PART I
Deep Learning Basics

Slides Borrowed From:

Angjoo Kanazawa (University of Maryland, College Park)

First of all what is Deep Learning?

- Composition of non-linear transformation of the data.
- Goal: **Learn useful representations, aka features, directly from data.**

- Many varieties, can be unsupervised or supervised.
- Today is about ConvNets, which is a **supervised** deep learning method.

Recap: Supervised Learning

$\{(\boldsymbol{x}^i, y^i), i = 1 \ldots P\}$ training dataset

$\boldsymbol{x}^i$ i-th input training example

y^i i-th target label

P number of training examples

Slide: M. Ranzato

Supervised Learning: Examples

Classification

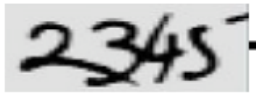
"dog"

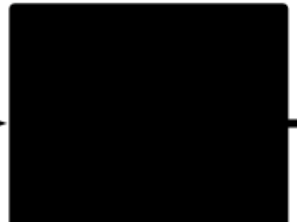
classification

Denoising

regression

OCR





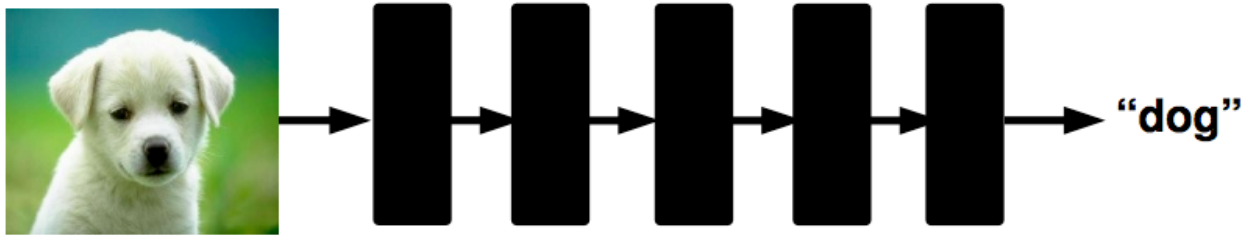
"2 3 4 5"

structured prediction

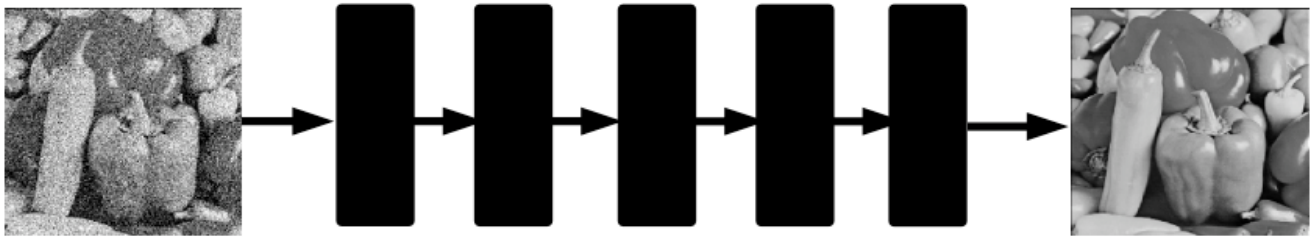
Slide: M. Ranzato

Supervised Deep Learning

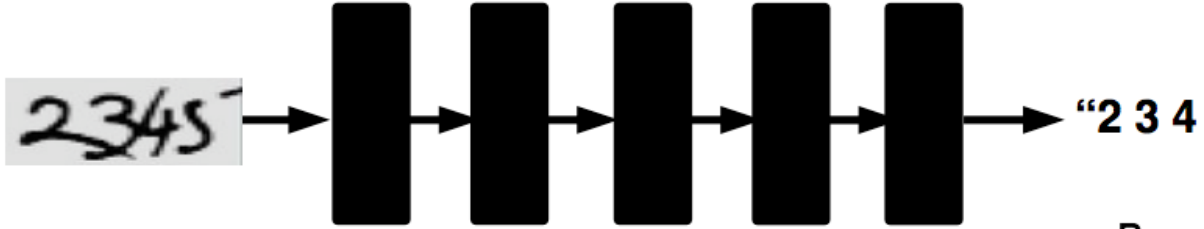
Classification



Denoising



OCR

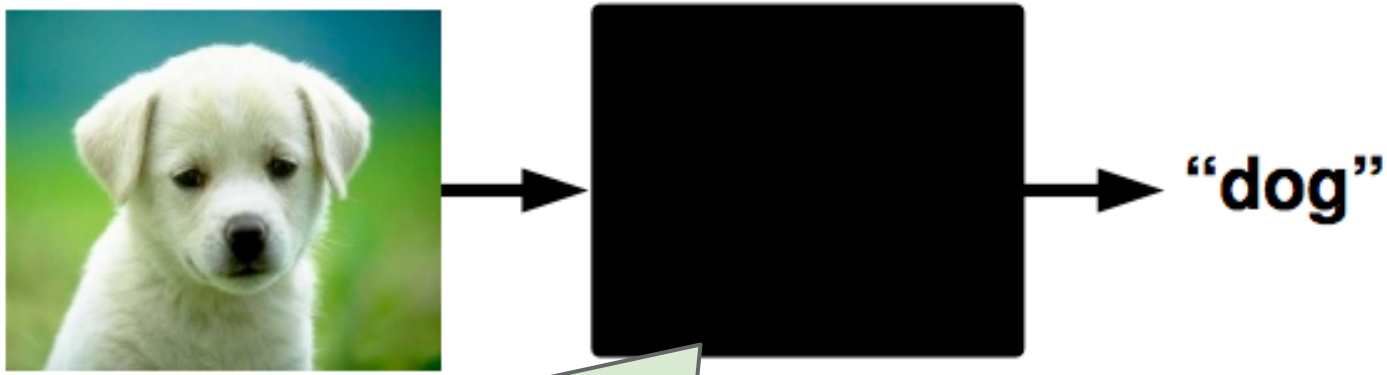


So deep learning is about learning feature representation in a compositional manner.

But wait,

why learn features?

The Black Box in a Traditional Recognition Approach



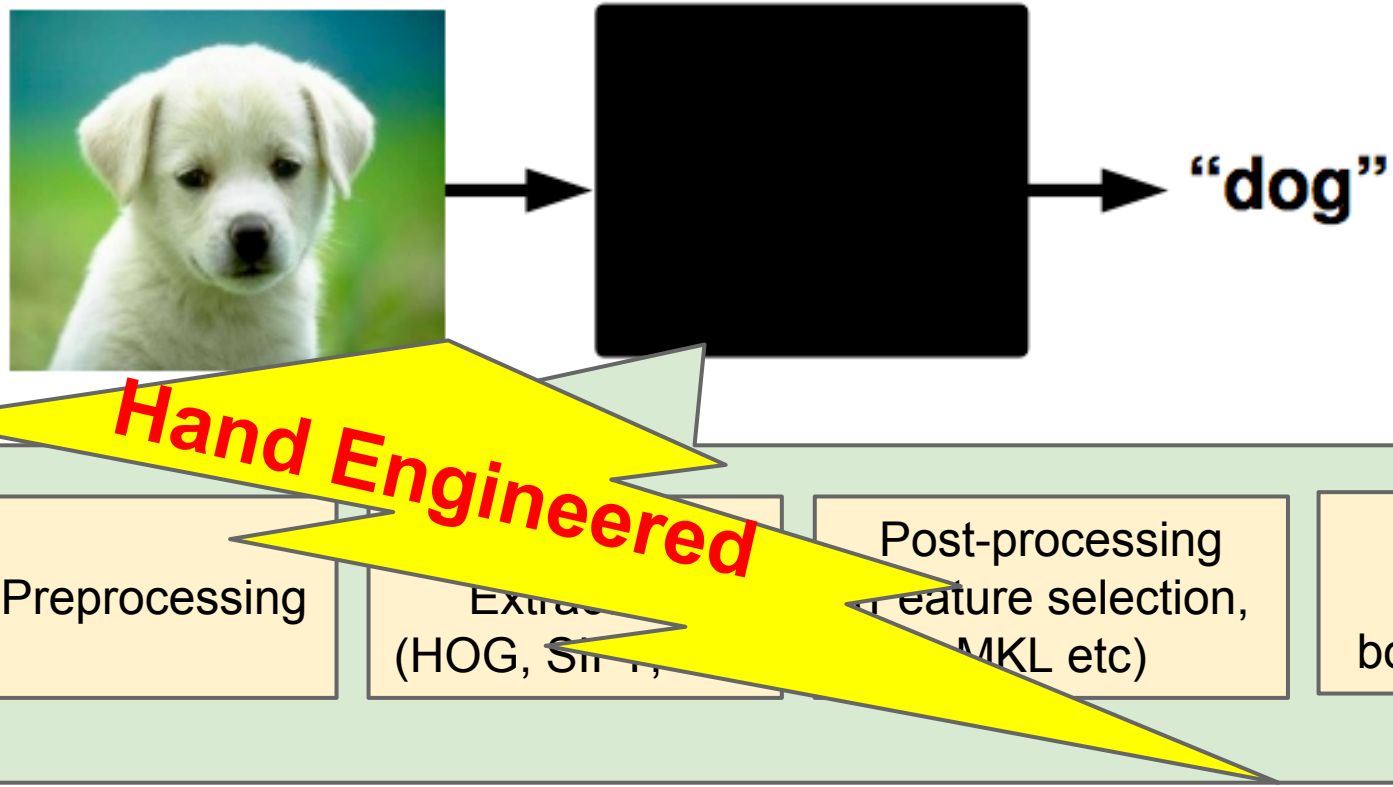
Preprocessing

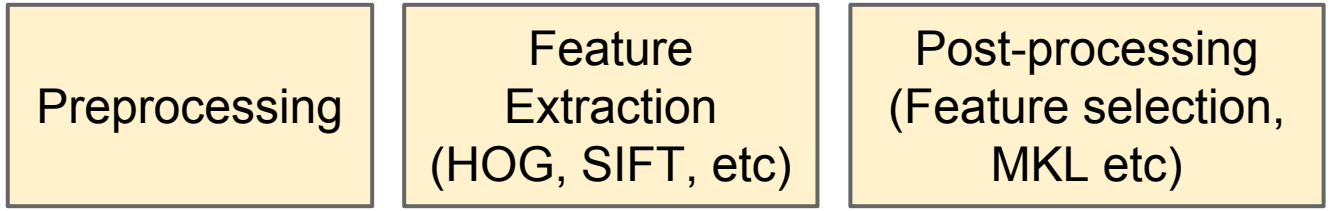
Feature Extraction (HOG, SIFT, etc)

Post-processing (Feature selection, MKL etc)

Classifier (SVM, boosting, etc)

The Black Box in a Traditional Recognition Approach





- Most critical for accuracy
- Most time-consuming in development
- What is the best feature???
- What is next?? Keep on crafting better features?

⇒ Let's learn feature representation directly from data.

Learn features and classifier *together*

⇒ Learn an end-to-end recognition system.

A non-linear map that takes raw pixels directly to labels.

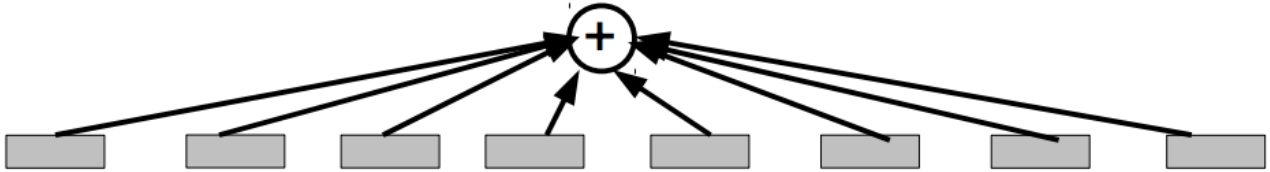
Q: How can we build such a highly non-linear system?

A: By combining simple building blocks we can make more and more complex systems.

Slide: M. Ranzato

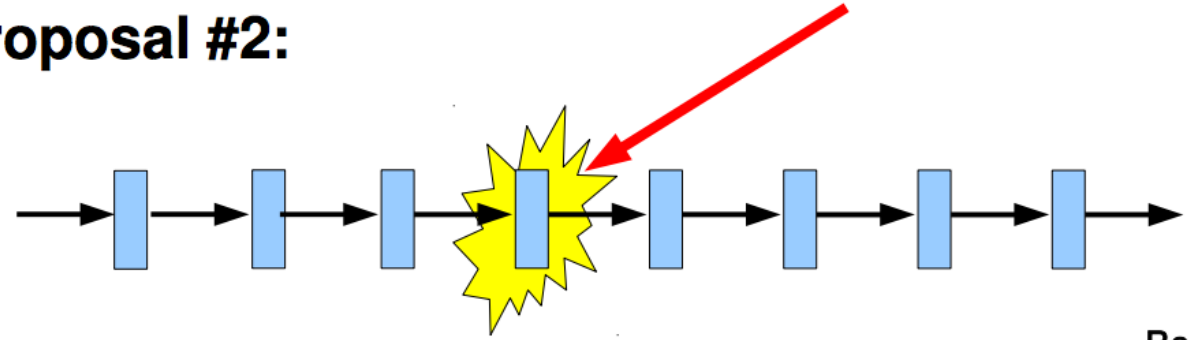
Building a complicated function

Proposal #1:



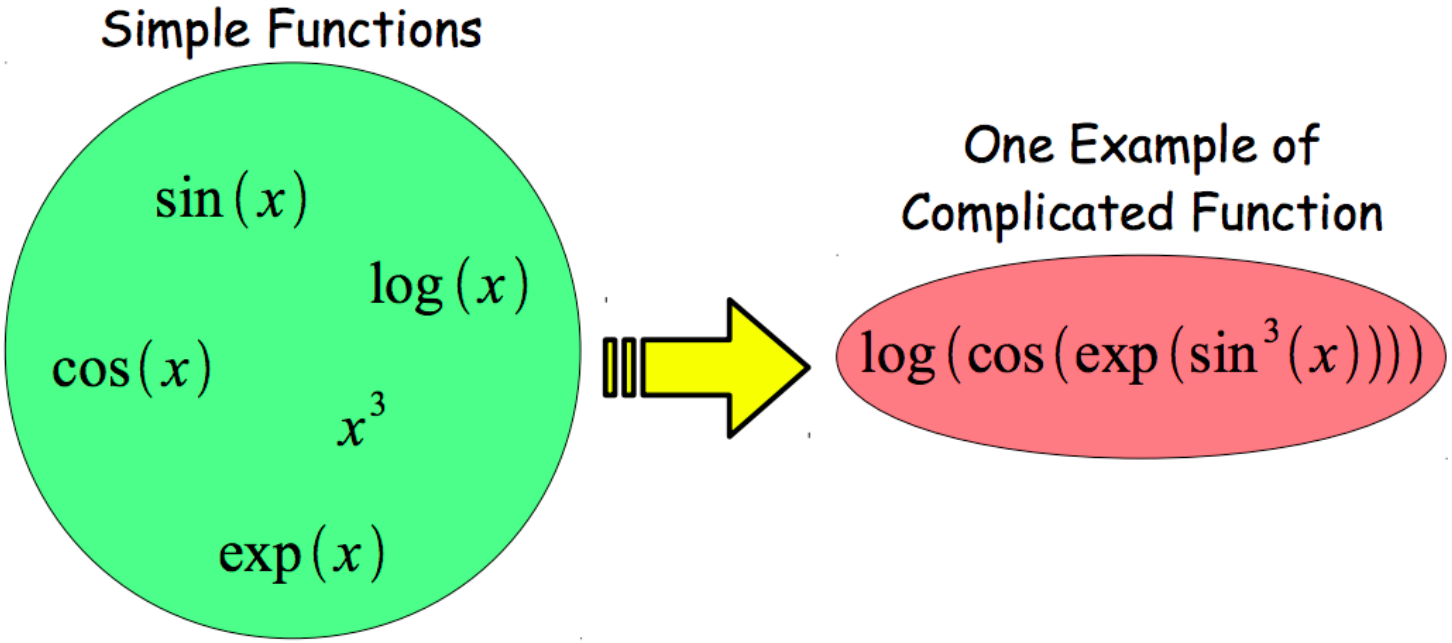
Each box is a simple nonlinear function

Proposal #2:



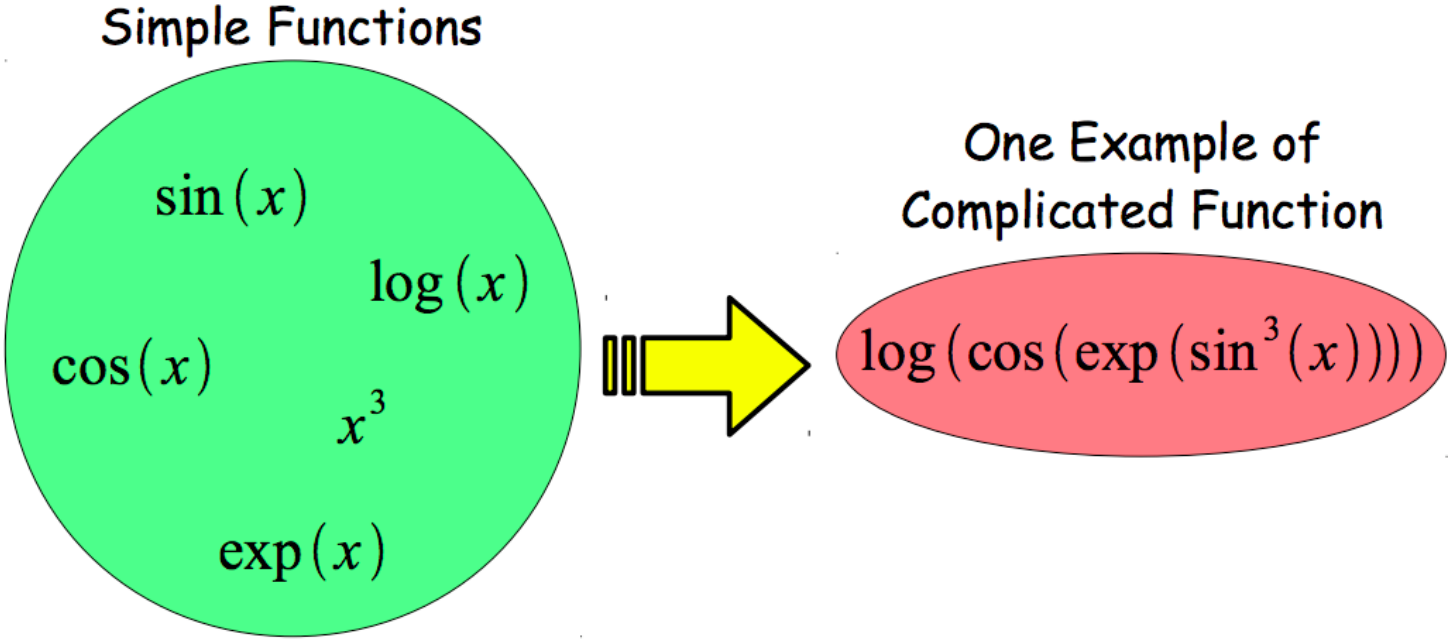
Ranzato

Building a complicated function



Slide: M. Ranzato

Building a complicated function



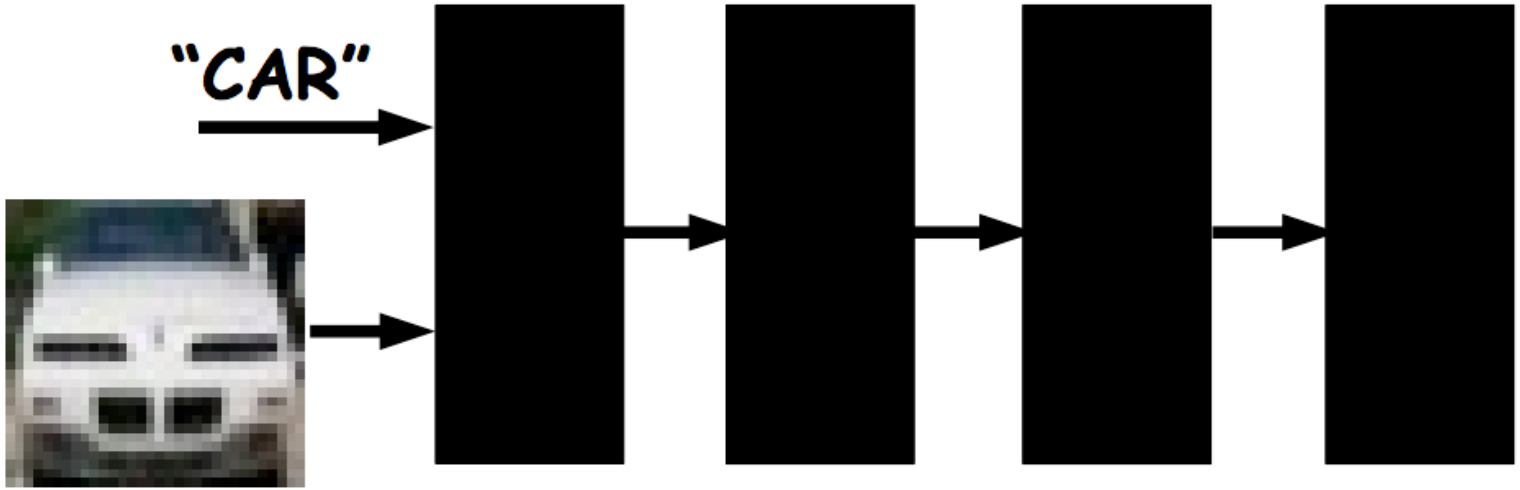
- Composition is at the core of deep learning methods
- Each “simple function” will have parameters subject to learning

Slide: M. Ranzato

Intuition behind Deep Neural Nets

Slide: M. Ranzato

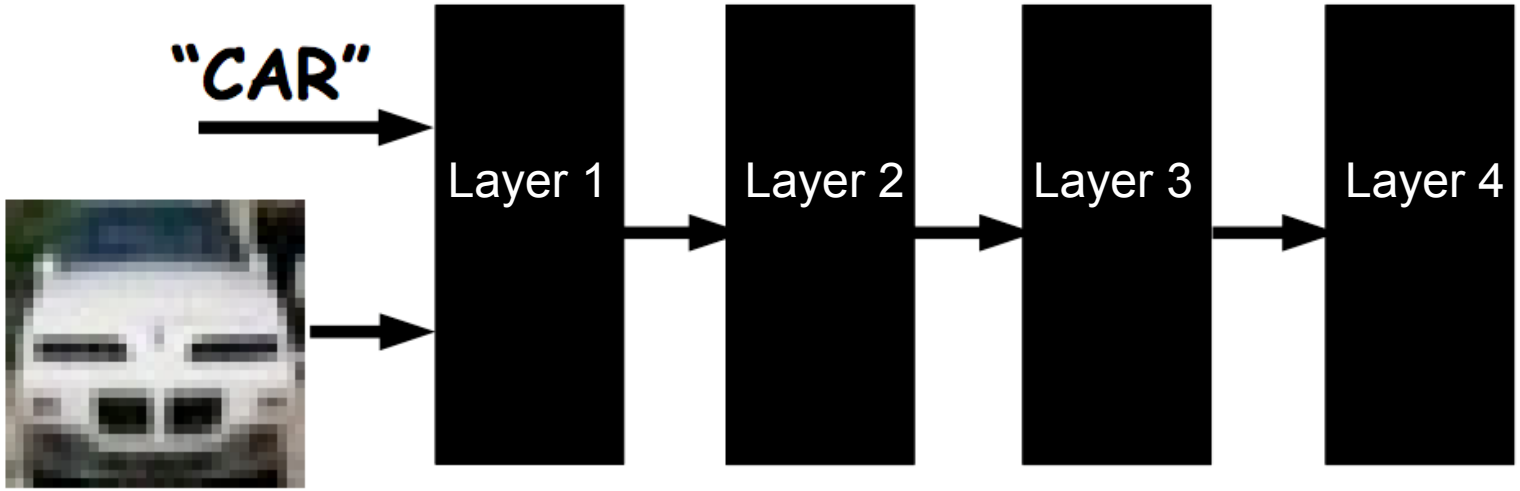
Intuition behind Deep Neural Nets



NOTE: Each black box can have trainable parameters.
Their composition makes a highly non-linear system.

Slide: M. Ranzato

Intuition behind Deep Neural Nets



NOTE: Each black box can have trainable parameters.
Their composition makes a highly non-linear system.

The final layer outputs a probability distribution of categories.

Slide: M. Ranzato

A simple single layer Neural Network

Consists of a linear combination of input through a nonlinear function:

$$z = Wx + b$$
$$a = f(z)$$

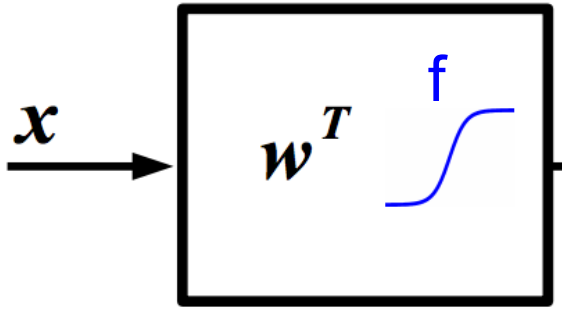
W is the weight parameter to be learned.

x is the output of the previous layer

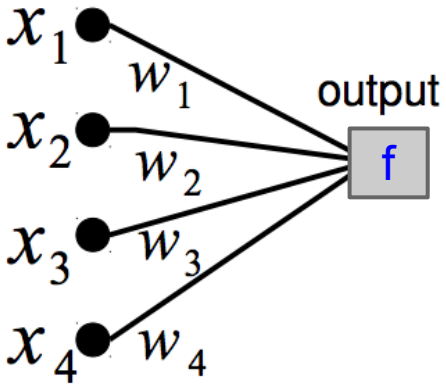
f is a simple nonlinear function. Popular choice is max(x,0), called ReLu (Rectified Linear Unit)

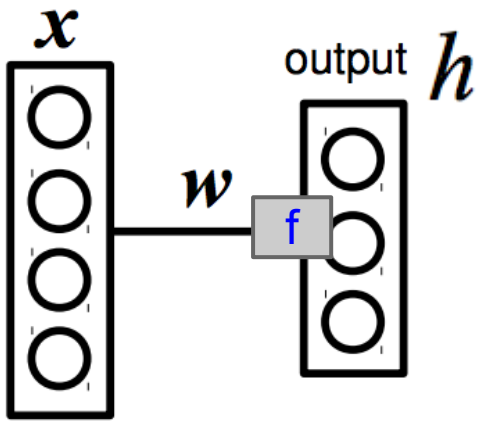
1 layer: Graphical Representation

$f(X;W)$ =

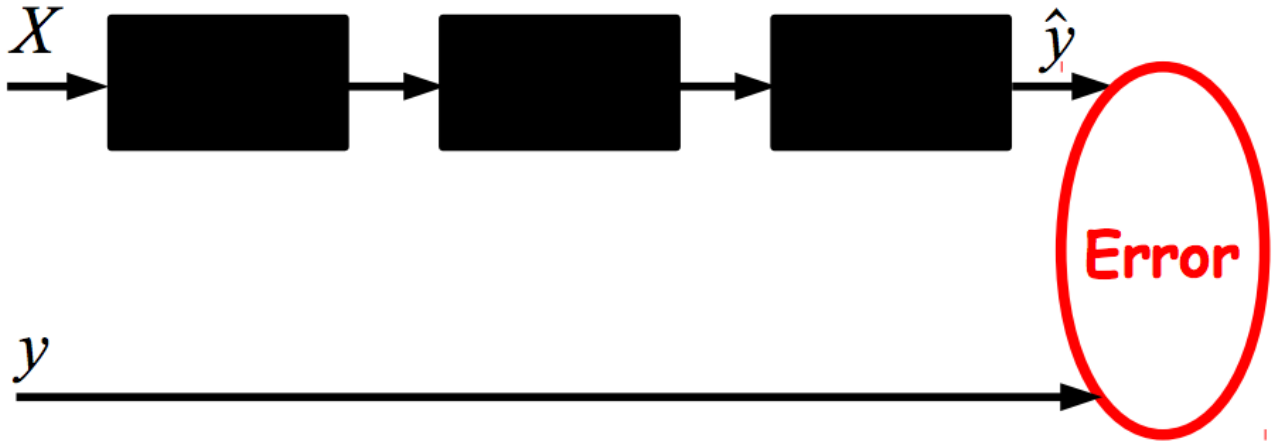


h is called a neuron, hidden unit or feature.





Joint training architecture overview



NOTE: Multi-layer neural nets with more than two layers are nowadays called **deep nets**!!

NOTE: User must specify number of layers, number of hidden units, type of layers and loss function.

Neural Net Training

A) Compute loss on small mini-batch

Slide: M. Ranzato

Neural Net Training

A) Compute loss on small mini-batch

Slide: M. Ranzato

Neural Net Training

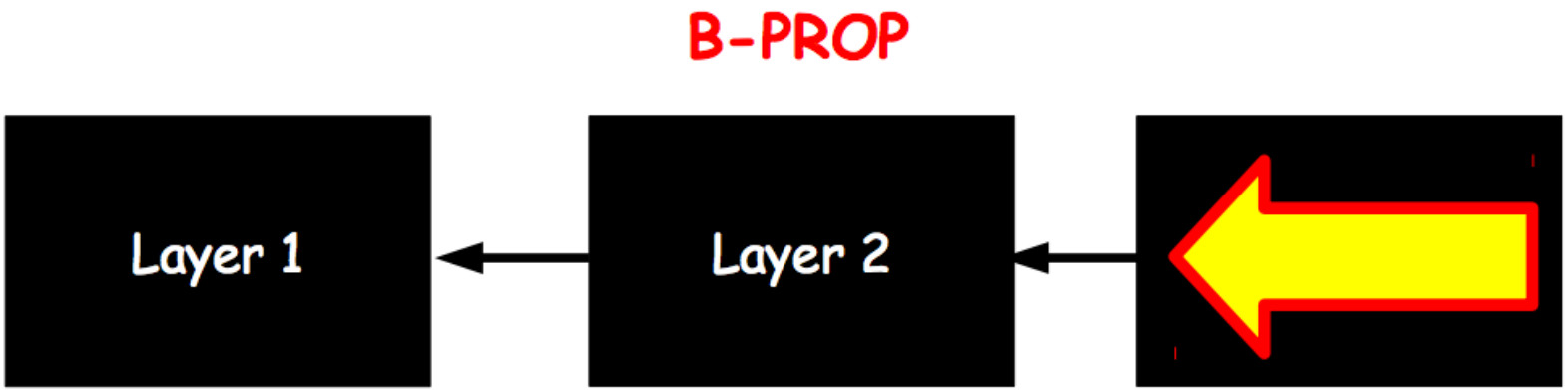
A) Compute loss on small mini-batch

Slide: M. Ranzato

Neural Net Training

A) Compute loss on small mini-batch

B) Compute gradient w.r.t. parameters

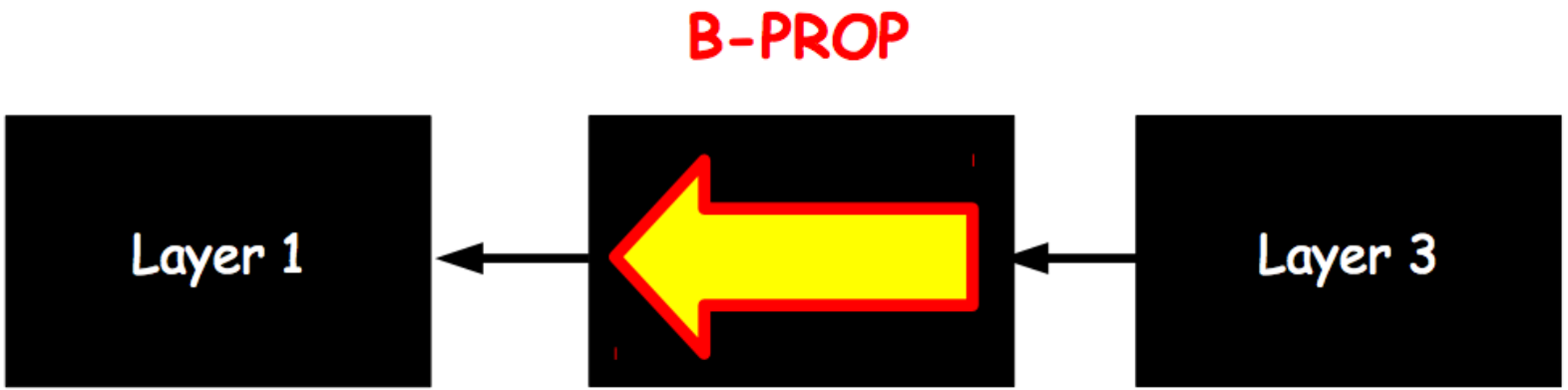


Slide: M. Ranzato

Neural Net Training

A) Compute loss on small mini-batch

B) Compute gradient w.r.t. parameters

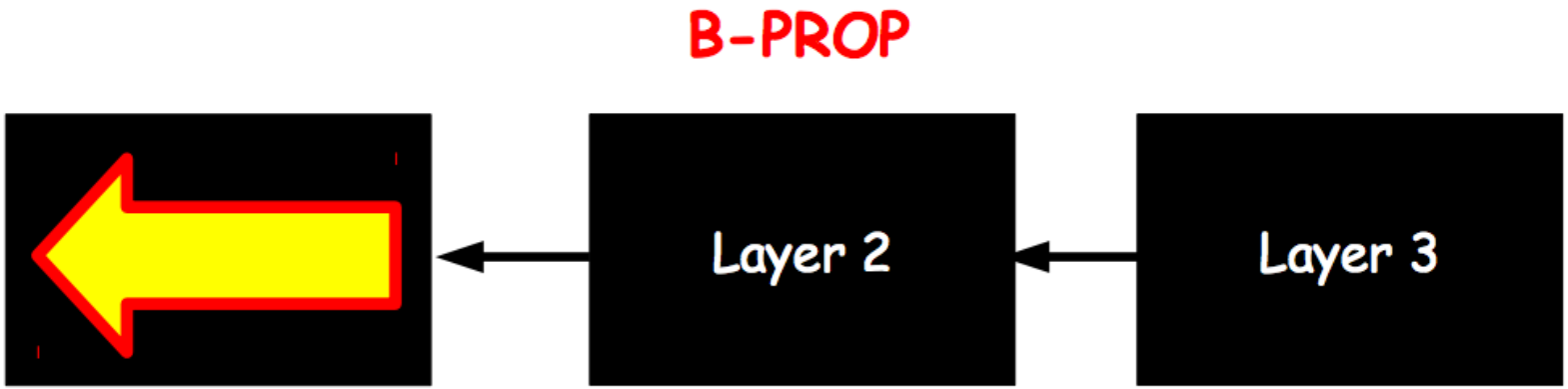


Slide: M. Ranzato

Neural Net Training

A) Compute loss on small mini-batch

B) Compute gradient w.r.t. parameters



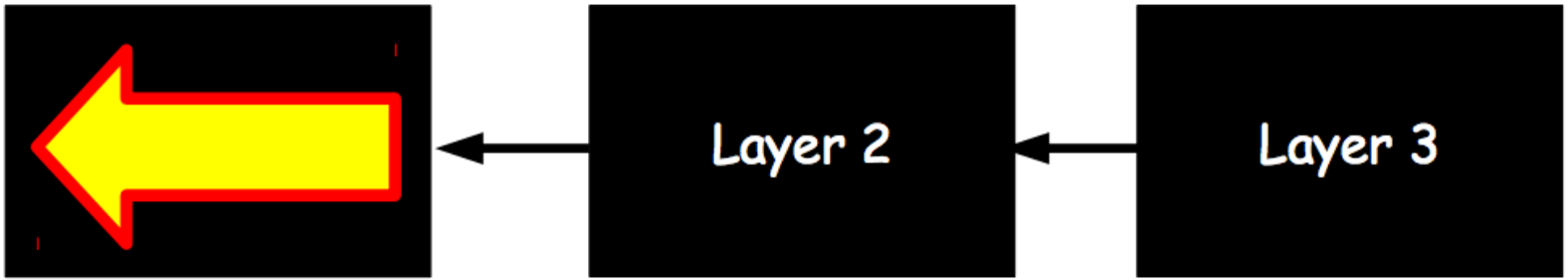
Slide: M. Ranzato

Neural Net Training

A) Compute loss on small mini-batch

B) Compute gradient w.r.t. parameters

C) Use gradient to update parameters $W \leftarrow W - \eta \frac{dL}{dW}$



Slide: M. Ranzato

PART II
ImageNet Classification with Convolutional Neural Networks

Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton,

Advances in Neural Information Processing Systems 2012

Slides Borrowed From:

Barnabás Póczos & Aarti Singh
(Carnegie Mellon University)

ImageNet

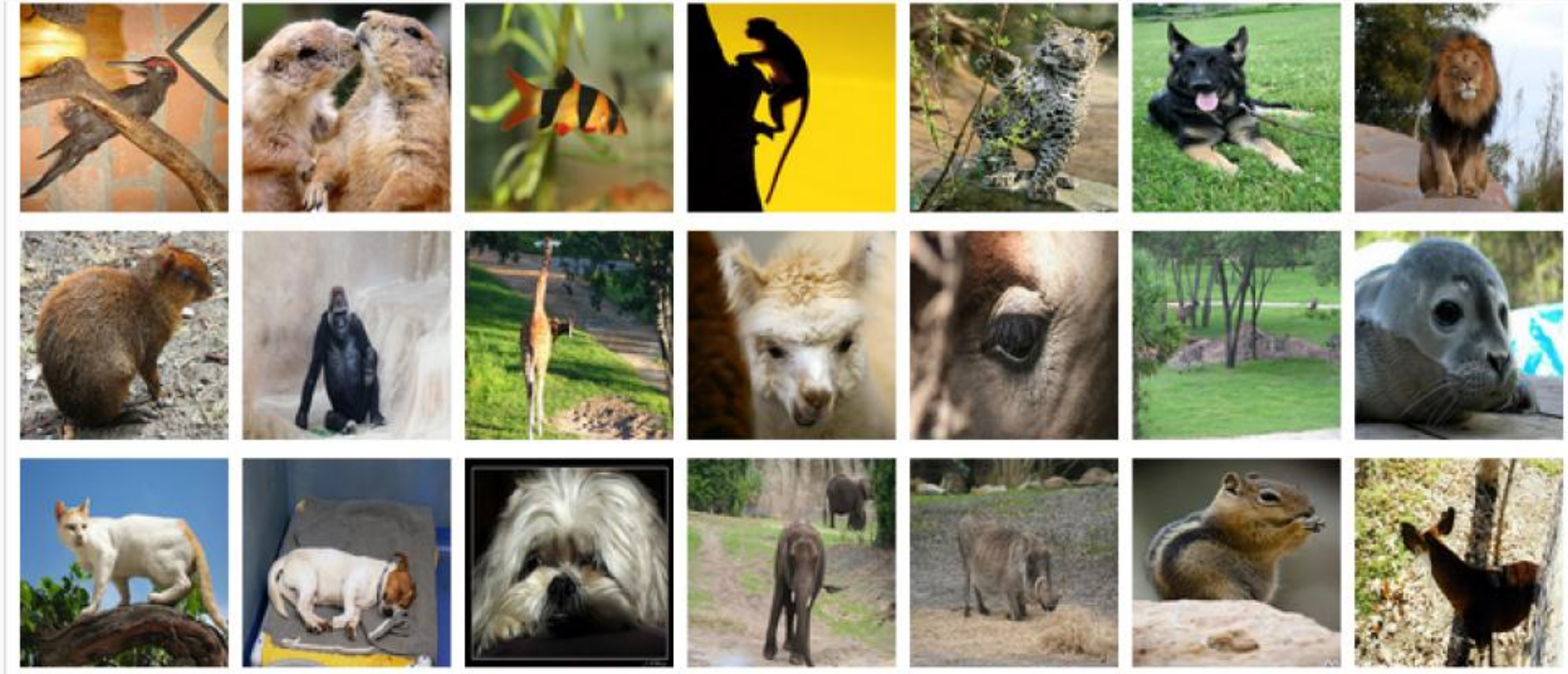
- 15M images
- 22K categories
- Images collected from Web
- Human labelers (Amazon's Mechanical Turk crowd-sourcing)
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC-2010)
  - 1K categories
  - 1.2M training images (~1000 per category)
  - 50,000 validation images
  - 150,000 testing images

- RGB images
- Variable-resolution, but this architecture scales them to 256x256 size

ImageNet

Classification goals:

- Make 1 guess about the label (Top-1 error)
- make 5 guesses about the label (Top-5 error)



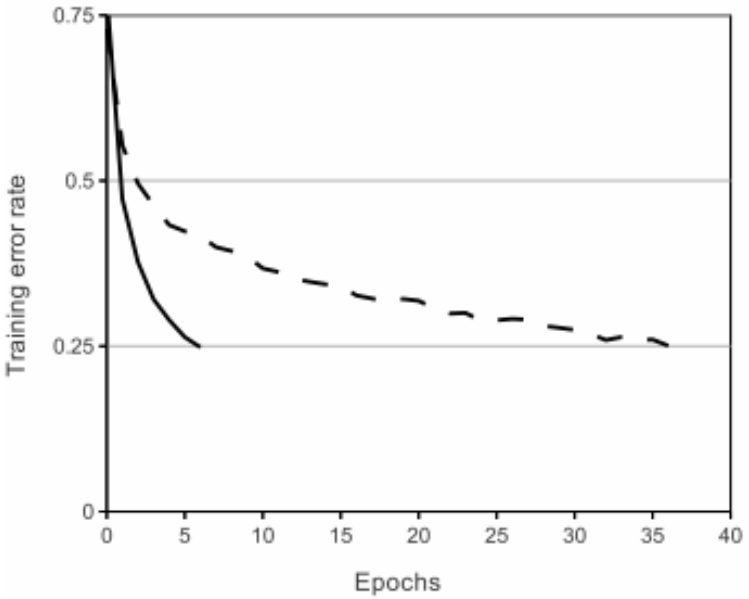
The Architecture

Typical nonlinearities: $f(x) = \tanh(x)$

$$f(x) = (1 + e^{-x})^{-1}$$

Here, however, Rectified Linear Units (ReLU) are used: $f(x) = \max(0, x)$

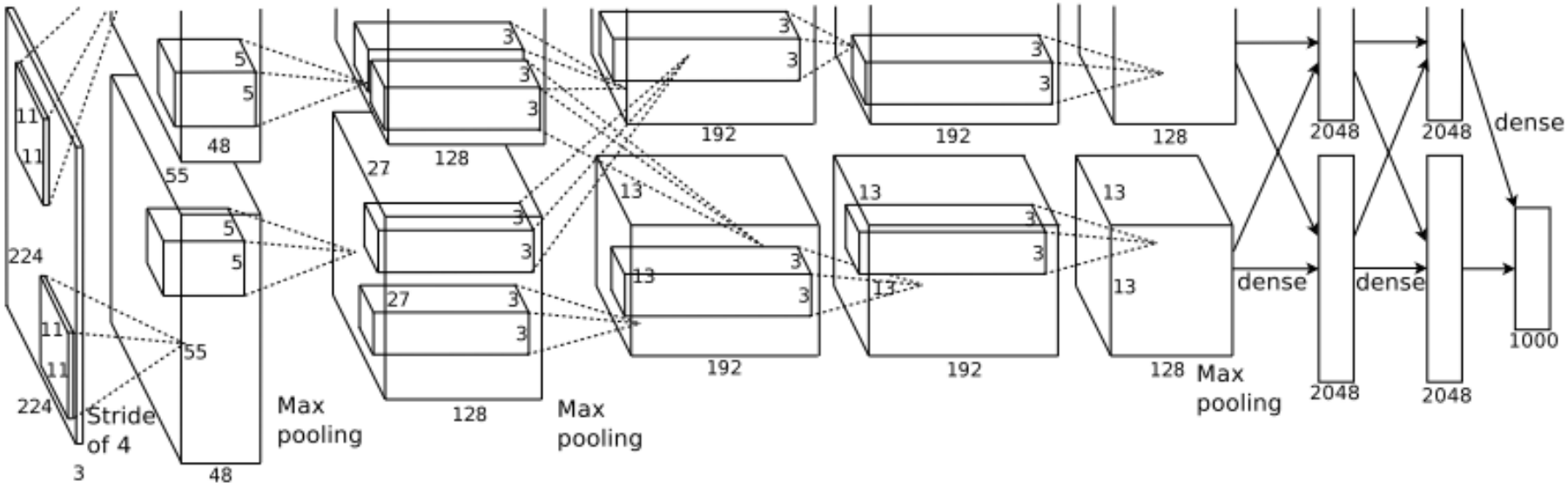
Empirical observation: Deep convolutional neural networks with ReLUs train several times faster than their equivalents with tanh units



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 Architecture



The first convolutional layer filters the 224×224×3 input image with 96 kernels of size 11×11×3 with a stride of 4 pixels (this is the distance between the receptive field centers of neighboring neurons in the kernel map. 224/4=56

The pooling layer: form of non-linear down-sampling. Max-pooling partitions the input image into a set of rectangles and, for each such sub-region, outputs the maximum value

The Architecture

- Trained with stochastic gradient descent
- on two NVIDIA GTX 580 3GB GPUs
- for about a week

- 650,000 neurons
- 60,000,000 parameters
- 630,000,000 connections
- 5 convolutional layer, 3 fully connected layer
- Final feature layer: 4096-dimensional

Dropout

- We know that combining different models can be very useful (Mixture of experts, majority voting, boosting, etc)

- Training many different models, however, is very time consuming.

The solution:

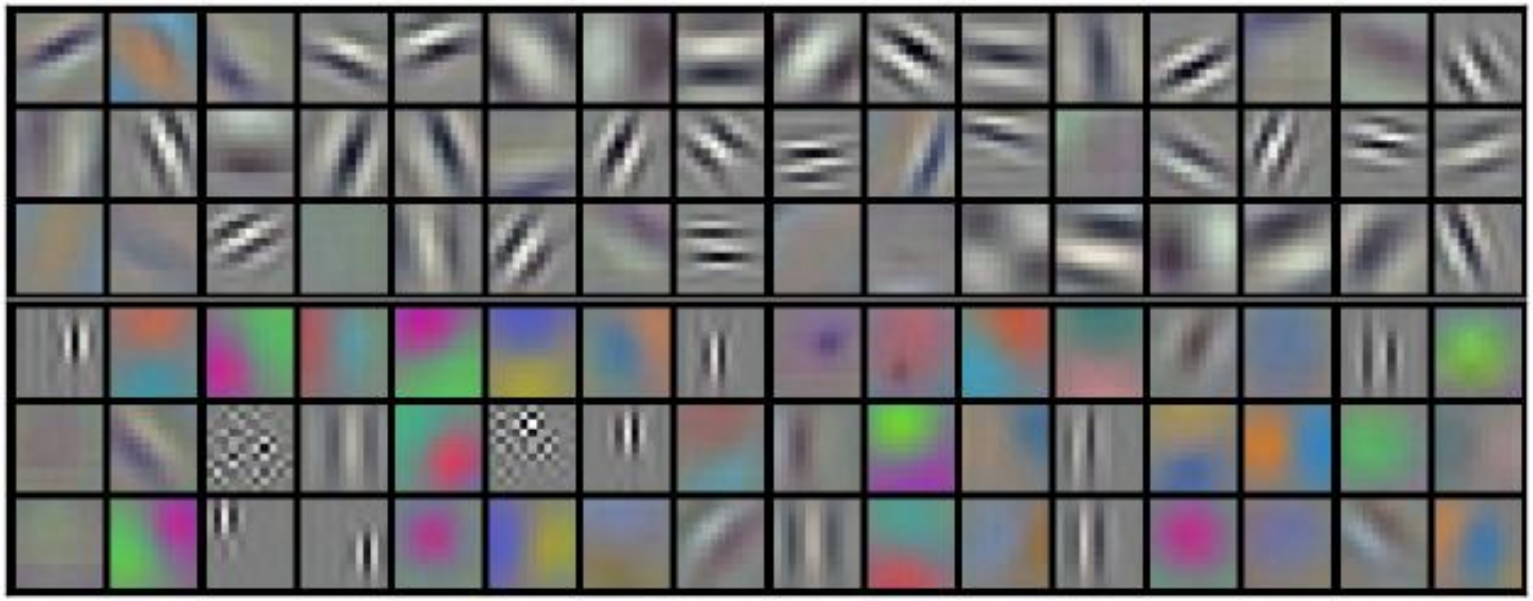
Dropout: set the output of each hidden neuron to zero w.p. 0.5.

Dropout

Dropout: set the output of each hidden neuron to zero w.p. 0.5.

- The neurons which are "dropped out" in this way do not contribute to the forward pass and do not participate in backpropagation.
- So every time an input is presented, the neural network samples a different architecture, but all these architectures share weights.
- This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons.
- It is, therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons.
- Without dropout, our network exhibits substantial overfitting.
- Dropout roughly doubles the number of iterations required to converge.

The 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

Looks like Gabor wavelets, ICA filters…

Results

Results on the test data:

top-1 error rate: 37.5%

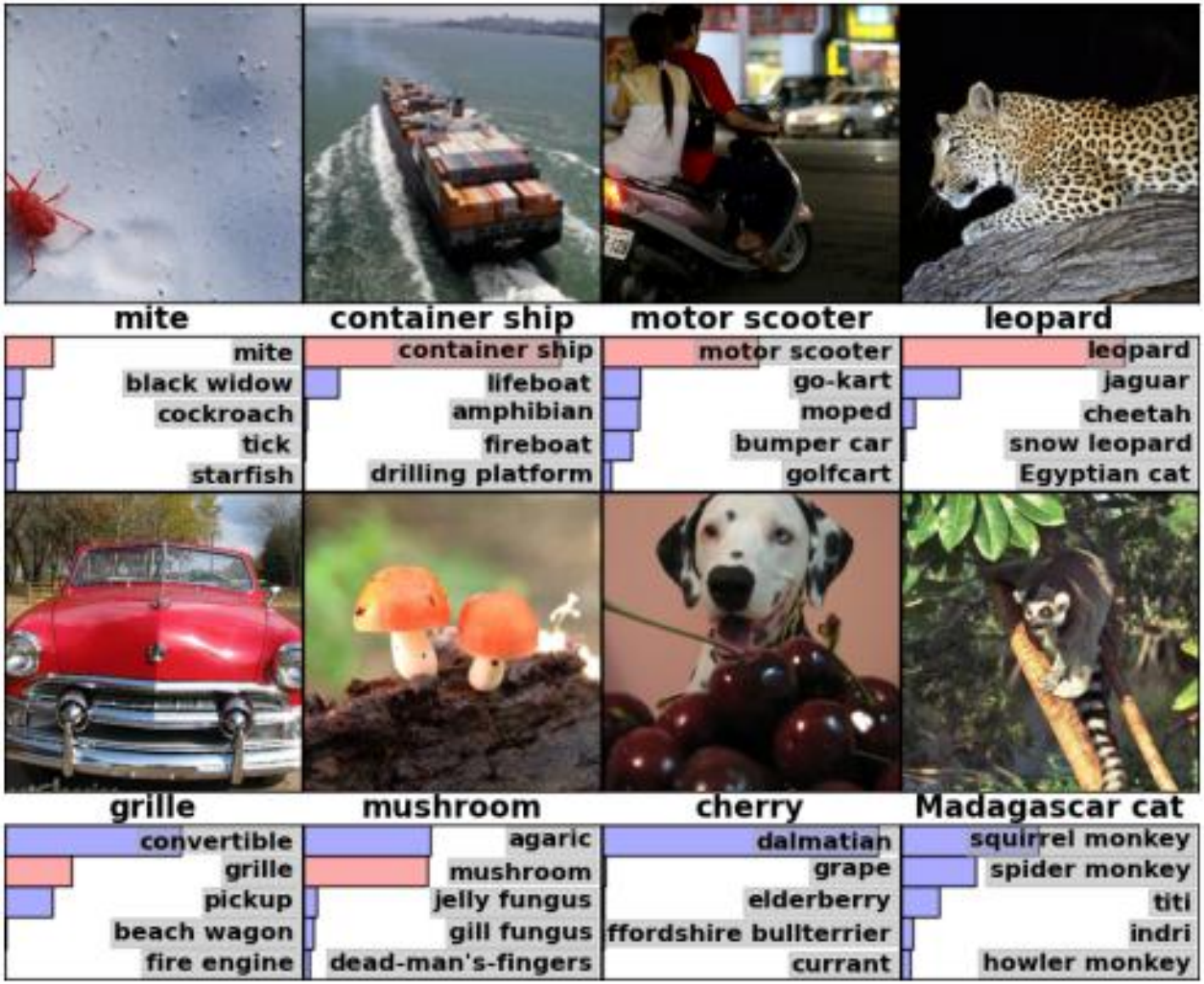
top-5 error rate: 17.0%

ILSVRC-2012 competition:

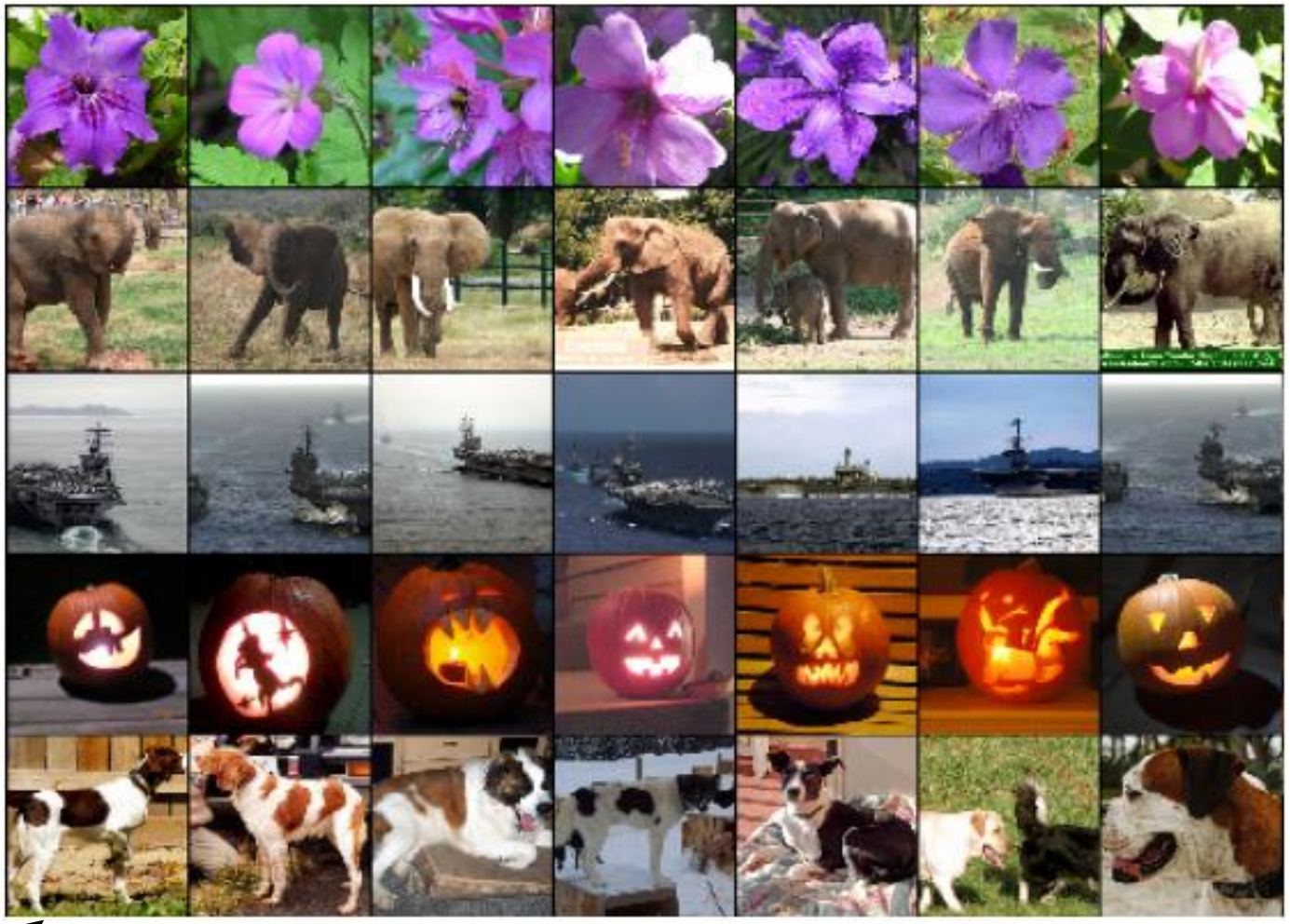
15.3% accuracy

2nd best team: 26.2% accuracy

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: Image similarity



Test column

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.