Deconvolutions in Convolutional Neural Networks

Bohyung Han
bhhan@postech.ac.kr
Computer Vision Lab.

Overview

• Convolutional Neural Networks (CNNs)
• Deconvolutions in CNNs
• Applications
  ▪ Network visualization and analysis
  ▪ Object generation
  ▪ Semantic segmentation

• Disclaimer
  ▪ This talk may not be a comprehensive presentation about deconvolutions in convolutional neural networks.
  ▪ It is limited to computer vision applications.

Convolutional Neural Networks

• Feed-forward network
  ▪ Convolution
  ▪ Non-linearity: Rectified Linear Unit (ReLU)
  ▪ Pooling: (typically) local maximum
• Supervised learning
• Representation learning

**Convolutional Neural Network (CNN)**

CNN had not shown impressive performance.

- **Reasons for failure**
  - Insufficient training data
  - Slow convergence
    - Bad activation function: Sigmoid function
    - Too many parameters
    - Limited computing resources
  - Lack of theory: needed to rely on trials-and-errors

**CNN recently draws a lot of attention due to its great success.**

- **Reasons for recent success**
  - Availability of larger training datasets, e.g., ImageNet
  - Powerful GPUs
  - Better model regularization strategy such as dropout
  - Simple activation function: ReLU

---

**AlexNet** [Krizhevsky12]

- **Winner of ILSVRC 2012 challenge**
  - Same architecture with [Lecun89] but trained with larger data
  - Bigger model: 7 hidden layers, 650K neurons, 60 million parameters
  - Trained on 2 GPUs for a week
  - Training with error back-propagation using stochastic gradient method


---

**Main Reasons for Success**

- **Improving training speed**
  - New activation function: Rectified Linear Unit (ReLU)
    
    \[
    f(x) = \max(0, x)
    \]

  - Optimization techniques
    - Use of high-performance GPUs
    - Stochastic gradient method with mini-batches
    - Optimized library, e.g., Caffe

---

**AlexNet** [Krizhevsky12]

- **ILSVRC-2012 results**

  ![Error (5 predictions)](chart)

  - Supervision: 0.15
  - IS: 0.14
  - OXORD-235G: 0.17
  - KRC/IRRA: 0.18
  - U Amsterdam: 0.22
  - LEAR-IAR: 0.48

  **Runner-up**
  - Top-5 error rate: 26.172%

  **AlexNet**
  - Top-5 error rate: 16.422%
Main Reasons for Success

• Dropout: reducing overfitting problem
  - Setting to zero the output of each hidden neuron with probability 0.5
  - Employed in the first two fully-connected layers
  - Simulating ensemble learning without additional models
    - Every time an input is presented, the neural network samples a different architecture.
    - But, all these architectures share weights.
  - At test time, we use all the neurons but multiply their outputs by 0.5.

A hidden layer’s activity on a given training image

A hidden unit turned off by dropout

A hidden unit unchanged

Other CNNs for Classification

• Very Deep ConvNet by VGG [Simonyan15]
  - Smaller filters: 3x3
    - More non-linearity
    - Less parameters to learn: ~140 millions
  - A significant performance improvement with 16–19 layers
  - Generalization to other datasets
  - The first place for localization and the second place for classification in ILSVRC 2014

Other CNNs for Classification

• GoogLeNet [Szegedy15]
  - Network in network
  - Hebbian principle: Neurons that fire together, wire together
  - Inception modules
  - The winner of ILSVRC 2014 classification task

Deconvolution Networks


Deconvolutions in Convolutional Neural Networks

By Prof. Bohyung Han

Deconvolution Networks

- Generative convolutional neural network
- Advantages
  - Capable of structural prediction
    - Segmentation
    - Matching
    - Object generation
    - Others
  - More general than classification: extending applicability of CNNs
- Challenges
  - More parameters
    - Difficult to train
    - Requires more training data, which may need heavy human efforts
  - Task specific network: typically not transferrable

Operations in Deconvolution Network

- Unpooling
  - Place activations to pooled location
  - Preserve structure of activations
- Deconvolution
  - The size of output layer is larger than that of input.
  - Densify sparse activations
  - Conceptually similar to convolution
  - Bases to reconstruct shape
- ReLU
  - Same with convolution network

Deconvolution Papers in Computer Vision

- Visualization and analysis of CNNs
  - M. Zeiler, G. W. Taylor and R. Fergus, Adaptive Deconvolutional Networks for Mid and High Level Feature Learning, ICCV 2011
  - M. Zeiler and R. Fergus, Visualizing and Understanding Convolutional Networks, ECCV 2014
- Object generation
- Semantic segmentation

Analysis of Convolutional Neural Networks
**Questions in CNNs**

- Despite encouraging progress
  - There is still little insight into the internal operation and behavior of these complex models
  - How CNNs achieve such good performance
  
  *Without clear understanding of CNNs, the development of better models is reduced to trial-and-error.*

- Visualization of CNNs
  - Reveals the input stimuli that excite individual feature maps at any layer in the model.
  - Allows us to observe the evolution of features during training and to diagnose potential problems with the model.

**Visualization with Deconvnet**

- **Unpooling**
  - Approximate inverse: Max pooling operation is non-invertible
  - Switch variables: recording the locations of maxima

- **Rectification by ReLU**
  - Ensuring the positivity of feature maps

- **Filtering**
  - Using transposed filters as other autoencoder models
  - Flipping each filter vertically and horizontally, in practice

---

**Visualizing CNNs**

- **Main idea**
  - Mapping activations at high layers back to the input pixel space
  - Showing what input patterns originally caused a given activation in the feature maps

- **Deconvnet**
  - Originally proposed as a way of unsupervised learning method [Zeiler11]
  - Used as a probe: no inference, no learning

- **Same operations as CNNs, but in reverse**
  - Unpool feature maps
  - Convolve unpool maps

---


Training Details

- Similar architecture to AlexNet
  - Smaller filter in the 1st layer and smaller stride
  - Determined through visualization of trained model
  - Dropout with a rate of 0.5 for the fully connected layers
- Data and optimization
  - 10 different sub-crops of size 224x224 from 256x256 image
  - Stochastic gradient descent with a mini-batch size of 128
Feature Invariance: Scale

Layer 1

Layer 7

Probability of true label

Feature Invariance: Rotation

Layer 1

Layer 7

Probability of true label

Occlusion Sensitivity

Architecture Selection

- Observations from AlexNet
  - The 1st layer filters
    - A mix of extremely high and low frequency information
    - Little coverage of the mid frequencies.
  - The 2nd layer visualization: aliasing artifacts caused by the large stride 4 used in the 1st layer convolutions.
Architecture Selection

• Model revisions
  ▪ Reducing the 1st layer filter size from 11x11 to 7x7
  ▪ Making the stride of the convolution 2, rather than 4.

These updates lead to classification performance improvement.

Performance in ILSVRC 2012 Dataset

<table>
<thead>
<tr>
<th>Error %</th>
<th>Val Top-1</th>
<th>Val Top-5</th>
<th>Test Top-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gunji et al. [12]</td>
<td>-</td>
<td>-</td>
<td>26.2</td>
</tr>
<tr>
<td>DeCAF [7]</td>
<td>-</td>
<td>-</td>
<td>19.2</td>
</tr>
<tr>
<td>Krizhevsky et al. [18], 1 convnet</td>
<td>40.7</td>
<td>18.2</td>
<td>——</td>
</tr>
<tr>
<td>Krizhevsky et al. [18], 5 convnets</td>
<td>38.1</td>
<td>16.4</td>
<td>16.4</td>
</tr>
<tr>
<td>Krizhevsky et al. *[18], 1 convnets</td>
<td>39.0</td>
<td>16.6</td>
<td>——</td>
</tr>
<tr>
<td>Krizhevsky et al. *[18], 7 convnets</td>
<td>36.7</td>
<td>15.4</td>
<td>15.3</td>
</tr>
<tr>
<td>Our replication of Krizhevsky et al., 1 convnet</td>
<td>40.5</td>
<td>18.1</td>
<td>——</td>
</tr>
<tr>
<td>1 convnet as per Fig. 3</td>
<td>38.4</td>
<td>16.5</td>
<td>——</td>
</tr>
<tr>
<td>5 convnets as per Fig. 3 - (a)</td>
<td>36.7</td>
<td>15.3</td>
<td>15.3</td>
</tr>
<tr>
<td>1 convnet as per Fig. 3 but with layers 3,4,5: 512,1024,512 maps – (b)</td>
<td>37.5</td>
<td>16.0</td>
<td>16.1</td>
</tr>
<tr>
<td>6 convnets, (a) &amp; (b) combined</td>
<td>36.0</td>
<td>14.7</td>
<td>14.8</td>
</tr>
</tbody>
</table>

ILSVRC 2013 Results

Object Generation
Discriminative vs. Generative CNN

- **Discriminative CNN**

  ![Image of chair]

  

  

- **Generative CNN**

  ![Image of chair]

  

  

Goal

- Generate an object based on high-level inputs such as
  - Class
  - Orientation with respect to camera
  - Additional parameters
    - Rotation, translation, zoom
    - Stretching horizontally or vertically
    - Hue, saturation, brightness

Contribution

- Knowledge transfer
  - Given limited number of viewpoints of an object, the network can use the knowledge learned from other similar objects to infer remaining viewpoints.

- Interpolation between different objects
  - Generative CNN learns the manifold of chairs.

Network Architecture

\[ g = u \circ h \]

32M parameters altogether

## Operations

- **Unpooling:** 2x2

![Fixed location unpooling](image)

- **Deconvolution:** 5x5

- **ReLU**

## Data

- **Using 3D chair model dataset** ([Aubry14])
  - Original dataset: 1393 chair models, 62 viewpoints, 31 azimuth angles, 2 elevation angles
  - Sanitized version: 809 models, tight cropping, resizing to 128x128

### Notation

- $c$: class label
- $v$: viewpoint
- $\theta$: additional parameters
- $D = \{(c^1, v^1, \theta^1), (c^2, v^2, \theta^2), \ldots, (c^N, v^N, \theta^N)\}$
  - $c$: class label
  - $v$: viewpoint
  - $\theta$: additional parameters
- $O = \{(x^1, s^1), (x^2, s^2), \ldots, (x^N, s^N)\}$
  - $x$: target RGB output image
  - $s$: segmentation mask


## Training

- **Objective function**
  - Minimizing the Euclidean error in 2D of reconstructing the segmented-out chair image and the segmentation mask

\[
\min_w \sum_{i=1}^{N} \left\| u_{\text{RGB}} \left( h(c^i, v^i, \theta^i) \right) - T_{\theta^i}(x^i \cdot s^i) \right\|_2^2 + \left\| u_{\text{seg}} \left( h(c^i, v^i, \theta^i) \right) - T_{\theta^i}s^i \right\|_2^2
\]

- **Optimization**

  - Stochastic gradient descent with momentum of 0.9

  - Learning rate
    - 0.0002 for the first 500 epochs
    - Dividing by 2 after every 100 epoch

  - Orthogonal matrix initialization ([Saxe14])


## Network Capacity

- **Translation**
- **Rotation**
- **Zoom**
- **Stretch**
- **Saturation**
- **Brightness**
- **Color**
Learned Filters

- Visualization of uconv-3 layer filters in 128x128 network

RGB stream

Segmentation stream

- Facts and observations
  - The final output at each position is generated from a linear combination of these filters.
  - They include edges and blobs.

Single Unit Activation

- Images generated from single unit activations

Hidden Layer Analysis

- Zoom neuron
  - Increasing activation of the “zoom neuron” found in FC-4 feature map

- Spatial mask
  - Chairs generated from spatially masked 8x8 FC-5 feature map

Interpolation between Angles

With knowledge transfer

Without knowledge transfer
Summary

- Supervised Training of CNN can also be used to generate images.
- Generative network does not merely learn, but also generalizes well.
- The proposed network is capable of processing very different inputs using the same standard layers.

Semantic Segmentation using CNN

- Image classification
- Semantic segmentation
  - Given an input image, obtain pixel-wise segmentation mask using a deep Convolutional Neural Network (CNN)
Fully Convolutional Network (FCN)

- Converting fully connected layers to convolution layers
  - Each fully connected layer is interpreted as a convolution with a large spatial filter that covers entire input field

Fully connected layers
Convolution layers
For the larger Input field

Deconvolution Filter

- Bilinear interpolation filter
  - Same filter for every class
  - There is no learning!
  - Not a real deconvolution
- How does this deconvolution work?
  - Deconvolution filter is fixed.
  - Fining-tuning convolution layers of the network with segmentation ground-truth.

64x64 bilinear interpolation

FCN for Semantic Segmentation

- Network architecture[^Long15]
  - End-to-End CNN architecture for semantic segmentation
  - Convert fully connected layers to convolutional layers


Skip Architecture

- Ensemble of three different scales

More semantic

Finer
Limitations of FCN-based Semantic Segmentation

- Coarse output score map
  - A single bilinear filter should handle the variations in all kinds of object classes.
  - Difficult to capture detailed structure of objects in image
- Fixed size receptive field
  - Unable to handle multiple scales
  - Difficult to delineate too small or large objects compared to the size of receptive field
- Noisy predictions due to skip architecture
  - Trade off between details and noises
  - Minor quantitative performance improvement

Results and Limitations

DeepLab-CRF

- A variation of FCN-based semantic segmentation [Chen15]
  - Hole algorithm: denser output production from 16x16 to 39x39
  - Post processing based on Conditional Random Field (CRF)
- Characteristics
  - No skip architecture in basic model
  - Simple output score map upscaling without deconvolution layer

Deconvolutions in Convolutional Neural Networks

By Prof. Bohyung Han

CRF-RNN

- End-to-end learning CRF using recurrent neural network

DeconvNet for Semantic Segmentation

- Learning a deconvolution network
  - Conceptually more reasonable
  - Better to identify fine structures of objects
  - Designed to generate outputs from larger solution space
  - Capable of predicting dense output scores
  - Difficult to learn: memory intensive

DeconvNet for Semantic Segmentation

- Instance-wise training and prediction
  - Easy data augmentation
  - Reducing solution space
  - Inference on object proposals, then aggregation
  - Labeling objects in multiple scales

Why Not Trying Deconvolution?

- Too many parameters
  - Approximately 252M parameters in total
    - Involves large output space
    - Twice as many as VGG 16-layer net [Simonyan15]
  - Potentially requires a large dataset
    - Difficult to obtain annotated data for semantic segmentation
  - Needs large GPU memory

Is it really difficult to train deconvolution network for semantic segmentation?


Training Strategy

- Data augmentation
  - Training per proposal: also reduces the size of output space
  - Random cropping and horizontal flipping

- Progressive training
  - First stage
    - Training with object ground-truth bounding boxes: 0.2M examples
    - Binary annotation
  - Second stage
    - Training with real object proposals: 2.7M examples
    - Annotation of all available labels
    - This approach makes the network generalize better.

Challenge in Training

- Internal-covariate-shift
  - Input distributions in each layer change over iteration during training as the parameters of its previous layers are updated.
  - Problematic in optimizing very deep networks since the changes in distribution are amplified through propagation across layers

- Batch Normalization\[^{1}\]
  - Normalize each input channel in a layer to standard Gaussian distribution
  - Prevent drastic changes of input distribution in upper layers
  - A batch normalization layer is added to the output of every convolutional and deconvolutional layer

\[^{1}\] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. ICML 2015

Training Details

- Initialization
  - Convolution network: VGG 16-layer net trained on ImageNet
  - Deconvolution network: zero mean Gaussians

- Optimization
  - Learning rates
    - Initial values: 0.01
    - Reduce learning rate in an order of magnitude whenever validation accuracy does not improve
  - Mini-batch size: 64
  - Convergence
    - 20K and 40K SGD iterations for the first and second stage training, respectively
    - Takes approximately 2 and 4 days in the stages.

\[^{1}\] Nvidia GeForce GTX Titan X
  - Maxwell GPU architecture
  - 3072 CUDA cores
  - 1000MHz base clock / 1075MHz boost clock
  - 12G memory
Deconvolutions in Convolutional Neural Networks
By Prof. Bohyung Han

**How Deconvolution Network Works?**

- Visualization of activations

   ![Deconv: 14x14](image1)
   ![Unpool: 28x28](image2)
   ![Deconv: 28x28](image3)
   ![Unpool: 56x56](image4)
   ![Deconv: 56x56](image5)
   ![Unpool: 112x112](image6)
   ![Deconv: 112x112](image7)

**Inference**

- Instance-wise prediction

  ![1. Input Image](image8)
  ![2. Object proposals](image9)
  ![3. Prediction and aggregation](image10)
  ![4. Results](image11)

  - Inference on object proposals
    - Each class corresponds to one of the channels in the output layer.
    - Label of a pixel is given by max operation over all channels.
  - Aggregation of object proposals
    - Max operation with all proposals overlapping on each pixel
    - Number of proposals: not sensitive to accuracy
    - 50 proposals for evaluation

---

**How Deconvolution Network Works?**

- Would FCN work equivalently if applied to a proposal?

   ![Input](image12)
   ![FCN8s](image13)
   ![DeconvNet](image14)

**Inference**

- Handling multi-scale objects naturally

   ![Number of proposals](image15)
Deconvolutions in Convolutional Neural Networks
By Prof. Bohyung Han

Results

PASCAL VOC 2012 Leaderboard

<table>
<thead>
<tr>
<th>Dataset</th>
<th>max</th>
<th>aper</th>
<th>airplane</th>
<th>bicyl</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>dining table</th>
<th>horse</th>
<th>motorbike</th>
<th>person</th>
<th>pedial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adelaide_Coon, CIN, CR, COG [1]</td>
<td>76.4</td>
<td>95.8</td>
<td>55.3</td>
<td>20.9</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>PASCAL VOG [2]</td>
<td>75.2</td>
<td>92.5</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>Oxford_TVC, CR, RNN, COG [3]</td>
<td>74.7</td>
<td>95.0</td>
<td>55.3</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>Adelaide_Coon, CIN, CR, VOC [4]</td>
<td>73.5</td>
<td>89.9</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-CRF-COCO, Largefov [5]</td>
<td>72.7</td>
<td>89.1</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>POSTECH, EDefense, CIN, CR, VOC [6]</td>
<td>72.7</td>
<td>89.1</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>Oxford_TVC, CR, RNN, COG [3]</td>
<td>72.0</td>
<td>87.2</td>
<td>35.9</td>
<td>78.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-MSC-CRF, Largefov [5]</td>
<td>71.6</td>
<td>84.4</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>PASCAL VOG [2]</td>
<td>71.0</td>
<td>86.4</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-CRF-COCO, Largefov [5]</td>
<td>70.4</td>
<td>85.3</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-CRF-Largefov [5]</td>
<td>70.4</td>
<td>85.3</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>TTI Testnet [7]</td>
<td>68.5</td>
<td>85.4</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-CRF-MSC [5]</td>
<td>67.3</td>
<td>84.4</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>DeepLab-CRF-Largefov [5]</td>
<td>66.4</td>
<td>84.4</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>CRF, RNN [3]</td>
<td>65.3</td>
<td>83.9</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>TTI Testnet [7]</td>
<td>64.4</td>
<td>83.9</td>
<td>55.9</td>
<td>18.2</td>
<td>68.1</td>
<td>73.5</td>
<td>59.3</td>
<td>87.1</td>
<td>39.1</td>
<td>84.5</td>
<td>64.1</td>
<td>84.0</td>
<td>87.4</td>
<td>65.7</td>
<td>84.7</td>
<td>63.2</td>
<td>81.3</td>
</tr>
<tr>
<td>ResNet [8]</td>
<td>62.6</td>
<td>82.7</td>
<td>51.9</td>
<td>15.0</td>
<td>60.1</td>
<td>71.8</td>
<td>51.9</td>
<td>71.8</td>
<td>31.9</td>
<td>66.8</td>
<td>60.6</td>
<td>71.8</td>
<td>63.0</td>
<td>74.9</td>
<td>63.0</td>
<td>74.9</td>
<td>63.0</td>
</tr>
<tr>
<td>CRF, RNN [3]</td>
<td>62.7</td>
<td>82.7</td>
<td>51.9</td>
<td>15.0</td>
<td>60.1</td>
<td>71.8</td>
<td>51.9</td>
<td>71.8</td>
<td>31.9</td>
<td>66.8</td>
<td>60.6</td>
<td>71.8</td>
<td>63.0</td>
<td>74.9</td>
<td>63.0</td>
<td>74.9</td>
<td>63.0</td>
</tr>
</tbody>
</table>

Contribution

• Confirmation of some conjectures
  ▪ Deconvolution network is conceptually reasonable.
  ▪ Learning a deep deconvolution network is a feasible option for semantic segmentation.

• Presenting a few critical training strategies
  ▪ Data augmentation
  ▪ Multi-stage training
  ▪ Batch normalization

• Very neat formulation

• Good performance
  ▪ Best in all algorithms trained on PASCAL VOC dataset
  ▪ The 3rd overall
Concluding Remark

Deconvolutions in CNNs

- Useful for structured predictions
  - 2D/3D object generation
  - Semantic segmentation
  - Human pose estimation
  - Visual tracking
  - ...

- More parameters but trainable
- Having a lot of potential and applications