# 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.



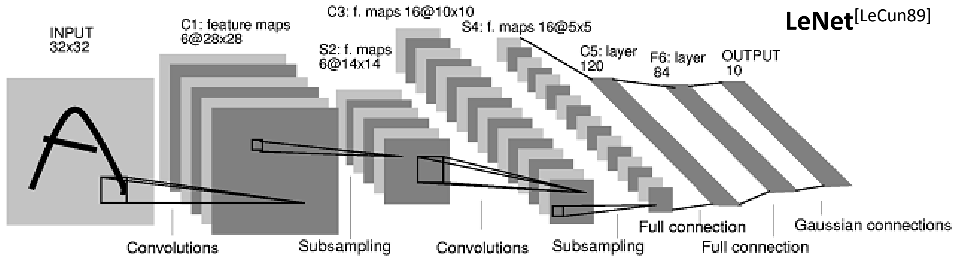
## Convolutional Neural Networks



## Convolutional Neural Network (CNN)

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

**LeNet**[LeCun89]




[Lecun89] Y. LeCun et al.: **Handwritten Digit Recognition with a Back-Propagation Network**. NIPS 1989

# Convolutional Neural Network (CNN)

**CNN had not shown impressive performance.**

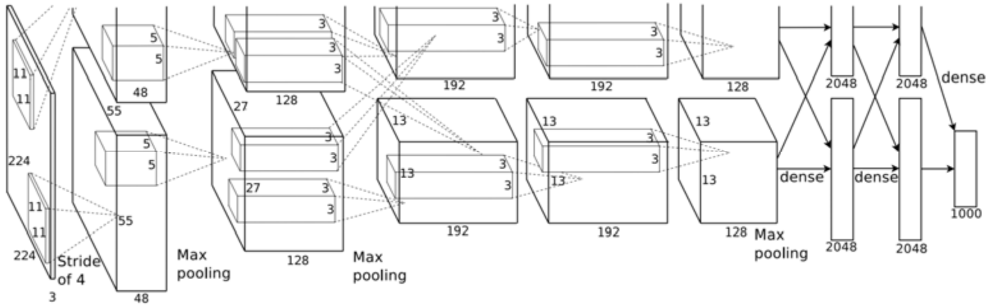
- Reasons for failure
  - Insufficient training data
  - Slow convergence
    - Bad activation function: Sigmoid function
    - Too many parameters
    - Limited computing resources
  - Lack of theory: needed to rely on trials-and-errors

**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]



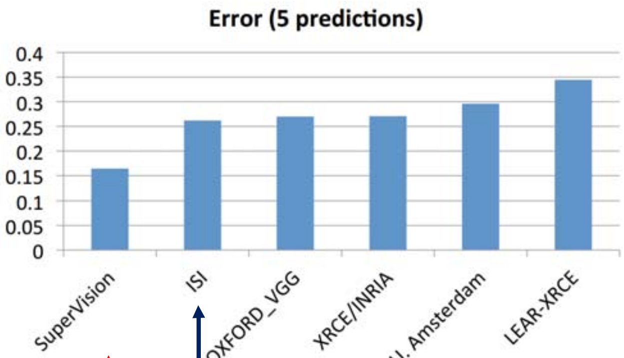

- Winner of ILSVRC 2012 challenge
  - Same architecture with [Lecun89] but trained with larger data
  - Bigger model: 7 hidden layers, 650K neurons, 60 million parameters
  - Trained on 2 GPUs for a week
  - Training with error back-propagation using stochastic gradient method

[Krizhevsky12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, **ImageNet Classification with Deep Convolutional Neural Networks**, NIPS 2012



# AlexNet[Krizhevsky12]

- ILSVRC-2012 results




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

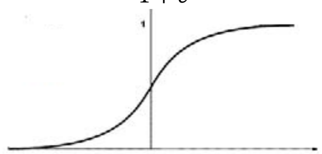
**AlexNet**
**Top-5 error rate : 16.422%**



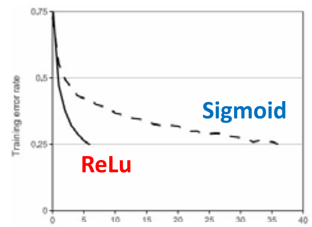
# Main Reasons for Success

- Improving training speed
  - New activation function: Rectified Linear Unit (ReLU)

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

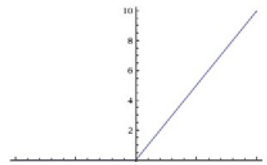


**Sigmoid function**




**Convergence rates**

$$f(x) = \max(0, x)$$



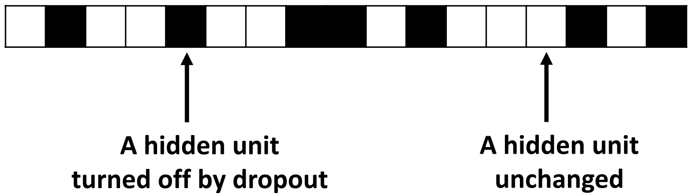
**ReLU**

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

# 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**






# Other CNNs for Classification

- Very Deep ConvNet by VGG[Simonyan15]

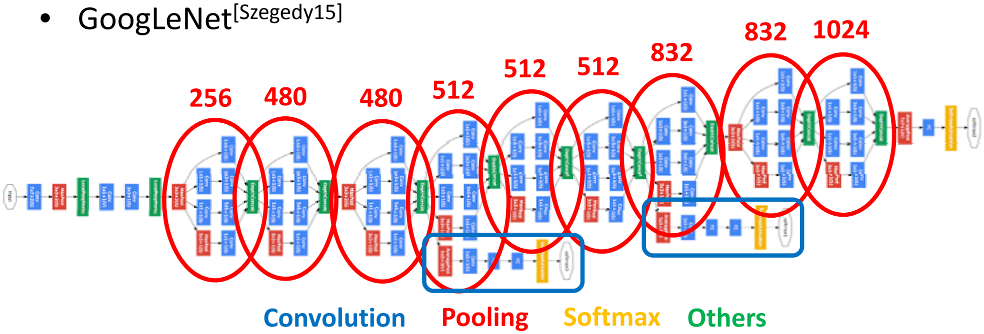

  - Smaller filters: 3x3
    - More non-linearity
    - Less parameters to learn: ~140 millions
  - A significant performance improvement with 16–19 layers
  - Generalization to other datasets
  - The first place for localization and the second place for classification in ILSVRC 2014

[Simonyan15] K. Simonyan, A. Zisserman: **Very Deep Convolutional Networks for Large-Scale Image Recognition**, ICLR 2015



# 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

[Szegedy15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich: **Going deeper with convolutions**. CVPR 2015



# Deconvolution Networks

# Deconvolution Networks

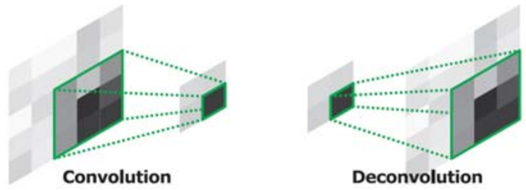
- Generative convolutional neural network
- Advantages
  - Capable of structural prediction
    - Segmentation
    - Matching
    - Object generation
    - Others
  - More general than classification: extending applicability of CNNs
- Challenges
  - More parameters
    - Difficult to train
    - Requires 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
  - A. Dosovitskiy, J. T. Springenberg and T. Brox. **Learning to generate chairs with convolutional neural networks**. CVPR 2015
- Semantic segmentation
  - J. Long, E. Shelhamer, and T. Darrell, **Fully Convolutional Network for Semantic Segmentation**. CVPR 2015
  - H. Noh, S. Hong, and B Han, **Learning Deconvolution Network for Semantic Segmentation**, arXiv:1505.04366, 2015
  - S. Hong, H. Noh, and B. Han, **Decoupled Deep Neural Network for Semi-supervised Semantic Segmentation**, arXiv:1506.04924, 2015



# 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

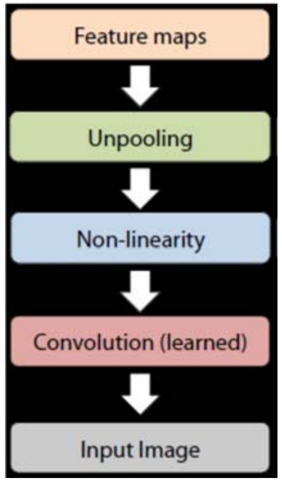
  **Without clear understanding of CNNs, the development of better models is reduced to trial-and-error.**

- 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



# 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 unpooled maps

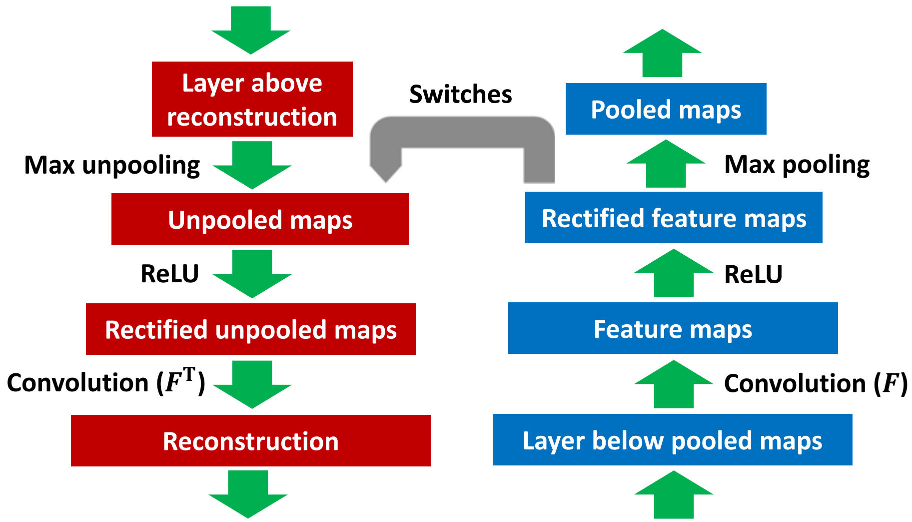



[Zeiler11] M. Zeiler, G. Taylor, and R. Fergus: **Adaptive Deconvolutional Networks for Mid and High Level Feature Learning**. ICCV 2011

[Zeiler14] M. Zeiler and R. Fergus: **Visualizing and Understanding Convolutional Networks.** ECCV 2014



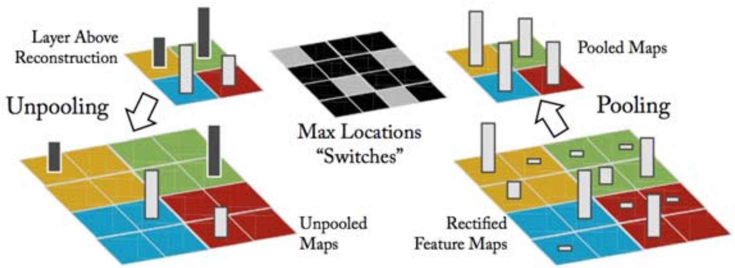
# Visualization with Deconvnet



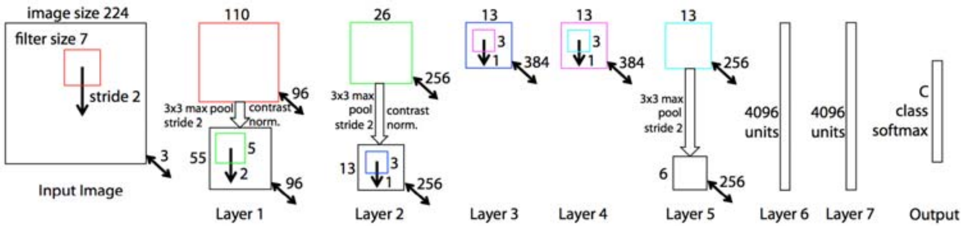



# Visualization with Deconvnet

- 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



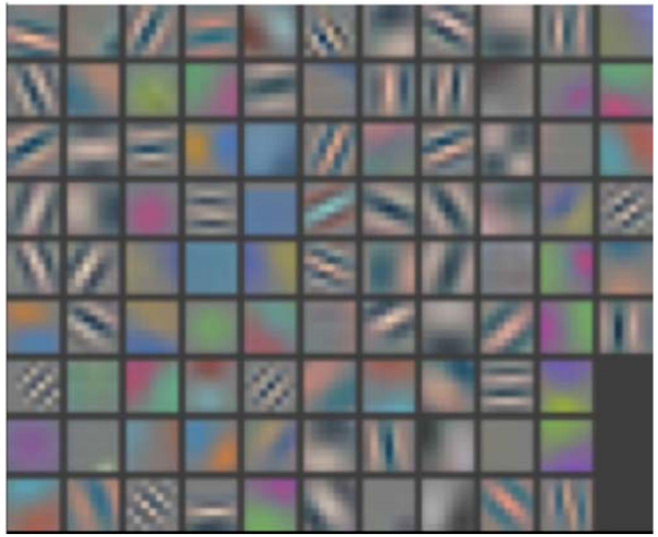
# Training Details




- 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

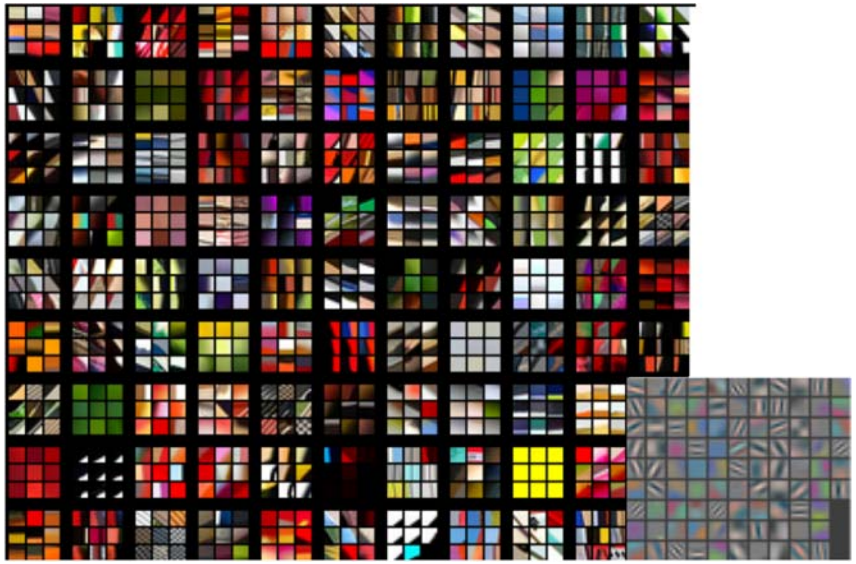


# Layer1 Filters



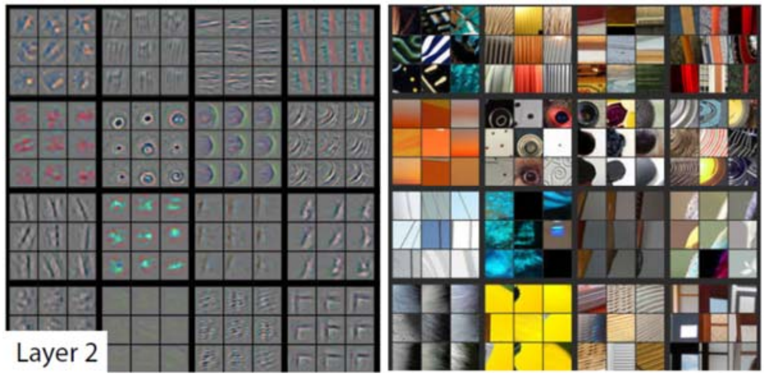


# Layer1: Top 9 Patches

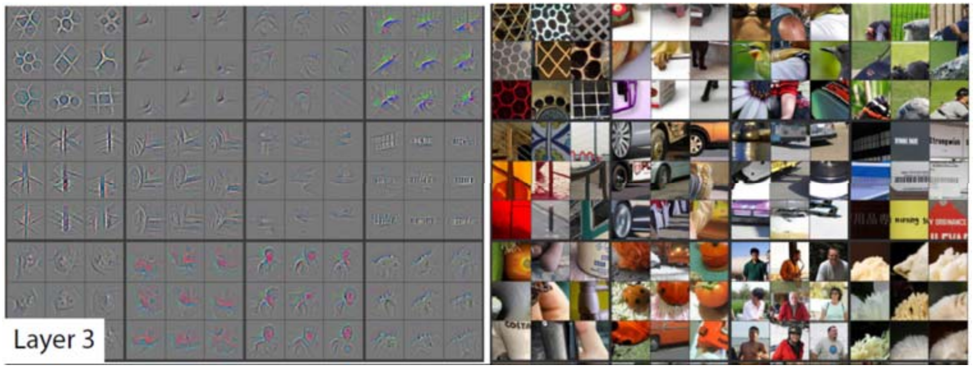




# Top 9 Activations

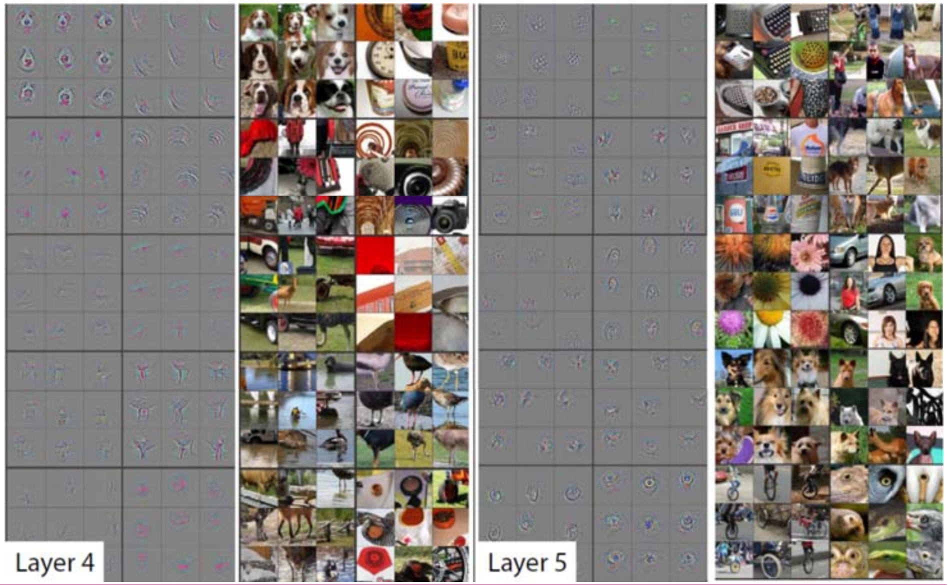


# Top 9 Activations



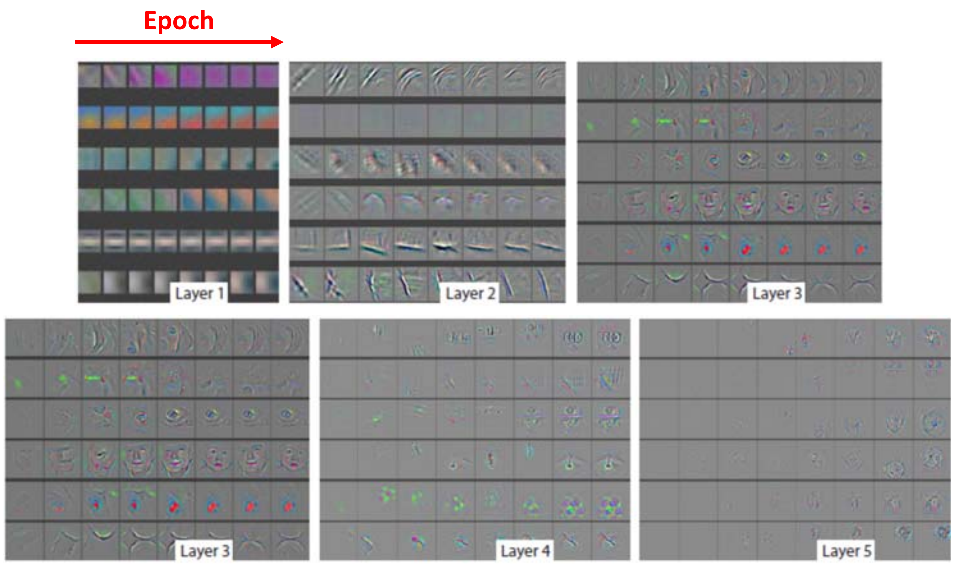



# Top 9 Activations



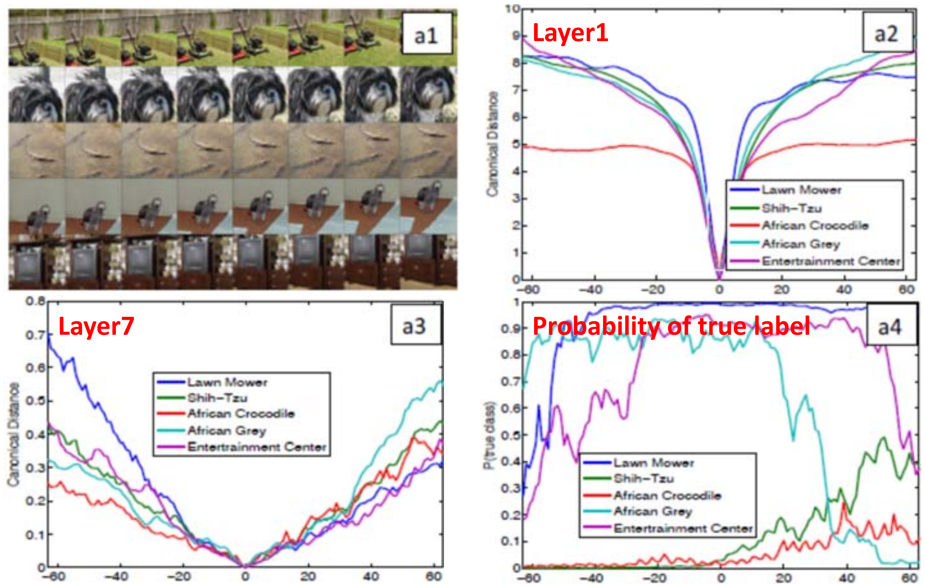



# Feature Evolution during Training

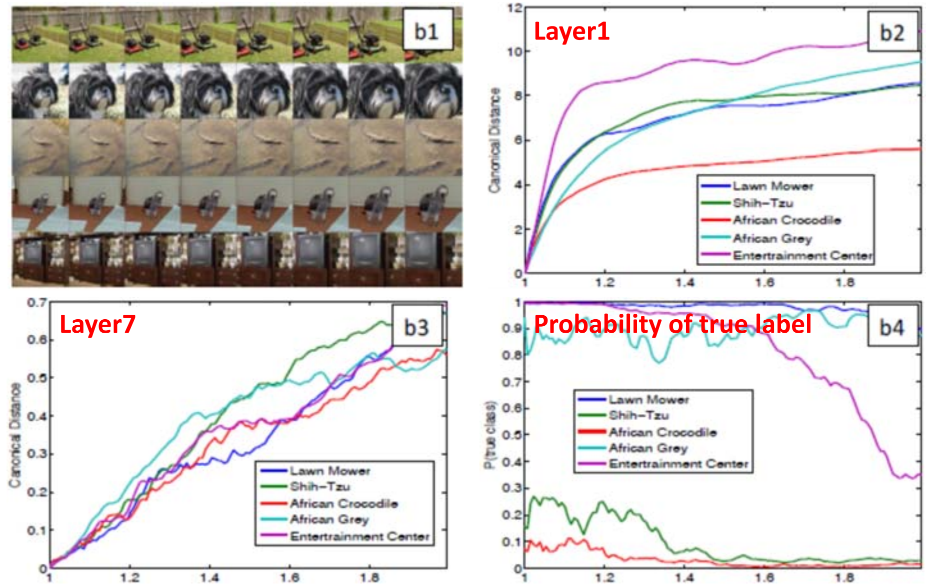





# Feature Invariance: Translation

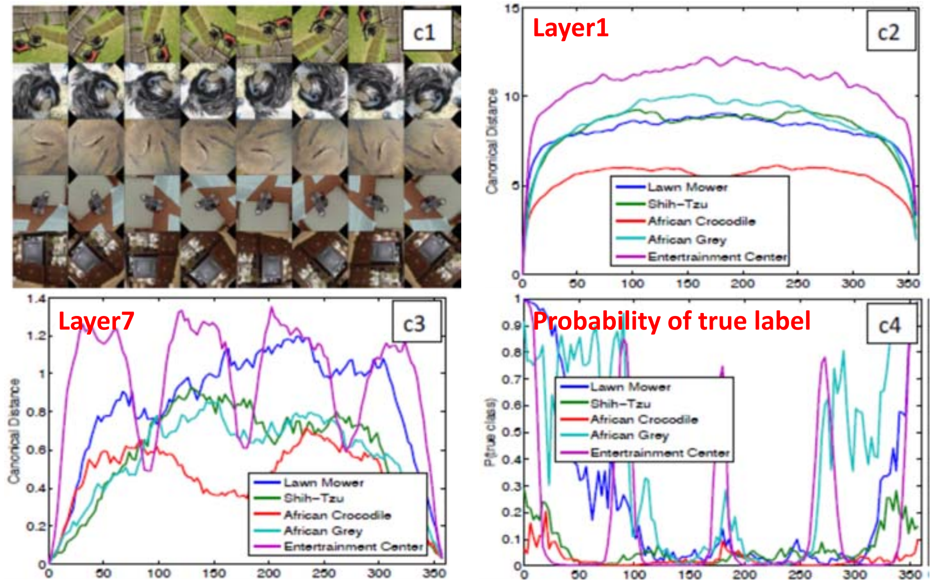


# Feature Invariance: Scale



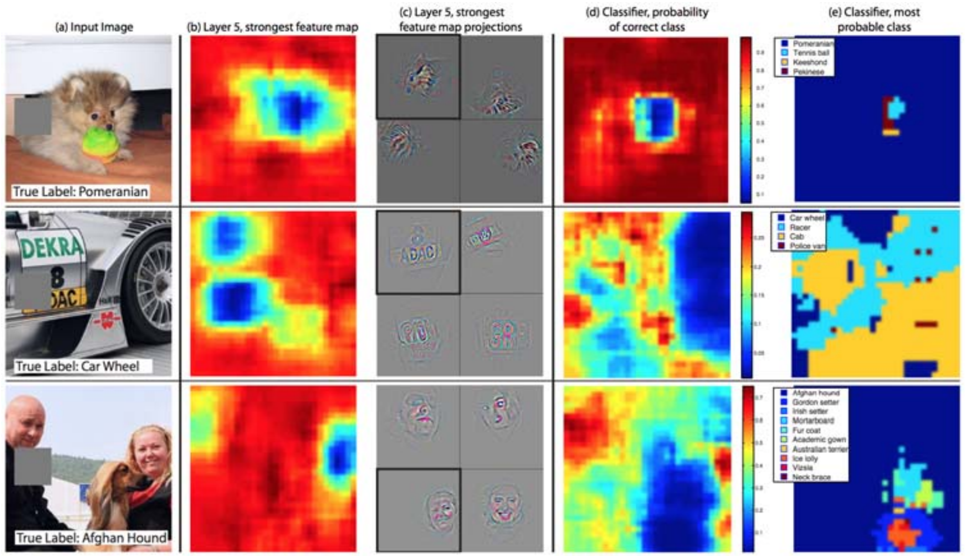



# Feature Invariance: Rotation





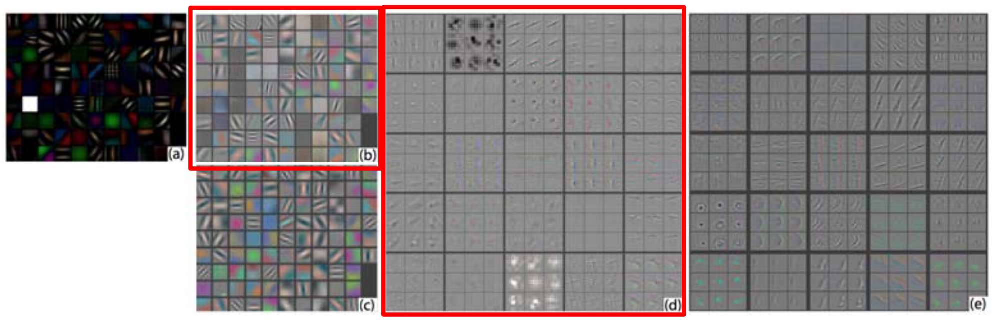

# Occlusion Sensitivity






# Architecture Selection

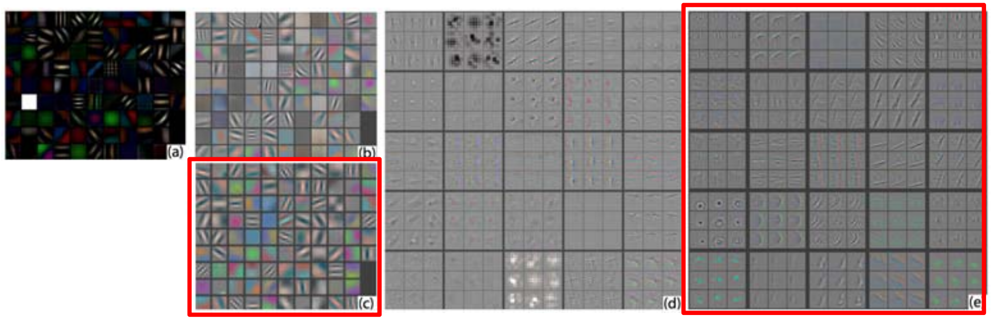
- Observations from AlexNet
  - The 1st layer filters
    - A mix of extremely high and low frequency information
    - Little coverage of the mid frequencies.
  - The 2nd layer visualization: aliasing artifacts caused by the large stride 4 used in the 1st layer convolutions.



## 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.**



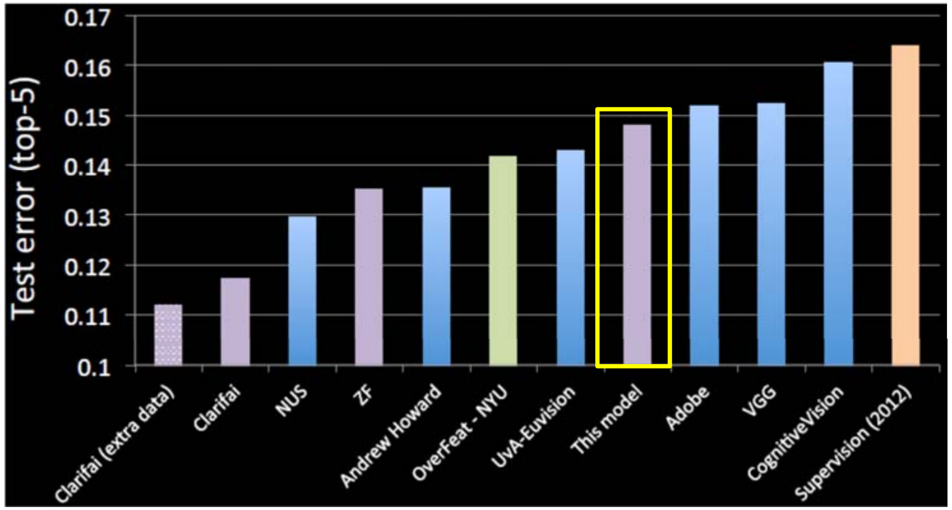



## Performance in ILSVRC 2012 Dataset

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



## ILSVRC 2013 Results

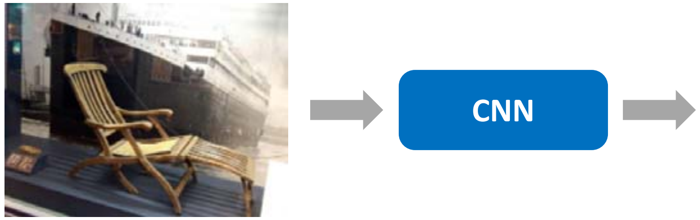





## Object Generation

# Discriminative vs. Generative CNN

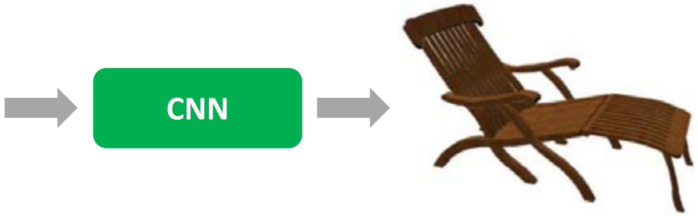
- Discriminative CNN




**Object class**
**Viewpoint**
**Style**
**…**

- Generative CNN

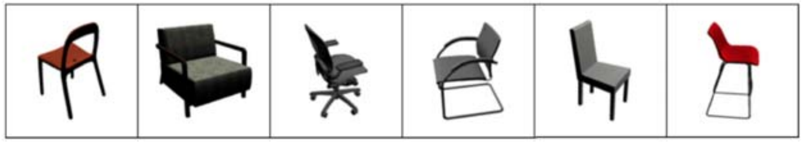
**Object class**
**Viewpoint**
**Style**
**…**






# Goal

- Generate an obejct 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



[Dosovitskiy15] A. Dosovitskiy, J. T. Springenberg and T. Brox. **Learning to generate chairs with convolutional neural networks**. CVPR 2015

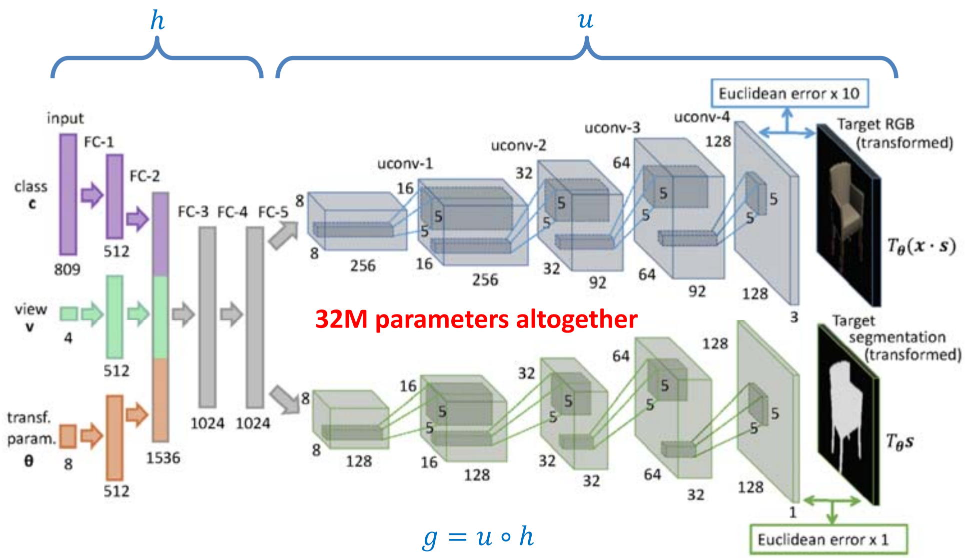


# Contribution

- 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.

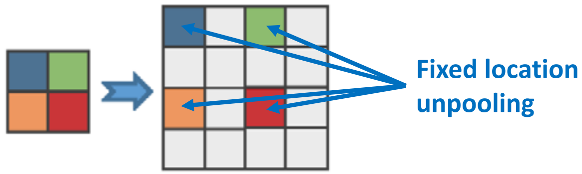


# Network Architecture

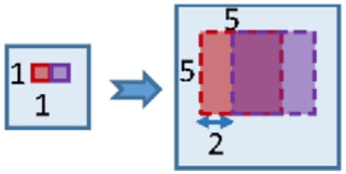


# Operations

- Unpooling: 2x2




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

[Aubry14] M. Aubry, D. Maturana, A. Efros, and J. Sivic, **Seeing 3D Chairs: Exemplar Part-based 2D-3D Alignment using a Large Dataset of CAD Models**. CVPR 2014



# Training

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

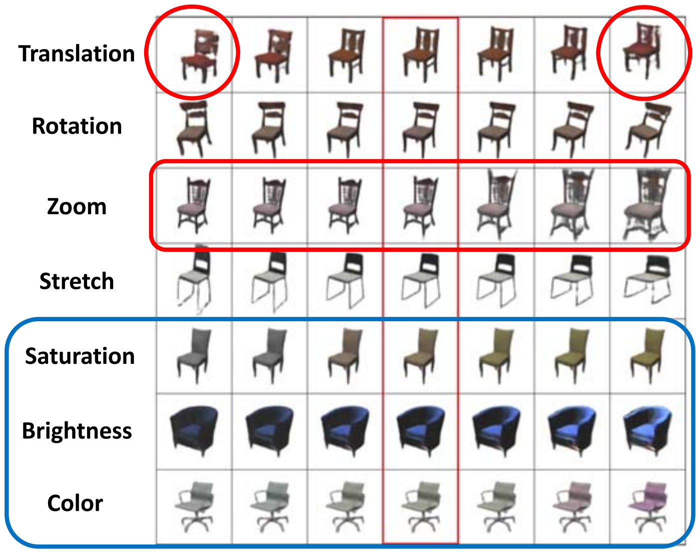
$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} \lambda \left\| u_{\mathrm{RGB}}\left(h(\boldsymbol{c}^i, \boldsymbol{v}^i, \theta^i)\right) - T_{\theta^i}(\boldsymbol{x}^i \cdot \boldsymbol{s}^i) \right\|_2^2 + \left\| u_{seg}\left(h(\boldsymbol{c}^i, \boldsymbol{v}^i, \theta^i)\right) - T_{\theta^i} \boldsymbol{s}^i \right\|_2^2$$

- 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]

[Saxe14] A. M. Saxe, J. L. McClelland, and S. Ganguli, **learning a Nonlinear Embedding by Preserving Class Neighbourhood**. ICLR 2014



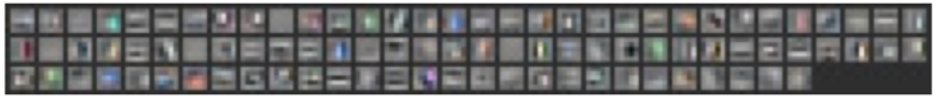
# Network Capacity



# 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

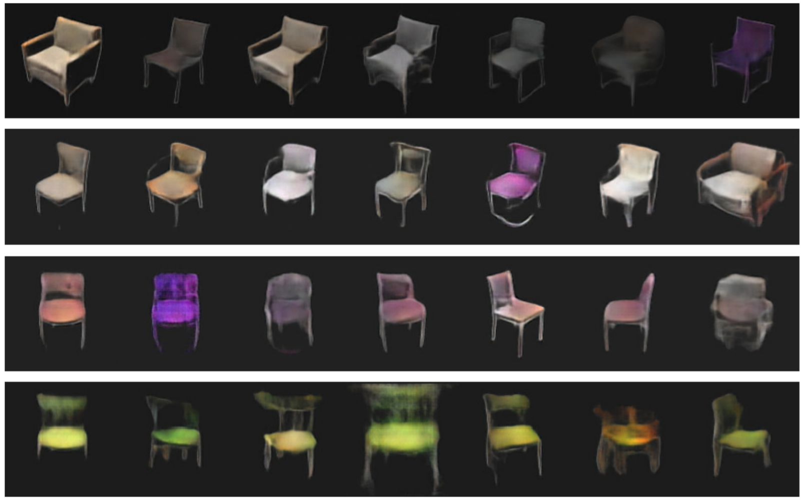
- Images generated from single unit activations

**FC-1 (class)**

**FC-2 (class)**

**FC-3**

**FC-4**

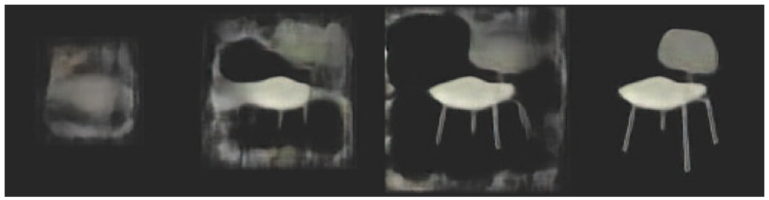




# Hidden Layer Analysis

- 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



**2x2** **4x4** **6x6** **8x8**



# Interpolation between Angles

**With knowledge transfer**

**Without knowledge transfer**

# Morphing Different Chairs




# 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.

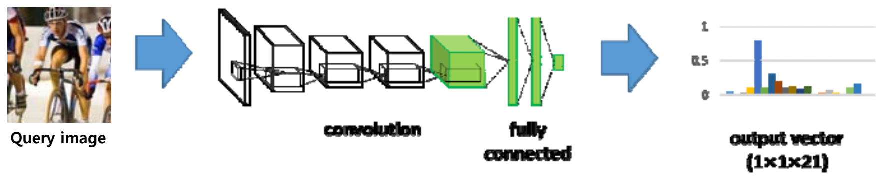


# Semantic Segmentation

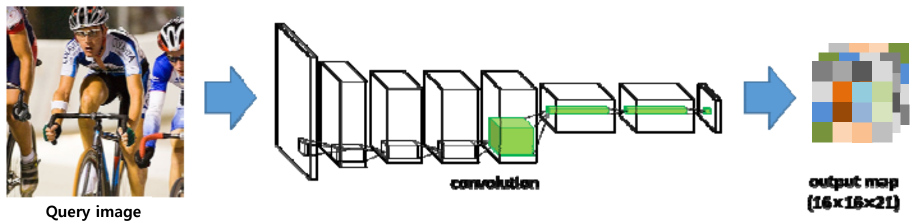


# Semantic Segmentation using CNN

- Image classification

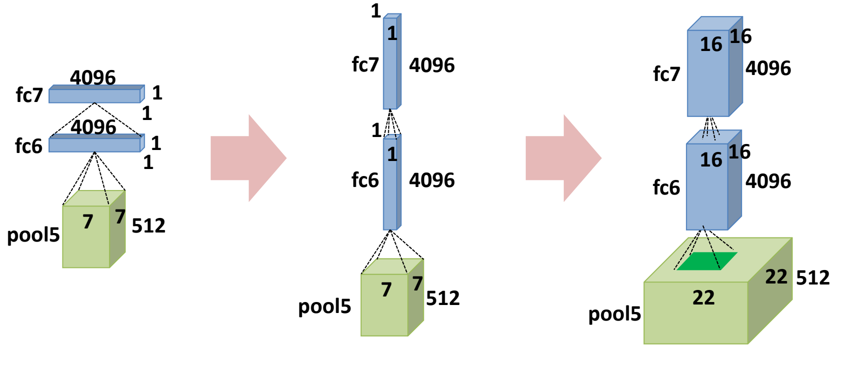



- Semantic segmentation
  - Given an input image, obtain pixel-wise segmentation mask using a deep Convolutional Neural Network (CNN)



# 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

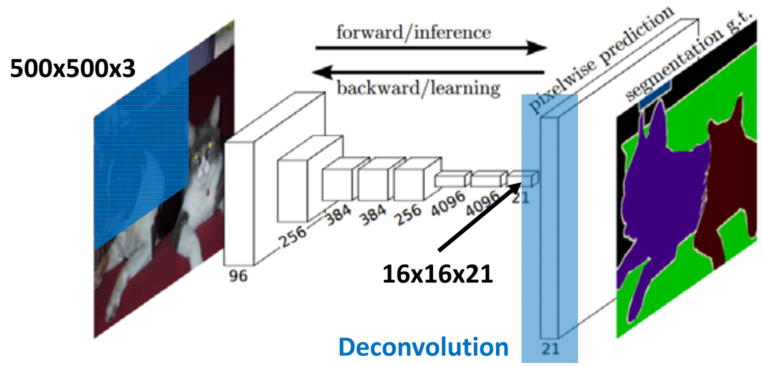



**Fully connected layers** **Convolution layers** **For the larger Input field**



# FCN for Semantic Segmentation

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

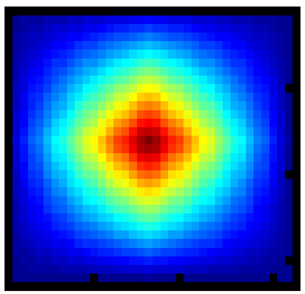



[Long15] J. Long, E. Shelhamer, and T. Darrell, **Fully Convolutional Network for Semantic Segmentation**. CVPR 2015



# 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.

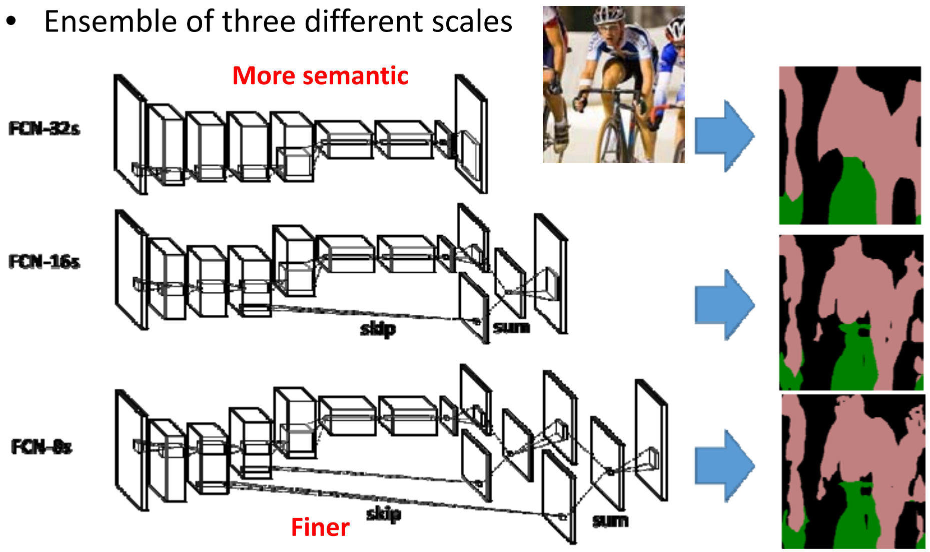


**64x64 bilinear interpolation**



# Skip Architecture

- Ensemble of three different scales

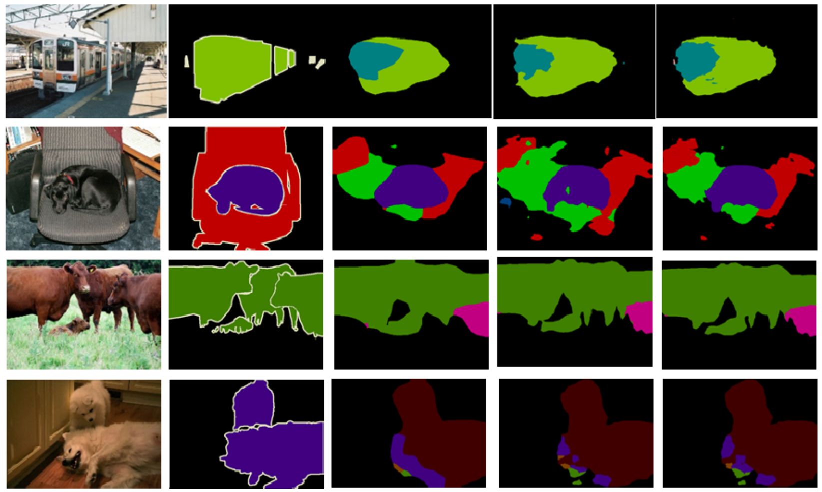


# Limitations of FCN-based Semantic Segmentation

- 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

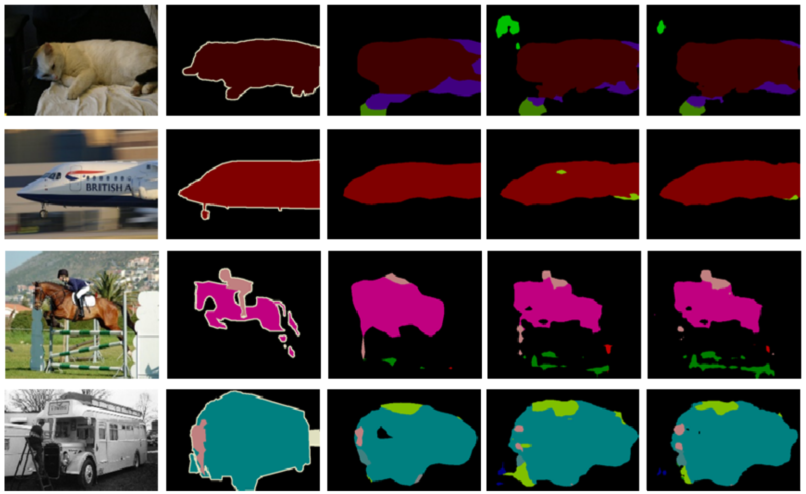


# Results and Limitations





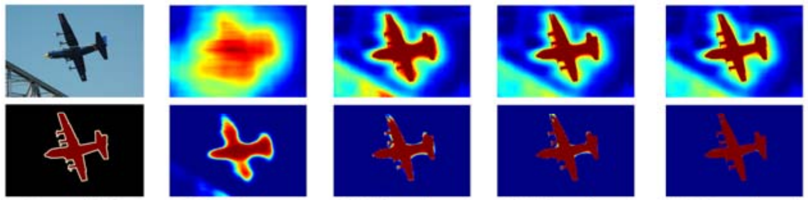

# Results and Limitations






# 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)

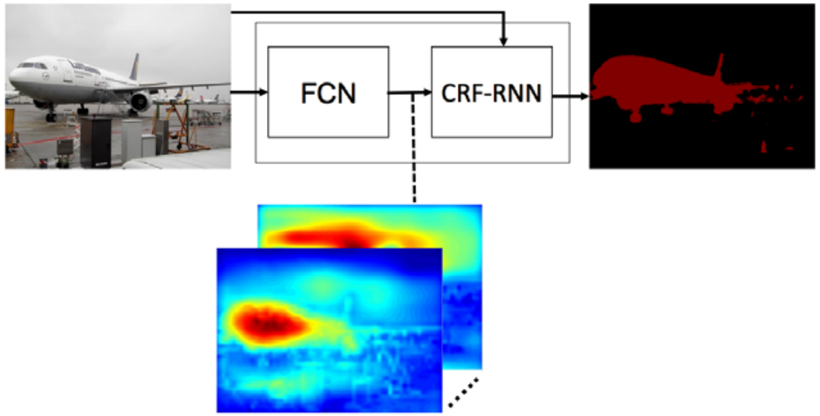



- Characteristics
  - No skip architecture in basic model
  - Simple output score map upscaling without deconvolution layer

[Chen15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. **Semantic image segmentation with deep convolutional nets and fully connected CRFs**. ICLR 2015

# CRF-RNN

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

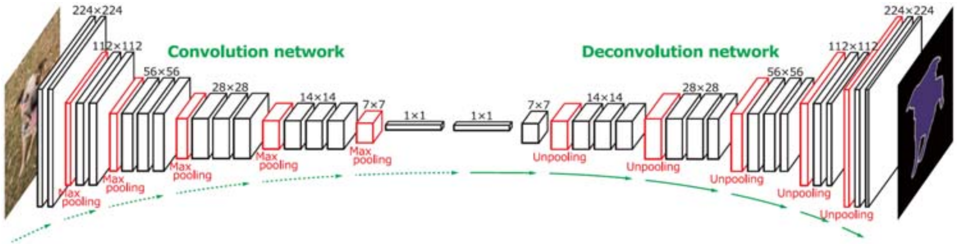



[Zheng2015] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr, **Conditional Random Fields as Recurrent Neural Networks**, arXiv:1502:03240, 2015



# 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

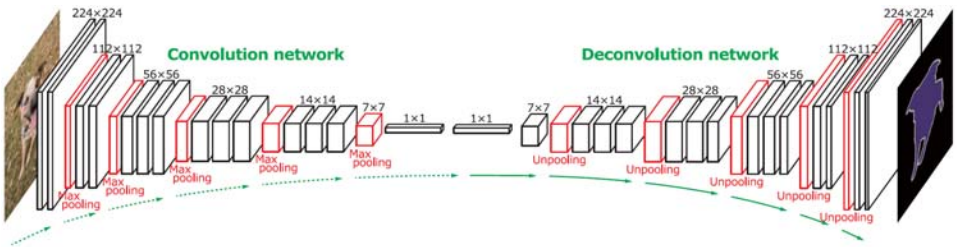



Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, **Learning Deconvolution Network for Semantic Segmentation**, arXiv:1505.04366, 2015



# 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




Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, **Learning Deconvolution Network for Semantic Segmentation**, arXiv:1505.04366, 2015



# 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?**

[Simonyan15] K. Simonyan and A. Zisserman: **Very Deep Convolutional Neural Networks for Large-Scale Image Recognition**. ICLR 2015

## 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.



## Training Strategy

- New GPU: NVIDIA GeForce GTX Titan X
  - Maxwell GPU architecture
  - 3072 CUDA cores
  - 1000MHz base clock / 1075MHz boost clock
  - 12G memory




## 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[Ioffe15]
  - 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

[Ioffe15] S. Ioffe and C. Szegedy. **Batch normalization: Accelerating deep network training by reducing internal covariate shift**. ICML 2015

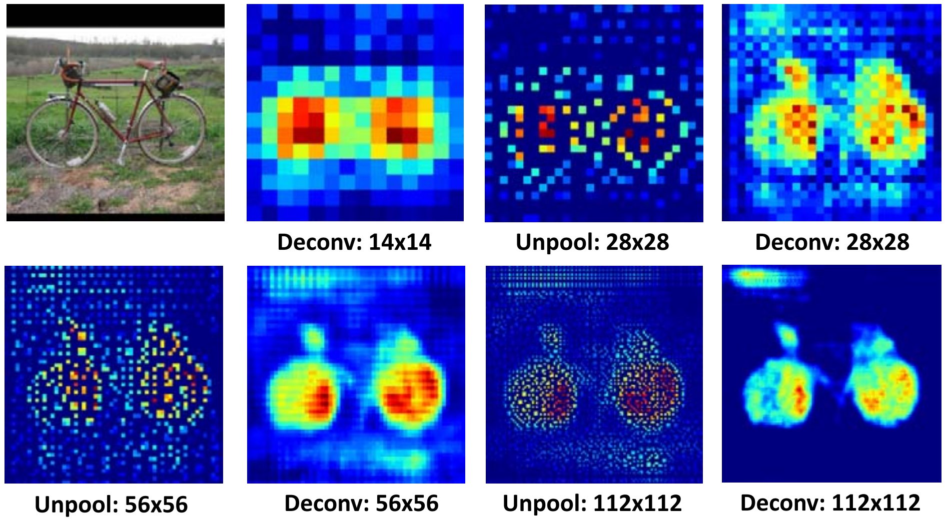


## Training Details

- 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.

# How Deconvolution Network Works?

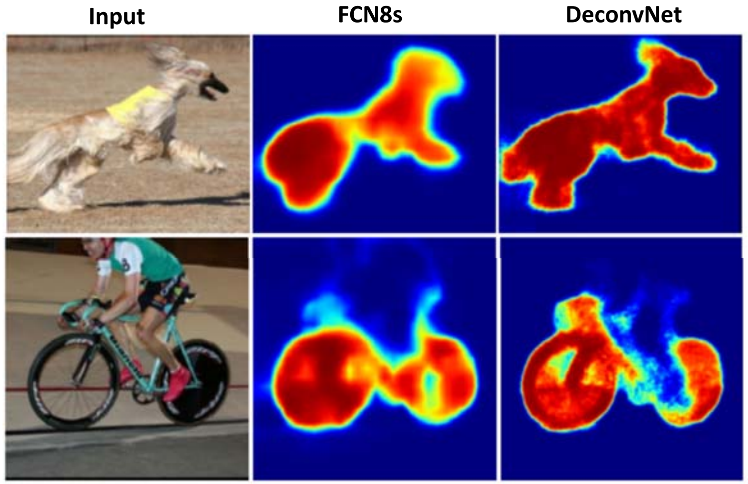
- Visualization of activations






# How Deconvolution Network Works?

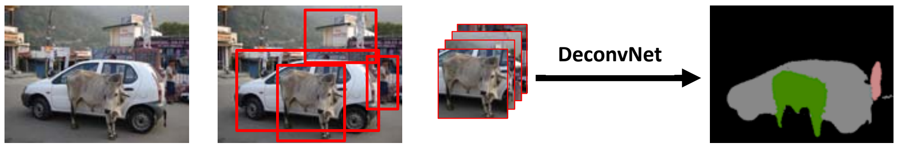
- Would FCN work equivalently if applied to a proposal?






# Inference

- Instance-wise prediction



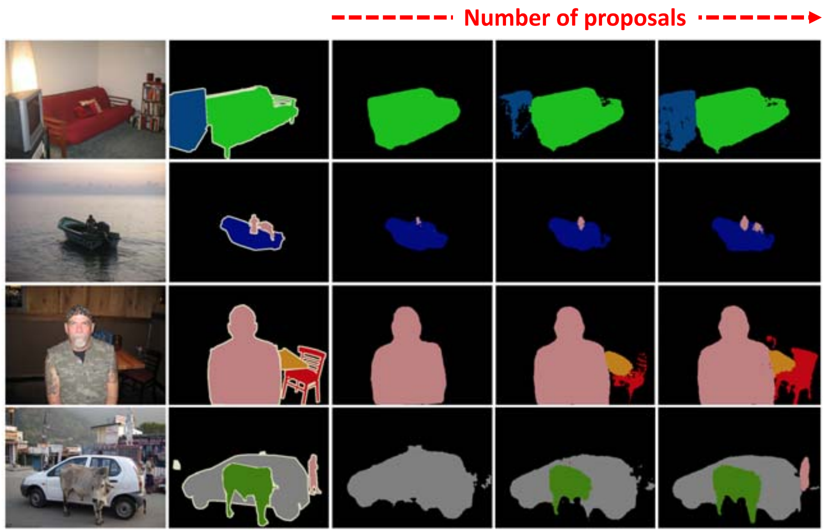

1. Input image 2. Object proposals 3. Prediction and aggregation 4. Results

  - 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

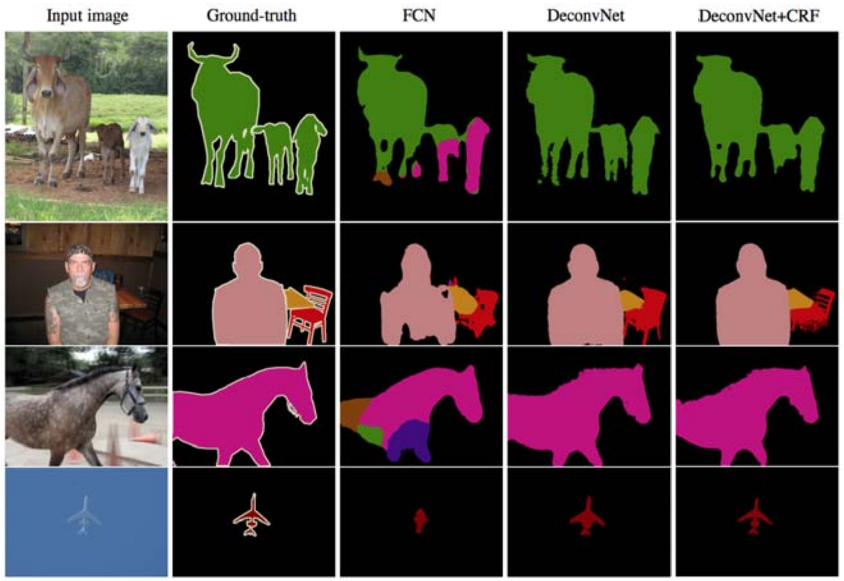


# Inference

- Handling multi-scale objects naturally

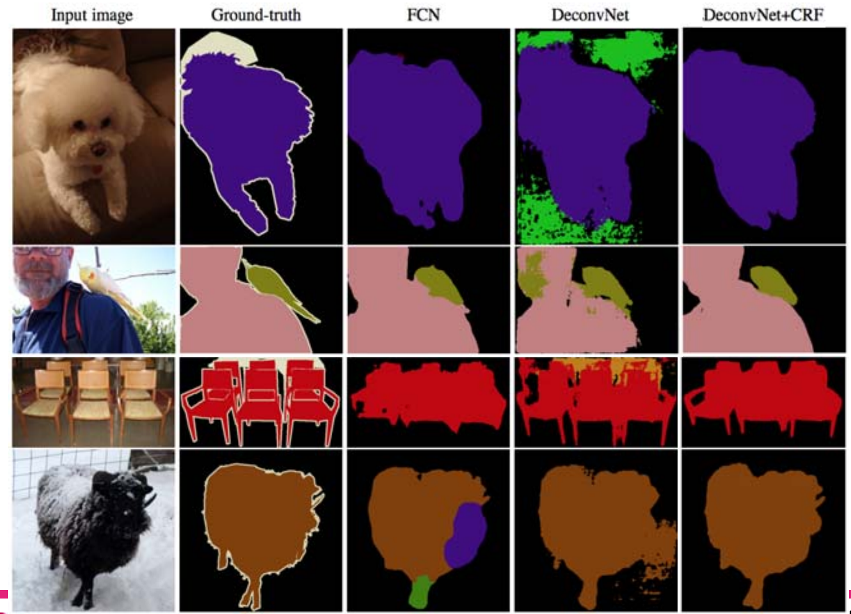


# Results






# Results






# PASCAL VOC 2012 Leaderboard

| | mean | aero plane | bicycle | bird | boat | bottle | bus | car | cat | chair | cow | dining table | dog | horse | motor bike | person | potted plant |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Adelaide_Context_CNN_CRF_COCO [7] | 76.4 | 91.8 | 39.3 | 82.0 | 66.3 | 75.0 | 93.0 | 83.6 | 87.1 | 39.1 | 84.5 | 64.3 | 84.9 | 87.4 | 83.7 | 84.7 | 66.2 |
| MSRA_BoxSup [7] | 75.2 | 89.8 | 38.0 | 89.2 | 68.9 | 68.0 | 89.6 | 83.0 | 87.7 | 34.4 | 83.6 | 67.1 | 81.5 | 83.7 | 85.2 | 83.5 | 58.6 |
| POSTECH_DeconvNet_CRF_VOC [7] | 74.8 | 90.0 | 40.8 | 84.2 | 67.3 | 70.7 | 90.9 | 84.8 | 87.4 | 34.8 | 83.0 | 58.7 | 82.3 | 87.1 | 86.9 | 82.4 | 64.5 |
| Oxford_TVG_CRF_RNN_COCO [7] | 74.7 | 90.4 | 55.3 | 88.7 | 68.4 | 69.8 | 88.3 | 82.4 | 85.1 | 32.6 | 78.5 | 64.4 | 79.6 | 81.9 | 86.4 | 81.8 | 58.6 |
| Adelaide_Message_Learning_VOC [7] | 74.3 | 89.9 | 38.0 | 79.2 | 64.5 | 76.1 | 89.6 | 84.5 | 86.8 | 37.5 | 80.4 | 57.0 | 83.1 | 84.2 | 83.8 | 83.2 | 58.5 |
| DeepLab-MSc-CRF-LargeFOV-COCO-CrossJoint [7] | 73.9 | 89.2 | 46.7 | 88.5 | 63.5 | 68.4 | 87.0 | 81.2 | 86.3 | 32.6 | 80.7 | 62.4 | 81.0 | 81.3 | 84.3 | 82.1 | 56.2 |
| Adelaide_Context_CNN_CRF_VOC [7] | 72.9 | 89.7 | 37.6 | 77.4 | 62.1 | 72.9 | 88.1 | 84.8 | 81.9 | 34.4 | 80.0 | 55.9 | 79.3 | 82.3 | 84.0 | 82.9 | 59.7 |
| DeepLab-CRF-COCO-LargeFOV [7] | 72.7 | 89.1 | 38.3 | 88.1 | 63.3 | 69.7 | 87.1 | 83.1 | 85.0 | 29.3 | 76.5 | 56.5 | 79.8 | 77.9 | 85.8 | 82.4 | 57.4 |
| POSTECH_EDeconvNet_CRF_VOC [7] | 72.5 | 89.9 | 39.3 | 79.7 | 63.9 | 68.2 | 87.4 | 81.2 | 86.1 | 28.5 | 77.0 | 62.0 | 79.0 | 80.3 | 83.6 | 80.2 | 58.8 |
| Oxford_TVG_CRF_RNN_VOC [7] | 72.0 | 87.5 | 39.0 | 79.7 | 64.2 | 68.3 | 87.6 | 80.8 | 84.4 | 30.4 | 78.2 | 60.4 | 80.5 | 77.8 | 83.1 | 80.6 | 59.5 |
| DeepLab-MSc-CRF-LargeFOV [7] | 71.6 | 84.4 | 54.5 | 81.5 | 63.6 | 65.9 | 85.1 | 79.1 | 83.4 | 30.7 | 74.1 | 59.8 | 79.0 | 76.1 | 83.2 | 80.8 | 59.7 |
| MSRA_BoxSup [7] | 71.0 | 86.4 | 35.5 | 79.7 | 65.2 | 65.2 | 84.3 | 78.5 | 83.7 | 30.5 | 76.2 | 62.6 | 79.3 | 76.1 | 82.1 | 81.3 | 57.0 |
| DeepLab-CRF-COCO-Strong [7] | 70.4 | 85.3 | 36.2 | 84.8 | 61.2 | 67.5 | 84.6 | 81.4 | 81.0 | 30.8 | 73.8 | 53.8 | 77.5 | 76.5 | 82.3 | 81.6 | 56.3 |
| DeepLab-CRF-LargeFOV [7] | 70.3 | 83.5 | 36.6 | 82.5 | 62.3 | 66.5 | 85.4 | 78.5 | 83.7 | 30.4 | 72.9 | 60.4 | 78.5 | 75.5 | 82.1 | 79.7 | 58.2 |
| TTI_zoomout_v2 [7] | 69.6 | 85.6 | 37.3 | 83.2 | 62.5 | 66.0 | 85.1 | 80.7 | 84.9 | 27.2 | 73.2 | 57.5 | 78.1 | 79.2 | 81.1 | 77.1 | 53.6 |
| DeepLab-CRF-MSc [7] | 67.1 | 80.4 | 36.8 | 77.4 | 55.2 | 66.4 | 81.5 | 77.5 | 78.9 | 27.1 | 68.2 | 52.7 | 74.3 | 69.6 | 79.4 | 79.0 | 56.9 |
| DeepLab-CRF [7] | 66.4 | 78.4 | 33.1 | 78.2 | 55.6 | 65.3 | 81.3 | 75.5 | 78.6 | 25.3 | 69.2 | 52.7 | 75.2 | 69.0 | 79.1 | 77.6 | 54.7 |
| CRF_RNN [7] | 65.2 | 80.9 | 34.0 | 72.9 | 52.6 | 62.5 | 79.8 | 76.3 | 79.9 | 23.6 | 67.7 | 51.8 | 74.8 | 69.9 | 76.9 | 76.9 | 49.0 |
| TTI_zoomout_16 [7] | 64.4 | 81.9 | 35.1 | 78.2 | 57.4 | 56.5 | 80.5 | 74.0 | 79.8 | 22.4 | 69.6 | 53.7 | 74.0 | 76.0 | 76.6 | 68.8 | 44.3 |
| Hypercolumn [7] | 62.6 | 68.7 | 33.5 | 69.8 | 51.3 | 70.2 | 81.1 | 71.9 | 74.9 | 23.9 | 60.6 | 46.9 | 72.1 | 68.3 | 74.5 | 72.9 | 52.6 |
| FCN-8s [7] | 62.2 | 76.8 | 34.2 | 68.9 | 49.4 | 60.3 | 75.3 | 74.7 | 77.6 | 21.4 | 62.5 | 46.8 | 71.8 | 63.9 | 76.5 | 73.9 | 45.2 |



# 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