Convolutional Neural Networks

Jie Tang

Tsinghua University

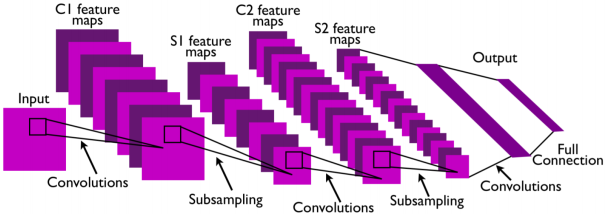
April 3, 2019

Overview

Outline

Introduction

- Convolutional Networks are a specialized kind of Feedforward Neural Networks
  - Neuron Science: connectivity pattern between its neurons is inspired by the organization of the animal visual cortex
  - Computer Science: matrix multiplication is replaced with convolution
  - Optimization: still use the same objective function: maximum likelihood or minimum square error and the same solving algorithm: back-propagation.



convolution with different kernels

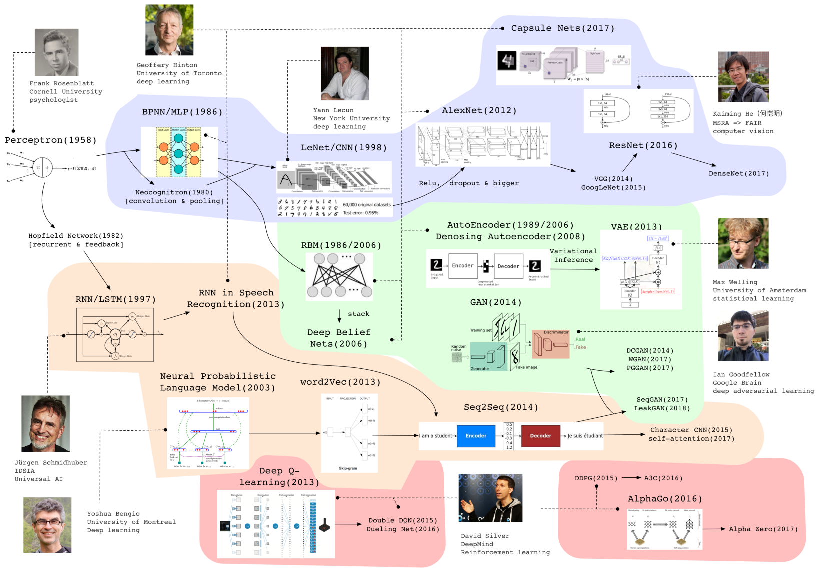
Introduction

- Convolutional Networks are a specialized kind of Feedforward Neural Networks
  - CNN is also known as shift invariant or space invariant artificial neural network (SIANN)
  - wide applications in image and video recognition, recommendation systems, natural language processing, and recently network/graph data
- It might be hard to say who actually invented CNN
  - but, clearly, LeNet-5 is a pioneering 7-layer convolutional neural network[1]
  - Yann LeCun is also viewed as the founder of convolutional nets (CNN)
  - The Turing Triangle: Geoffrey Hinton, Yann LeCun, Yoshua Bengio[2],[3]

[1] LeCun, Yann; Lon Bottou; Yoshua Bengio; Patrick Haffner (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE. 86 (11): 2278–2324.

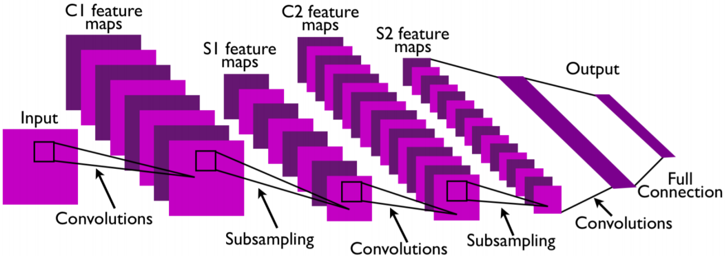
[2] Deep Learning. Yann LeCun, Yoshua Bengio & Geoffrey Hinton. Nature 521, 436444 (28 May 2015)

[3] CHRONOLOGICAL LISTING OF A.M. TURING AWARD WINNERS https://amturing.acm.org/byyear.cfm


Frank Rosenblatt
Cornell University
psychologist
Geoffery Hinton
University of Toronto
deep learning
Yann Lecun
New York University
deep learning
Capsule Nets(2017)
Kaiming He (何恺明)
MSRA => FAIR
computer vision
Perceptron(1958)
BPNN/MLP(1986)
AlexNet(2012)
LeNet/CNN(1998)
ResNet(2016)
Neocognitron(1980)
[convolution & pooling]
60,000 original datasets
Test error: 0.95%
Relu, dropout & bigger
VGG(2014)
GoogLeNet(2015)
DenseNet(2017)
Hopfield Network(1982)
[recurrent & feedback]
AutoEncoder(1989/2006)
Denosing Autoencoder(2008)
VAE(2013)
Variational
Inference
RBM(1986/2006)
Encoder
Decoder
Max Welling
University of Amsterdam
statistical learning
RNN/LSTM(1997)
RNN in Speech
Recognition(2013)
stack
Deep Belief
Nets(2006)
GAN(2014)
Training set
Random noise
Discriminator
Real
Fake
Generator
Fake image
DCGAN(2014)
WGAN(2017)
PGGAN(2017)
Ian Goodfellow
Google Brain
deep adversarial learning
Neural Probabilistic
Language Model(2003)
word2Vec(2013)
Skip-gram
Seq2Seq(2014)
I am a student
Encoder
Decoder
Je suis étudiant
SeqGAN(2017)
LeakGAN(2018)
Character CNN(2015)
self-attention(2017)
Jürgen Schmidhuber
IDSIA
Universal AI
Deep Q-
learning(2013)
Double DQN(2015)
Dueling Net(2016)
David Silver
DeepMind
Reinforcement learning
DDPG(2015)
A3C(2016)
AlphaGo(2016)
Alpha Zero(2017)
Yoshua Bengio
University of Montreal
Deep learning

Why Convolutional Neural Networks

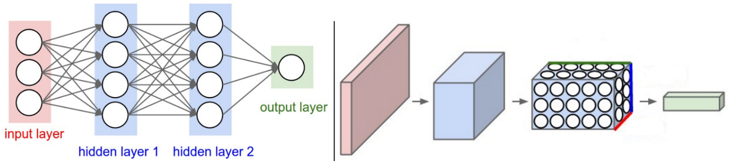
- Example: CIFAR-10 classification— The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class.
    - Scalability: In deep feedforward networks, each latent unit is fully connected to all neurons in the previous layer and thus cannot not scale well to higher resolution images, e.g., an image of $200 \times 200 \times 3$ would result in neurons that have $200 * 200 * 3 = 120,000$ weights each.



convolution with different kernels

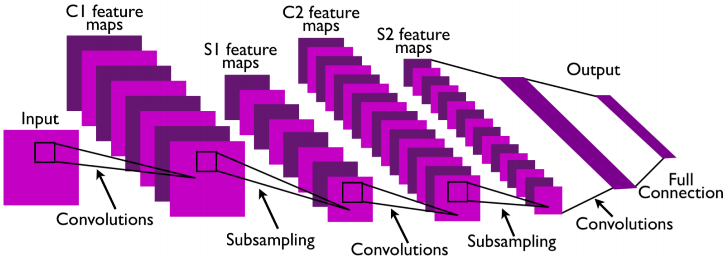
Why Convolutional Neural Networks

- Example: CIFAR-10 classification— The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class.
  - Scalability: In deep feedforward networks, each latent unit is fully...
  - 3D volumes of neurons: unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. In this way, every layer of a CNN transforms the 3D input volume to the 3D output volume.



A regular 3-layer Neural Network vs. A Convolutional Neural Network

What is Convolutional Neural Network?



- convolutional layers, pooling layers, fully-connected layers.
- Goal: approximate some map function f^* (like deep forward networks)
  – e.g., a classifier $y = f^*(x;\theta)$ to map an input x into a category y
  – and learns parameters $\theta = (W'; b')$ that result in the best approximation.

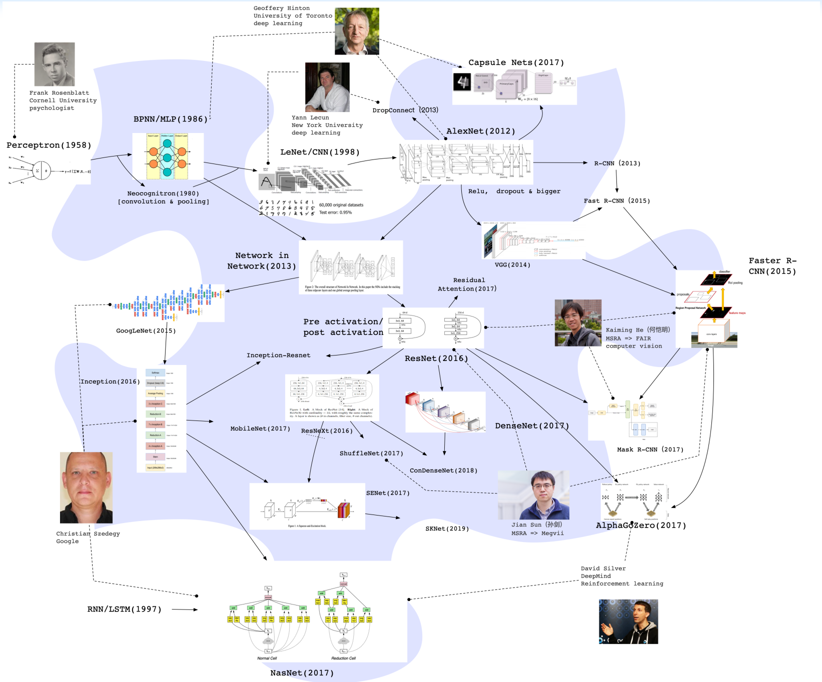
What is CNN: CIFAR-10 example

- A simple CNN for CIFAR-10 classification could have the architecture [INPUT→CONV→RELU→POOL→FC]:
  - INPUT is the raw pixel values of the image of $[32 \times 32 \times 3] - --$width:32, height:32, and three color channels: R, G, B.
  - CONV layer uses convolution to compute the output of "neurons", with each unit computing a dot product between their weights for a small region in the input volume. This results in volume such as $[32 \times 32 \times 12]$ if we decided to use 12 convolution kernels.
  - RELU layer applies an activation function, such as the $max(0, z)$. This leaves the size of volume unchanged ($[32 \times 32 \times 12]$).
  - POOL layer performs a subsampling operation along the spatial dimensions (width, height), resulting in volume such as $[16 \times 16 \times 12]$.
  - FC (i.e. fully-connected) layer computes the class scores, resulting volume of size $[1 \times 1 \times 10]$.

What is CNN

In summary:

- A CNN architecture is in the simplest case a list of Layers that transform the image volume into an output volume (e.g. holding the class scores)
- There are a few distinct types of Layers (e.g. CONV/FC/RELU/POOL are by far the most popular)
- Each Layer accepts an input 3D volume and transforms it to an output 3D volume through a differentiable function
- Each Layer may or may not have parameters (e.g. CONV/FC do, RELU/POOL dont)
- Each Layer may or may not have additional hyperparameters (e.g. CONV/FC/POOL do, RELU doesnt)


Geoffery Hinton
University of Toronto
deep learning
Frank Rosenblatt
Cornell University
psychologist
Capsule Nets(2017)
Yann Lecun
New York University
deep learning
DropConnect (2013)
BPNN/MLP(1986)
Perceptron(1958)
AlexNet(2012)
LeNet/CNN(1998)
R-CNN (2013)
Neocognitron(1980)
[convolution & pooling]
60,000 original datasets
Test error: 0.95%
Relu, dropout & bigger
Fast R-CNN (2015)
Network in
Network(2013)
VGG(2014)
Faster R-CNN(2015)
Residual
Attention(2017)
GoogLeNet(2015)
Pre activation/
post activation
Kaiming He (何恺明)
MSRA => FAIR
computer vision
Inception-Resnet
ResNet(2016)
Inception(2016)
DenseNet(2017)
MobileNet(2017)
ResNeXt(2016)
Mask R-CNN (2017)
ShuffleNet(2017)
ConDenseNet(2018)
SENet(2017)
Christian Szedegy
Google
SKNet(2019)
Jian Sun (孙剑)
MSRA => Megvii
AlphaGoZero(2017)
David Silver
DeepMind
Reinforcement learning
RNN/LSTM(1997)
Normal Cell
Reduction Cell
NasNet(2017)

Outline

Overview

What is Convolution?

- In mathematics, convolution is an operation on two functions (f and g),

$$\begin{aligned}(f * g)(t) &= \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau\end{aligned} \tag{1}$$

- For functions f and g defined on the set Z of integers, we can define the discrete convolution of f and g

$$\begin{aligned}(f * g)(n) &= \sum_{m=-\infty}^{\infty} f[m]g[n-m] \\ &= \sum_{-\infty}^{\infty} f[n-m]g[m]\end{aligned} \tag{2}$$

What is Convolution?

- Suppose we are estimating the location with a GPS. $f[n]$ is the value from GPS, but possibly with noise.
- To obtain a less noisy estimation, we can use recent values of the GPS to smooth estimation with a weighting function $g[m]$, but of course recent value should have a higher weight

$$(f * g)(n) = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \tag{3}$$

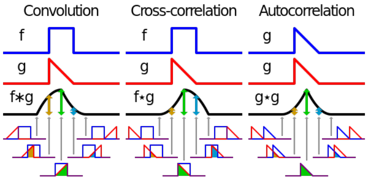
What is Convolution?

- In ML, the input usually forms a multi-dimensional structure. For example, convolution is used to transform a 2D input I with a 2D kernel K:

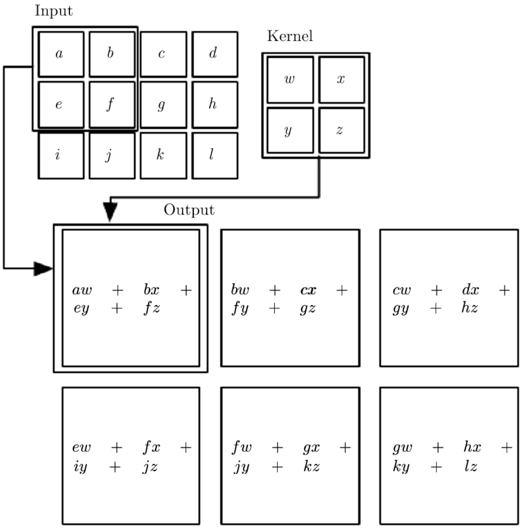
$$Z(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

- In ML, learning algorithm based on convolution with kernel flipping will learn a kernel that is flipped relative to the kernel learned without flipping.
- So, many ML libraries implement cross-correlation but call it convolution:

$$Z(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n)$$

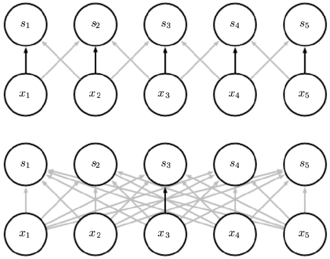


2D Convolution

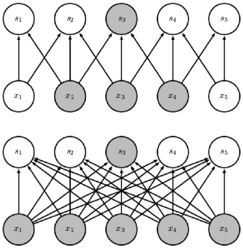


Advantages of CNN

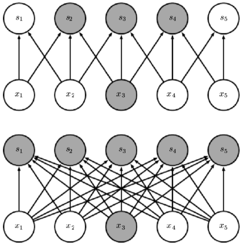
- CNN leverages three important ideas
  - Sparse connectivity: hidden neurons only connected to small regions in the input
  - Parameter sharing: the same parameters are shared across all spatial locations
  - Equivariance: i.e., $f(g(x)) = g(f(x))$



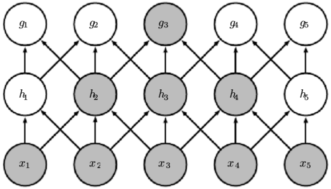
Sparse connectivity vs. Full connection



view from below



view from above



full connection obtained with deep structure

Edge detection by convolution

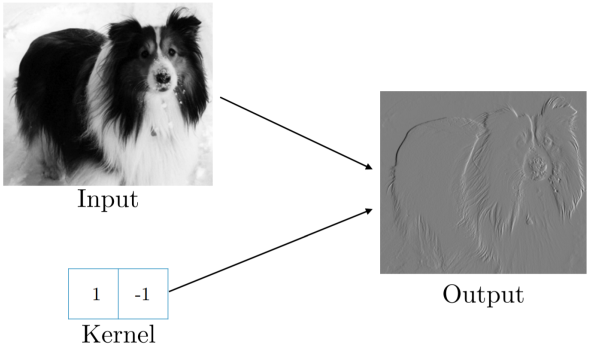
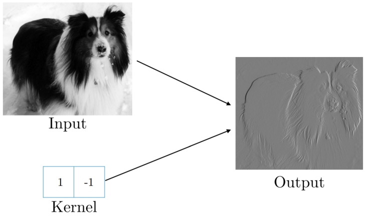


Figure 1: The image on the right was formed by taking each pixel in the original image and subtracting the value of its neighboring pixel on the left. Both images are 280 pixels tall. The input image is 320 pixels wide while the output image is 319 pixels wide. This transformation can be described by a convolution kernel containing two elements, and requires $319 \times 280 \times 3 = 267,960$ floating point operations (two multiplications and one addition per output pixel) to compute using convolution.

Efficiency of convolution

- Input size: 320×280
- Kernel Size: 2×1
- Output Size: 319×280



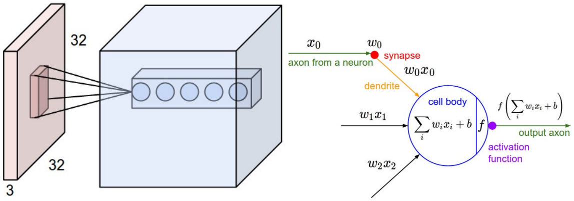
	Convolution	Dense matrix	Sparse matrix
Stored floats	2	319*280*320*280 $>$ 8e9	2*319*280 = 178,640
Float muls or adds	319*280*3 = 267,960	$>$ 16e9	Same as convolution (267,960)

Convolutional Layer

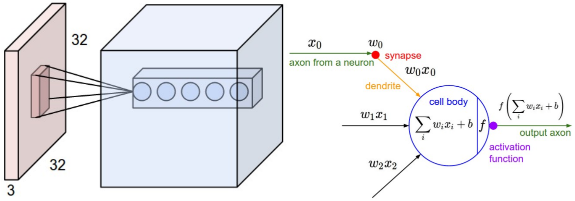
The convolutional layer is the core building block of a CNN that does most of the computational heavy lifting.

Local Connectivity.

Each neuron in CNN only connects to a local region of the input volume. The local region is called the receptive field of the neuron. The extent of the connectivity along the depth axis is always equal to the depth of the input volume.



Convolutional Layer (cont.)

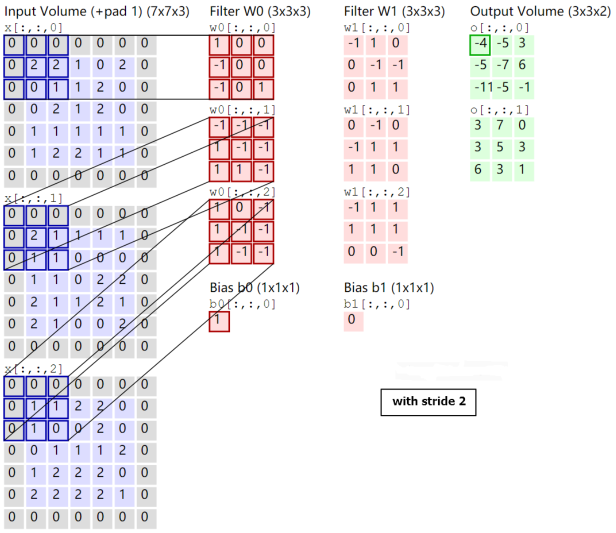


- Example 1. Suppose that the input volume has size [32x32x3]. If the receptive field (or the filter size) is 5x5, then each neuron in the Conv Layer will have weights to a [5x5x3] region in the input volume. Notice that the extent of the connectivity along the depth axis must be 3, since this is the depth of the input volume.
- Example 2. Similarly for the input [16x16x20] with the receptive field size of 3x3, every neuron in the Conv Layer would now have a total of 3*3*20 = 180 connections to the input volume. Again, the connectivity is local in space (e.g. 3x3), but full along the input depth (20).

Spatial arrangement

- In the convolution layer, three hyperparameters control the size of the output volume: **depth**, **stride** and **zero-padding**:
    - the **depth** of the output volume is a hyperparameter: it corresponds to the number of kernels we would like to use;
    - the **stride** is which we slide the kernel. When the stride is 1 then we move the kernels one grid at a time. When the stride is 2 then the kernels jump 2 grids at a time. This will produce smaller output volumes;
    - sometimes it will be convenient to pad the input volume with zeros around the border. The nice feature of **zero-padding** is that it will allow us to control the spatial size of the output volumes (One special case is when enough zero-padding is added to keep the size of the output equal to the size of the input).

Convolution



x[:,:,0]

0	0	0	0	0	0	0
0	2	2	1	0	2	0
0	0	1	1	2	0	0
0	0	2	1	2	0	0
0	1	1	1	1	1	0
0	1	2	2	1	1	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	2	1	1	1	1	0
0	1	1	0	0	0	0
0	1	1	0	2	2	0
0	2	1	1	2	1	0
0	2	1	0	0	2	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	1	1	2	2	0	0
0	1	0	0	2	0	0
0	0	1	1	1	2	0
0	1	2	2	2	0	0
0	2	2	2	2	1	0
0	0	0	0	0	0	0

w0[:,:,0]

1	0	0
-1	0	0
-1	0	1

w0[:,:,1]

-1	-1	-1
1	-1	-1
1	1	-1

w0[:,:,2]

1	0	-1
1	-1	-1
1	-1	-1

b0[:,:,0]

1

w1[:,:,0]

-1	1	0
0	-1	-1
0	1	1

w1[:,:,1]

0	-1	0
-1	1	1
1	1	0

w1[:,:,2]

-1	1	1
1	1	1
0	0	-1

b1[:,:,0]

0

o[:,:,0]

-4	-5	3
-5	-7	6
-11	-5	-1

o[:,:,1]

3	7	0
3	5	3
6	3	1

with stride 2

padding $= 1$
stride $= 2$
#kernel (filter)$=$ 2
spatial extent $=$ 3×3

Convolution

- When working with images, we usually think of the input I and output Z of convolution as being 3D tensors:

$$Z^{(i)}_{j,k} = \sum_{l,m,n} I^{(l)}_{j+m-1,k+n-1} K^{(i,l)}_{m,n} + b^{(i)}$$

  - $Z^{(i)}_{j,k}$ is the value of output unit within channel i at row j and column k
  - kernel k is a 4D tensor with element $K^{(i,l)}_{m,n}$ giving the connection strength between an output unit in channel i and an input unit in channel l, with an offset of m rows and n columns between them.
- We may skip over some positions of the kernel in order to reduce the computational cost:

$$Z^{(i)}_{j,k} = \sum_{l,m,n} I^{(l)}_{(j-1)\times s+m,(k-1)\times s+n} K^{(i,l)}_{m,n} + b^{(i)}$$

  - s is the stride of this downsampled convolution
- Suppose we want to minimize some loss function $J(K, b)$

Backpropagation

- During back-propagation, we will receive a tensor G such that $G_{j,k}^{(i)} = \frac{\partial}{\partial Z_{j,k}^{(i)}} J(K, b)$
- Using chain rule, the derivatives with respect to the kernel can be written as:

$$\frac{\partial}{\partial K_{j,k}^{(i,l)}} J(K, b) = \sum_{m,n} \frac{\partial J(K, b)}{\partial Z_{m,n}^{(i)}} \frac{\partial Z_{m,n}^{(i)}}{\partial K_{j,k}^{(i,l)}} = \sum_{m,n} I_{(m-1)\times s+j,(n-1)\times s+k}^{(l)} G_{m,n}^{(i)}$$

$$\frac{\partial}{\partial b^{(i)}} J(K, b) = \sum_{m,n} \frac{\partial J(K, b)}{\partial Z_{m,n}^{(i)}} \frac{\partial Z_{m,n}^{(i)}}{\partial b^{(i)}} = \sum_{m,n} G_{m,n}^{(i)}$$

- Parameters $K_{j,k}^{(i,l)}$ and $b^{(i)}$ can be updated as follows:

$$K_{j,k}^{(i,l)} = K_{j,k}^{(i,l)} - \alpha \frac{\partial}{\partial K_{j,k}^{(i,l)}} J(K, b)$$

$$b^{(i)} = b^{(i)} - \alpha \frac{\partial}{\partial b^{(i)}} J(K, b)$$

Backpropagation

- The gradient with respect to I for back-propagating the error farther:

$$\frac{\partial}{\partial I^{(i)}_{j,k}} J(K,b) = \sum_{l,m,n} \frac{\partial J(K,b)}{\partial Z^{(l)}_{m,n}} \frac{\partial Z^{(l)}_{m,n}}{\partial I^{(i)}_{j,k}}$$
$$= \sum_{\substack{m,p \\ s.t. \\ (m-1)\times s+p=j}} \sum_{\substack{n,q \\ s.t. \\ (n-1)\times s+q=k}} \sum_{l} K^{(l,i)}_{p,q} G^{(l)}_{m,n}$$

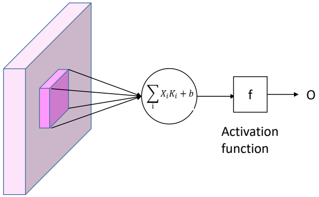
- As we can see, the backward pass for a convolution operation (for both the input and the kernel) is also a convolution (but with spatially-flipped filters).

Non-linearity

Linear functions do not work–
Multi-layer Linear Transformation can still be represented by a single-layered one.

$$O = f(\sum_i X_i K_i + b)$$

- sigmoid function: $f(z) = \frac{1}{1+exp(-z)}$
- hyperbolic tangent: $tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$
- rectified linear function (ReLU): $f(z) = max(0, z)$
- leaky ReLU: $f(z) = 1(z < 0)(\alpha z) + 1(z >= 0)(z)$, where $0 < \alpha << 1$ is a small constant



Pooling

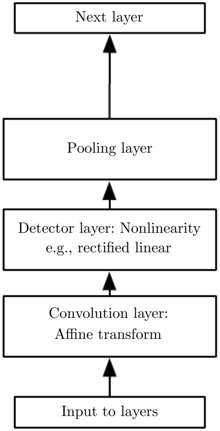
- A convolutional layer consists of several layers (stages)
  - Convolution layer (stage)

$$Z_{j,k}^{(i)} = \sum_{l,m,n} I_{j+m-1,k+n-1}^{(l)} K_{m,n}^{(i,l)}$$

  - Nonlinerity layer (stage)

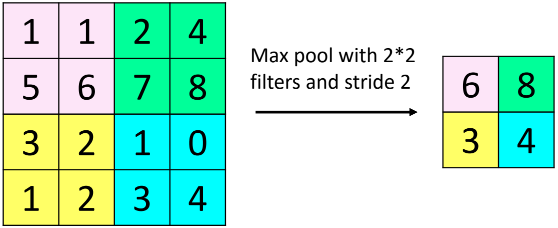
$$O = f(\sum_i X_i K_i + b)$$

  - Pooling layer (stage)



Pooling

- A pooling function replaces the output of the layer at a certain location with a summary statistic of the nearby outputs.
  - Max pooling
  - Average pooling
  - L2-norm pooling
  - Probability weighted pooling



Pooling

Properties of pooling:

- invariance to small translations of the input
- improve statistical efficiency and reduce memory requirements
  - reduce the input of next layer
- handle inputs of variable size

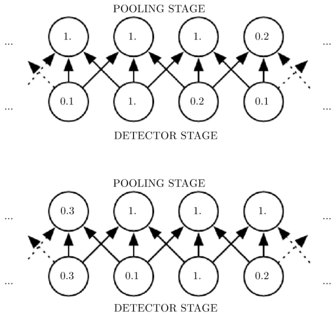
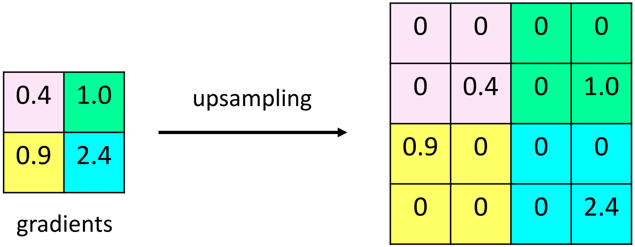


Figure 2: Max pooling introduces invariance.

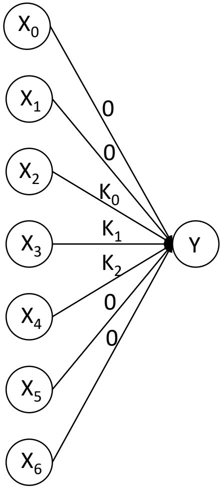
Backpropagation

- The backpropagation of a pooling layer is an upsampling operation that inverses the subsampling in the forward pass.
- E.g. the backward pass for a max pooling layer routes the gradient to the input that had the highest value in the forward pass.



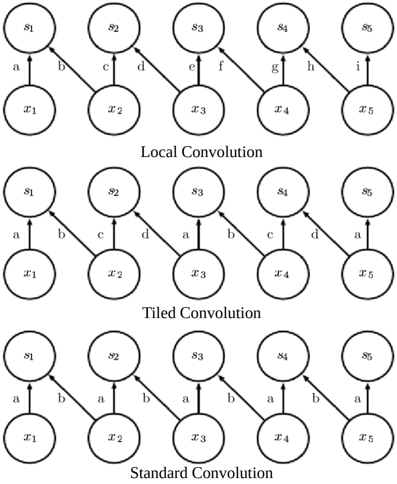
Convolution and pooling as an infinitely strong prior

- A convolutional layer can be viewed as a fully-connected layer with an infinitely strong prior over its weights.
- This prior is that the weights for one hidden neuron must be identical to the weights of its neighbor, but shifted in space.
- In other words, the weights must be zero, except for in the small field assigned to that hidden neuron.
- Likewise, the use of pooling is an infinitely strong prior that each unit should be invariant to small translations.



Other convolution

- local convolution
  - without sharing parameters across locations
- tiled convolution
  - a set of t different kernels is circularly used
  - neighboring units in the output have different parameters
  - after we have gone through t kernels, we cycle back to the first kernel
- standard convolution
  - equal to tiled convolution with $t = 1$



Outline

History of CNNs

History:

- The idea called Multi-Stage Hubel-Wiesel Architectures was rooted in Hubel and Wiesel's classic 1962 work on the cat's primary visual cortex;
- A similar model to be simulated on a computer was Fukushima's Neocognition;
- In 1989, Lecun developed the modern convolutional network by applying back-propagation algorithm to 2D convolution.

Applications:

- In 1996, an operational bank check reading system based on CNNs was developed at AT&T (Lecun). By the late 90's, it was reading over 10% of all the checks in the US;
- In 2009, Google deployed a ConvNet to detect faces and license plate in StreetView images so as to protect privacy;
- Vidient Technologies had developed a ConvNet-based video surveillance system deployed in several airports in the US.

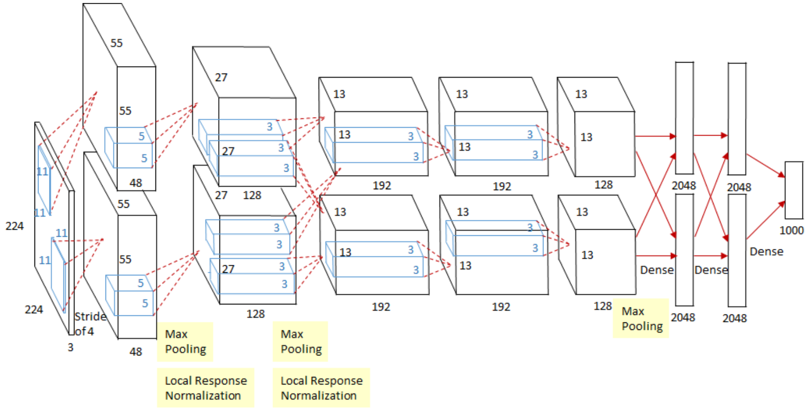
ILSVRC

- ILSVRC (Large Scale Visual Recognition Challenge) is a well-known computer vision competition, held from 2010 to 2017.
- The challenges include classification, localization and object detection.
- The classification task provides more than 1,000,000 labeled training images and 100,000 test images. The competitors need to output the top 5 (out of 1,000) most likely categories for each image. Performance is evaluated by error rate. (Rate of failure to contain the correct category in the output 5 categories.)

Non-CNN methods

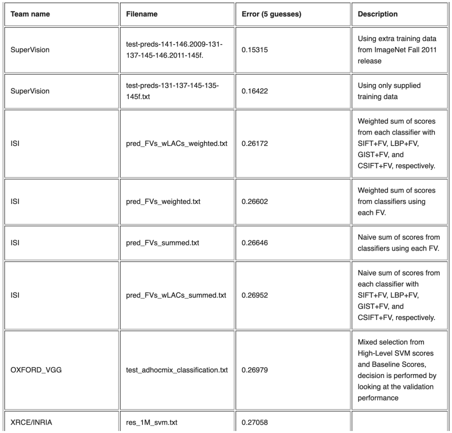
- The ILSVRC competition is introduced by Alex Berg from Stony Brook, Jia Deng from Princeton & Stanford, and Fei-Fei Li from Stanford in 2010. (Also with a naive baseline model of 0.80 error rate.)
- In 2010 and 2011, the high-ranked methods of ILSVRC are using non-CNN methods, like SIFT,LBP, FV, SVM,GIST,CSIFT, stacking, weighted sum, boosting, etc.
  - The winner (classification) of 2010 is Descriptor Coding + SVM from NEC & UIUC, with a 0.28 error rate on classification task.
  - The winner (classification) of 2011 is XRCE, with a 0.26 error rate on classification task.
- However, the human level error rate is 0.051.

AlexNet



- The earliest representative "deep learning" model is the AlexNet (Krizhevsky, Alex. "ImageNet Classification with Deep Convolutional Neural Networks" 2013)
- Alexnet is the first deep CNN method that achieves significant improvement in image classification.

AlexNet



Team name	Filename	Error (5 guesses)	Description
SuperVision	test-preds-141-146.2009-131-137-145-146.2011-145f.	0.15315	Using extra training data from ImageNet Fall 2011 release
SuperVision	test-preds-131-137-145-135-145f.txt	0.16422	Using only supplied training data
ISI	pred_FVs_wLACs_weighted.txt	0.26172	Weighted sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively.
ISI	pred_FVs_weighted.txt	0.26602	Weighted sum of scores from classifiers using each FV.
ISI	pred_FVs_summed.txt	0.26646	Naive sum of scores from classifiers using each FV.
ISI	pred_FVs_wLACs_summed.txt	0.26952	Naive sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively.
OXFORD_VGG	test_adhocmix_classification.txt	0.26979	Mixed selection from High-Level SVM scores and Baseline Scores, decision is performed by looking at the validation performance
XRCE/INRIA	res_1M_svm.txt	0.27058	

- AlexNet uses a deep CNN with 7 hidden layers.
- The top-5 classification error rate of AlexNet is 0.15, which is a significant improvement, comparing with best mixtures of non-CNN methods (0.26).
- This great success innovates the development of CNN.
- The submitted version is an ensemble of 7 AlexNets.
- AlexNet single model achieves 0.17 error rate.

ILSVRC 2012 & 2013

- AlexNet (SuperVision) won ILSVRC 2012 classification task with a 0.15 top-5 error rate.
- The ILSVRC 2013 classification winner, Clarifai, uses an ensembled model of CNNs and achieves 0.11 top-5 error rate.
- Although team Clarifai has only 1 member, Matthew Zeiler in ILSVRC 2013, Clarifai is now a sucessful company focusing on computer vision.

Deeper CNNs — GoogleNet

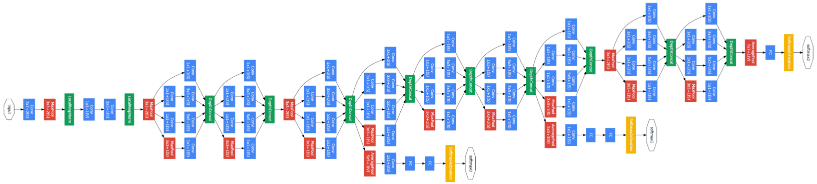


Figure 3: Illustration of GoogleNet.

- The ILSVRC 2014 classification winner is the GoogleNet, a 22-layered CNN which achieves 0.066 error rate.

Deeper CNNs — VGG

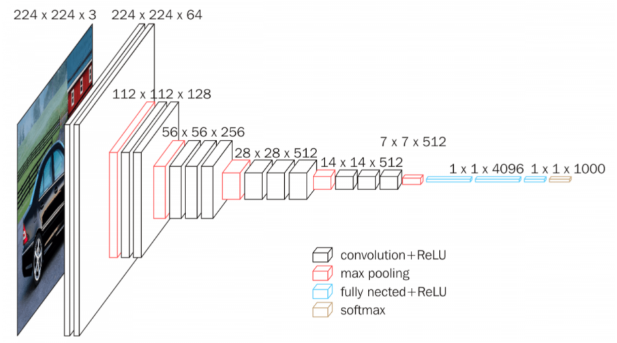
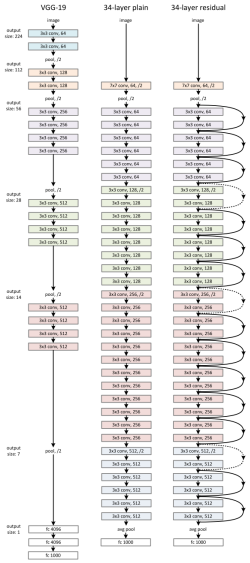


Figure 4: Illustration of VGG16.

- The ILSVRC 2014 classification runner-up is VGG. the combination of 11,13,16,19-layered CNNs achieves 0.073 error rate. A single VGGNet has a 0.084 error rate.

Deeper CNNs — ResNet



- MSRA uses the ensemble of ResNet, and wins most of the competitions ILSVRC 2015. Specifically, on image classification task, the top-5 error rate is 0.036. (As for a 152 layer ResNet single model, the error rate is 0.045).
- Left illustrates ResNet-34 architecture (compared with VGG-19 and CNN-34)
- ResNet uses residual connection, therefore the network can reach as much as 1,202 layers while still gaining performance.

ILSVRC 2016 & 2017

- In ILSVRC 2016, the best classification result comes from Trimps-Soushen (The Third Research Institute of Ministry of Public Security), with an error rate of 0.030. (Using ensemble of multiple ResNet and Inception models.)
- In ILSVRC 2017 (the last ILSVRC), the best classification result comes from WMW (Momenta & Oxford), with an error rate of 0.0225. (Using ensemble of variations of SENet)

Outline

Network in Network

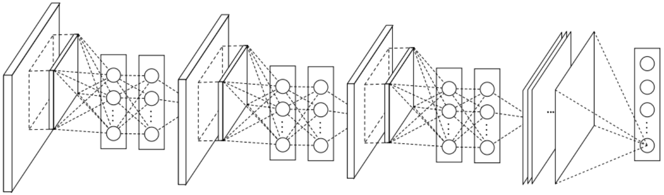
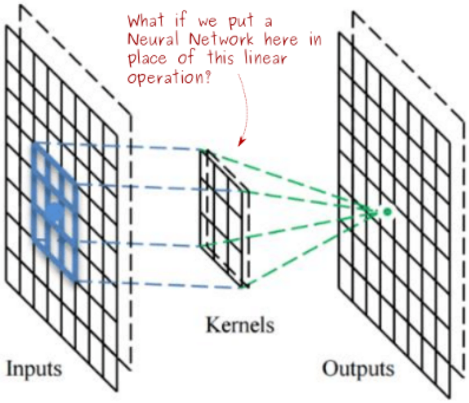


Figure 2: The overall structure of Network In Network. In this paper the NINs include the stacking of three mlpconv layers and one global average pooling layer.

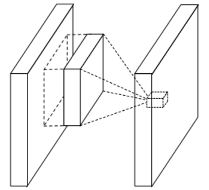
- Network in network (Min Lin, Qiang Chen, Shuicheng Yan, 2013) proposes two techniques that greatly innovate the development of modern CNN architectures.

Network in Network — Tech 1

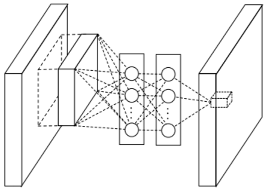

What if we put a Neural Network here in place of this linear operation?
Kernels
Inputs
Outputs

- Instead of a simple matrix multiplication and activation function between two layers, we can put some fancier things between that. (e.g. a fully-connected MLP network).
- MLP network in Convolutional network.

Network in Network — Tech 1 (Cont.)



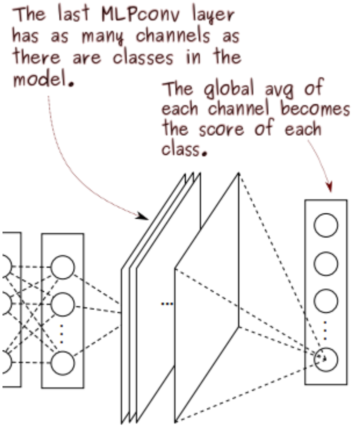
(a) Linear convolution layer



(b) Mlpconv layer

- Local MLP (applies to every single neuron instead of the whole network).
- So an Mlpconv layer can also be viewed as one or more 1x1 convolution layers concatenated with a traditional (e.g., 3x3/5x5) convolution layer.

Network in Network — Tech 2


The last MLPconv layer has as many channels as there are classes in the model.
The global avg of each channel becomes the score of each class.

- Instead of the fully-connected layer in the final classification stage, we can do global average pooling that averages all output neurons to be the output.
- This makes the output of the network robust to position of the object.
- Avoid the overfitting problem of fully-connected layers!

Network in Network

- Review the convolutional layers and classification layers
- Convolution:

$$O = f(\sum_i X_i K_i + b), \qquad (CNN) \tag{4}$$

$$O = f(\sum_i MLP(X_i) K_i + b), \qquad (NIN) \tag{5}$$

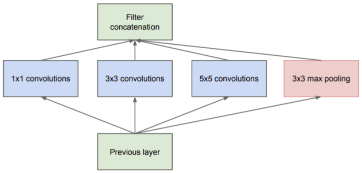
- Classification

$$O = MLP(X_{11}, X_{12}, ..., X_{1m}, X_{21}, ..., X_{nm}), \qquad (CNN) \tag{6}$$

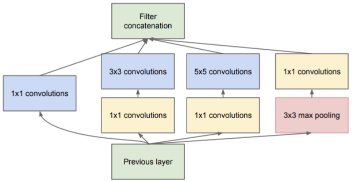
$$O = average(X_1, ...X_n), \qquad (\text{Global Pooling}) \tag{7}$$

- Idea: Add parameter sharing parts (local MLP) to avoid the original overfitting parts (Global MLP)

Inception



(a) Inception module, naïve version

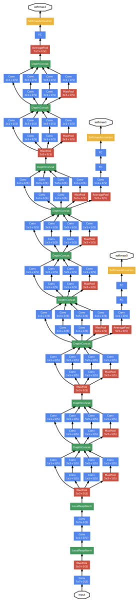


(b) Inception module with dimensionality reduction

Figure 2: Inception module

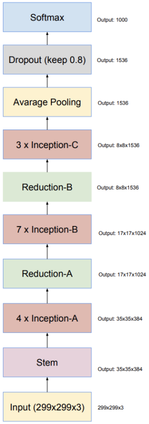
- Inspired by NIN, researchers are trying to find an optimal local sparse structure (Szegedy et al., 2015).
- 1x1,3x3,5x5,pooling are commonly used filters, stacking with an 1x1 is also useful in NIN, so why not use all of them ?
- Inception was introduced. Inception-v1 can be also viewed as GoogleNet (ILSVRC 2014 winner).

Inception

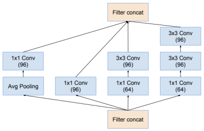


- Idea: Parallel convolution. Instead of layer by layer convolution, the convolution forms various paths in Inception models.
- Single model achieves 0.079 top-5 error rate on ILSVRC, the stack of 7 models achieve 0.067 error rate.

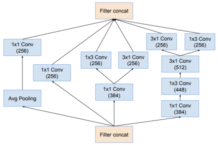
Inception



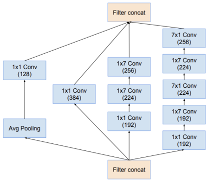
Inception-v4 Model



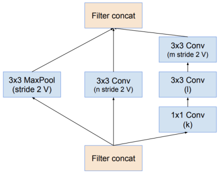
Inception A



Inception C

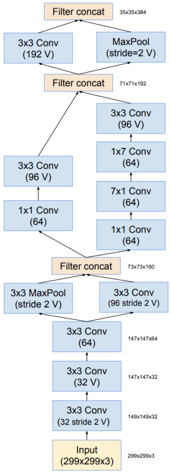


Inception B

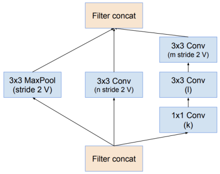


Reduction A

Inception



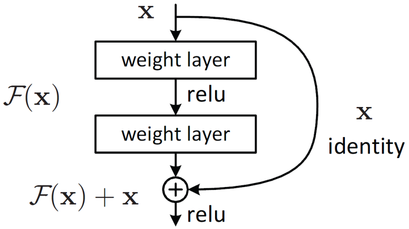
Stem



Reduction B

- Inception grows to be a very complicated serie of deep CNN stacking models (Szegedy et al. 2016).
- Instead of simply combining all filters, better local structures are found by researchers.
- Without Residual connection, pure CNN (single model) (Inception-v4) can reach 0.050 error rate on ILSVRC.

ResNet

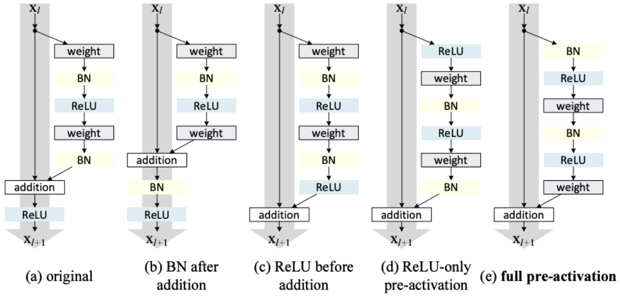


- ResNet (He et al. 2015) employs the idea of residual representation and shortcut connections, and introduced ResNet.

ResNet

Table 2. Classification error (%) on the CIFAR-10 test set using different activation functions.

case	Fig.	ResNet-110	ResNet-164
original Residual Unit [1]	Fig. 4(a)	6.61	5.93
BN after addition	Fig. 4(b)	8.17	6.50
ReLU before addition	Fig. 4(c)	7.84	6.14
ReLU-only pre-activation	Fig. 4(d)	6.71	5.91
full pre-activation	Fig. 4(e)	**6.37**	**5.46**



Comparison between different ResNet activation methods.

ResNet

- Review the convolutional layers
- Convolution:

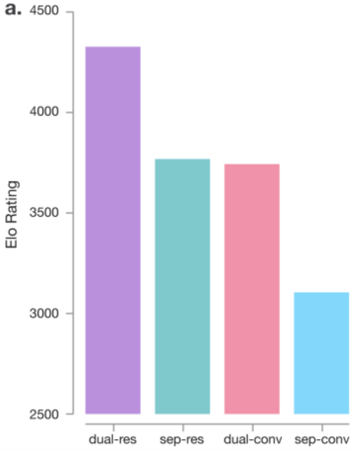
$$O_{i,p} = f(\sum_{j \in N_i} MLP(O_{i,p-1})K_{(j,i),p} + b_p), \ (NIN) \tag{8}$$

$O_{i,p}$ is the p^{th} layer representation of node i.

$$O_{i,p} = f(\sum_{j \in N_i} MLP(O_{i,p-1})K_{(j,i),p} + b_p + O_{i,p-1}), \ (ResNet) \tag{9}$$

$$O_{i,p} = f(\sum_{j \in N_i} MLP(O_{i,p-1})K_{(j,i),p} + b_p) + O_{i,p-1}, \ (ResNet-pre) \tag{10}$$

ResNet

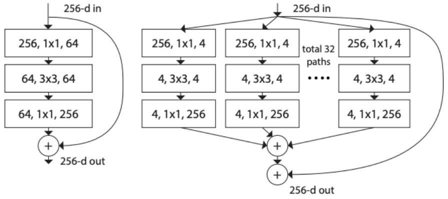


- Despite being revolutionary in computer vision, ResNet has wide applications, not only in computer vision.
- ResNet AlphaGo is ~ 600 ELO better than CNN AlphaGo.
- 600 ELO difference is $\sim 96\%$ win rate. Much Better!

ResNet

- ResNet helps Alphago-Master achieves 60-0 against human professionals, beating Ke Jie and the union team of 5 other top professionals.

ResNeXt



Left: Resnet Block, Right: ResNeXt Block

- ResNeXt (Saining Xie, Ross Girshick, Piotr Dollr, Zhuowen Tu, Kaiming He, 2016) is the runner-up of 2016 ILSVRC.
- Instead of layer by layer convolution, the convolution forms various paths in ResNeXt models.
- Group Convolution reduces calculation complexity/number of parameters while still preserving representational information.

ResNeXt

- Group Convolution: Convolution by groups.
- Convolution:

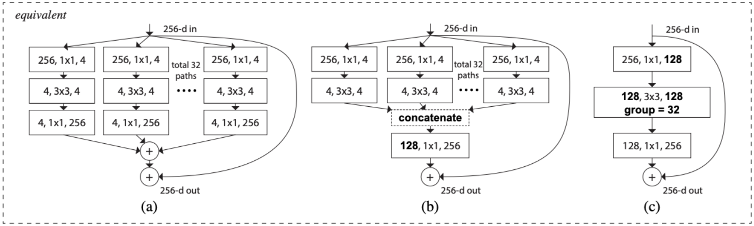
$$O_{i,p} = f(\sum_{j \in N_i} MLP(O_{i,p-1}) K_{(j,i),p} + b_p) + O_{i,p-1}, \ (Resnet) \tag{11}$$

Assume $group = n$, $O_{i,p-1} = concat(O_{i,p-1,1}, ..., O_{i,p-1,n})$ (separation to n groups)

$$O_{i,p,m} = f(\sum_{j \in N_i} MLP(O_{i,p-1,m}) K_{(j,i),p,m} + b_{p,1}) \tag{12}$$

$$O_{i,p} = \mathrm{concat}(O_{i,p,1}, O_{i,p,2}, ..., O_{i,p,n}) + O_{i,p-1}, \ (ResNeXt) \tag{13}$$

ResNeXt



Equivalent ResNeXt structures

- ResNeXt can be viewed as
  - (a).sum of 32-path, 256-dimensional convolution
  - (b).concatenation of 32-path, 4-dimensional convolution
  - (c).group convolution with group=32

  Three views are Equivalent!
- ResNeXt-101 single model has an error rate of 0.053 on ILSVRC. (0.030 when ensembled)

ShuffleNet

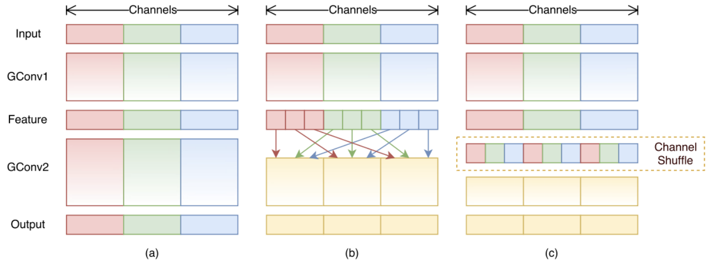


Figure 1. Channel shuffle with two stacked group convolutions. GConv stands for group convolution. a) two stacked convolution layers with the same number of groups. Each output channel only relates to the input channels within the group. No cross talk; b) input and output channels are fully related when GConv2 takes data from different groups after GConv1; c) an equivalent implementation to b) using channel shuffle.

- ShuffleNet (Zhang et al., 2017) adds channel shuffle on top of ResNeXt.

ShuffleNet

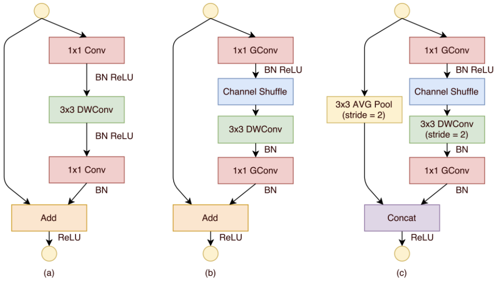
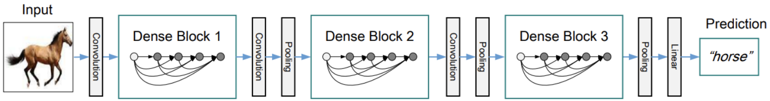


Figure 2. ShuffleNet Units. a) bottleneck unit [9] with depthwise convolution (DWConv) [3, 12]; b) ShuffleNet unit with pointwise group convolution (GConv) and channel shuffle; c) ShuffleNet unit with stride = 2.

- The channel shuffle helps increase the efficiency of group convolution.
- Shufflenet performs better than ResNeXt when restricting the number of parameters/flops to be small.
- This enables real-world application on mobile devices.

DenseNet



- Instead of only connecting with the previous layer, DenseNet collects information from all previous layers inside a dense block.
- The densely connected property makes sense and DenseNet-161 (k=48) single model achieves 0.053 error rate on ILSVRC.

DenseNet

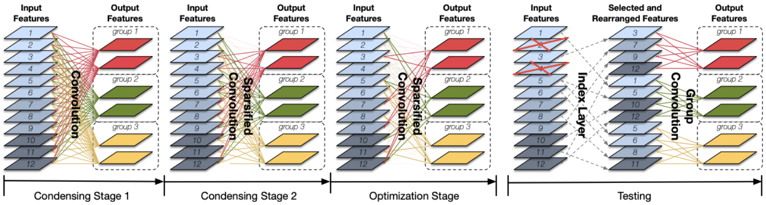
- DenseNet:
- Convolution:

$$O_{i,p} = f(\sum_{j \in N_i} MLP(O_{i,p-1}) K_{(j,i),p} + b_p) + O_{i,p-1}, \ (ResNet) \tag{14}$$

$$O_{i,p} = \mathrm{concat}(O_{i,p-1}, f(\sum_{j \in N_i} MLP(O_{i,p-1}) K_{(j,i),p} + b_p)), \ (DenseNet) \tag{15}$$

- The structure of DenseNet can be viewed as concatenating the feature representation of a new layer to the feature representation of the previous layer.

CondenseNet



- CondenseNet (Liu et al. 2018) is the group convolution version of DenseMet.
- Due to the property of DenseNet (features come from all previous layers, two channels come from different stages of the network), naive group convolution doesn't work.
- Thus need to learn the best way to group.

CondenseNet

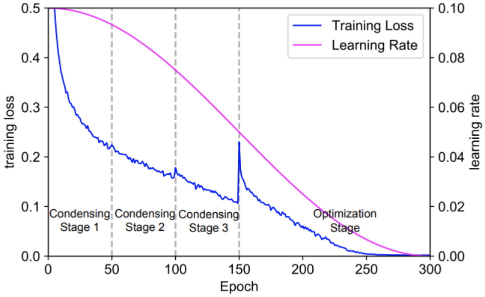


Figure 4. The cosine shape learning rate and a typical training loss curve with a condensation factor of $C = 4$.

- Remove edges with low activation (smallest L1-norm on edge weight) repeatedly during training, until the model size is small enough.
- Finally, forms a group convolution structure for inference.
- CondenseNet performs better than DenseNet and Shufflenet under same number of parameters/flops.

SENet

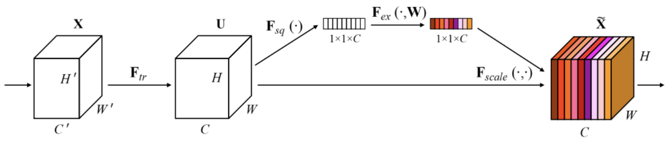


Figure 1: A Squeeze-and-Excitation block.

- The SENet (Squeeze and Excitation Networks) (Jie Hu, Li Shen, Samuel Albanie, Gang Sun, Enhua Wu, 2018) is the last winner of ILSVRC classification task.
- Idea: All channels are created unequal, and shall be assigned different weights.

SENet

- Squeeze-and-Excitation Block: Weight the channels!
- How to weight? Global average pooling
- Given feature representations of the current layer $F : H \times W \times C$, do global average pooling $G : 1 \times 1 \times C$

$$G_k = \frac{1}{HW} \sum_{i,j} F_{i,j,k} \tag{16}$$

- Use an MLP to determine the weight of each channel.

$$W_k = MLP(G_k) \tag{17}$$

$$F'_{i,j,k} = F_{i,j,k} W_k, \forall i, j, k \tag{18}$$

- Use F' as the output of the current layer.

SENet

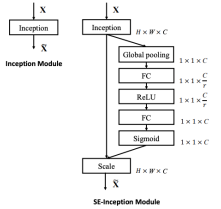


Fig. 2. The schema of the original Inception module (left) and the SE-Inception module (right).

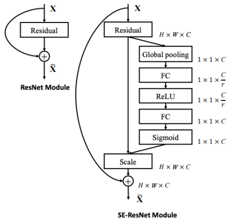


Fig. 3. The schema of the original Residual module (left) and the SE-ResNet module (right).

- The Squeeze-and-Excitation block can be adopted in Inception/Resnet architectures.
- A single SENet-154 model achieves 0.038 top-5(0.174 top-1) error rate on ILSVRC dataset. (When ensembled this goes to 0.0225)

SKnet

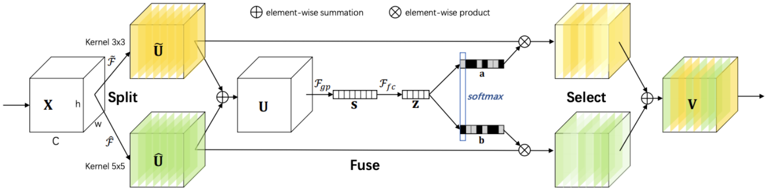
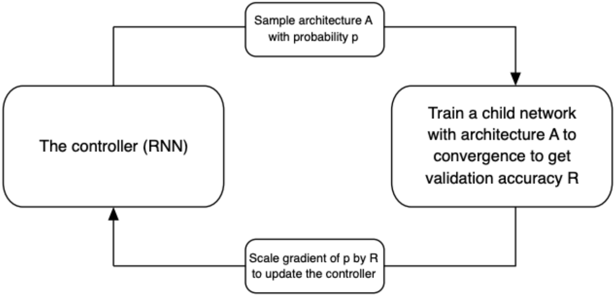


Figure 1. Selective Kernel Convolution.

- SKnet(Selective Kernel network) (Xiang Li, Wenhai Wang, Xiaolin Hu, Jian Yang, 2019) is a recent enhancement of SENet.
- Not only channels, but also kernels are created unequal.
- SKnet adopts different weights on different kernels using attention mechanism. Performs slightly better than SENet.

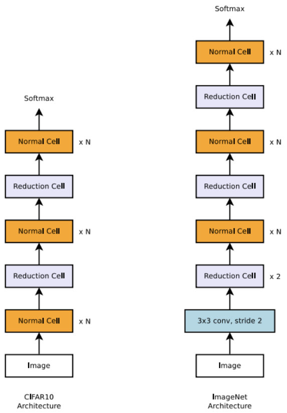
NasNet-AutoML

- CNNs structures are becoming more and more complicated.
- Relies on personalized experiences to find useful CNN architectures.
- Idea: automatically learn CNN architectures. NasNet(Neural Architecture Search Networks)(Zoph et al. 2017)


Sample architecture A with probability p
The controller (RNN)
Train a child network with architecture A to convergence to get validation accuracy R
Scale gradient of p by R to update the controller

The Neural Architecture Search Framework

NasNet-AutoML



NasNet Architecture

- The NasNet searches a best architecture (best Normal/Reduction cell structure) on a small dataset (Cifar-10), and transfers the learned structure to a large dataset (ImageNet).
- The transfer learning process avoids doubts of overfitting to a certain dataset.

NasNet-AutoML

- NasNet Algorithm
  - For each cell, the input hidden state is h_i, h_{i1} from two previous cells.
  - 1.Select a hidden state from h_i, h_{i-1} or from the set of hidden states created in previous blocks.
  - 2.Select a second hidden state from the same options as in 1.
  - 3.Select an operation to apply to the hidden state selected in 1.
  - 4.Select an operation to apply to the hidden state selected in 2.
  - 5. Select a method to combine the outputs of 3 and 4 to create a new hidden state.
- Available Operations

- identity
- 3x3 max pooling
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 1x7 then 7x1 convolution
- 5x5 depthwise-seperable conv
- 7x7 max pooling
- 3x3 convolution
- 3x3 dilated convolution

NasNet-AutoML

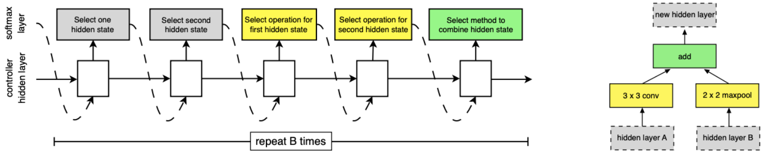
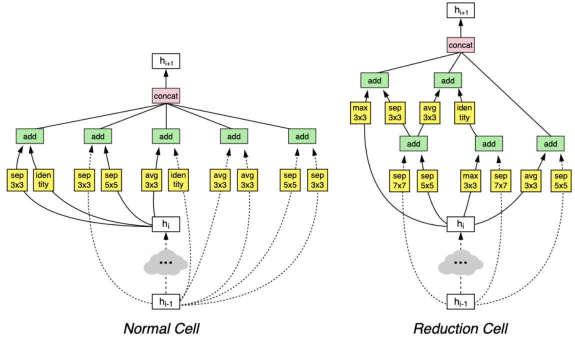


Figure 3. Controller model architecture for recursively constructing one block of a convolutional cell. Each block requires selecting 5 discrete parameters, each of which corresponds to the output of a softmax layer. Example constructed block shown on right. A convolutional cell contains B blocks, hence the controller contains $5B$ softmax layers for predicting the architecture of a convolutional cell. In our experiments, the number of blocks B is 5.

- The NasNet samples and evaluates decisions using the output of the softmax classifiers from a controller RNN.

NasNet-AutoML

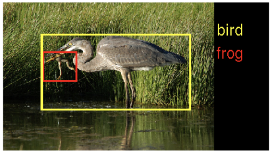


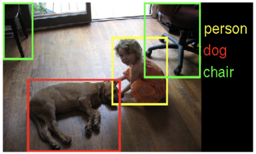
The searched best Normal Cell and Reduction Cell on Cifar-10.

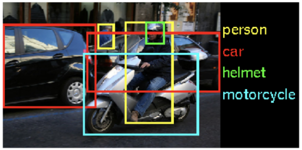
- The NasNet finally gets a searched best structure on cifar-10 for Normal and Reduction cells, illustrated above.
- A single NasNet model gets 0.038 top-5 error rate on ILSVRC classification dataset.

Outline

ILSVRC Detection Task





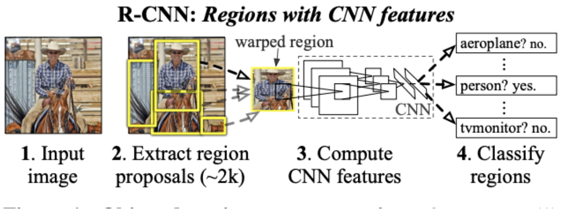


- In year 2013, ILSVRC first introduces the object detection task.
- Participants are required to detect all objects on an image and label them.
- This challenge is much harder than classification.

ILSVRC Detection 2013 & 2014

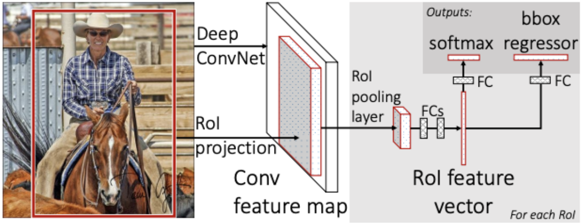
- The winner of 2013,UvA-EuVision achieves MAP=0.226 using non-CNN methods (SIFT+selective search), and use a CNN to compute the prior for the presence of an object in the image.
- The winner of 2014, NUS, combines NIN with kernel regression, SVM and data augmentation techniques, and achieve MAP=0.372.

R-CNN



- Ross Girshick introduced R-CNN (Region Proposal CNN) (Girshick et al., 2013).
- R-CNN uses traditional method to extract region proposals, then use Alexnet to extract features for these region proposals, and use a category-specific SVM to classify whether a region correspond to an item or not.
- MAP=0.314 on ILSVRC detection dataset.

Fast R-CNN



- Fast R-CNN (Ross Girshick, 2015) lets the input image passes through a CNN (VGG-16) first, then examine on region of interests. Thus the image has to pass through the CNN only once instead of once for every region proposal.
- Different from R-CNN, Fast R-CNN is trained end-to-end, and learns the accurate bounding box itself.

Faster R-CNN

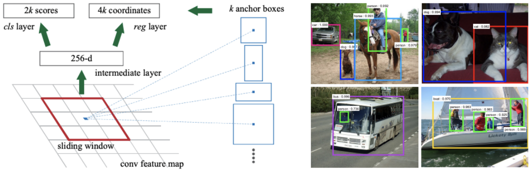
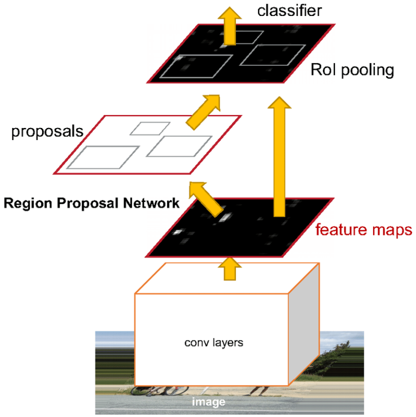


Figure 1: **Left**: Region Proposal Network (RPN). **Right**: Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

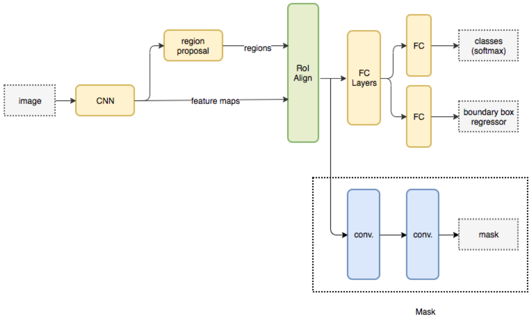
- Faster R-CNN (Ren et al., 2015) is a revolutionary work on object detection, for it takes the idea that the feature representations of CNNs (VGG-16) can not only be used to classify objects, but also be used to learn the RoI(region of interests) much more efficiently and effectively.

Faster R-CNN



- Instead of 2,000 region proposals in R-CNN from traditional methods, faster R-CNN only generates 9 ROIs per image.
- However, faster R-CNN get MAP=0.621 on ILSVRC detection task, and wins the ILSVRC 2015 detection competition.

Mask R-CNN



- Mask R-CNN (He et al., 2017) adds a mask branch besides the classification and bounding box regressor.
- The mask is used to mask out the object (For each RoI we create a mask, target=1 if there exist an object on that pixel, target=0 otherwise).
- The mask branch forces the model to learn finer spatial information of the object. Therefore the performance is increased even if we don't use the mask branch in inference.

ILSVRC Detection 2016 & 2017

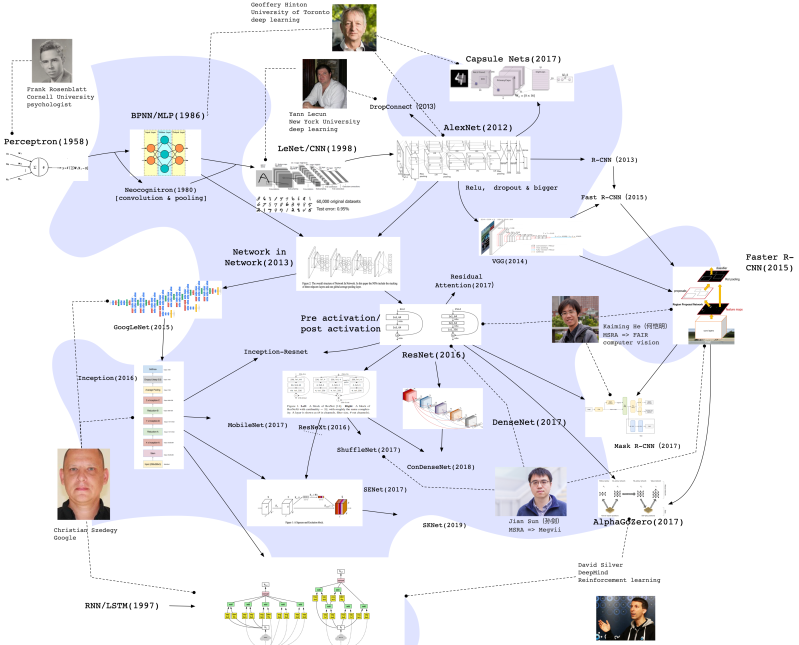
- The winner of 2016, CUImage achieves MAP=0.663 using ensembled methods of fast R-CNN and some other methods.
- The winner of 2017, BDAT, achieves MAP=0.731 using an ensemble of lots of methods.
- The ILSVRC detection dataset is not as influential as the image classification dataset. Currently researchers working on image classification still prefers Imagenet/ILSVRC, but researchers working on object detection prefers the COCO dataset, which is larger and more representative,

Outline

Summary of CNNs

- Scale up neural networks to process very large images / video sequences
  - Sparse connections
  - Parameter sharing
- Automatically generalize across spatial translations of input
- Applicable to any input that is laid out on a grid (1D,2D,3D,$\cdots$)
- The implementation of CNN is a revolution in the subject of computer vision.
- The implementation of AlexNet lowers ImageNet top-5 error rate from 0.26 to 0.15.
- Many modern variations of CNN structures and their ensembles further lowers this error rate to 0.0225, surpassing human level(error rate=0.051). Even single model can achieve 0.038 error rate.

Summary of CNNs



Summary of CNNs

- Finally, researchers develop AutoML to automatically search CNN structures.
- Within 5 years, CNN revolutionized the image classification field. Now even a small CNN inside your cell phone may do facial recognition/image classification better than you.
- We also briefly introduced CNN implementations on object detection. During 4 years of ILSVRC detection competition, the winner performance grows from 0.226 MAP to 0.731 MAP.

Thanks.

HP: http://keg.cs.tsinghua.edu.cn/jietang/
Email: jietang@tsinghua.edu.cn