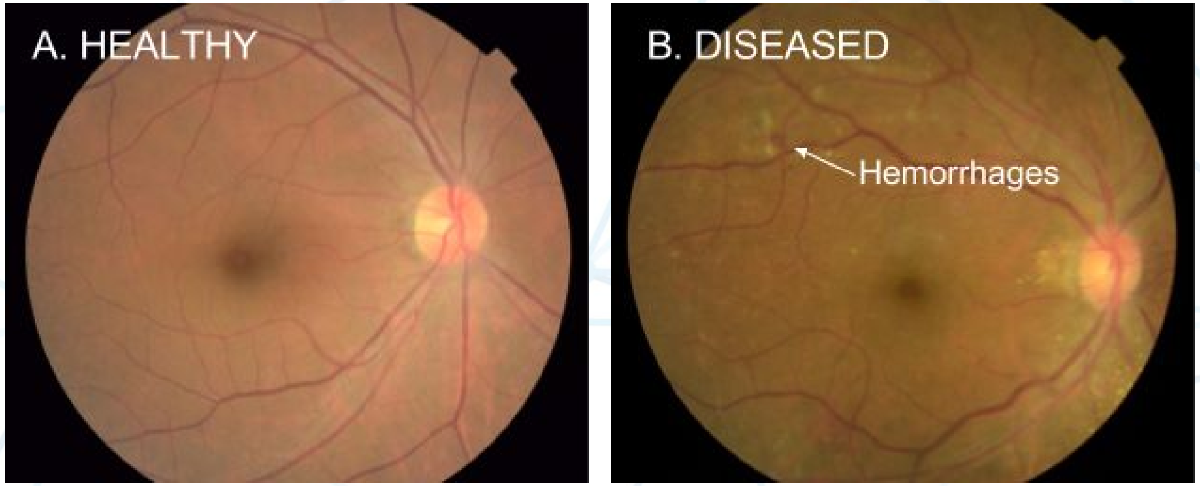
FORNAX

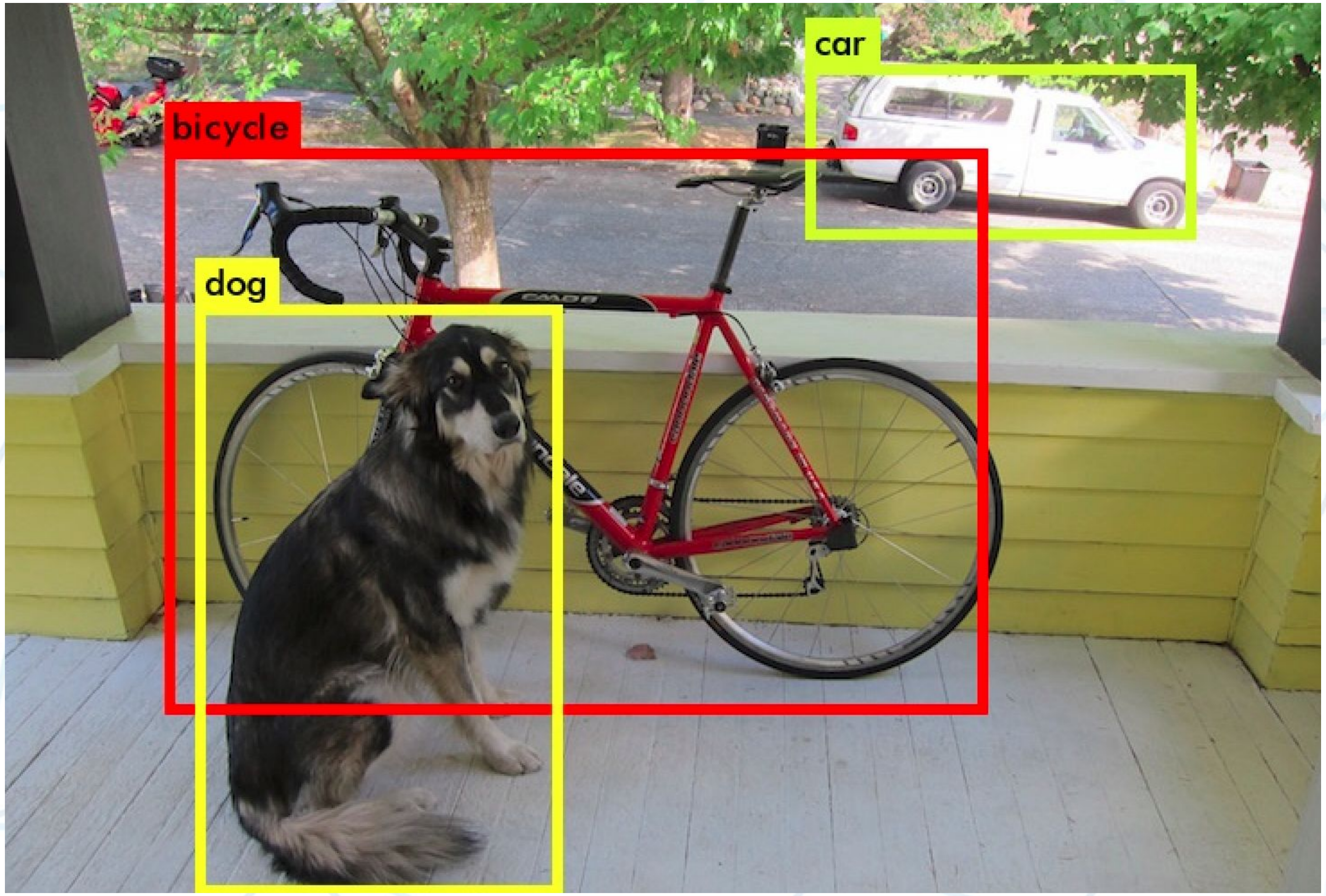
Introduction to Deep Learning, part 2

- Rafał Cycoń, FORNAX

Image classification



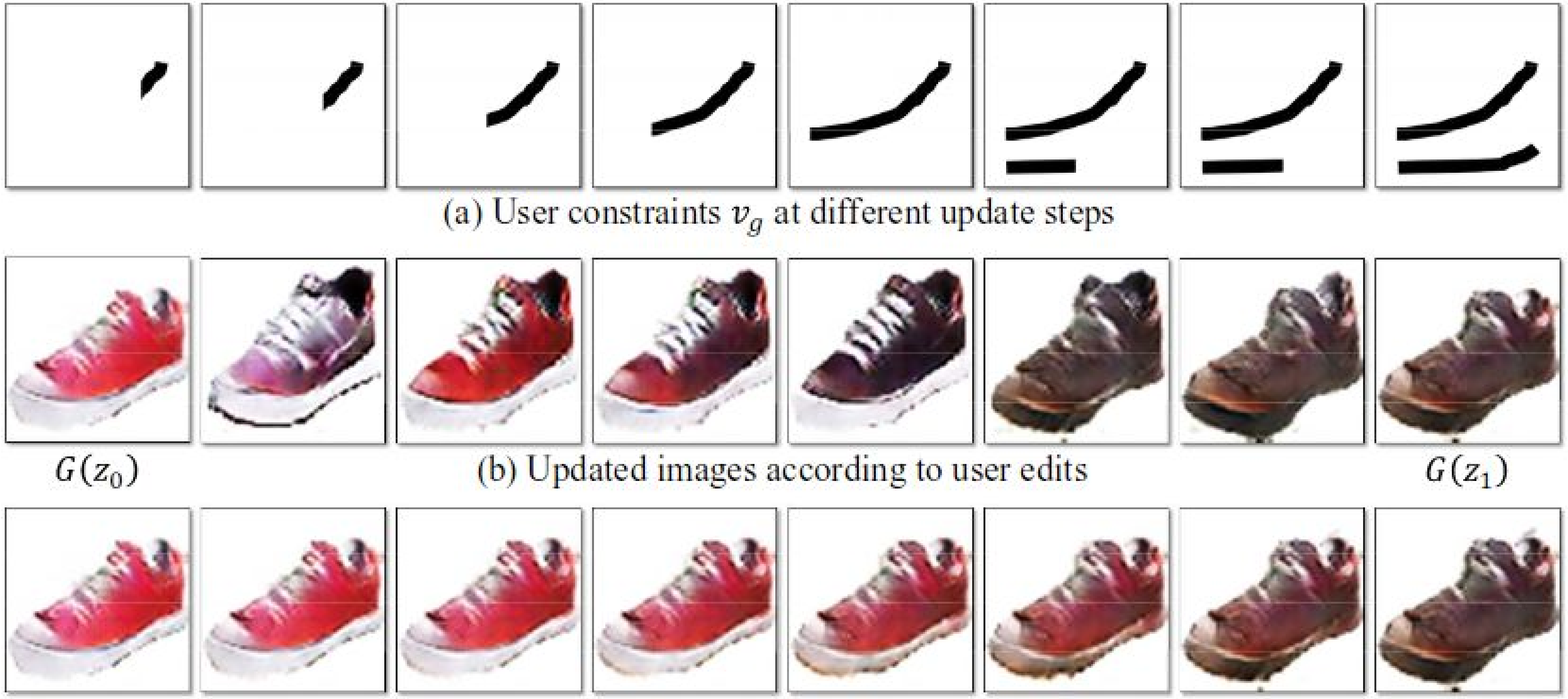
Localization



Photorealistic style transfer



Image generation

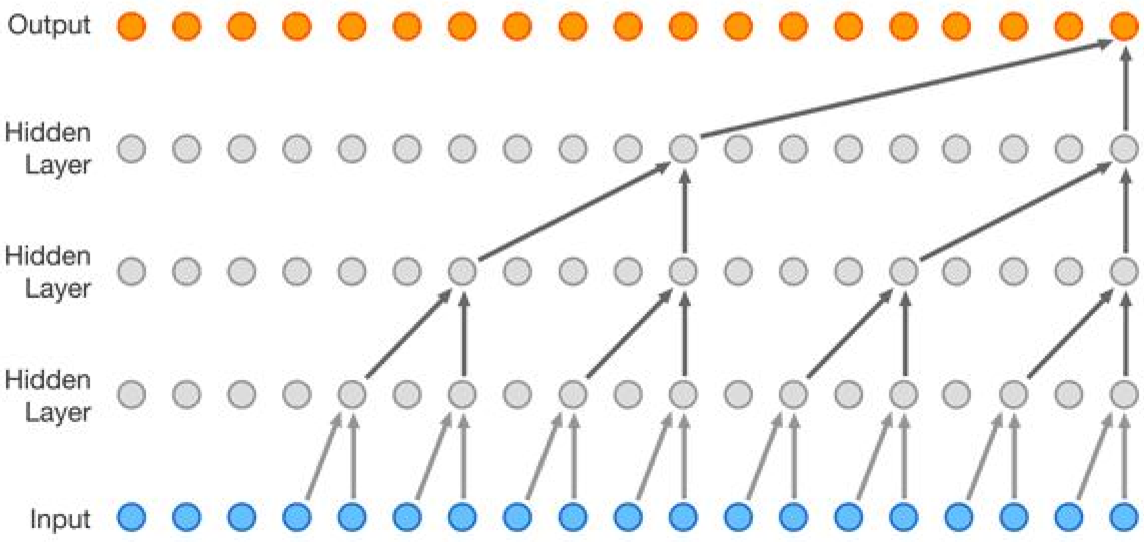


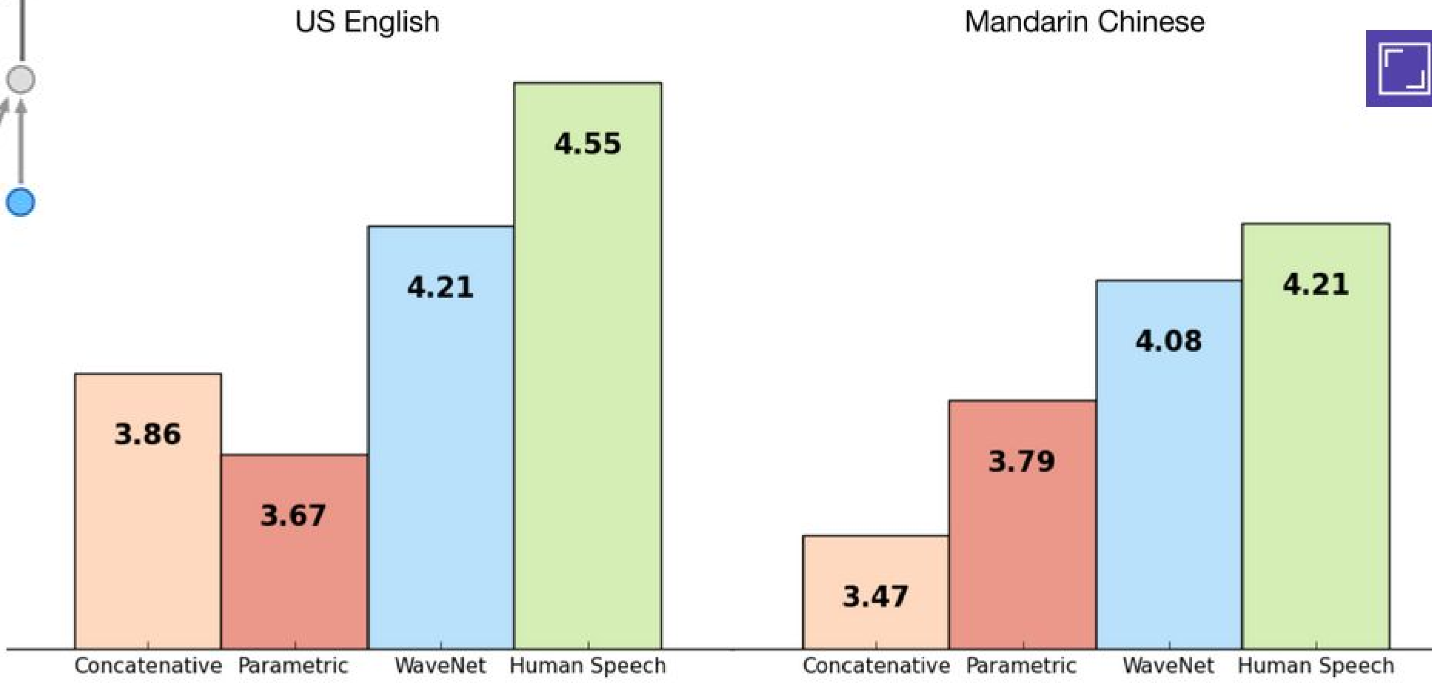
(a) User constraints v_g at different update steps

$G(z_0)$ (b) Updated images according to user edits $G(z_1)$

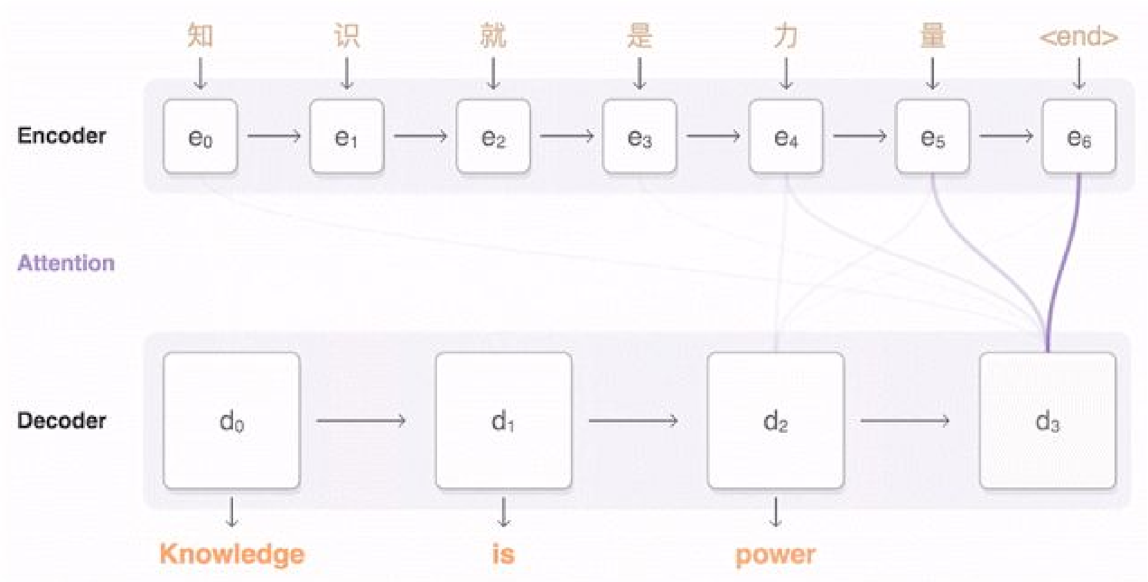
(c) Linear interpolation between $G(z_0)$ and $G(z_1)$

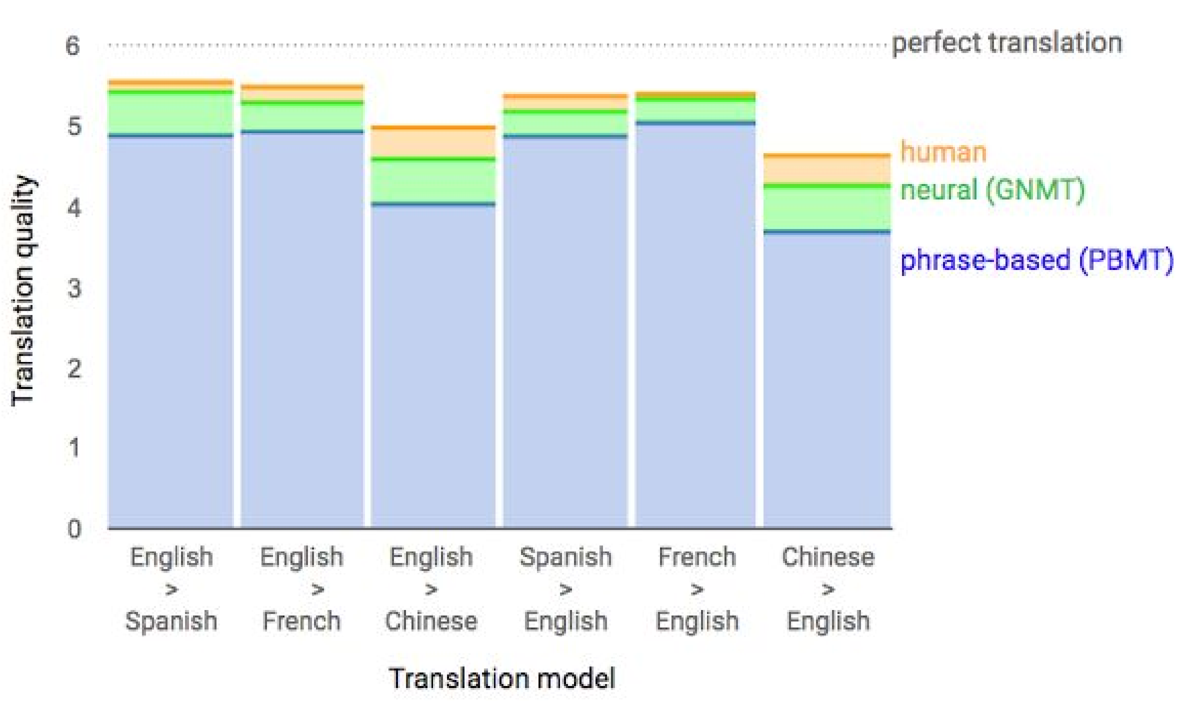
Sound synthesis





Translation

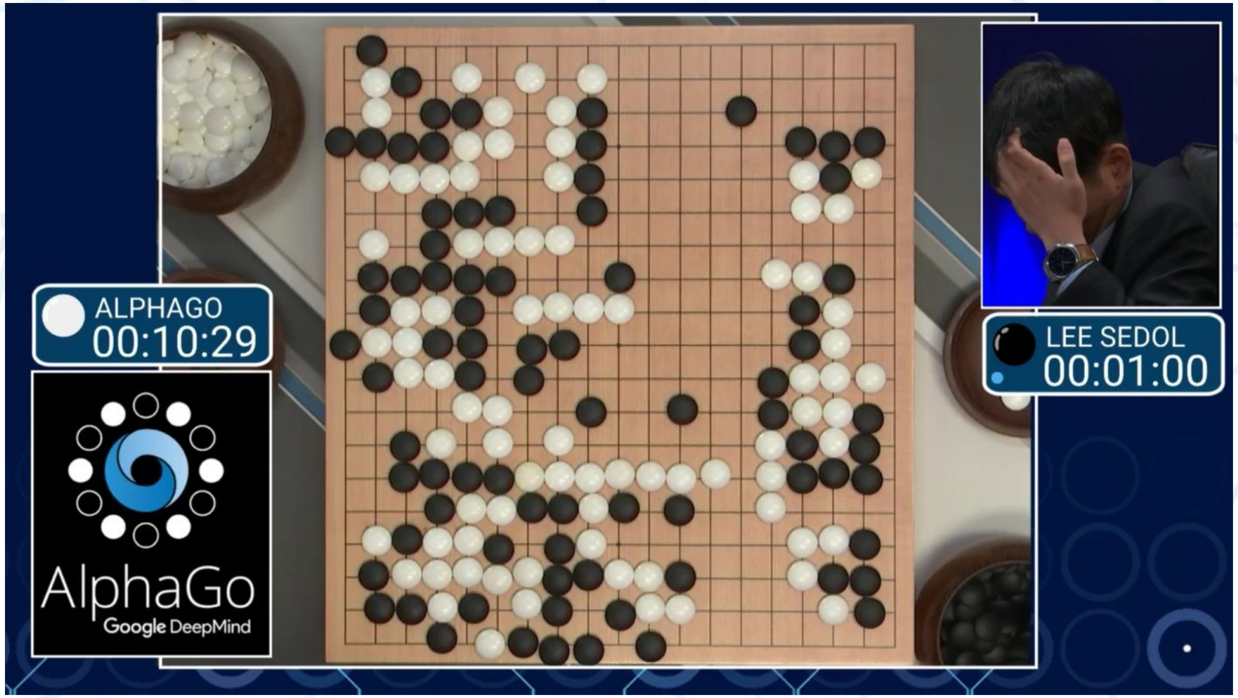




Text generation

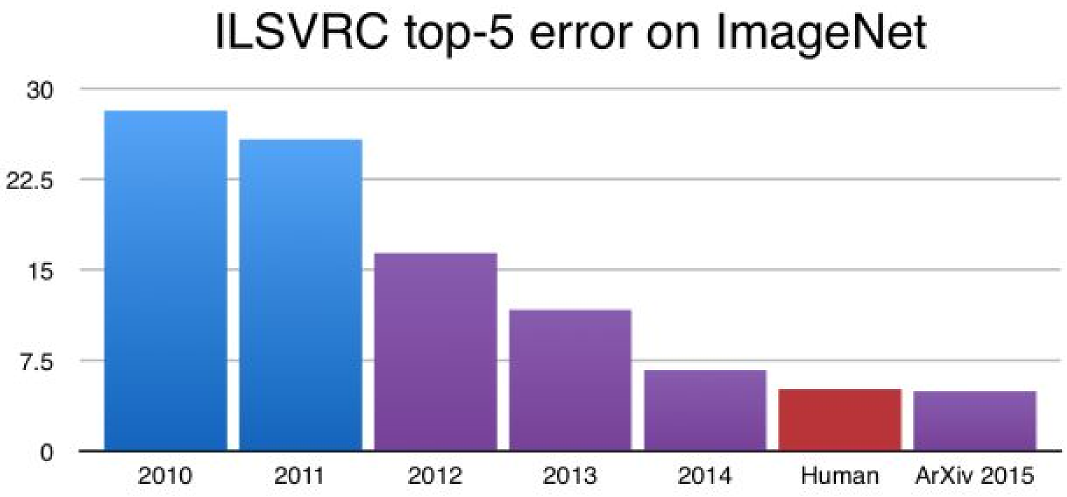
```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

Reinforcement learning



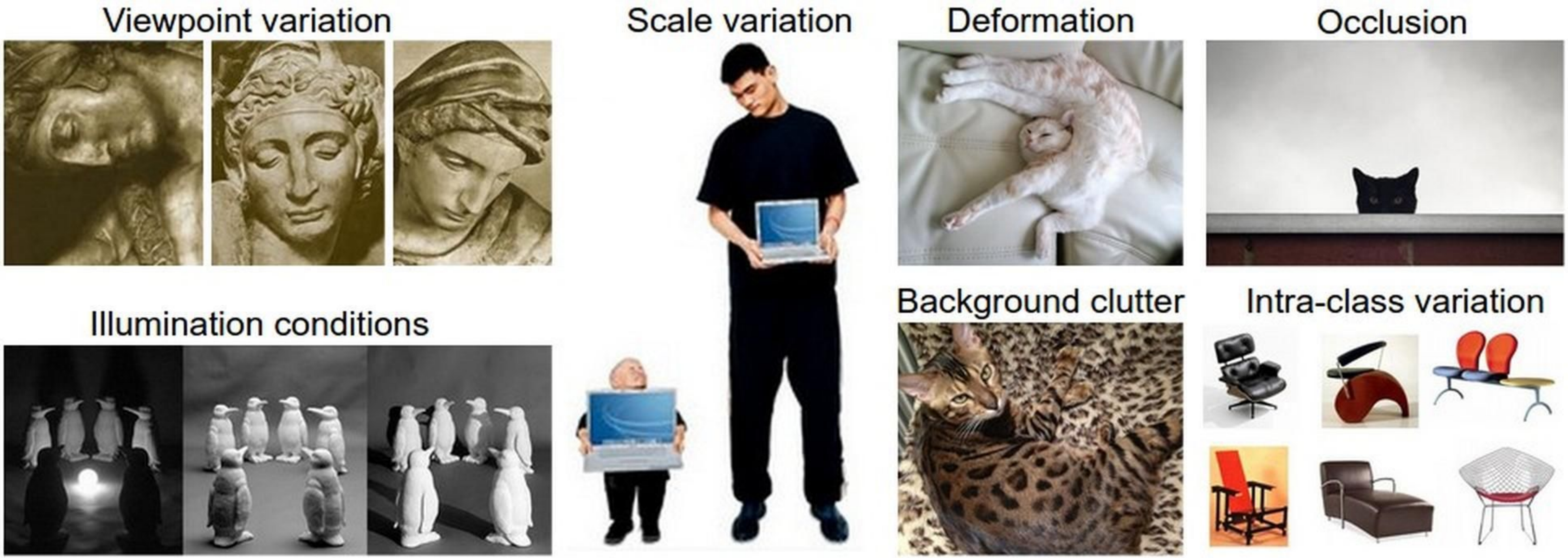
ImageNet

mite
container ship
motor scooter
leopard
mite
black widow
cockroach
tick
starfish
container ship
lifeboat
amphibian
fireboat
drilling platform
motor scooter
go-kart
moped
bumper car
golfcart
leopard
jaguar
cheetah
snow leopard
Egyptian cat



ILSVRC top-5 error on ImageNet
30
22.5
15
7.5
0
2010
2011
2012
2013
2014
Human
ArXiv 2015

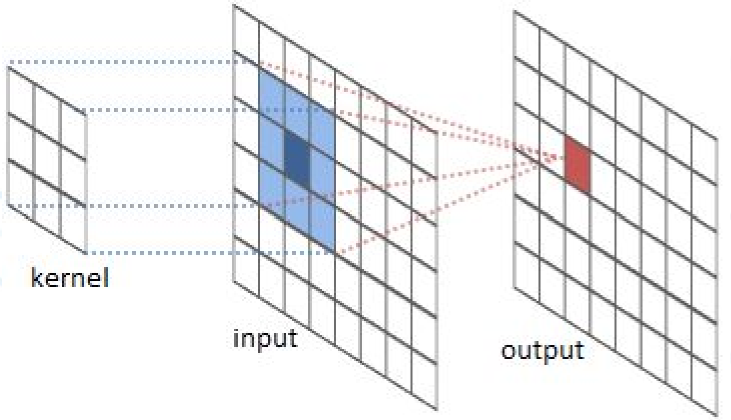
Challenges in image processing

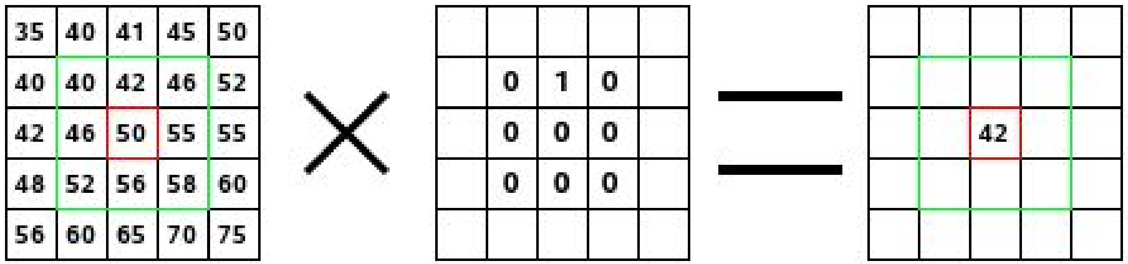


Convolution

$$(f * g)(t) \stackrel{\mathrm{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$
$$= \int_{-\infty}^{\infty} f(t-\tau)\, g(\tau)\, d\tau.$$

$$(f * g)[n] \stackrel{\mathrm{def}}{=} \sum_{m=-\infty}^{\infty} f[m]\, g[n-m]$$
$$= \sum_{m=-\infty}^{\infty} f[n-m]\, g[m].$$

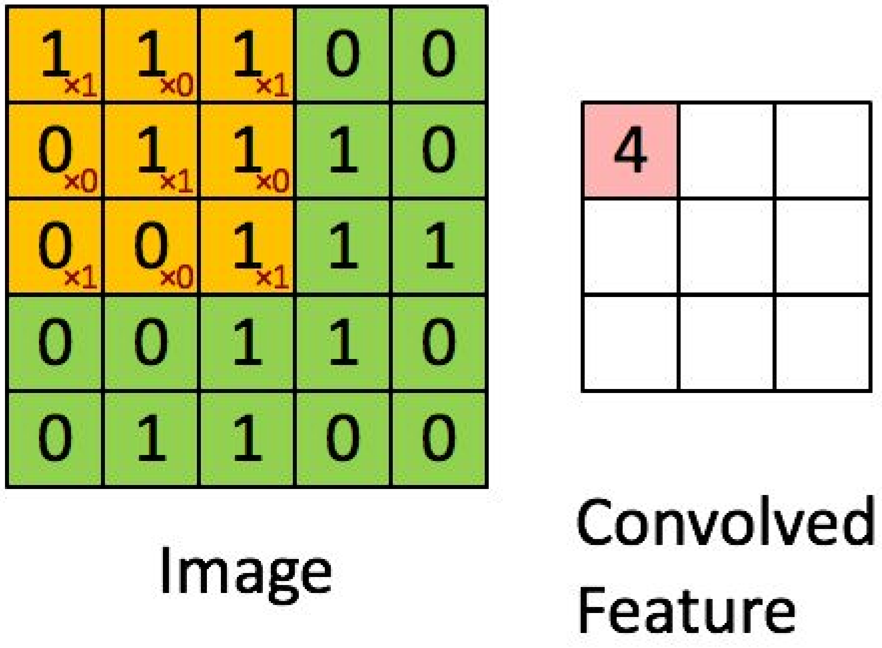




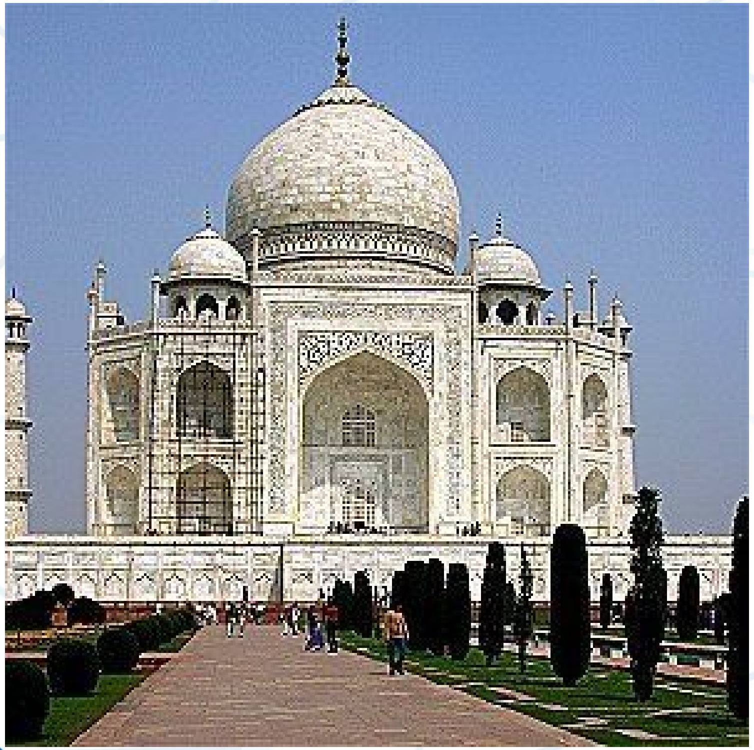
Convolution

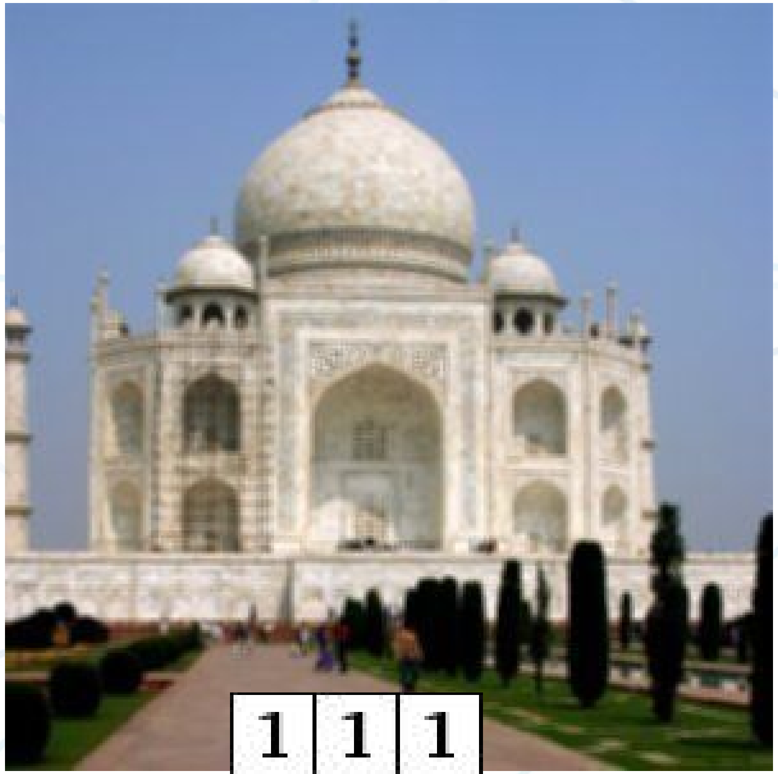
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

1	0	1
0	1	0
1	0	1

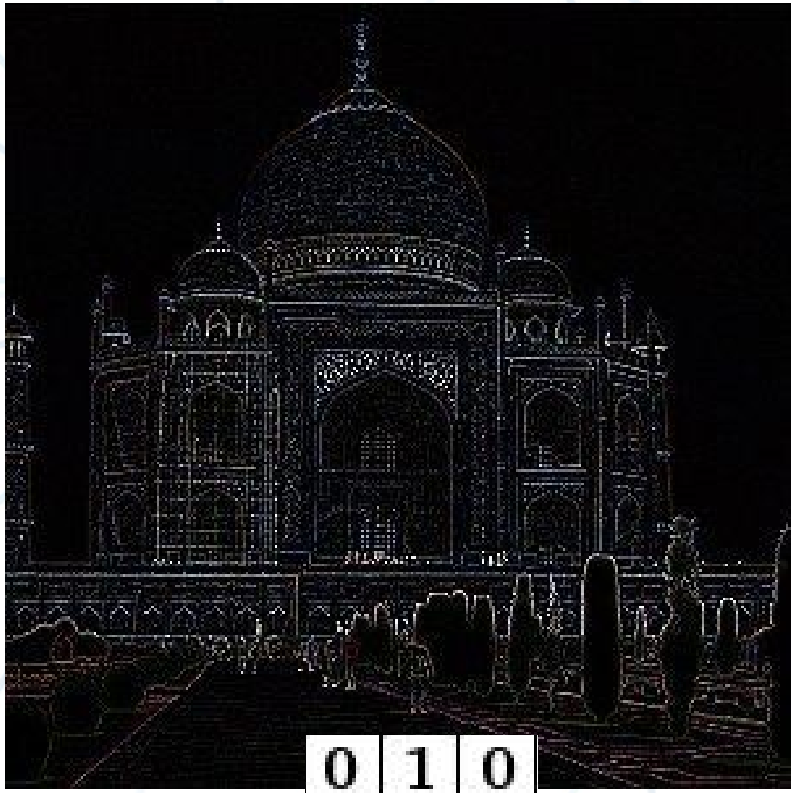


Convolution



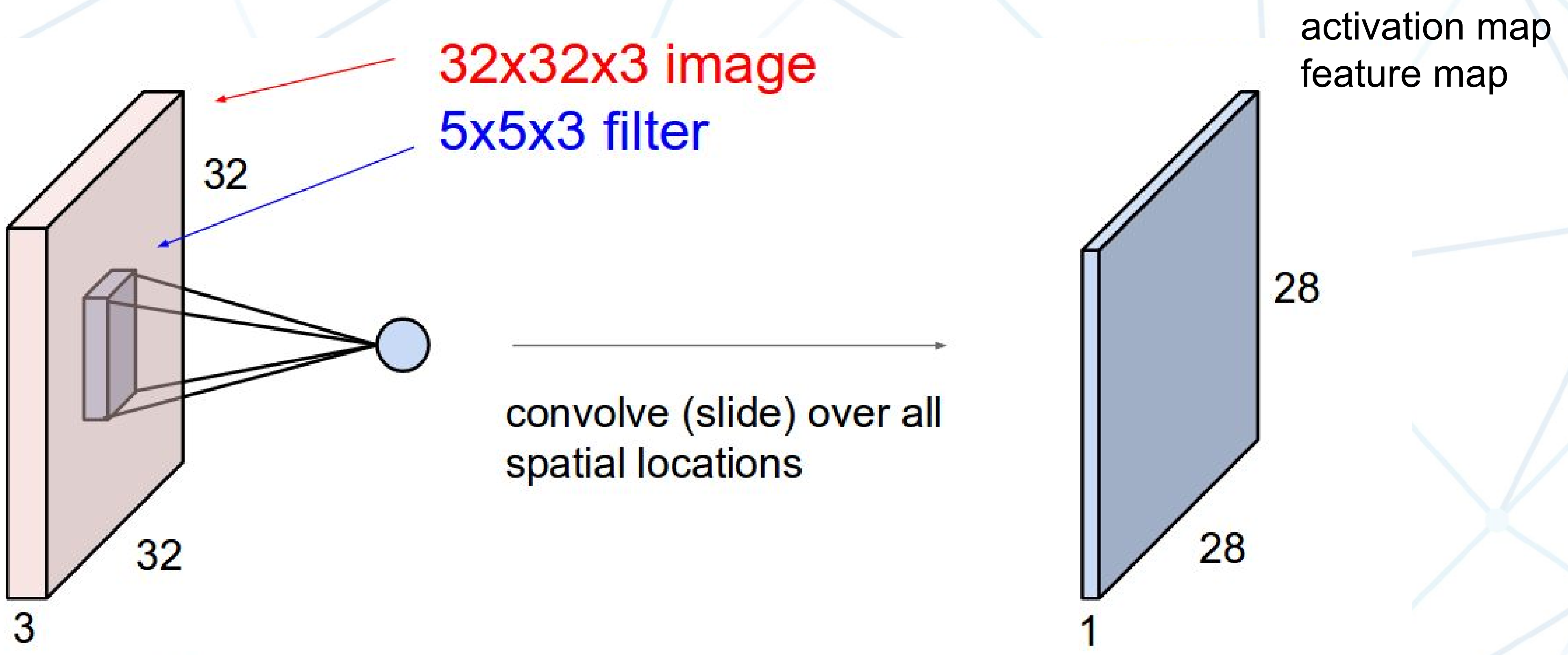


1	1	1
1	1	1
1	1	1



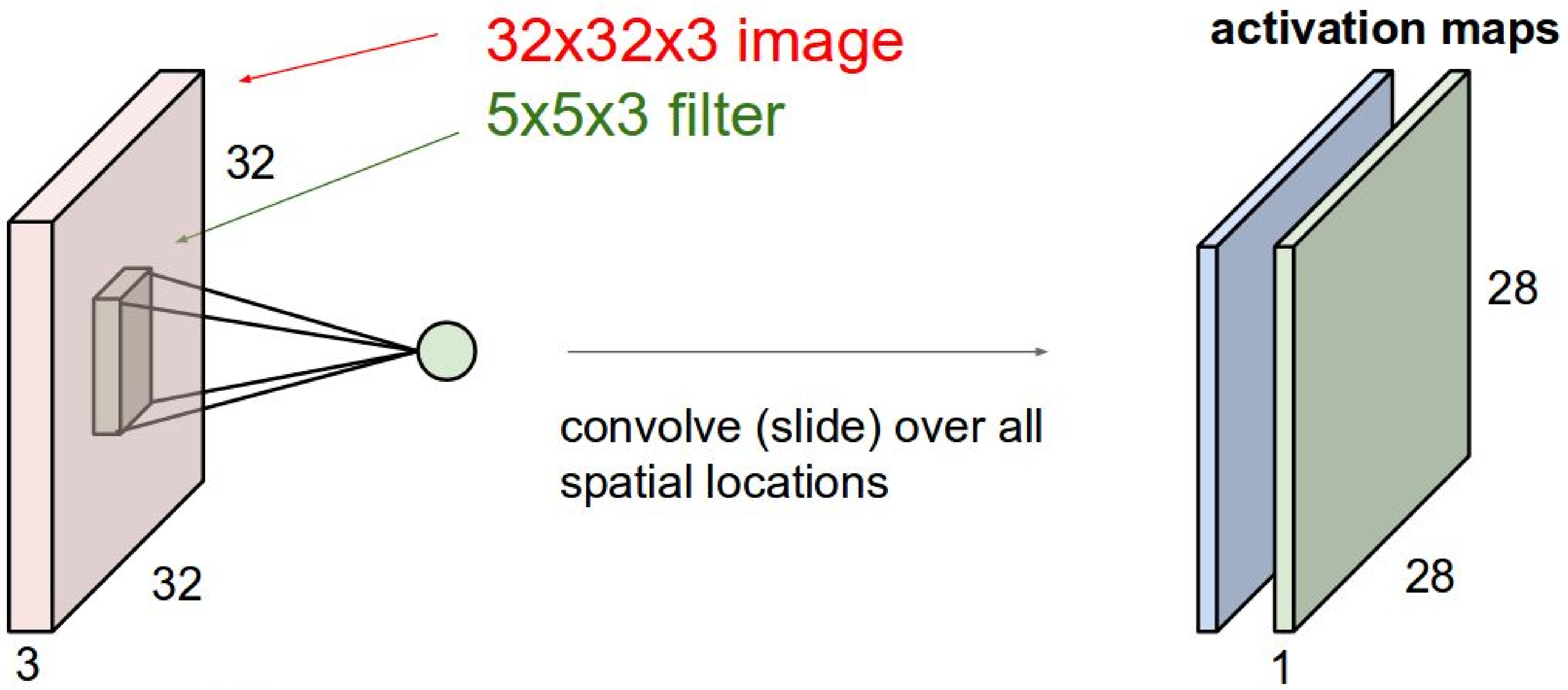
0	1	0
1	-4	1
0	1	0

Convolution



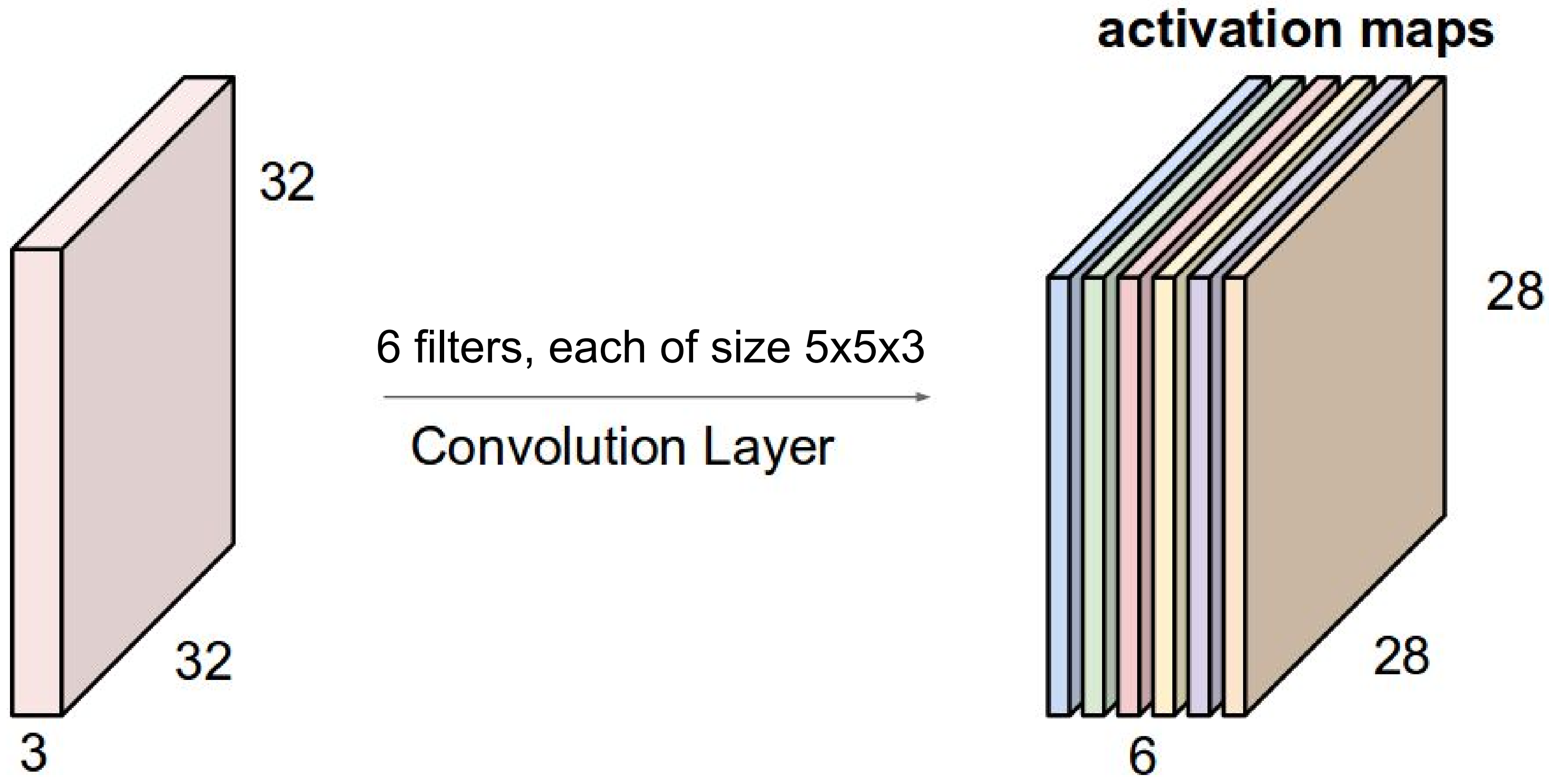
$$w^T x + b$$

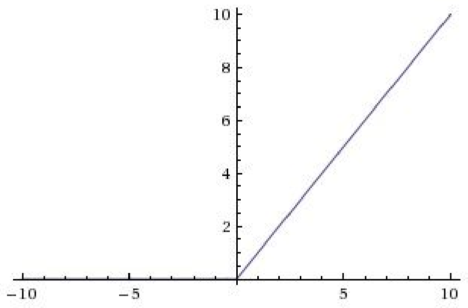
Convolution



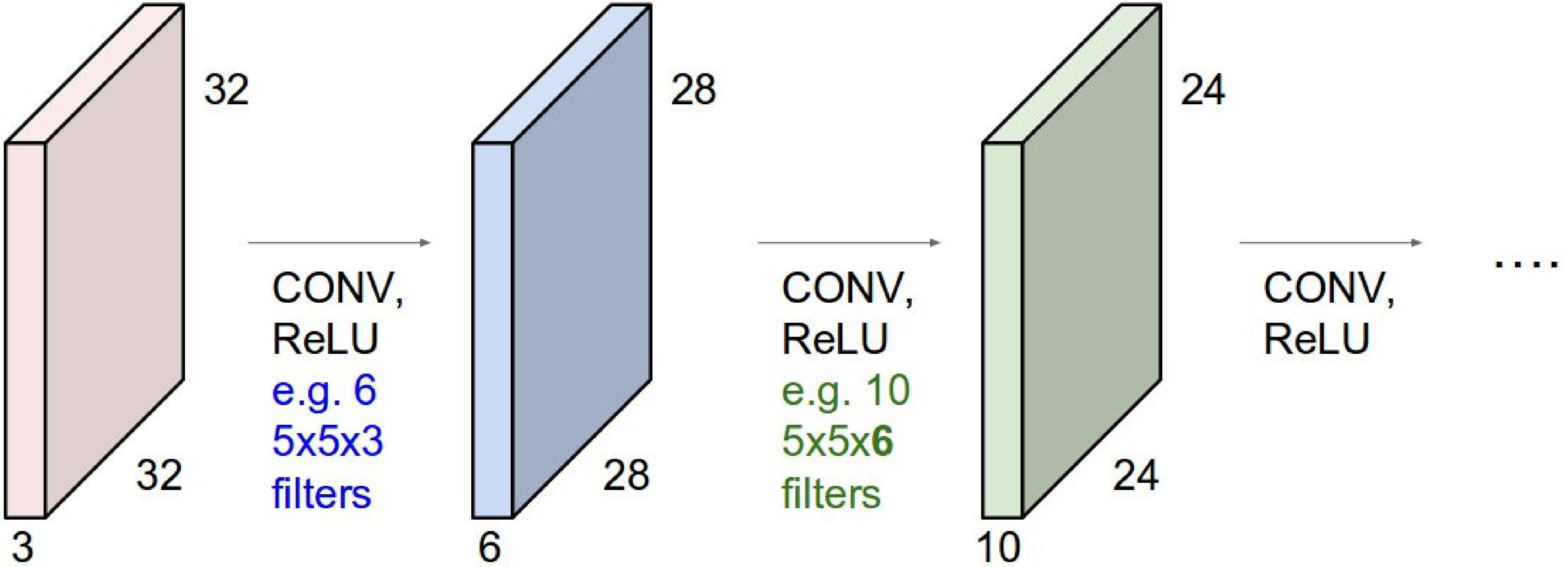
$$w^T x + b$$

Convolution Layer

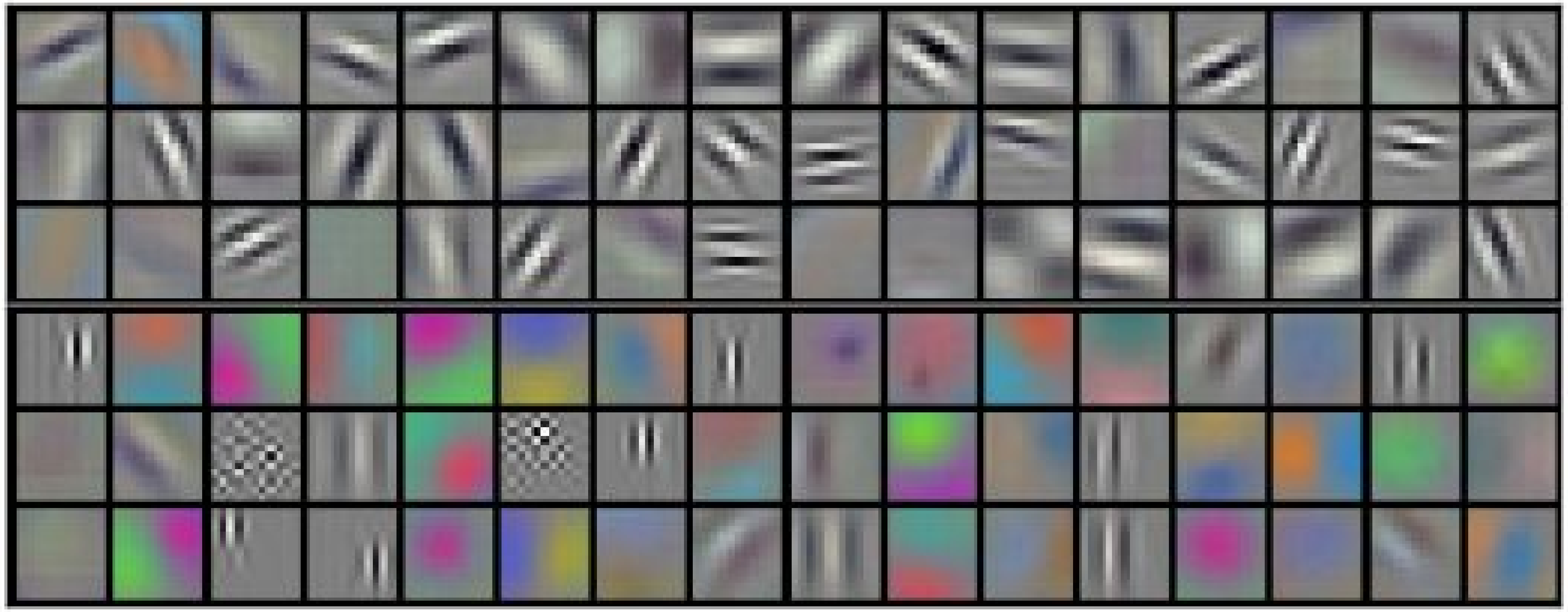


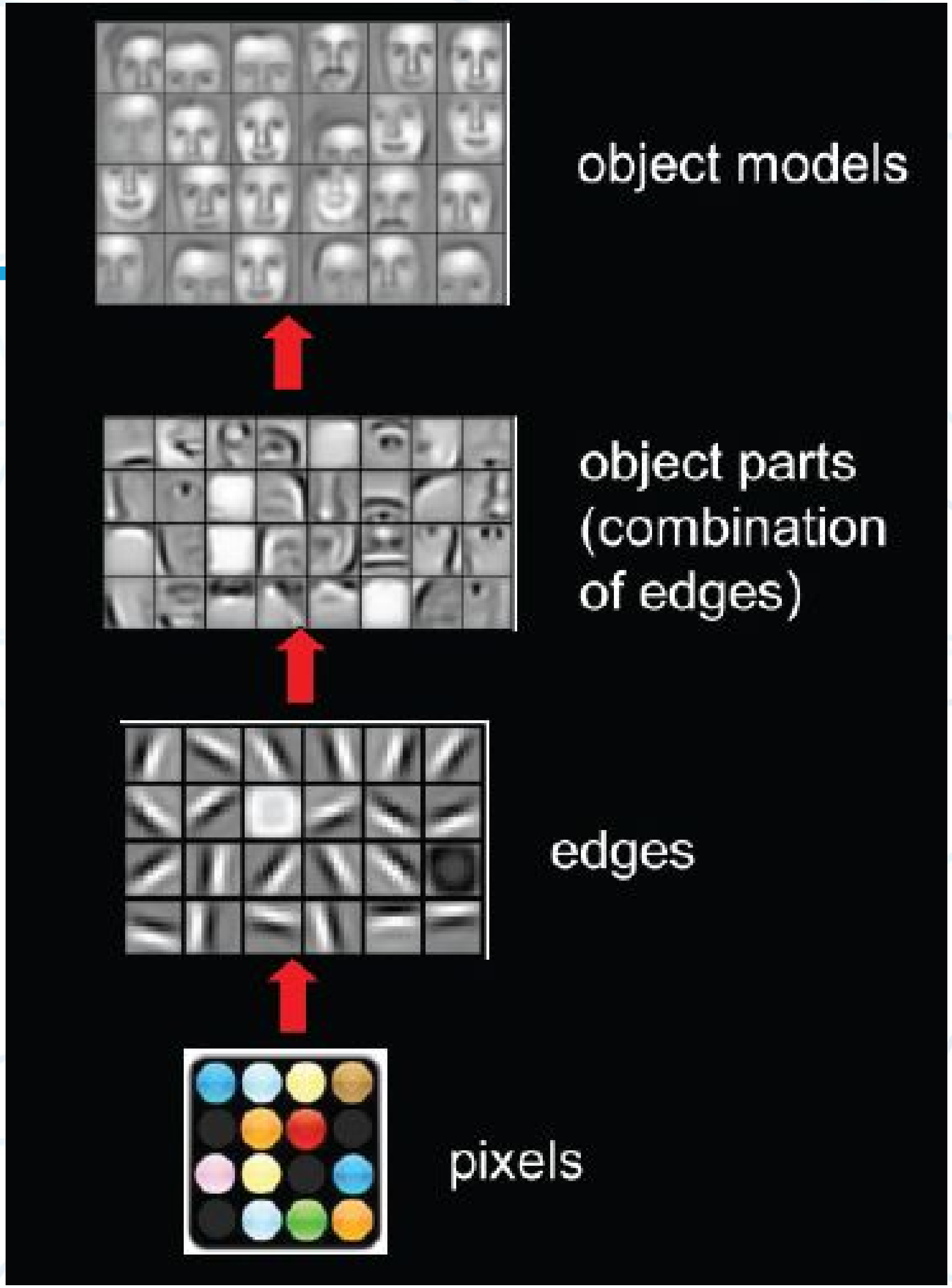


ConvNet

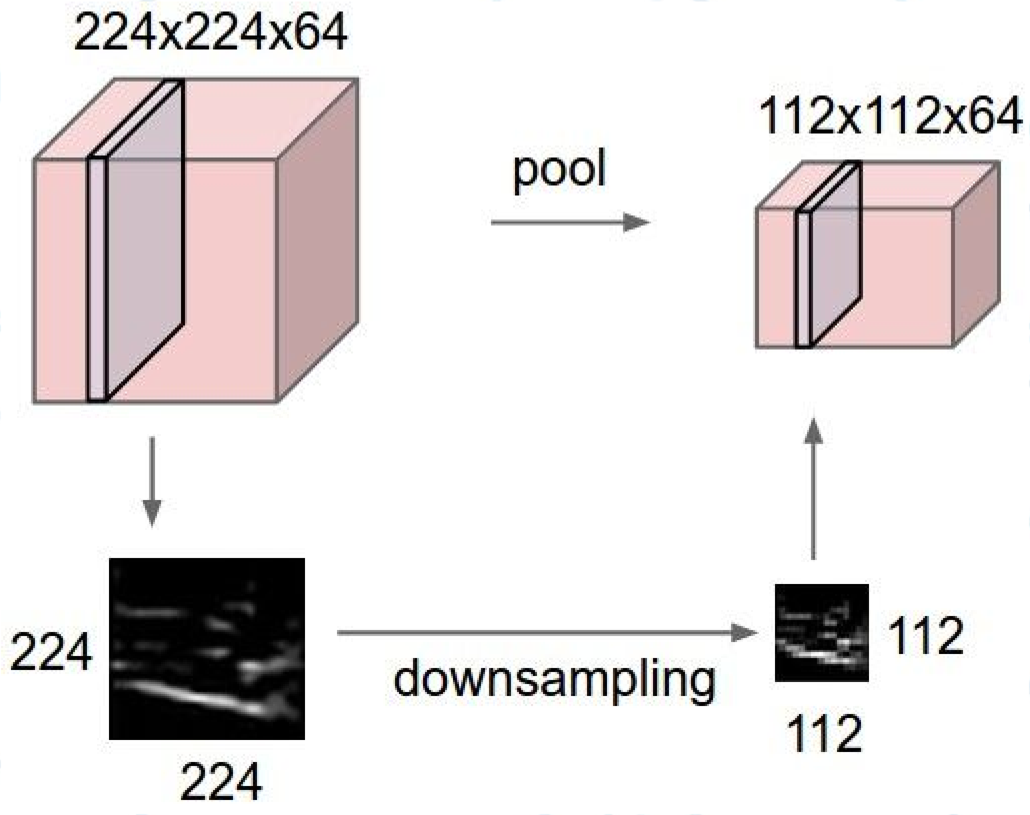


32
32
3
CONV,
ReLU
e.g. 6
5x5x3
filters
28
28
6
CONV,
ReLU
e.g. 10
5x5x**6**
filters
24
24
10
CONV,
ReLU
....

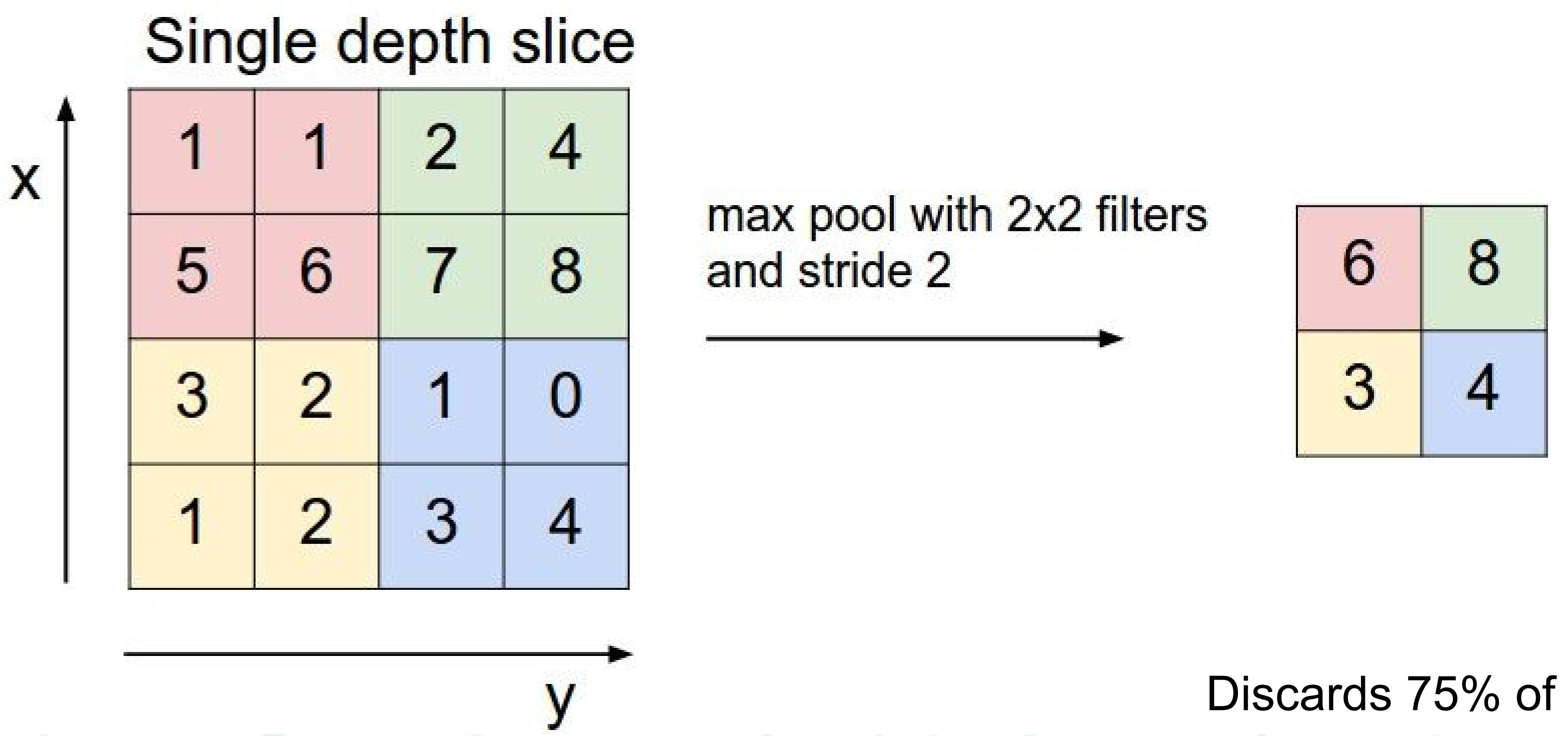



object models
object parts
(combination
of edges)
edges
pixels

Pooling



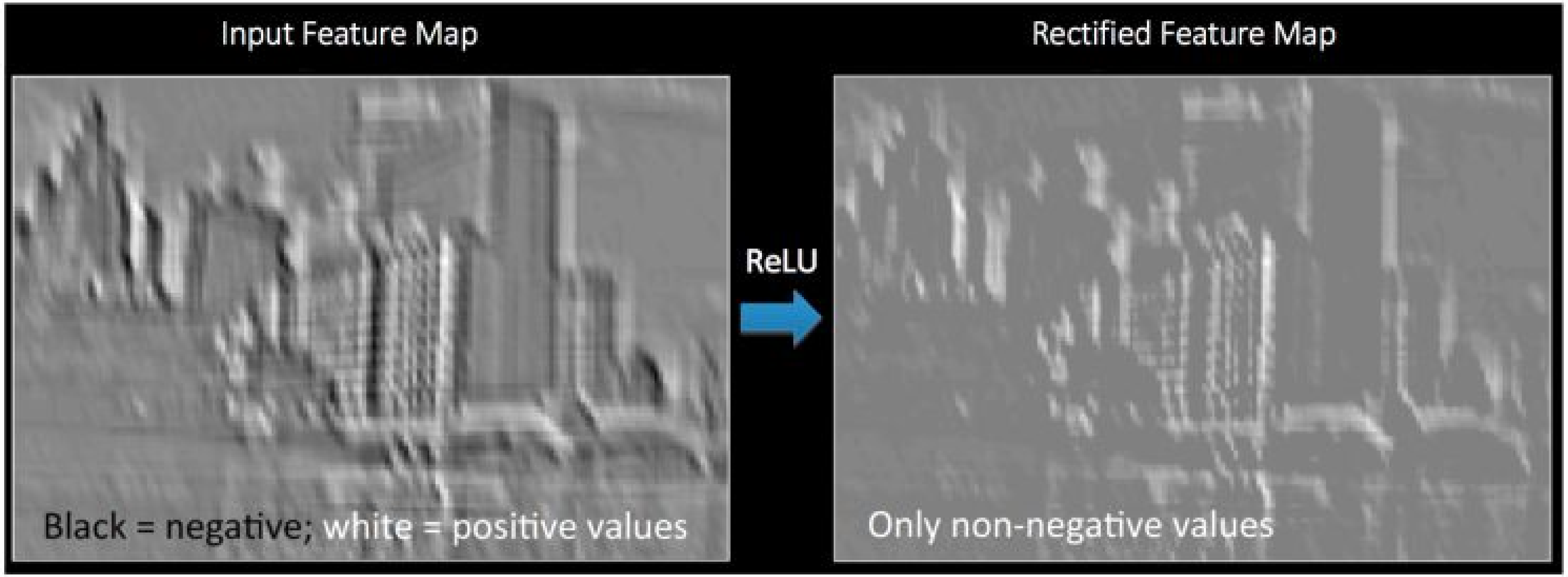
Pooling



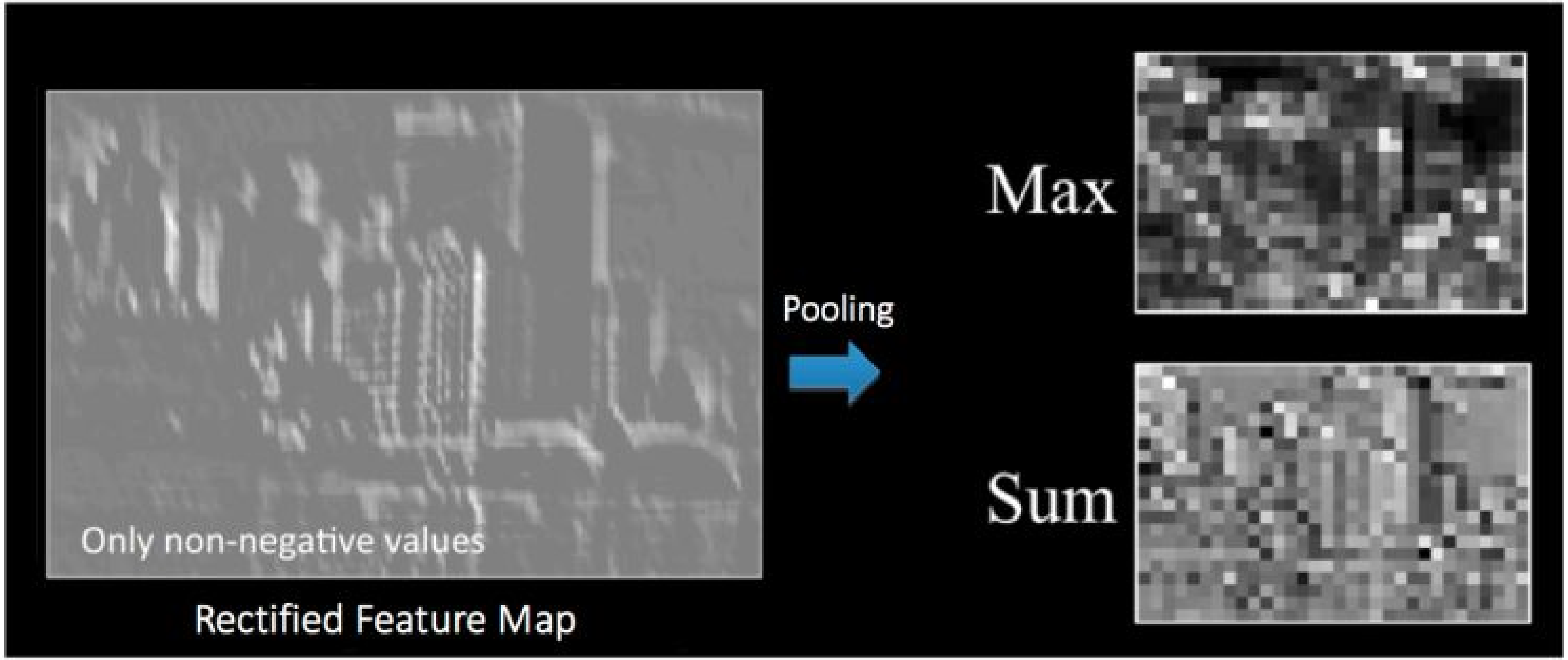
Discards 75% of info!

Convolution

Nonlinearity

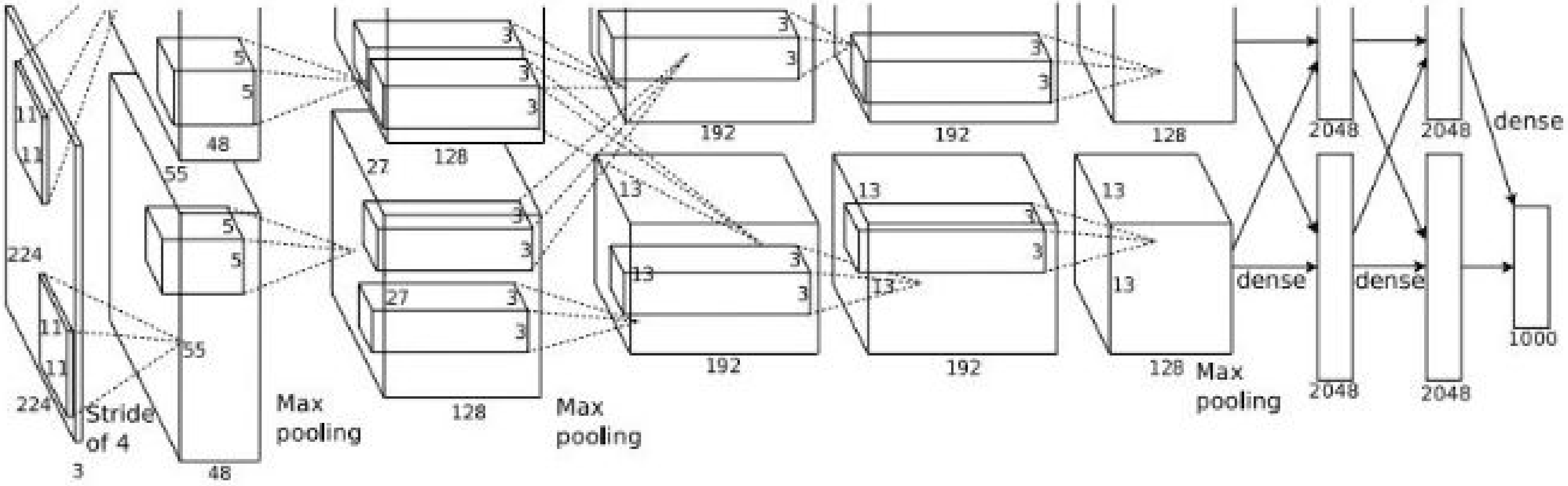


Pooling



Only non-negative values
Rectified Feature Map
Pooling
Max
Sum

AlexNet

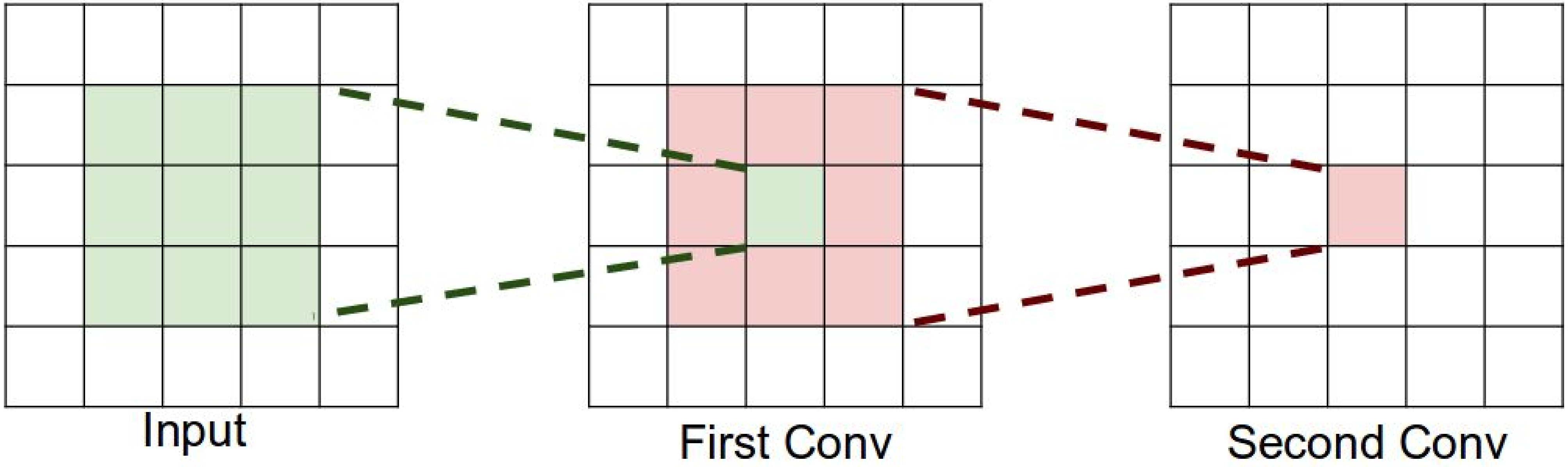


11
11
224
11
11
224
Stride of 4
3
55
55
48
48
5
5
5
5
Max pooling
27
27
128
128
3
3
3
3
Max pooling
13
13
192
192
13
13
13
192
192
13
13
128
128
Max pooling
dense
dense
dense
2048
2048
2048
2048
1000

VGG-16

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					

Smaller filters



Smaller filters – what do we gain?

Consider an input with height H, width W, C channels.
Assume stride=1, padding such that H and W do not change on the output.

A single 7x7 filter:

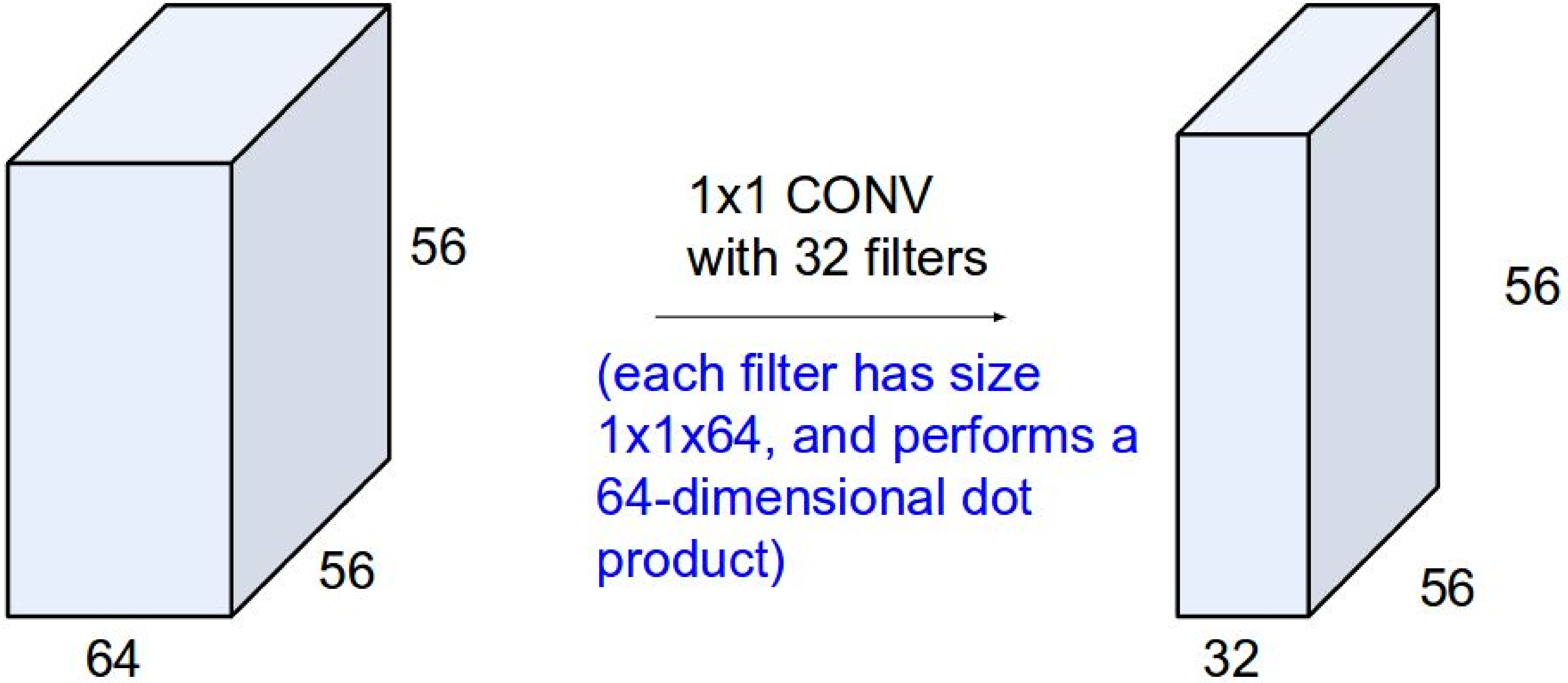
- parameters: C*(7*7*C) = $49C^2$ (and a bias)
- operations: (H*W*C) * (7*7*C) = $49\ HWC^2$

A stack of three 3x3 filters (same receptive field):

- parameters: 3*C*(3*3*C) = $27C^2$ (and a bias)
- operations: (H*W*C) * 3 * (3*3*C) = $27\ HWC^2$

- Less parameters
- Less computations
- More nonlinearity

1x1 filters

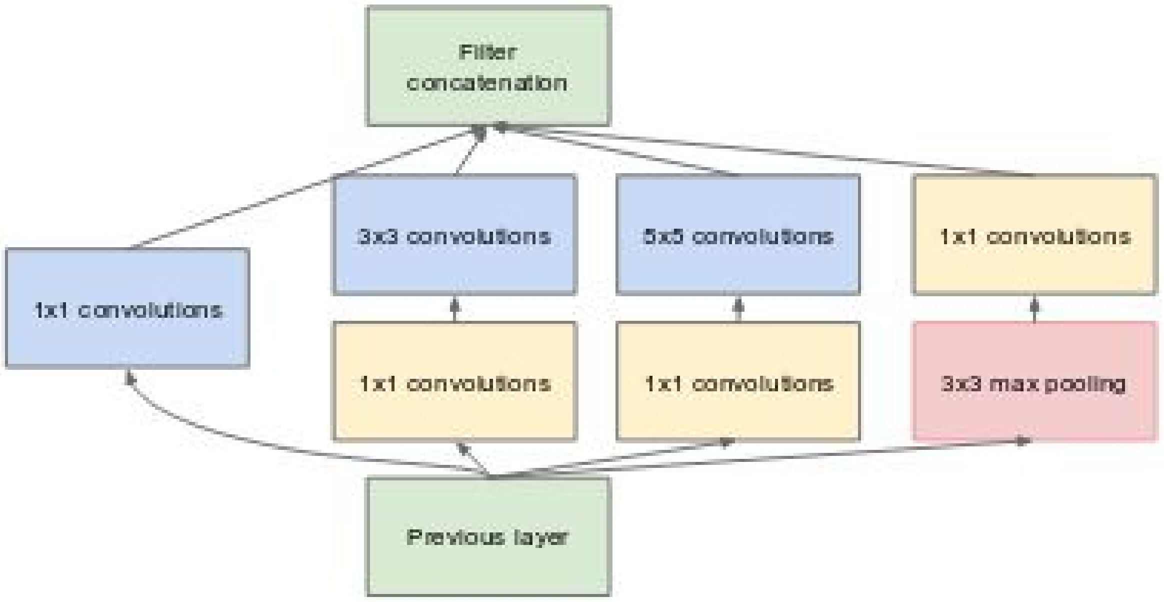


Even less params, less computations, more nonlinearity

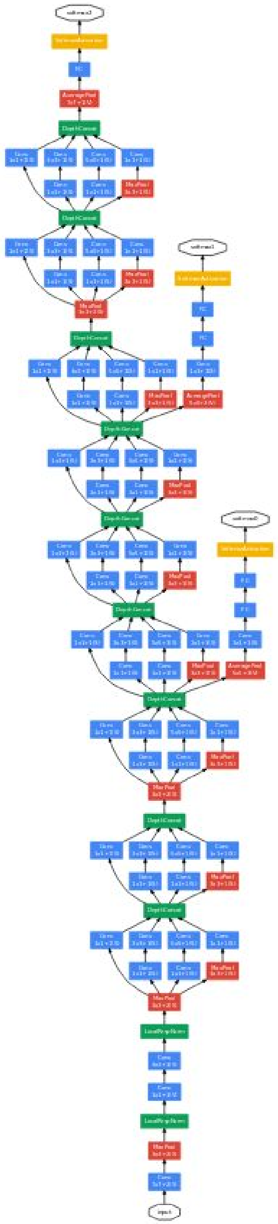
Can be used to increase/decrease number of channels (bottlenecks)

GoogLeNet

https://arxiv.org/abs/1409.4842

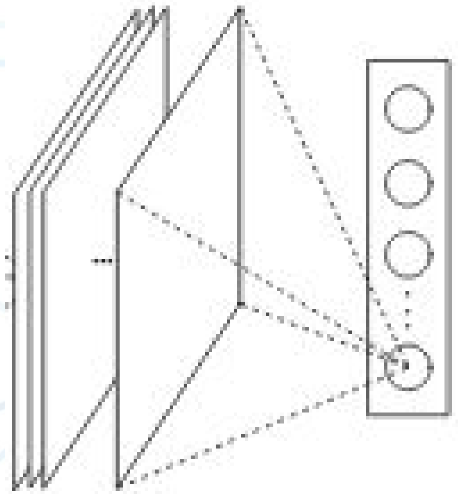


(b) Inception module with dimensionality reduction



Global pooling

First appeared in "Network in Network" (https://arxiv.org/pdf/1312.4400.pdf)
Facilitates the use of all conv networks (ConvNet can work with all input resolutions!)



ResNet

https://arxiv.org/abs/1512.03385

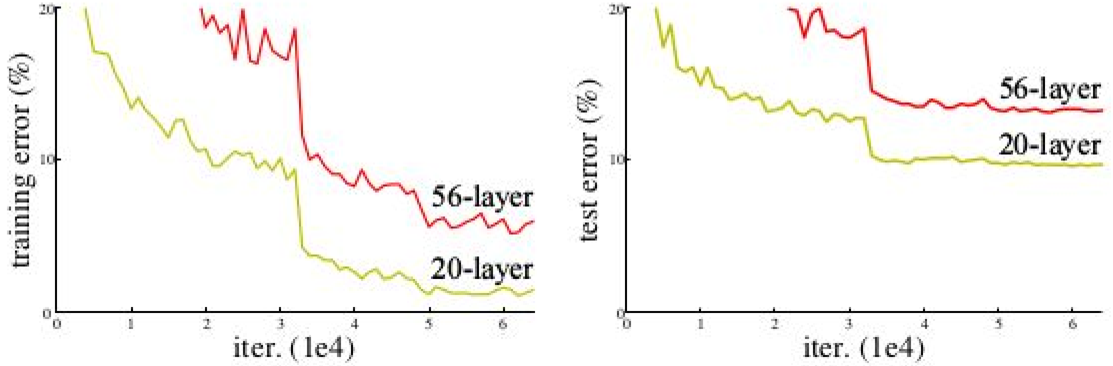
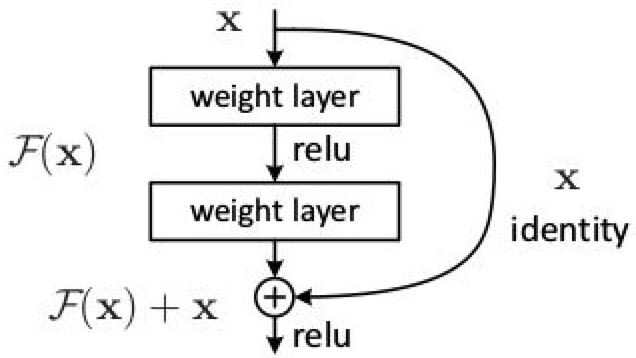
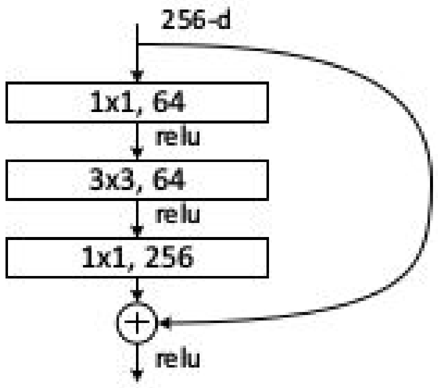


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

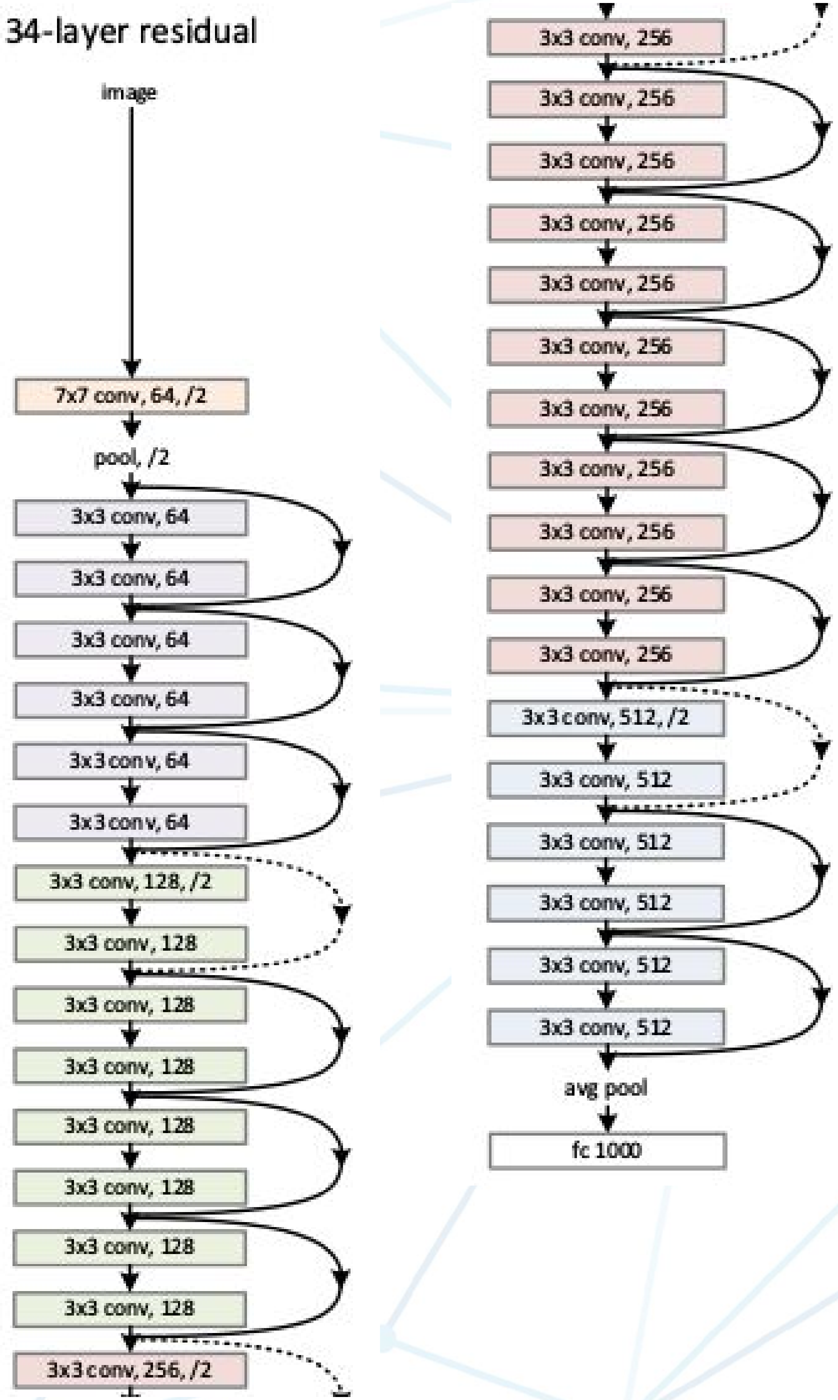


ResNet

https://arxiv.org/abs/1512.03385

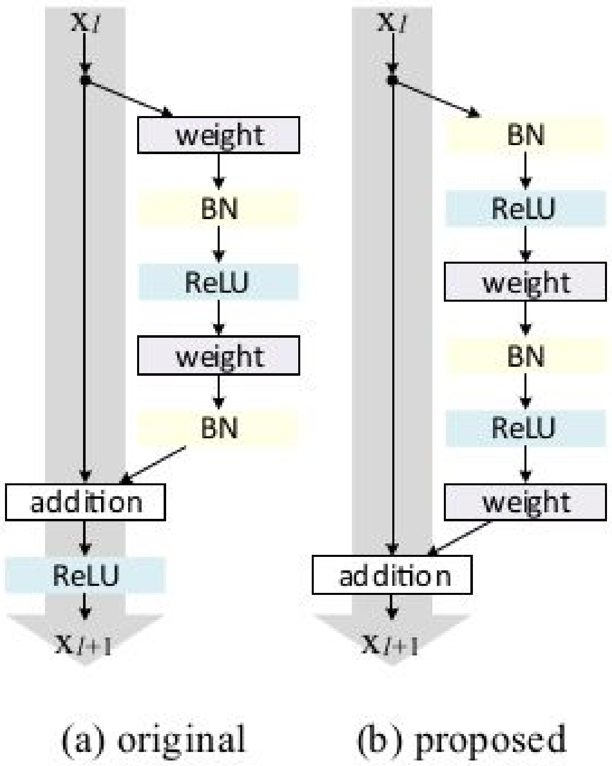


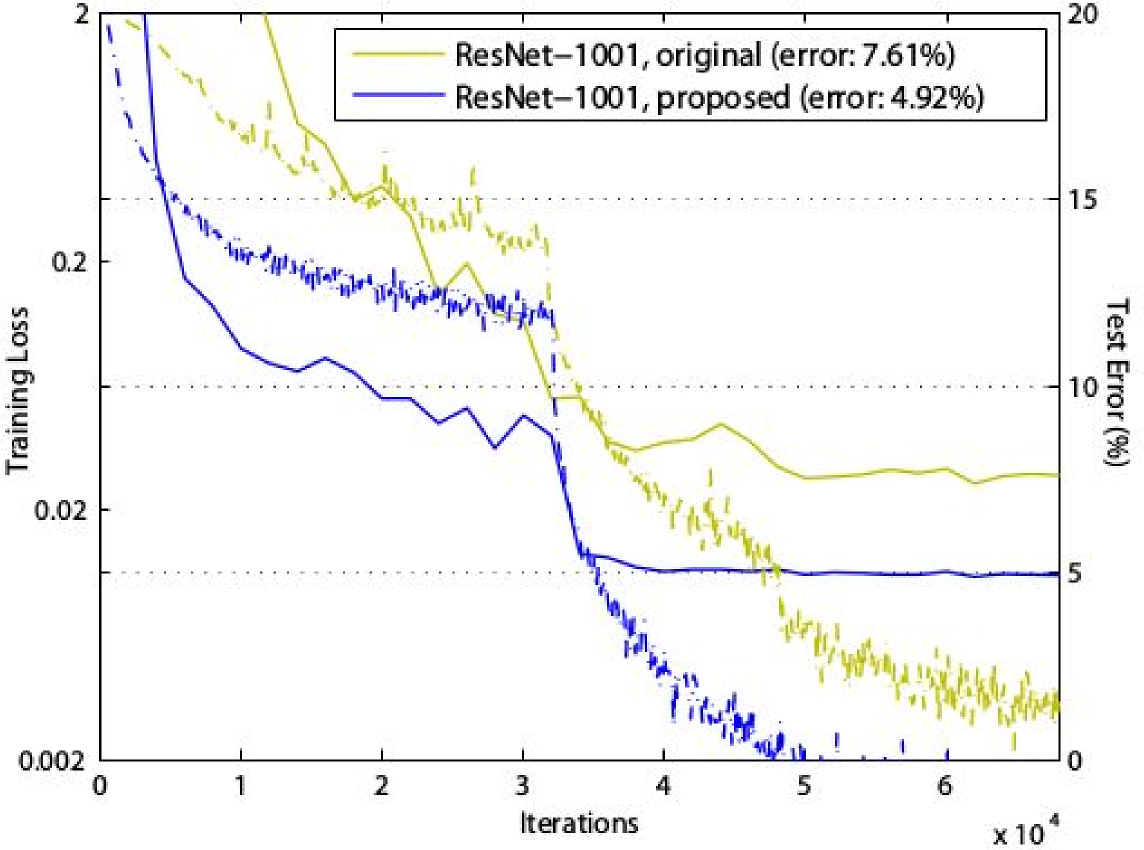
“Bottleneck” building block



ResNet v2

https://arxiv.org/abs/1603.05027





WideResNet

https://arxiv.org/abs/1605.07146

	depth-k	# params	CIFAR-10	CIFAR-100
NIN [20]			8.81	35.67
DSN [19]			8.22	34.57
FitNet [24]			8.39	35.04
Highway [28]			7.72	32.39
ELU [5]			6.55	24.28
original-ResNet[11]	110	1.7M	6.43	25.16
	1202	10.2M	7.93	27.82
stoc-depth[14]	110	1.7M	5.23	24.58
	1202	10.2M	4.91	-
pre-act-ResNet[13]	110	1.7M	6.37	-
	164	1.7M	5.46	24.33
	1001	10.2M	4.92(4.64)	22.71
WRN (ours)	40-4	8.9M	4.53	21.18
	16-8	11.0M	4.27	20.43
	28-10	36.5M	**4.00**	**19.25**

Model	top-1 err, %	top-5 err, %	#params	time/batch 16
ResNet-50	24.01	7.02	25.6M	49
ResNet-101	22.44	6.21	44.5M	82
ResNet-152	22.16	6.16	60.2M	115
WRN-50-2-bottleneck	21.9	6.03	68.9M	93
pre-ResNet-200	21.66	5.79	64.7M	154

Simple tips

ConvNets need lots of data

Use data augmentation or transfer learning

Lots of parameters = overfitting

Regularize! L2, dropout, BatchNorm

Use residual blocks and bottleneck convs (e.g. 1x1 -> 3x3 -> 1x1)

Global pooling prior to softmax (or any other dense layer)

Gives a possibility to process images of arbitrary resolution

We recommend using Adam as a state-of-the-art optimization algorithm

Transfer learning

Get a model trained on a similar dataset.

Got "lots" of data?

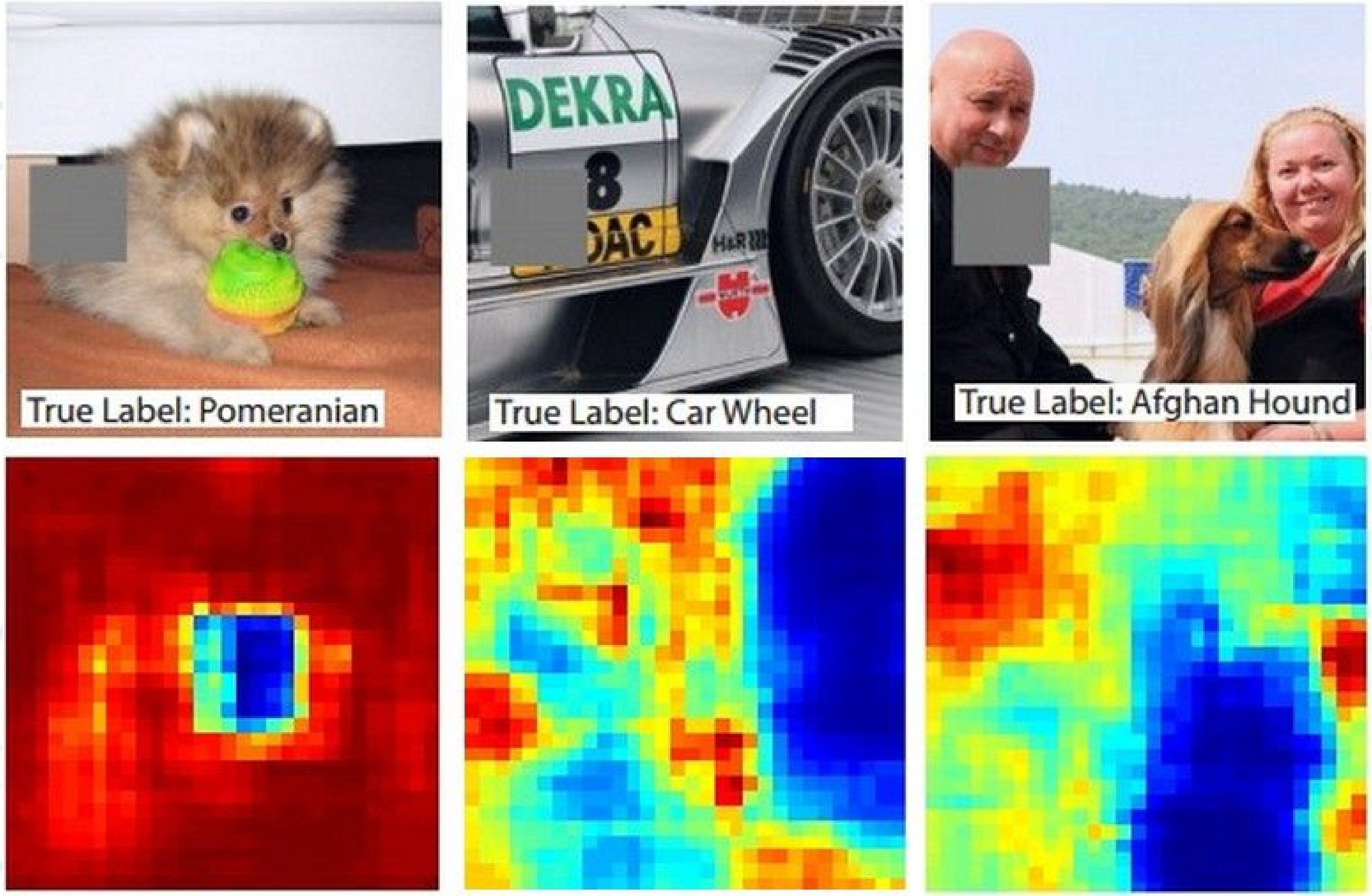
Yes!

Fine-tune last few layers, freeze the rest

No…

Train a (simple) classifier on network output and/or on features (i.e. filter maps) from different layers

Debugging


True Label: Pomeranian
DEKRA
True Label: Car Wheel
True Label: Afghan Hound

FORNAX

Rafał Cycoń
Co-founder, CTO

+48 607 697 054
rafal.cycon@fornax.co

WWW.FORNAX.CO