Outline

ConvNets as Deep Neural Networks for Vision

1. Neural Nets
2. Deep Convolutional Neural Networks
3. Modern Deep Architectures
   • The big picture
   • AlexNet
   • GoogleNet
   • VGG
   • ResNet
Results 2015: 3.6% top5 error

Result in ILSVRC over the years

Classification

ILSVRC year

Localization

ILSVRC year
ResNet
The deeper, the better

+ Deeper network covers more complex problems
  • Receptive field size ↑
  • Non-linearity ↑

- Training deeper network more difficult because of vanishing/exploding gradients problem

@ Kaiming He ILSVRC & COCO 2015
Deep Neural Network

- Escape from few layers
  - ReLU for solving gradient vanishing problem
  - Dropout ...

Revolution of Depth

<table>
<thead>
<tr>
<th>Year</th>
<th>Architecture</th>
<th>Layers</th>
<th>Top-5 Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILSVRC'10</td>
<td>AlexNet</td>
<td>25.8</td>
<td></td>
</tr>
<tr>
<td>ILSVRC'11</td>
<td>VGG</td>
<td>28.2</td>
<td></td>
</tr>
<tr>
<td>ILSVRC'13</td>
<td>ResNet</td>
<td>11.7</td>
<td></td>
</tr>
<tr>
<td>ILSVRC'14</td>
<td>VGG</td>
<td>7.3</td>
<td></td>
</tr>
<tr>
<td>ILSVRC'14</td>
<td>GoogleNet</td>
<td>6.7</td>
<td></td>
</tr>
<tr>
<td>ILSVRC'15</td>
<td>ResNet</td>
<td>3.57</td>
<td></td>
</tr>
</tbody>
</table>

ImageNet Classification top-5 error (%)
Deep Neural Network

- Escape from 10 layers
  - Normalized initialization
  - Intermediate normalization layers

![Revolution of Depth](chart.png)
Deep Neural Network

- Escape from 100 layers
  - Residual network

![ImageNet Classification top-5 error (%)](chart.png)
Deeper VGG: 56 Plain Network

Plain nets: stacking 3x3 conv layers
- 56-layer net has higher training error and test error than 20-layers net
Deeper VGG:

“Overly deep” plain nets have higher training error
A general phenomenon, observed in many datasets
Residual Network

Naïve solution
If extra layers identity mapping, training error not increase
Residual Network

• Deeper networks maintain the tendency of results
  Features in same level will be almost same
  An amount of changes is fixed
  Adding layers make smaller differences
  Optimal mappings closer to an identity
Residual Network

Plain block

Difficult to make identity mapping because of multiple non-linear layers
Residual Network

Residual block
If identity were optimal, easy to set weights as 0
If optimal mapping is closer to identity, easier to find small fluctuations

-> Appropriate for treating perturbation as keeping a base information
Residual Network

- Difference between an original image and a changed image

Preserving base information

Some Network → residual

can treat perturbation
Residual Network

Deeper ResNets have lower training error
Residual Network

• Residual block
  • Very simple
  • Parameter-free

A naïve residual block

“bottleneck” residual block (for ResNet-50/101/152)
Residual Network

- Shortcuts connections
  - Identity shortcuts
    \[ y = \mathcal{F}(x, \{W_i\}) + x. \]
  - Projection shortcuts
    \[ y = \mathcal{F}(x, \{W_i\}) + W_s x. \]
Network Design

Basic design (VGG-style)
All 3x3 conv (almost)
Spatial size/2 => #filters x2
Batch normalization
Simple design, just deep

Other remarks
No max pooling (almost)
No hidden fc
No dropout
Network Design

**ResNet-152**

Use bottlenecks
ResNet-152 (11.3 billion FLOPs) lower complexity than VGG-16/19 nets (15.3/19.6 billion FLOPs)

About 64M parameters
Results

• Deep Resnets can be trained without difficulties
• Deeper ResNets have lower training error, and also lower test error
Results

• Deep Resnets can be trained “without difficulties”
• Deeper ResNets have lower training error, and also lower test error
Results

• 1\textsuperscript{st} places in all five main tracks in “ILSVRC & COCO 2015 Competitions”
  • ImageNet Classification
  • ImageNet Detection
  • ImageNet Localization
  • COCO Detection
  • COCO Segmentation
Deep ConvNets for image classification

- **ResNeXt**
  - Multi-branch architecture

---

Saining Xie, Ross Girshick, Piotr Dollàr, Zhuowen Tu and Kaiming He
Aggregated Residual Transformations for Deep Neural Networks.
FractalNet: Ultra-Deep Neural Networks without Residuals

Comments from paper: “Our experiments demonstrate that residual representation is not fundamental to the success of extremely deep convolutional neural networks. A fractal design achieves an error rate of 22.85% on CIFAR-100, matching the state-of-the-art held by residual networks.”
Exploring type of deep modules in Neural Nets

*Neural Architecture Search*
Conclusion

• ResNet: currently the best archi for large scale image classification
• Not yet consensus about the design of the Net (cf. FractalNet), Neural Architecture Search

• Fully Convolutional Net (FCN) very interesting option

Beyond classification!
COMPLEMENTS
Batch normalization

**Input:** Values of $x$ over a mini-batch: $\mathcal{B} = \{x_1 \ldots m\}$; Parameters to be learned: $\gamma$, $\beta$  

**Output:** $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

- Improves gradient flow through the network
- Allows higher learning rates
- Reduces the strong dependence on initialization
- Reduces need for dropout

Un-normalization!! Re-compute and apply the optimal scaling and bias for each neuron! Learn $\gamma$ and $\beta$ (same dims as $\mu$ and $\sigma^2$). It can (should?) learn the identity mapping!
Where to do BatchNorm?

BatchNorm is just a linear scale/bias layer in the limit of large batch sizes ($\mu, \sigma^2$)
BatchNorm is just a linear scale/bias layer in the limit of large batch sizes ($\mu, \sigma^2$).

**Where to do BatchNorm?**

?? Linear weight layer should already have optimal scale/bias in its output
BatchNorm is just a linear scale/bias layer in the limit of large batch sizes ($\mu, \sigma^2$).

Where to do BatchNorm?

?? Any bias/scaling by the batch norm layer should be over-rulled by the second linear weight layer.
Multitask learning / auxiliary loss in GoogLeNet