An Introduction to Deep Convolutional Neural Networks (CNN)

Mahdi S. Hosseini

March 2018

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING

Outline

- Some computer vision and image processing applications
- Classical solutions: brief overview
- Concept of multi-layered decomposition
- Convolutional Neural Networks (CNN)
- CNN Progress
- More Application Study

Application Example: Image Classification

- Images are categorized in different labeled classes
- Inter-class variability
  - different variations of the same class look differently
  - Preferably maximize its effect for learning method
- Intra-class variability
  - Describes how strongly units in the same group resemble each other
  - Preferably minimize its effect for learning method

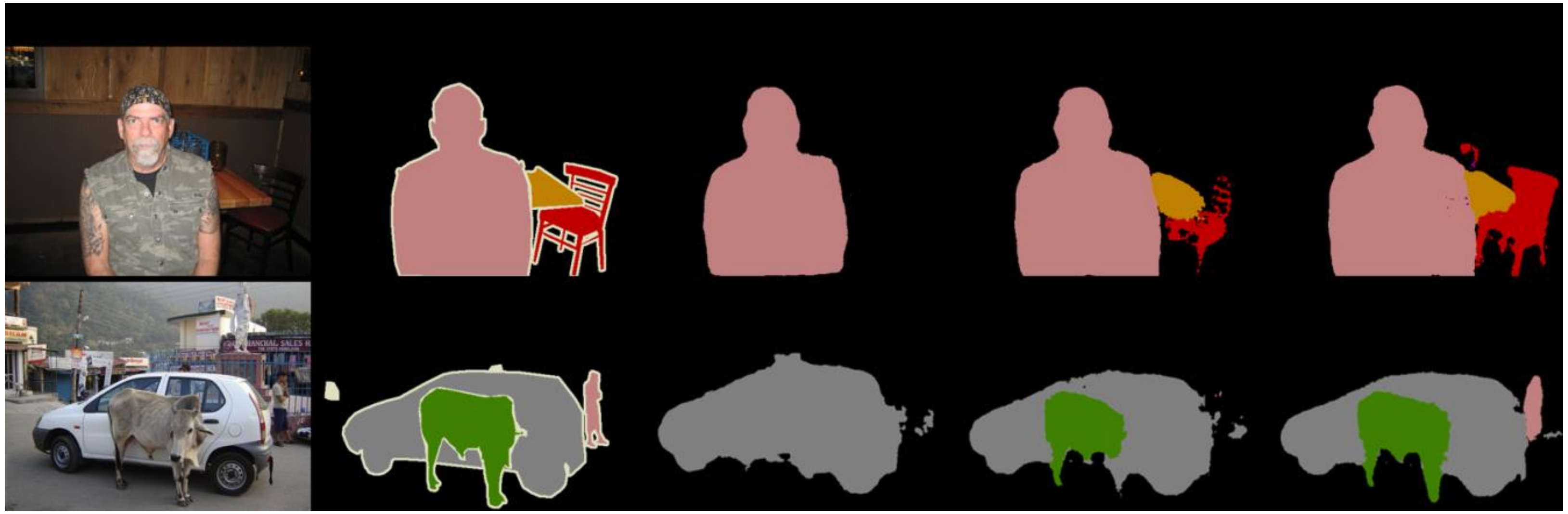
Inter-class variability

Intra-class variability

Application Example: Semantic Segmentation

- Understanding image in pixel level
- Transform image in to more abstract representations such as line segments, curve segments, circles, etc

Input Image Ground Truth Number of proposals



Application Example: Image Denoising

- Acquired image through certain modality is contaminated by artifacts

$$\mathbf{y} = \mathbf{x} + \mathbf{v}$$

- Objective: clean the noise effect while preserve meaningful information

Ground-truth

Noisy

Recovered: Method-A

Recovered: Method-B

Application Example: Image Deblurring

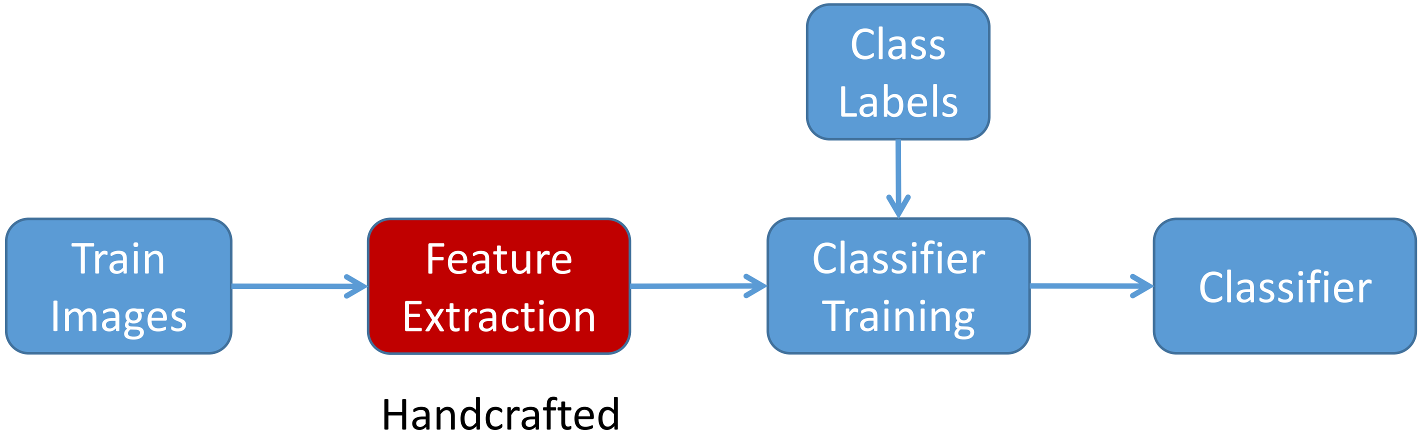
- Blurry observation from imaging modality: $f_B = f_T * h + \eta$
- Cause of blur: PSF/Optical Aberration, weather conditions e.g fog, haze
- Objective: reconstruct sharp image by canceling blur effect h

Ground-truth

Blurry Observation

Recovered: Method-A

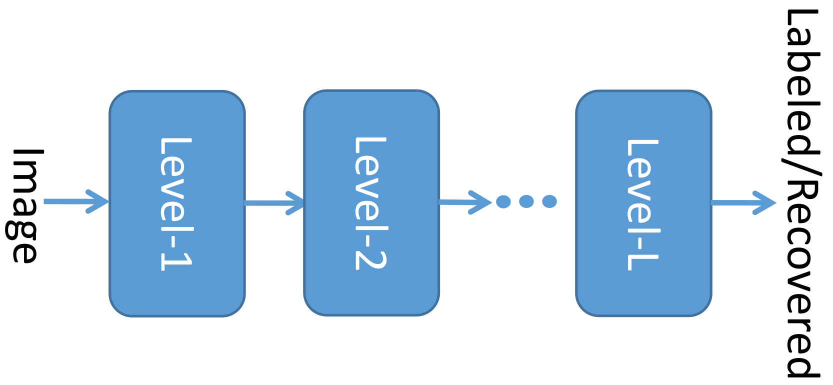
Classical Solutions

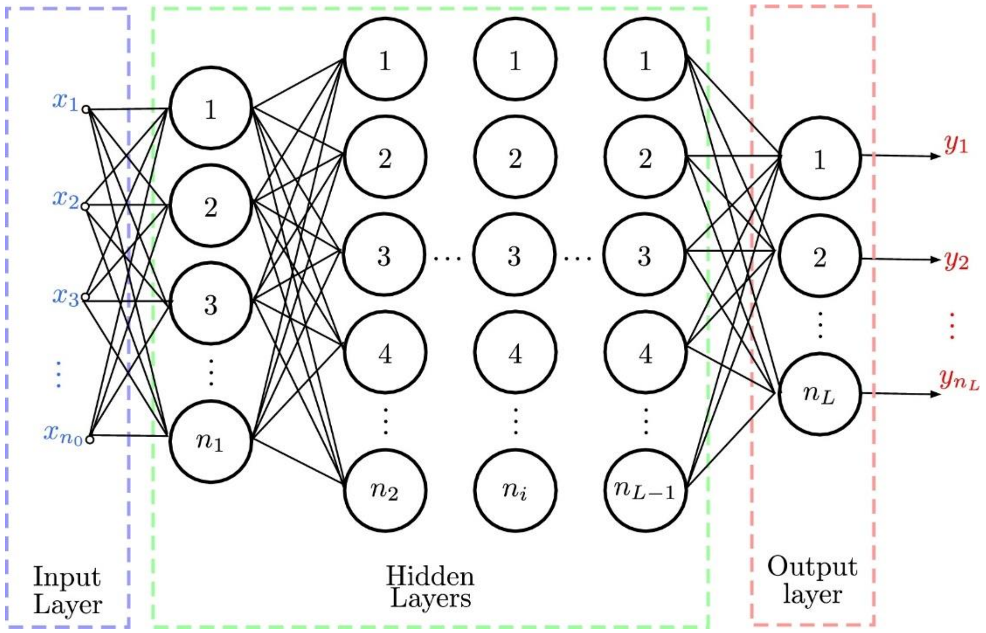


- Image Representation:
  - Classification
  - Semantic Segmentation
- Image Reconstruction:
  - Denoising/Deblurring:
    - Multi-Resolution approaches e.g. Wavelet
    - Variational regularization e.g. TV1/TV2
    - Sparse models
    - Statistical prior models
- A Common disadvantage to all:
  - Feature extraction/processing is done in one-layer mode
  - In other words: Processed feature are not processed again

Why Deep-Layered Design?

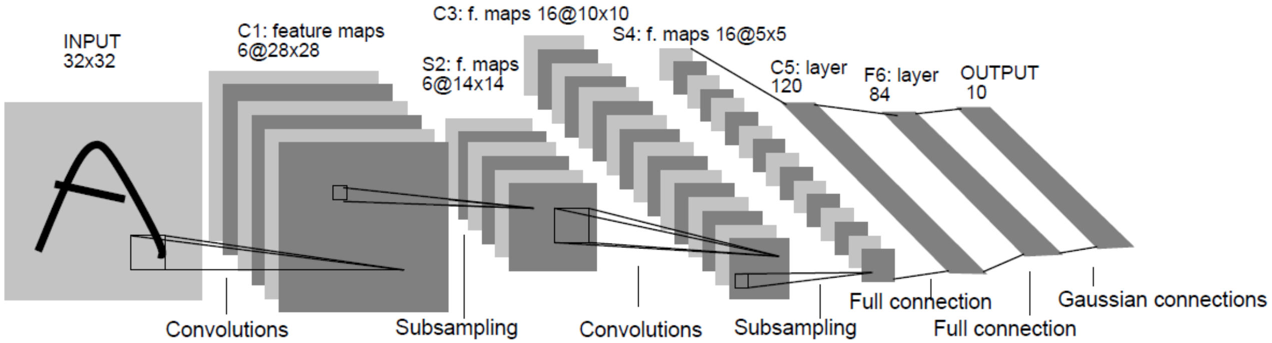
- Learn representation of data in multiple level of decomposition (abstraction)
    - Learning hierarchy of feature extractors
    - Each level in hierarchy extracts features from the output of previous layer
- Nested decomposition provides meaningful interpretation of complex structures
- Neural Networks (NN) had similar approach to break input data in to multi layered processing level towards classification e.g. Multi-Layer-Perceptron (MLP)



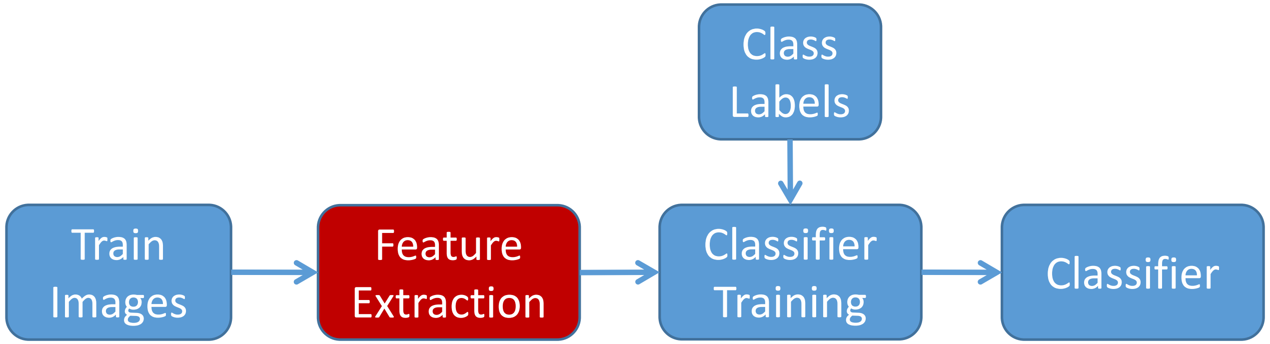


Rise of Convolutional Neural Networks (CNN)

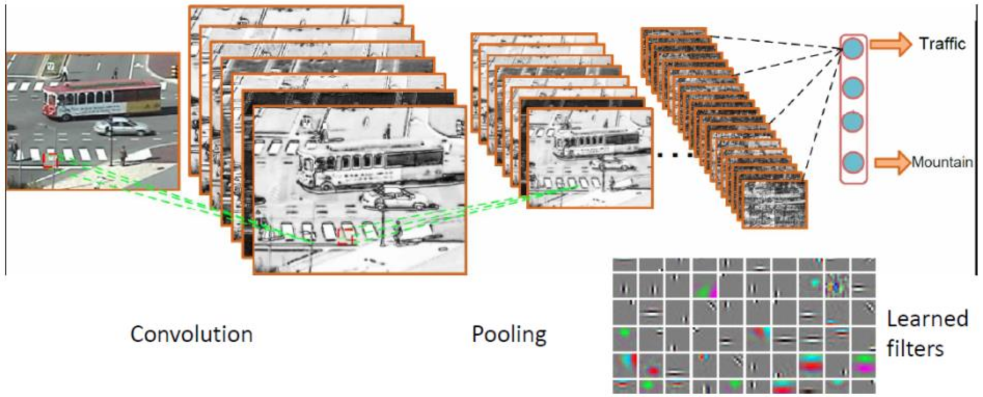
- CNN is the extension of MLP based on 3 main ideas proposed in 1998:
  - Local 2D-convolution operation
  - Multiple convolutions, sharing the same information
  - Spatial sub-sampling



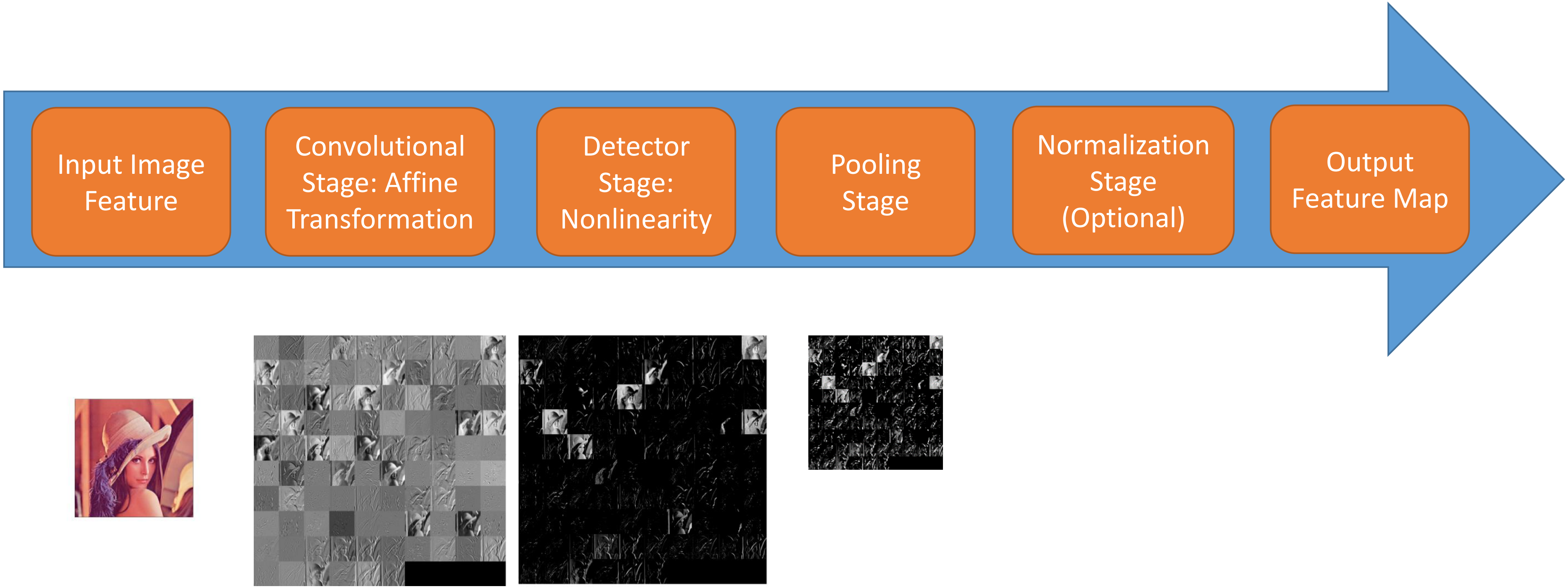
Architecture of LeNet-5: Yann LeCun et al., 1998



Lets TRAIN it!



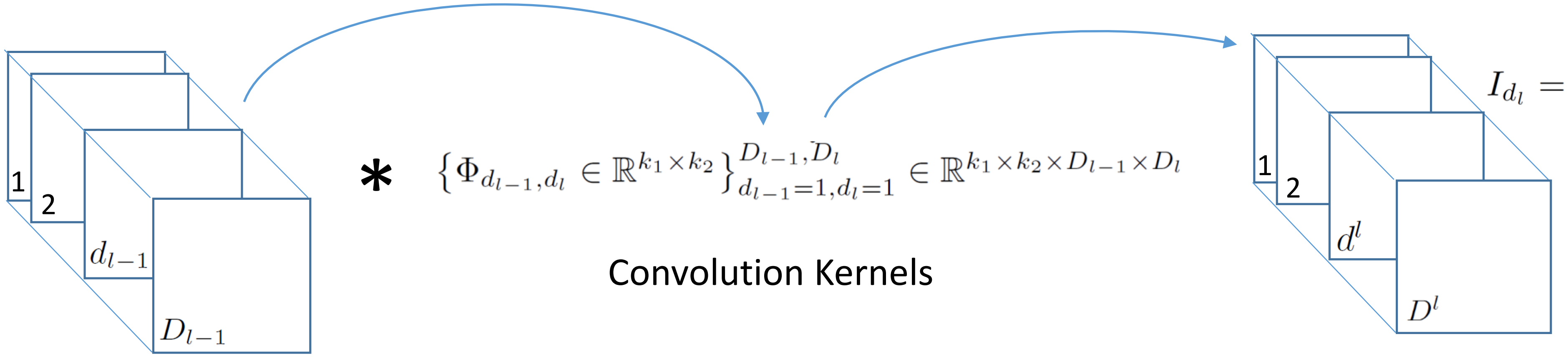
Typical Layer of CNN



CNN Sub-module: Convolution Layer

Convolutional Stage: Affine Transformation

- Stack of image features are provided as input and convolved by series of 2D convolution filters for feature transformation
- Refer to particular input image channel by $d^l = 1, \cdots, D^l$
- Output image channels are obtained by super-position of 2D convolutions of previous image channels



$$* \; \{\Phi_{d_{l-1},d_l} \in \mathbb{R}^{k_1 \times k_2}\}_{d_{l-1}=1,d_l=1}^{D_{l-1},D_l} \in \mathbb{R}^{k_1 \times k_2 \times D_{l-1} \times D_l}$$

$$I_{d_l} = \sum_{d_{l-1}=1}^{D_{l-1}} I_{d_{l-1}} * \Phi_{d_{l-1},d_l} + b_{d_l}$$

Input Image Stacks

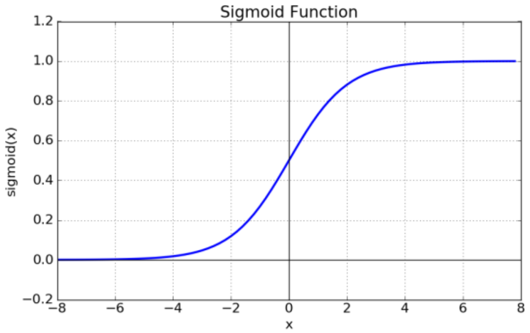
Convolution Kernels

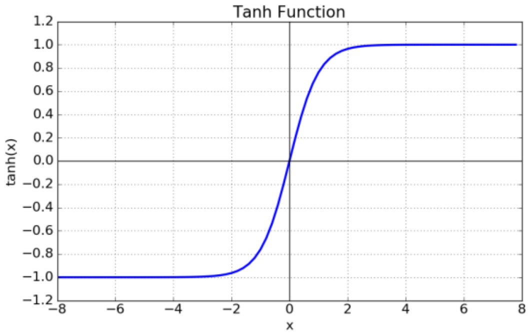
Output Image Stacks

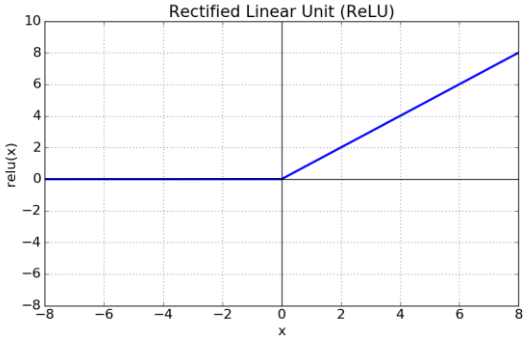
CNN Sub-module: Nonlinearity function

Detector Stage: Nonlinearity

- Nonlinearity function known as activation function
- Why do we need it?
  - Should all feature maps passed to proceeding layer?
  - Output cannot be reproduced by linear combination of input layers which separates the two consecutive layers, otherwise it will be redundant!
  - Nonlinearity on the output makes learning algorithms converge faster



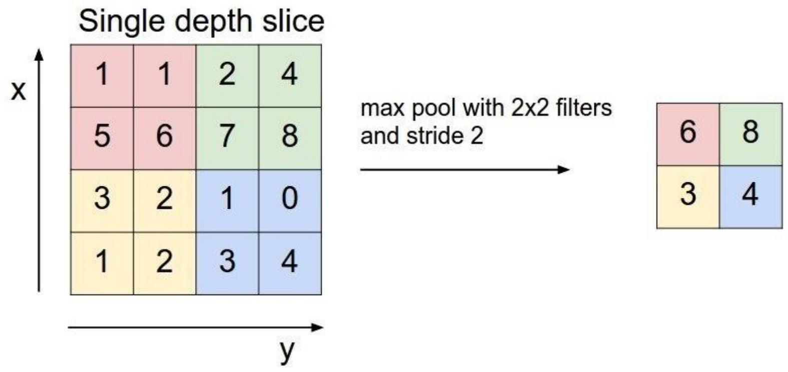


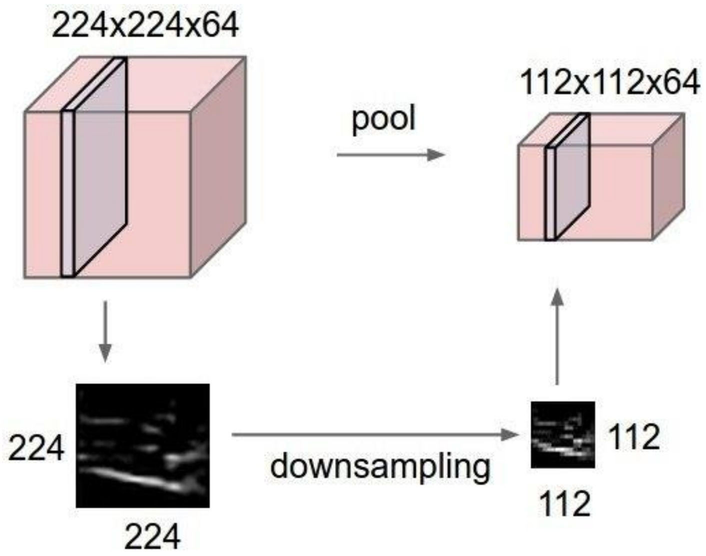


CNN Sub-module: Pooling

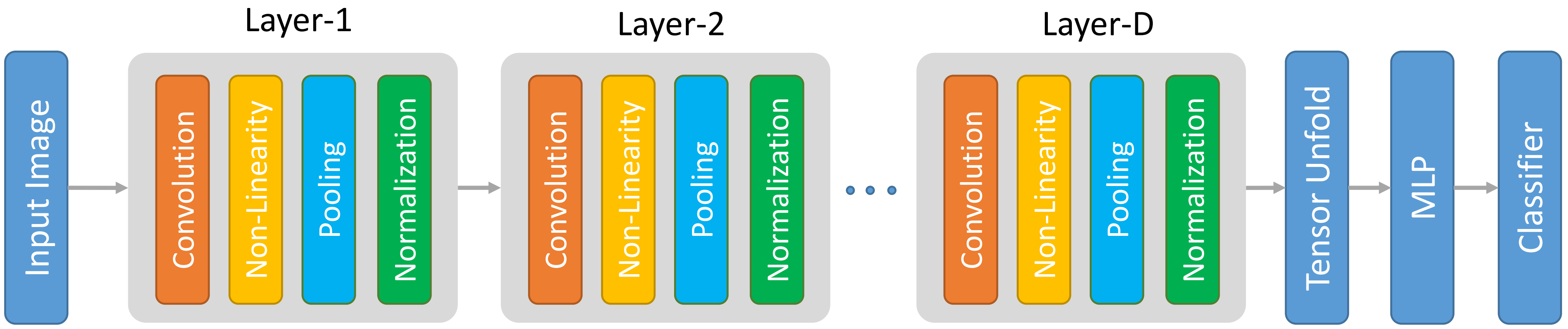
Pooling Stage

- Pooling refers to a process of linear/nonlinear selection of feature pixel within a window e.g. 2x2
- The operation is basically a down-sampler either max/avg
- Image scales are changed by pooling operator



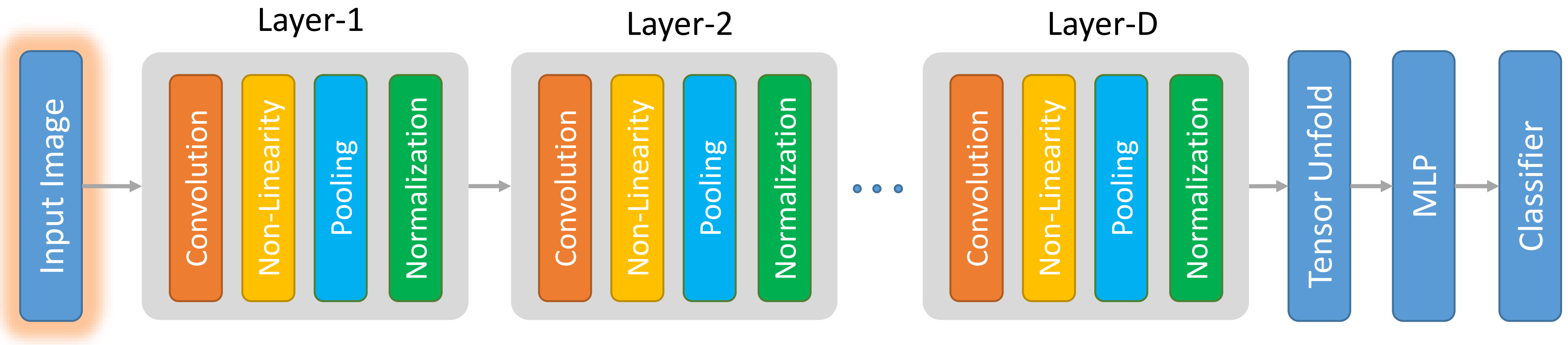


CNN Architectural Overview

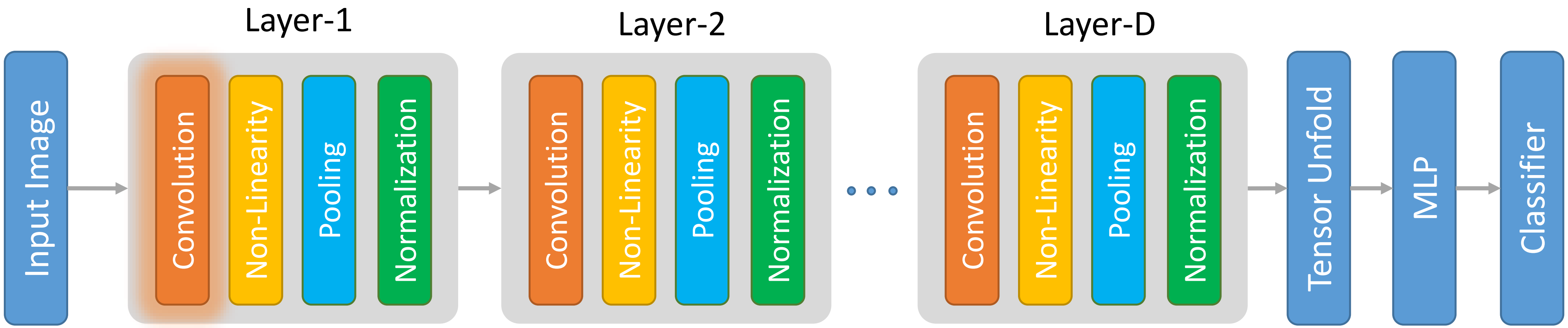


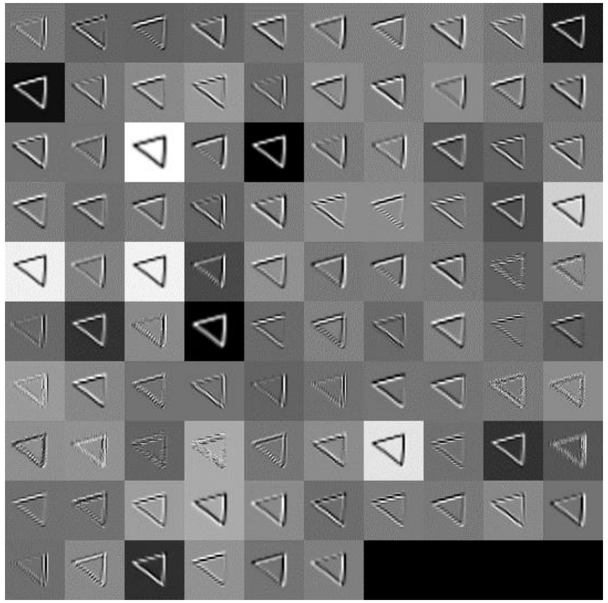
- Select D-Layers for training, stride/pooling
- Training variables:
  - Convolution filter weights Φ_{d_{l-1},d_l}
  - Bias values b_{d_l}
  - Weights W^z and bias b^z for z-th Fully-Connected (FC) layer from MLP
- Once the CNN is trained ➡ pre-trained network for feature extraction

Progress of Image Decomposition in AlexNet (Example)

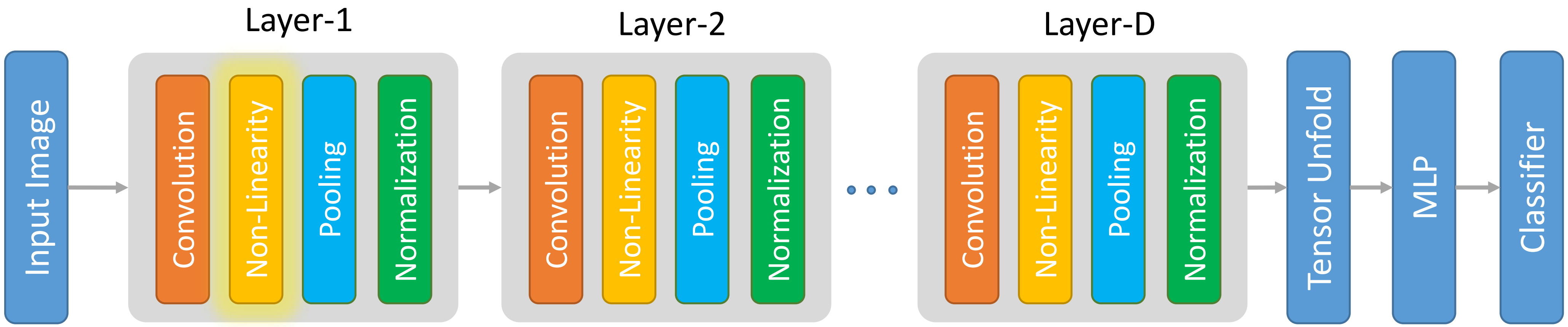


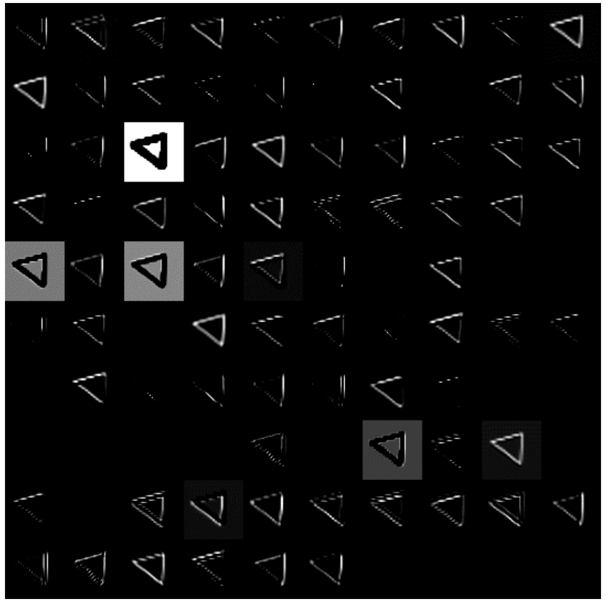
Progress of Image Decomposition in AlexNet (Example)



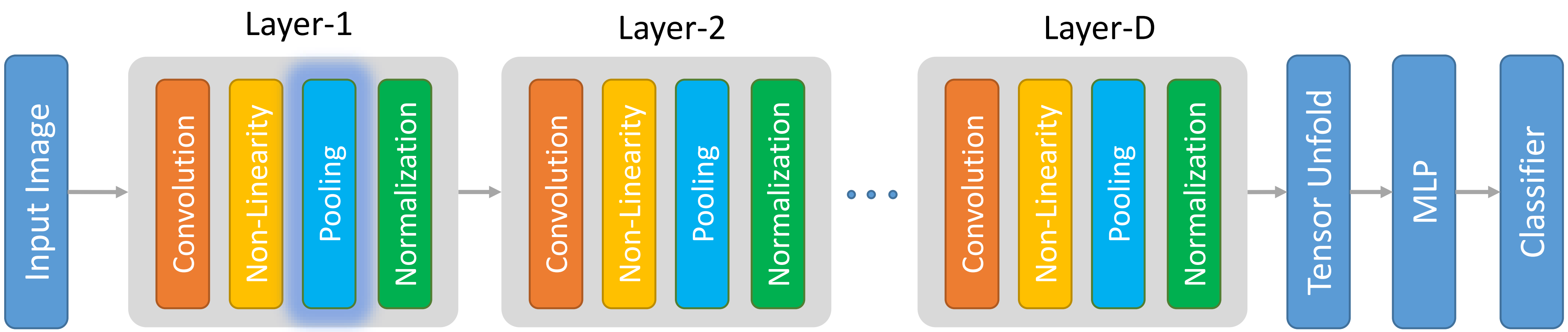


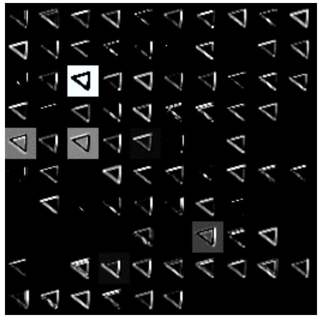
Progress of Image Decomposition in AlexNet (Example)



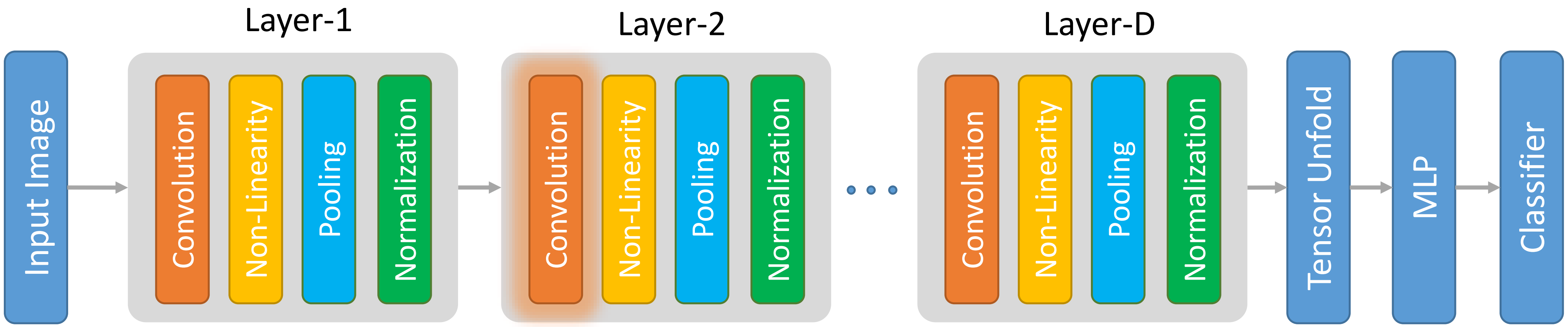


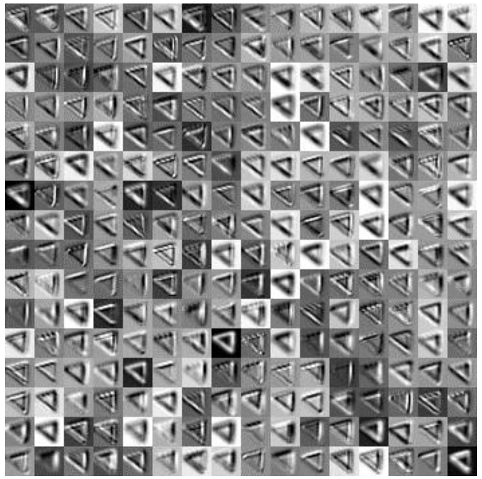
Progress of Image Decomposition in AlexNet (Example)



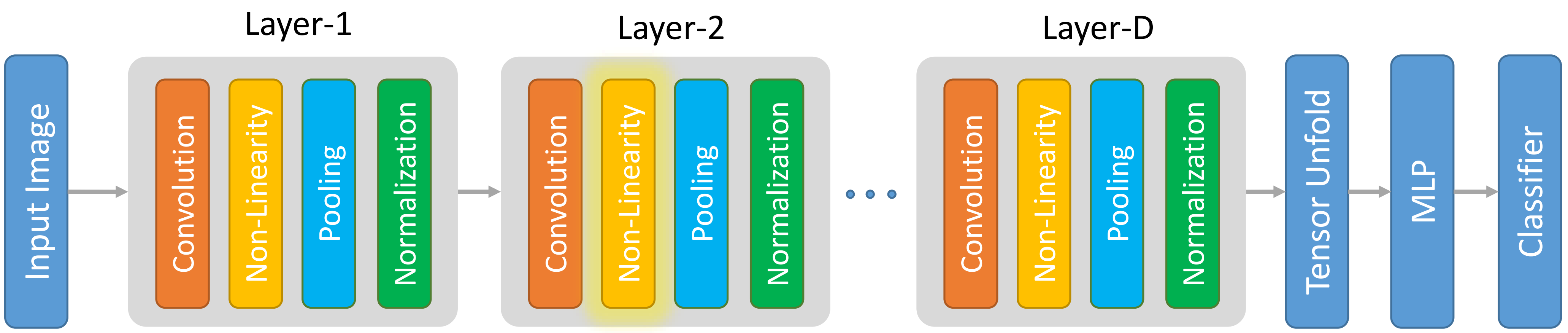


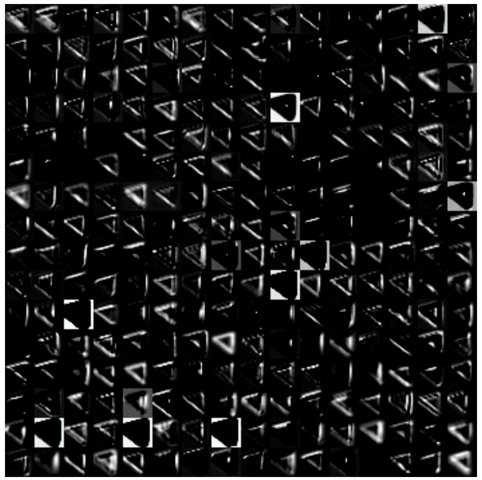
Progress of Image Decomposition in AlexNet (Example)



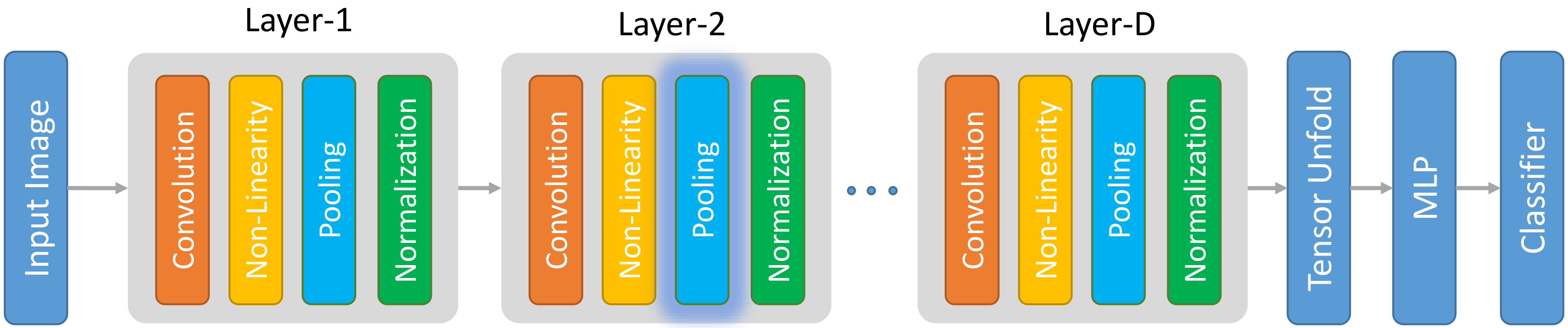


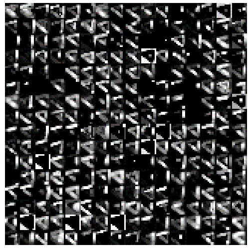
Progress of Image Decomposition in AlexNet (Example)



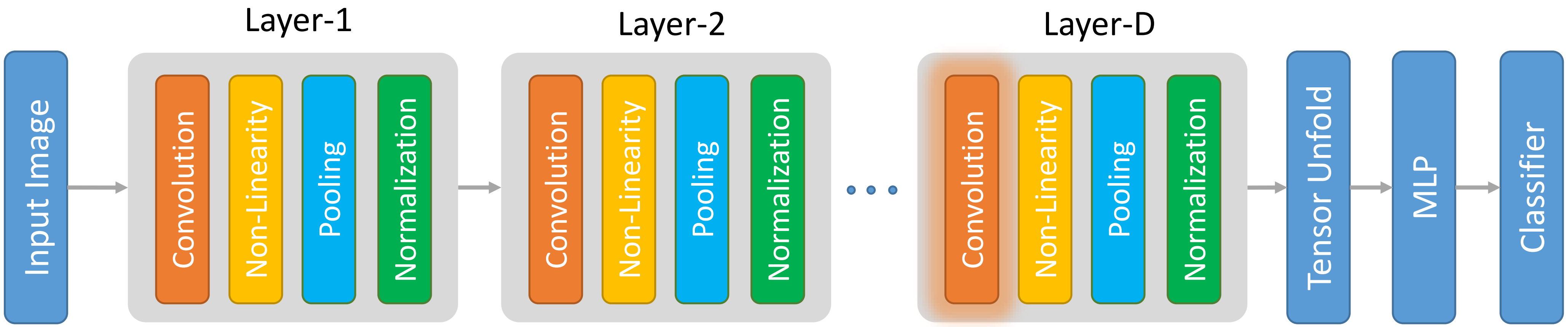


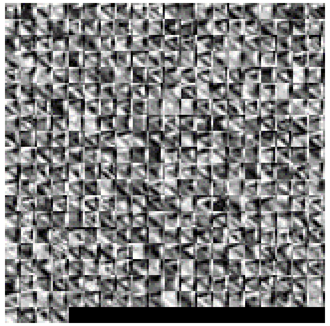
Progress of Image Decomposition in AlexNet (Example)



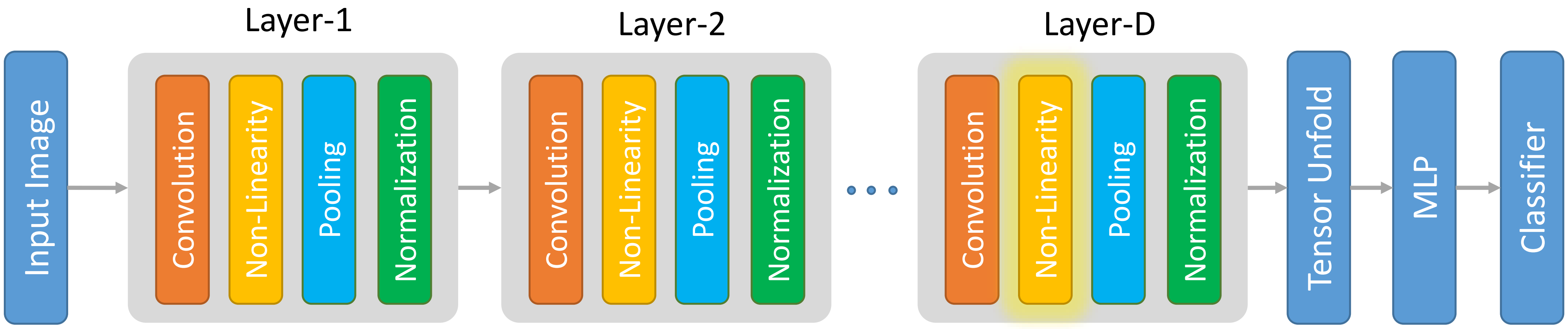


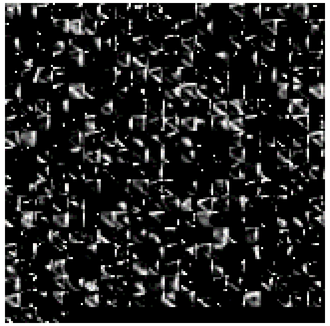
Progress of Image Decomposition in AlexNet (Example)





Progress of Image Decomposition in AlexNet (Example)

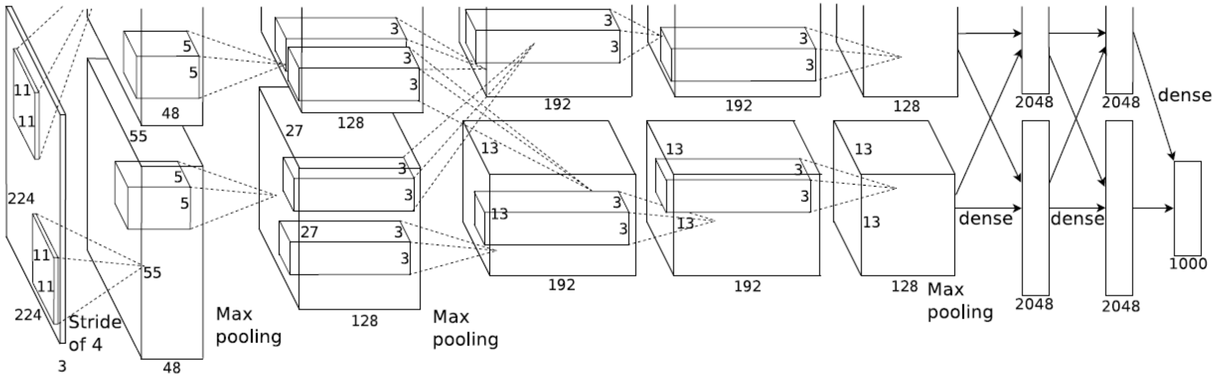


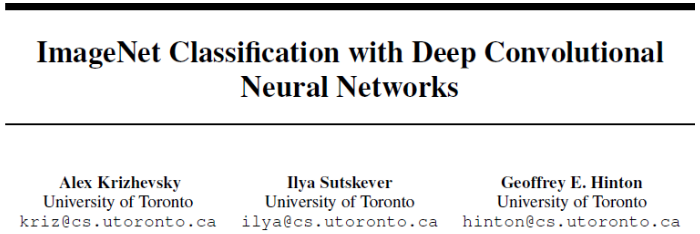


How to Train CNN?

- Select train image set
- Initialized network parameters e.g. Φ_{d_{l-1},d_l} b_{d_l} W^z b^z
- Learning via stochastic gradient descent (SGD)
  - **Shuffle** the train set and select a batch
  - **Feed-forward pass**: calculate network's input/output variables
  - **Back-propagation pass**: calculate error gradient with respect to all variables
  - **Update variables** e.g. $\Phi_{d_{l-1},d_l} \leftarrow \Phi_{d_{l-1},d_l} - \eta \frac{\partial E}{\partial \Phi_{d_{l-1},d_l}}$
- GPUs are used to boost the computation speed

AlexNet





ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

- Five convolution layers D=5
- Three fully connected layers Z=3
- Number of learning weights = 60M
- Trained on ImageNet

Table: Comparison of error rates on ILSVRC-2010

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

AlexNet: 1^{st}-Layer Convolution Filter Inspection

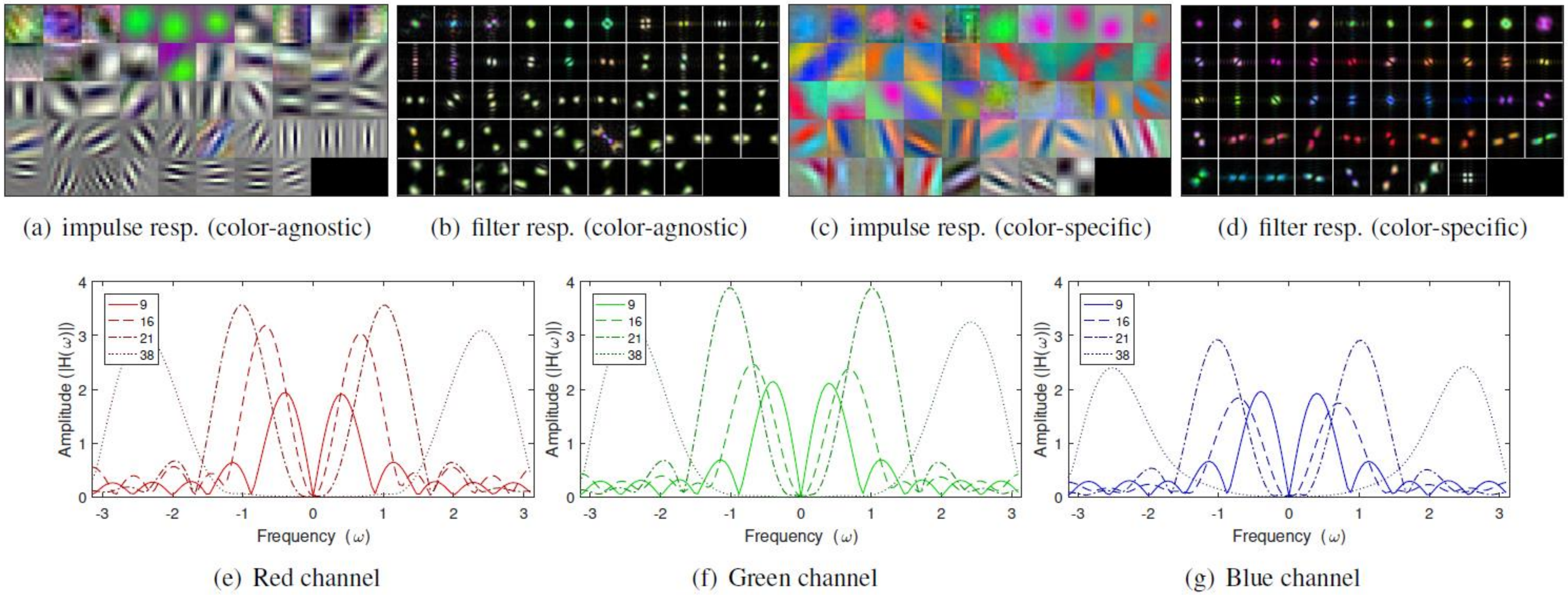


Figure 1. Top row: AlexNet [25] first layer convolution kernels for color-agnostic and color-specific sets. Bottom row: Examples of frequency response envelope for AlexNet color-agnostic kernels (cut horizontally) for kernel numbers $\{9, 16, 21, 30\}$.

Evolution of Deep-CNN

- Since AlexNet in 2012, research studies suggest:
  - Lower size of convolution filter e.g. 3x3 (computational efficiency)
  - More deeper layers
    - First layer: basic edge information
    - Second Layer: collection of edges such as shape
    - Third Layer: collections of shapes like eyes or noses
- Pros:
  - more generalization compared to shallow network
  - Capable of learning more complex structures
- Cons:
  - Increasing the layer size could cause algorithm more prone to overfitting and its generalization error is likely to increase

Published as a conference paper at ICLR 2015

VGG16

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* & Andrew Zisserman+
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

- Pushing depths to 16-19 Layers
- 3x3 convolution filters
- Number of learning weights ~ 130M
- Accuracy is saturated by increasing depth

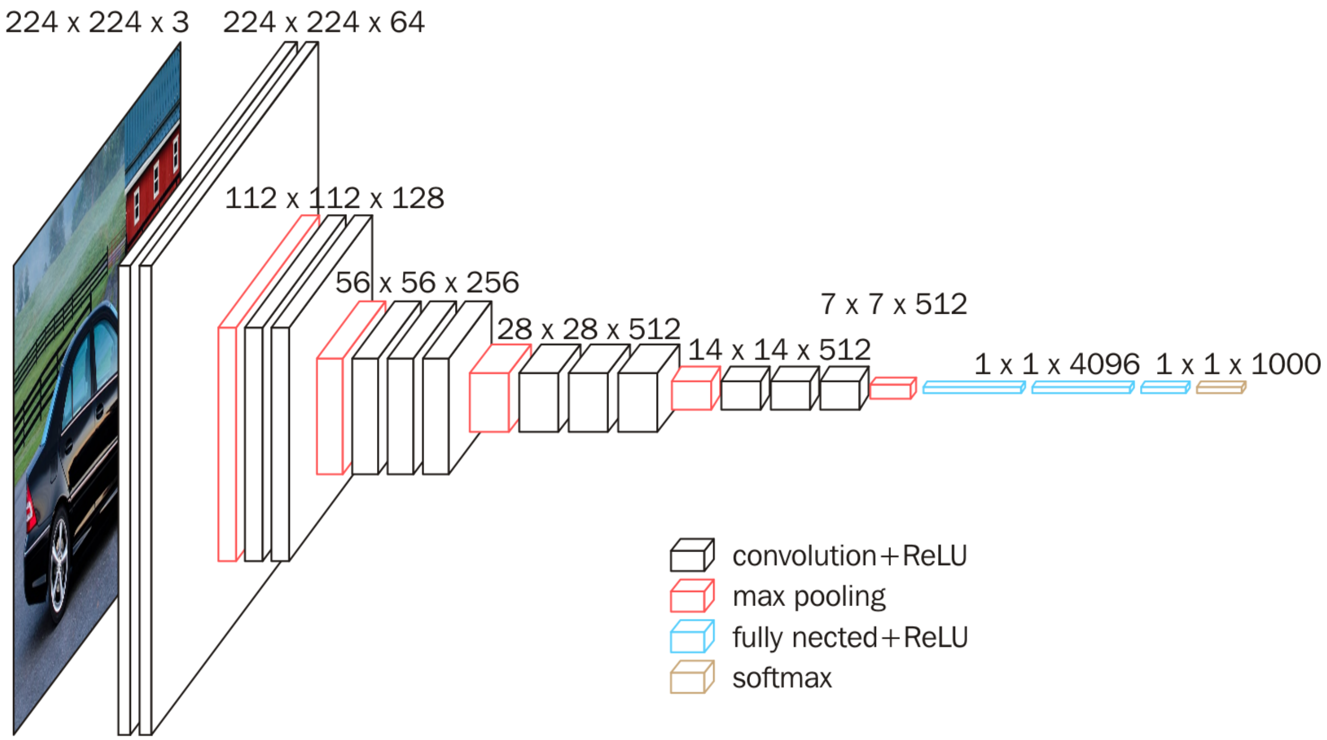


Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

ResNet

Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

- Reformulate the layers as learning residual functions with reference to the layer inputs
- gain accuracy from considerably increased depth ~152
- Lower complexity ~2M compared to VGG16
- 1st place on the ILSVRC 2015 classification task

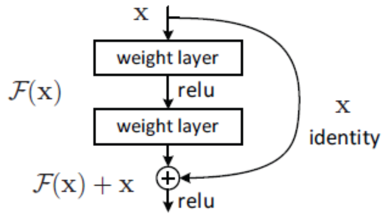
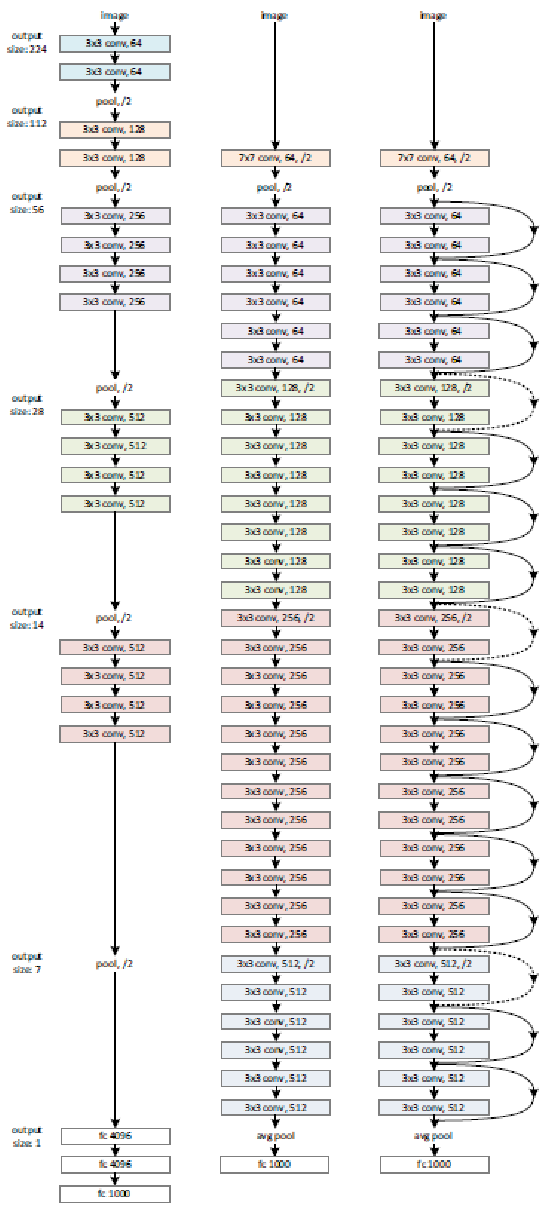


Figure 2. Residual learning: a building block.

model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

Table 3. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.



DenseNet

Densely Connected Convolutional Networks

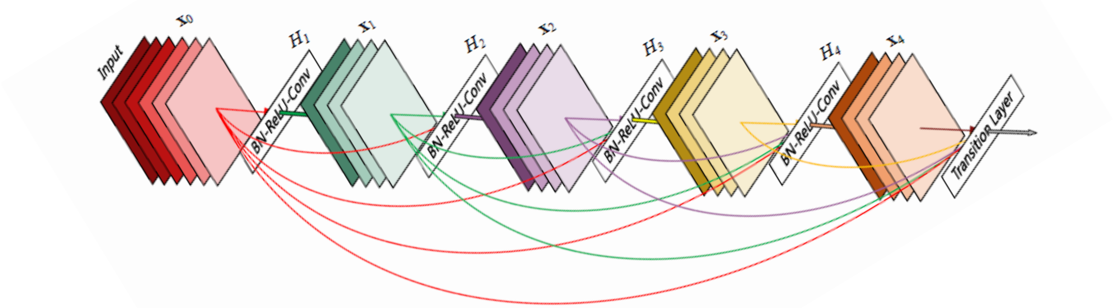
Gao Huang*
Cornell University
gh349@cornell.edu

Zhuang Liu*
Tsinghua University
liuzhuang13@mails.tsinghua.edu.cn

Laurens van der Maaten
Facebook AI Research
lvdmaaten@fb.com

Kilian Q. Weinberger
Cornell University
kqw4@cornell.edu

- Connects each layer to every other layer in a feed-forward fashion: L(L+1)/2 layers
- Alleviates vanishing gradient problem
- Strengthen feature propagation
- Reduction of parameters
- Trained on CIFAR10/100



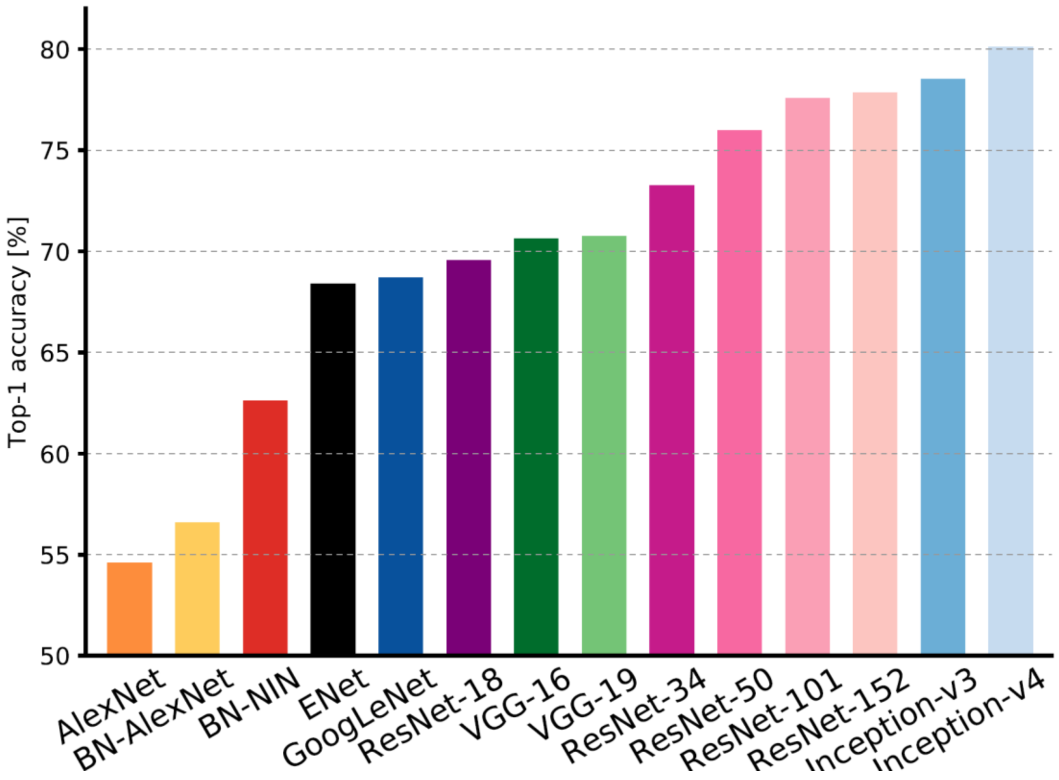
Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k=12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k=12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k=24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k=12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k=24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k=40$)	190	25.6M	-	**3.46**	-	**17.18**	-

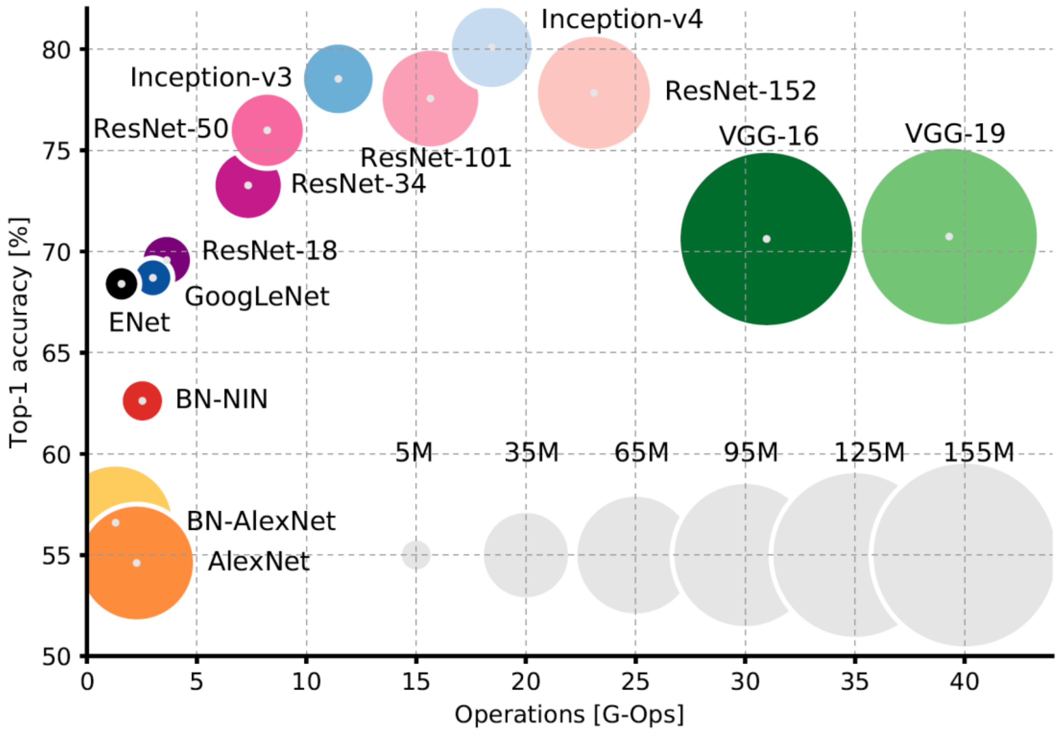
Table 2: Error rates (%) on CIFAR and SVHN datasets. k denotes network's growth rate. Results that surpass all competing methods are **bold** and the overall best results are blue. "+" indicates standard data augmentation (translation and/or mirroring). $*$ indicates results run by ourselves. All the results of DenseNets without data augmentation (C10, C100, SVHN) are obtained using Dropout. DenseNets achieve lower error rates while using fewer parameters than ResNet. Without data augmentation, DenseNet performs better by a large margin.

Overall Comparison on CNNs

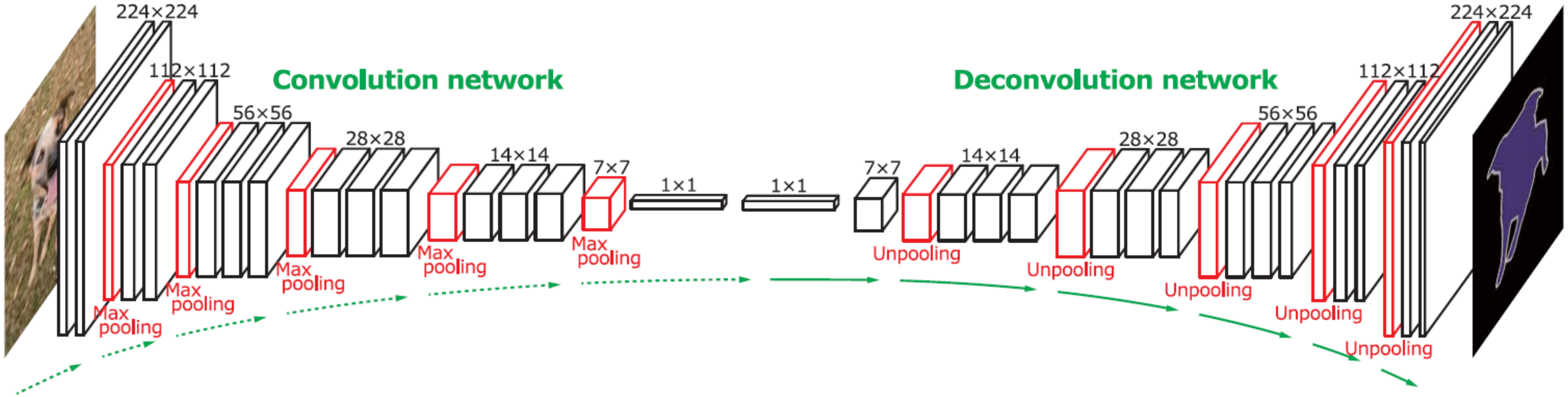
- Width vs Depth
  - shallow networks can require exponentially more components than deeper networks i.e. more learning weights
  - Lets make each layer components as thin as possible in favor of increasing network depth
  - Such design however could be used against its performance
    - Limited information flow during back-propagation
    - Few blocks share informative representations and the rest become redundant
  - Widening the block can compensate such information loss (Wide ResNet)
  - Are we going back to initial configurations?
- Second order minimization (curvature matrix) takes less iteration for training convergence

Overall Comparison on CNNs





Application Example: Semantic Segmentation



- Learning a deconvolution network on top of VGG16

Learning Deconvolution Network for Semantic Segmentation

Hyeonwoo Noh Seunghoon Hong Bohyung Han
Department of Computer Science and Engineering, POSTECH, Korea
{hyeonwoonoh_,maga33,bhhan}@postech.ac.kr

Table 1. Evaluation results on PASCAL VOC 2012 test set. (Asterisk (∗) denotes the algorithms trained with additional data.)

Method	bkg	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbk	person	plant	sheep	sofa	train	tv	mean
Hypercolumn [10]	88.9	68.4	27.2	68.2	47.6	61.7	76.9	72.1	71.1	24.3	59.3	44.8	62.7	59.4	73.5	70.6	52.0	63.0	38.1	60.0	54.1	59.2
MSRA-CFM [3]	87.7	75.7	26.7	69.5	48.8	65.6	81.0	69.2	73.3	30.0	68.7	51.5	69.1	68.1	71.7	67.5	50.4	66.5	44.4	58.9	53.5	61.8
FCN8s [17]	91.2	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	72.4	37.4	70.9	55.1	62.2
TTI-Zoomout-16 [18]	89.8	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	70.2	40.2	68.9	55.3	64.4
DeepLab-CRF [1]	93.1	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**	82.2	50.4	73.1	63.7	71.6
DeconvNet	92.7	85.9	42.6	78.9	62.5	66.6	87.4	77.8	79.5	26.3	73.4	60.2	70.8	76.5	79.6	77.7	58.2	77.4	52.9	75.2	59.8	69.6
DeconvNet+CRF	92.9	87.8	41.9	80.6	63.9	67.3	**88.1**	78.4	81.3	25.9	73.7	61.2	72.0	77.0	79.9	78.7	59.5	78.3	**55.0**	75.2	61.5	70.5
EDeconvNet	92.9	88.4	39.7	79.0	63.0	67.7	87.1	**81.5**	84.4	27.8	76.1	61.2	78.0	79.3	83.1	79.3	58.0	82.5	52.3	80.1	64.0	71.7
EDeconvNet+CRF	93.1	**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	**83.4**	54.3	**80.7**	65.0	**72.5**
* WSSL [19]	93.2	85.3	36.2	**84.8**	61.2	67.5	84.7	81.4	81.0	**30.8**	73.8	53.8	77.5	76.5	82.3	**81.6**	56.3	78.9	52.3	76.6	63.3	70.4
* BoxSup [2]	**93.6**	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	78.2	**55.0**	72.5	**68.1**	71.0

Application Example: Image Super-Resolution

- Single Image Super Resolution (SISR)
- Upsample image using cubic spline
- Feed the upsampled image to network and recover the high resolution
- Number of layers = 3/4

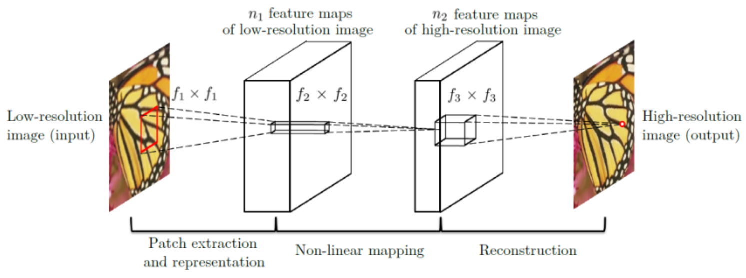


Image Super-Resolution Using Deep Convolutional Networks

Chao Dong, Chen Change Loy, *Member, IEEE,* Kaiming He, *Member, IEEE,* and Xiaoou Tang, *Fellow, IEEE*

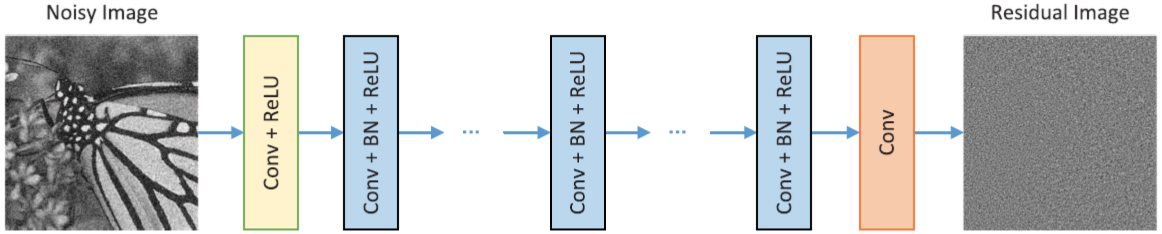
Application Example: Image Denoising

- Aim to learn residuals
- Train noisy image set is synthetically build by additive white Gaussian noise with various levels
- No pooling is involved
- Number of layers = 17
- Split the image to multiple patches 40x40 for training
- Any residual artifacts could be learned e.g. SISR, JPEG, etc.

3142 IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 26, NO. 7, JULY 2017

Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising

Kai Zhang, Wangmeng Zuo, *Senior Member, IEEE*, Yunjin Chen, Deyu Meng, and Lei Zhang, *Senior Member, IEEE*

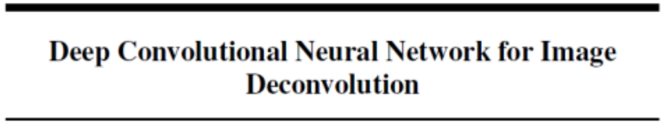


THE PSNR(dB) RESULTS OF DIFFERENT METHODS ON 12 WIDELY USED TESTING IMAGES

Images	C.man	House	Peppers	Starfish	Monar.	Airpl.	Parrot	Lena	Barbara	Boat	Man	Couple	Average
Noise Level	$\sigma = 15$												
BM3D [2]	31.91	34.93	32.69	31.14	31.85	31.07	31.37	34.26	33.10	32.13	31.92	32.10	32.372
WNNM [13]	32.17	**35.13**	32.99	31.82	32.71	31.39	31.62	34.27	**33.60**	32.27	32.11	32.17	32.696
EPLL [33]	31.85	34.17	32.64	31.13	32.10	31.19	31.42	33.92	31.38	31.93	32.00	31.93	32.138
CSF [14]	31.95	34.39	32.85	31.55	32.33	31.33	31.37	34.06	31.92	32.01	32.08	31.98	32.318
TNRD [16]	32.19	34.53	33.04	31.75	32.56	31.46	31.63	34.24	32.13	32.14	32.23	32.11	32.502
DnCNN-S	**32.61**	34.97	**33.30**	**32.20**	**33.09**	**31.70**	**31.83**	**34.62**	32.64	**32.42**	**32.46**	**32.47**	**32.859**
DnCNN-B	32.10	34.93	33.15	32.02	32.94	31.56	31.63	34.56	32.09	32.35	32.41	32.41	32.680
Noise Level	$\sigma = 25$												
BM3D [2]	29.45	32.85	30.16	28.56	29.25	28.42	28.93	32.07	30.71	29.90	29.61	29.71	29.969
WNNM [13]	29.64	**33.22**	30.42	29.03	29.84	28.69	29.15	32.24	**31.24**	30.03	29.76	29.82	30.257
EPLL [33]	29.26	32.17	30.17	28.51	29.39	28.61	28.95	31.73	28.61	29.74	29.66	29.53	29.692
MLP [24]	29.61	32.56	30.30	28.82	29.61	28.82	29.25	32.25	29.54	29.97	29.88	29.73	30.027
CSF [14]	29.48	32.39	30.32	28.80	29.62	28.72	28.90	31.79	29.03	29.76	29.71	29.53	29.837
TNRD [16]	29.72	32.53	30.57	29.02	29.85	28.88	29.18	32.00	29.41	29.91	29.87	29.71	30.055
DnCNN-S	**30.18**	33.06	**30.87**	**29.41**	**30.28**	**29.13**	**29.43**	**32.44**	30.00	**30.21**	**30.10**	**30.12**	**30.436**
DnCNN-B	29.94	33.05	30.84	29.34	30.25	29.09	29.35	32.42	29.69	30.20	30.09	30.10	30.362
Noise Level	$\sigma = 50$												
BM3D [2]	26.13	29.69	26.68	25.04	25.82	25.10	25.90	29.05	27.22	26.78	26.81	26.46	26.722
WNNM [13]	26.45	**30.33**	26.95	25.44	26.32	25.42	26.14	29.25	**27.79**	26.97	26.94	26.64	27.052
EPLL [33]	26.10	29.12	26.80	25.12	25.94	25.31	25.95	28.68	24.83	26.74	26.79	26.30	26.471
MLP [24]	26.37	29.64	26.68	25.43	26.26	25.56	26.12	29.32	25.24	27.03	27.06	26.67	26.783
TNRD [16]	26.62	29.48	27.10	25.42	26.31	25.59	26.16	28.93	25.70	26.94	26.98	26.50	26.812
DnCNN-S	**27.03**	30.00	27.32	25.70	26.78	25.87	**26.48**	**29.39**	26.22	27.20	**27.24**	26.90	27.178
DnCNN-B	**27.03**	30.02	**27.39**	**25.72**	**26.83**	**25.89**	**26.48**	29.38	26.38	**27.23**	27.23	**26.91**	**27.206**

Application Example: Image Deblurring

- Capable of addressing variety of blurring effects e.g. optical blur and motion blur



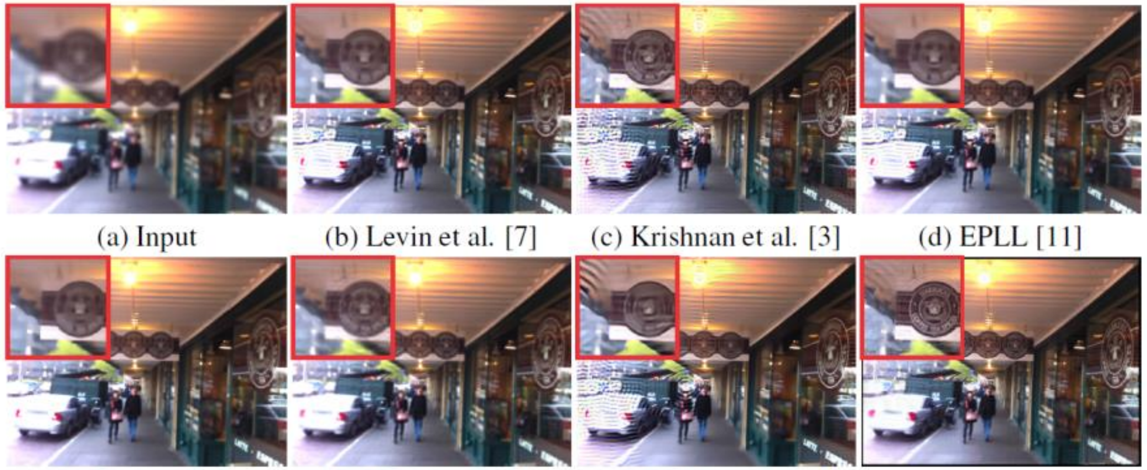
Deep Convolutional Neural Network for Image Deconvolution

Li Xu *
Lenovo Research & Technology
xulihk@lenovo.com

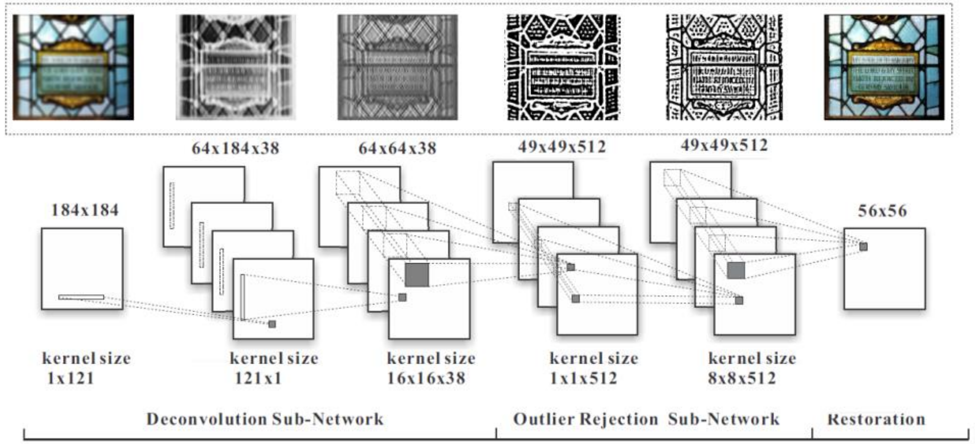
Jimmy SJ. Ren
Lenovo Research & Technology
jimmy.sj.ren@gmail.com

Ce Liu
Microsoft Research
celiu@microsoft.com

Jiaya Jia
The Chinese University of Hong Kong
leojia@cse.cuhk.edu.hk



(a) Input (b) Levin et al. [7] (c) Krishnan et al. [3] (d) EPLL [11]
(e) Cho et al. [9] (f) Whyte et al. [10] (g) Schuler et al. [13] (h) Ours



Thank you!