ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton
University of Toronto, NIPS 2012

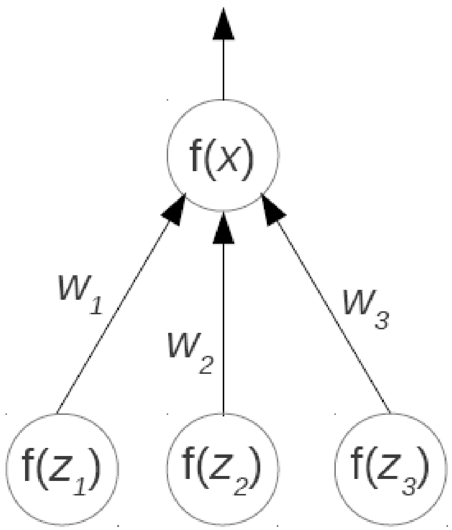
Presenter: Guangnan Ye

Main Idea

- A deep convolutional neural network is trained to classify the 1.2 million ImageNet images into 1000 different classes.
- The neural network contains 60 million parameters and 650,000 neurons.
- The state-of-the-art performance is achieved with the error rate improving from 26.2% to 15.3%.

Neural Networks

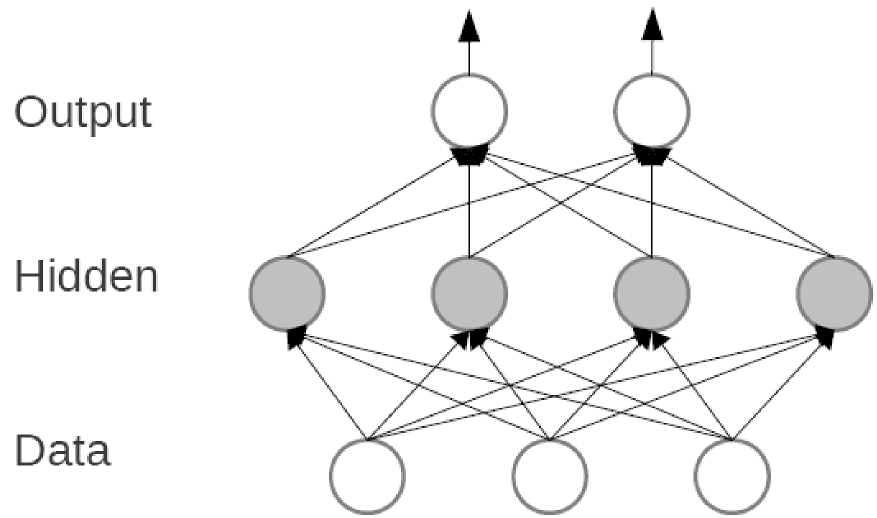
- A neuron



$x = w_1 f(z_1) + w_2 f(z_2) + w_3 f(z_3)$

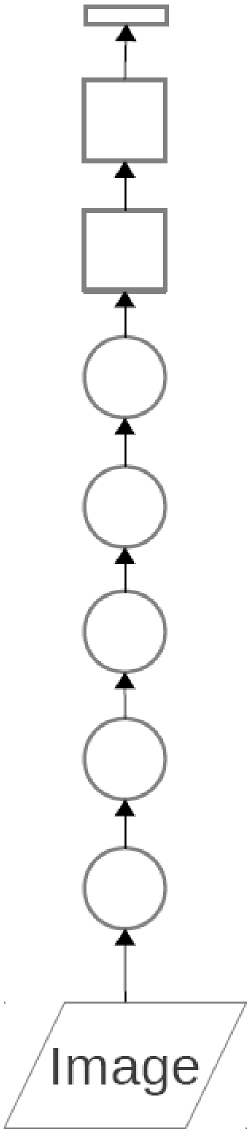
x is called the total input to the neuron, and f(x) is its output

- A neural network

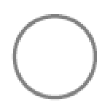


A neural network computes a differentiable function of its input. For example, ours computes: p(label | an input image)

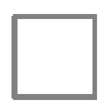
Model Overview



- **Deep**: 7 hidden “weight” layers
- **Learned**: all feature extractors initialized at white Gaussian noise and learned from the data
- Entirely supervised
- **More data = good**

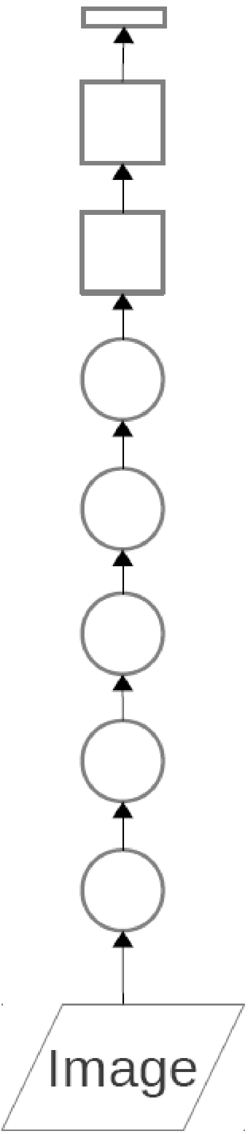


Convolutional layer: convolves its input with a bank of 3D filters, then applies point-wise non-linearity



Fully-connected layer: applies linear filters to its input, then applies point-wise non-linearity

Model Overview



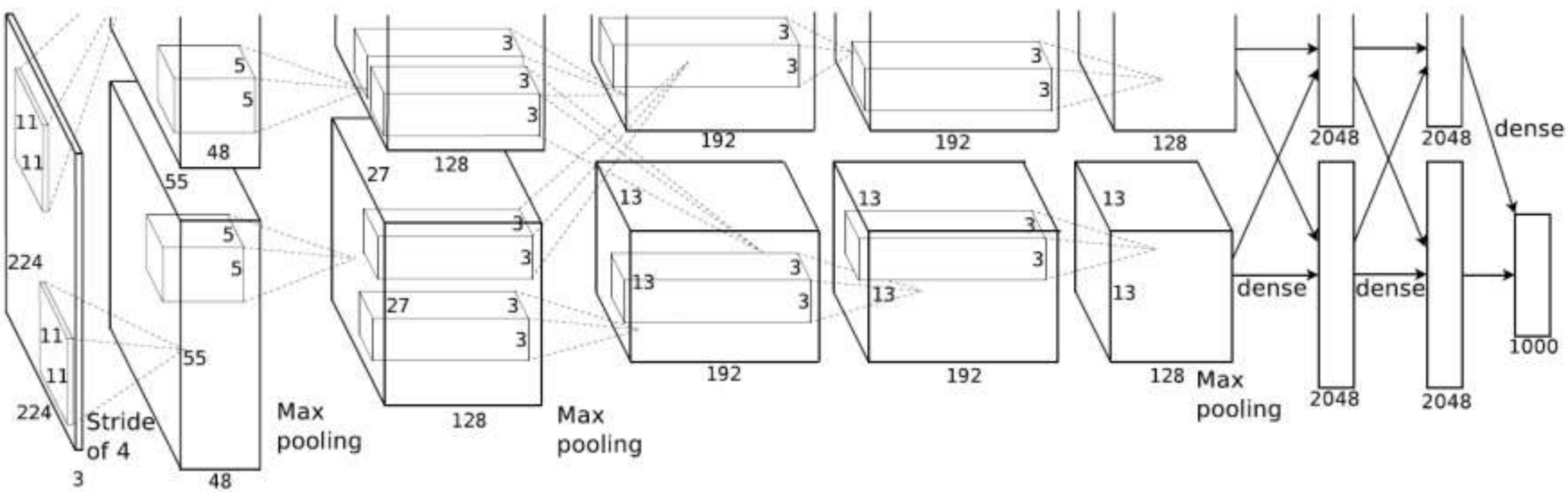
- Trained with stochastic gradient descent on two NVIDIA GPUs for about a week
- 650,000 neurons
- 60,000,000 parameters
- 630,000,000 connections
- **Final feature layer:** 4096-dimensional

Convolutional layer: convolves its input with a bank of 3D filters, then applies point-wise non-linearity

Fully-connected layer: applies linear filters to its input, then applies point-wise non-linearity

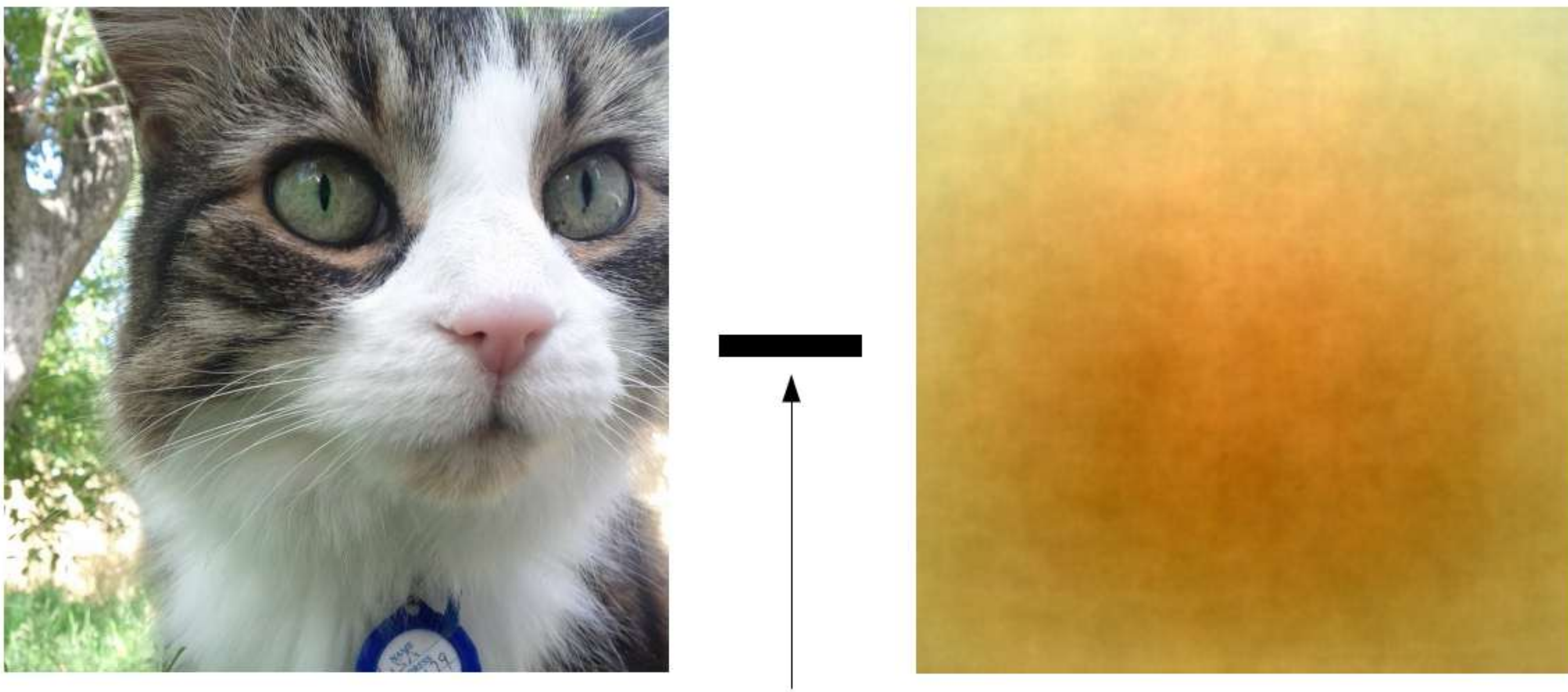
Model Achitecture

- Max-pooling layers follow first, second, and fifth convolutional layers
- The number of neurons in each layer is given by 253440, 186624, 64896, 64896, 43264, 4096, 4096, 1000



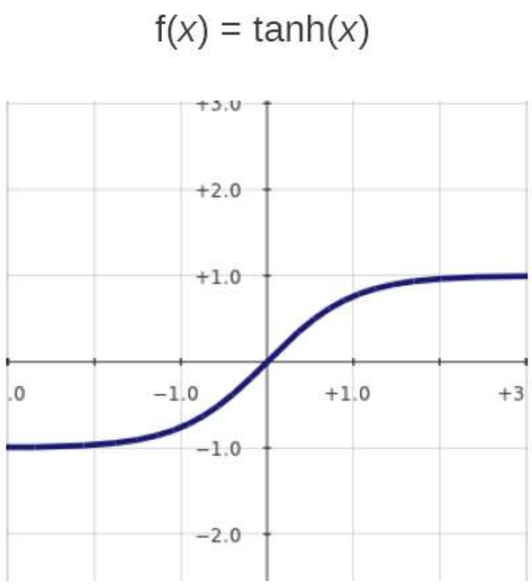
Detail: Input Representation

- Centered (0-mean) RGB values.



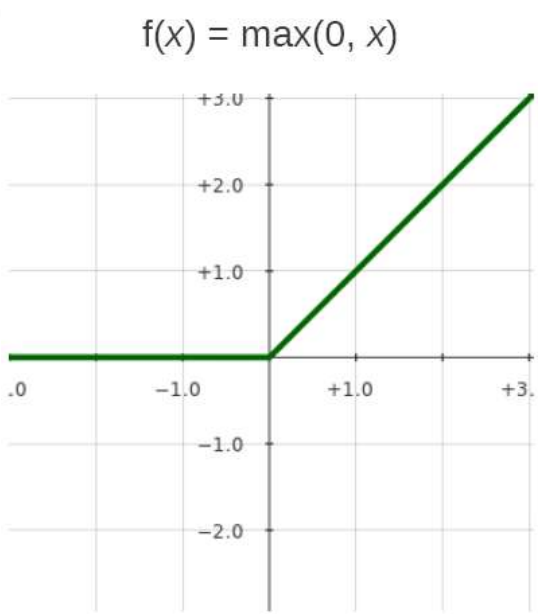
An input image (256x256) | Minus sign | The mean input image

Detail: Neurons



Very bad
(slow to train)

X



Very good
(quick to train)

√

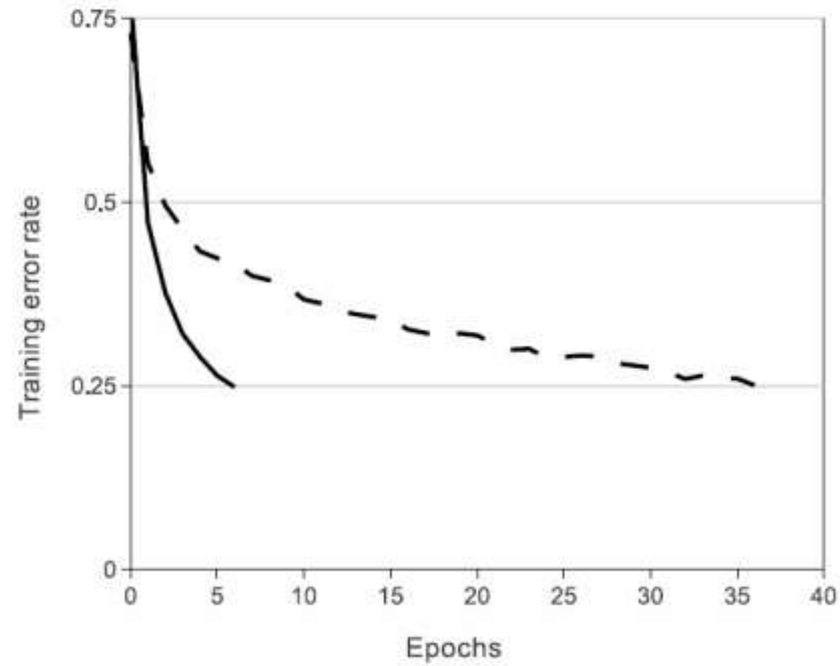
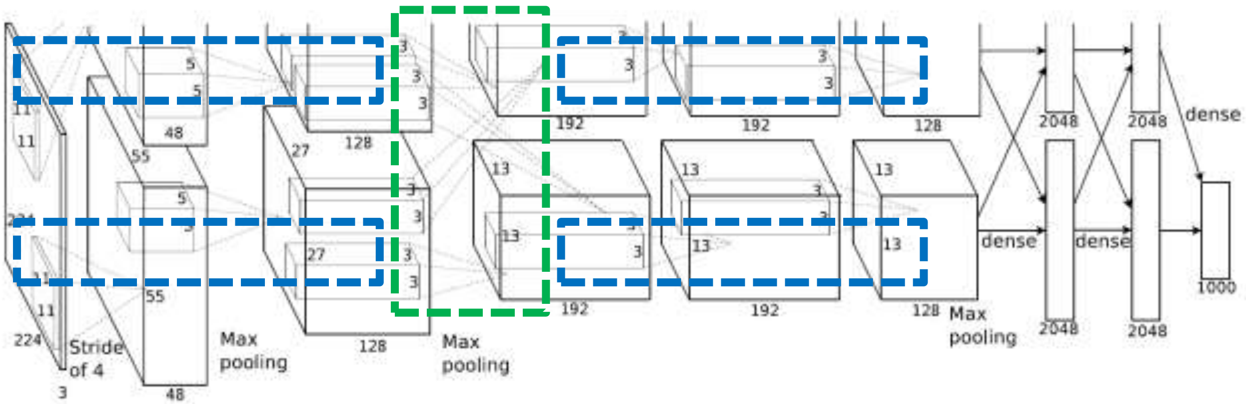


Figure. A four layer convolutional neural network with ReLUs (**solid line**) reaches a 25% training error rate on CIPAR-10 faster than tanh neurons(**dashed line**)

Other Details

- Training on Multiple GPUs (error rate ↓1.2%)



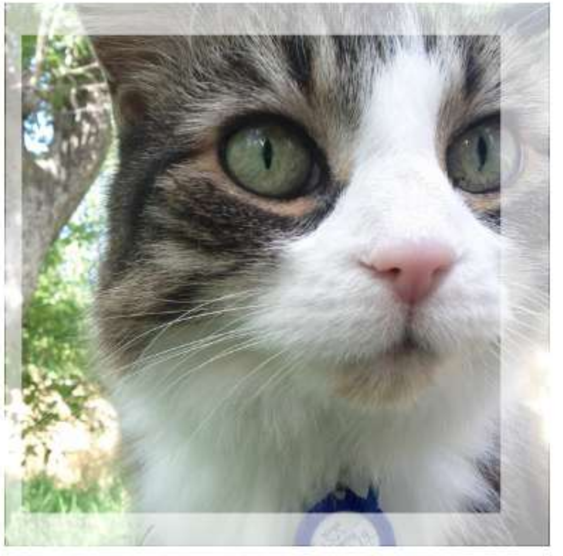
- Local Response Normalization (error rate ↓1.2%)

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

- Overlapping Pooling (error rate ↓0.3%)

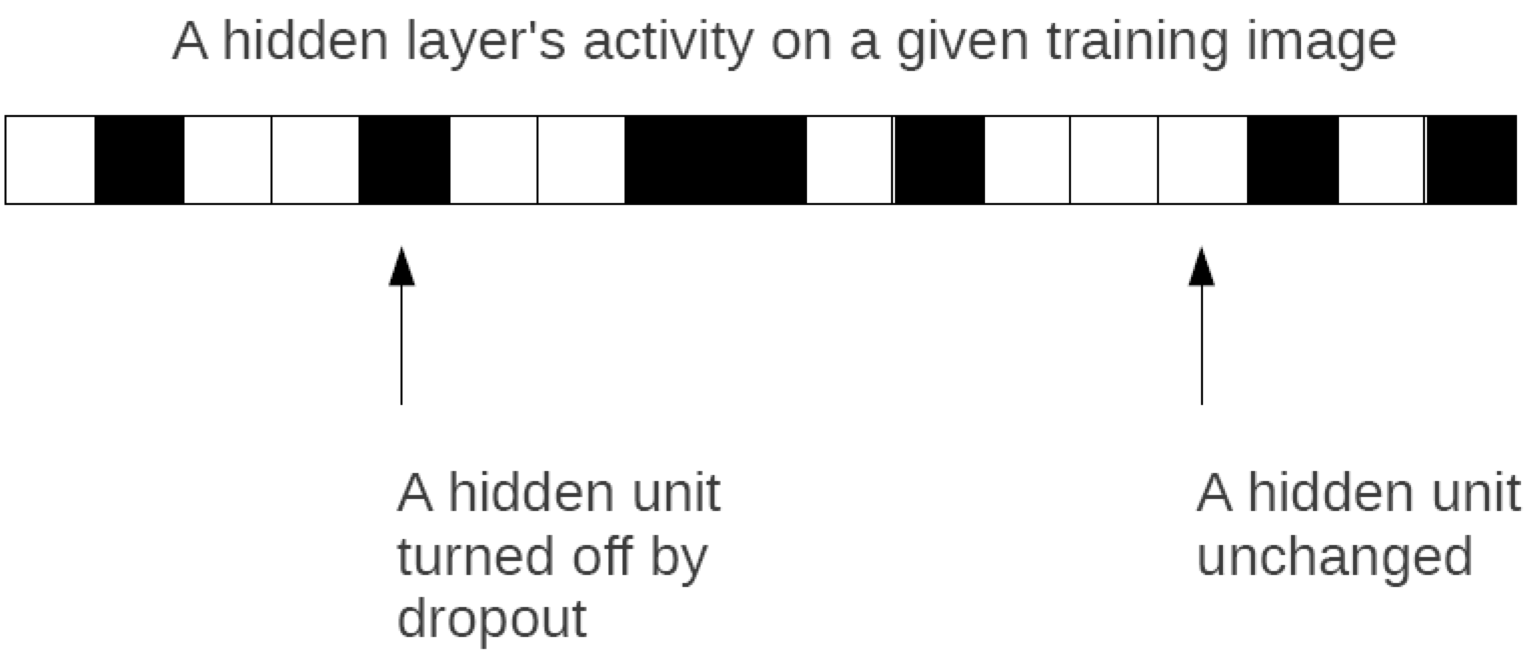
Reducing Overfitting

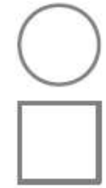
- Data augmentation
  - The neural net has 60M real-valued parameters and 650,000 neurons which overfits a lot. 224x224 patches extracted randomly from 256x256 images, and also their horizontal reflections
  - RGB intensities altered by PCA so that invariant to change in the intensity and color of the illumination



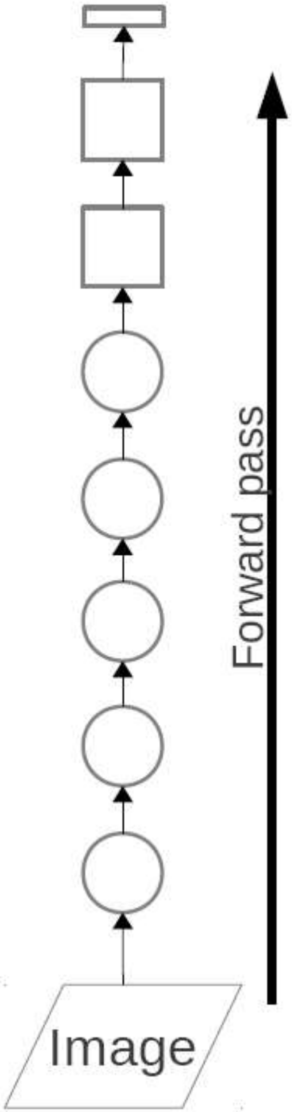
Reducing Overfitting

- Dropout
  - Motivation: Combining many different models is a successful way to reduce test errors.
  - Independently set each hidden unit activity to zero with 0.5 probability





Training

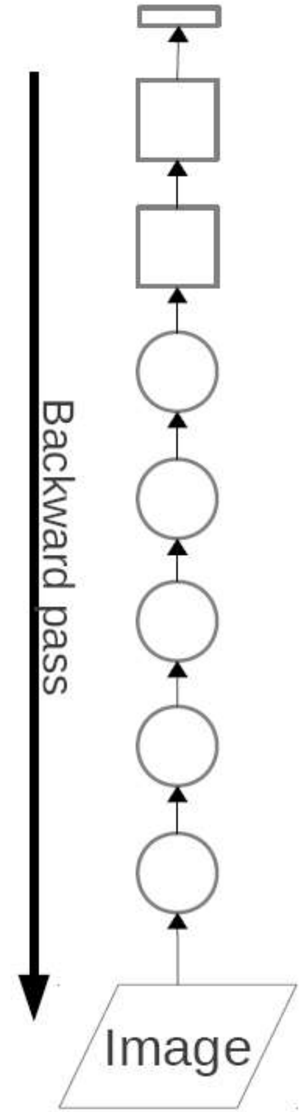


Using stochastic gradient descent and the *backpropagation algorithm* (just repeated application of the chain rule)

Update rule for weight w:

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

Figure. 96 convolutional kernels of size 11X11X3 learned by the first convolutional layer on the 224X224X3 input images



Results on ImageNet

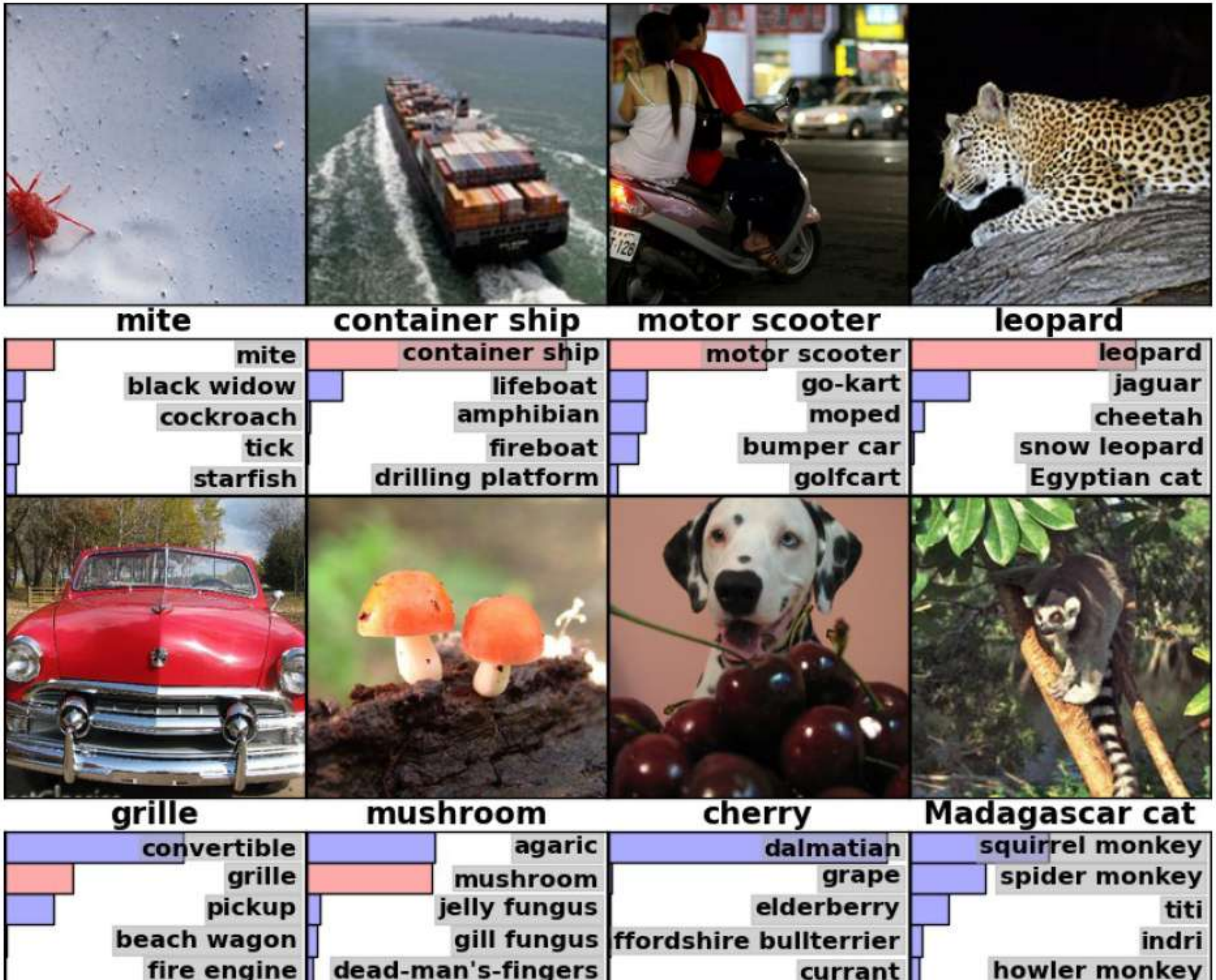
Model	**Top-1**	**Top-5**
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Table. Comparison of results on ILSVRC-2010 test set.

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. Comparison of error rates on ILSVRC-2012 validation and test sets.

Qualitative Evaluations- Validation Classification



Qualitative Evaluations- Retrieval

Query

OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

Pierre Sermanet, David Eigen,

Xiang Zhang, Michael Mathieu, Rob Fergus,

Yann LeCun

Courant Institute, NYU

Classification

Model:

- Layer 1-5 for feature extraction
- Layer 6++ for classification
- Drop out (0.5) on layer 6++
- Convolution with linear filter + nonlinear function (max pooling)
- Trained on ImageNet 2012
  (1.2 million images, 1000 classes)
- Fixed input size
- Trainin using gradient descent

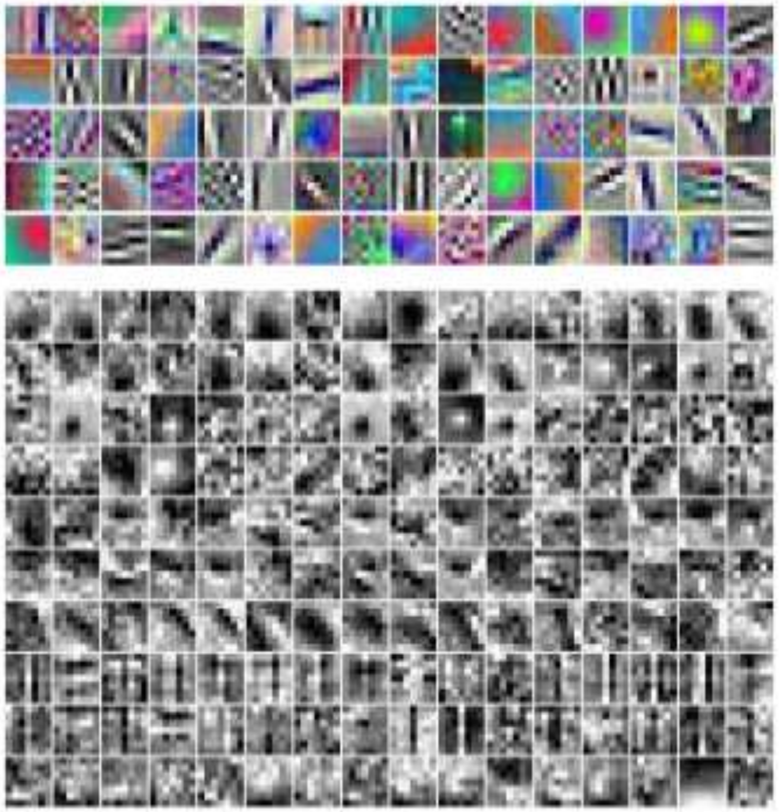


Figure 2: **Layer 1 (top) and layer 2 filters (bottom).**

ConvNets and Sliding Windows

- Inherently efficient with convolution because computation is shared for overlapping windows
- explore image at each location, at multiple scales
- More views for voting = robust while efficcient

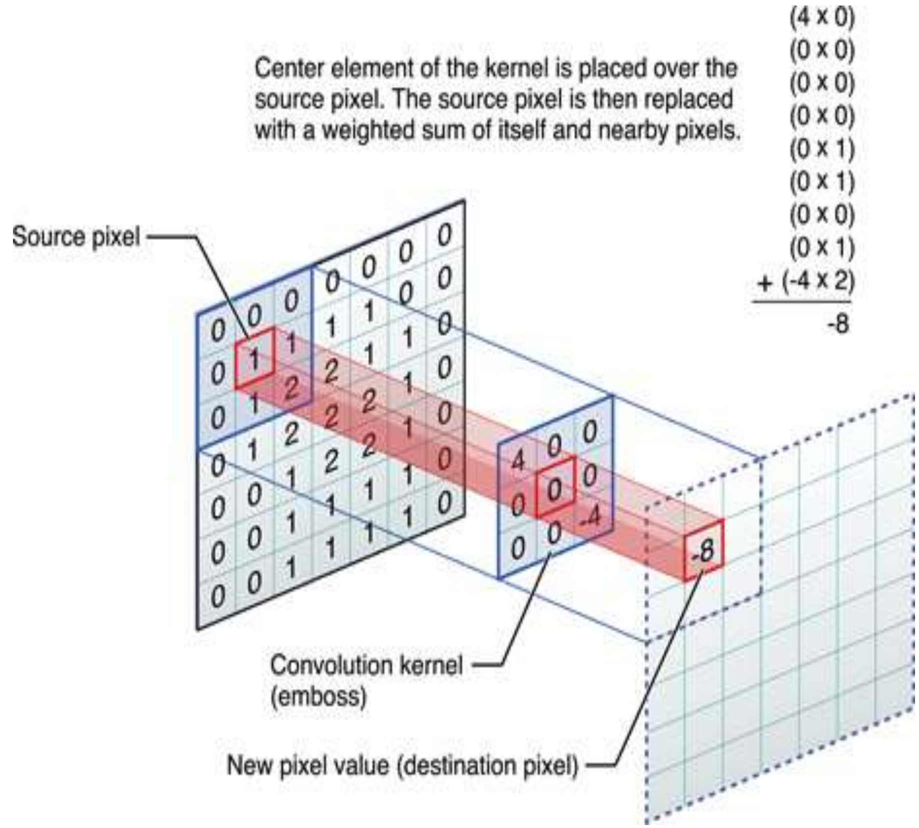
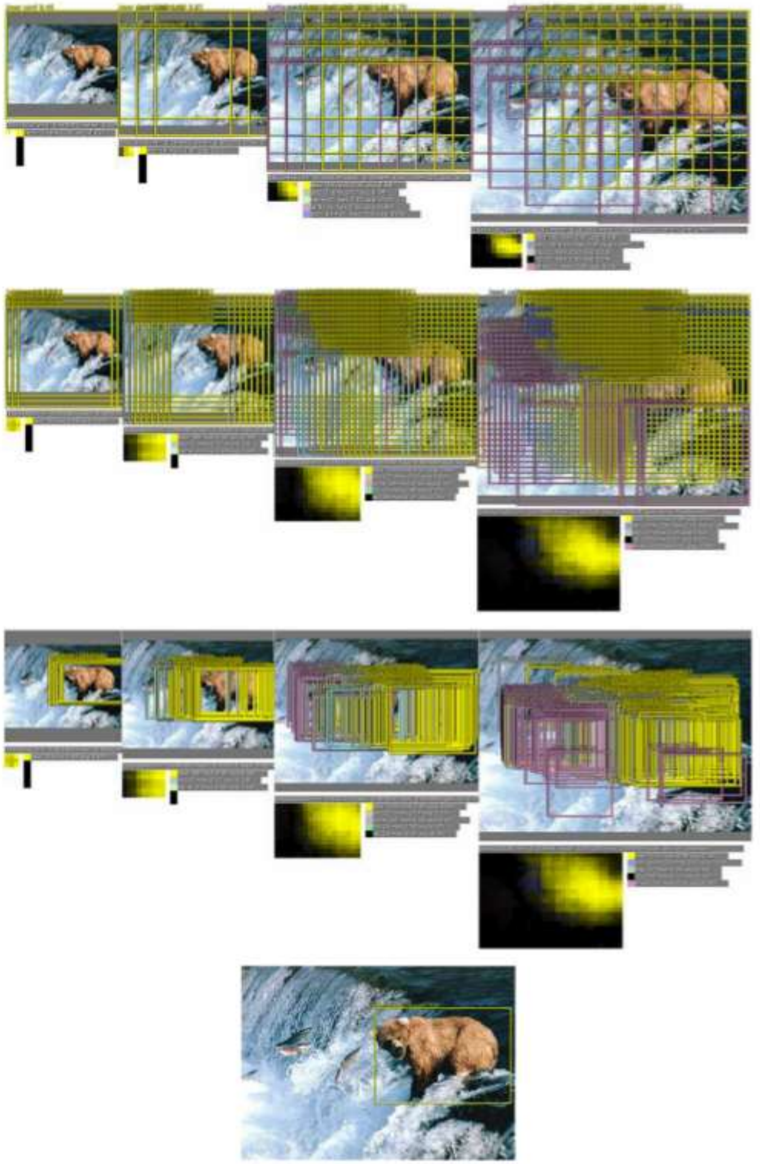


Image from developer.apple.com

Download your own trained network from GitHub!!

Localization

- Start with classification trained network
- Replace classification layer by a regression network
- Train it to predict object bounding boxes at each location and scale.
- Allow results to boost each other by merging bounding boxes
- Rewards bounding box coherence
- more robust than non-max surpression.

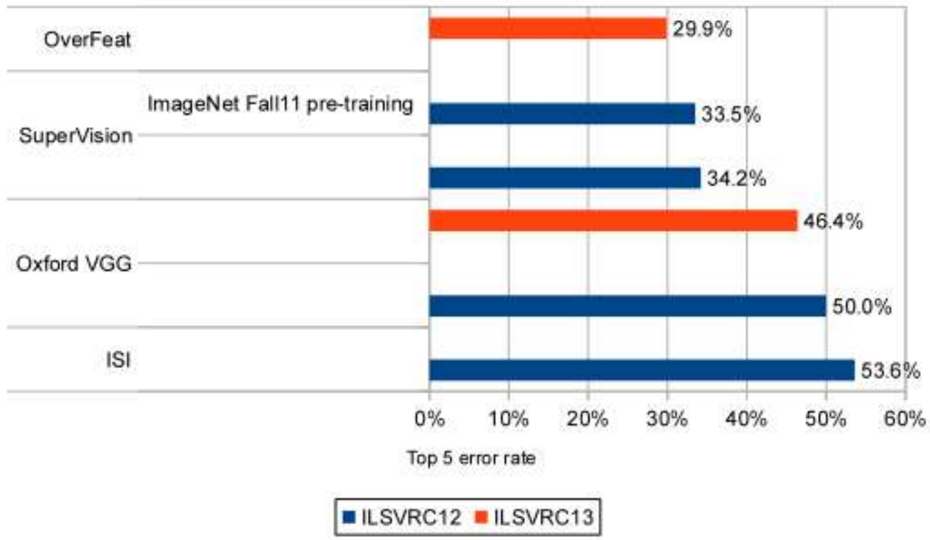


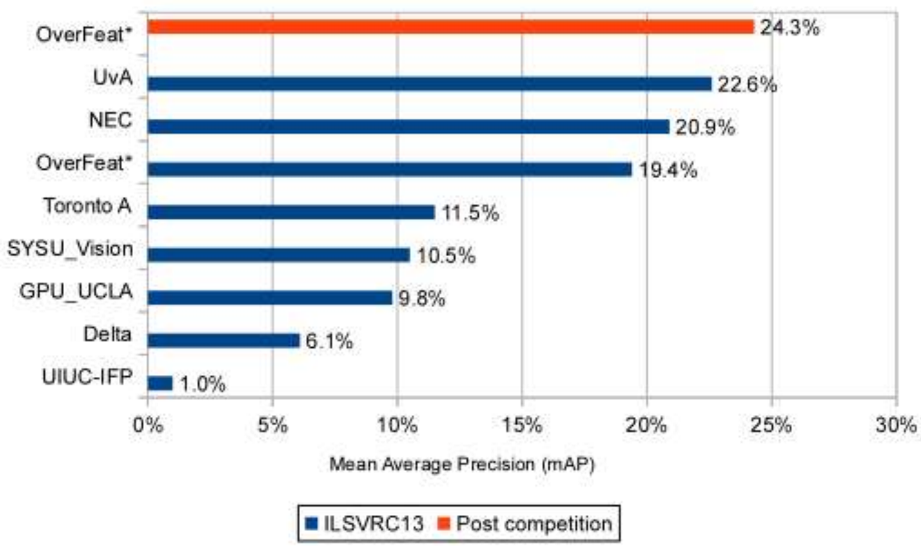
Detection

The main difference to the localization task is the necessety to predict a background class when no object is present.

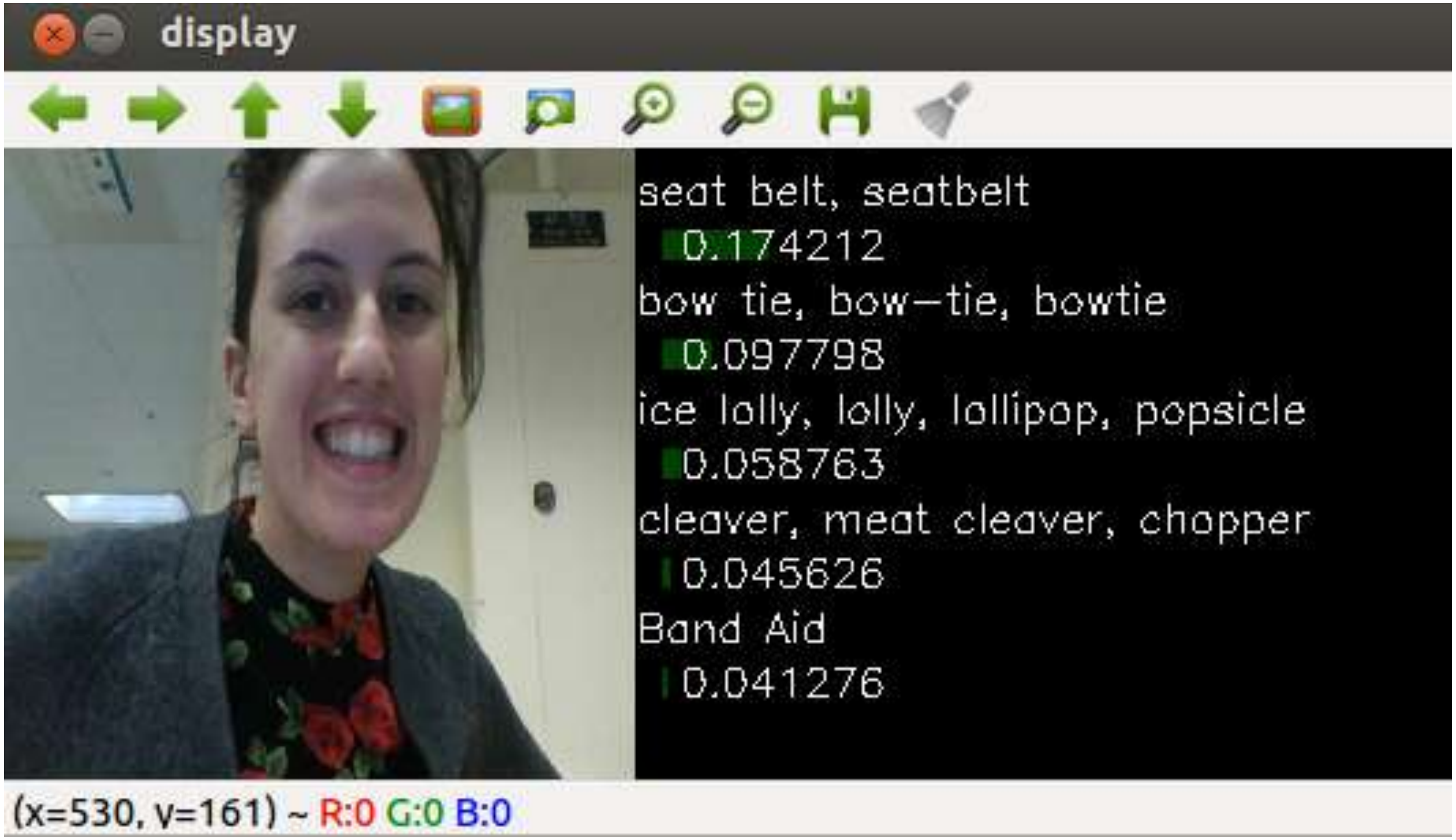
Negative training, by manually selecting negative examples such as random images or the most offensive miss classifications.

Results: ILSVRC 2013: 4th in classification, 1st in localization, 1st in detection





I'm offended!


display
seat belt, seatbelt
0.174212
bow tie, bow-tie, bowtie
0.097798
ice lolly, lolly, lollipop, popsicle
0.058763
cleaver, meat cleaver, chopper
0.045626
Band Aid
0.041276
(x=530, y=161) ~ R:0 G:0 B:0

And it refused to recognize my apple!

Well, at least it can tell a cardigan!