

Convolutional Neural Network (CNN)

Rowel Atienza

rowel@eee.upd.edu.ph

University of the Philippines

Updated: 26 Sept 2020

Convolutional Neural Network

Convolutional Neural Network (**CNN**) or **CovNet**

Closest model on how human vision works

Uses an operation called **Convolution** instead of Matrix Multiplication (in NN)

Con – Latin word for *together*

Volvere – Latin word for *roll up*

CNN

$$y = x * k$$

$x \in \mathbb{R}^{w \times h \times d}$ is input: d feature maps with dimensions $w \times h$

$k \in \mathbb{R}^{k \times k \times f}$ is kernel: f filters or kernels with dimensions $k \times k$

$y \in \mathbb{R}^{w \times h \times f}$ is output: f feature maps with dimensions $m \times n$ assuming sufficient padding is applied

Dimensions are based on 2D inputs and square kernels

Kernel

In MLP, we learn weights and biases

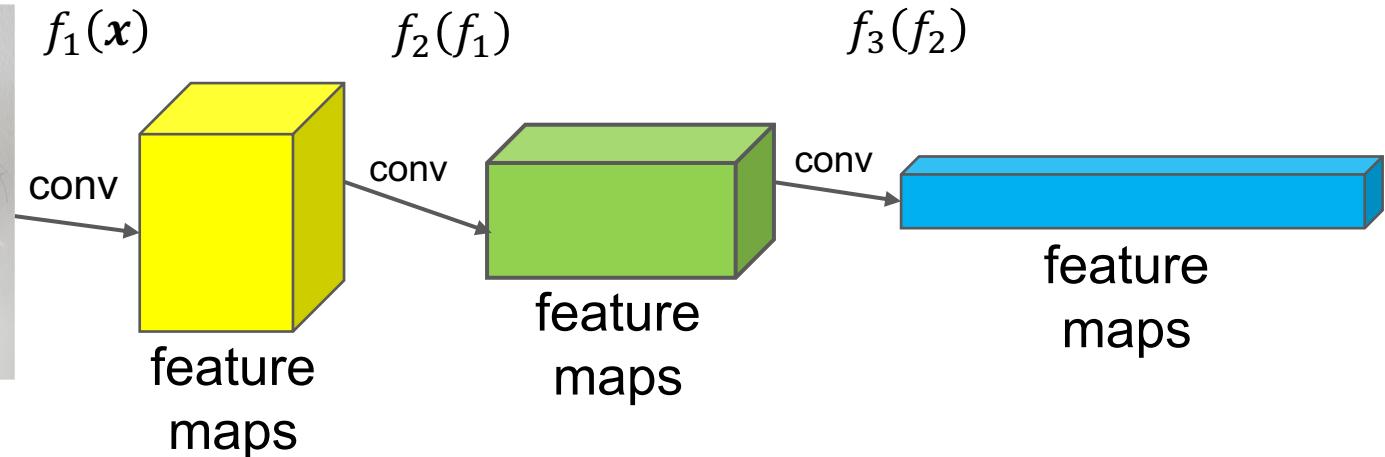
In CNN, we learn the parameters in the kernel

CNN

The deeper the network, the more representations the network learns

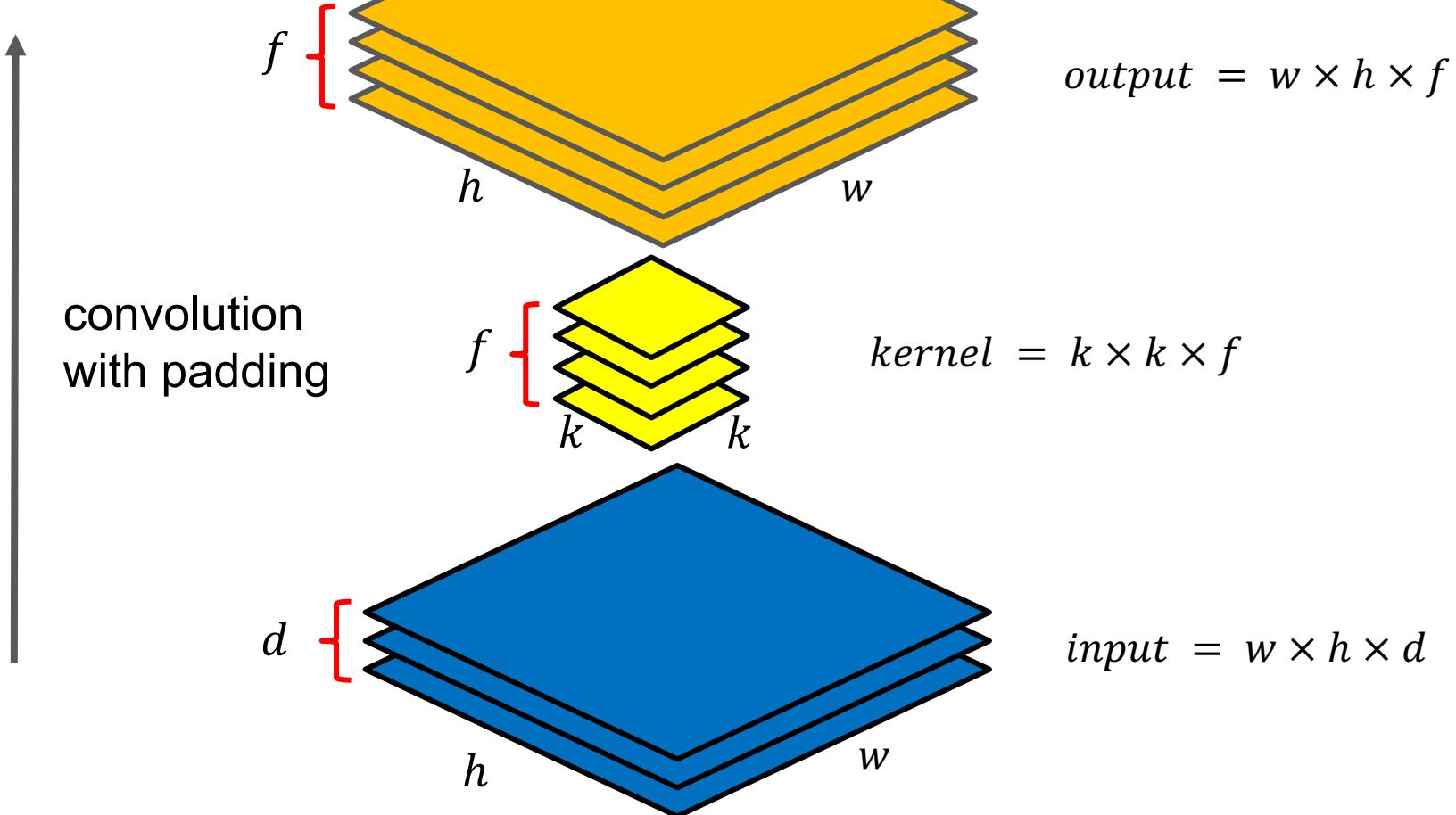


input image

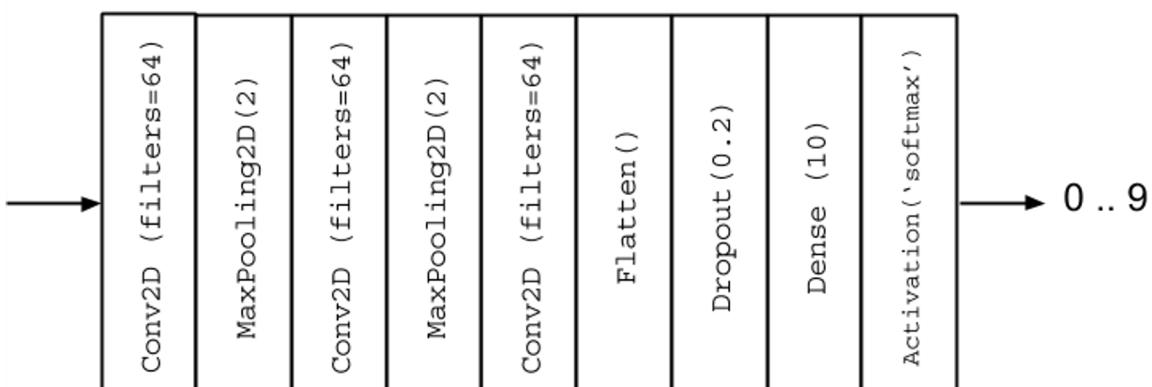
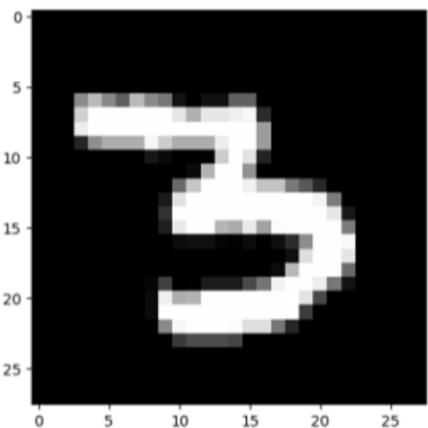


$$y = f(x) \approx f_n \circ f_{n-1} \circ f_{n-2} \circ \cdots \circ f_1 (x)$$

CNN



CNN Model on MNIST



Keras Conv2D has built-in ReLU activation

Properties

Sparse Interaction - kernel as a feature detector is small compared to the input image thus requires few interaction only

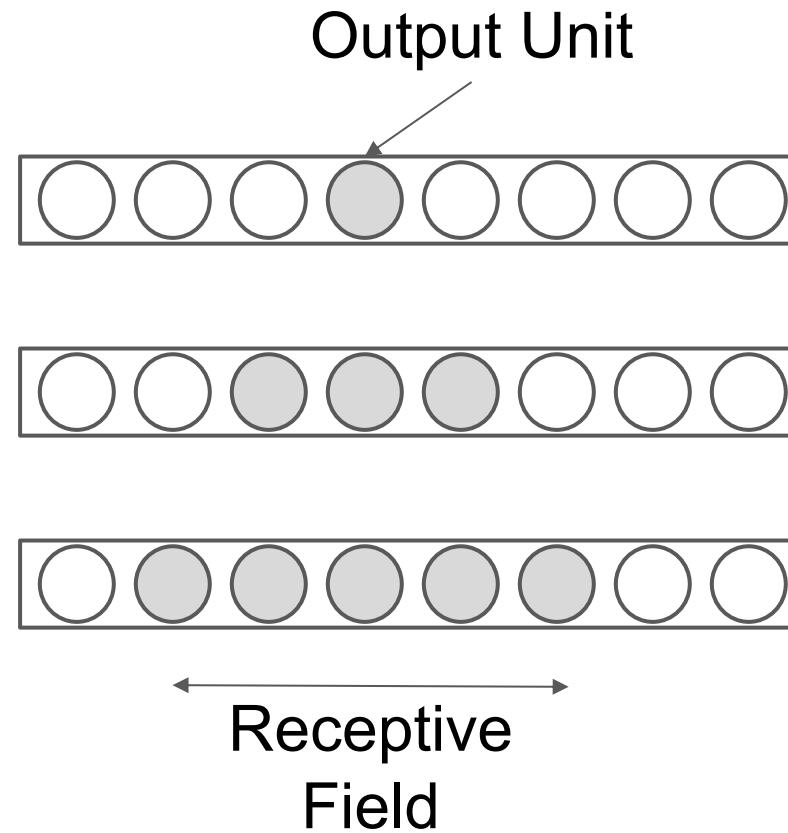
Parameter Sharing - same set of parameters used for more than 1 function in the model

Receptive Field - the input units that affect the output unit

The deeper the network, the larger is the receptive field

Equivariance to Translation (Scale? Rotation?) - if the input changes, the output changes in the same way

Receptive Field



Equivariance to Translation



CNN Ops

Convolution

Activation

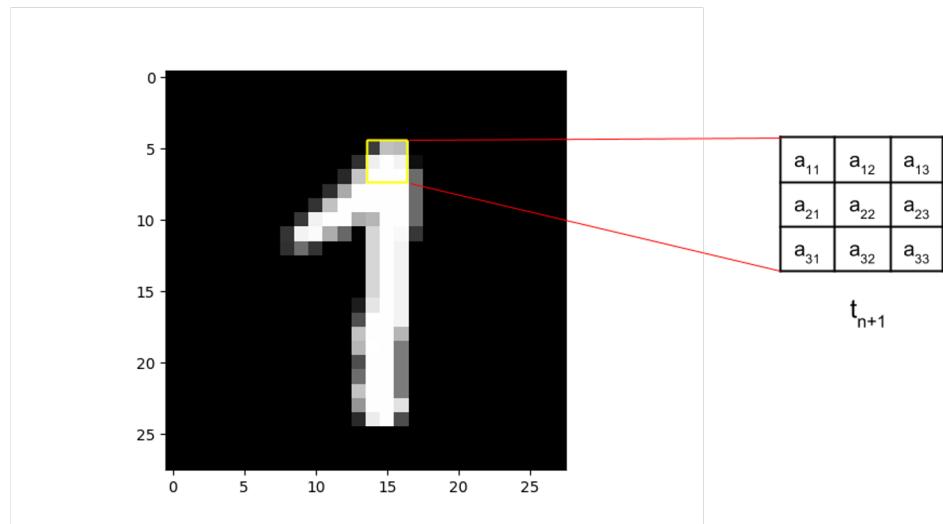
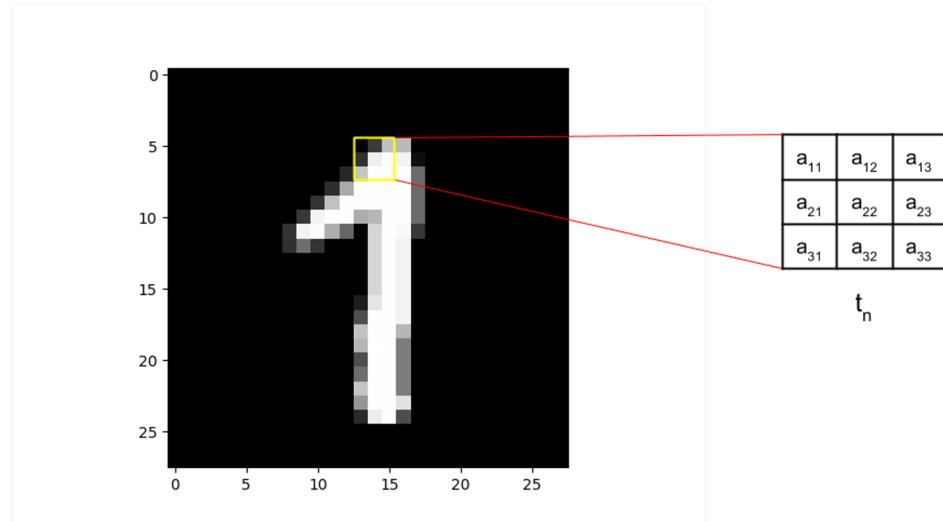
Padding

Pooling

Strides

Filters

Convolution



CNN: Convolution No Padding (Valid)

$$\begin{matrix} & X \\ \begin{matrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{matrix} & * & \begin{matrix} K \\ \begin{matrix} w & x \\ y & z \end{matrix} \end{matrix} & = & \begin{matrix} y \\ \begin{matrix} y_{11} & y_{12} & y_{13} \\ y_{21} & y_{22} & y_{23} \\ y_{31} & y_{32} & y_{33} \end{matrix} \end{matrix} \end{matrix}$$

$$y_{11} = aw + bx + ey + fz$$

$$i = 1, 2, 3$$

$$j = 1, 2, 3$$

$$m = 1, 2, 3$$

$$n = 1, 2, 3$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

=

$$y_{12} = bw + cx + fy + gz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

$$y_{13} = cw + dx + gy + hz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

$$y_{21} = ew + fx + iy + jz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

=

$$y_{22} = fw + gx + jy + kz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

$$y_{23} = gw + hx + ky + lz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

$$y_{31} = iw + jx + my + nz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

$$y_{32} = jw + kx + ny + oz$$

CNN

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

w	x
y	z

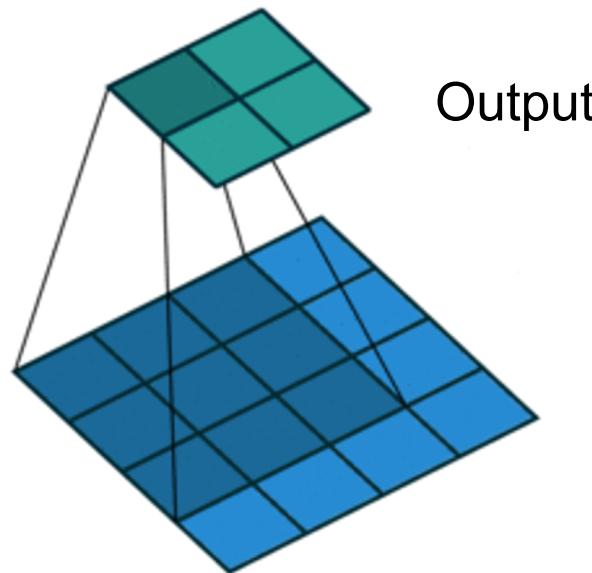
=

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

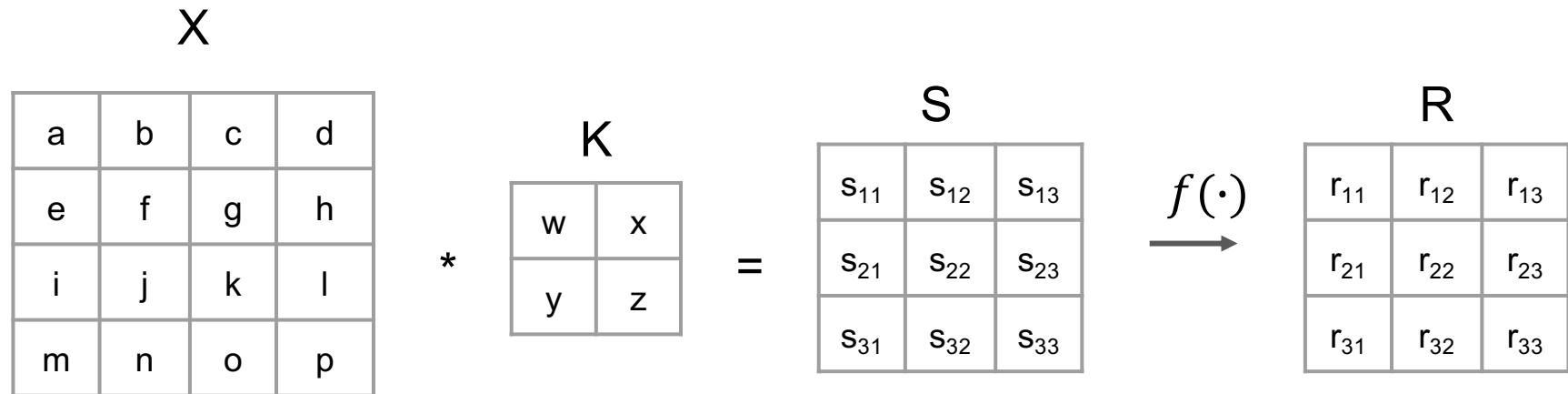
$$y_{33} = kw + lx + oy + pz$$

CNN in GIF (No padding, stride=1)



https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

CNN: Activation Function - ReLU



$$R = \text{ReLU}(S)$$

$$r_{ij} = \text{ReLU}(s_{ij})$$

Element-wise ReLU

CNN: Downsampling - Pooling (eg MaxPooling)

R

r_{11}	r_{12}	r_{13}
r_{21}	r_{22}	r_{23}
r_{31}	r_{32}	r_{33}

P

=

p_{11}

$$p_{11} = \max(r_{11}, r_{12}, r_{21}, r_{22})$$

CNN: Downsampling - Pooling

Other Pooling Functions

Average

Median

CNN: Downsampling using Stride > 1, (e.g. 2)

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

$$\begin{array}{ccc}
 X & K & y \\
 \begin{matrix} w & x \\ y & z \end{matrix} & = & \begin{matrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{matrix} \\
 * & & \\
 y_{11} = aw + bx + ey + fz & &
 \end{array}$$

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

$$\begin{array}{ccc}
 X & K & y \\
 \begin{matrix} w & x \\ y & z \end{matrix} & = & \begin{matrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{matrix} \\
 * & & \\
 y_{12} = cw + dx + gy + hz & &
 \end{array}$$

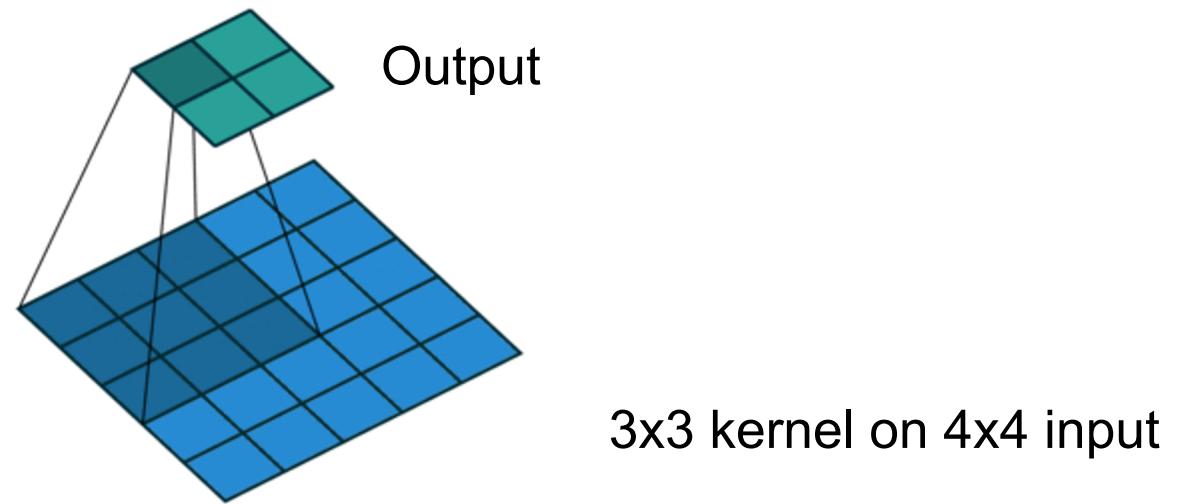
a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

$$\begin{array}{ccc}
 X & K & y \\
 \begin{matrix} w & x \\ y & z \end{matrix} & = & \begin{matrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{matrix} \\
 * & & \\
 y_{21} = iw + jx + my + nz & &
 \end{array}$$

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

$$\begin{array}{ccc}
 X & K & y \\
 \begin{matrix} w & x \\ y & z \end{matrix} & = & \begin{matrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{matrix} \\
 * & & \\
 y_{22} = kw + lx + oy + pz & &
 \end{array}$$

CNN: Downsampling using Stride = 2



https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

CNN: Downsampling using MaxPooling (MP)

$$\text{MP}(X) = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}$$

$y_{11} = \max(a, b, e, f)$

$$\text{MP}(X) = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}$$

$y_{12} = \max(c, d, g, h)$

$$\text{MP}(X) = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}$$

$y_{21} = \max(i, j, m, n)$

$$\text{MP}(X) = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}$$

$y_{22} = \max(k, l, o, p)$

CNN: Zero Padding

X

0	0	0	0	0	0
0	a	b	c	d	0
0	e	f	g	h	0
0	i	j	k	l	0
0	m	n	o	p	0
0	0	0	0	0	0

*

K

r	s	t
u	v	w
x	y	z

=

y_{11}	y_{12}	y_{13}	y_{14}
y_{21}	y_{22}	y_{23}	y_{24}
y_{31}	y_{32}	y_{33}	y_{34}
y_{41}	y_{42}	y_{43}	y_{44}

$$y_{11} = av + bw + ey + fz$$

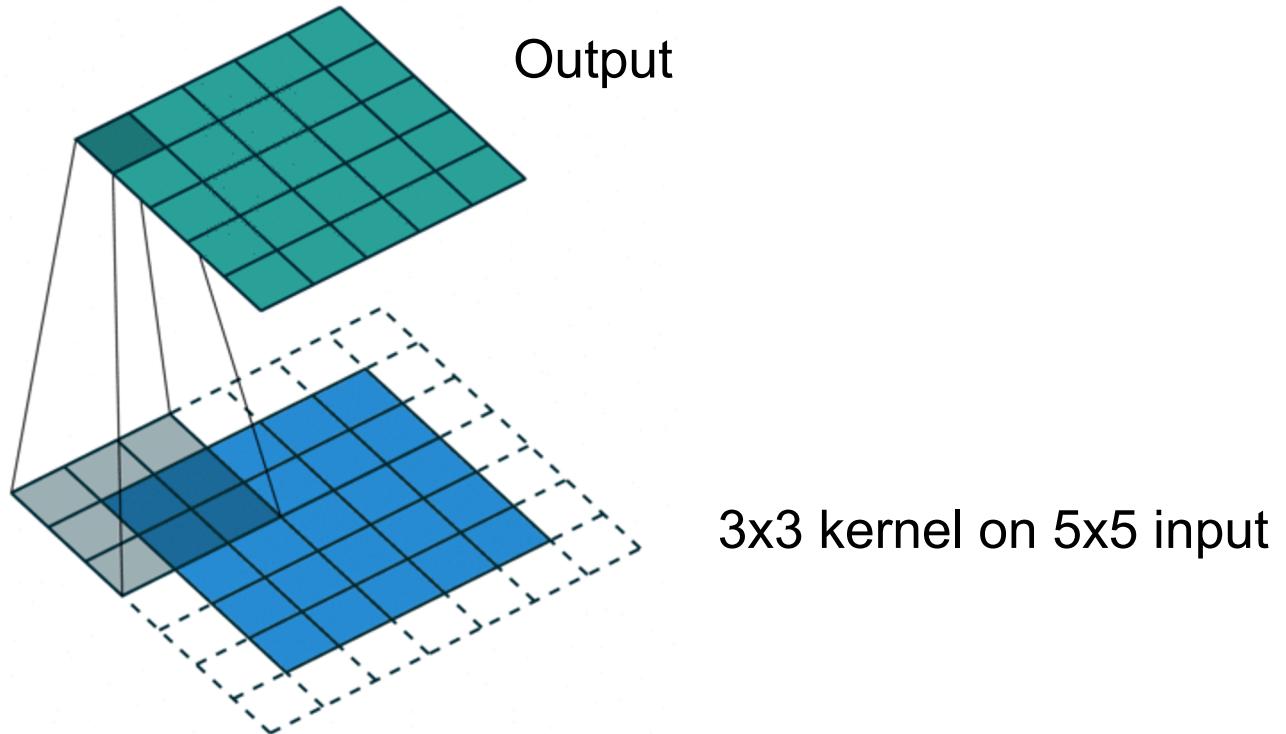
$$y_{12} = au + bv + cw + ex + fy + gz$$

$$y_{13} = bu + cv + dw + fx + gy + hz$$

$$y_{14} = cu + dv + gx + hy$$

etc

CNN: Zero Padding, strides=1



CNN: K kernels/filters

X

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

*

K

s	t
u	v

w	x
y	z

K = 2

y

y_{11}	y_{12}	y_{13}
y_{21}	y_{22}	y_{23}
y_{31}	y_{32}	y_{33}

=

$$y_{11} = ay + bt + eu + fv$$

etc.

t_{11}	t_{12}	t_{13}
t_{21}	t_{22}	t_{23}
t_{31}	t_{32}	t_{33}

$$t_{11} = aw + bx + ey + fz$$

etc.

CNN: Dilated Convolution

Dilation rate > 1 increases kernel coverage w/o increasing computation time

a_{11}	a_{12}	a_{13}
a_{21}	a_{22}	a_{23}
a_{31}	a_{32}	a_{33}

dilation_rate=1

a_{11}		a_{12}		a_{13}
a_{21}		a_{22}		a_{23}
a_{31}		a_{32}		a_{33}

dilation_rate=2

Dilated Convolution No Padding (Valid)

X	a	b	c	d
e	a	b	c	d
i	e	f	g	h
m	i	j	k	l
n	m	n	o	p

$$X \quad * \quad K = y$$

dilation_rate=1

$y_{11} = aw + bx + ey + fz$

The diagram illustrates a convolution operation with dilation rate 1. The input matrix X has dimensions 4x4, and the kernel K has dimensions 2x2. The output matrix y has dimensions 3x3. The calculation for the first element y_{11} is shown as a weighted sum of elements from X and K .

X	a	b	c	d
e	a	b	c	d
i	e	f	g	h
m	i	j	k	l
n	m	n	o	p

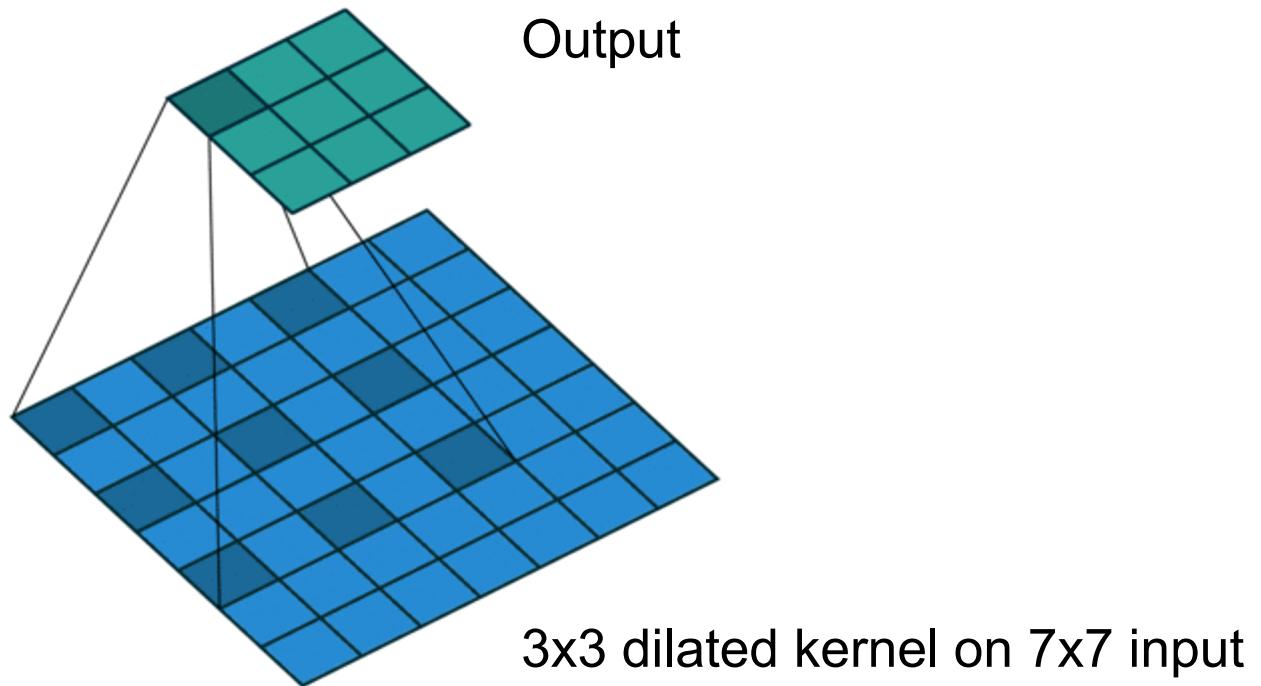
$$X \quad * \quad K = y$$

dilation_rate=2

$y_{11} = aw + cx + iy + kz$

The diagram illustrates a convolution operation with dilation rate 2. The input matrix X has dimensions 4x4, and the kernel K has dimensions 2x2. The output matrix y has dimensions 2x2. The calculation for the first element y_{11} is shown as a weighted sum of elements from X and K .

Dilated convolution



UpSampling

$$\text{UP}(\begin{array}{|c|c|}\hline a & b \\ \hline c & d \\ \hline\end{array}) =$$

a	a	b	b
a	a	b	b
c	c	d	d
c	c	d	d

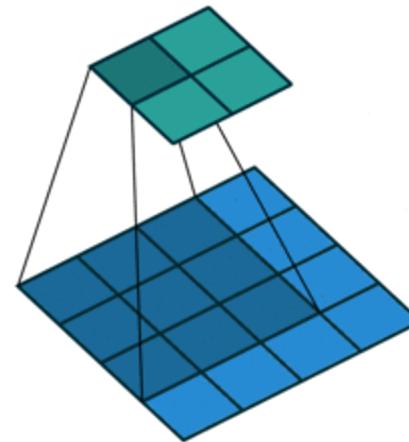
Interpolation: same data repeated n times
Other interpolation algorithms: Bilinear

Transposed Convolution

Convolution + Upsampling (if strides>2)

See more examples of CNN operations in

https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md



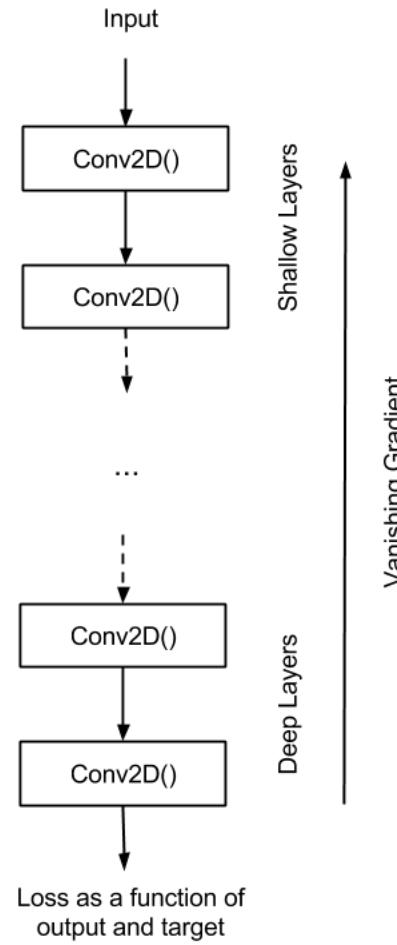
Issues with Deep Neural Networks

Vanishing Gradients

Exploding Gradients

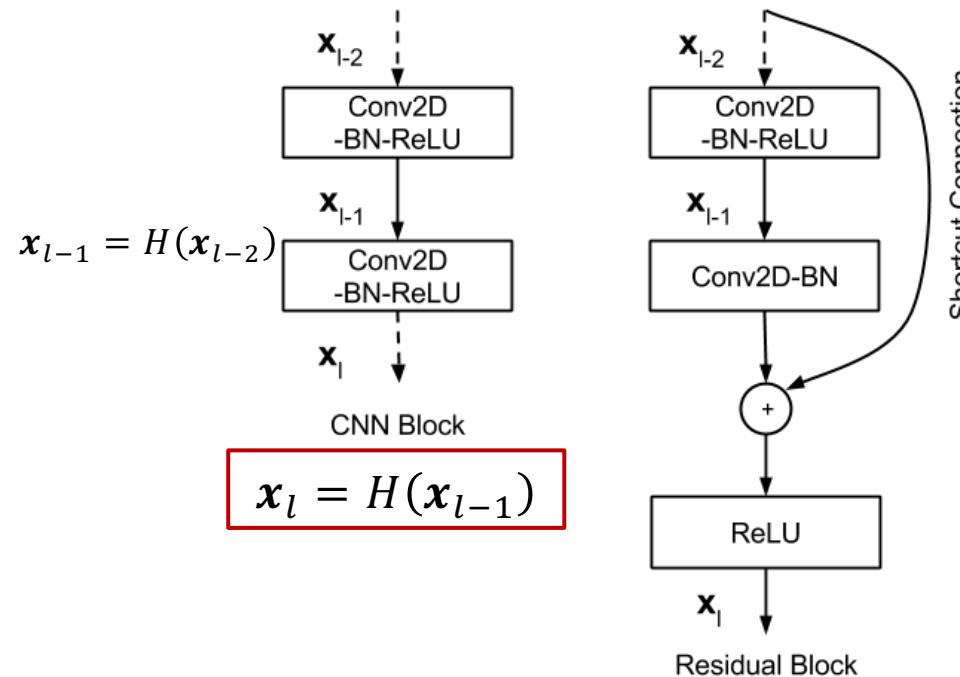
Unstable Training

Vanishing Gradients

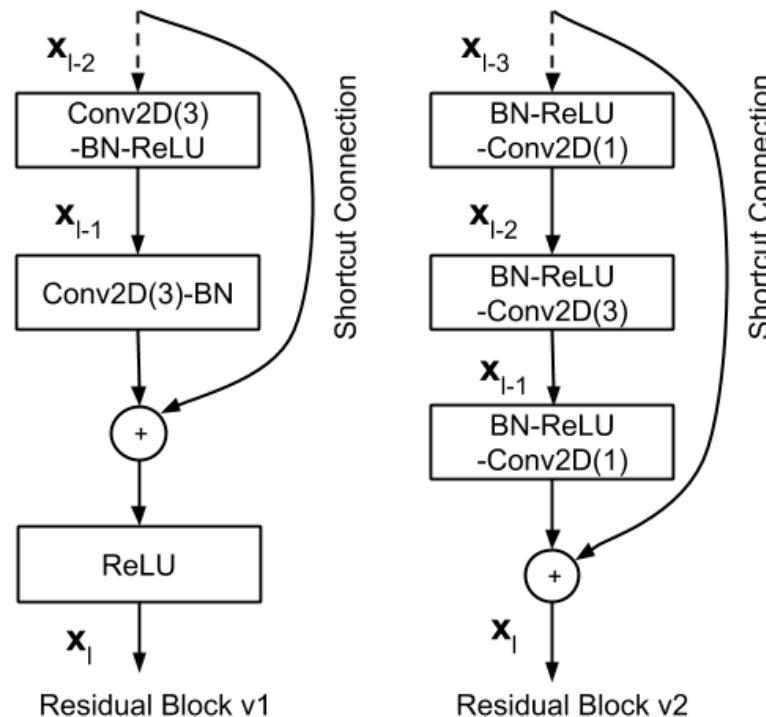


ResNet

By introducing a skip connection, ResNet avoids the problem of vanishing gradients

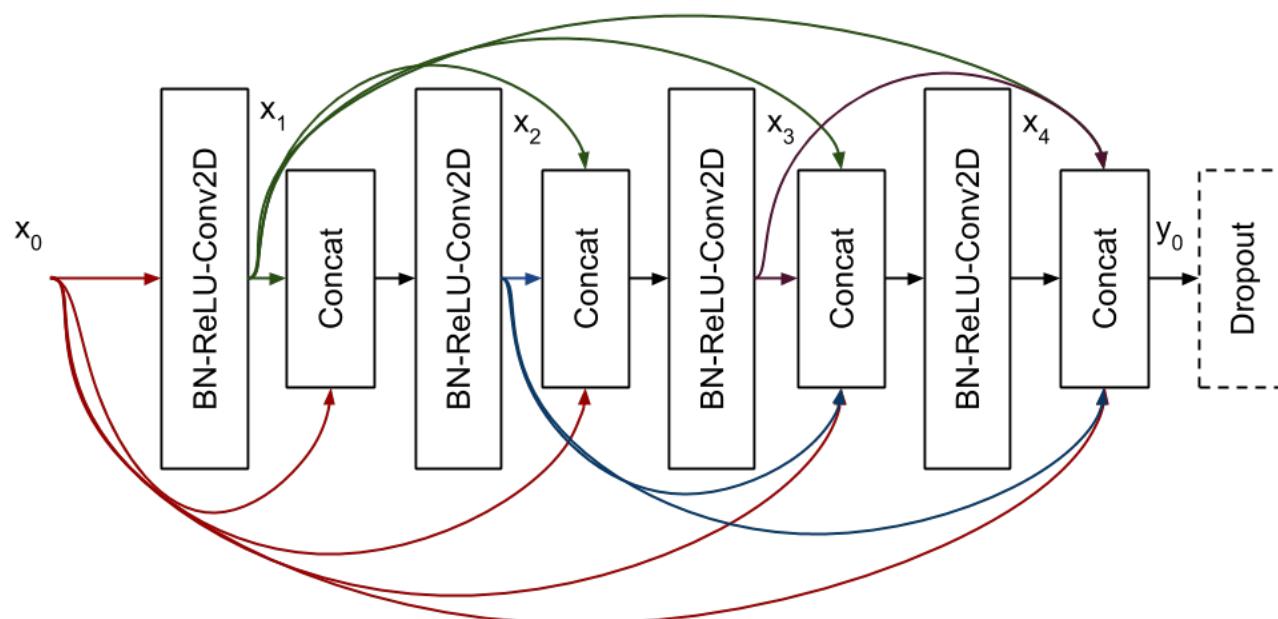


Improved ResNet



DenseNet

Why not interconnect all layers?



Batch Normalization

Applied layer-wise to maintain zero mean and variance of 1 for activation outputs

Batch normalization reduces the amount by what the hidden unit values shift around (covariance shift)

Allows use of larger learning rate in deep models w/o causing instability

Batch Normalization

Applied to any input or hidden layer

$$H' = (H - \mu)/\sigma \quad (\text{row-wise operation, one sample is one row})$$

where H matrix is a minibatch of activations of the layer to normalize

μ vector of mean of each unit, σ vector of std of each unit
(broadcast op)

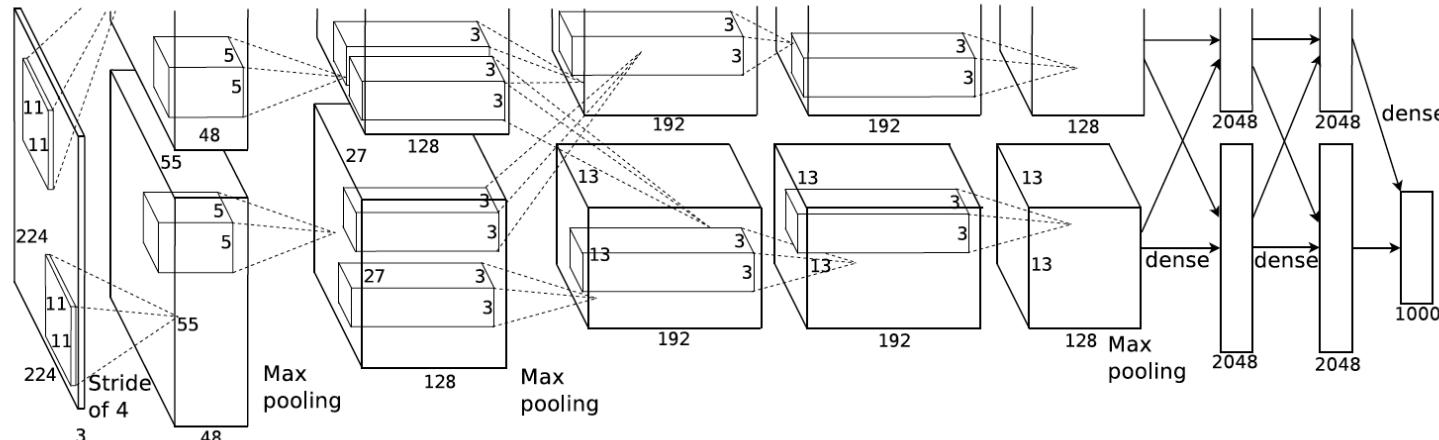
Assignment (Due: Oct 10)

1. Build a classifier on CIFAR10 dataset using
 - a. 3-layer MLP
 - b. 3-layer CNN
2. The last layer is Dense
3. Up to you to determine the hyperparameters (ie kernel size, # hidden units, # of filters, learning rate, etc) but the optimizer must be SGD
4. Show your solution using Jupyter notebook
5. Implement using Keras or Pytorch
6. Compare performance of both networks (best MLP vs best CNN)
7. Best performing network has additional points

In Summary

CNN is parameter efficient, parallelizable and translation equivariant
(invariant with depth) layer

Deep CNN exhibits state of the start performances not only in vision tasks



AlexNet [Krizhevsky et al (2012)]