Deep Learning Approaches in Computational Drug Discovery

Jung-Hsin Lin (林榮信)

Research Center for Applied Sciences &
Institute of Biomedical Sciences, Academia Sinica
School of Pharmacy, National Taiwan University
College of Engineering, Chang Gung University

http://www.rcas.sinica.edu.tw/faculty/jhlin.html

2018 Frontiers in Computational Drug Design, Academia Sinica, March 16-20, 2018

Drug Discovery Today • Volume 00, Number 00 • February 2018 REVIEWS

ELSEVIER

Reviews • INFORMATICS

The rise of deep learning in drug discovery

Hongming Chen[1], Ola Engkvist[1], Yinhai Wang[2], Marcus Olivecrona[1] and Thomas Blaschke[1]

[1] Hit Discovery, Discovery Sciences, Innovative Medicines and Early Development Biotech Unit, AstraZeneca R&D Gothenburg, Mölndal 43183, Sweden
[2] Quantitative Biology, Discovery Sciences, Innovative Medicines and Early Development Biotech Unit, AstraZeneca, Unit 310, Cambridge Science Park, Milton Road, Cambridge CB4 0WG, UK

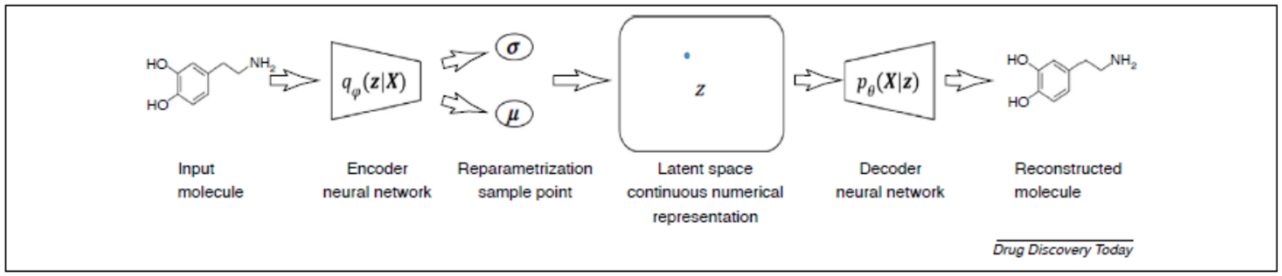


FIGURE 4
The illustration of a variational autoencoder (VAE) method. The encoder neural network (NN) converts a discrete molecule into Gaussian distribution deterministically. After the latent variables are reparameterized against the gaussian distribution with given mean and variance, a new point is sampled and fed into the decoder NN. In the generation mode, only the decoder is used to generate a new molecule from the sampled latent point.

Review

www.molinf.com

molecular informatics

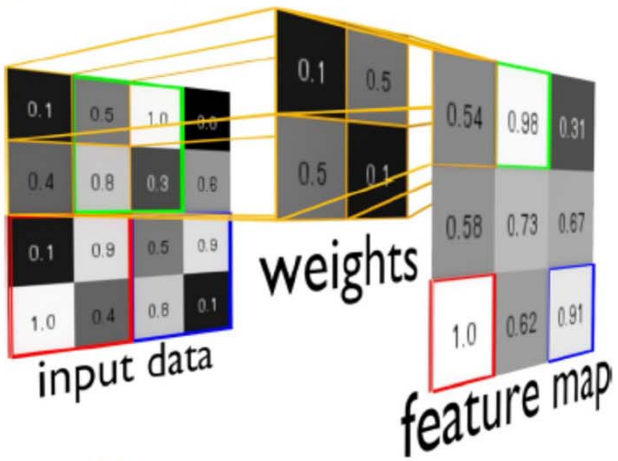
DOI: 10.1002/minf.201501008

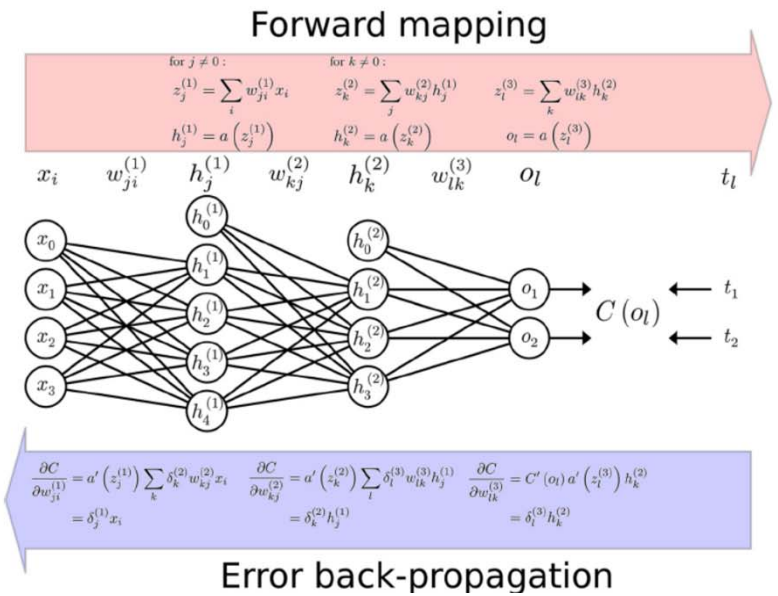
Deep Learning in Drug Discovery

Erik Gawehn,[a] Jan A. Hiss,[a] and Gisbert Schneider*[a]

Abstract: Artificial neural networks had their first heyday in molecular informatics and drug discovery approximately two decades ago. Currently, we are witnessing renewed interest in adapting advanced neural network architectures for pharmaceutical research by borrowing from the field of "deep learning". Compared with some of the other life sciences, their application in drug discovery is still limited. Here, we provide an overview of this emerging field of molecular informatics, present the basic concepts of prominent deep learning methods and offer motivation to explore these techniques for their usefulness in computer-assisted drug discovery and design. We specifically emphasize deep neural networks, restricted Boltzmann machine networks and convolutional networks.

Keywords: bioinformatics · cheminformatics · drug design · machine-learning · neural network · virtual screening





Journal of COMPUTATIONAL CHEMISTRY WWW.C-CHEM.ORG REVIEW

Deep Learning for Computational Chemistry

Garrett B. Goh,*[a] Nathan O. Hodas,[b] and Abhinav Vishnu[a]

Wiley Online Library *Journal of Computational Chemistry* **2017**, *38*, 1291–1307

Editorial www.molinf.com molecular informatics

Generative Models for Artificially-intelligent Molecular Design

Gisbert Schneider[a]

Gisbert Schneider

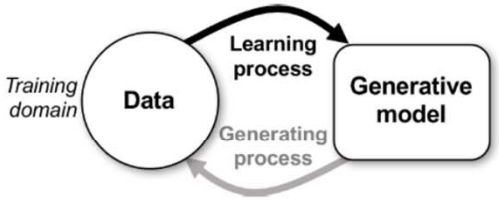
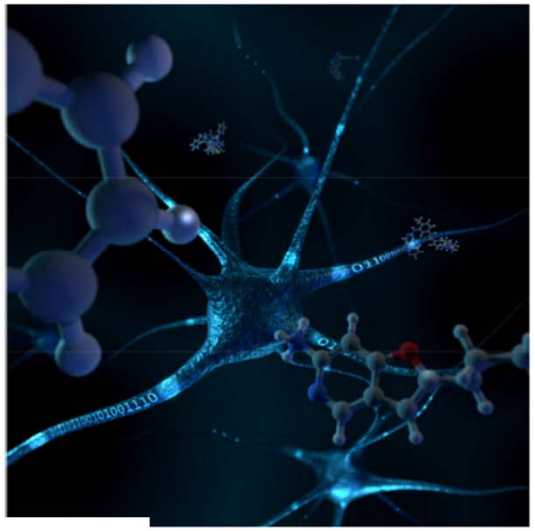


Figure 1. Schematic of generative modeling. Data distributions are learned by a generative model, which is able to generate new data instances based on the learned internal representation of the training domain. Such an approach may be considered artificially-intelligent, bearing promise for drug design.



Wiley Online Library *Mol. Inf.* **2018**, *37*, 1880131

AlphaGo, AlphaGo Zero, Alpha Zero

AlphaGo is a computer programthat plays the board game Go. It was developed by Alphabet Inc.'s Google DeepMind in London.

At the 2017 Future of Go Summit, AlphaGo beat Ke, Jie, the world No.1 ranked player at the time, in a three-game match.

AlphaGo Zero is a version of DeepMind's Go software AlphaGo. AlphaGo's team published an article in the journal *Nature* on 19 October 2017, introducing AlphaGo Zero, a version created without using data from human games, and stronger than any previous version.

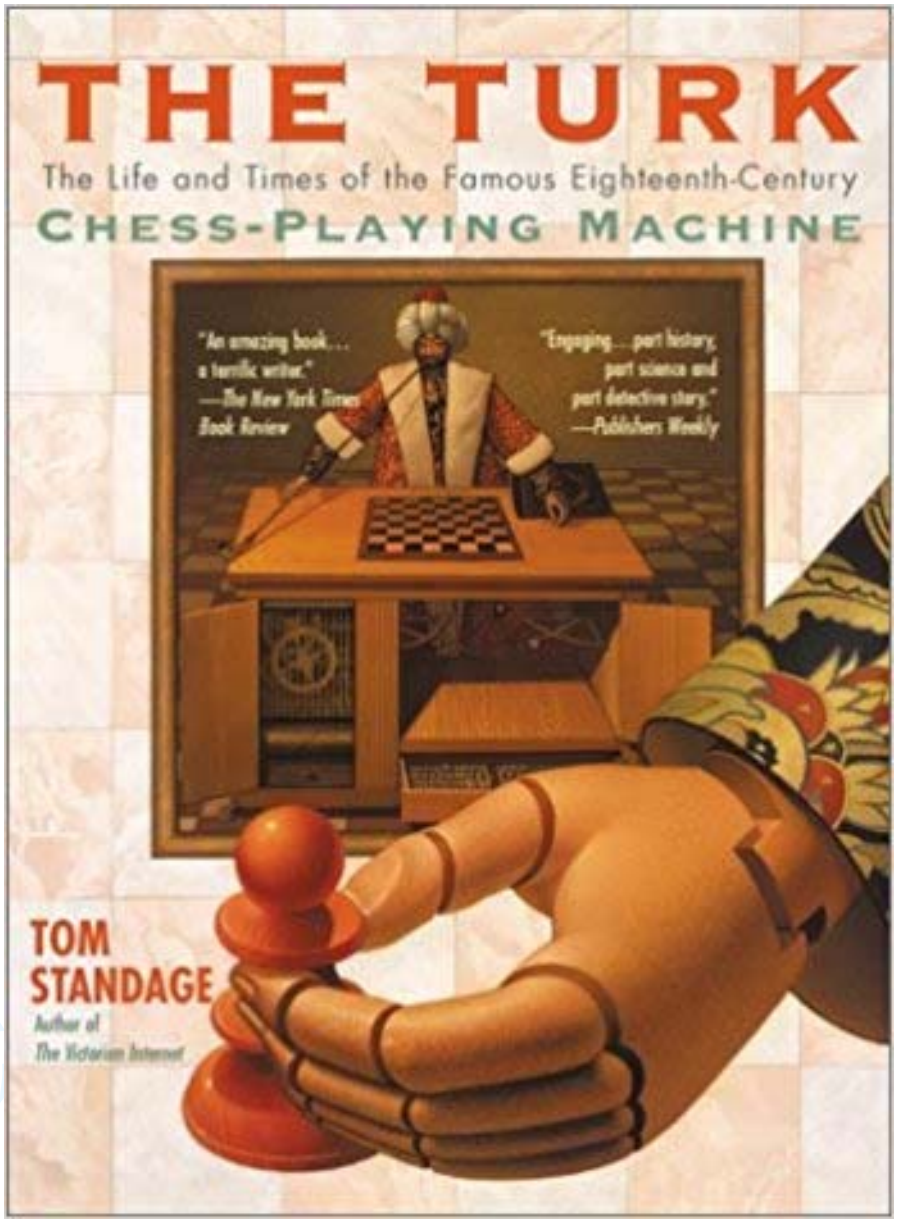
In December 2017, a generalized version of AlphaGo Zero, named **AlphaZero**, beat the 3-day version of AlphaGo Zero by winning 60 games to 40.

AlphaGo

Deep Blue

Deep Blue was a chess-playing computer developed by IBM. It is known for being the first computer chess-playing system to win both a chess game and a chess match against a reigning world champion under regular time controls.

Deep Blue won its first game against a world champion on 10 February 1996, when it defeated Garry Kasparov in game one of a six-game match.


THE TURK
The Life and Times of the Famous Eighteenth-Century
CHESS-PLAYING MACHINE
"An amazing book... a terrific writer."
—The New York Times Book Review
"Engaging... part history, part science and part detective story."
—Publishers Weekly
TOM STANDAGE
Author of
The Victorian Internet

Ernst Theodor Wilhelm Hoffmann
24 January 1776 - 25 June 1822

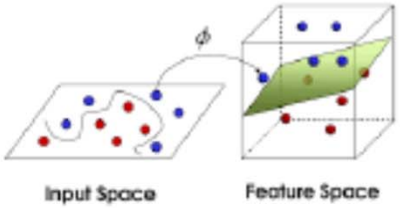
Jacques Offenbach
20 June 1819 – 5 October 1880

Conventional approach to object recognition

- Training phase

image collection → feature extraction → classifier training → trained classifier

histogram of oriented gradients (HoG)



support vector machines (SVMs)

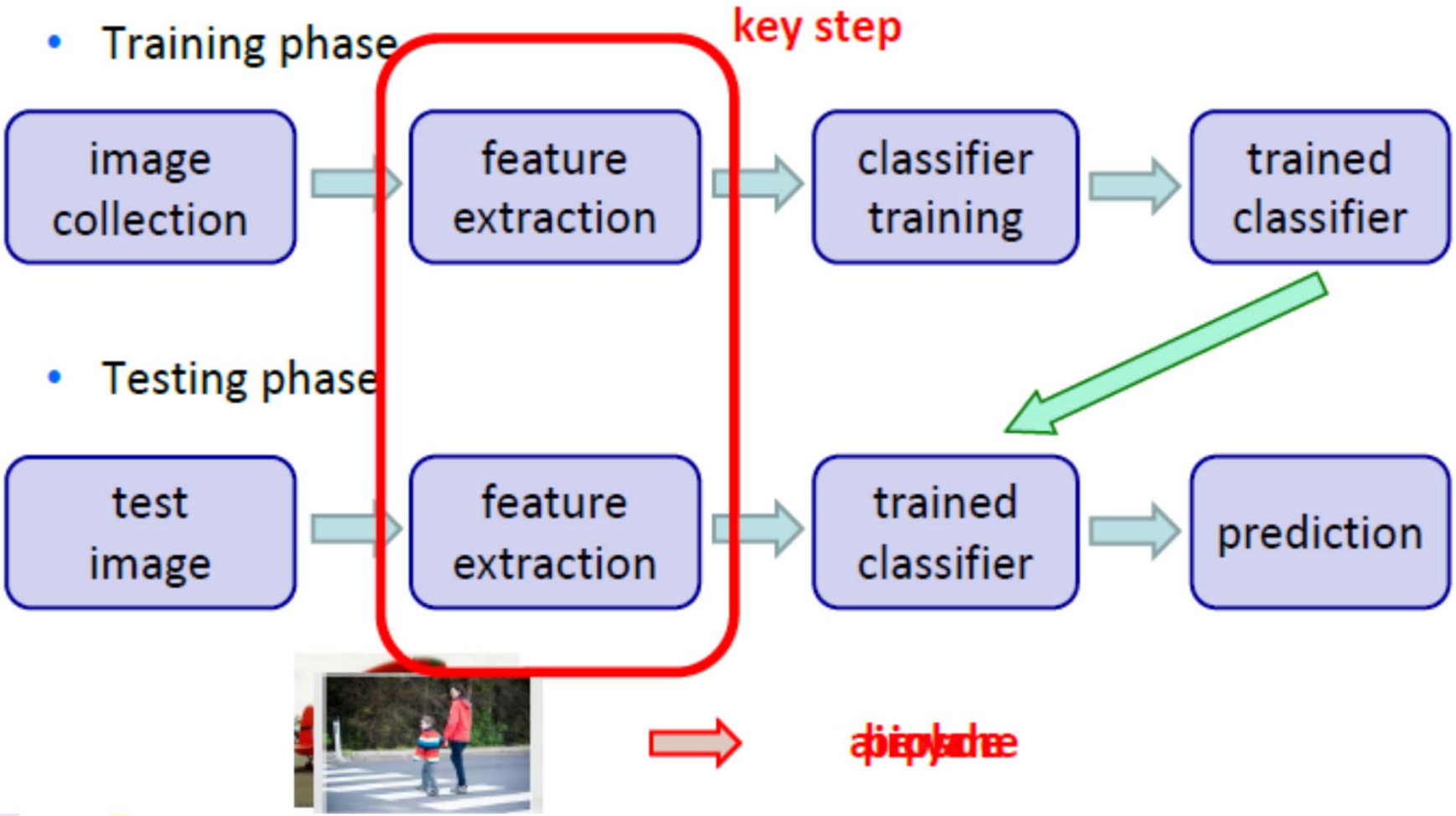
dogs? ✖

flowers? ○

trains? ✖

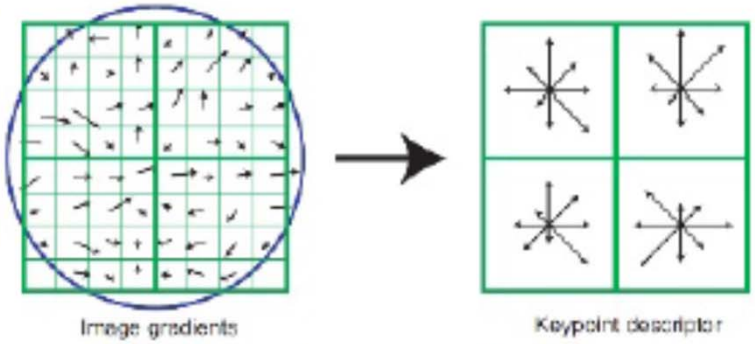
persons? ✖

Conventional approach to object recognition



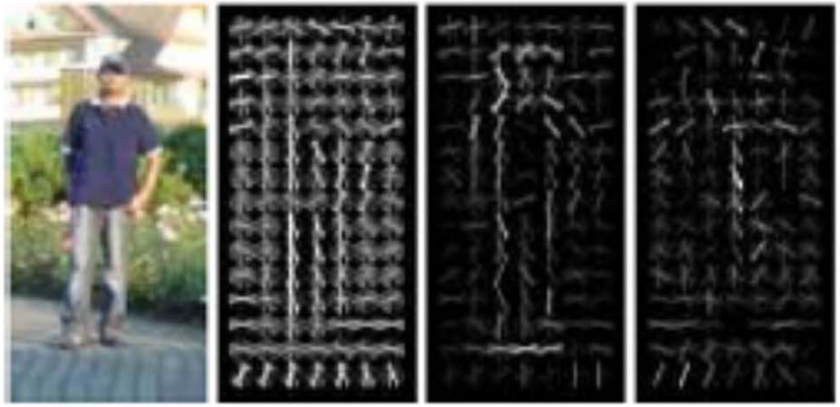
Features are the keys

- Off-the-shelf visual features

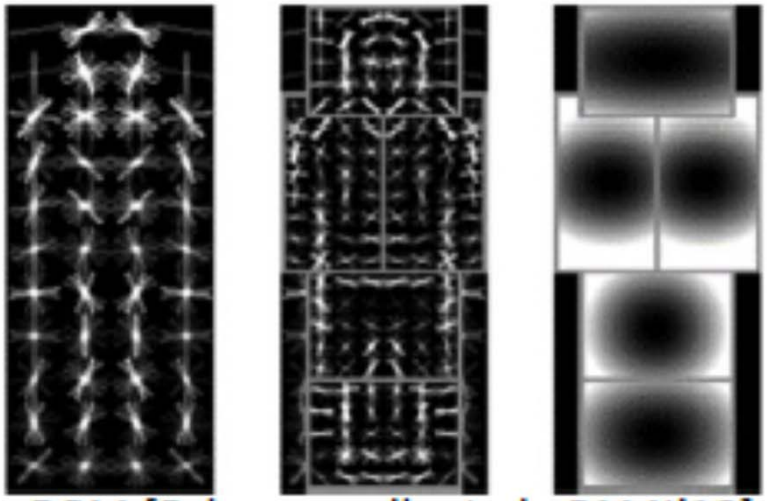


SIFT [Lowe, IJCV'04]
Citations: 43465

Constellation model [Fergus et al., CVPR'03]
Citations: 2551



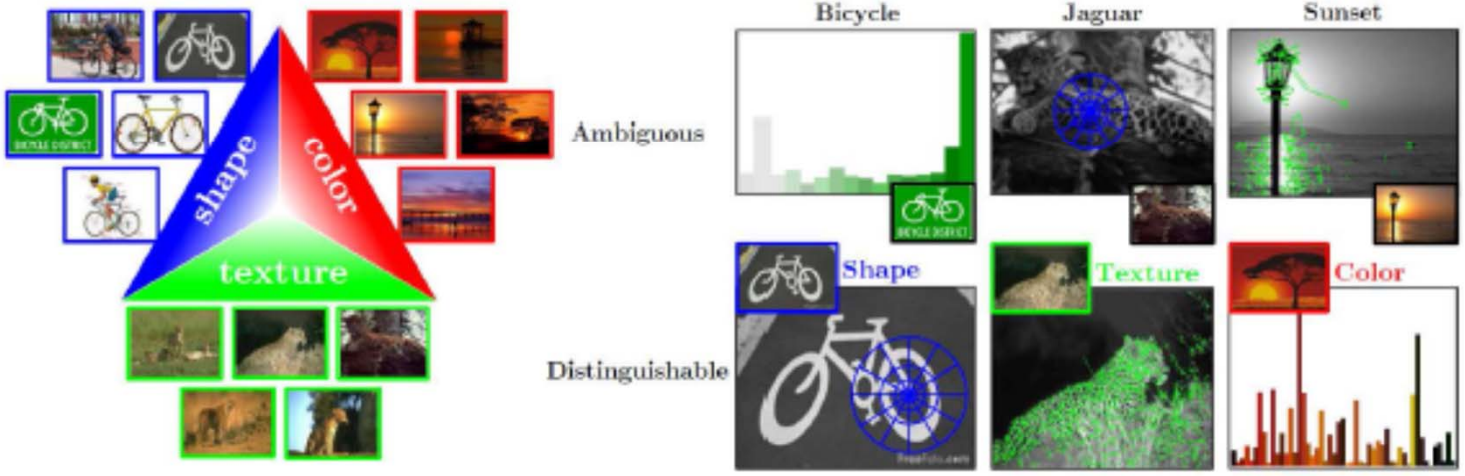
HoG [Dalal & Triggs, CVPR'05]
Citations: 20174



DPM [Felzenszwalb et al., PAMI'10]
Citations: 5093

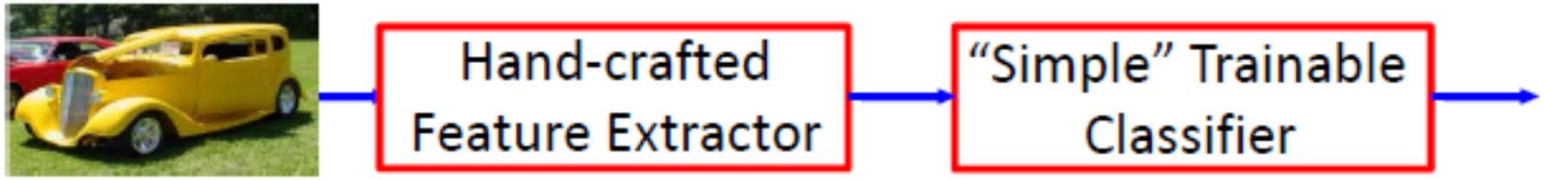
Features are the keys

- Features are the keys to recent progress in classification
- Are handcrafted features optimal?
- The optimal features for classification in general vary from task to task, even from category to category

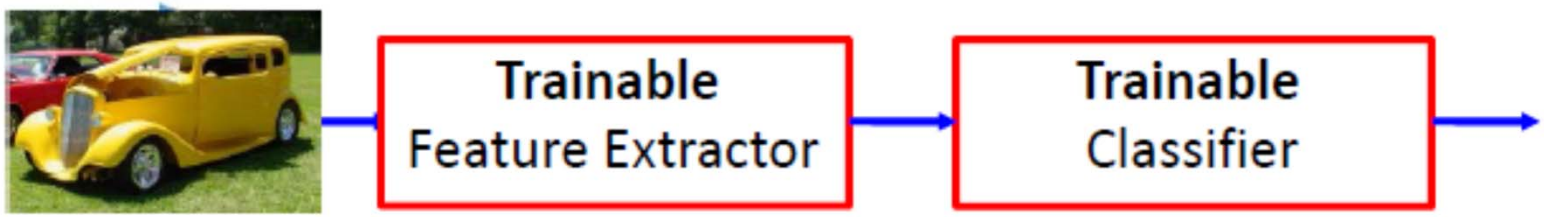


Conventional approaches vs. Deep learning

- Conventional approaches
  - Fixed/engineered features + trainable classifier

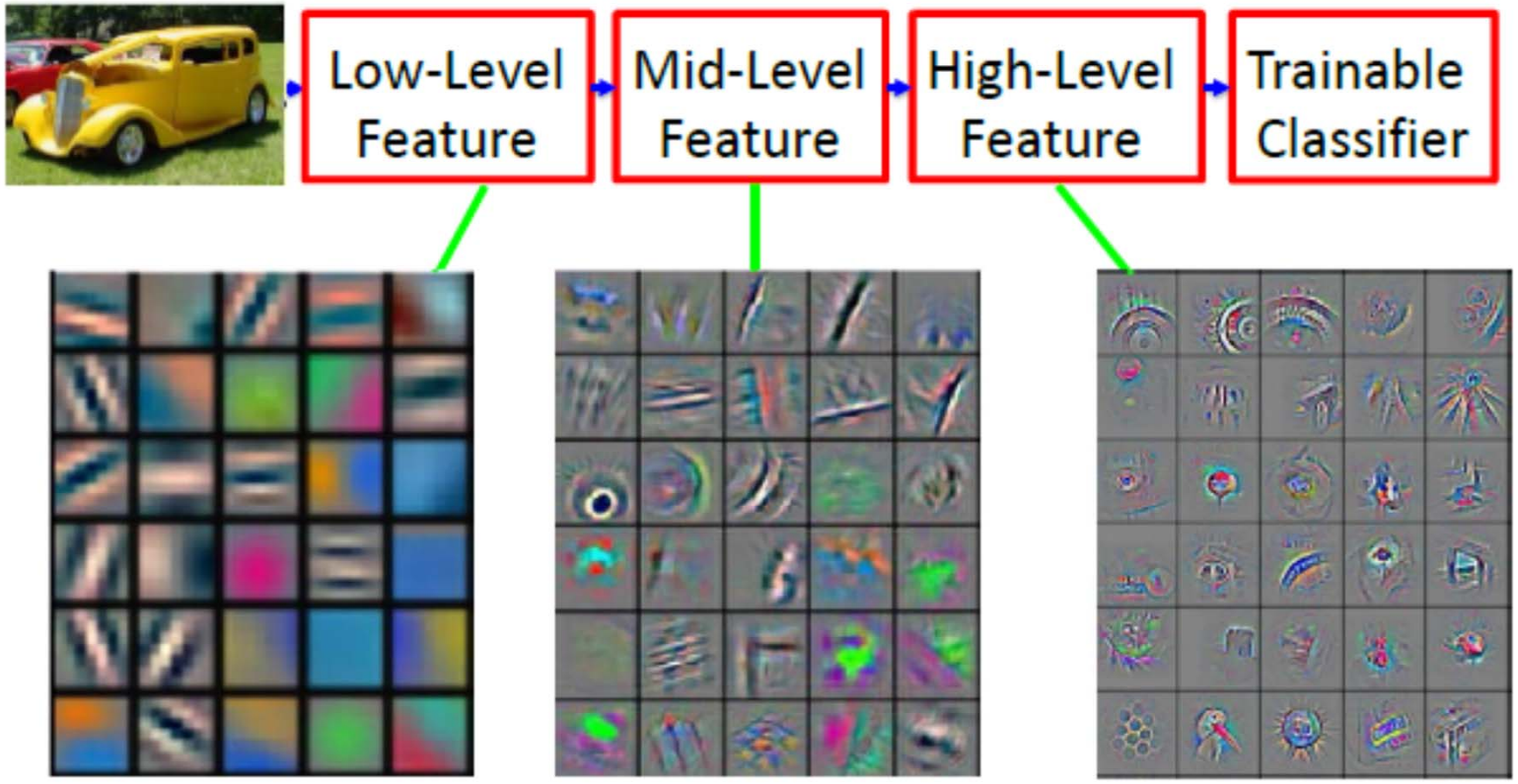


- Deep learning / End-to-end learning / Feature learning
  - Trainable features + trainable classifier



slide: Y LeCun & MA Ranzato

Deep learning = Learning hierarchical representations

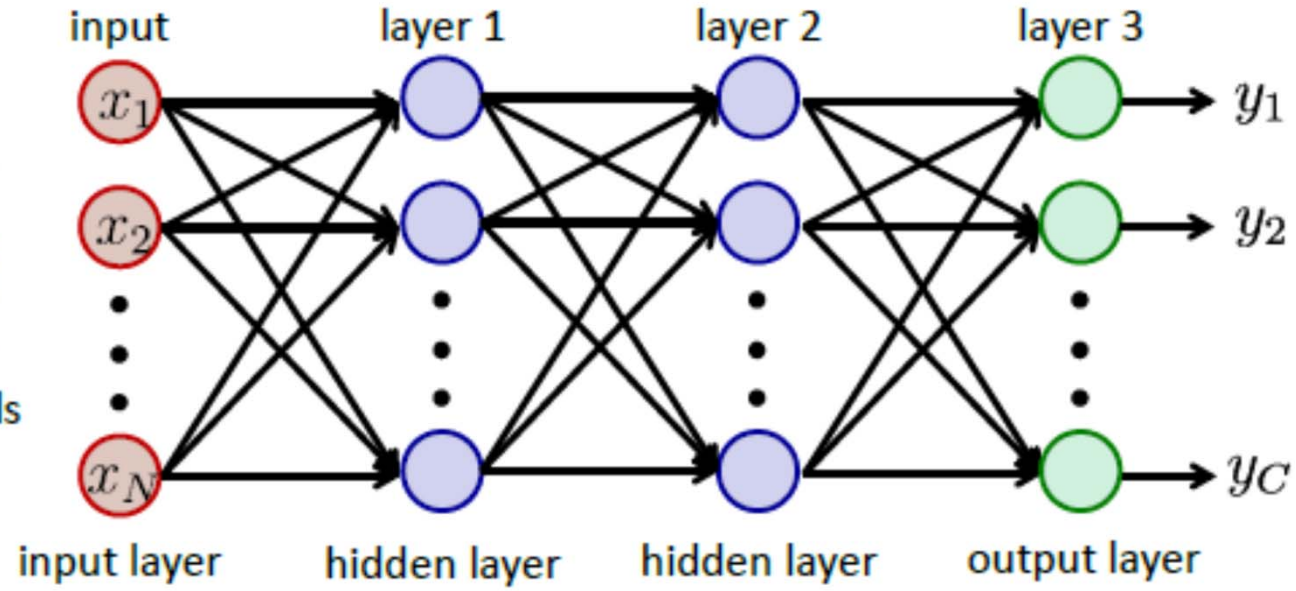


slide: Y LeCun & MA Ranzato

Neural networks and neurons

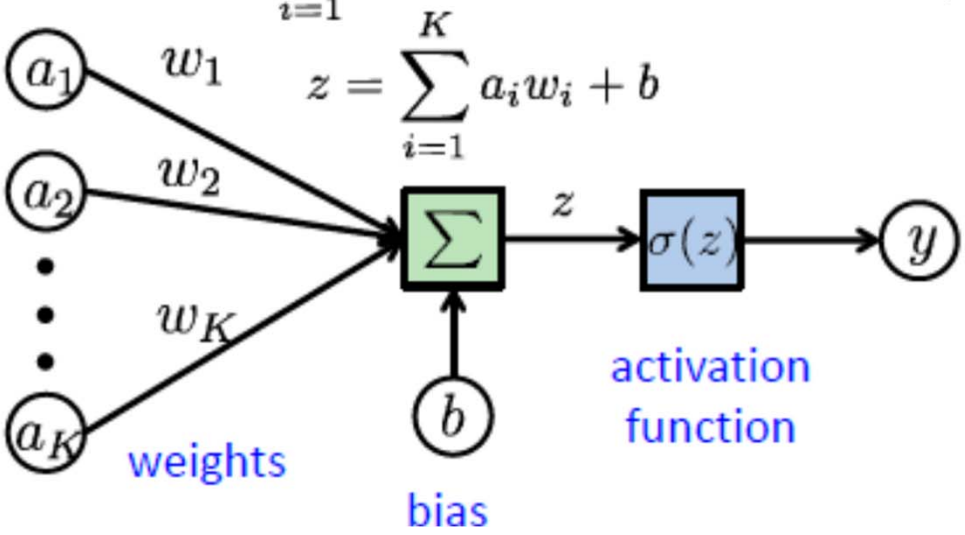
- Neural networks are presented as layers of interconnected neurons
  - Each layer of neurons takes messages from output of previous layer

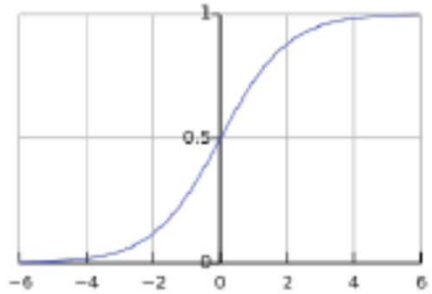
image of N pixels



A single neuron

- A function $f : R^K \mapsto R$
  - Map K inputs to 1 output
  - Compute the biased weighted sum
  - Apply a non-linear mapping function (activation function)
  - $f((a)) = \sigma(\sum_{i=1}^{K} a_i w_i + b)$, where $\sigma(z) = \dfrac{1}{1 + \exp(-z)}$





sigmoid function

Training neural networks

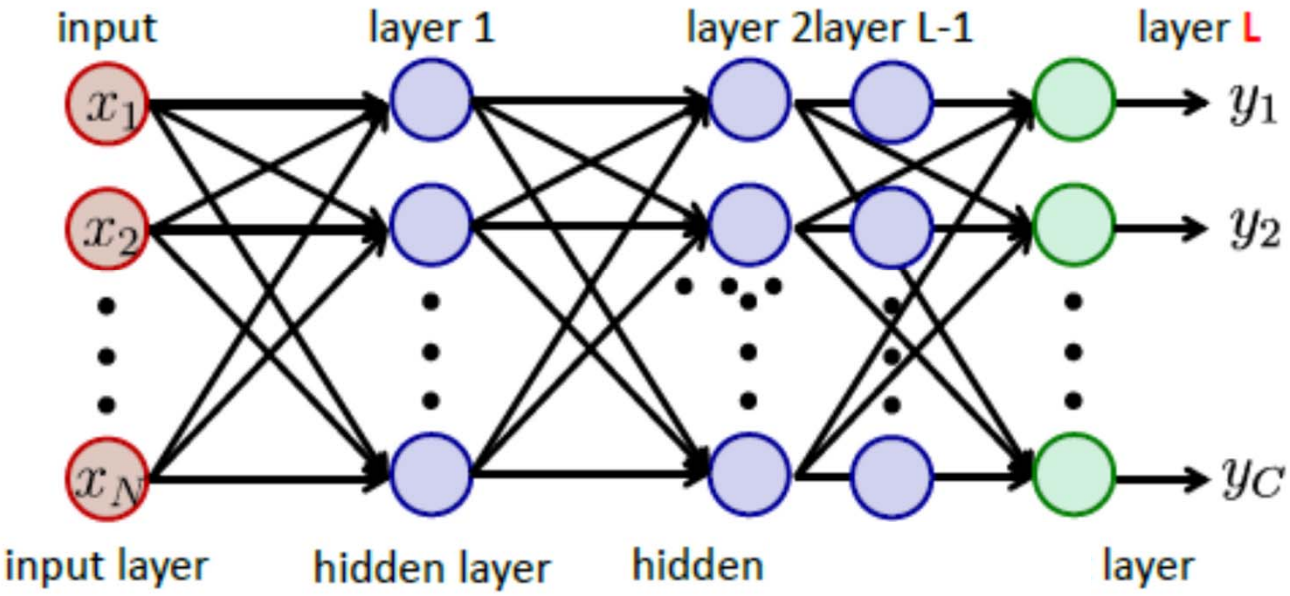
- Collect a set of labeled training data $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$
- Training neural networks: Finding network parameters $\theta = \{\mathbf{w}, \mathbf{b}\}$ to minimize the loss between true training label $\mathbf{y}_i$ and the estimated label, e.g.,

$$L(\theta) = \sum_{i=1}^{N} \|\mathbf{y}_i - g_{\mathbf{w}}(\mathbf{x}_i)\|^2$$

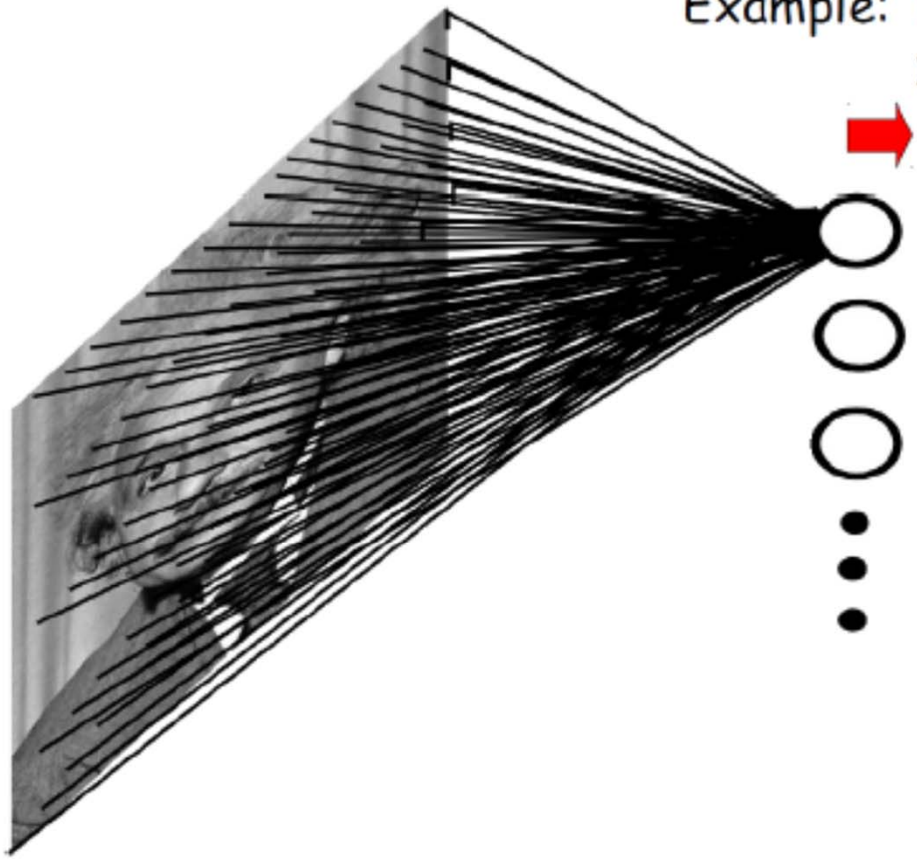
- Minimization can be done by gradient descent if $L(\cdot)$ is differentiable with respect to θ
- Back-propagation: a widely used method for optimizing multi-layer neural networks

What is deep neural networks (DNN)

- DNN is neural networks with many hidden layers



of parameters in fully connected NN



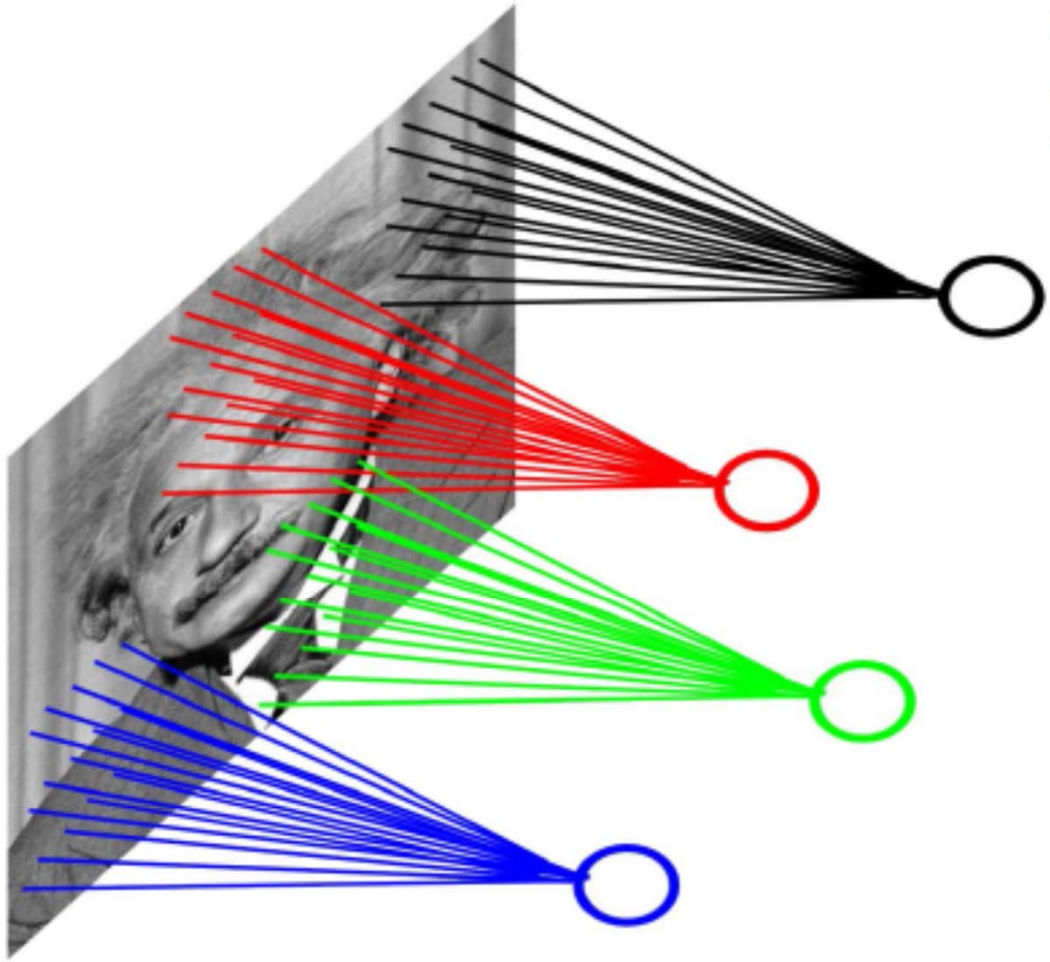
Example: 1000x1000 image
1M hidden units
10^12 parameters!!!

slide: MA Ranzato

Convolutional neural networks (CNN)

- CNN: a multi-layer neural network with
  1. Local connectivity
  2. Weight sharing
- Why local connectivity?
  - Spatial correlation is local (locality of spatial dependencies)
  - Reduce # of parameters
- Why weight sharing?
  - Statistics is at different locations (stationarity of statistics)
  - Reduce # of parameters

of parameters in fully connected NN



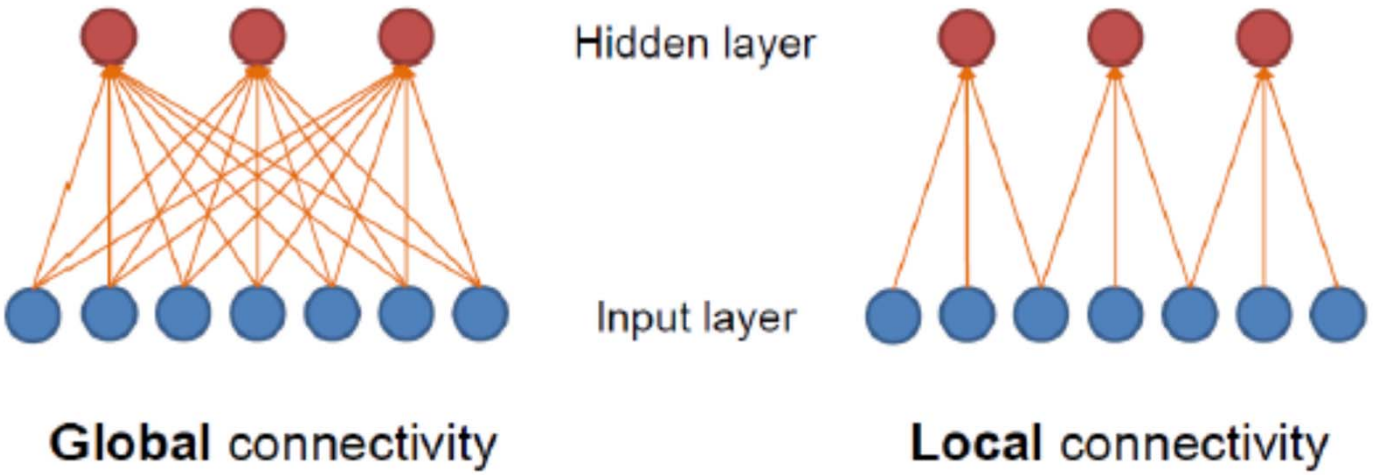
0x1000 image

nidden units

^12 parameters!!!

slide: MA Ranzato

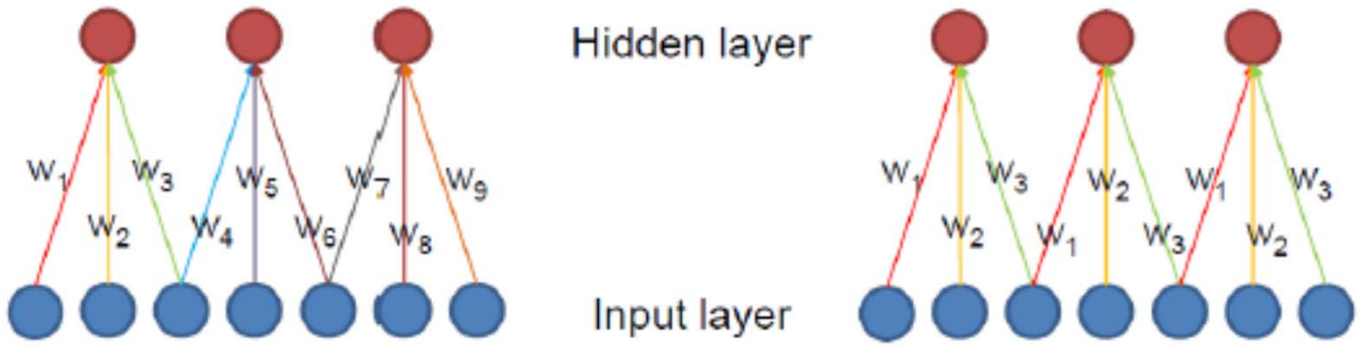
CNN: Local connectivity



- # input units (neurons): 7
- # hidden units: 3
- Number of parameters
  - Global connectivity: 3 x 7 = 21
  - Local connectivity: 3 x 3 = 9

slide: J.-B. Huang

CNN: Weight sharing



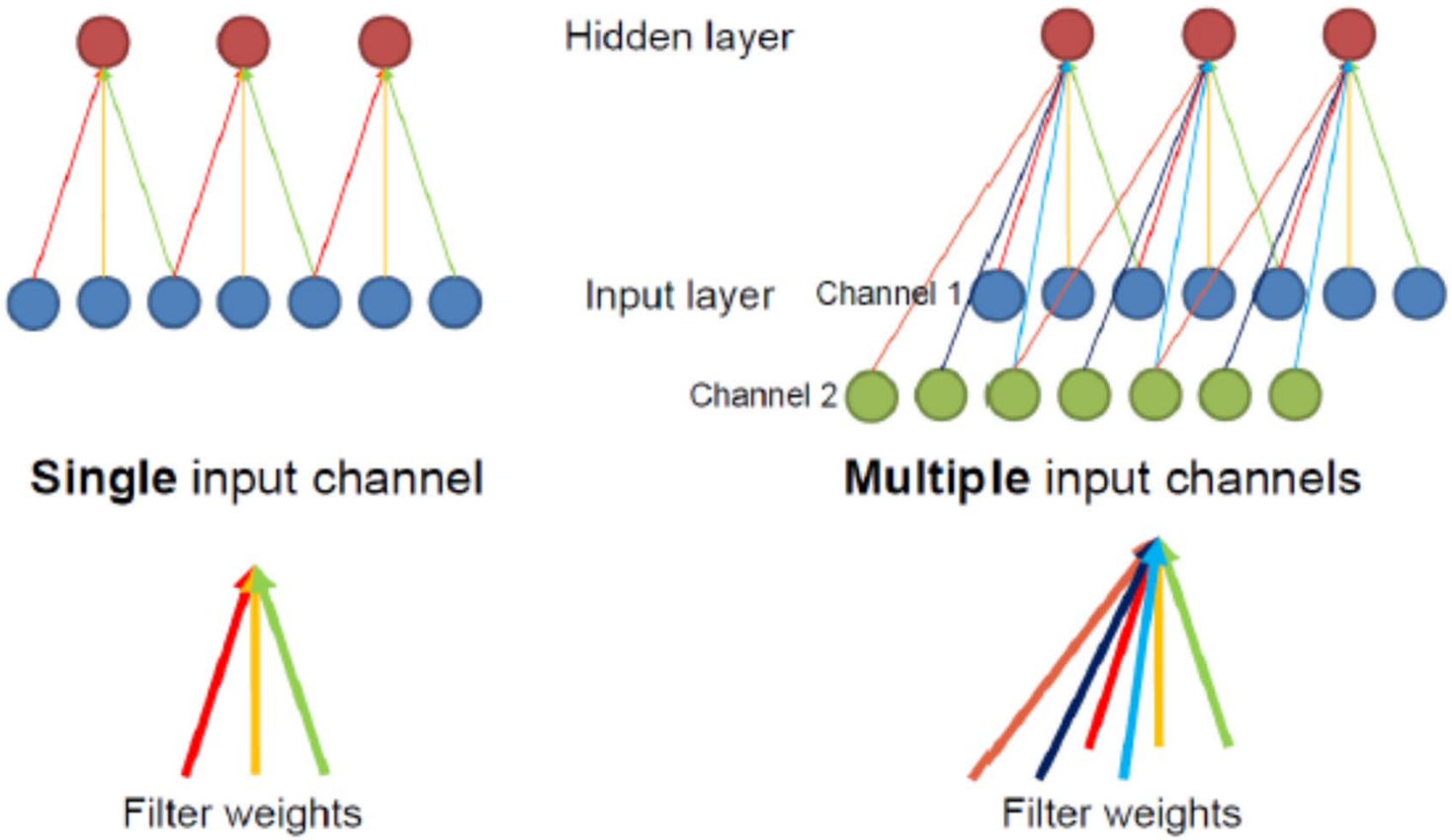
Without weight sharing

With weight sharing

- # input units (neurons): 7
- # hidden units: 3
- Number of parameters
  - Without weight sharing: 3 x 3 = 9
  - With weight sharing : 3 x 1 = 3

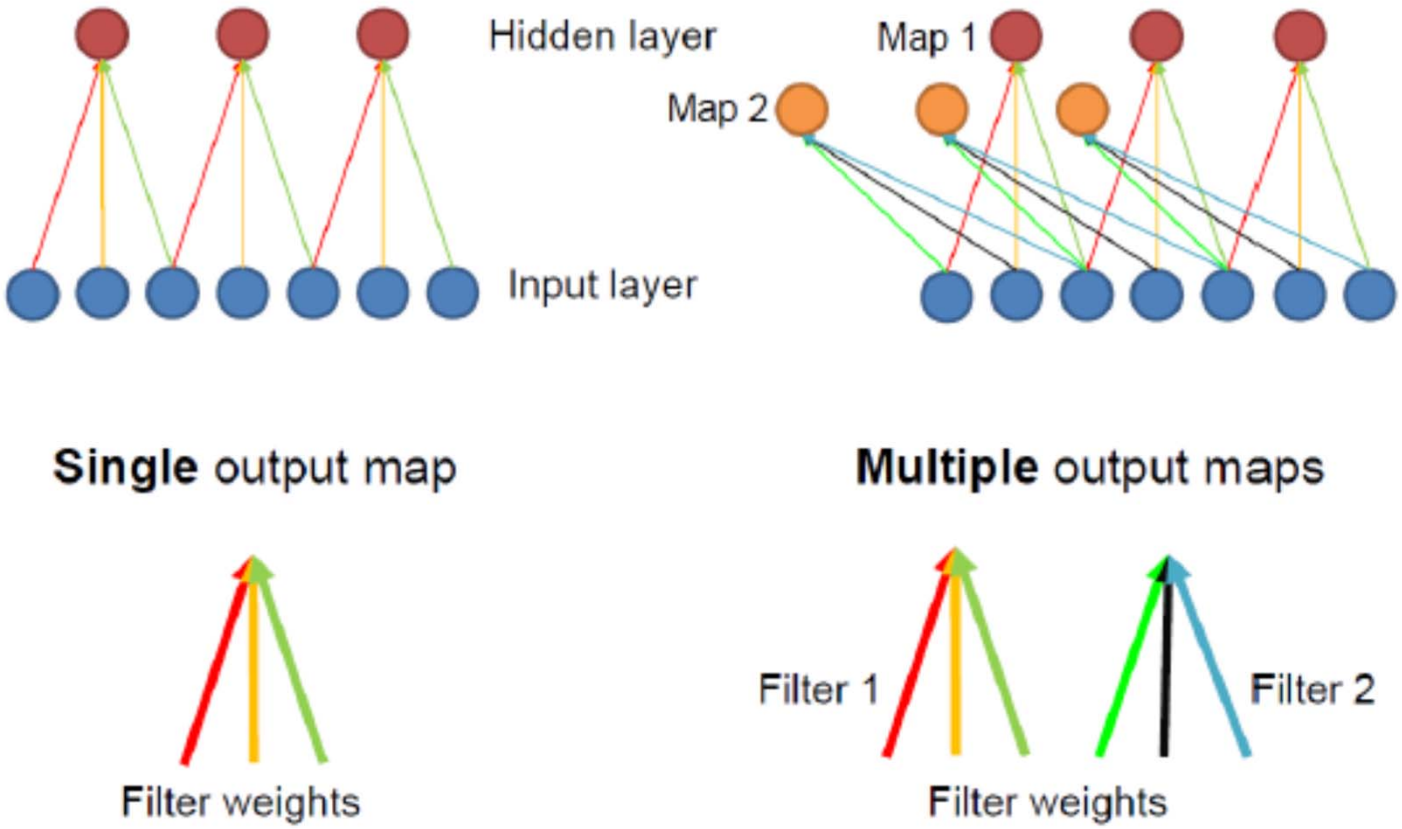
slide: J.-B. Huang

CNN with multiple input channels



slide: J.-B. Huang

CNN with multiple output channels



slide: J.-B. Huang

Putting them together

- Local connectivity
- Weight sharing
- Handling multiple input channels
- Handling multiple output maps

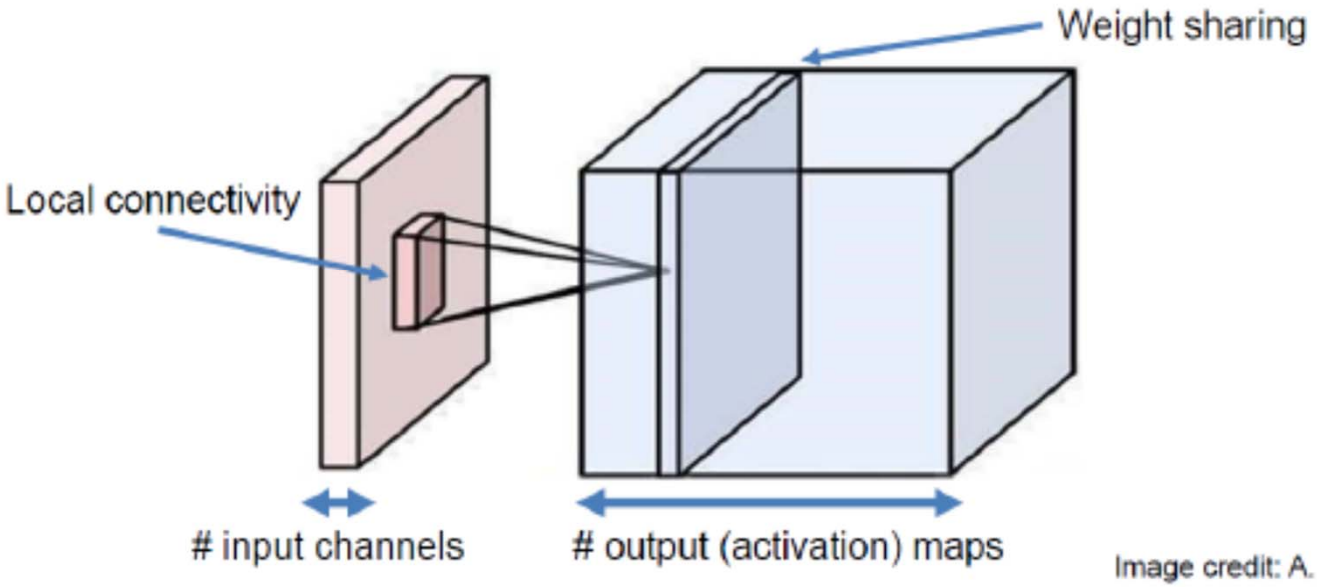
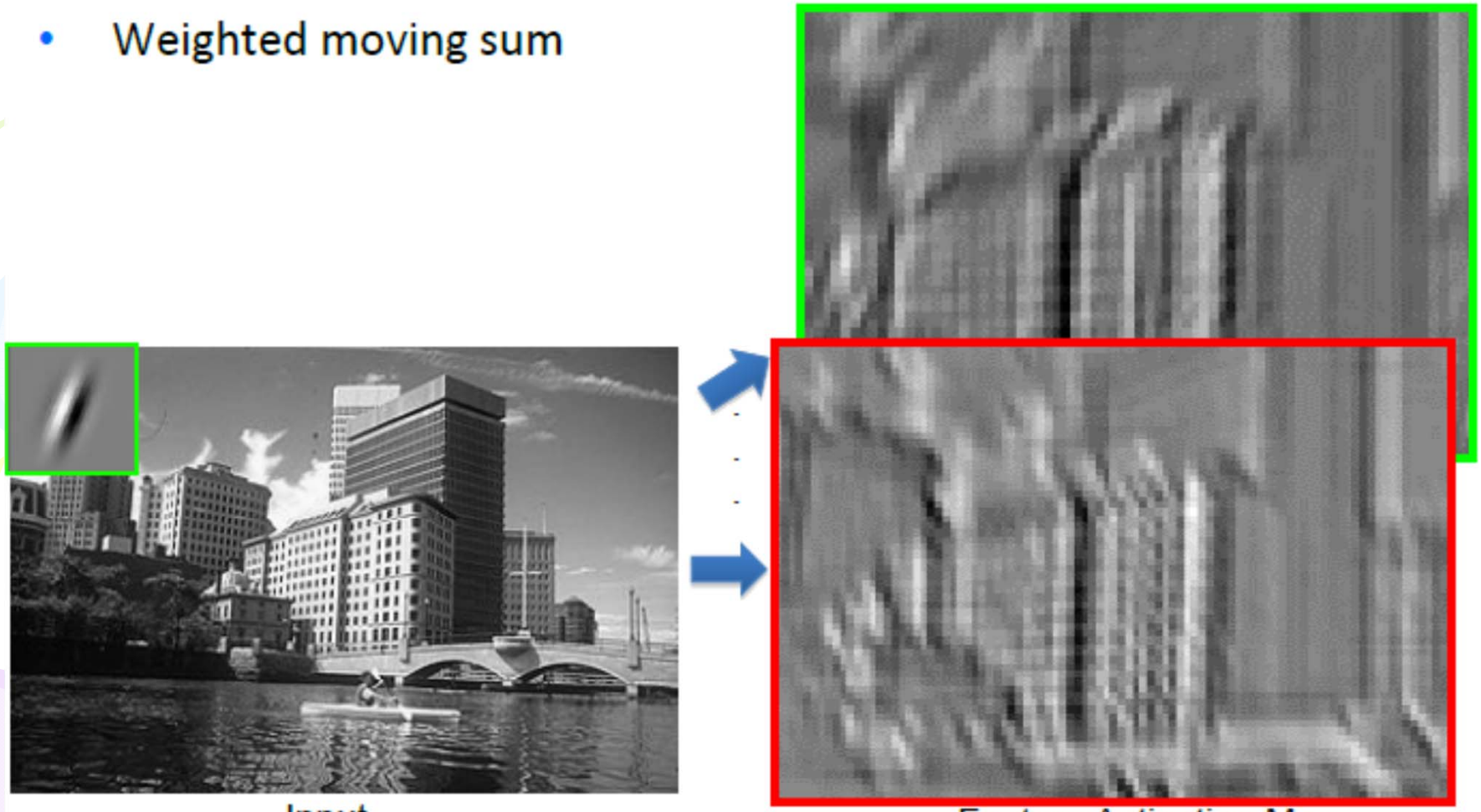


Image credit: A. Karpathy

slide: J.-B. Huang

What is a Convolution?

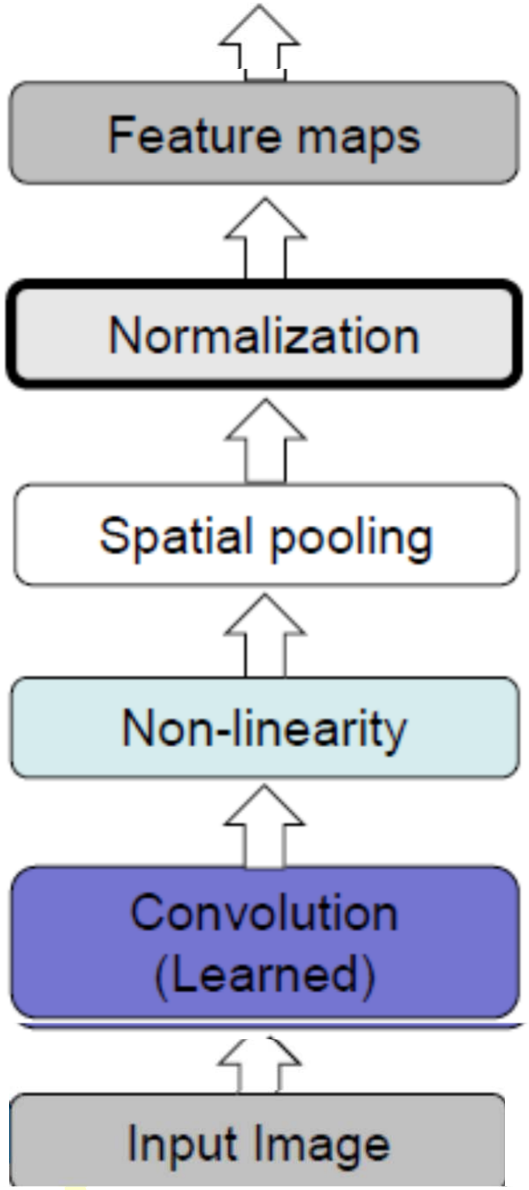
- Weighted moving sum

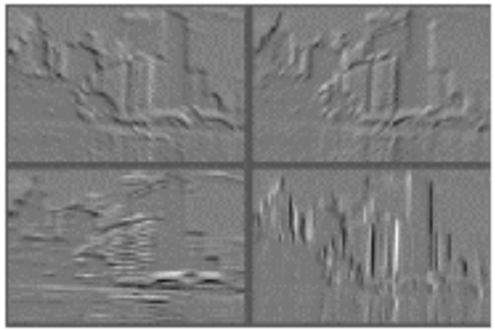


Input

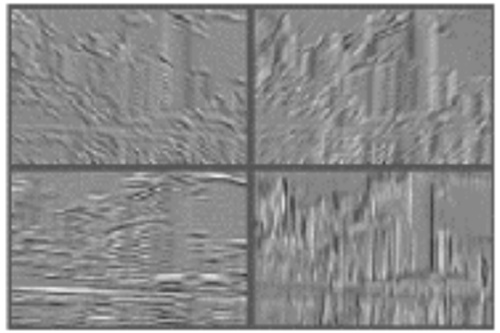
Feature Activation Map

Convolutional Neural Networks





Feature Maps

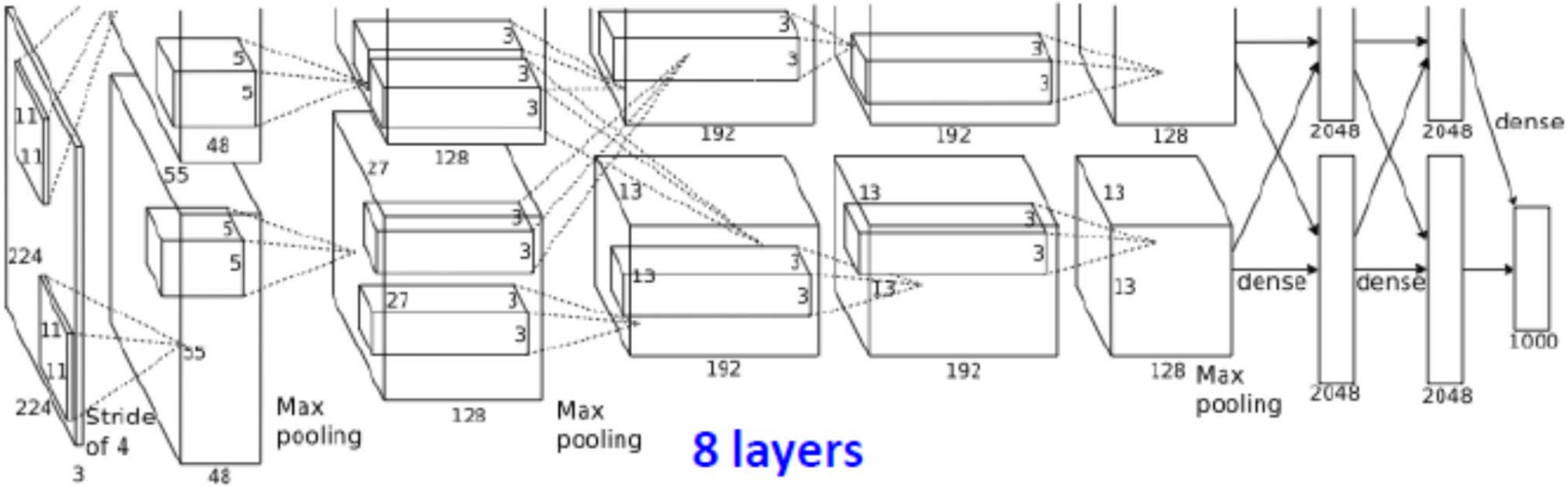


Feature Maps
After Contrast
Normalization

Input

Feature Map

Modern CNN: AlexNet



Input: 224*224*3=**150K**

Neurons: 290400+186624+64896+64896+43264+4096+4096+1000=**650K**

Weights: 11*11*3*48*2(35K)+5*5*48*128*2(307K)+128*3*3*192*4(884K)+192*3*3*192*2(663K)+192*3*3*128*2(442K)+6*6*128*2048*4(38M)+4096*4096(17M)+4096*1000(4M)=**60M**

- **More data (1.2M)**
- **Trained on two GPUs for a week**
- **Dropout**

ImageNet ISLVRC 2012-2014: Object Recognition

Best non-convnet in 2012: 26.2%

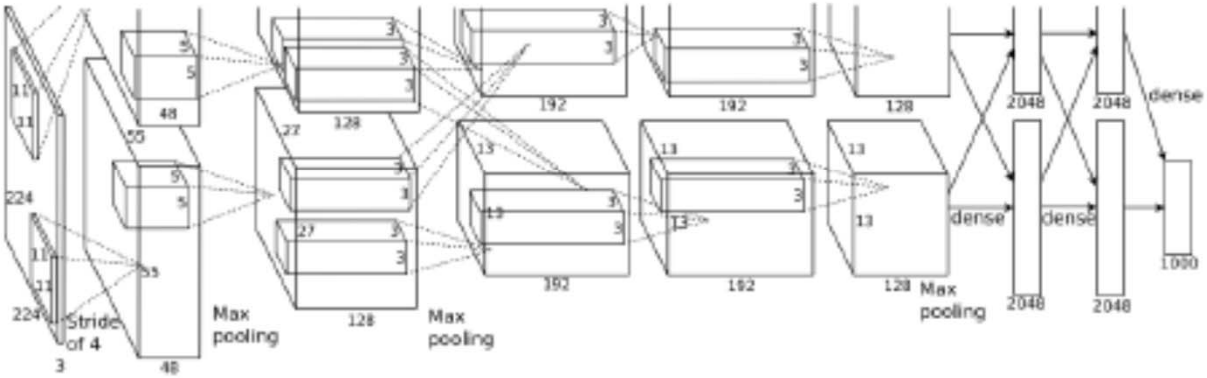
Team	Year	Place	Error (top-5)	External data
SuperVision – Toronto (7 layers)	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai – NYU (7 layers)	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
VGG – Oxford (16 layers)	2014	2nd	7.32%	no
GoogLeNet (19 layers)	2014	1st	6.67%	no
Human expert*			5.1%	

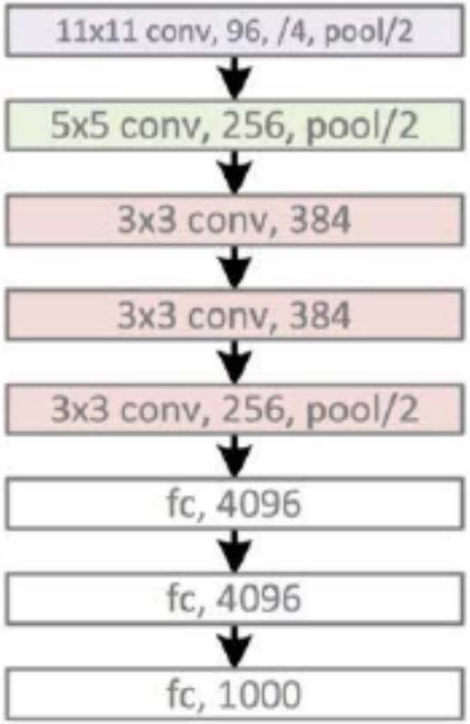
Team	Method	Error (top-5)
DeepImage - Baidu	Data augmentation + multi GPU	5.33%
PReLU-nets - MSRA	Parametric ReLU + smart initialization	4.94%
BN-Inception ensemble - Google	Reducing internal covariate shift	4.82%

AlexNet

[Krizhevsky et al., NIPS'12]

- Architecture overview
  - Five convolutional layers
  - Three fully connected layers



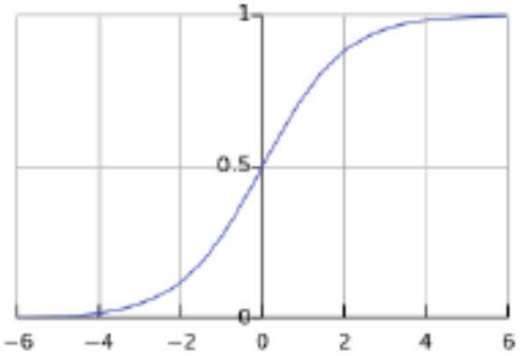


AlexNet

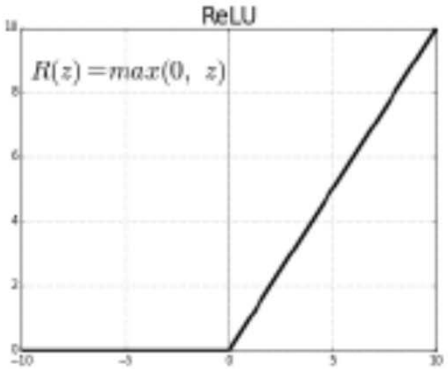
- Five major contributions:
  - Nonlinearity: ReLU
  - Multiple GPUs
  - Local response normalization (LRN)
  - Overlapping pooling
  - Reducing overfitting: data augmentation and dropout

AlexNet: ReLU and multiple GPUs

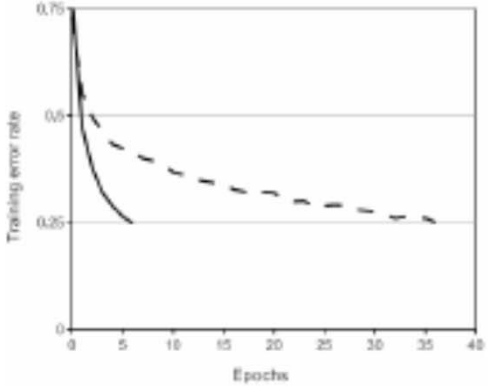
- Nonlinearity: Rectified Linear Units (ReLU)
  - Instead of tanh or sigmoid
  - Faster convergence and lower gradient vanishing



sigmoid function



ReLU



ReLU (solid line)
tanh (dashed line)

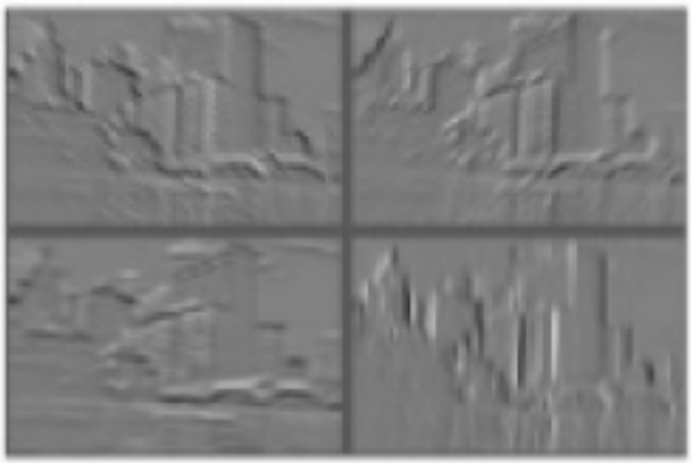
- Use multiple GPUs to speed up training

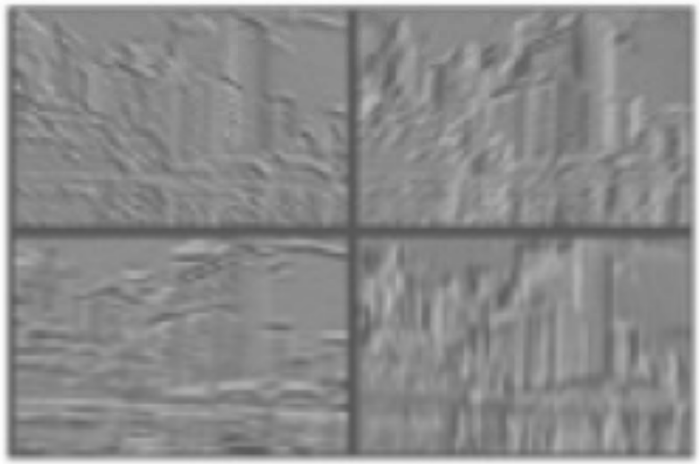
AlexNet: Local response normalization

- Local response normalization (LRN)

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

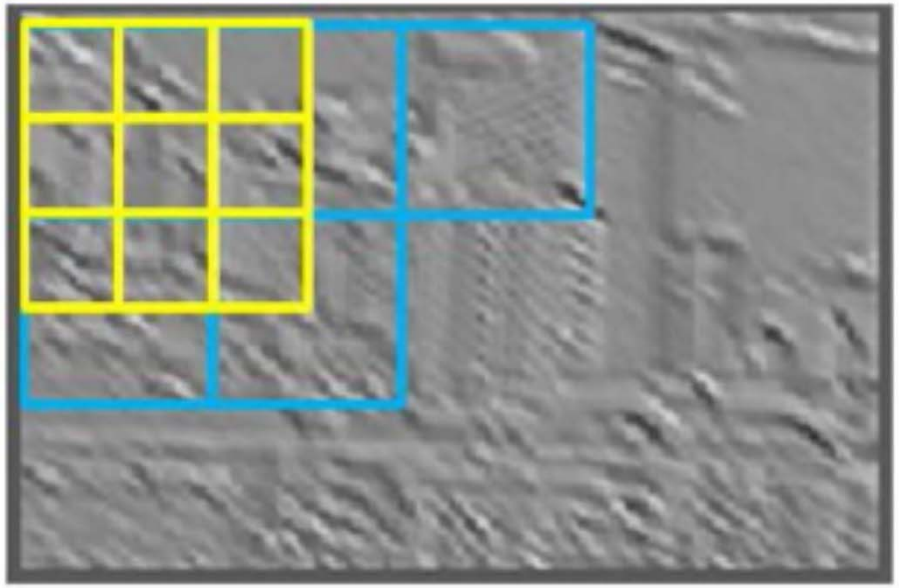
- Reduce the top-1 and top-5 error rates by 1.4% and 1.2%





AlexNet: Overlapping pooling

- Overlapping pooling
  - Slightly alleviate the risk of overfitting
  - Reduce the top-1 and top-5 error rates by 0.4% and 0.3%

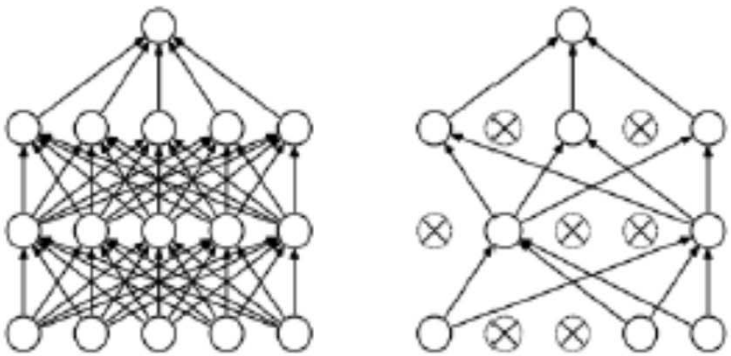


non-overlapping pooling

overlapping pooling

AlexNet: Data augmentation and dropout

- Reduce overfitting: data augmentation and dropout
- Data augmentation
  - Image translations and horizontal reflections
  - Altering the intensities of the RGB channels
- Dropout
  - Apply to fully connect layers
  - Setting the output of each hidden neuron to zero with probability 0.5 during the training stage



(a) Standard Neural Net (b) After applying dropout.

[Srivastava et al., JMLR'15]

AlexNet: Experimental results

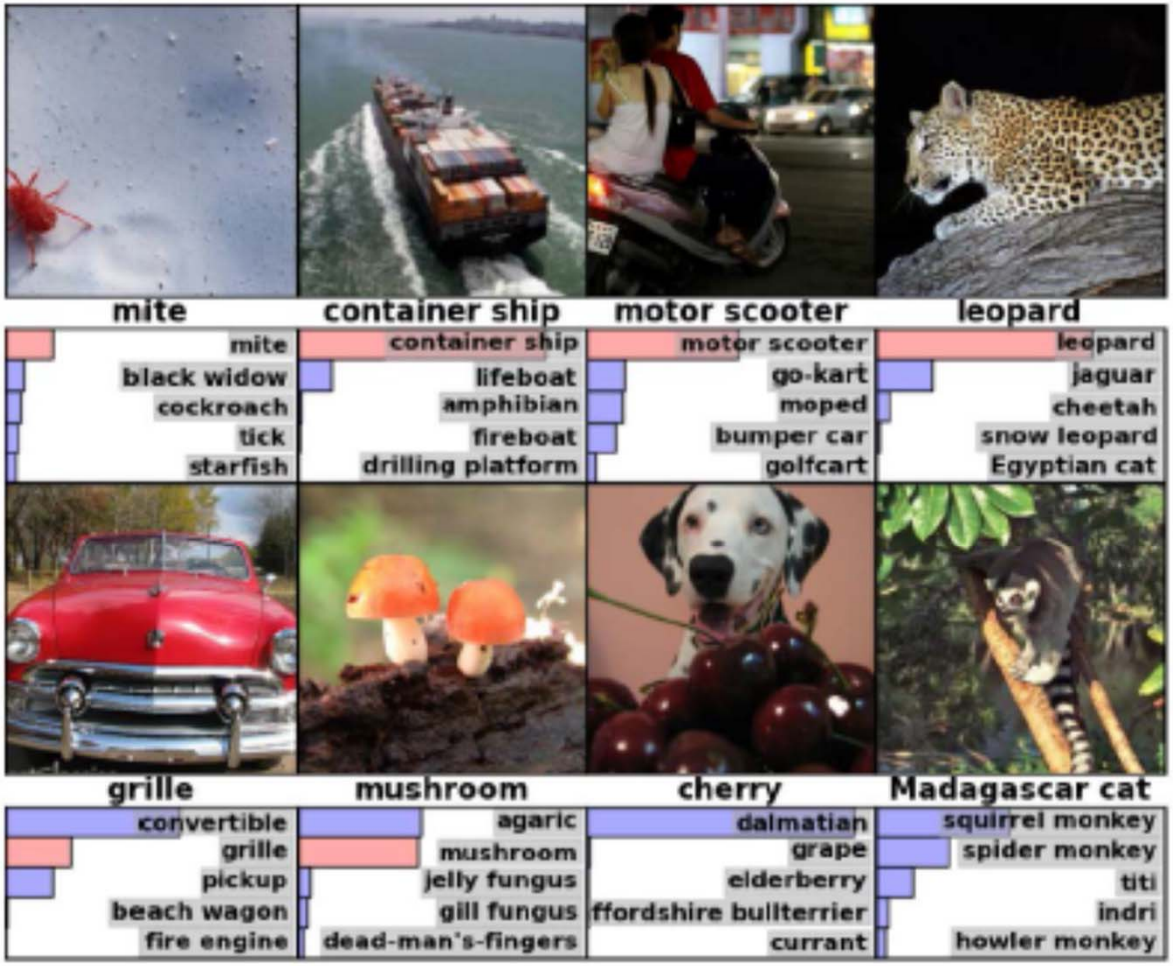
- ILSVRC-2012 competition
- Top-1 and top-5 errors

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	26.2%
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

- X CNNs: Average the predictions of X similar CNNs
- * denotes that models pre-trained to classify ImageNet 2011

AlexNet: Experimental results

- Eight examples of top-5 prediction



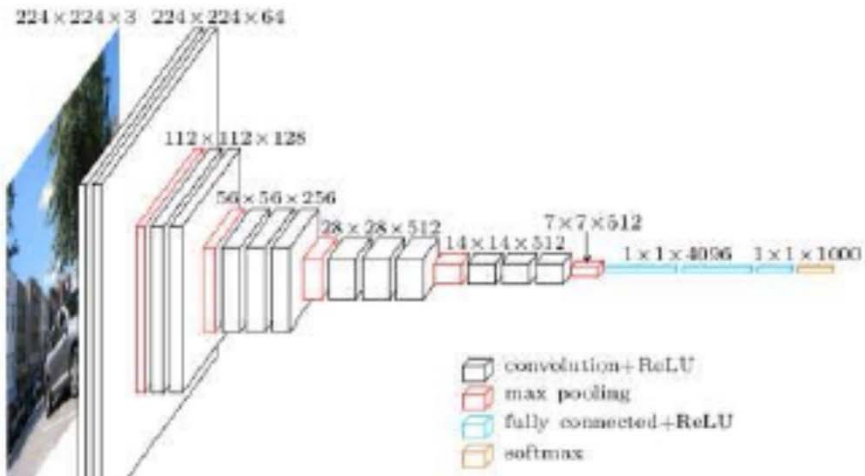
AlexNet: Experimental results

- First column: Testing image
- The remaining columns: Its six nearest neighbors

VGG-Net

[Simonyan and Zisserman, ICLR'15]

- Architecture overview
  - Effect of CNN depths on accuracy
  - 16 layers or 19 layers
  - Deeper than AlexNet



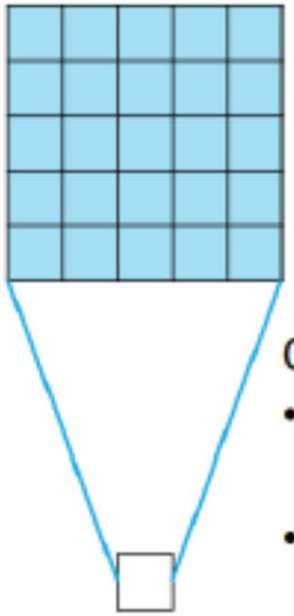
https://goo.gl/Kgyei5

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

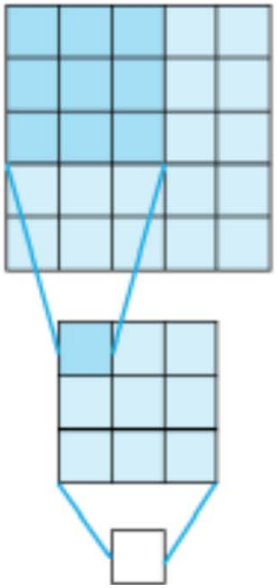
VGG-Net

- 3x3 convolutional kernels – less parameters
  - Stacked convolutional layers have large receptive fields
  - More non-linearity
  - Less parameters to learn
  - More numbers of channels

https://goo.gl/vAs3TZ



One 5x5 filter
- Parameters: 5x5 = 25
- Non-linear:1



Two 3x3 filters
- Parameters: 3x3x2 = 18
- Non-linear:2

Performance analysis

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

- A vs. A-LRN: LRN may not be effective
- A vs. B, C, D, E: The deeper, the better
- A vs. C: 1 x 1 convolutional layers work
- C vs. D: Spatial context is important
- Multiple scale training improves the performance

VGG-Net

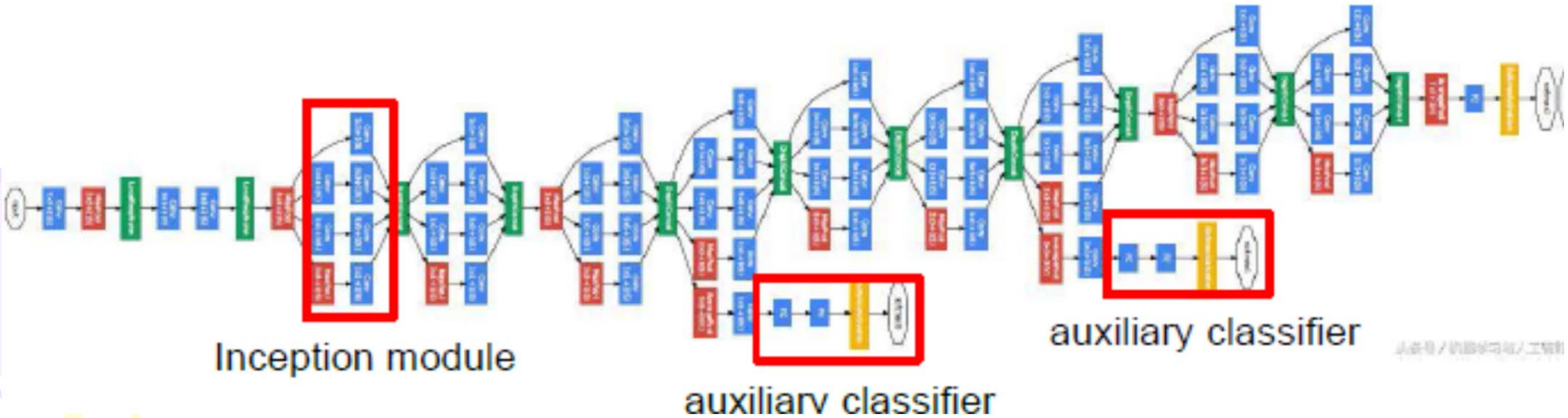
- ILSVRC-2012 competition

Team	Year	Place	Error (top-5)	Uses external data	
SuperVision	2012	1st	16.4%	no	AlexNet
SuperVision	2012	1st	15.3%	Imagenet 22k	
Clarifai	2013	1st	11.7%	no	
Clarifai	2013	1st	11.2%	Imagenet 22k	
MSRA	2014	3rd	7.35%	no	
VGG	2014	2nd	7.32%	no	VGG-Net
GoogLeNet	2014	1st	6.67%	no	

GoogleNet (Inception V1)

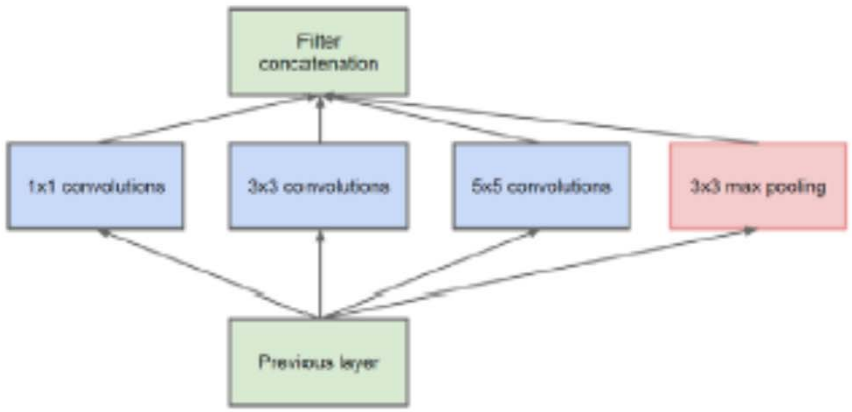
[Szegedy et al., CVPR'15]

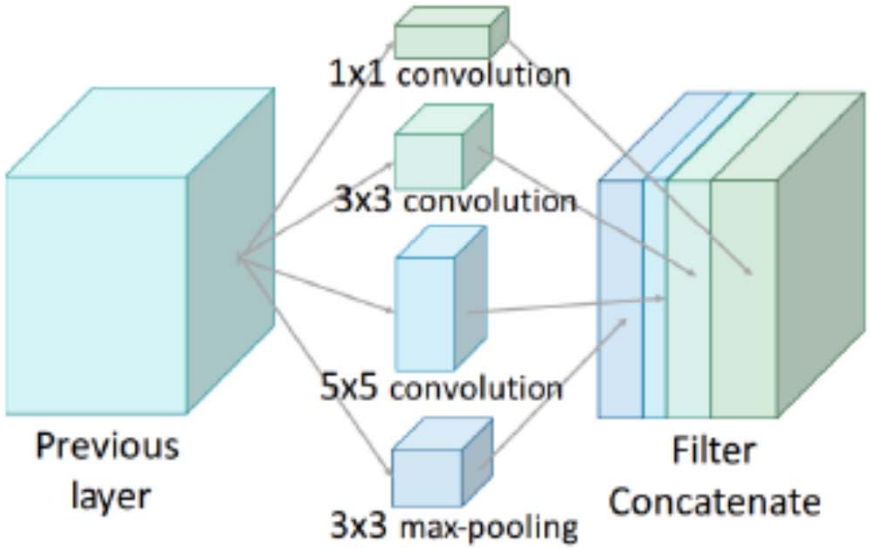
- Architecture overview
  - 22 layers
  - 12 times lesser parameters than AlexNet
  - Significantly more accurate than AlexNet
  - Inception module
  - Auxiliary classifiers



GoogleNet: Inception module

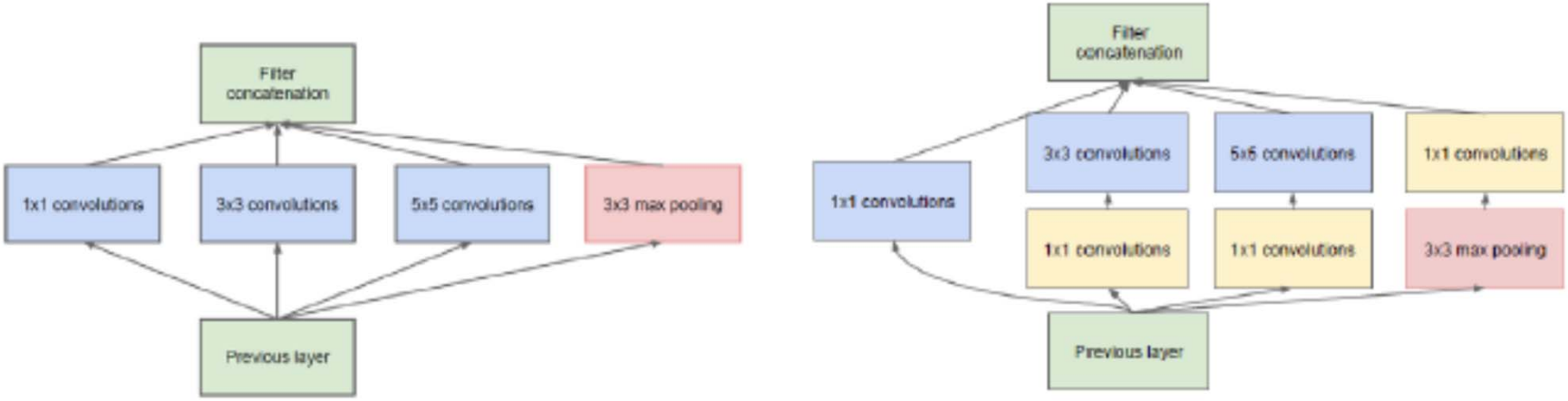
- Choose filter sizes of 1x1, 3x3, and 5x5
- Concatenate all feature maps
- Concatenate one additional pooling path, which is essential to the success of CNNs





GoogleNet: Inception module

- Linear decrease in feature maps results in quadratic decrease in computation
- Use inexpensive 1x1 convolutional layers to reduce the number feature maps



GoogleNet: Inception module stacking

- Stacking inception modules
- Conventional convolutional layers in lower layers
- Max pooling for size reduction

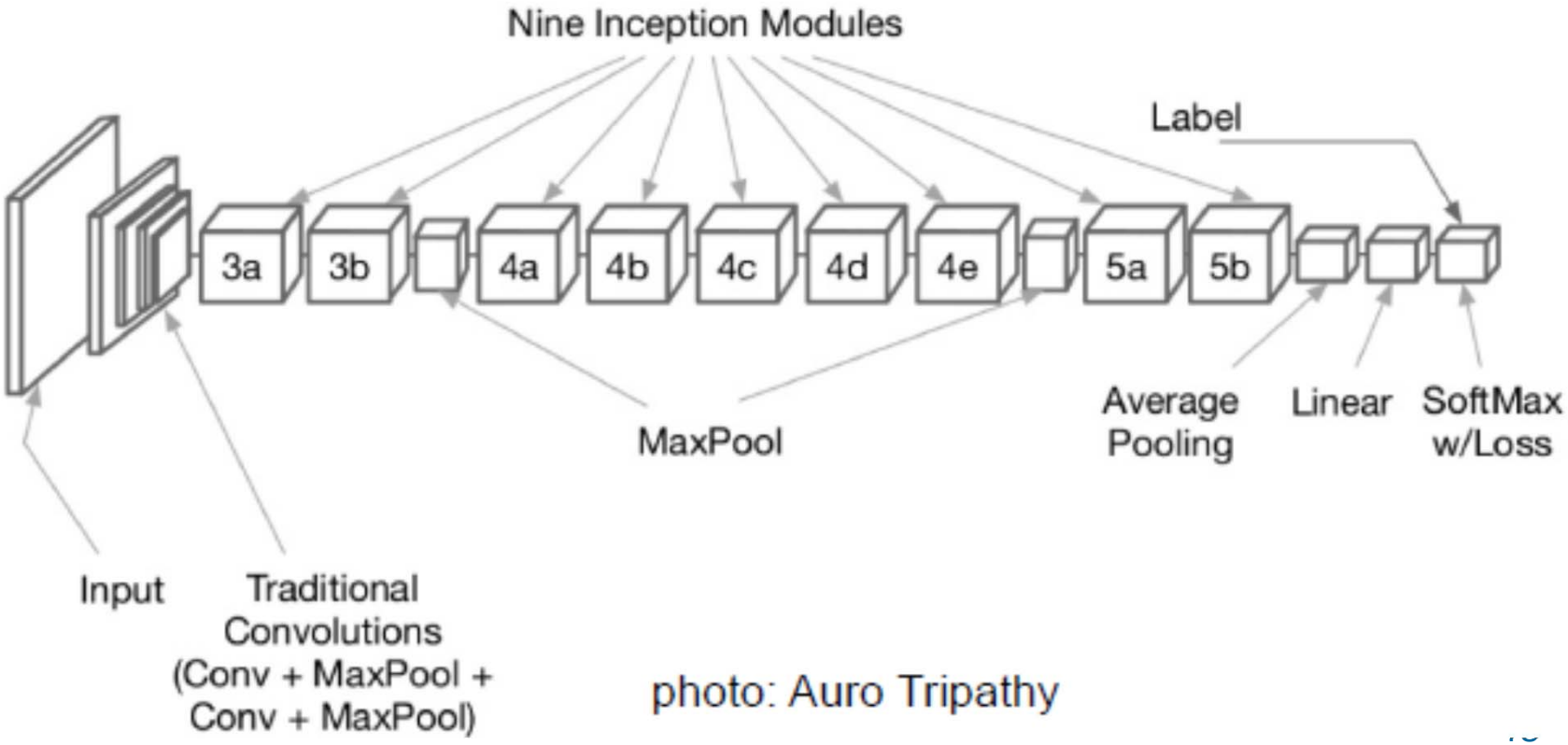


photo: Auro Tripathy

GoogleNet: Global average pooling

- Fully connected layers are prone to overfitting
- Average pooling has no parameters -> no overfitting
- Use average pooling instead of fully connected layers

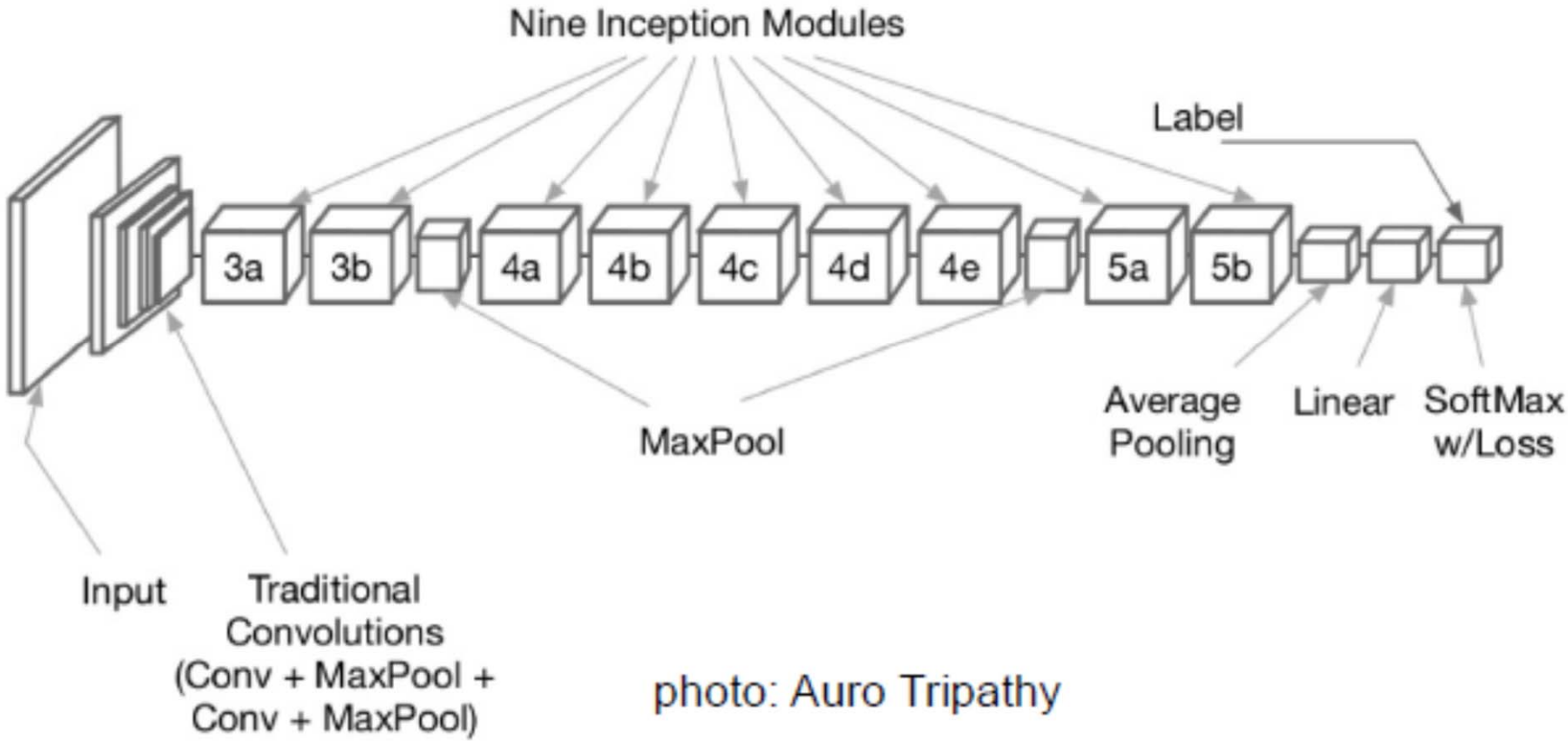


photo: Auro Tripathy

GoogleNet: Auxiliary classifier

- Deep networks result in vanishing gradients
- Auxiliary classifiers are appended

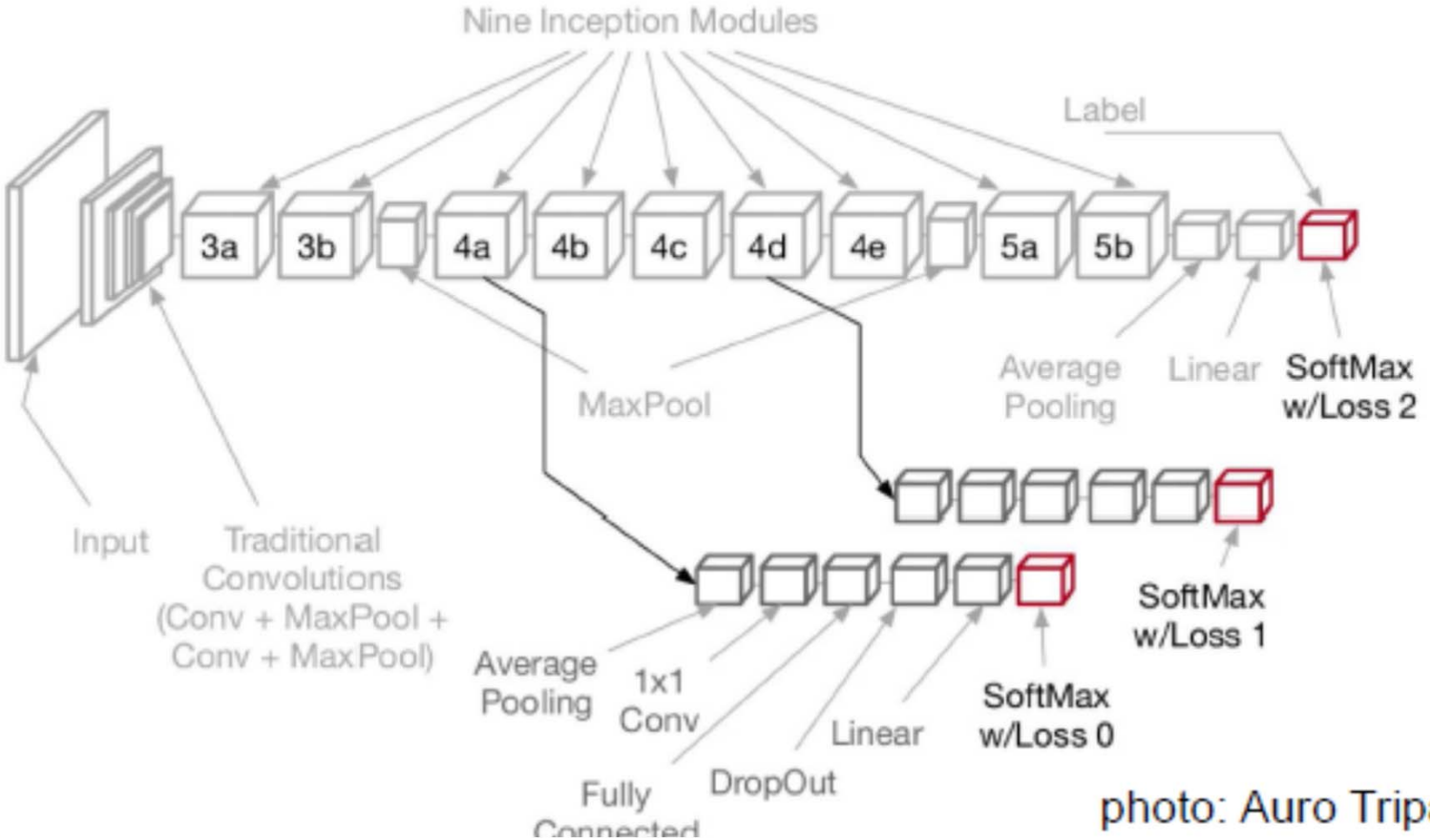


photo: Auro Tripathy

GoogleNet: Auxiliary classifier

- Intermediate layers become discriminative
- Intermediate losses alleviate the problem of vanishing graident

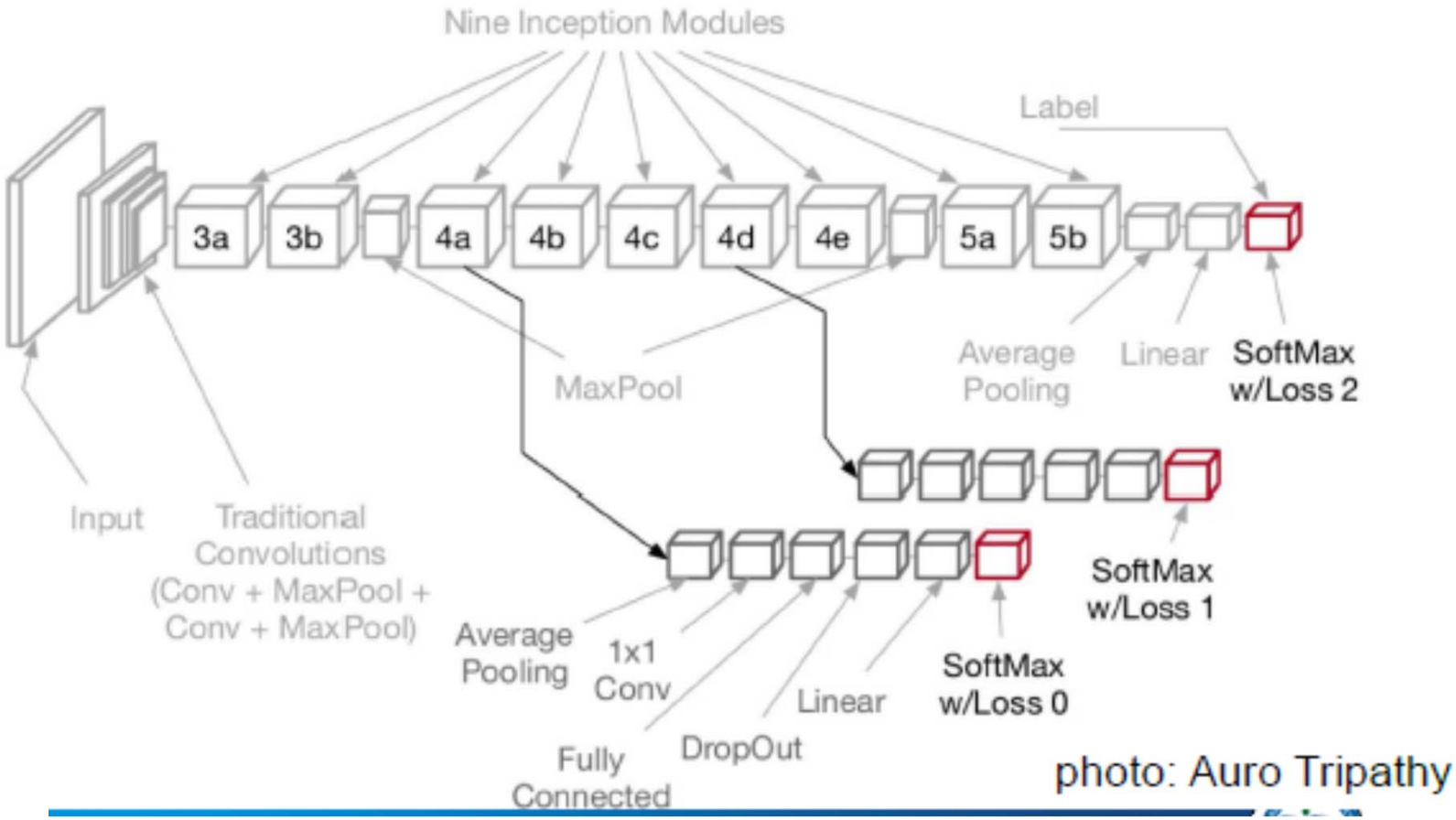


photo: Auro Tripathy

GoogleNet (Inception V1)

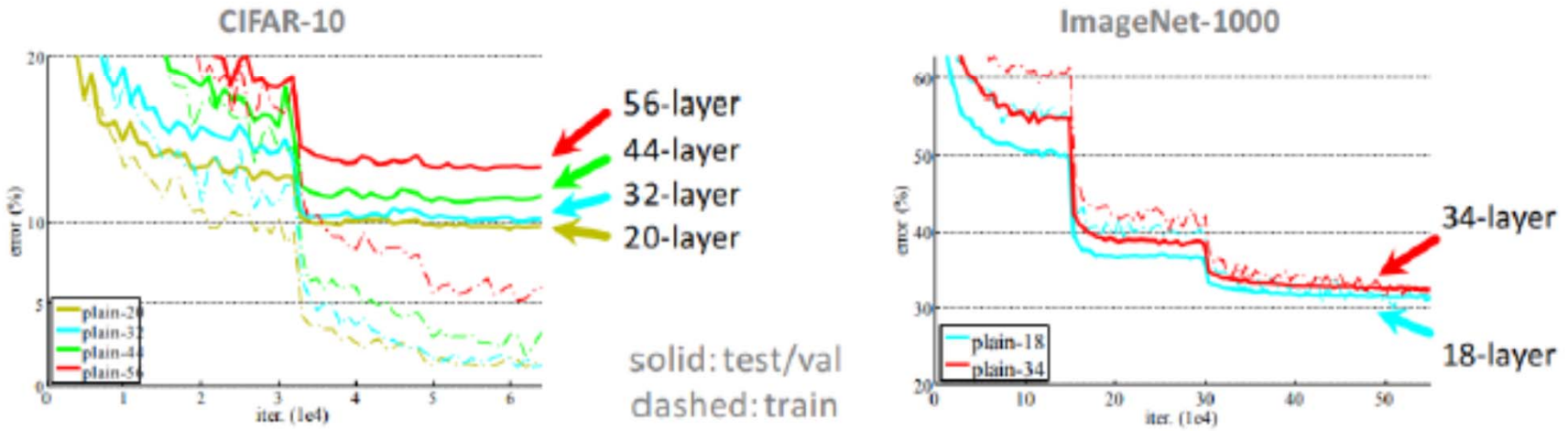
- ILSVRC-2012 competition

Team	Year	Place	Error (top-5)	Uses external data	
SuperVision	2012	1st	16.4%	no	AlexNet
SuperVision	2012	1st	15.3%	Imagenet 22k	
Clarifai	2013	1st	11.7%	no	
Clarifai	2013	1st	11.2%	Imagenet 22k	
MSRA	2014	3rd	7.35%	no	
VGG	2014	2nd	7.32%	no	VGG-Net
GoogLeNet	2014	1st	6.67%	no	GoogleNet

ResNet

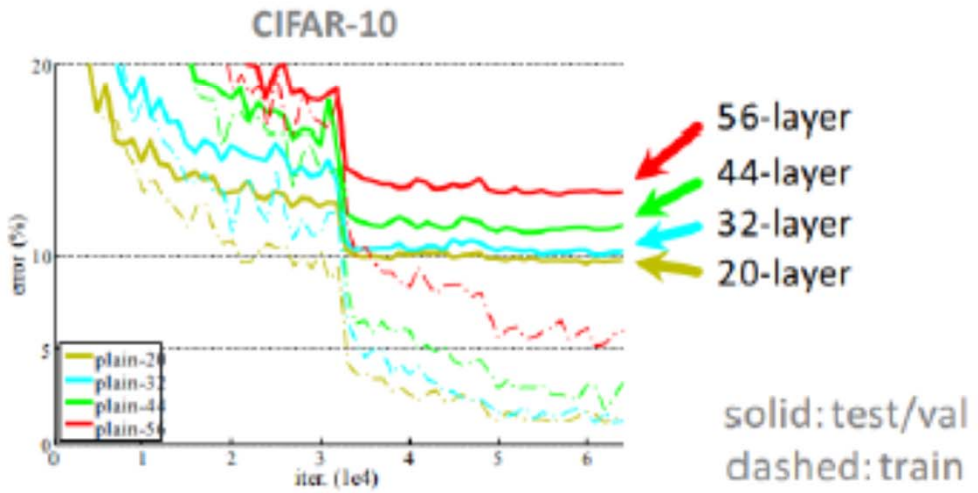
[He et al., CVPR'16]
Best Paper Award!

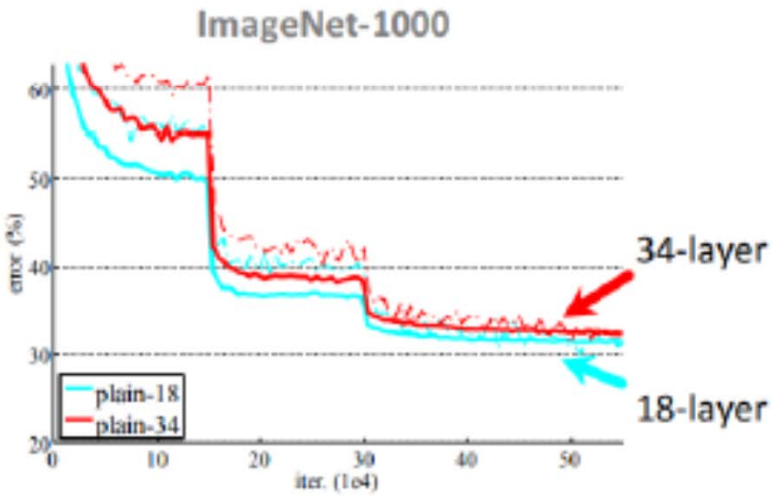
- The deeper, the better
  - Large receptive field size
  - High non-linearity
  - Better fitting power
- Really?
  - Performance drops when using an overly deep CNN model



ResNet

- Overfitting?
  - No. Not only testing but also training errors increase!
- This is a general phenomenon, observed in many datasets
- It is caused by gradient exploding/vanishing



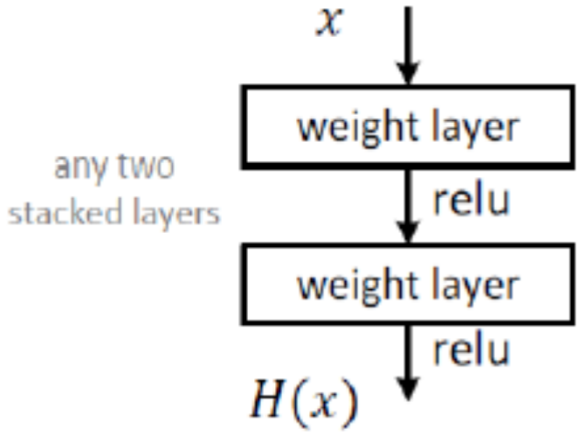


ResNet

- Architecture overview
  - 50, 101, 152 layers
  - Very deep, even more than 1,000 layers
  - It is constructed by stacking residual modules
  - Superior performances

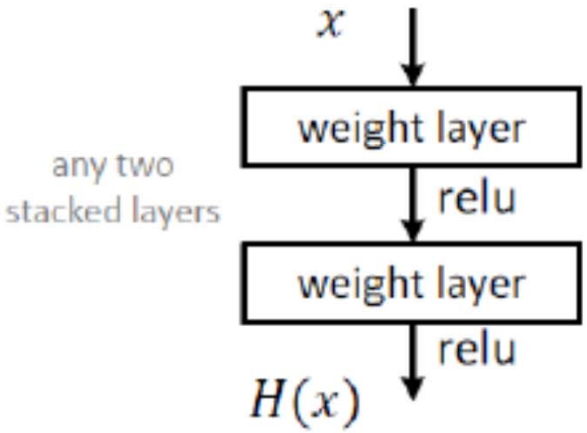
ResNet: Residual module

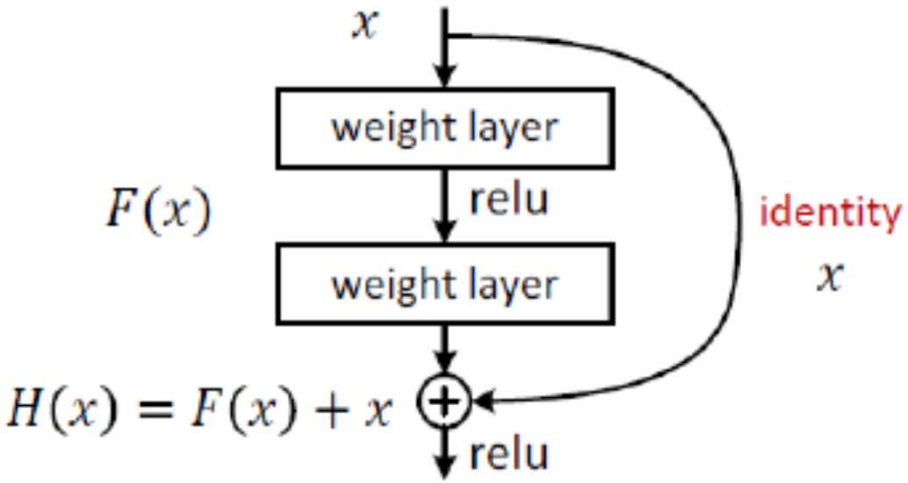
- The amount of changes (output - input) is fixed
- Adding more layers makes changes between two adjacent layers smaller
- Optimal mappings are close to identity
- Difficult to approximate an identity mapping due to ReLU



ResNet: Residual module

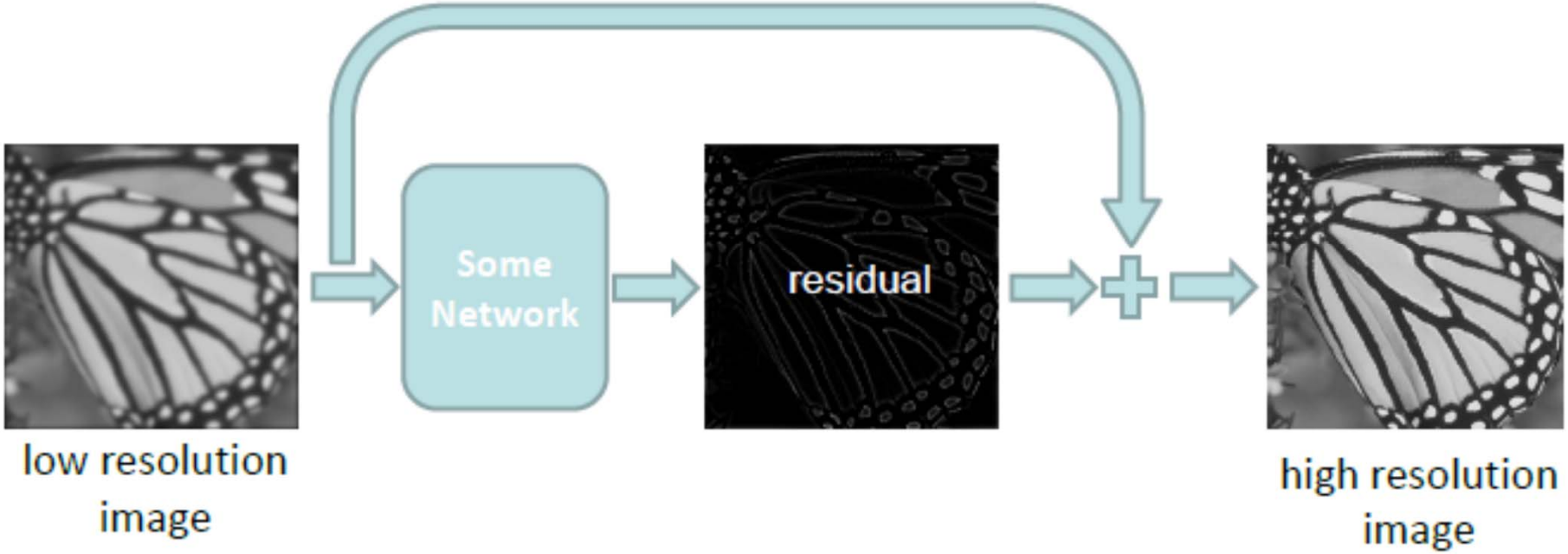
- Learn the residual F(x), instead of the desired output H(x)
- Residual: Difference between the input and the desired output
- Shortcuts connections





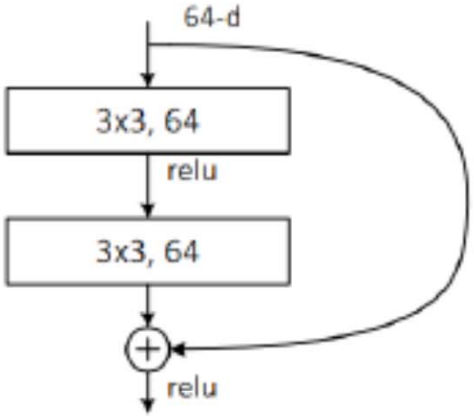
ResNet: Residual module

- Learn the residual F(x), instead of the desired output H(x)
- Residual: Difference between the input and the desired output
- Shortcuts connections

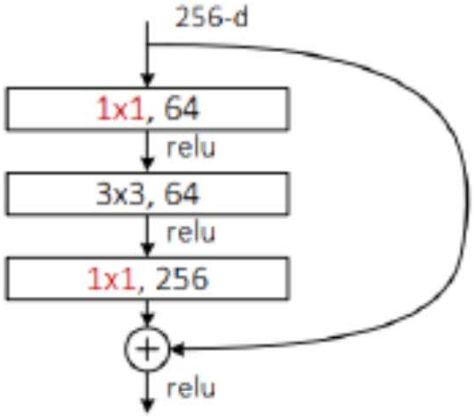


ResNet: Residual module

- A practical way to go deeper
- Inexpensive 1x1 convolutional layers for channel reduction



all-3x3



bottleneck
(for ResNet-50/101/152)

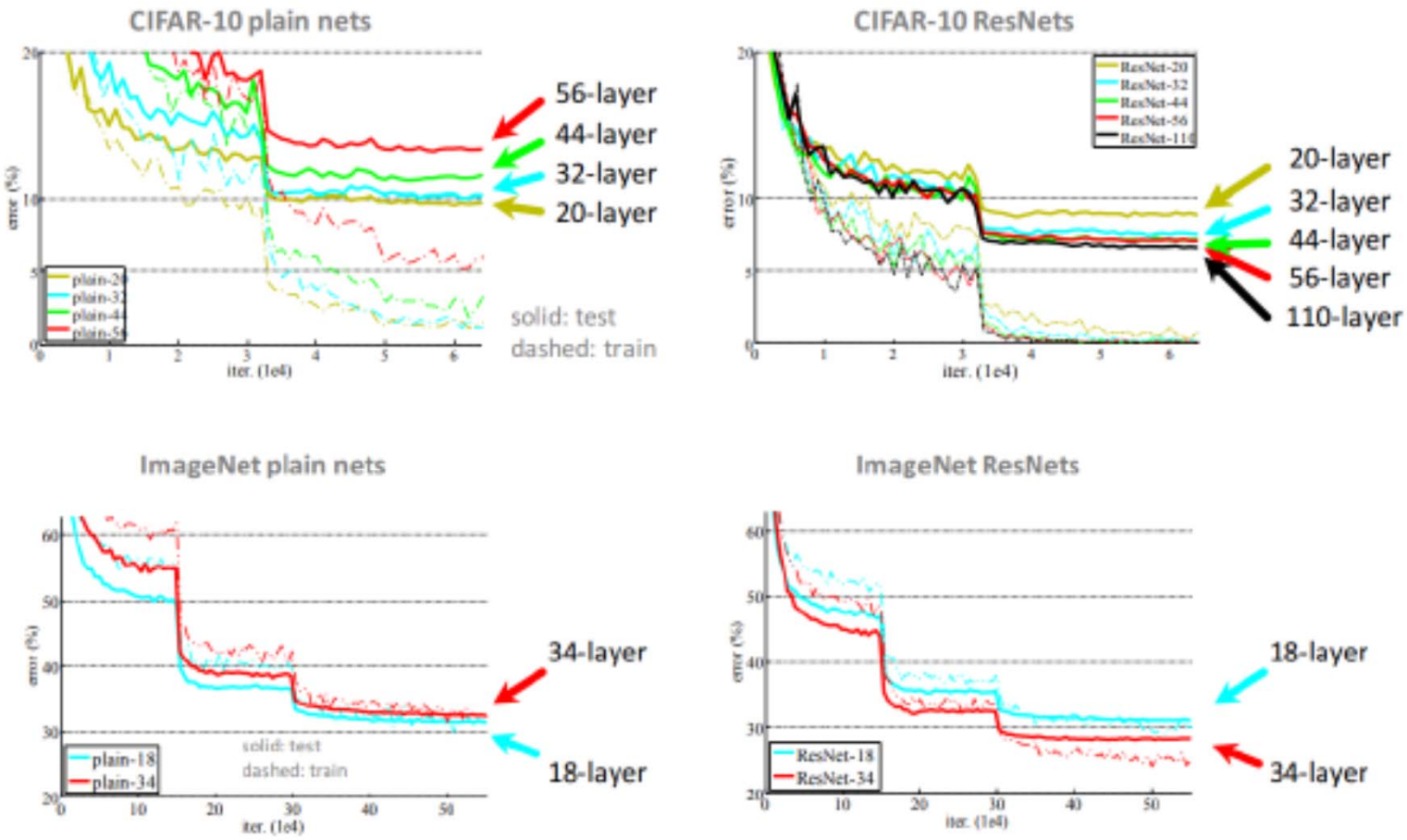
photo: H. Son

ResNet: Residual module

- Less parameters in a single layer
- Accelerate the training of deep networks
- Reduce the risk of vanishing gradient
- Increase depth of the network
- Achieve higher accuracy in many vision applications

Experimental results: Convergence

- Convergence



Experimental results: Comparison with SOTA

method	top-5 err. (**test**)
VGG [41] (ILSVRC'14)	7.32
GoogLeNet [44] (ILSVRC'14)	6.66
VGG [41] (v5)	6.8
PReLU-net [13]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

Experimental results: Depth vs. Error

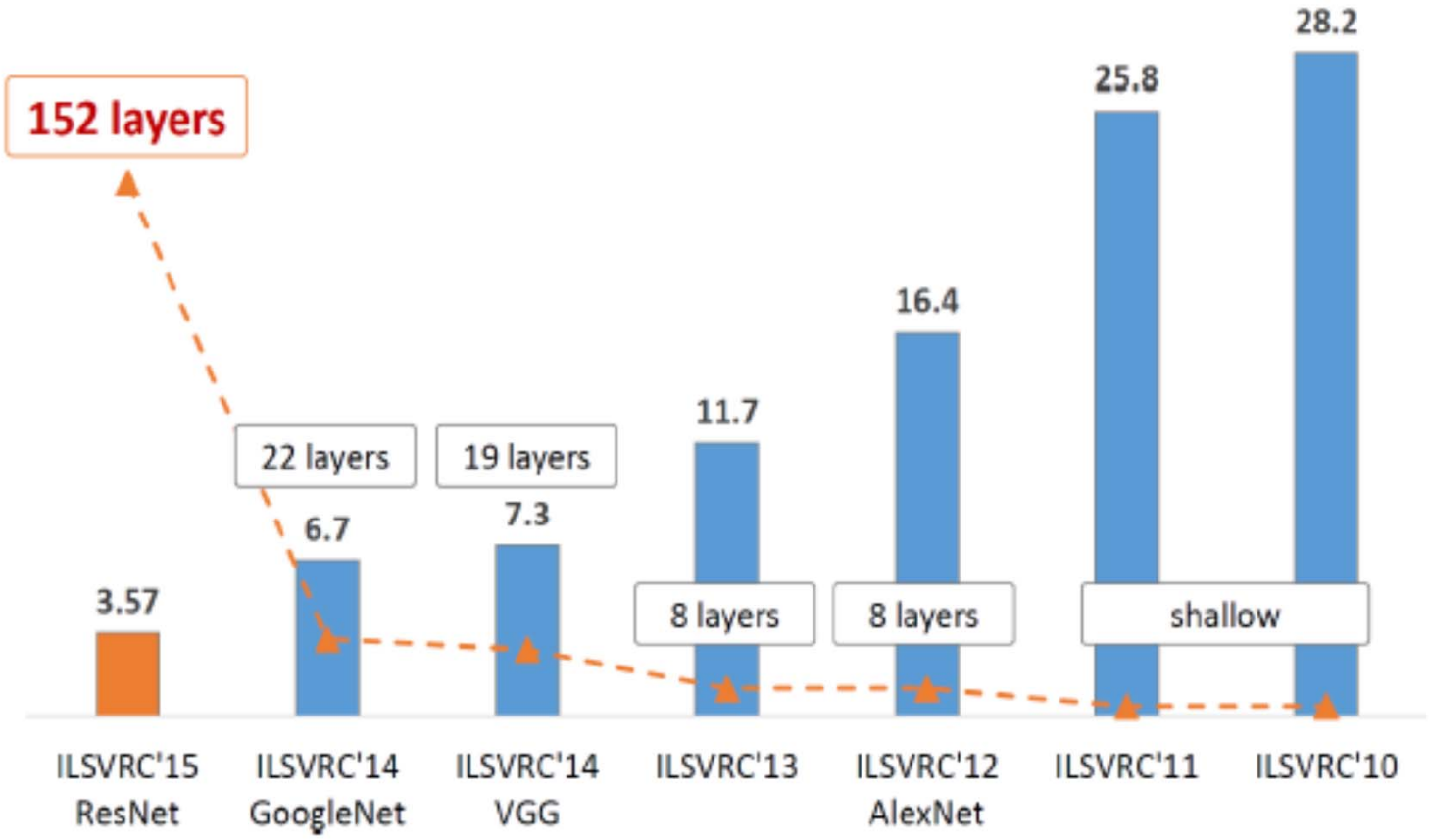


photo: He et al.

Revolution of depth

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

photo: He et al.

Experimental results on more datasets and applications

- Applications: localization, detection, segmentation, …
- Datasets: ImageNet, MS COCO, …

task	2nd-place winner	MSRA	margin (relative)
ImageNet Localization (top-5 error)	12.0	9.0	**27%**
ImageNet Detection (mAP@.5)	53.6	62.1	**16%**
COCO Detection (mAP@.5:.95)	33.5	37.3	**11%**
COCO Segmentation (mAP@.5:.95)	25.1	28.2	**12%**

absolute 8.5% better!

photo: He et al.

Experimental results: Object detection

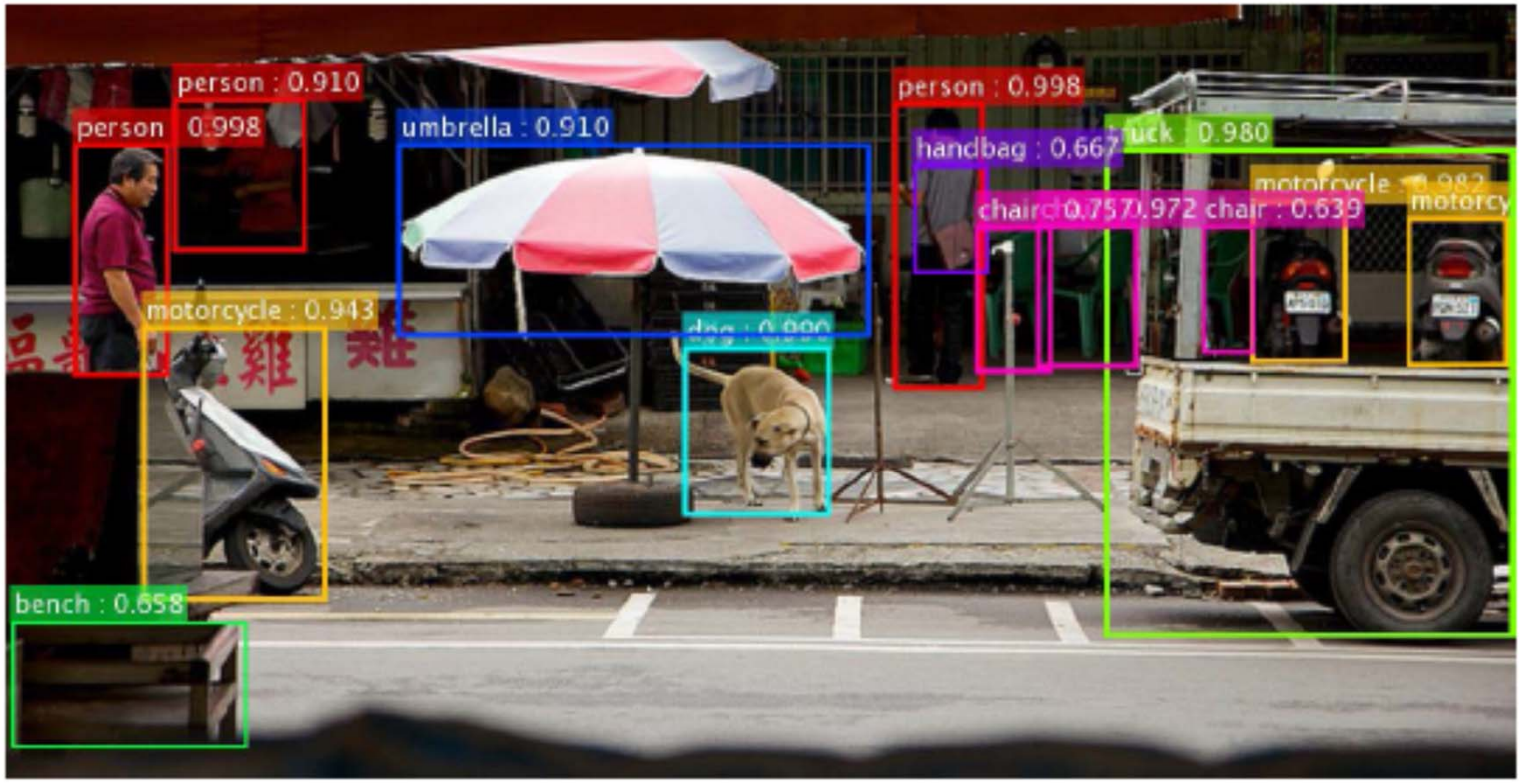


photo: He et al.

Experimental results: Instance segmentation

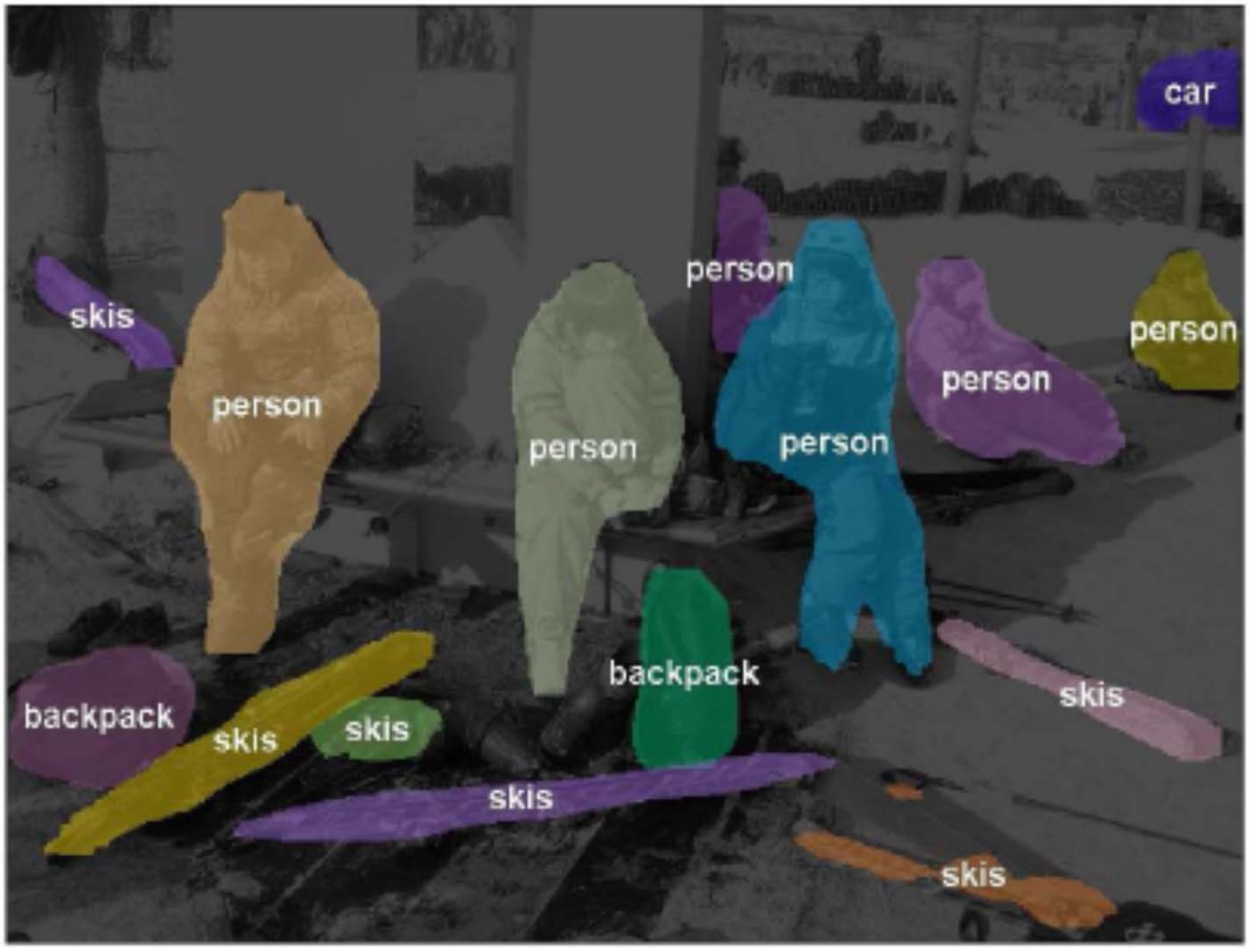
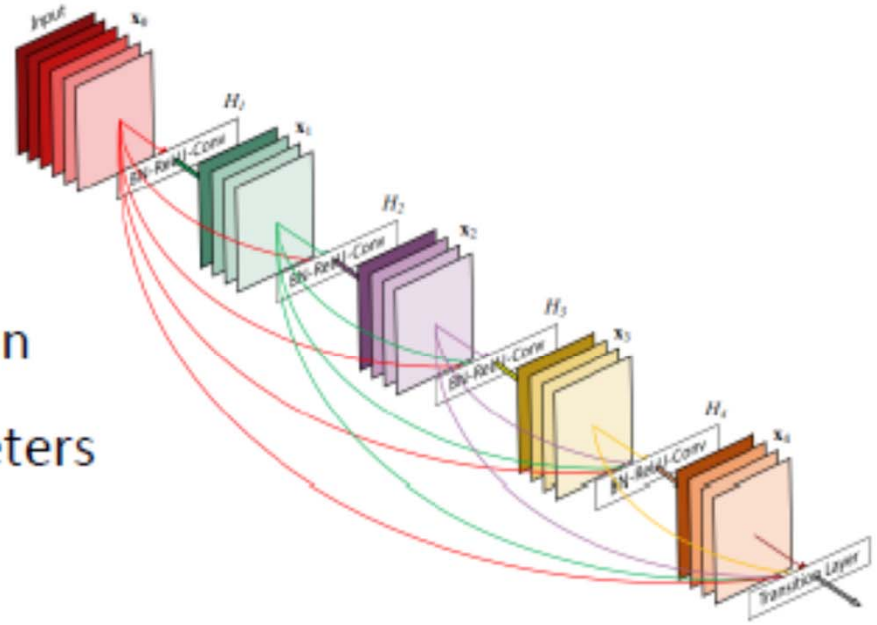


photo: He et al.

DenseNet

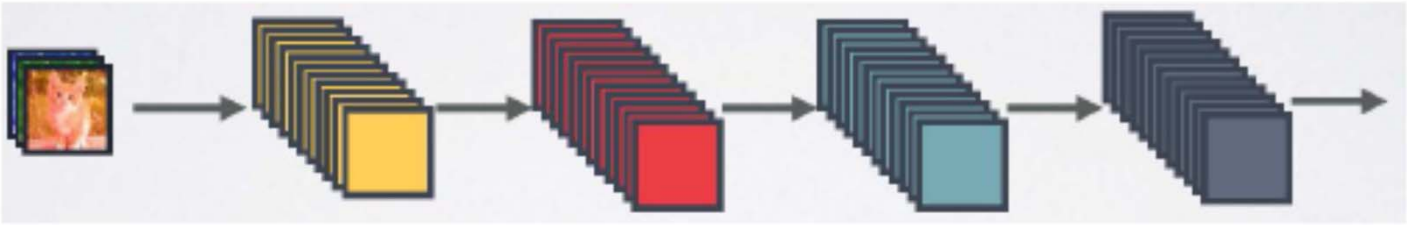
[Huang et al., CVPR'17]
Best Paper Award!

- Network architecture
- Densely connect each layer to every other layer
- For a layer, feature maps of all preceding layers are its input
- Advantages:
  - Alleviate vanishing gradient
  - Encourage feature reuse
  - Strengthen feature propagation
  - Reduce the number of parameters

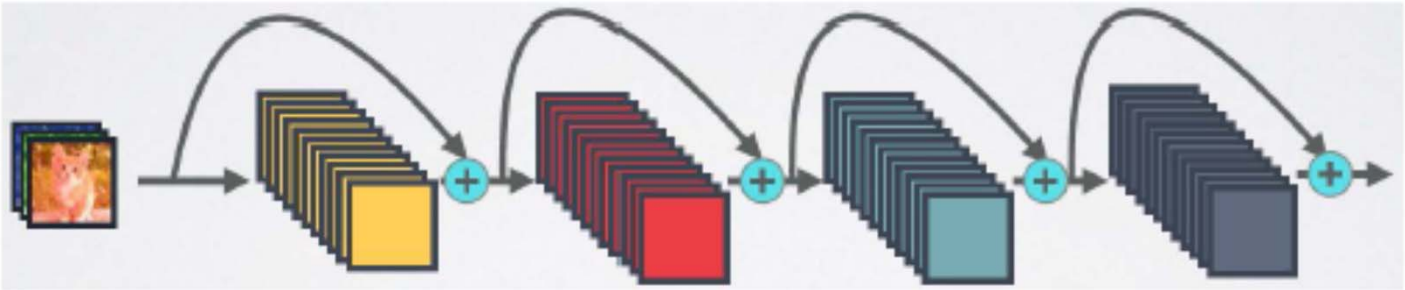


CNNs vs. ResNet

- Conventional CNNs
  - A convolutional layer takes feature maps from its preceding layer

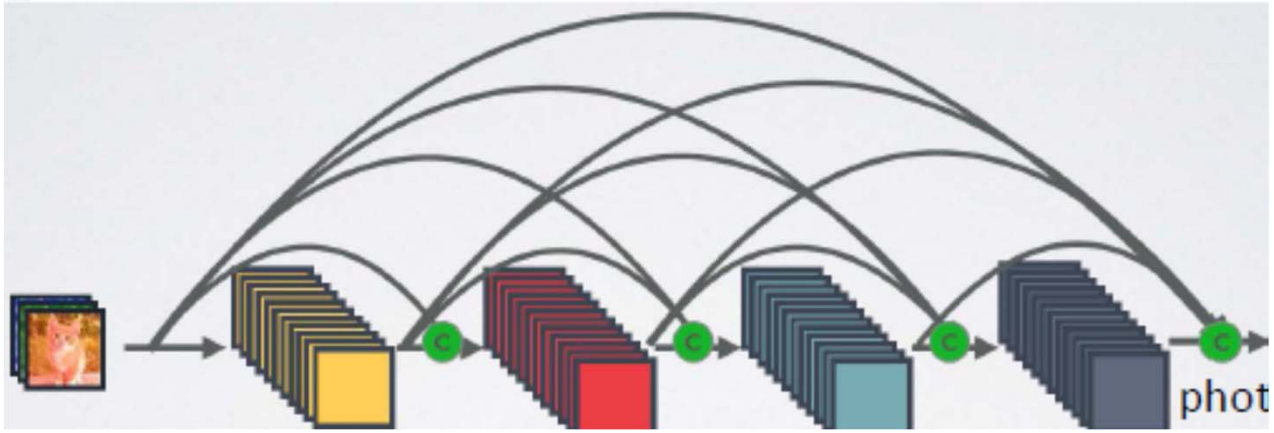


- ResNet
  - One additional path (identity mapping) with element-wise addition



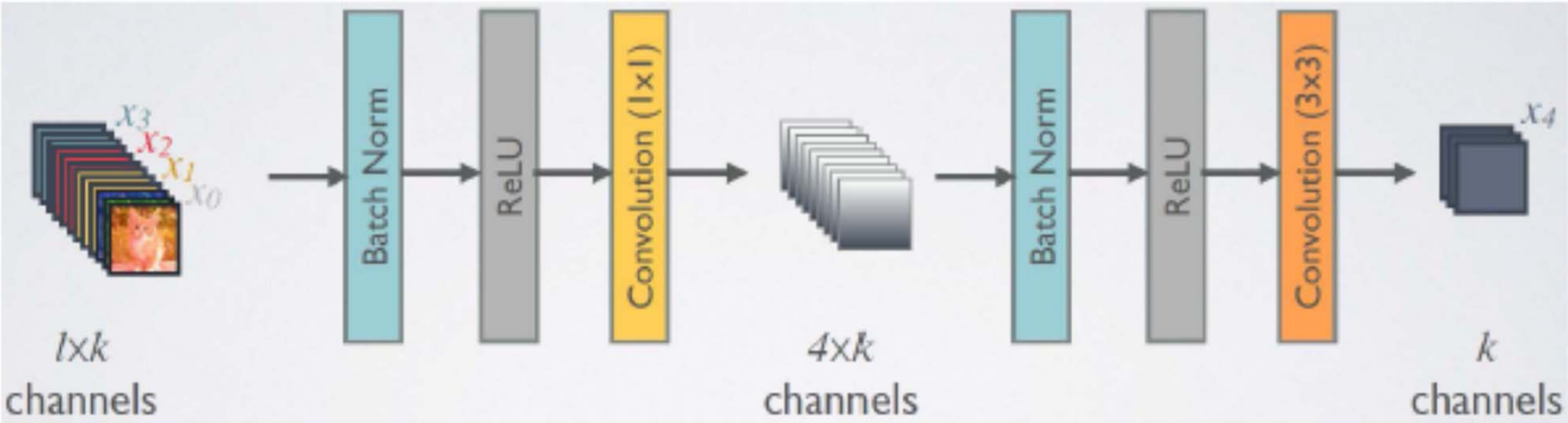
DenseNet

- Dense connectivity: a link between every layer pair
  - Each layer takes all preceding feature maps as input by channel-wise concatenation
  - Reduce the risk of gradient vanishing
  - Feature reuse
  - Fewer filters per layer
  - High computation cost and more parameters per filter?



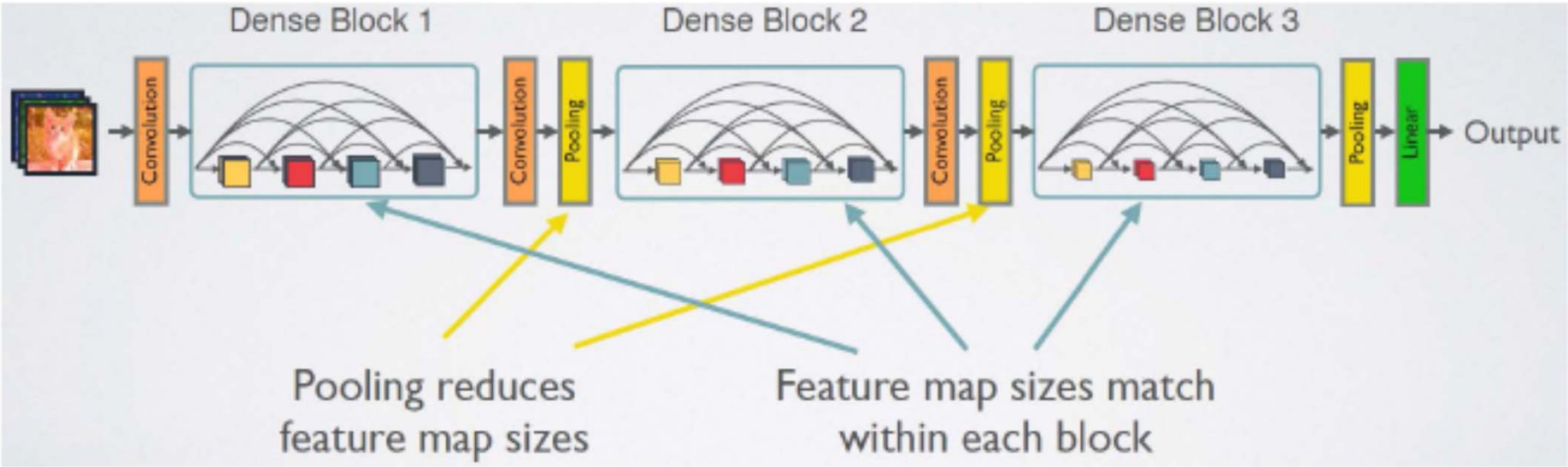
DenseNet

- 1x1 convolutional layer for channel (feature map) reduction
- Suppose each layer produces *k* feature maps
- There will be *l*x*k* channels at the *l*th convolutional layer
- Reduce to *4*x*k* channels before producing its *k* feature maps
- No pooling for map size reduction?



DenseNet

- Dense block: a few densely connected convolutional layers
- Stack dense blocks
- Insert a convolutional layer and a pooling layer into two connected blocks



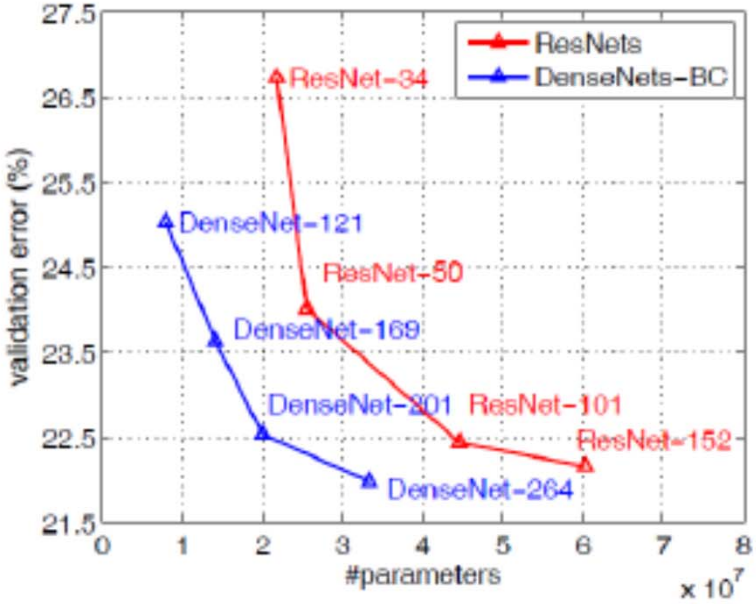
Experimental results: Comparison with SOTA

- ILSVRC, top-5 error

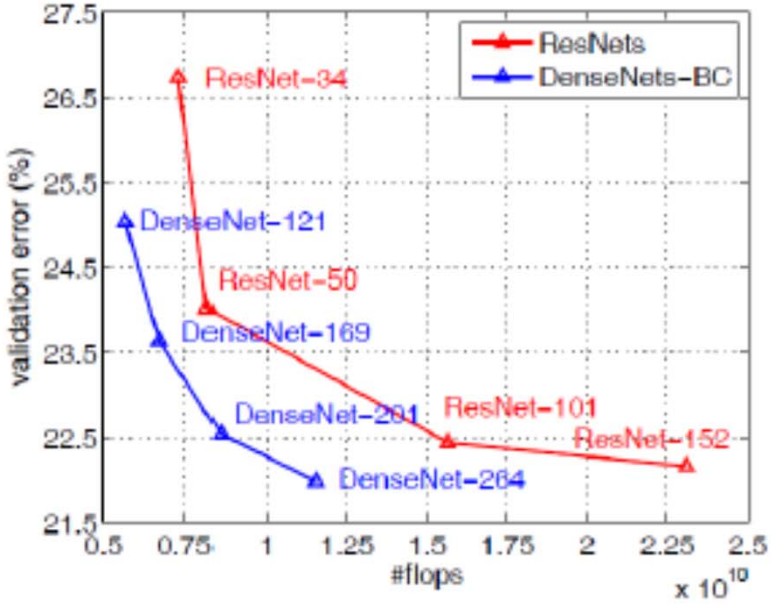
Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	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**	-

+: data augmentation, *: run by the authors

Experimental results: Comparison with ResNet



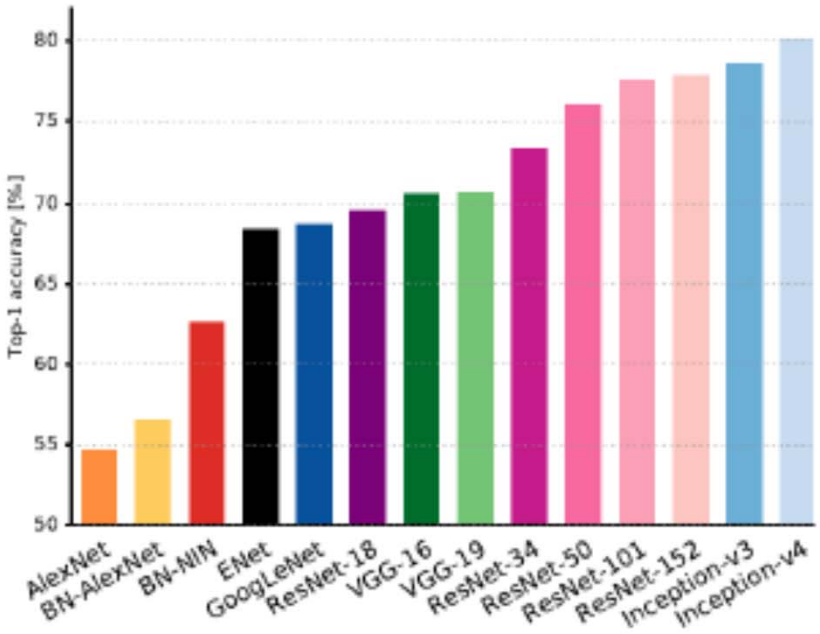
error vs. #parameters



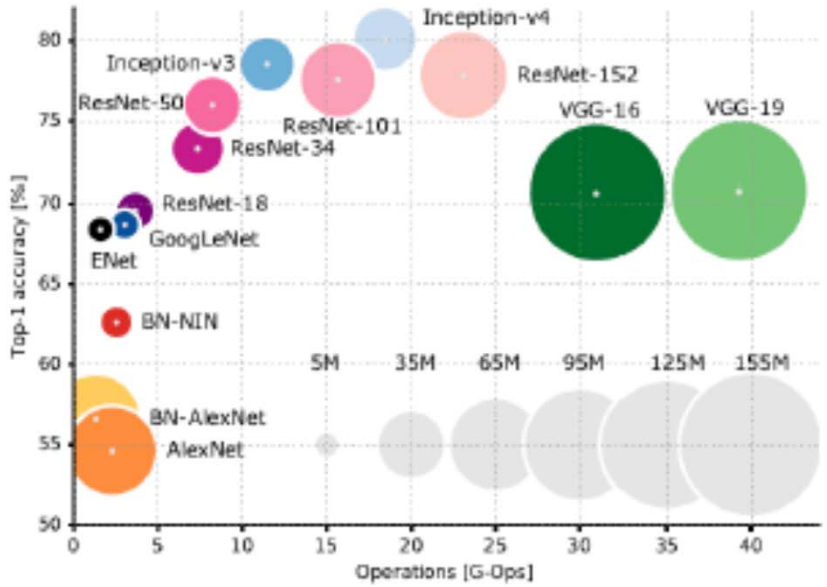
error vs. #flops

Performance comparison

[Canziani et al., arXiv'17]

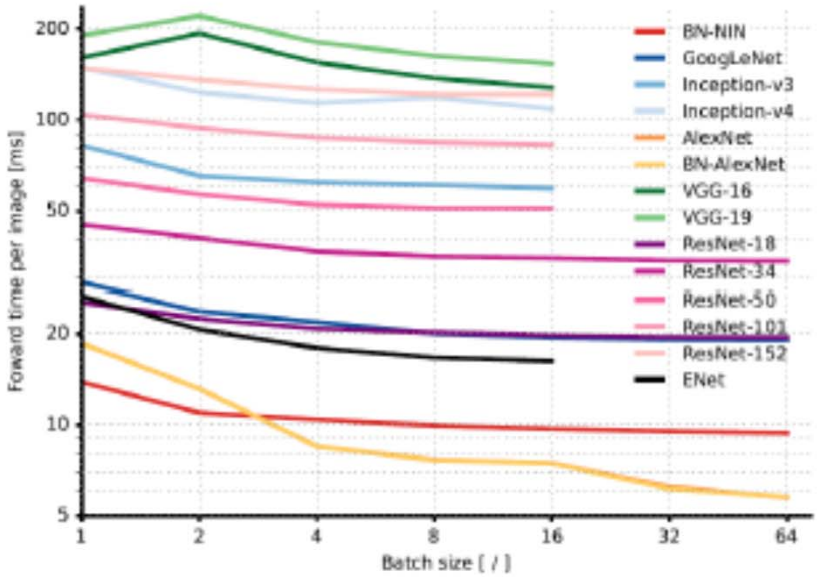


top-1 acc. vs. network

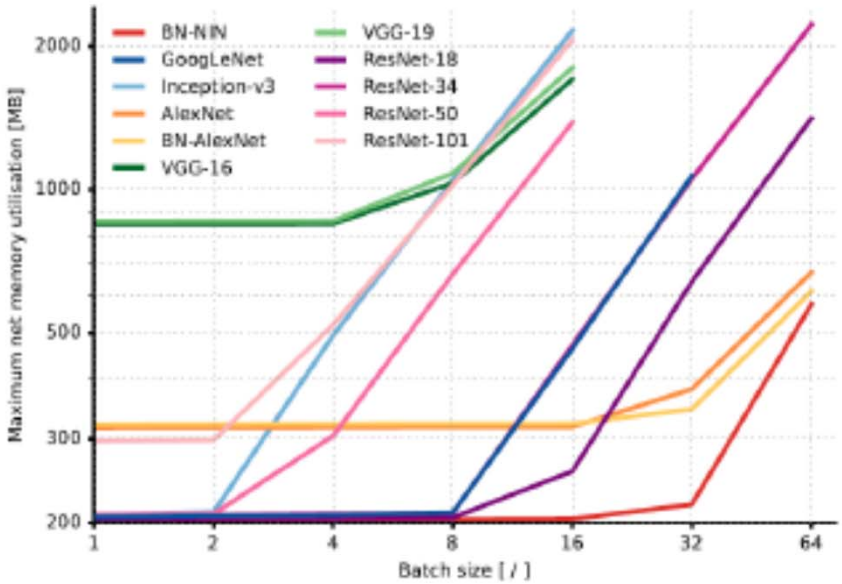


top-1 acc. vs. #operations with #parameters

Performance comparison



inference time vs. batch size



memory size vs. batch size

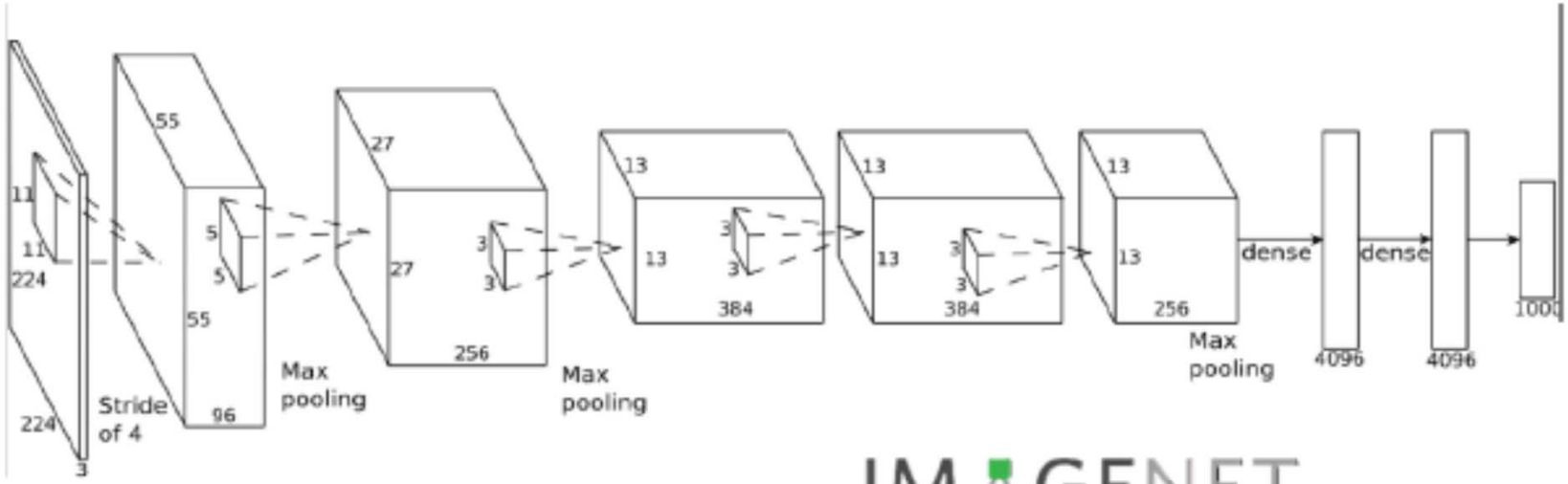
Performance comparison

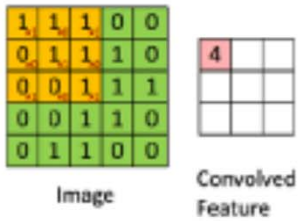
模型名	AlexNet	VGG	GoogLeNet	ResNet
初入江湖	2012	2014	2014	2015
层数	8	19	22	152
Top-5错误	16.4%	7.3%	6.7%	3.57%
Data Augmentation	+	+	+	+
Inception(NIN)	-	-	+	-
卷积层数	5	16	21	151
卷积核大小	11,5,3	3	7,1,3,5	7,1,3,5
全连接层数	3	3	1	1
全连接层大小	4096,4096,1000	4096,4096,1000	1000	1000
Dropout	+	+	+	+
Local Response Normalization	+	-	+	-
Batch Normalization	-	-	-	+

https://goo.gl/sKfjPj

Convolutional Neural Networks

Most state-of-the-art recognition methods are developed upon CNNs.





http://ufldl.stanford.edu/wiki/images/6/6c/Convolution_schematic.gif

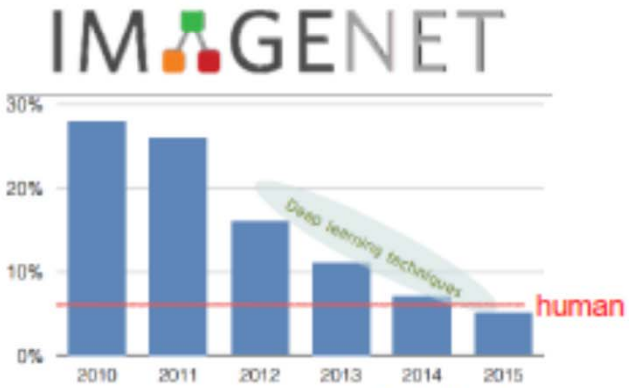


Fig. by Urs Köster

CAFFE (**C**onvolutional **A**rchitecture for **F**ast **F**eature **E**mbedding)

Caffe: Convolutional Architecture for Fast Feature Embedding*

Yangqing Jia*, Evan Shelhamer*, Jeff Donahue, Sergey Karayev,
Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell
UC Berkeley EECS, Berkeley, CA 94702
{jiayq,shelhamer,jdonahue,sergeyk,jonlong,rbg,sguada,trevor}@eecs.berkeley.edu

(Submitted on 20 Jun 2014)

https://arxiv.org/abs/1408.5093

Times cited: 6804 (2018/3/6)

TensorFlow:
Large-Scale Machine Learning on Heterogeneous Distributed Systems
(Preliminary White Paper, November 9, 2015)

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng
Google Research*

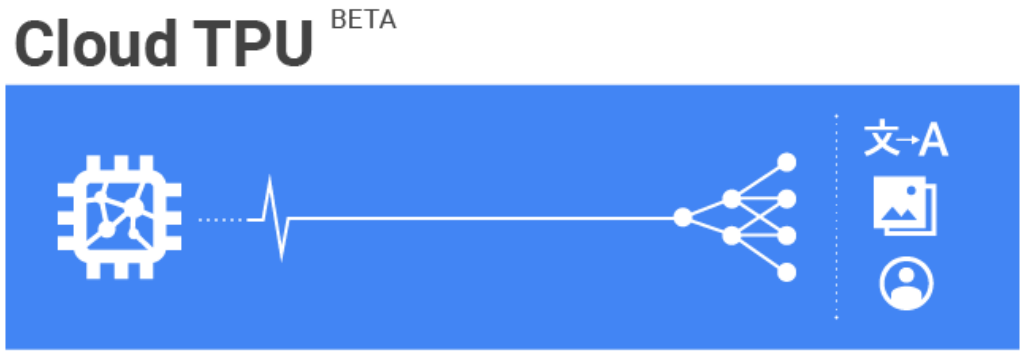
Released on November 9, 2015

Cloud TPU machine learning accelerators now available in beta

Monday, February 12, 2018

By John Barrus, Product Manager for Cloud TPUs, Google Cloud and Zak Stone, Product Manager for TensorFlow and Cloud TPUs, Google Brain Team

Starting today, Cloud TPUs are available in beta on Google Cloud Platform (GCP) to help machine learning (ML) experts train and run their ML models more quickly.

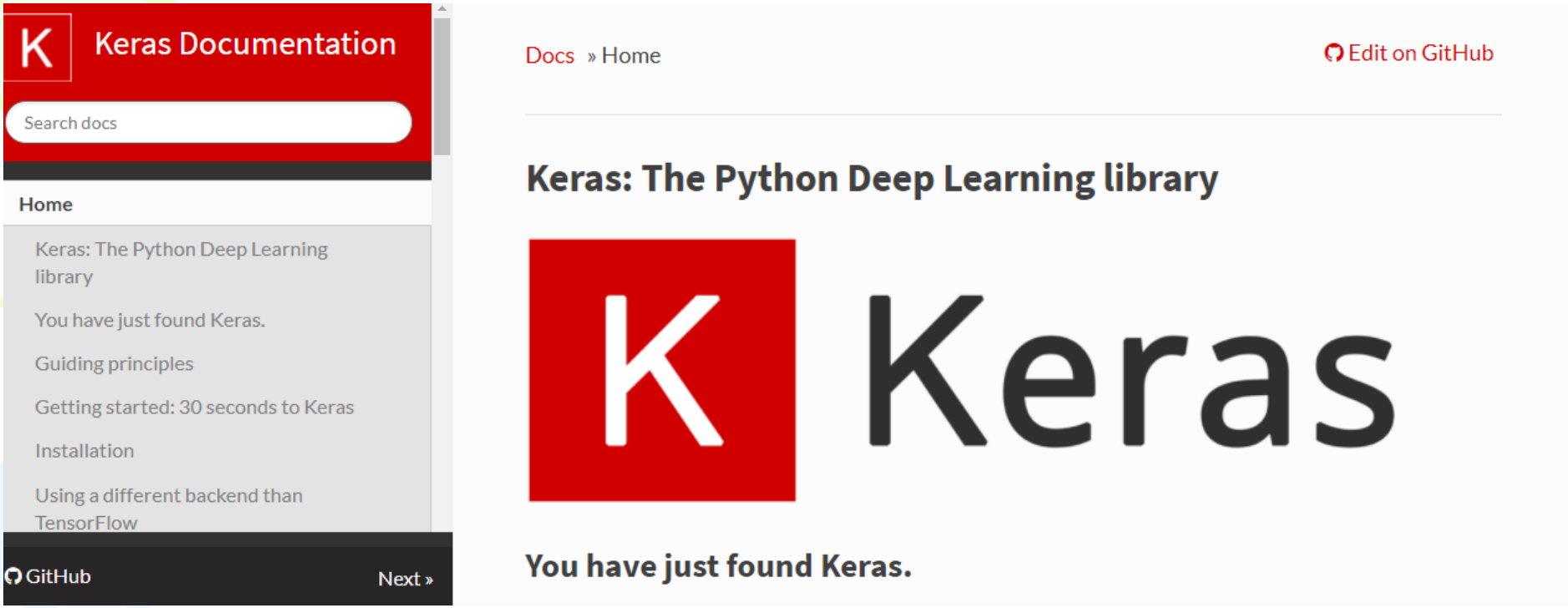


CXO

Google offers free 15-hr machine learning crash course as part of AI resource center

The Learn with Google AI website provides ways for people to develop and hone machine learning skills, and apply the technology to real-world problems.

By Alison DeNisco Rayome | March 1, 2018, 6:32 AM PST



Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research.*

Software	Creator	Software license[a]	Open source	Platform	Written in	Interface	OpenMP support	OpenCL support	CUDA support	Automatic differentiation[1]	Has pretrained models	Recurrent nets	Convolutional nets	RBM/DBNs	Parallel execution (multi node)
Caffe	Berkeley Vision and Learning Center	BSD license	Yes	Linux, macOS, Windows[2]	C++	Python, MATLAB	Yes	Under development[3]	Yes	Yes	Yes[4]	Yes	Yes	No	?
Caffe2	Facebook	Apache 2.0	Yes	Linux, macOS, Windows[5]	C++, Python	Python, MATLAB	Yes	Under development[6]	Yes	Yes	Yes[7]	Yes	Yes	No	Yes
eeplearning4j	Skymind engineering team; Deeplearning4j community; originally Adam Gibson	Apache 2.0	Yes	Linux, macOS, Windows, Android (Cross-platform)	C++, Java	Java, Scala, Clojure, Python (Keras), Kotlin	Yes	On roadmap[8]	Yes[9][10]	Computational Graph	Yes[11]	Yes	Yes	Yes	Yes[12]
Dlib	Davis King	Boost Software License	Yes	Cross-Platform	C++	C++	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes
Intel Data Analytics Acceleration Library	Intel	Apache License 2.0	Yes	Linux, macOS, Windows on Intel CPU[13]	C++, Python, Java	C++, Python, Java[13]	Yes	No	No	Yes	No		Yes		Yes
Intel Math Kernel Library	Intel	Proprietary	No	Linux, macOS, Windows on Intel CPU[14]		C[15]	Yes[16]	No	No	Yes	No	Yes[17]	Yes[17]		No
Keras	François Chollet	MIT license	Yes	Linux, macOS, Windows	Python	Python, R	Only if using Theano as backend	Under development for the Theano backend (and on roadmap for the TensorFlow backend)	Yes	Yes	Yes[18]	Yes	Yes	Yes	Yes[19]
MatConvNet	Andrea Vedaldi, Karel Lenc	BSD license	Yes	Windows, Linux[20] (macOS via Docker on roadmap)	C++	MATLAB, C++,	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes
MATLAB + Neural Network Toolbox	MathWorks	Proprietary	No	Linux, macOS, Windows	C, C++, Java, MATLAB	MATLAB	No	No	Train with Parallel Computing Toolbox and generate CUDA code with GPU	No	Yes[22][23]	Yes[22]	Yes[22]	No	With Parallel Computing Toolbox[24]

Microsoft Cognitive Toolkit	Microsoft Research	MIT license[25]	Yes	Windows, Linux[20] (macOS via Docker on roadmap)	C++	Python (Keras), C++, Command line,[26] BrainScript[27] (.NET on roadmap[28])	Yes[29]	No	Yes	Yes	Yes[30]	Yes[31]	Yes[31]	No[32]	Yes[33]
Apache MXNet	Apache Software Foundation	Apache 2.0	Yes	Linux, macOS, Windows,[34][35] AWS, Android,[36] iOS, JavaScript[37]	Small C++ core library	C++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl	Yes	On roadmap[38]	Yes	Yes[39]	Yes[40]	Yes	Yes	Yes	Yes[41]
Neural Designer	Artelnics	Proprietary	No	Linux, macOS, Windows	C++	Graphical user interface	Yes	No	No	?	?	No	No	No	?
OpenNN	Artelnics	GNU LGPL	Yes	Cross-platform	C++	C++	Yes	No	Yes	?	?	No	No	No	?
PaddlePaddle	Baidu PaddlePaddle team	Apache 2.0	Yes	Linux, macOS, Android,[42] Raspberry Pi[43]	C++, Go	C/C++, Python	Yes	No	Yes	Yes	Yes[44]	Yes	Yes	No	Yes
PyTorch	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan	BSD license	Yes	Linux, macOS	Python, C, CUDA	Python	Yes		Yes	Yes	Yes	Yes			Yes
Apache SINGA	Apache Incubator	Apache 2.0	Yes	Linux, macOS, Windows	C++	Python, C++, Java	No	No	Yes	?	Yes	Yes	Yes	Yes	Yes
TensorFlow	Google Brain team	Apache 2.0	Yes	Linux, macOS, Windows[45]	C++, Python	Python (Keras), C/C++, Java, Go, R[46]	No	On roadmap[47] but already with SYCL[48] support	Yes	Yes[49]	Yes[50]	Yes	Yes	Yes	Yes
Theano	Université de Montréal	BSD license	Yes	Cross-platform	Python	Python (Keras)	Yes	Under development[51]	Yes	Yes[52][53]	Through Lasagne's model zoo[54]	Yes	Yes	Yes	Yes[55]
Torch	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet	BSD license	Yes	Linux, macOS, Windows,[56] Android,[57] iOS	C, Lua	Lua, LuaJIT,[58] C, utility library for C++/OpenCL[59]	Yes	Third party implementations[60][61]	Yes[62][63]	Through Twitter's Autograd[64]	Yes[65]	Yes	Yes	Yes	Yes[66]
Wolfram Mathematica	Wolfram Research	Proprietary	No	Windows, macOS, Linux, Cloud computing	C++	Wolfram Language	No	No	Yes	Yes	Yes[67]	Yes	Yes	Yes	Yes
LaonSill	Laonbud	Apache 2.0	Yes	Linux, Cloud computing	C++	Python	No	No	Yes	No	Yes[68]	No	Yes	No	Yes

Molecular Descriptors

- **Topological Descriptors:** deal with the type and connection of atoms in 2D space.
- **Geometrical Descriptors:** deal with the arrangement of atoms in 3D space, in terms of bond length, angles, and dihedral angles.
- **Electronic Descriptors:** deal with the electronic distribution resulting from the molecular wave function.

Usages of Molecular Descriptors

- Search for similar compounds based on fragments with calculated molecular descriptors. Coarse-grained virtual screening.
- Analyze the chemical diversity of a compound library.
- Perform the quantitative structure-activity relationships (QSAR).
- Predict the physical chemical or pharmacokinetic properties of a novel compound.

Energy-Related Descriptors

Total energy

$$E_{total} = E_{el} + \sum_{A \neq B} \frac{Z_A + Z_B}{R_{AB}}$$

Ionization energy

$$\mathrm{IE} = E_{total}(\mathrm{A}^+) - E_{total}(\mathrm{A})$$

Electron affinity

$$\mathrm{EA} = E_{total}(\mathrm{A}) - E_{total}(\mathrm{A}^-)$$

Energy of protonation

$$\Delta E = E_{total}(\mathrm{BH}^+) - E_{total}(\mathrm{B})$$

Electronic exchange energy

$$E_{exc}(\mathrm{AB}) = \sum_{\mu,\nu \in A} \sum_{\lambda,\sigma \in A} P_{\mu\lambda} P_{\nu\sigma} \langle \mu\lambda | \nu\sigma \rangle$$

Resonance energy

$$E_R(\mathrm{AB}) = \sum_{\mu \in A} \sum_{\nu \in B} P_{\mu\lambda} \beta_{\mu\nu}$$

Heat of formation

$$\Delta H_f^0 = H_f - \sum_A H_f^A$$

Ionization Potential, Electron Affinity, and Electronegativity

Ionization potential:

$$-I = E_N - E_{N-1} = \int_0^1 \varepsilon_{HOMO}(n)dn \approx \varepsilon_{HOMO}\Big|_{n=\frac{1}{2}}$$

Electron affinity:

$$-A = E_{N+1} - E_N = \int_0^1 \varepsilon_{LUMO}(n)dn \approx \varepsilon_{LUMO}\Big|_{n=\frac{1}{2}}$$

Electronegativity:

$$\chi = \frac{I + A}{2} \approx -\frac{\varepsilon_{HOMO}\Big|_{n=\frac{1}{2}} + \varepsilon_{LUMO}\Big|_{n=\frac{1}{2}}}{2}$$

Electrostatic Descriptors

Sum of absolute values of charges

$$QT = \sum_{i=1}^{N} |q_i|$$

Sum of squared charges

$$Q^2 = \sum_{i=1}^{N} q_i^2$$

Molecular dipole moment

$$\boldsymbol{\mu} = \sum_{i=1}^{N} q_i \mathbf{r_i}$$

Partial positively charged surface area

$$PPSA1 = \sum_a S_a \ , a \in \{q_a > 0\}$$

Partial negatively charged surface

$$PNSA1 = \sum_a S_a \ , a \in \{q_a < 0\}$$

Total charge weighted partial positively charged surface area

$$PPSA2 = \sum_a q_a \cdot \sum_a S_a \ , a \in \{q_a > 0\}$$

Atomic charge weighted partial positively charged surface area

$$PPSA3 = \sum_a q_a S_a \ , a \in \{q_a > 0\}$$

MO-related Descriptors

Absolute hardness

$$\eta = \frac{\varepsilon_{LUMO} - \varepsilon_{HOMO}}{2}$$

Nucleophilic atomic frontier electron densities

$$f_r^N = \sum_j \left(c_{LUMO,j}\right)^2$$

Electrophilic atomic frontier electron densities

$$f_r^E = \sum_j \left(c_{HOMO,j}\right)^2$$

Activation hardness

$$\Delta\eta = \eta_R - \eta_T$$

R : reactant

T : transition state

Nucleophilic superdelocalizability

$$S_{E,A} = 2\sum_j \sum_{m=1}^{N_A} \frac{\left(c_{jm}^A\right)^2}{\varepsilon_j}$$

Summation over occupied MOs (j) and over the valence AOs in the atom A (m).

Electrophilic superdelocalizability

$$S_{E,A} = 2\sum_j \sum_{m=1}^{N_A} \frac{\left(c_{jm}^A\right)^2}{\varepsilon_j}$$

Summation over unoccupied MOs (j) and over the valence AOs in the atom A (m).

Review

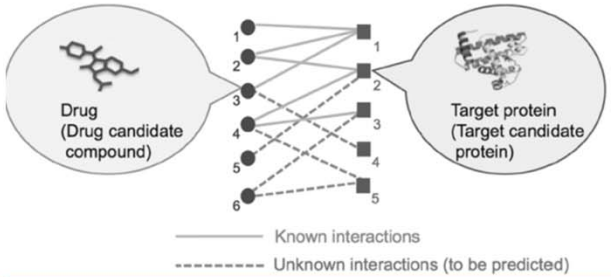
www.molinf.com

molecular informatics

DOI: 10.1002/minf.201400066

Benchmarking a Wide Range of Chemical Descriptors for Drug-Target Interaction Prediction Using a Chemogenomic Approach

Ryusuke Sawada,[a] Masaaki Kotera,[b] and Yoshihiro Yamanishi*[a, c]



Abstract: The identification of drug-target interactions, or interactions between drug candidate compounds and target candidate proteins, is a crucial process in genomic drug discovery. In silico chemogenomic methods are recently recognized as a promising approach for genome-wide scale prediction of drug-target interactions, but the prediction performance depends heavily on the descriptors and similarity measures of drugs and proteins. In this paper, we investigated the performance of various descriptors and similarity measures of drugs and proteins for the drug-target interaction prediction using a chemogenomic approach. We compared the prediction accuracy of 18 chemical descriptors of drugs (e.g., ECFP, FCFP,E-state, CDK, Klekota–Roth, MACCS, PubChem, Dragon, KCF-S, and graph kernels) and 4 descriptors of proteins (e.g., amino acid composition, domain profile, local sequence similarity, and string kernel) on about one hundred thousand drug-target interactions. We examined the combinatorial effects of drug descriptors and protein descriptors using the same benchmark data under several experimental conditions. Large-scale experiments showed that our proposed KCF-S descriptor worked the best in terms of prediction accuracy. The comparative results are expected to be useful for selecting chemical descriptors in various pharmaceutical applications.

Keywords: Chemogenomics · Descriptors · Fingerprint · Drug-target interactions · Machine learning

Applied Soft Computing 62 (2018) 251–258

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Full length article

Deep neural network in QSAR studies using deep belief network

Fahimeh Ghasemi[a], Alireza Mehridehnavi[a], Afshin Fassihi[b], Horacio Pérez-Sánchez[c,*]

[a] Department of Bioinformatic and Systems Biology, School of Advanced Technologies in Medicine, Isfahan University of Medical Sciences, Hezar-Jerib Ave., Isfahan 81746 73461, Islamic Republic of Iran

[b] Department of Medicinal Chemistry, School of Pharmacy and Pharmaceutical Sciences, Isfahan University of Medical Sciences, Hezar-Jerib Ave., Isfahan, Islamic Republic of Iran

[c] Bioinformatics and High Performance Computing Reserch Group (BIO-HPC), Computer Engineering Department, Universidad Católica de Murcia (UCAM), E30107 Murcia, Spain

There are two major challenges in the current high throughput screening drug design: the large number of descriptors which may also have autocorrelations and, proper parameter initialization in model prediction to avoid over-fitting problem. Deep architecture structures have been recommended to predict the compounds biological activity. Performance of deep neural network is not always acceptable in QSAR studies. This study tries to find a solution to this problem focusing on primary parameter computation. Deep belief network has been getting popular as a deep neural network model generation method in other fields such as image processing. In the current study, deep belief network is exploited to initialize deep neural networks. All fifteen targets of Kaggle data sets containing more than 70 k molecules have been utilized to investigate the model performance. The results revealed that an optimization in parameter initialization will improve the ability of deep neural networks to provide high quality model predictions. The mean and variance of squared correlation for the proposed model and deep neural network are $0.618 \pm 0.407e-4$ and $0.485 \pm 4.82e-4$, respectively. The outputs of this model seem to outperform those of the models obtained from deep neural network.

Enzymatic Reactions

Anhydroase inhibition activity

$$\log \Pi_{50} = 37.84 q_{SO_2NH_2} + 8.78$$

charge of the $-SO_2NH_2$ group

$$n = 28, r = 0.909, s = 0.336, F = 123.2$$

Inhibition potency of indanone-benzylpiperidine inhibitors of acetylcholinesterase

$$-\log IC_{50} = -757.52 + 2.21 C_4 - 162.9 E_{HOMO} - 8.85 E_{HOMO}^2 - 6.65\mu + 1.18\mu^2$$

C_4: the HOMO out-of-plane πorbital coefficient of the ring carbon

De Benedetti et al. Quant. Struct. Act. Relat. 6:51-53 (1987)
Cardozo et al. J. Med. Chem. 35: 584 (1992)

The Hammett Equation

Hammett (1935) studied a series of aromatic compounds and found:

$$\log K_{R-X} - \log K_{R-H} = \rho\sigma$$

$$\log k_{R-X} - \log k_{R-H} = \rho\sigma$$

K : equilibrium constants

k : rate constants

σ : substituent constants, which only depend on the nature of substituents X

ρ : parameter based on ionization constants of substituted benzoid acids

The First QSAR Analysis

$$\log {}^{1}\!/_{C} = -2.14\pi^2 + 4.08\pi + 2.78\sigma + 3.36$$

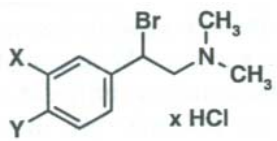
C : the molar dose that produce or prevent a certain biological response

σ : the Hammett parameter

π : calculated lipophilicity constant

Hansch et al. Nature 194:178-180 (1962)

Antiadrenergic activities and physical chemical properties of meta-, para-, and meta,para-disubstituted N,N-dimethyl-α-Bromaophenethylamines



meta (*X*)	*para* (*Y*)	log 1/*C* observed	π	σ^+	E_s^{meta}	log 1/*C* calculated[a]	log 1/*C* calculated[b]
H	H	7.46	0.00	0.00	1.24	7.82	7.88
H	F	8.16	0.15	−0.07	1.24	8.09	8.17
H	Cl	8.68	0.70	0.11	1.24	8.46	8.60
H	Br	8.89	1.02	0.15	1.24	8.77	8.94
H	I	9.25	1.26	0.14	1.24	9.06	9.26
H	Me	9.30	0.52	−0.31	1.24	8.87	8.98
F	H	7.52	0.13	0.35	0.78	7.45	7.43
Cl	H	8.16	0.76	0.40	0.27	8.11	8.05
Br	H	8.30	0.94	0.41	0.08	8.30	8.22
I	H	8.40	1.15	0.36	−0.16	8.61	8.51
Me	H	8.46	0.51	−0.07	0.00	8.51	8.36
Cl	F	8.19	0.91	0.33	0.27	8.38	8.34
Br	F	8.57	1.09	0.34	0.08	8.57	8.51
Me	F	8.82	0.66	−0.14	0.00	8.78	8.65
Cl	Cl	8.89	1.46	0.51	0.27	8.75	8.77
Br	Cl	8.92	1.64	0.52	0.08	8.94	8.94
Me	Cl	8.96	1.21	0.04	0.00	9.15	9.08
Cl	Br	9.00	1.78	0.55	0.27	9.06	9.11
Br	Br	9.35	1.96	0.56	0.08	9.25	9.29
Me	Br	9.22	1.53	0.08	0.00	9.46	9.43
Me	Me	9.30	1.03	−0.38	0.00	9.56	9.47
Br	Me	9.52	1.46	0.10	0.08	9.35	9.33

π = lipophilicity parameter; σ^+ = Hammett constant for benzyl cations; E_s = Taft's steric parameter.
[a] Calculated by Eq. (14).
[b] Calculated by Eq. (16).
Source: Refs. 28 and 30.

The Resultant Hansch Equation

$$\log {}^{1}\!/_{C} = 1.15\pi - 1.47\sigma^{+} + 7.82$$

$n = 22;$
$r = 0.994;$
$s = 0.197$

Or

$$\log {}^{1}\!/_{C} = 1.259(\pm 0.19)\pi - 1.460(\pm 0.34)\sigma^{+} + 0.208(\pm 0.17)E_{s}^{meta} + 7.619$$

$n = 22;$
$r = 0.959;$
$s = 0.173$

C : the molar dose that produce or prevent a certain biological response

σ^{+} : the Hammett constant for benzyl cations

π : calculated lipophilicity parameter

E_s : Taft's steric parameter

Kubinyi *J. Med. Chem.* 19:578-586 (1976)

A QSAR Equation

C = molar concentration that causes a certain biological effect

values of the regression coefficients

95% confidence intervals of the coefficients and the constant term

$$\text{Log } 1/C = 1.151\ (\pm 0.19)\ \pi - 1.464\ (\pm 0.38)\ \sigma^{+} + 7.817\ (\pm 0.19)$$

logarithms of reciprocal values are the correct scaling

lipophilicity parameter

electronic parameter

constant term

$$(n = 22;\ r = 0.945;\ s = 0.196;\ F = 78.63;\ Q^2 = 0.841;\ s_{PRESS} = 0.238)$$

number of compounds

correlation coefficient r; measure of the relative quality of fit of a model

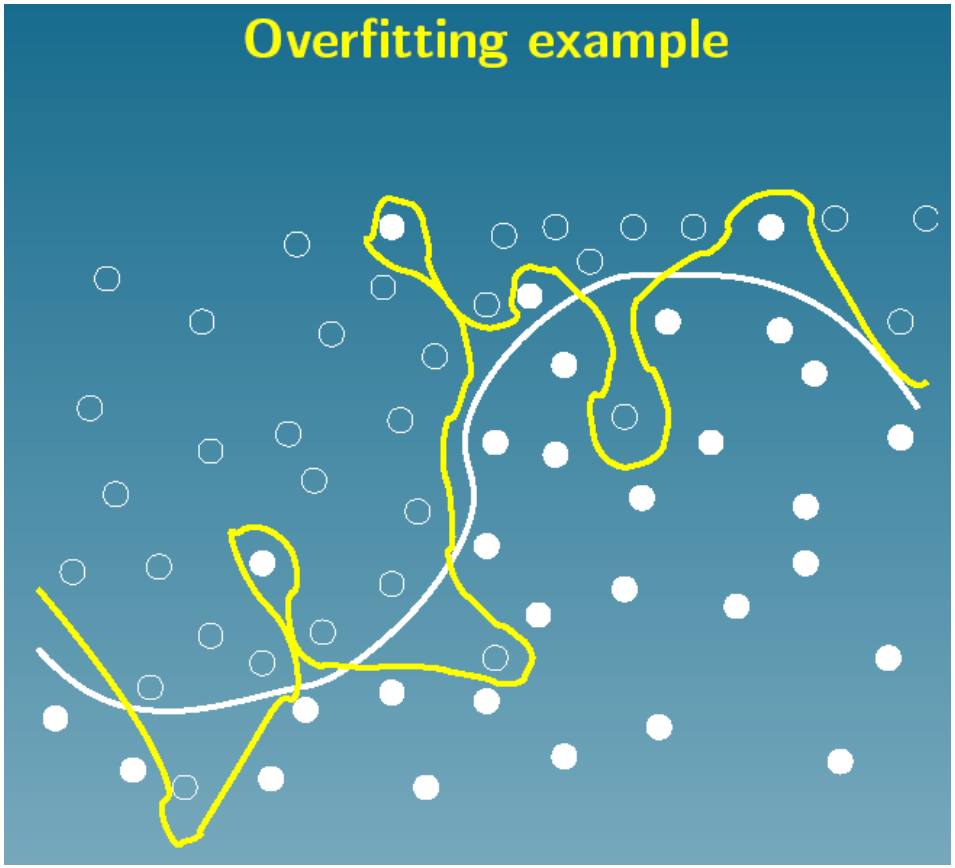
standard deviation s; measure of the absolute quality of fit of a model

Fisher value; measure of the statistical significance

standard deviation of crossvalidation predictions and squared crossvalidation regression coefficient; both are measures of internal predictivity

Validation and Selection of QSAR Models

1. **Selection of independent variables**. A wide range of different parameters, such as logP or π,σ, and steric parameters, should be tried; molecular orbital parameters should not be overlooked.
2. **Justification of the choice** of independent variables. All "reasonable" parameters must be validated by an appropriate statistical procedure. The "best" equation is normally the one with the lowest standard deviation, all terms being significant.
3. **Principle of parsimony** (Occam's Razor; William Ockham, 1285-1349, English philosopher and logician). All things being (approx.) equal, one should accept the simplest model.
4. Number of terms. One should have at least five to six data points per variable to avoid chance correlation.
5. Qualitative model. It is important to have a qualitative model that is **consistent with the known physical-organical and biomedicinal chemistry** of the process under consideration, in order to avoid chance correlations.


Overfitting example

DeepChem

DeepChem aims to provide a high quality open-source toolchain that democratizes the use of deep-learning in drug discovery, materials science, quantum chemistry, and biology. Previous deep learning frameworks, such as scikit-learn have been applied to chemiformatics, but DeepChem is the first to accelerate computation with NVIDIA GPUs.

DeepChem Assay Datasets

Dataset	Category	Description	Classification Type	Compounds
QM7	Quantum Mechanics	orbital energies atomization energies	Regression	7,165
QM7b	Quantum Mechanics	orbital energies	Regression	7,211
ESOL	Physical Chemistry	solubility	Regression	1,128
FreeSolv	Physical Chemistry	solvation energy	Regression	643
PCBA	Biophysics	bioactivity	Classification	439,863
MUV	Biophysics	bioactivity	Classification	93,127
HIV	Biophysics	bioactivity	Classification	41,913
PDBBind	Biophysics	binding activity	Regression	11,908
Tox21	Physiology	toxicity	Classification	8,014
ToxCast	Physiology	toxicity	Classification	8,615
SIDER	Physiology	side reactions	Classification	1,427
ClinTox	Physiology	clinical toxicity	Classification	1,491

MoleculeNet: A Benchmark for Molecular Machine Learning

Zhenqin Wu,[†,‖] Bharath Ramsundar,[‡,‖] Evan N. Feinberg,[¶,⊥] Joseph Gomes,[†,⊥] Caleb Geniesse,[¶] Aneesh S. Pappu,[‡] Karl Leswing,[§] and Vijay Pande[*,†]

†*Department of Chemistry, Stanford University*

‡*Department of Computer Science, Stanford University*

¶*Program in Biophysics, Stanford School of Medicine*

§*Schrodinger Inc.*

‖*Joint First Authorship*

⊥*Joint Second Authorship*

Molecular machine learning has been maturing rapidly over the last few years. Improved methods and the presence of larger datasets have enabled machine learning algorithms to make increasingly accurate predictions about molecular properties. However, algorithmic progress has been limited due to the lack of a standard benchmark to compare the efficacy of proposed methods; most new algorithms are benchmarked on different datasets making it challenging to gauge the quality of proposed methods. This work introduces MoleculeNet, a large scale benchmark for molecular machine learning. MoleculeNet curates multiple public datasets, establishes metrics for evaluation, and offers high quality open-source implementations of multiple previously proposed molecular featurization and learning algorithms (released as part of the DeepChem

DeepChem Featurizers

Featurizer	use cases
Extended-Connectivity Fingerprints (ECFP)	for molecular datasets not containing large numbers of non-bonded interactions
Graph Convolutions	Like ECFP, graph convolution produces granular representations of molecular topology. Instead of applying fixed hash functions, as with ECFP, graph convolution uses a set of parameters which can learned by training a neural network associated with a molecular graph structure.
Coloumb Matrix	Coloumb matrix featurization captures information about the nuclear charge state, and internuclear electric repulsion. This featurization is less granular than ECFP, or graph convolutions, and may perform better where intramolecular electrical potential may play an important role in chemical activity
Grid Featurization	datasets containing molecules interacting through non-bonded forces, such as docked protein-ligand complexes

Atomic Convolutional Networks for Predicting Protein-Ligand Binding Affinity

Joseph Gomes[1,+], Bharath Ramsundar[2,+], Evan N. Feinberg[3], and Vijay S. Pande[1,*]

[1]Department of Chemistry, Stanford University
[2]Department of Computer Science, Stanford University
[3]Program in Biophysics, Stanford University School of Medicine
[*]pande@stanford.edu
[+]these authors contributed equally to this work

Empirical scoring functions based on either molecular force fields or cheminformatics descriptors are widely used, in conjunction with molecular docking, during the early stages of drug discovery to predict potency and binding affinity of a drug-like molecule to a given target. These models require expert-level knowledge of physical chemistry and biology to be encoded as hand-tuned parameters or features rather than allowing the underlying model to select features in a data-driven procedure. Here, we develop a general 3-dimensional spatial convolution operation for learning atomic-level chemical interactions directly from atomic coordinates and demonstrate its application to structure-based bioactivity prediction. The atomic convolutional neural network is trained to predict the experimentally determined binding affinity of a protein-ligand complex by direct calculation of the energy associated with the complex, protein, and ligand given the crystal structure of the binding pose. Non-covalent interactions present in the complex that are absent in the protein-ligand sub-structures are identified and the model learns the interaction strength associated with these features. We test our model by predicting the binding free energy of a subset of protein-ligand complexes found in the PDBBind dataset and compare with state-of-the-art cheminformatics and machine learning-based approaches. We find that all methods achieve experimental accuracy (less than 1 kcal/mol mean absolute error) and that atomic convolutional networks either outperform or perform competitively with the cheminformatics based methods. Unlike all previous protein-ligand prediction systems, atomic convolutional networks are end-to-end and fully-differentiable. They represent a new data-driven, physics-based deep learning model paradigm that offers a strong foundation for future improvements in structure-based bioactivity prediction.

ACS central science

ACS AuthorChoice

Research Article

http://pubs.acs.org/journal/acscii

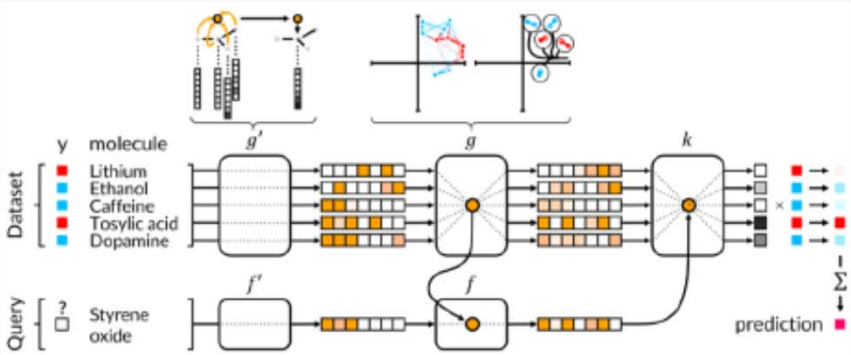
Low Data Drug Discovery with One-Shot Learning

Han Altae-Tran,[†,#] Bharath Ramsundar,[‡,#] Aneesh S. Pappu,[‡] and Vijay Pande*[,§]

[†]Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139-4307, United States

[‡]Department of Computer Science and [§]Department of Chemistry, Stanford University, Stanford, California 94305, United States

ABSTRACT: Recent advances in machine learning have made significant contributions to drug discovery. Deep neural networks in particular have been demonstrated to provide significant boosts in predictive power when inferring the properties and activities of small-molecule compounds (Ma, J. et al. *J. Chem. Inf. Model.* 2015, *55*, 263−274). However, the applicability of these techniques has been limited by the requirement for large amounts of training data. In this work, we demonstrate how one-shot learning can be used to significantly lower the amounts of data required to make meaningful predictions in drug discovery applications. We introduce a new architecture, the iterative refinement long short-term memory, that, when combined with graph convolutional neural networks, significantly improves learning of meaningful distance metrics over small-molecules. We open source all models introduced in this work as part of DeepChem, an open-source framework for deep-learning in drug discovery (Ramsundar, B. deepchem.io. https://github.com/deepchem/deepchem, 2016).



JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

pubs.acs.org/jcim

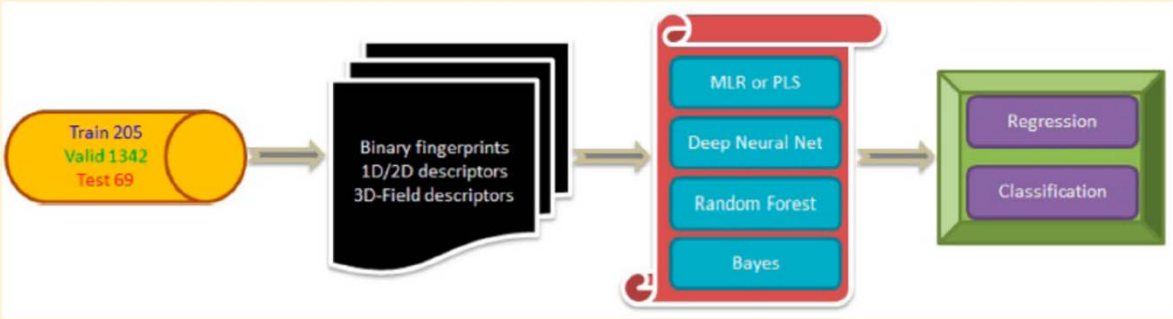
Computational Modeling of β-Secretase 1 (BACE-1) Inhibitors Using Ligand Based Approaches

Govindan Subramanian,[*,§] Bharath Ramsundar,[‡] Vijay Pande,[⊥] and Rajiah Aldrin Denny[†]

[§]VMRD Global Discovery, Zoetis, 333 Portage Street, Kalamazoo, Michigan 49007, United States

[‡]Department of Computer Science and [⊥]Department of Chemistry, Stanford University, 318 Campus Drive, Stanford, California 94305, United States

[†]Worldwide Medicinal Chemistry, Pfizer Inc., 610 Main Street, Cambridge, Massachusetts 02139, United States



ABSTRACT: The binding affinities (IC_{50}) reported for diverse structural and chemical classes of human β-secretase 1 (BACE-1) inhibitors in literature were modeled using multiple in silico ligand based modeling approaches and statistical techniques. The descriptor space encompasses simple binary molecular fingerprint, one- and two-dimensional constitutional, physicochemical, and topological descriptors, and sophisticated three-dimensional molecular fields that require appropriate structural alignments of varied chemical scaffolds in one universal chemical space. The affinities were modeled using qualitative classification or quantitative regression schemes involving linear, nonlinear, and deep neural network (DNN) machine-learning methods used in the scientific literature for quantitative–structure activity relationships (QSAR). In a departure from tradition, ~20% of the chemically diverse data set (205 compounds) was used to train the model with the remaining ~80% of the structural and chemical analogs used as part of an external validation (1273 compounds) and prospective test (69 compounds) sets respectively to ascertain the model performance. The machine-learning methods investigated herein performed well in both the qualitative classification (~70% accuracy) and quantitative IC_{50} predictions (RMSE ~ 1 log). The success of the 2D descriptor based machine learning approach when compared against the 3D field based technique pursued for *h*BACE-1 inhibitors provides a strong impetus for systematically applying such methods during the lead identification and optimization efforts for other protein families as well.

JOURNAL OF
CHEMICAL INFORMATION
AND MODELING

Article

pubs.acs.org/jcim

Protein–Ligand Scoring with Convolutional Neural Networks

Matthew Ragoza,[†,‡] Joshua Hochuli,[‡,¶] Elisa Idrobo,[§] Jocelyn Sunseri,[∥] and David Ryan Koes*[,∥]

[†]Department of Neuroscience, [‡]Department of Computer Science, [¶]Department of Biological Sciences, and [∥]Department of Computational and Systems Biology, University of Pittsburgh, Pittsburgh, Pennsylvania 15260, United States

[§]Department of Computer Science, The College of New Jersey, Ewing, New Jersey 08628, United States

ABSTRACT: Computational approaches to drug discovery can reduce the time and cost associated with experimental assays and enable the screening of novel chemotypes. Structure-based drug design methods rely on scoring functions to rank and predict binding affinities and poses. The ever-expanding amount of protein–ligand binding and structural data enables the use of deep machine learning techniques for protein–ligand scoring. We describe convolutional neural network (CNN) scoring functions that take as input a comprehensive three-dimensional (3D) representation of a protein–ligand interaction. A CNN scoring function automatically learns the key features of protein–ligand interactions that correlate with binding. We train and optimize our CNN scoring functions to discriminate between correct and incorrect binding poses and known binders and nonbinders. We find that our CNN scoring function outperforms the AutoDock Vina scoring function when ranking poses both for pose prediction and virtual screening.

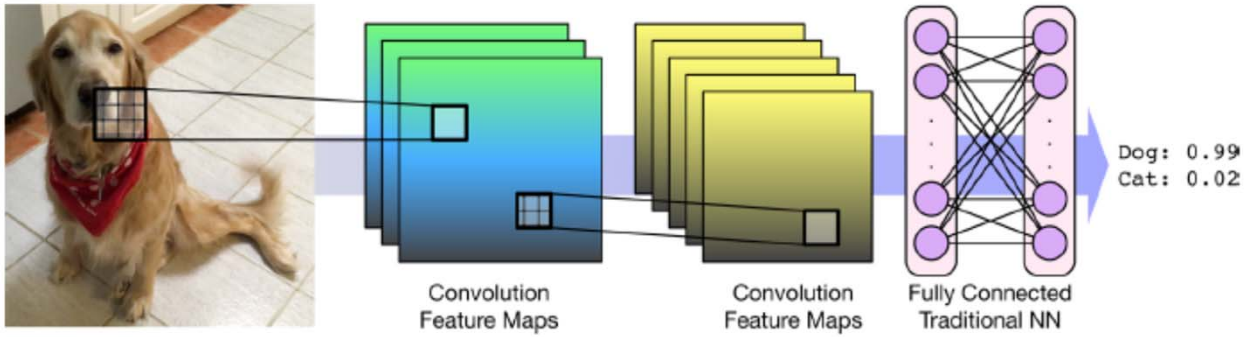


Figure 1. Classical convolutional neural network for image recognition. The first layer applies three different convolutions to the input image to create three maps of low level features that are the input for another convolutional layer that creates five maps. Feature maps preserve the spatial locality of the features. As a last step, a traditional neural net is applied to generate a classification.

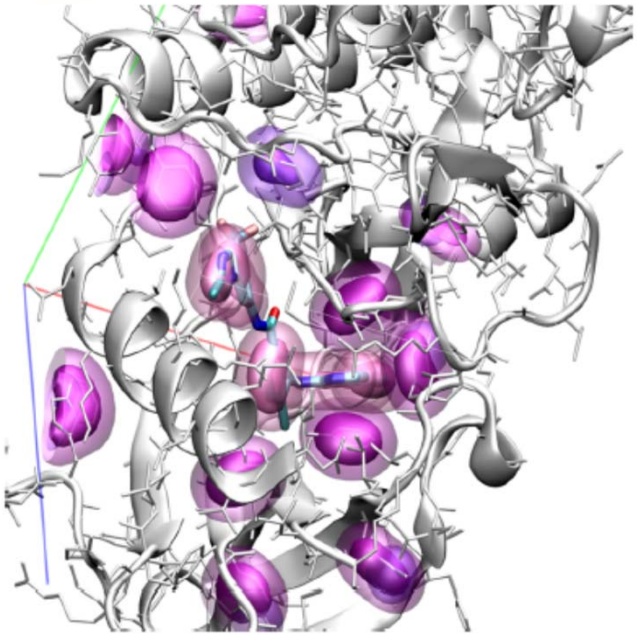


Figure 2. Visualization of atom densities used as input to CNN scoring. Aromatic carbon atom densities are shown at two isosurface levels (solid and transparent surfaces) for both the receptor (purple) and ligand (lavender).

Our default is to use smina[1] atom types for a total of 34 distinct types with 16 receptor types and 18 ligand types as shown in Table S1. Only smina atom types that were present in the ligands and proteins of the training set were retained. For example, halogens are not included as receptor atom types and metals are not included as ligand atom types. Hydrogen atoms are ignored except to determine acceptor/donor atom types. We also evaluate alternative atom typing schemes. Atom type information is represented as a density distribution around the atom center. We represent each atom as a function $A(d, r)$ where d is the distance from the atom center and r is the van der Waals radius:

$$A(d, r) = \begin{cases} e^{-2d^2/r^2} & 0 \le d < r \\ \frac{4}{e^2 r^2} d^2 - \frac{12}{e^2 r} d + \frac{9}{e^2} & r \le d < 1.5r \\ 0 & d \ge 1.5r \end{cases} \quad (1)$$

A is a continuous piecewise combination of a Gaussian (from the center to the van der Waals radius) and a quadratic (which goes to zero at 1.5 times the radius). This provides a continuous representation of the input. We also evaluate a "hard" discrete boolean representation.

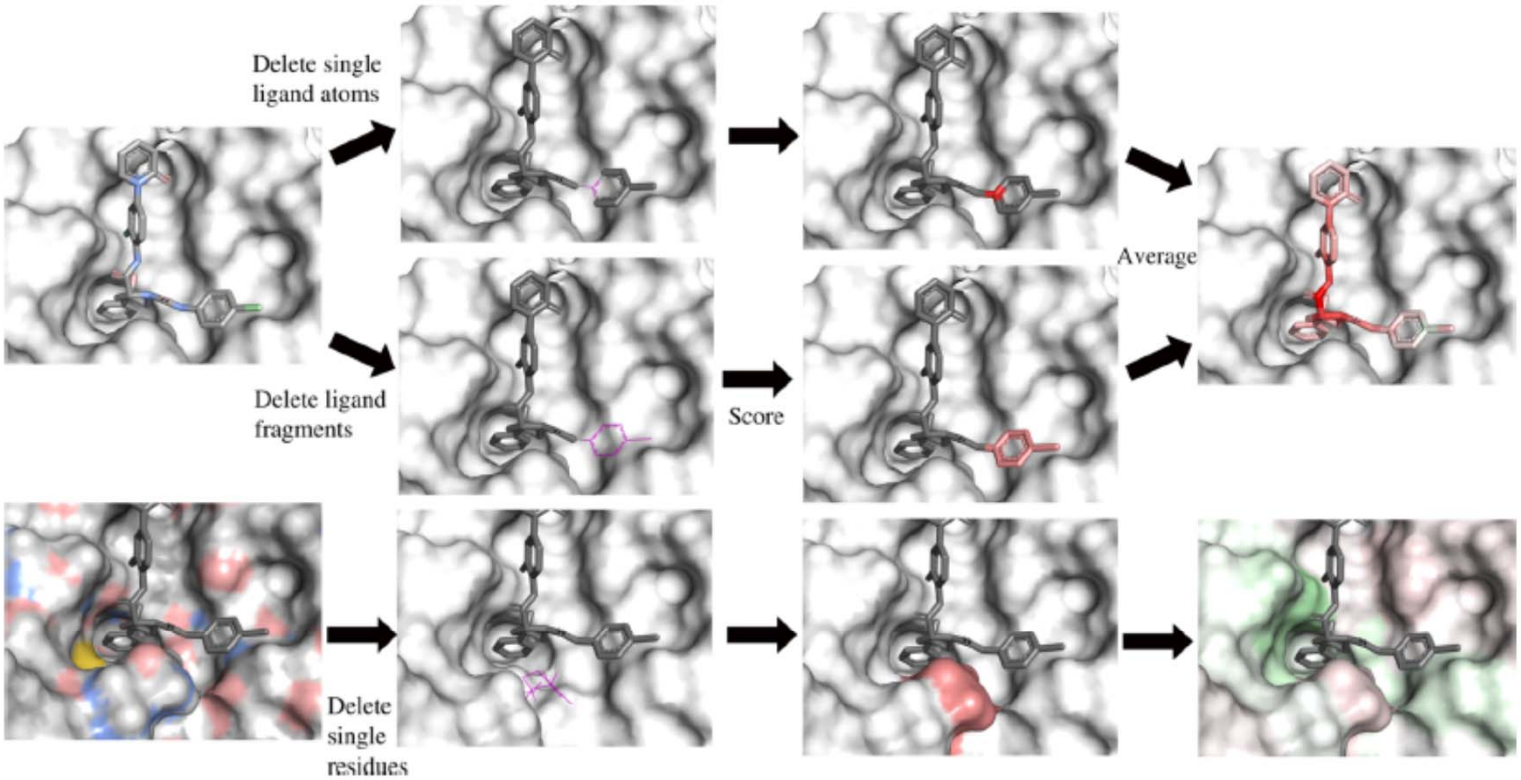


Figure 4. Visualization algorithm. In the ligand, atoms are removed individually or as fragments and each modified molecule is scored. The assigned color is the difference between the unmodified protein–ligand score and the score with the removed atom. The protein is treated similarly, but whole residues are removed. Positive score differences indicate a positive contribution by the atom to the overall score and are colored green, with the intensity depending on the magnitude of difference. Red represented negative score differences.

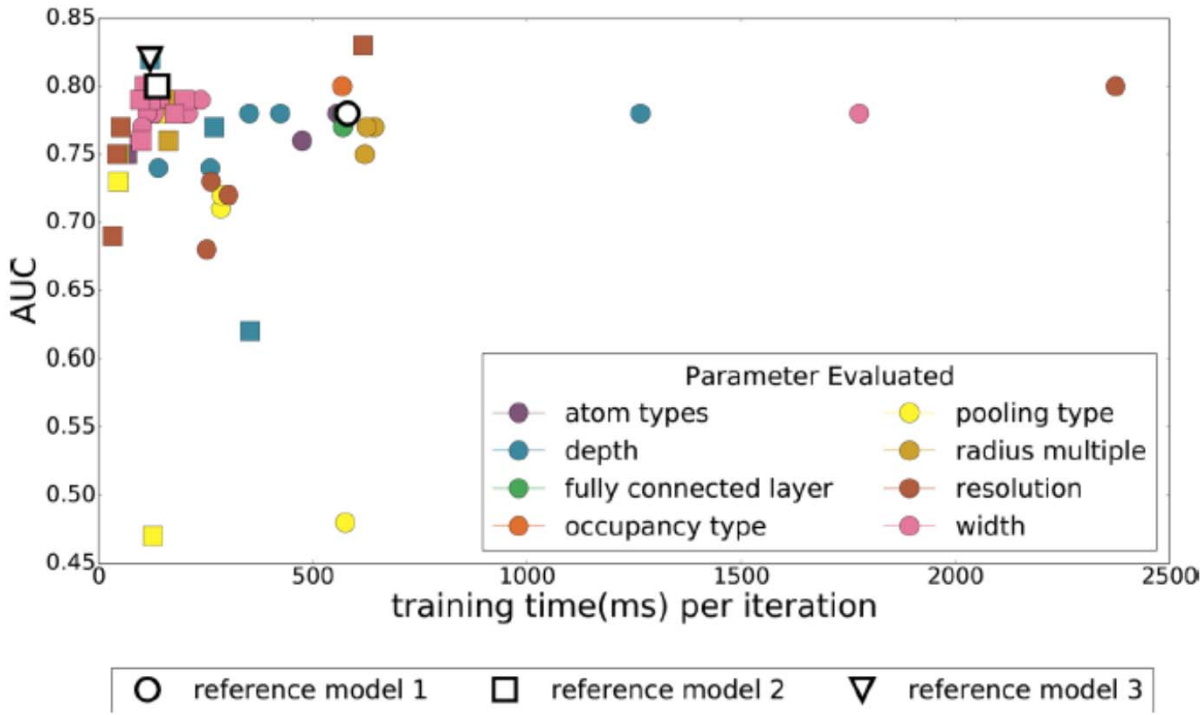


Figure 5. Training time and average cross-validation AUC of various models created by systematically varying parameters. Marker shape indicates iteration of optimization and the color what parameter was varied.

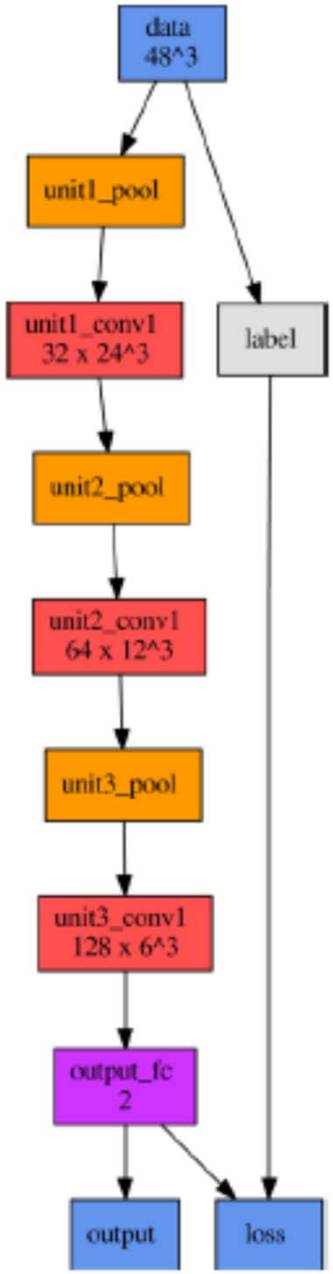


Figure 6. Network architecture of our final model.

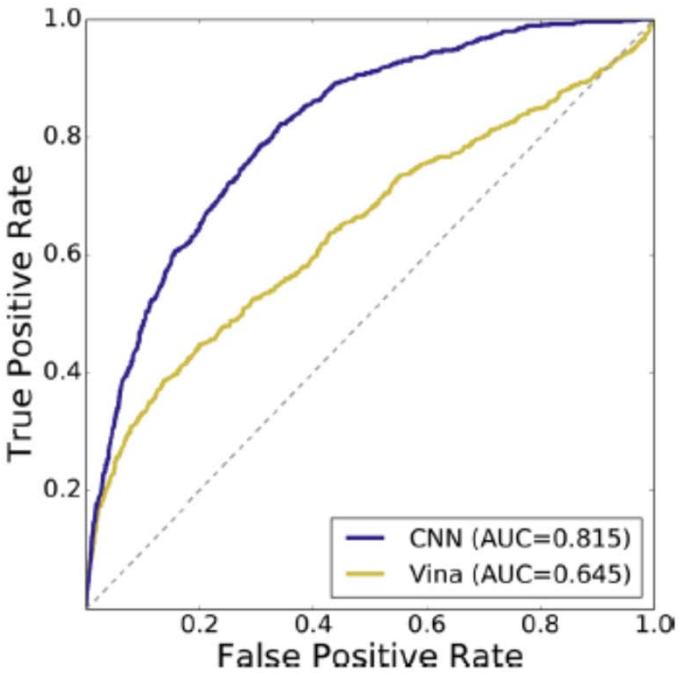


Figure 7. Intertarget cross-validated ROC curve of CNN scoring method compared to Autodock Vina on the CSAR pose prediction data set. The CNN performs better at classifying generated poses as low or high RMSD across targets.

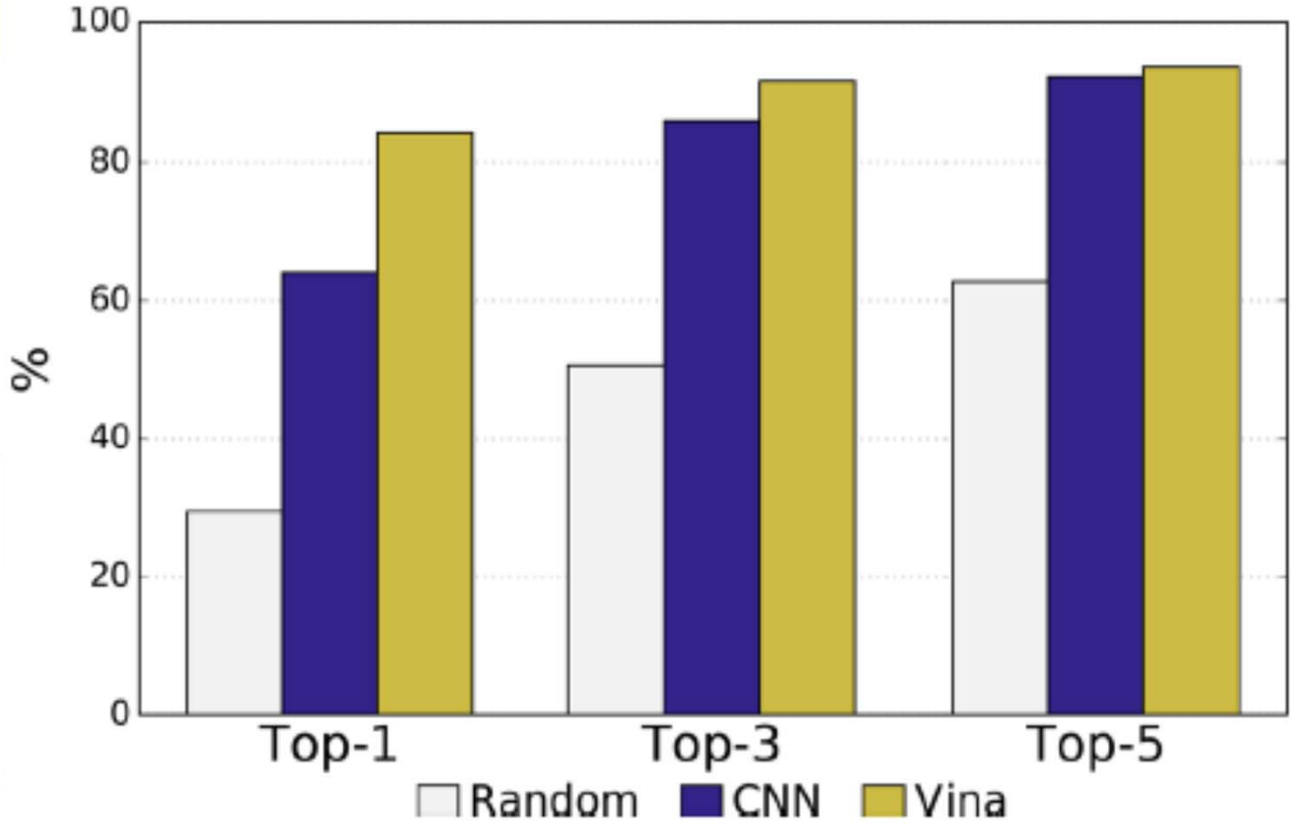


Figure 8. Intratarget pose ranking. The percent of targets with a low RMSD pose ranked as the top one, three, or five poses is shown. Vina and CNN have similar recovery rates among the top-5 ranked poses, but Vina more often ranks a low RMSD pose as the top-1 ranked pose.

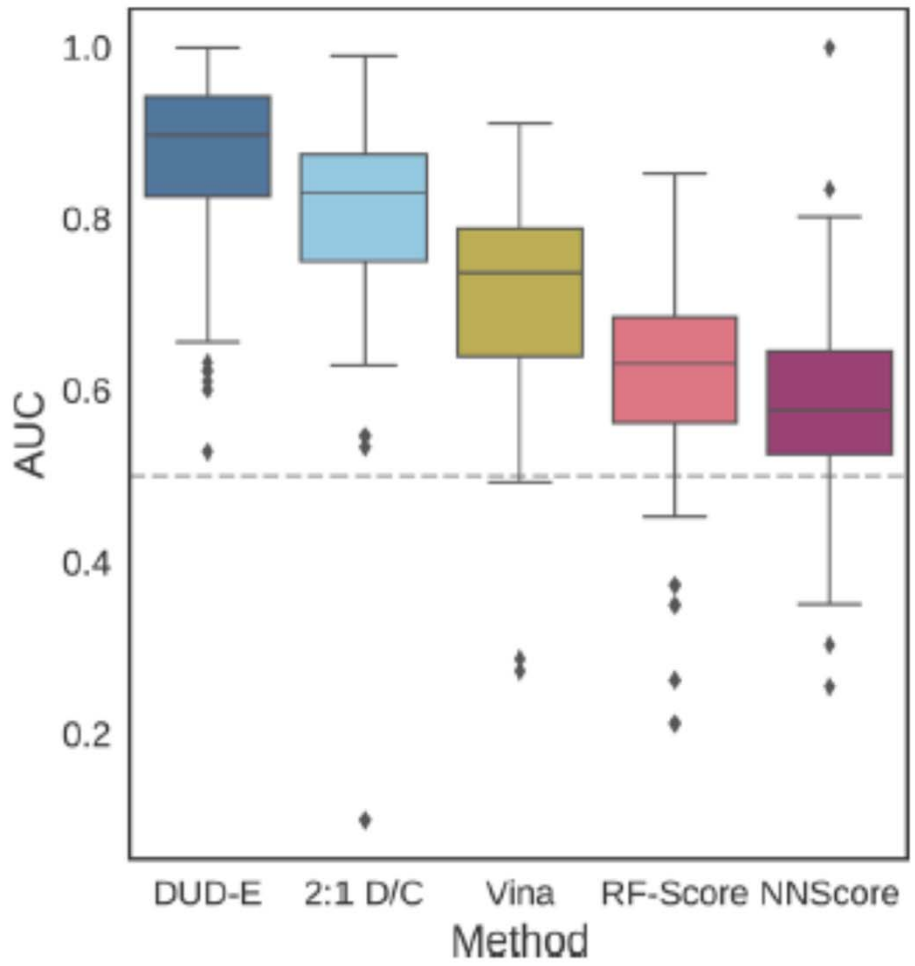


Figure 9. Distribution of the area under the ROC curve for targets of the DUD-E data set for the pose-insensitive CNN model trained only on DUD-E, the pose-sensitive DUD-E/CSAR 2:1 model, Vina, RF-Score, and NNScore.

Table 1. Mean AUC and ROC Enrichment (RE) Across Targets in the DUD-E Dataset for CNN Models, Vina, RF-Score, and NNScore

metric	DUD-E	2:1 D/C	Vina	RF-Score	NNScore
AUC	**0.868**	0.804	0.716	0.622	0.584
0.5% RE	**42.559**	22.366	9.139	5.628	4.166
1.0% RE	**29.654**	16.274	7.321	4.274	2.980
2.0% RE	**19.363**	11.888	5.881	3.499	2.460
5.0% RE	**10.710**	7.376	4.444	2.678	1.891

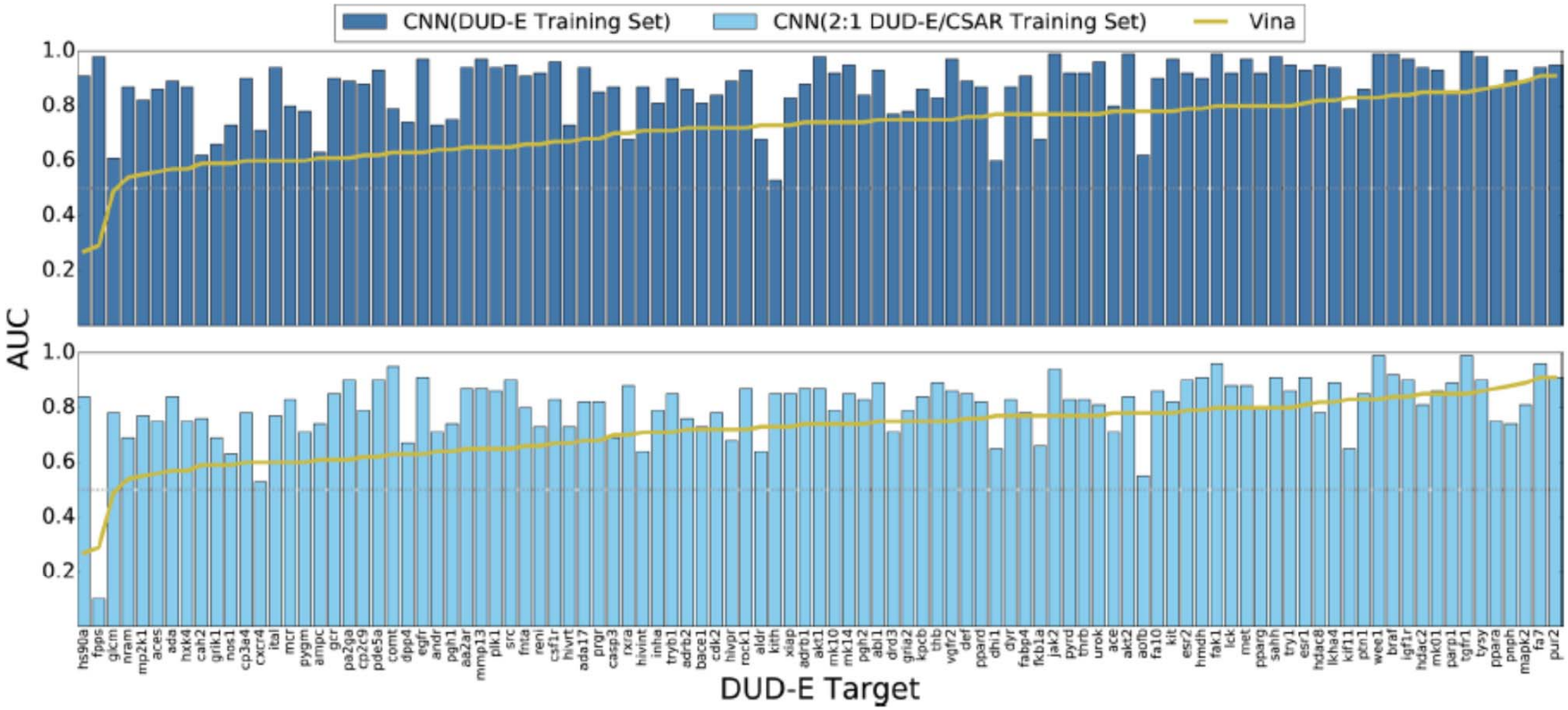


Figure 11. Cross-validation performance of CNN models on the DUD-E virtual screening benchmark compared to the Vina scoring function. Targets are sorted by performance with Vina. Identical sets of docked poses were ranked. The score of the top ranked pose of each ligand is used to predict activity (multipose scoring). CNN models trained only on DUD-E training data perform best, outperforming Vina in 90% of the targets. Models trained using a mix of DUD-E and CSAR data also perform well, achieving better AUCs than Vina in 81% of the targets.

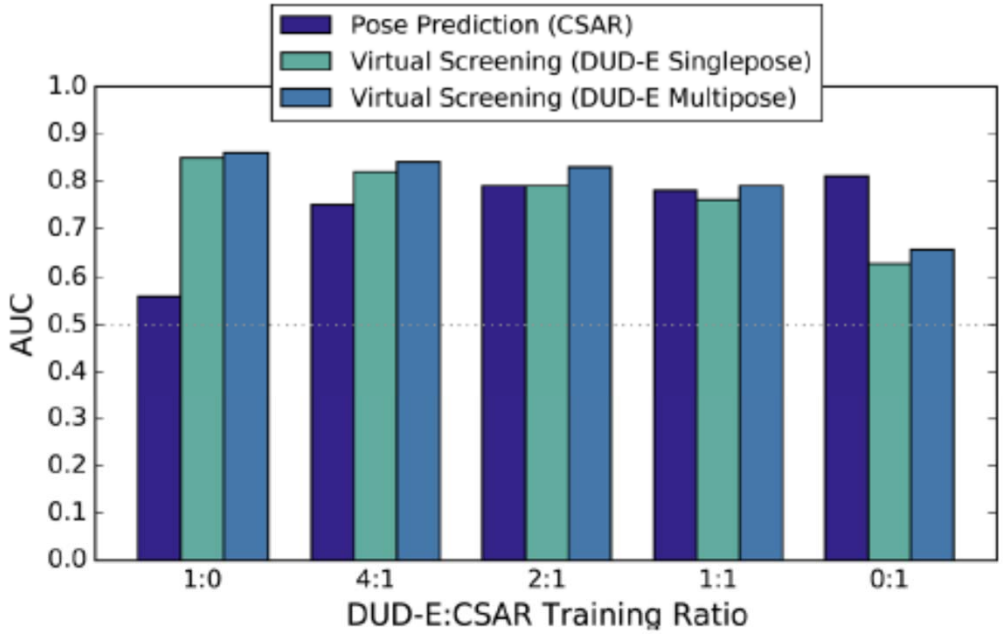
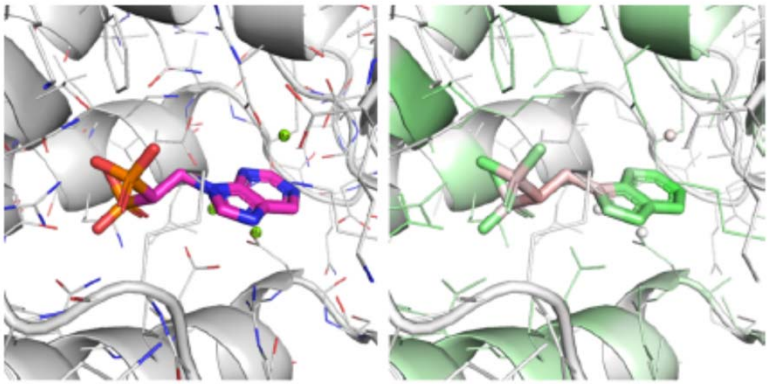


Figure 12. Cross-validation performance of the CNN model when trained with different ratios of CSAR and DUD-E data and evaluated in terms of pose prediction (CSAR) and virtual screening (DUD-E).



(a) (b)

Figure 13. Top ranked pose by Vina of the CHEMBL457424 ligand of the fpps DUD-E target. Visualization of a CNN model trained using only DUD-E training data. The pose is scored highly due to the polar parts of the structure regardless of the orientation of the ligand.

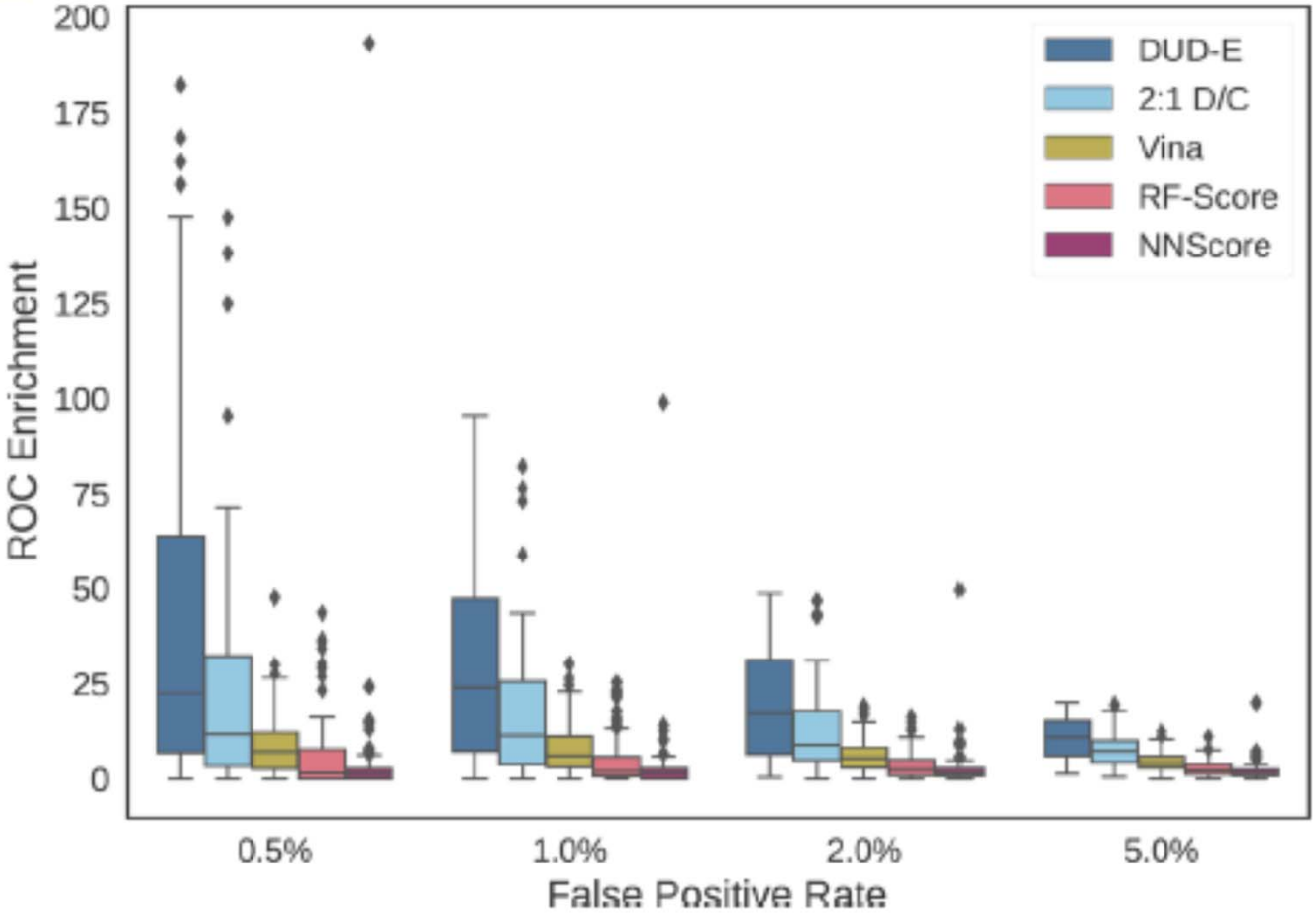


Figure 10. Distribution of ROC enrichment of at different false positive rates for CNN models compared to Vina, RF-Score, and NNScore scoring functions on the DUD-E data set.

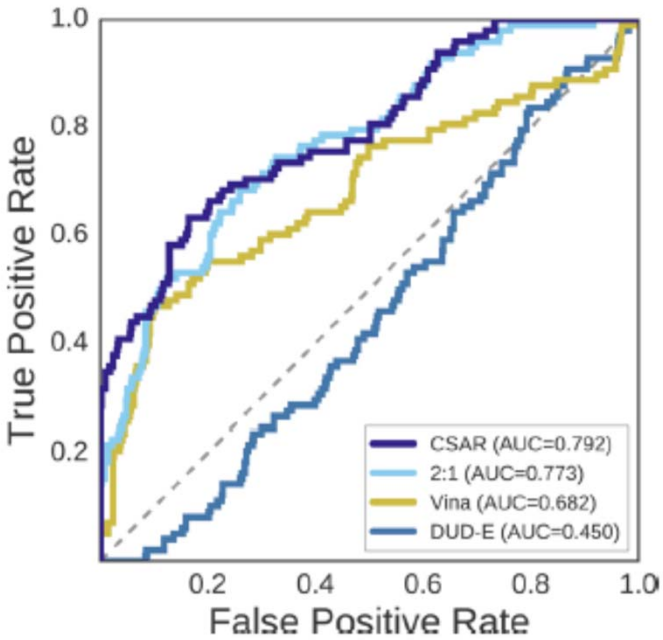


Figure 14. ROC plot for discriminating low RMSD from high RMSD poses generated from the PDBbind core set. The CSAR-trained CNN performs best at classifying generated poses as low or high RMSD across targets, with a steep initial slope evincing good performance at early recognition.

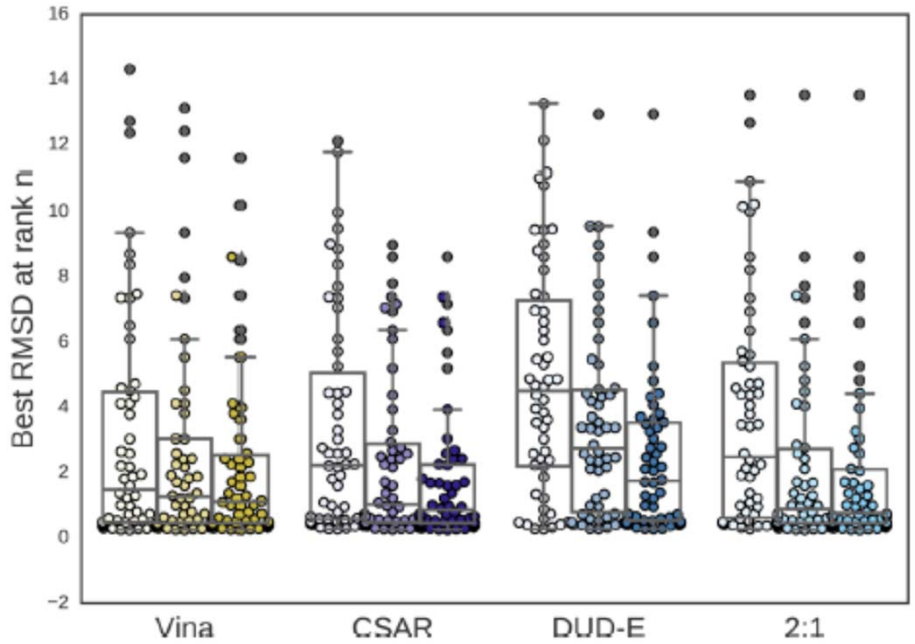


Figure 15. Boxplots of the best RMSD seen so far at ranks 1, 3, and 5 (shown from left to right) for all targets in the PDBbind core subset.

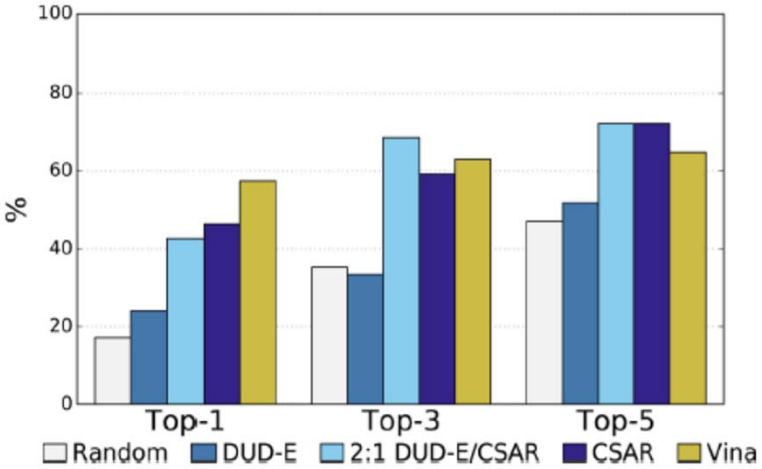


Figure 16. Percentage of complexes with low RMSD poses identified as the top-1, -3, or -5 poses for different scoring methods.

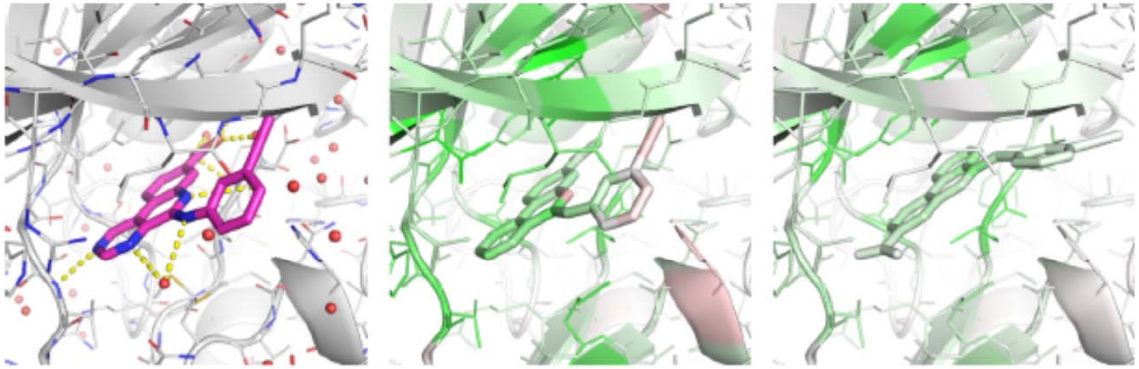


Figure 17. An example, PDB 3PE2, of a complex from the PDBbind core set where Vina correctly top-ranks a low RMSD pose (0.25 Å) and the CNN model does not (5.27 Å). The crystal pose is shown as magenta sticks and the two docked poses are visualized using the CSAR trained CNN model.

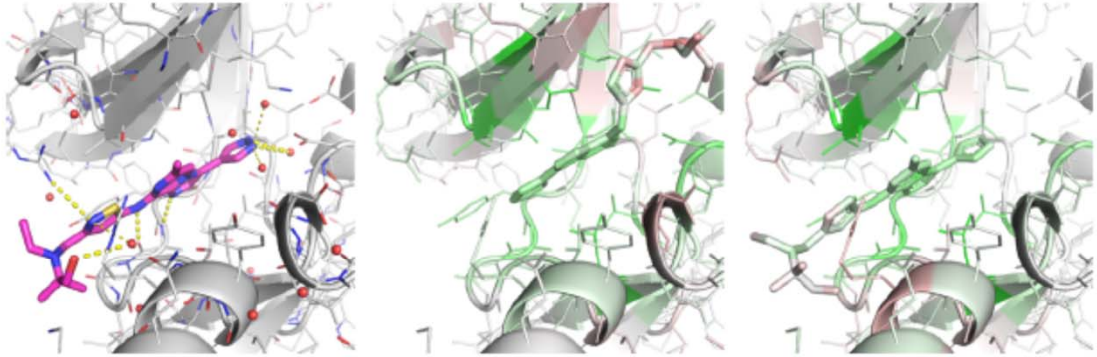


Figure 18. An example, PDB 3MYG, of a complex from the PDBbind core set where the CNN model correctly top-ranks a low RMSD pose (0.96 Å) and Vina does not (12.71 Å). The crystal pose is shown as magenta sticks, and the two docked poses are visualized using the CSAR trained CNN model.

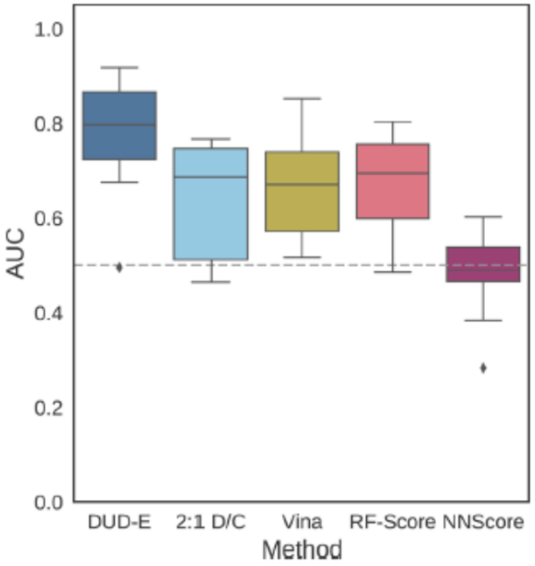


Figure 19. Distribution of the area under the ROC curve for targets of the ChEMBL data set for the pose-insensitive CNN model trained only on DUD-E, the pose-sensitive DUD-E/CSAR 2:1 model, Vina, RF-Score, and NNScore.

Table 2. Mean AUC and ROC Enrichment (RE) Across Targets in the ChEMBL Dataset for CNN Models, Vina, RF-Score, and NNScore

metric	DUD-E	2:1 D/C	Vina	RF-Score	NNScore
AUC	**0.779**	0.642	0.665	0.673	0.484
0.5% RE	**40.720**	7.579	9.579	16.005	1.474
1.0% RE	**25.506**	6.291	7.719	10.695	1.733
2.0% RE	**15.575**	4.756	5.503	7.300	1.282
5.0% RE	**8.303**	3.575	4.388	4.380	1.045

Table 3. Mean AUC and ROC Enrichment (RE) Across Targets in the MUV Dataset for CNN Models, Vina, RF-Score, and NNScore

metric	DUD-E	2:1 D/C	Vina	RF-Score	NNScore
AUC	0.522	0.499	**0.549**	0.512	0.441
0.5% RE	**1.481**	0.741	0.000	0.000	0.000
1.0% RE	**1.481**	1.111	1.111	**1.481**	0.370
2.0% RE	1.296	1.111	**1.852**	0.926	0.370
5.0% RE	**1.556**	0.593	1.333	1.053	0.667

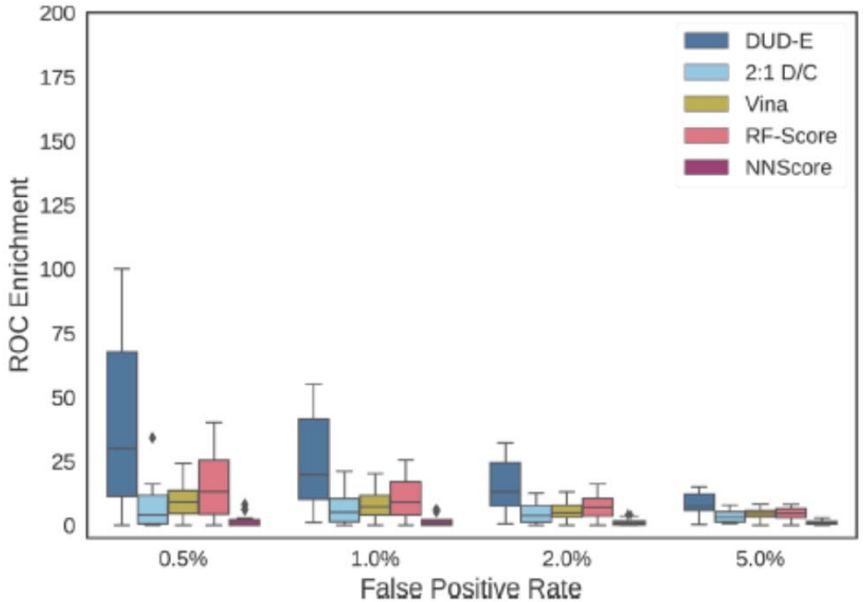


Figure 20. Distribution of ROC enrichment of ChEMBL targets at different false positive rates for CNN models compared to Vina, RF-Score, and NNScore scoring functions.

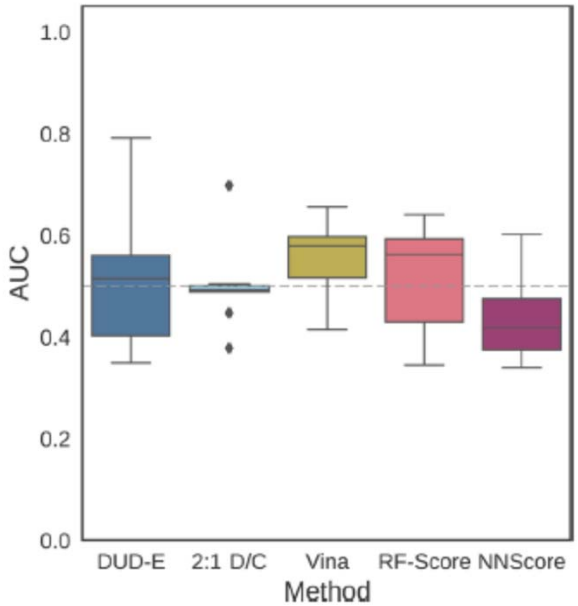


Figure 21. Distribution of the area under the ROC curve for targets of the MUV data set for the pose-insensitive CNN model trained only on DUD-E, the pose-sensitive DUD-E/CSAR 2:1 model, Vina, RF-Score, and NNScore.

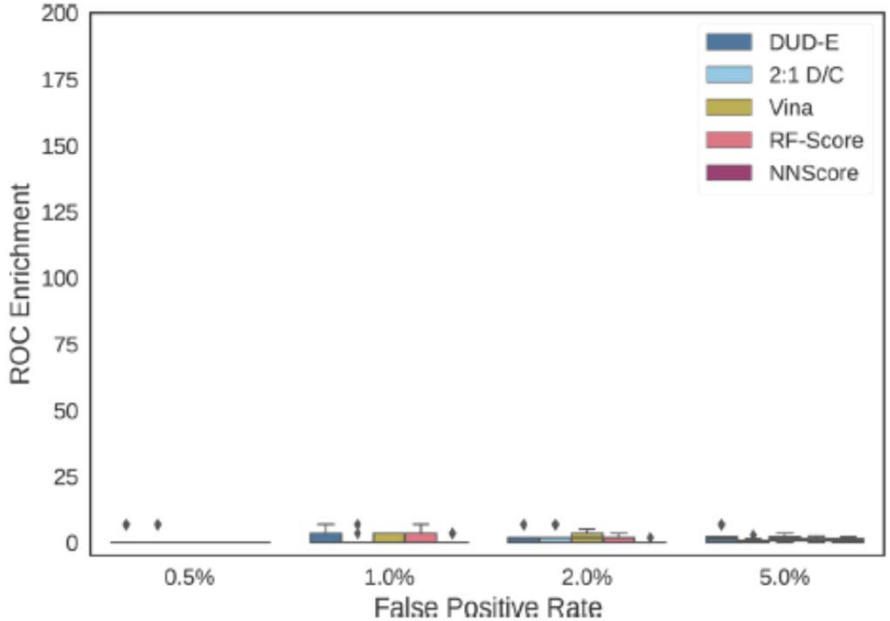


Figure 22. Distribution of ROC enrichment across MUV targets at different false positive rates for CNN models compared to Vina, RF-Score, and NNScore scoring functions.

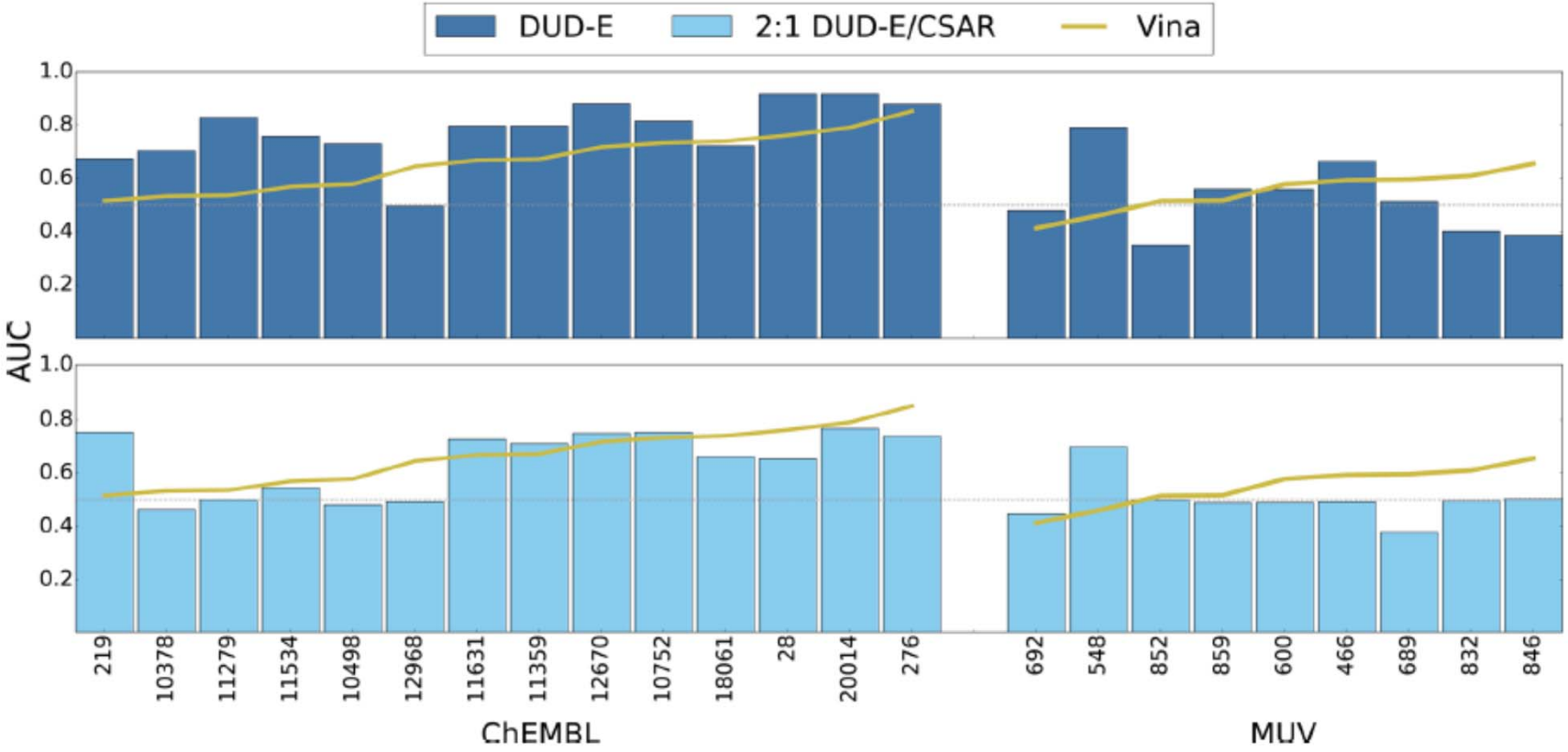
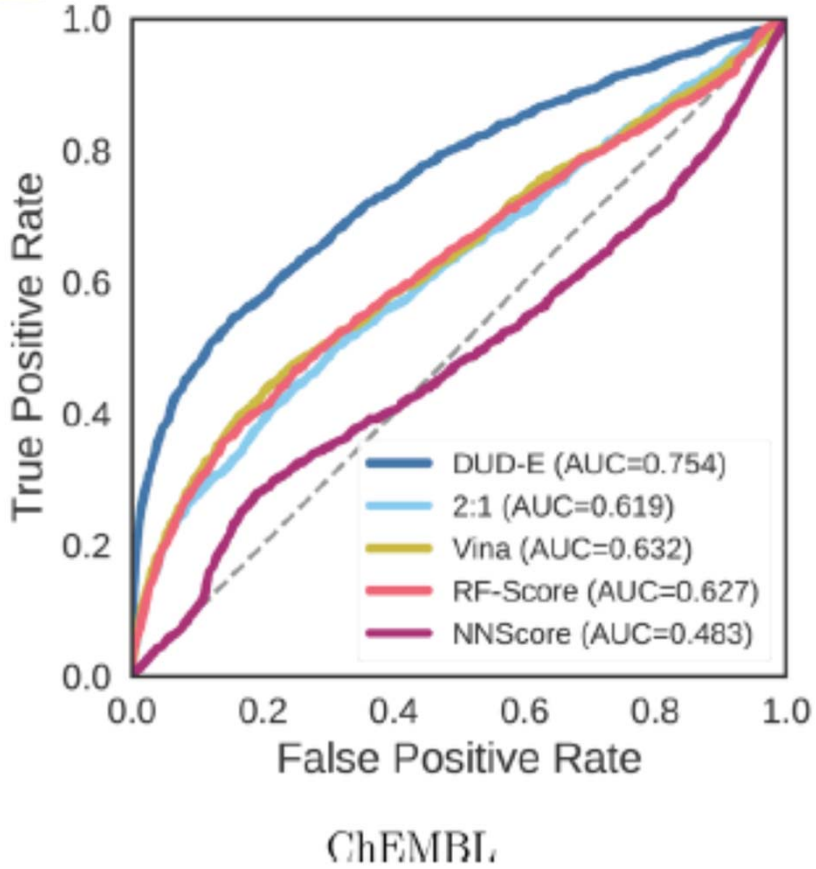


Figure 23. Performance of CNN models on ChEMBL and MUV screening benchmarks compared to the Vina scoring function. Targets are sorted by performance with Vina. Identical sets of docked poses were ranked. The score of the top ranked pose of each ligand is used to predict activity (multipose scoring). Consistent with the cross-validation results (Figure 11), a CNN model trained only on DUD-E training data performs best, outperforming Vina in 86% of the ChEMBL targets and 56% of the MUV targets. Models trained using a mix of DUD-E and CSAR data performed less well compared to Vina, achieving better AUCs than Vina in 36% of the ChEMBL targets and 22% of the MUV targets.



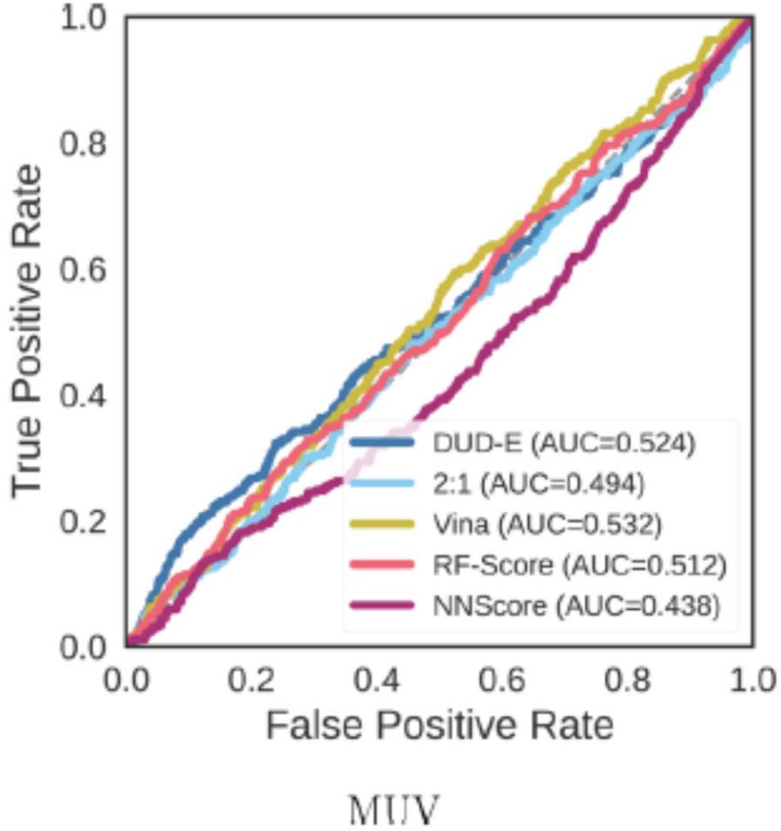


Figure 24. Overall virtual screening performance represented as a combined ROC curve for two CNN models trained on their full training sets and tested on the ChEMBL and MUV independent test sets and compared to Vina, RF-Score, and NNScore.

Table 4. Virtual Screening Performance for Sphingosine 1-Phosphate Receptor EDG-1 (PDB 3V2Y) with Different Choices of Active and Decoy Sets[a]

actives	decoys	Vina	DUD-E	2:1
MUV	MUV	0.593	0.663	0.492
MUV	ChEMBL	0.619	0.682	0.523
ChEMBL	ChEMBL	0.668	0.796	0.727
ChEMBL	MUV	0.667	0.793	0.696

[a]The active compounds were identified in different screens (biochemical for ChEMBL, cell-based for MUV), and the method used to construct the decoy sets is also different.

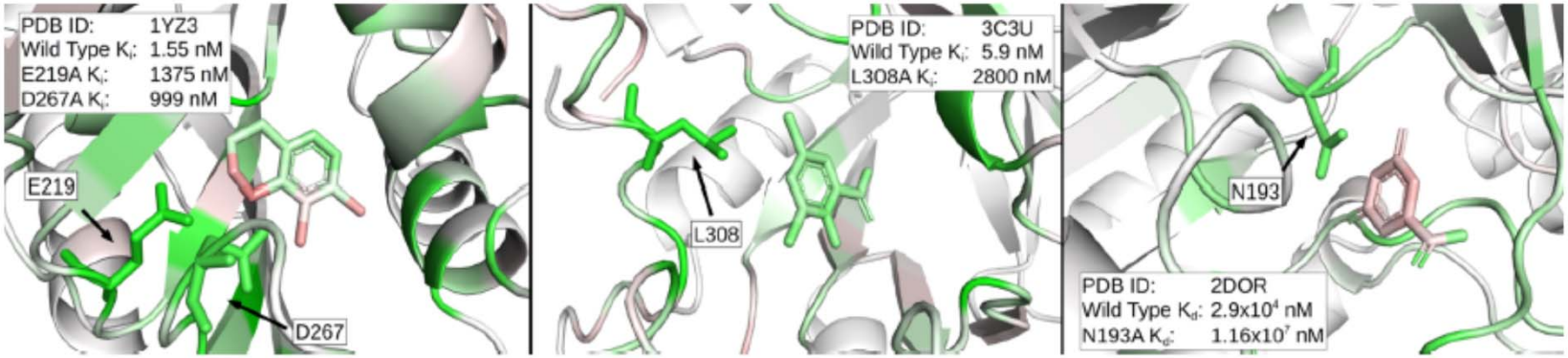
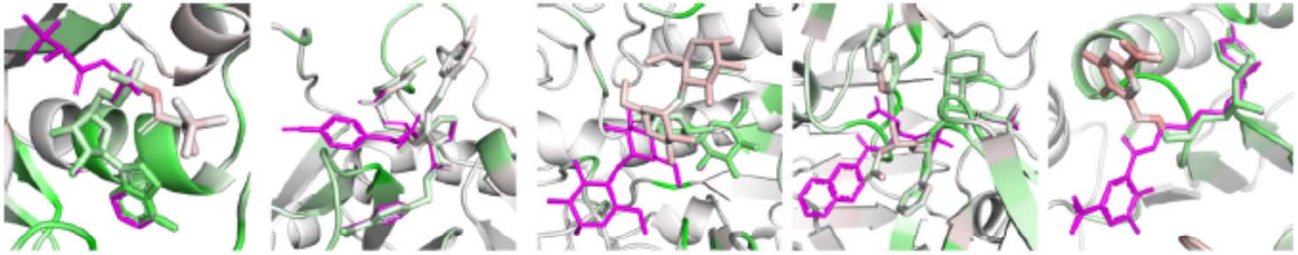


Figure 25. Visualizations of protein–ligand complexes with binding affinity data for point mutations in the protein. The top three most significant changes in binding affinity from the Platinum database are shown from left to right. Any residue that was mutated experimentally is shown in stick form, while the rest of the protein is shown as a cartoon. In all three cases, the green coloring supports the experimental results that the residues in question are important for ligand binding. Visualization is performed using the 2:1 DUD-E/CSAR model.



3COY 3UTU 2QMJ 3PWW 3OZT

Figure 26. Visualizations of partially aligned docked poses from the PDBbind core set. The crystal pose is shown as magenta sticks, and the docked pose and receptor are colored according to our visualization algorithm and the 2:1 DUD-E/CSAR model. None of these protein targets were included in training. The visualization highlights that the model assesses the part of the pose aligned to the crystal ligand as more favorable than the differing part.