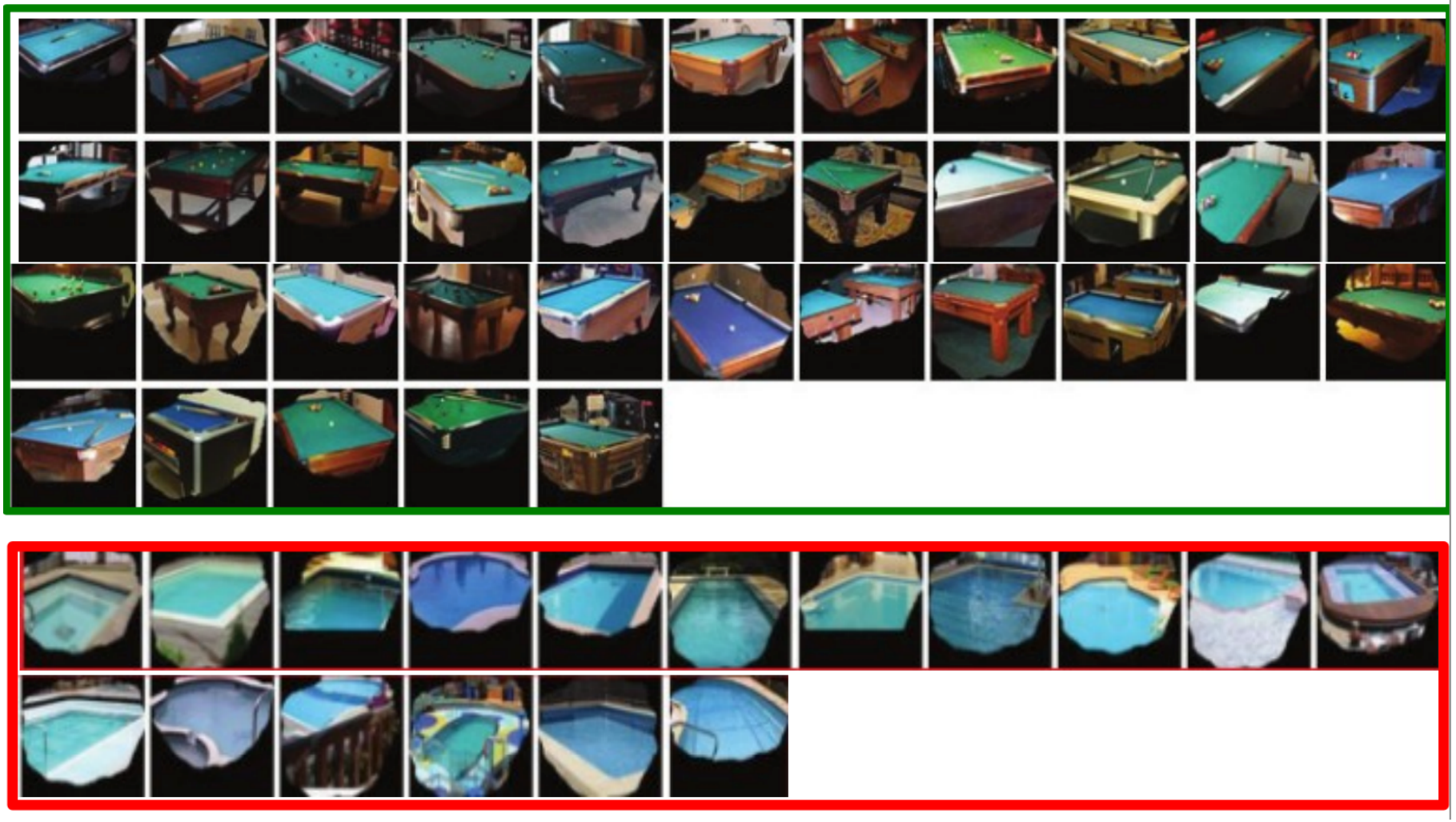
Annotating the semantics of unit.

Pool5, unit 76; Label: ocean; Type: scene; Precision: 93%



Annotating the semantics of unit.

Pool5, unit 112; Label: pool table; Type: object; Precision: 70%



Annotating the semantics of unit.

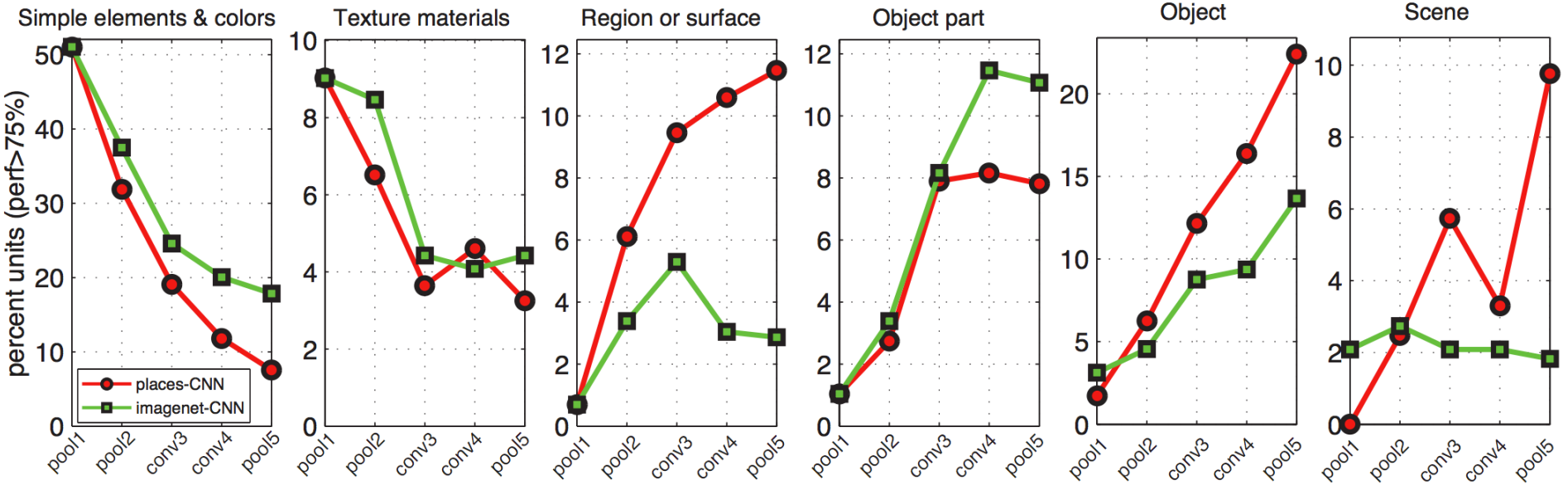
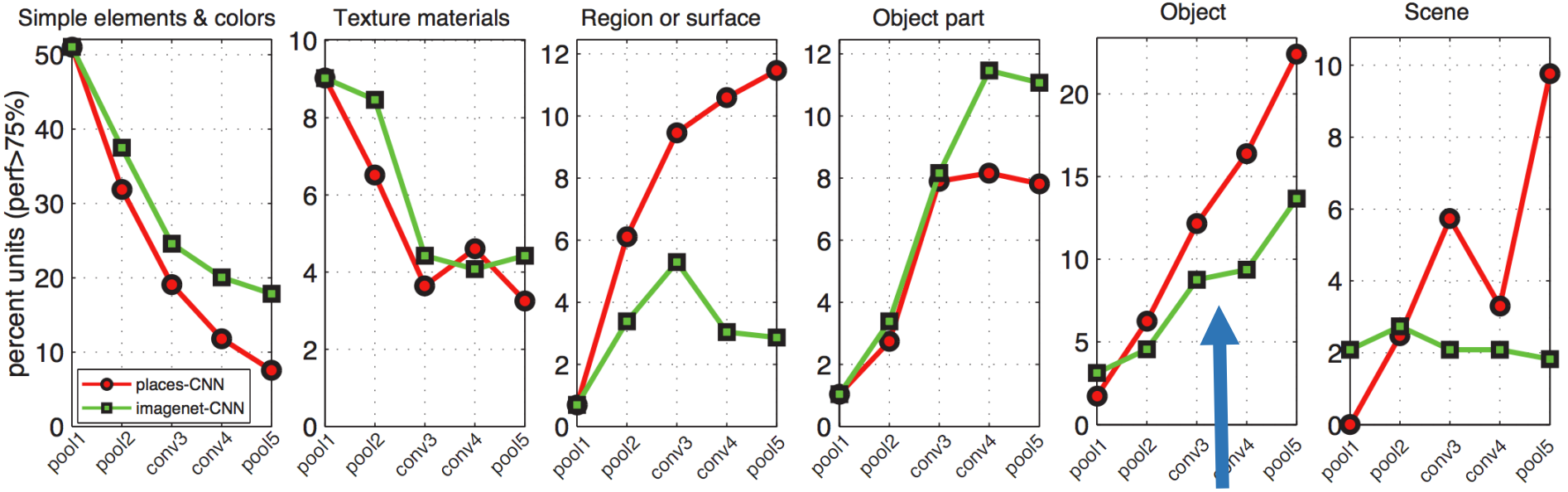


Figure 9: Distribution of semantic types found for all the units in both networks. From left to right, each plot corresponds to the distribution of units in each layer assigned to simple elements or colors, textures or materials, regions or surfaces, object parts, objects, and scenes. The vertical axis is the percentage of units with each layer assigned to each type of concept.

Annotating the semantics of unit.

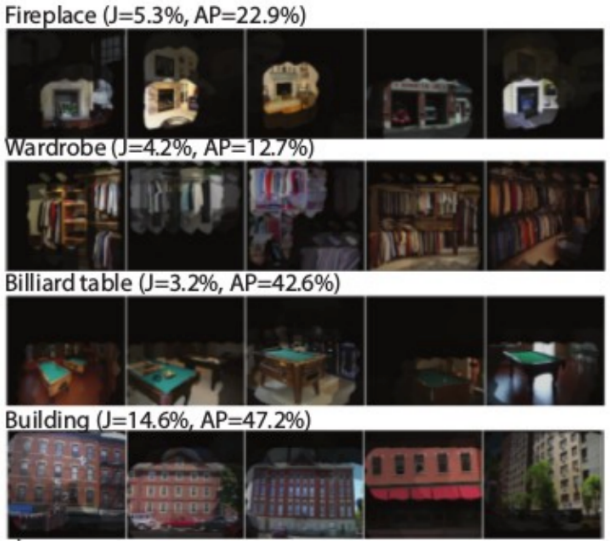


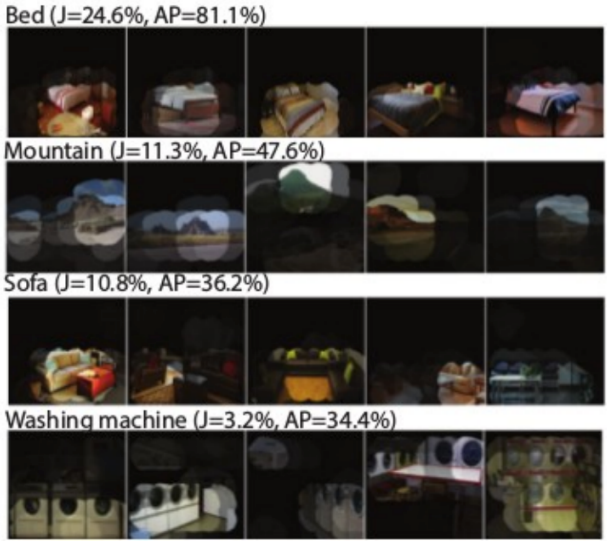
Object Detectors appear without any object supervision!

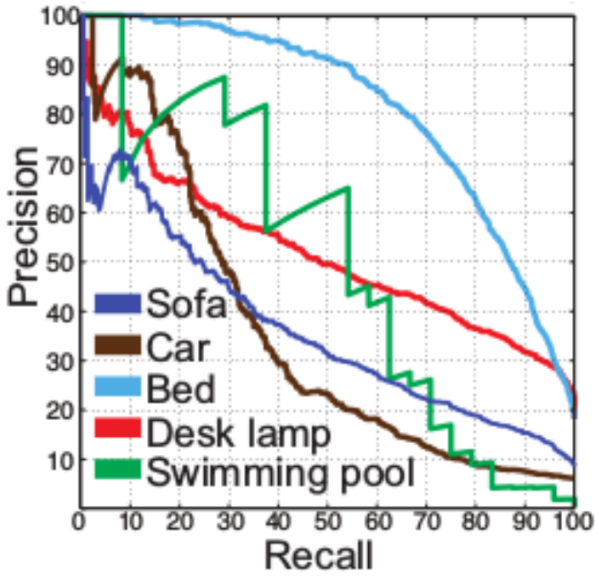
Fig ight, eac lors, textures or materials, regions or surfaces, object parts, objects, and scenes. The vertical axis is the percentage of units with each layer assigned to each type of concept.

Evaluation on the SUN Database

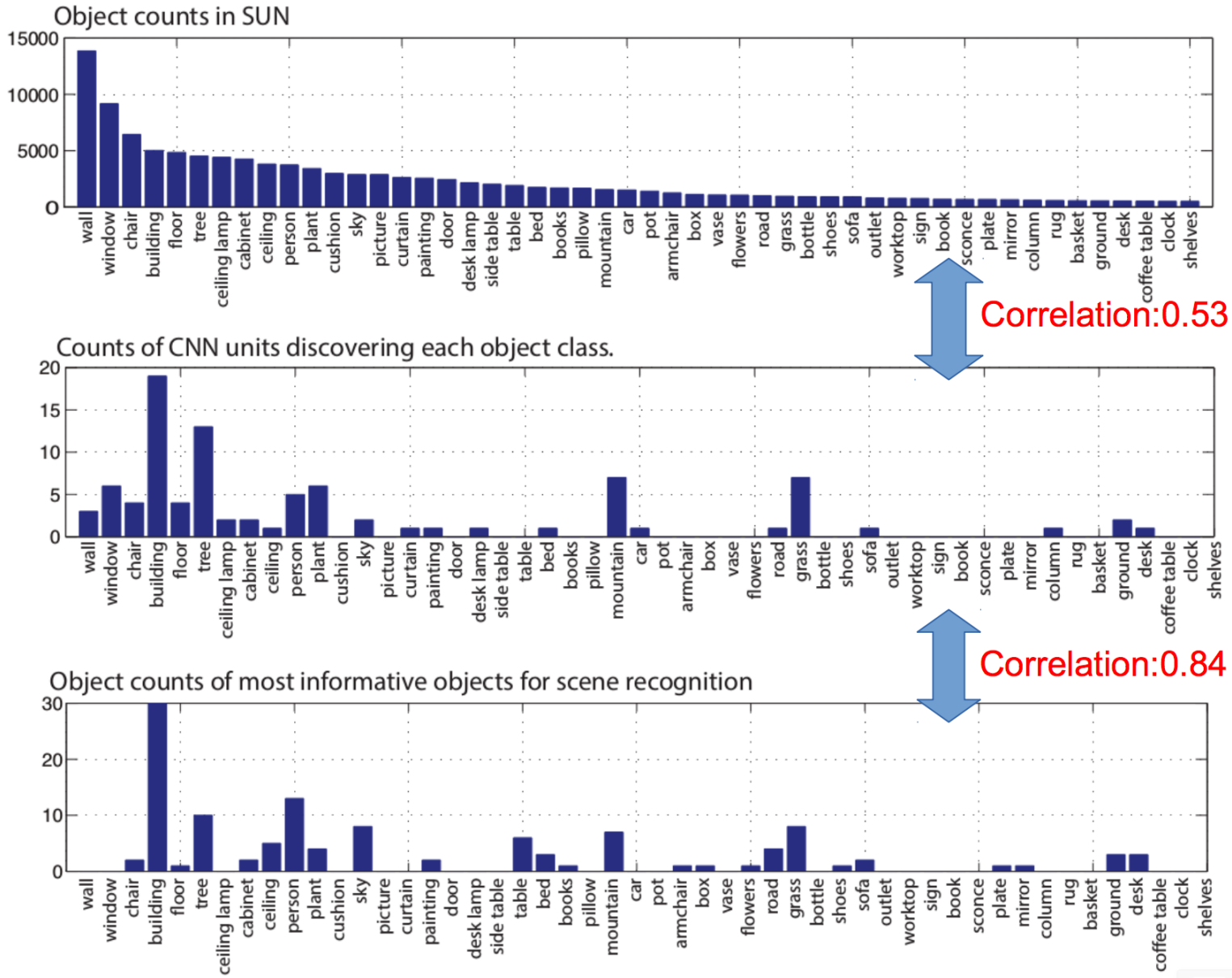
- Evaluating the performance on the emerged object detectors.
- The performance of many units is high, which provides strong evidence that they are indeed detecting those objects.







Evaluation on the SUN Database



Thank you!

- Fun Google+ post that I found very amusing regarding Alex Krizhevsky's talk at ECCV Workshop https://plus.google.com/+YannLeCunPhD/posts/JBBFfv2XgWM