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:

\[ p(\text{label} \mid \text{an input image}) \]
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 Architecture

- 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.
Detail: Neurons

$\text{f}(x) = \tanh(x)$

Very bad (slow to train)

$\text{f}(x) = \max(0, x)$

Very good (quick to train)

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

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

• Local Response Normalization (error rate ↓1.2%)

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

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

**Update rule for weight w:**

\[ v_{i+1} := 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left( \frac{\partial L}{\partial w} \right)_{w_i} \]

\[ w_{i+1} := w_i + v_{i+1} \]

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Top-1 (val)</th>
<th>Top-5 (val)</th>
<th>Top-5 (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>SIFT + FVs [7]</em></td>
<td>1 CNN</td>
<td>38.1%</td>
<td>16.4%</td>
</tr>
<tr>
<td></td>
<td>5 CNNs</td>
<td>39.0%</td>
<td>16.6%</td>
</tr>
<tr>
<td></td>
<td>7 CNNs*</td>
<td>36.7%</td>
<td>15.4%</td>
</tr>
</tbody>
</table>

Table. Comparison of error rates on ILSVRC-2012 validation and test sets.
Qualitative Evaluations - Validation Classification

<table>
<thead>
<tr>
<th>mite</th>
<th>container ship</th>
<th>motor scooter</th>
<th>leopard</th>
</tr>
</thead>
<tbody>
<tr>
<td>mite</td>
<td>container ship</td>
<td>motor scooter</td>
<td>leopard</td>
</tr>
<tr>
<td>black widow</td>
<td>lifeboat</td>
<td>go-kart</td>
<td>jaguar</td>
</tr>
<tr>
<td>cockroach</td>
<td>amphibian</td>
<td>moped</td>
<td>cheetah</td>
</tr>
<tr>
<td>tick</td>
<td>fireboat</td>
<td>bumper car</td>
<td>snow leopard</td>
</tr>
<tr>
<td>starfish</td>
<td>drilling platform</td>
<td>golfcart</td>
<td>Egyptian cat</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>grille</th>
<th>mushroom</th>
<th>cherry</th>
<th>Madagascar cat</th>
</tr>
</thead>
<tbody>
<tr>
<td>convertible</td>
<td>agaric</td>
<td>dalmatian</td>
<td>squirrel monkey</td>
</tr>
<tr>
<td>grille</td>
<td>mushroom</td>
<td>grape</td>
<td>spider monkey</td>
</tr>
<tr>
<td>pickup</td>
<td>jelly fungus</td>
<td>elderberry</td>
<td>titi</td>
</tr>
<tr>
<td>beach wagon</td>
<td>gill fungus</td>
<td>elderberry</td>
<td>indri</td>
</tr>
<tr>
<td>fire engine</td>
<td>dead-man’s-fingers</td>
<td>currant</td>
<td>howler monkey</td>
</tr>
</tbody>
</table>
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 efficient
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 suppression.
Detection

The main difference to the localization task is the necessity 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!
And it refused to recognize my apple!
Well, at least it can tell a cardigan!