We are starting at 14:00!

Grab a seat and get ready

Saturdays.AI
Kigali

Saturdays.AI
Kigali

#7 Modern CNNs

AI Saturdays Kigali

Agenda

14:00 - 14:15: Recap: convolution

14:15 - 14:45: AlexNet & VGG

14:45 - 15:15: Going deeper

15:15 - 15:30: Improving efficiency: part 1

15:30 - 16:00: Improving efficiency: part 2

16:00 - 16:30: Break

16:30 - 17:00: Transfer learning

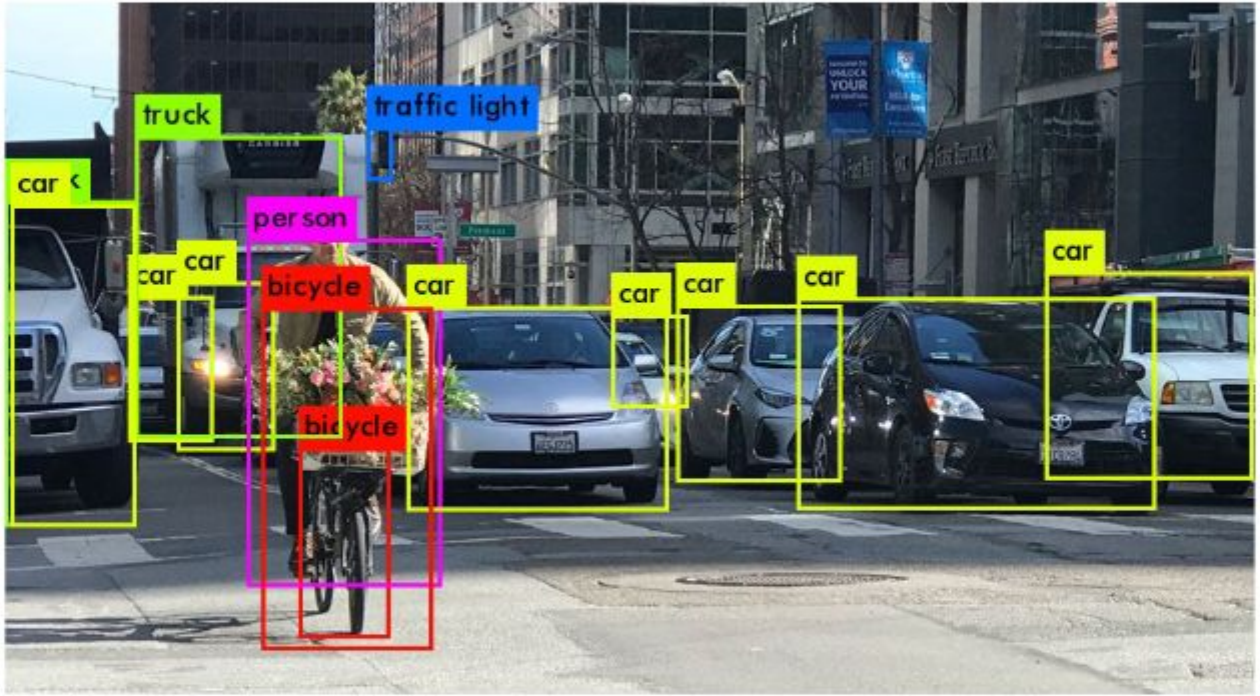
17:30 - 18:00: Challenges & Next steps

Last week: You learned to build a CNN

- Convolution
- Pooling/downsampling
- Training

Saturdays.AI
Kigali

Today: CNNs “in the real world”



Saturdays.AI
Kigali

CNNs are everywhere: image segmentation

Saturdays.AI
Kigali

CNNs are everywhere: pose estimation

Saturdays.AI
Kigali

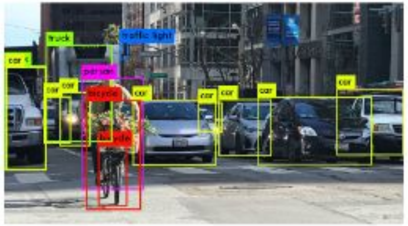
CNNs are everywhere: image style transfer

Saturdays.AI
Kigali

Today: Large-scale CNNs & transfer learning

The success of deep learning for image recognition has been driven by two key factors:

1. Large-scale CNNs
2. Transfer learning



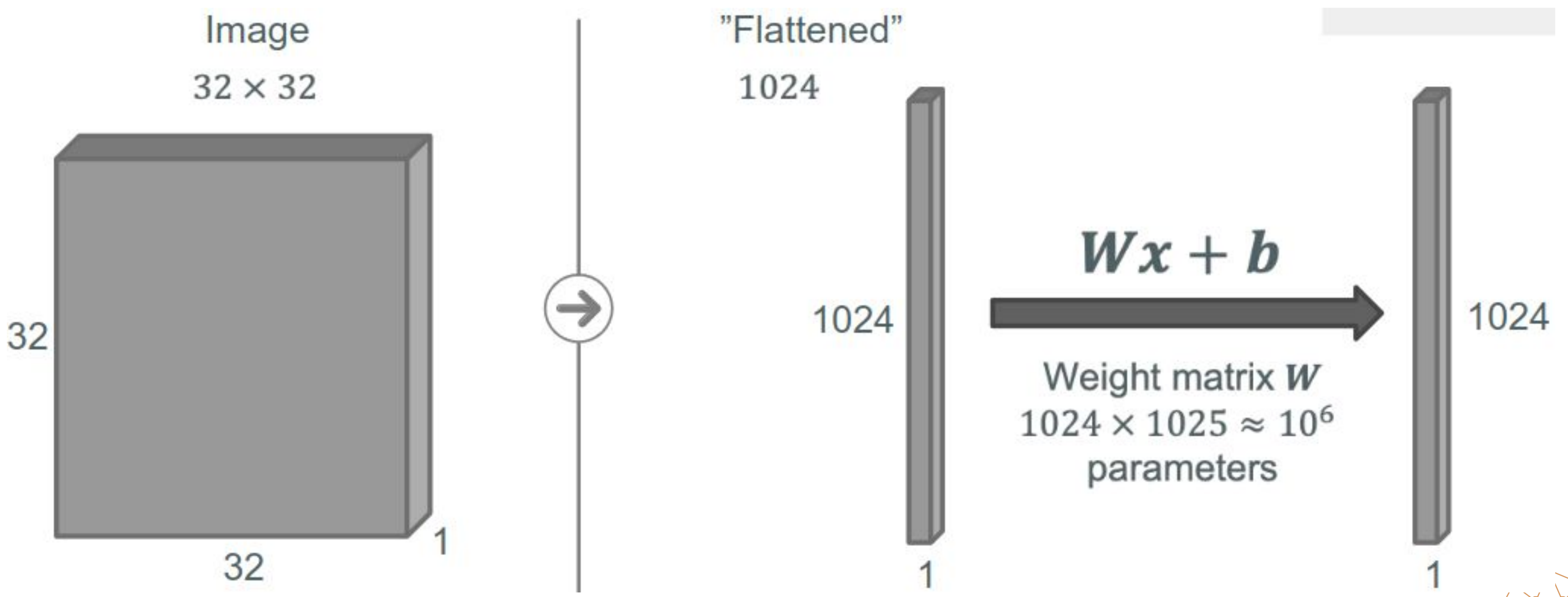
c

Saturdays.AI
Kigali

Recap: convolution

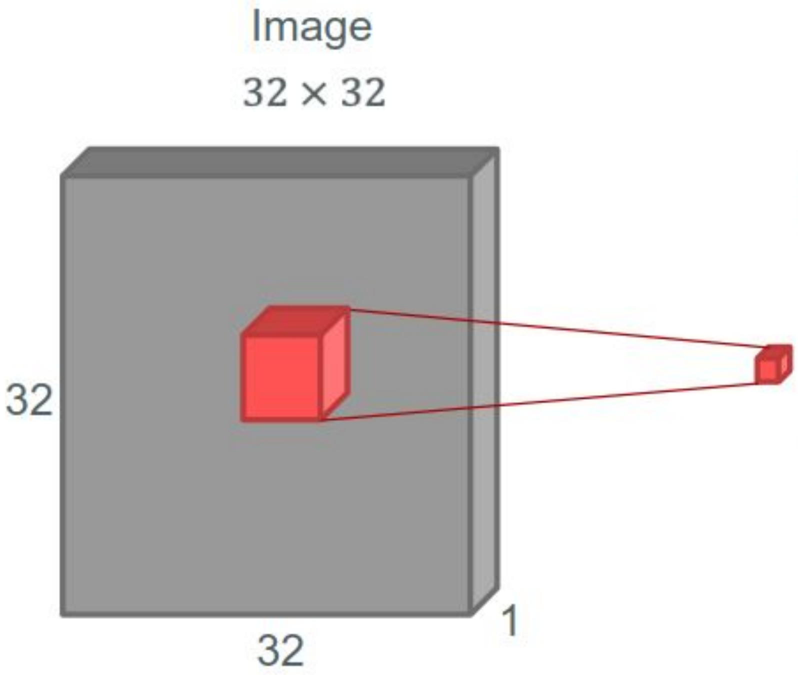
Saturdays.AI
Kigali

32 x 32 grayscale image: fully-connected layer



Saturdays.AI
Kigali

Convolution



Convolve the filter with the image:
Slide over the image, computing dot products

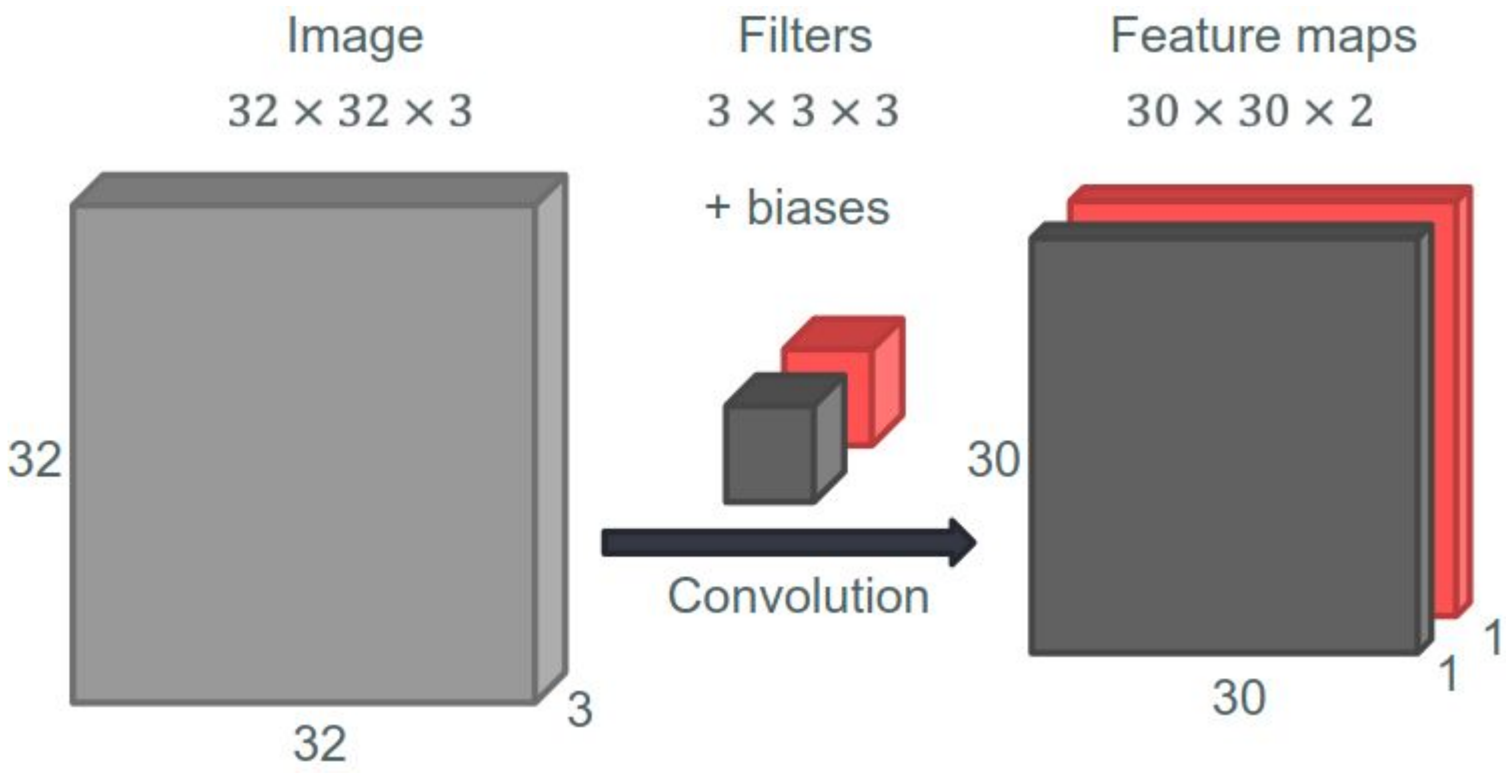
1 number: $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

The result is the inner product of

- a local 3×3 chunk $\boldsymbol{x}$ of the image
- a 3×3 filter $\boldsymbol{w}$
- plus a bias b

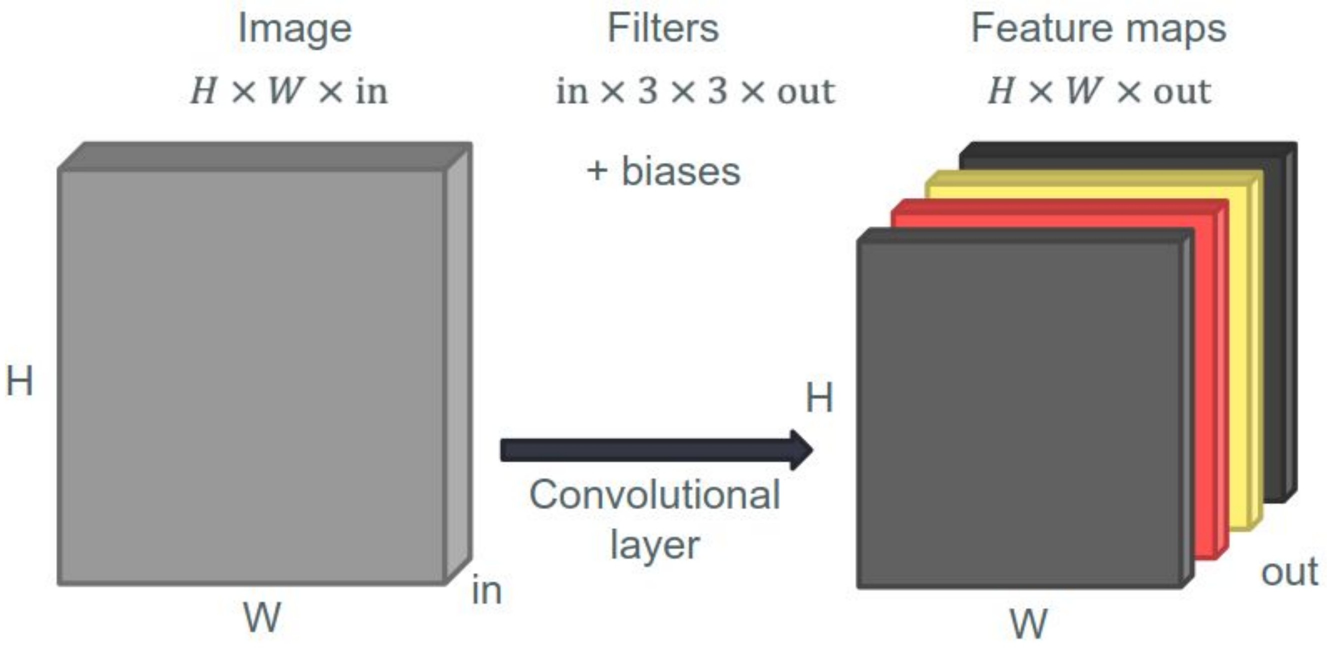
Saturdays.AI
Kigali

Convolution: color image



Saturdays.AI
Kigali

Convolution: in general



"Stack up" feature maps to get a new $H \times W \times \mathrm{out}$ "image"

Saturdays.AI Kigali

Practice

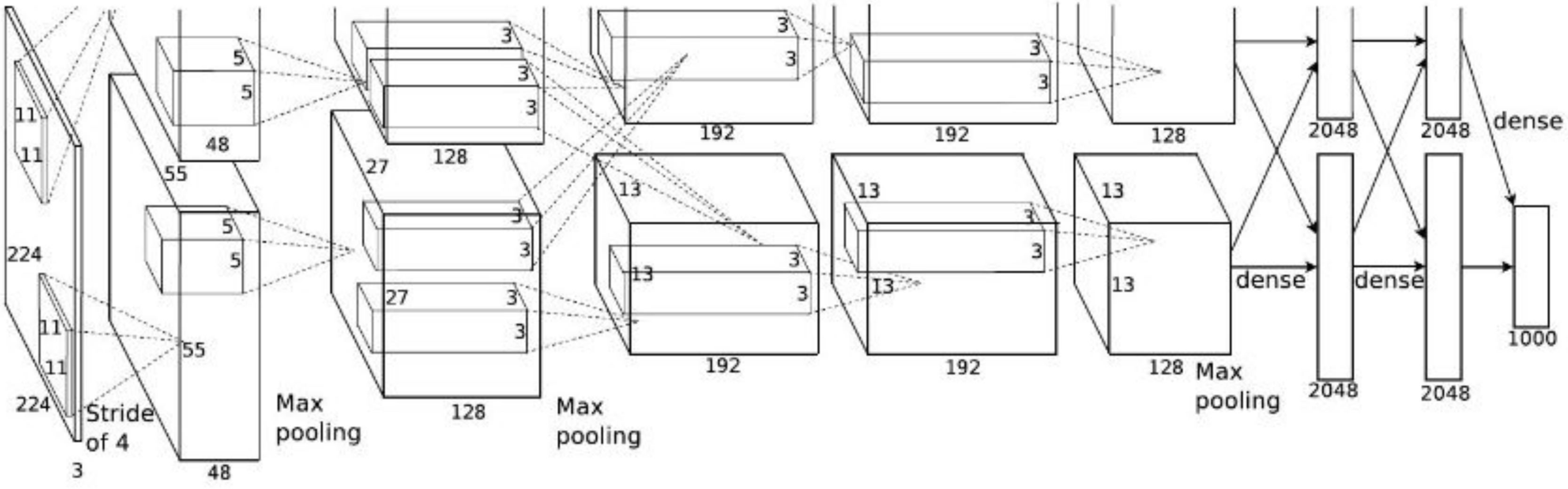
Recap: Recap how much convolutions reduce the number of parameters. Count parameters of a fully-connected and a convnet

Coding Exercise 1

AlexNet & VGG

Saturdays.AI
Kigali

AlexNet (2012)

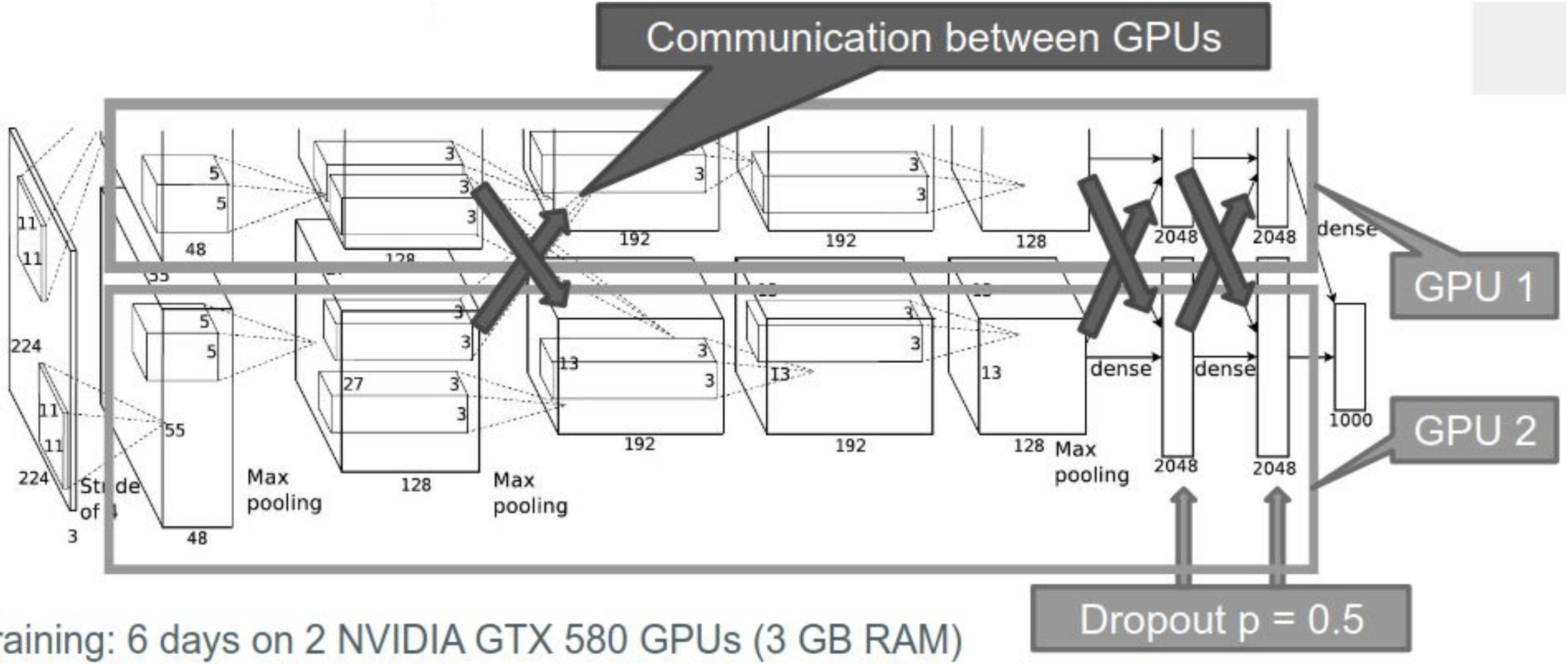


Alex Krizhevsky, Ilya Sutskeever, Geoffrey Hinton

“ImageNet classification with deep convolutional neural networks.” NeurIPS 2012

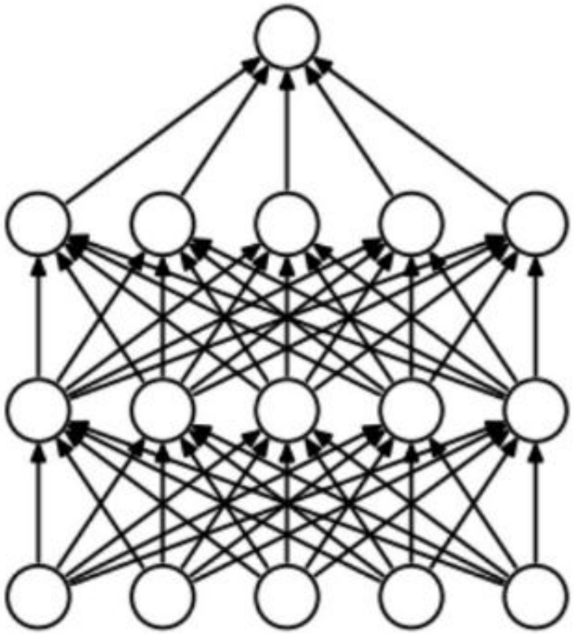
AlexNet (2012)

```
torchvision.models.alexnet()
```

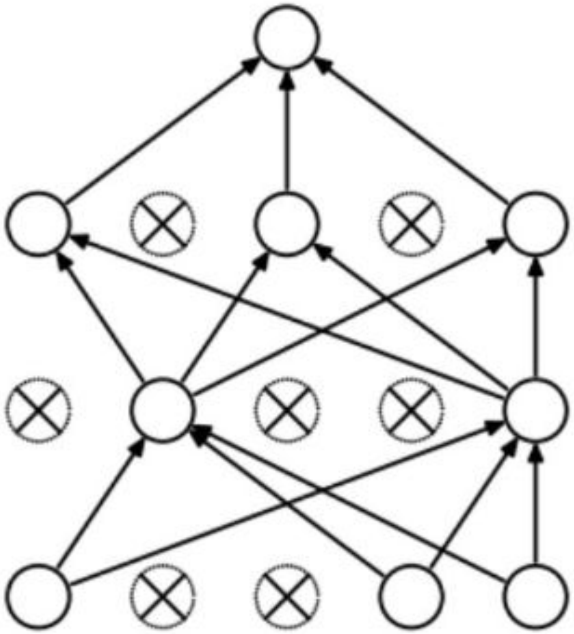


Training: 6 days on 2 NVIDIA GTX 580 GPUs (3 GB RAM)
8 layers / 60 million parameters

Dropout stochastically deactivates neurons



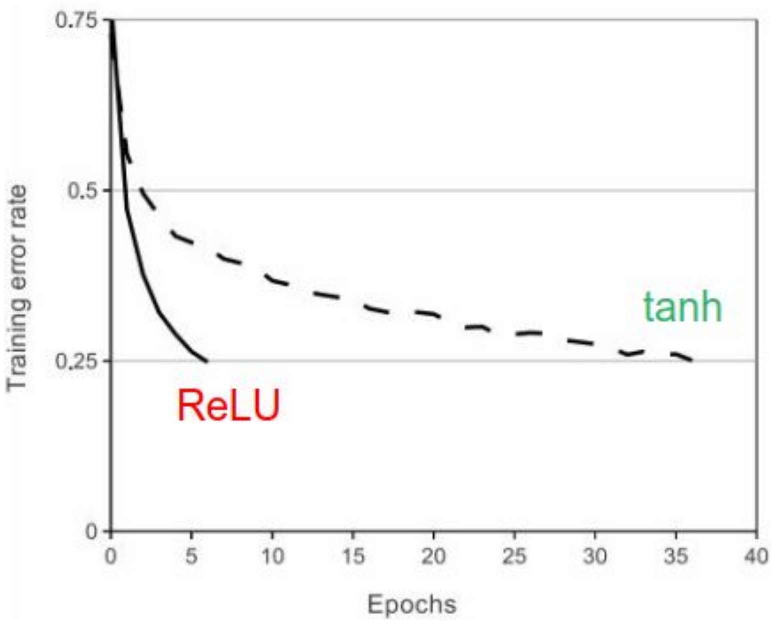
(a) Standard Neural Net

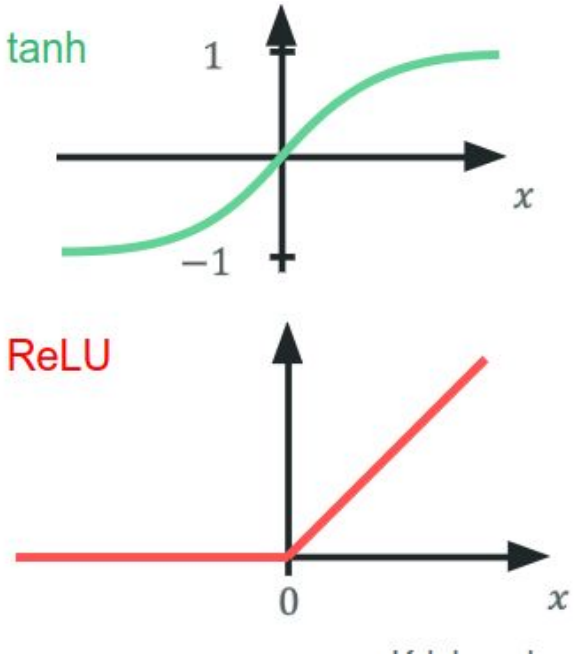


(b) After applying dropout.

Saturdays.AI
Kigali

ReLU trains faster than tanh





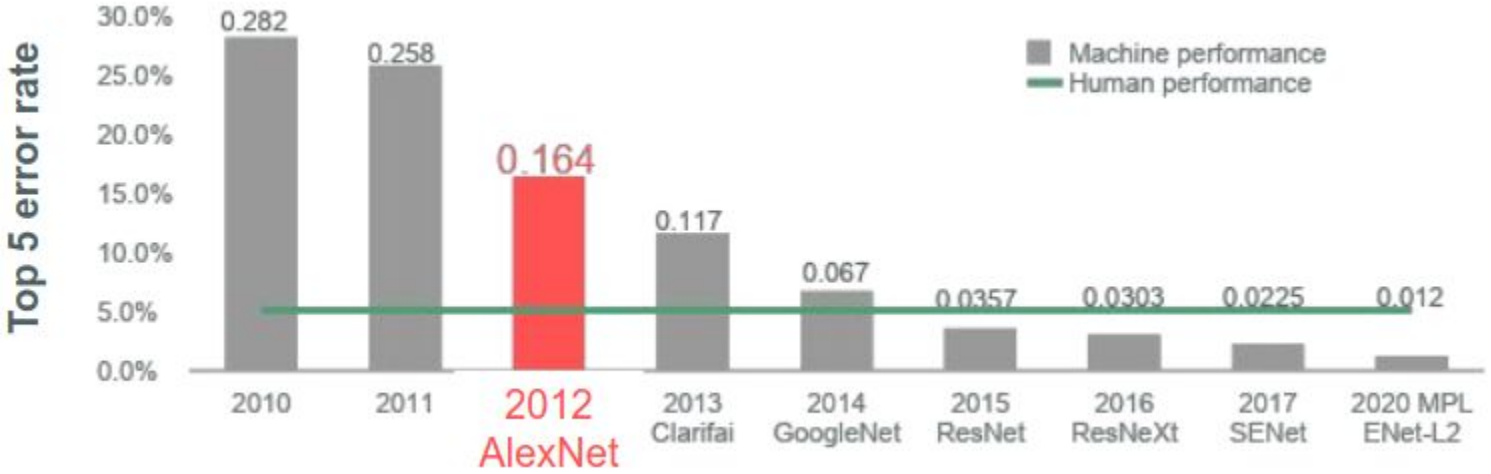
Saturdays.AI
Kigali

AlexNet (2012): Results

Smoked the competition with 16% top-5 error (runner-up had 26%)

One of the first neural nets trained on a GPU with CUDA.

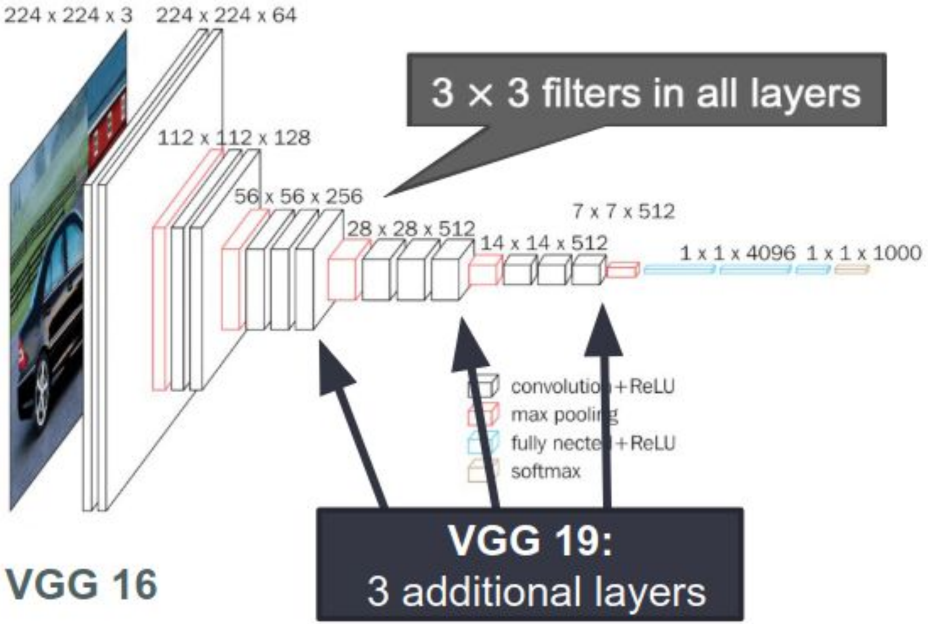
Paper cited over 80,000 times



Saturdays.AI
Kigali

VGG (2014)

```
torchvision.models.vgg16()
torchvision.models.vgg19()
torchvision.models.vgg19_bn()
...
```



By the Vision Geometry Group at Oxford

Only 3 × 3 filters and max pooling

Training: 3 weeks on 4 NVIDIA Titan Black GPUs, 6 GB RAM

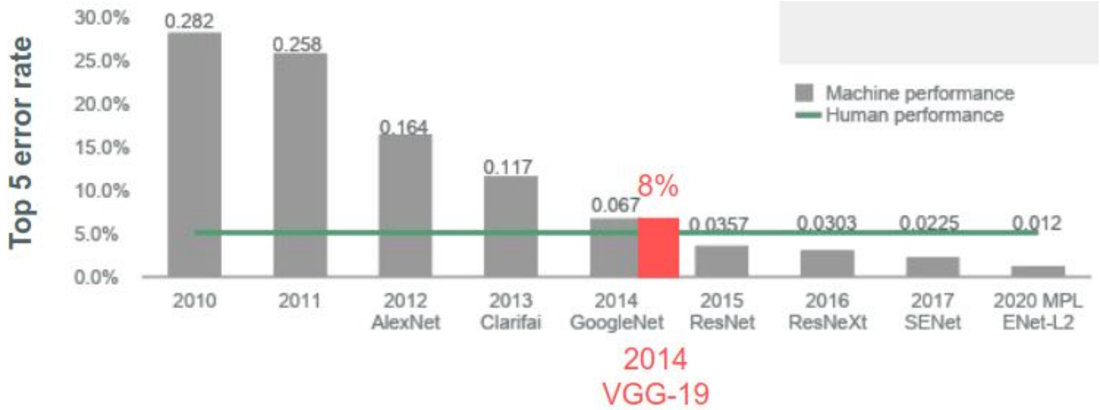
First train smaller configurations, then inject layers in between:
11 □ 13 □ 16 □ 19 layers

VGG-19: 138 million parameters / 500 MB

VGG (2014)

Has been very popular
for a long time

- Particularly simple architecture
- Half the error rate of AlexNet
- Pre-trained net available



A couple of drawbacks:

- Slow and complex to train. Started with 11-layer version, then add additional layers, iterate until at 19
- Weights themselves are large: over 530 MB, which makes deploying it a tiresome task
- First couple of layers are very expensive to compute as they operate at full resolution

Saturdays.AI
Kigali

Practice

Inspect what AlexNet has learned: filters and feature maps

[Interactive Demo 3.2]()

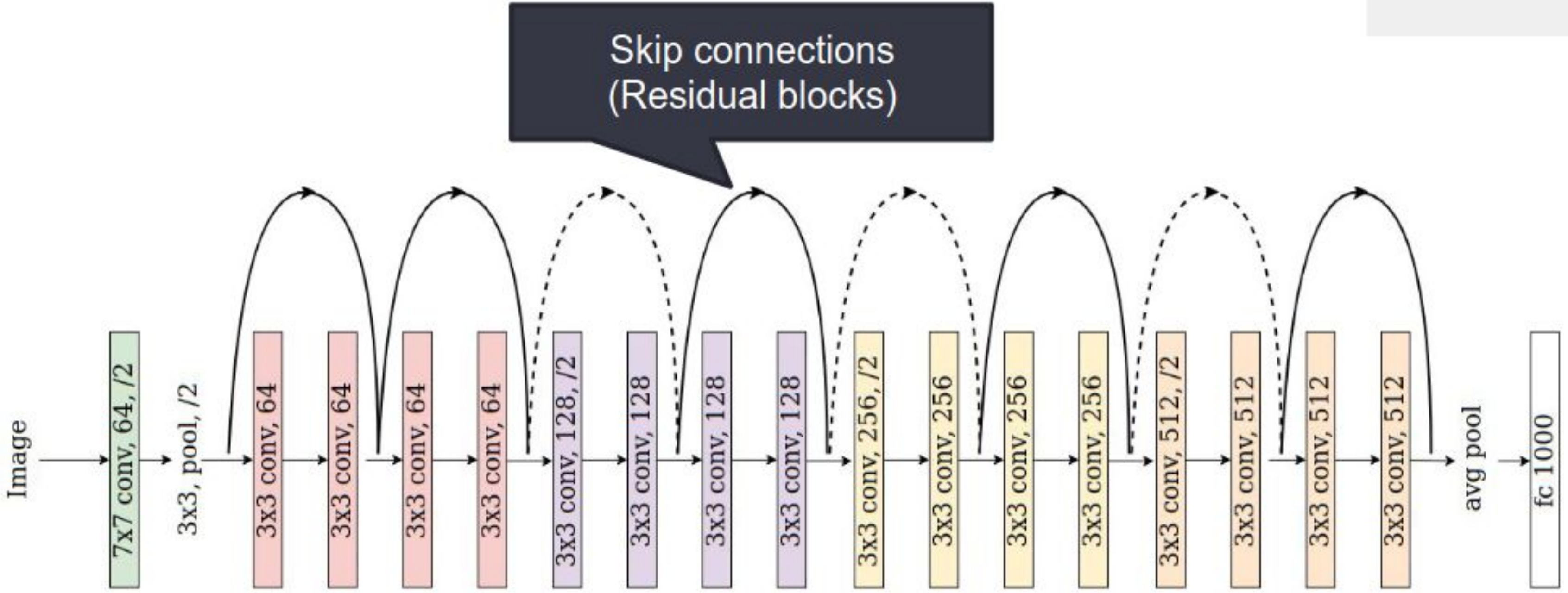
Going deeper

Residual Networks (ResNets)

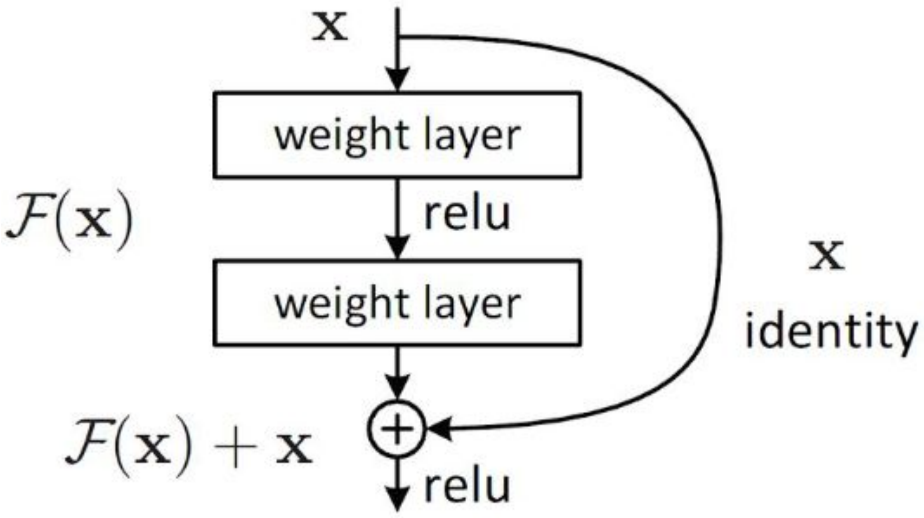

WE NEED TO GO
DEEPER
troll-face.ru

ResNet (2015)

```
torchvision.models.resnet18()
...
torchvision.models.resnet152()
```



Residual blocks ("skip connections")



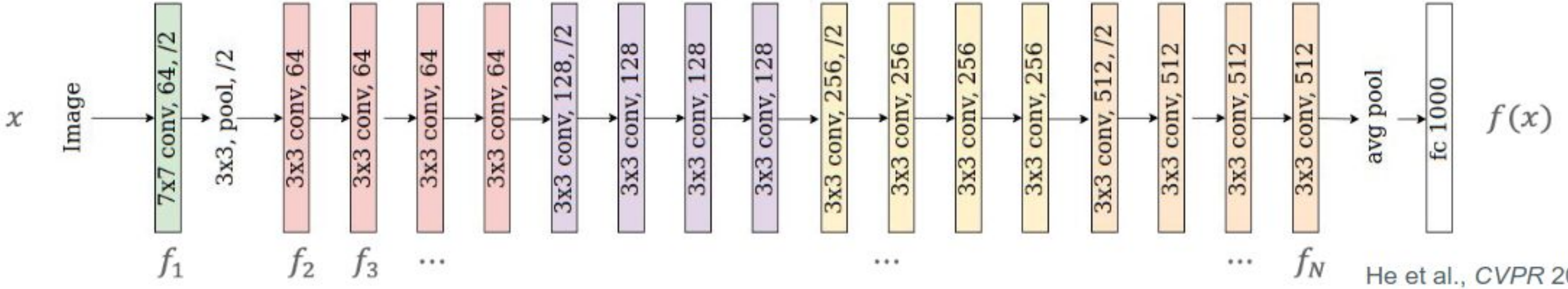
Each block learns to approximate a residual function

Better gradient flow because of skip connections

Saturdays.AI
Kigali

Skip connections avoid vanishing gradients

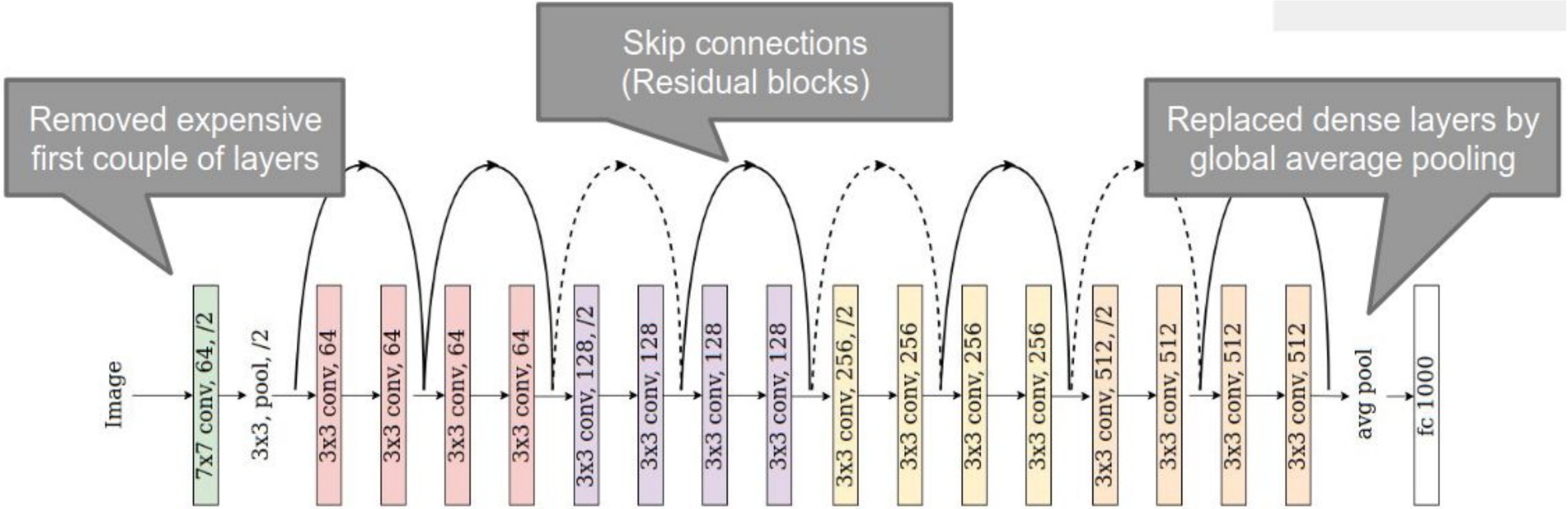
$$f(x) = f_1\left(f_2(\cdots f_N(x))\right) \quad \Rightarrow \quad f'(x) = f_1'(f_2(\cdots)) \cdot f_2'(\cdots) \cdot \ldots \cdot f_N'(x) \qquad \textit{(chain rule)}$$



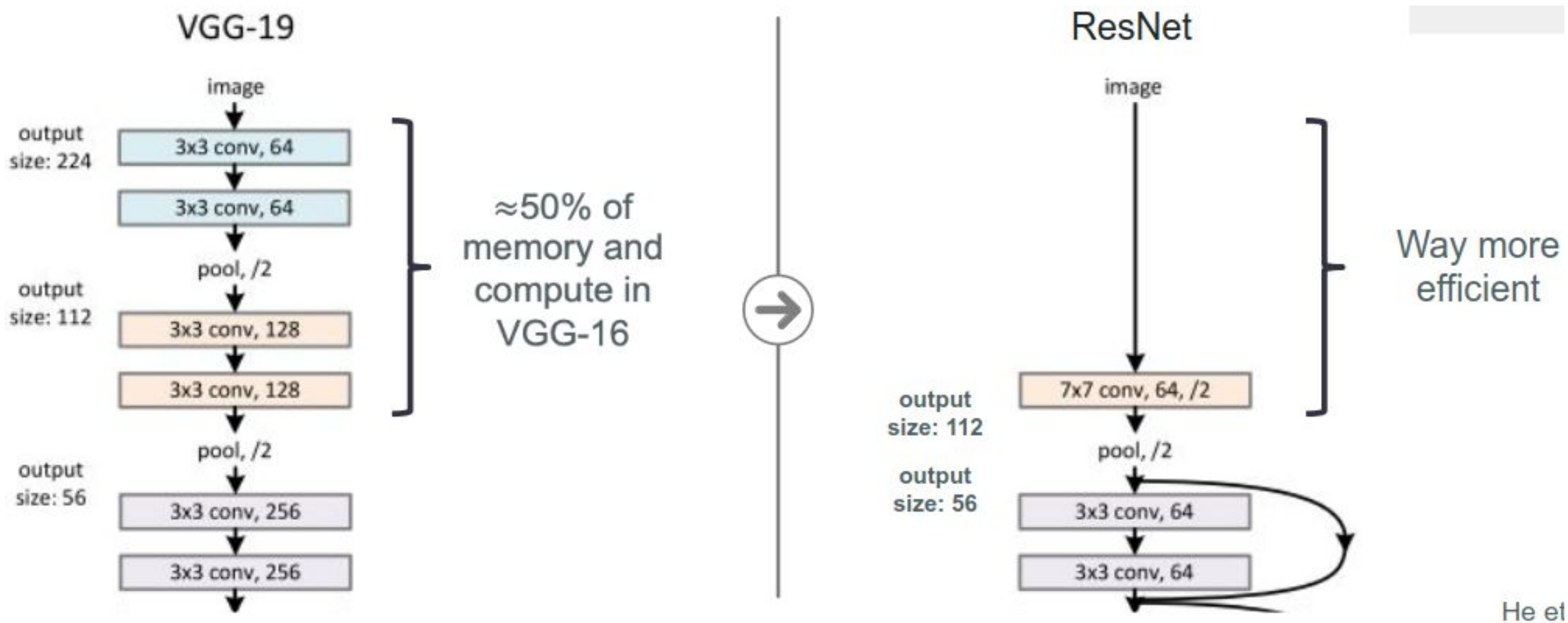
He et al., *CVPR* 2016

ResNet (2015)

```
torchvision.models.resnet18()
...
torchvision.models.resnet152()
```



ResNet: Remove expensive early layers



He et

Saturdays.AI
Kigali

Practice

- Load a pre-trained ResNet-18
- Run a couple of images through the net
- Inspect its predictions

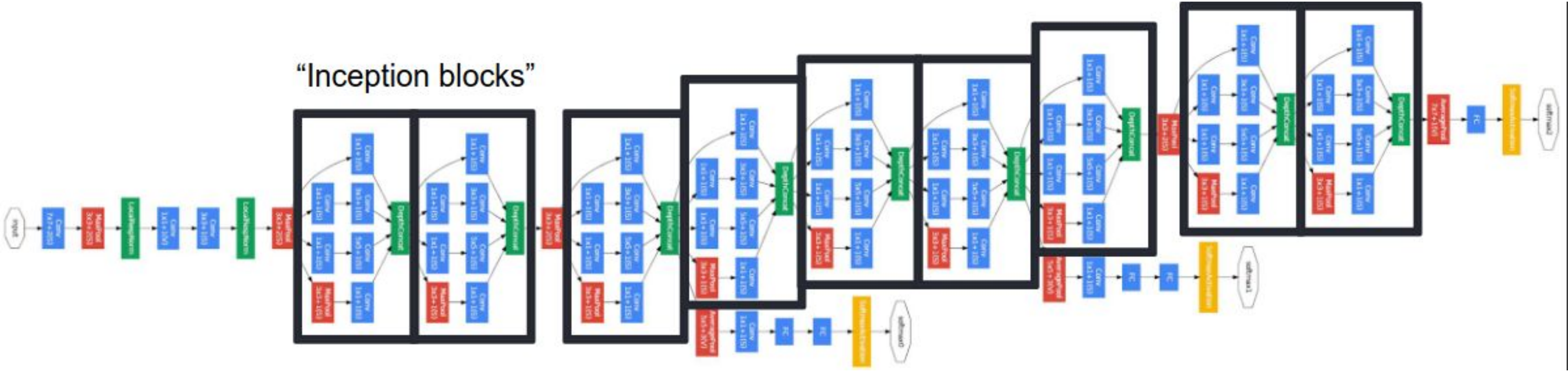
How does it fare with non-photographic images?

[Coding Exercise 4.1]

Improving efficiency

Part 1: Inception + ResNeXt

Inception / GoogLeNet (2014/15)



Saturdays.AI
Kigali

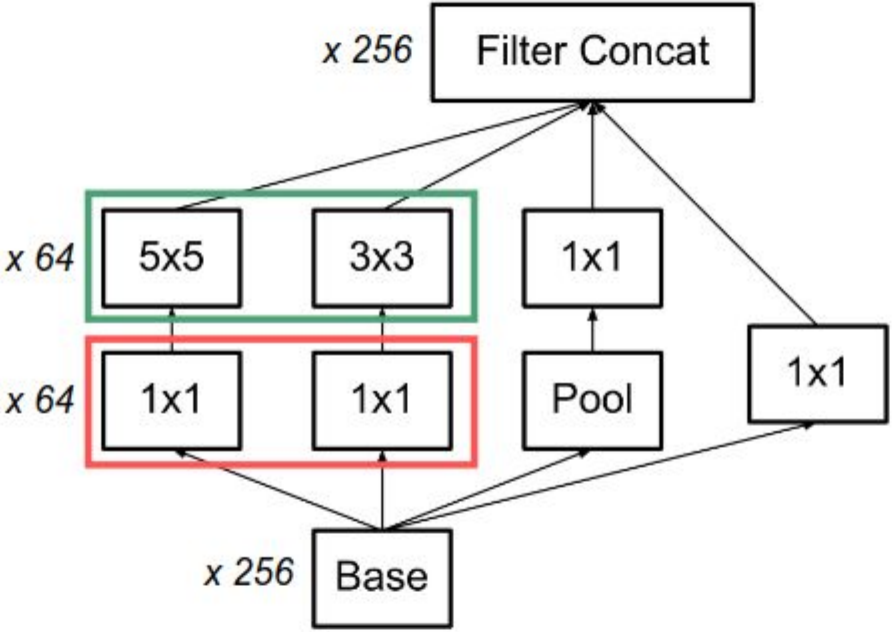
Inception v1: efficiency by 1 x 1 convolutions

```
torchvision.models.googlenet()
```

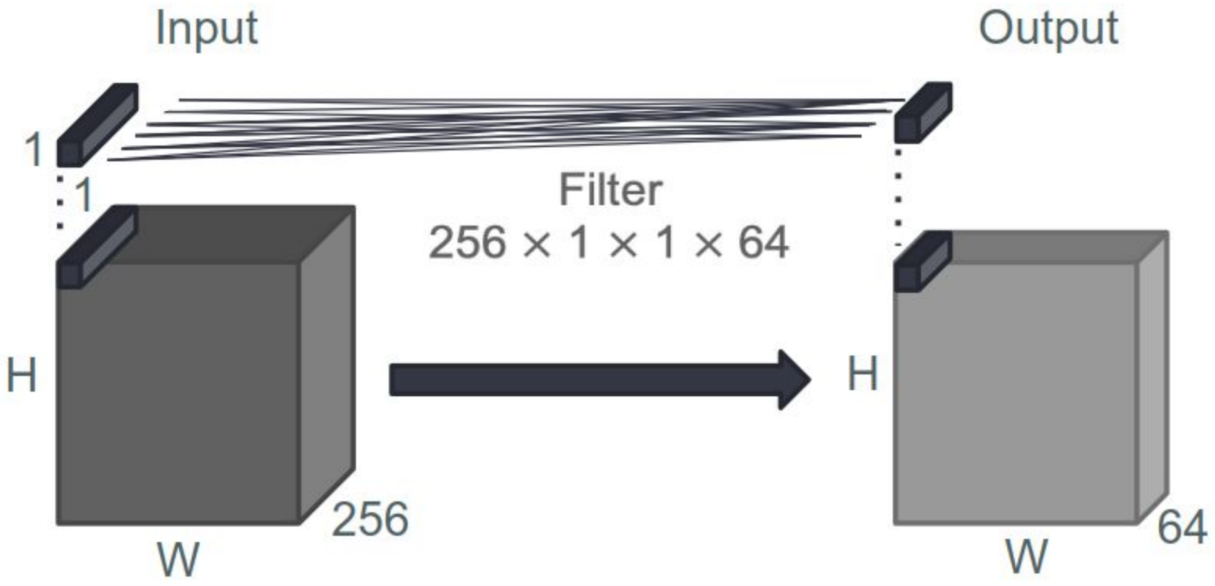
A single "Inception Module"

Spatial Processing

Reducing number of feature maps



1 x 1 convolutions



No spatial processing

Reduces number of channels

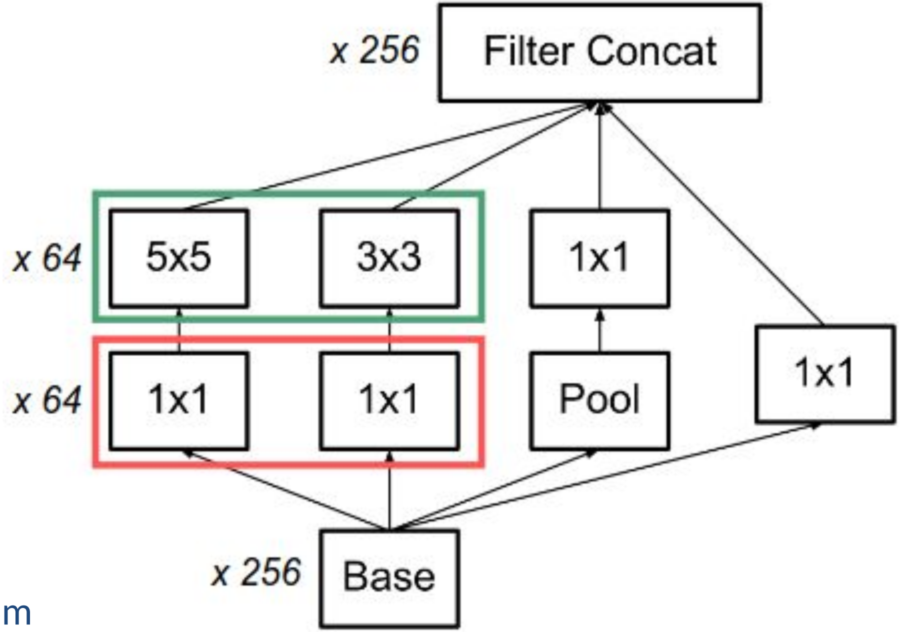
Inception v1: efficiency by 1 x 1 convolutions

```
torchvision.models.googlenet()
```

A single "Inception Module"

Spatial Processing

Reducing number of feature maps



 # parameters

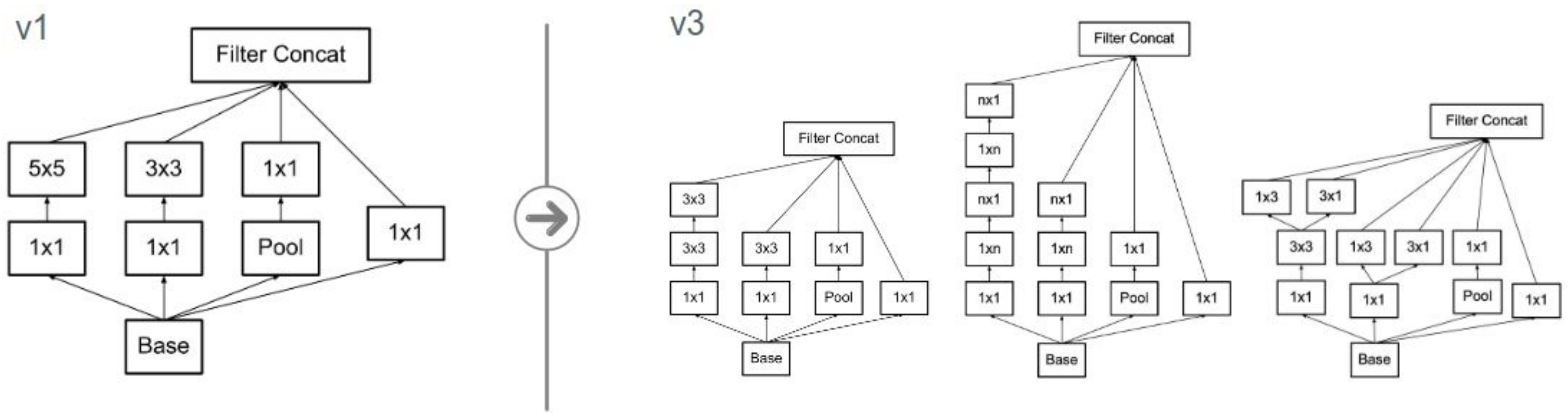
Standard convolution: 256 x 5 x 5 x 256 = 1.6m

Inception branch: 256 x 64 + 64 x 5 x 5 x 64 = 120k

Saturdays.AI Kigali

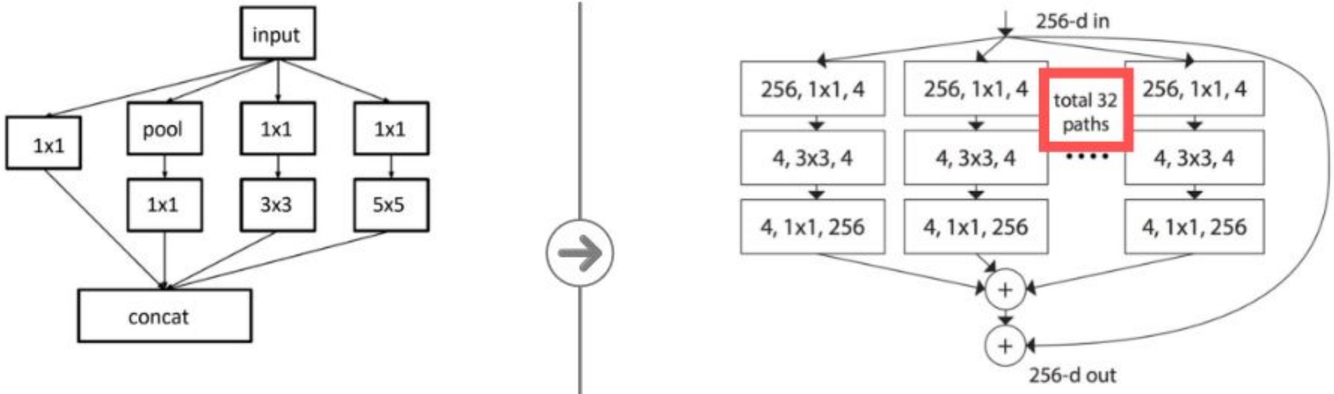
Inception v3: efficiency and bottlenecks

```
torchvision.models.inception_v3()
```



ResNeXt: combining Inception and ResNet

```
torchvision.models. models.resnext50_32x4d()
```



Inception:
heterogeneous multi-branch

ResNeXt:
uniform multi-branch

Compression

Spatial processing

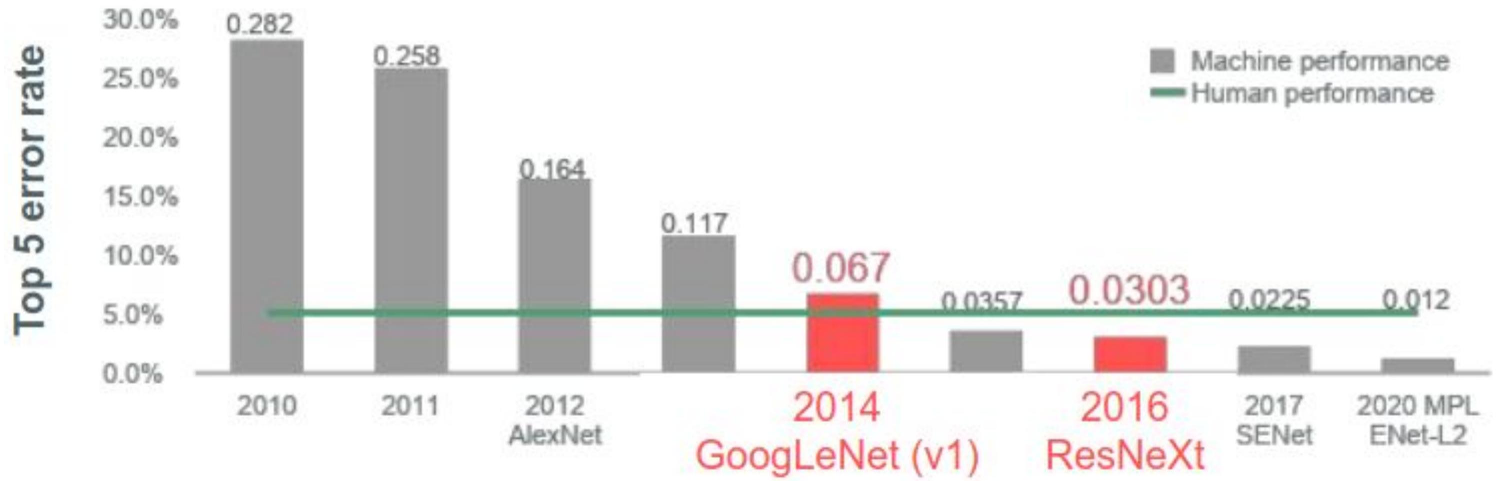
Channel mixing

Saturdays.AI
Kigali

Inception-v1 (2014) + ResNeXt (2016): Results

6.7% top-5 error rate in 2014, 3.0% in 2016!

Top-5 error rate of human expert (Andrej Karpathy) was ~5%



Saturdays.AI
Kigali

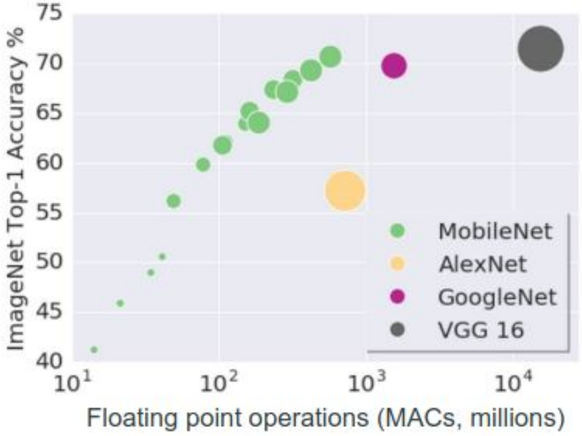
Improving efficiency

Part 2: MobileNet / depthwise separable convolutions

Saturdays.AI
Kigali

MobileNet (2017): pushing efficiency

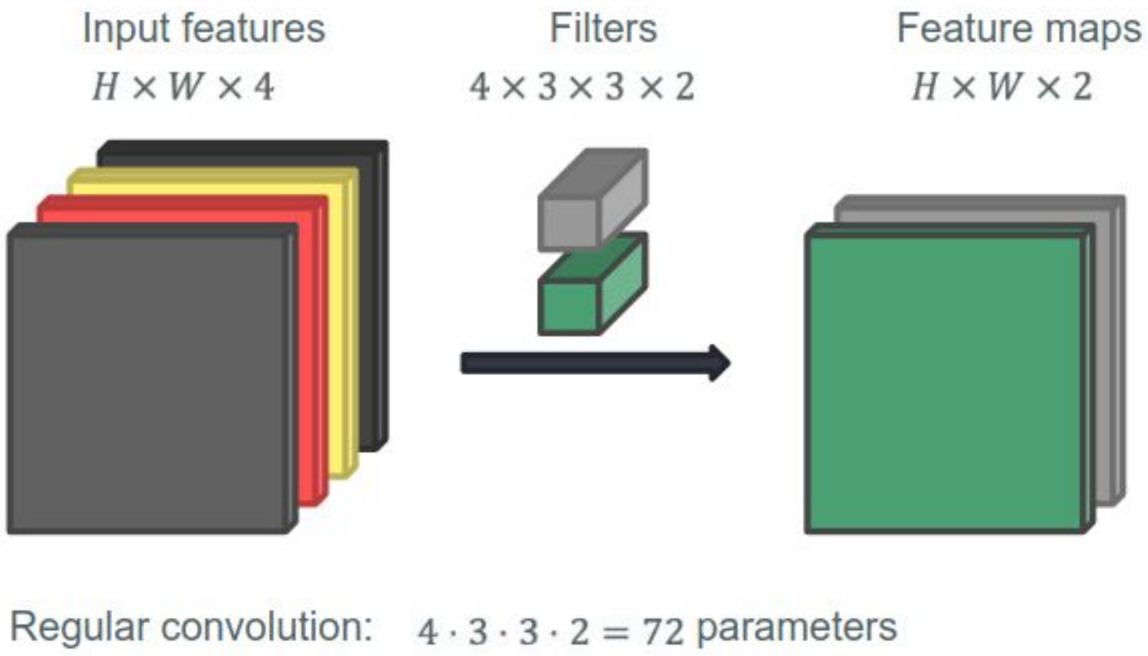
```
torchvision.models.mobilenet_v2()
```



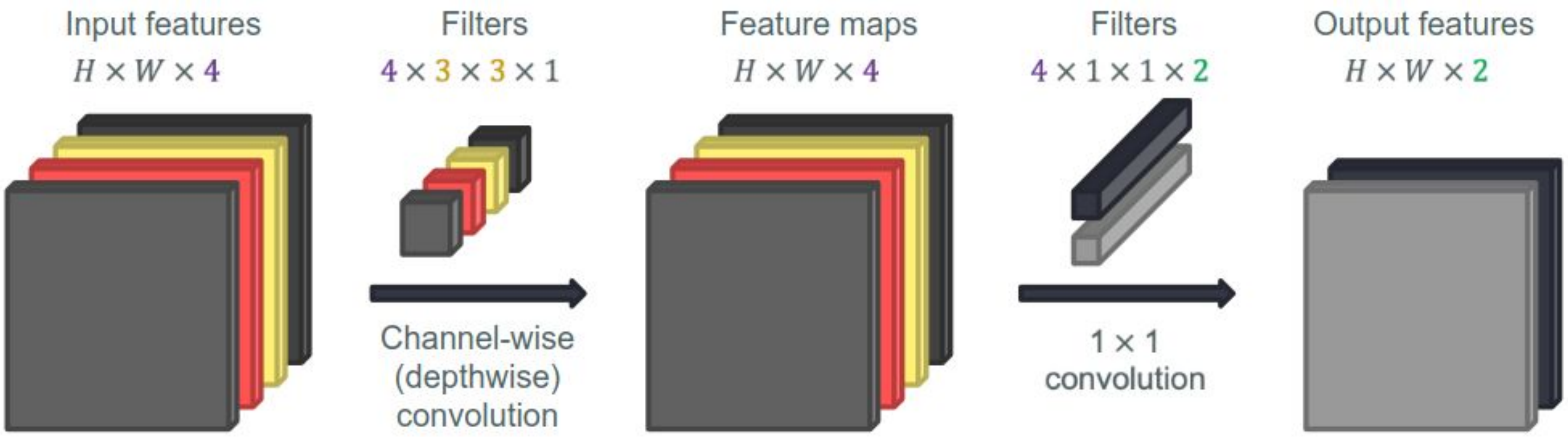
Same accuracy as VGG, >100 frames/sec on iPhone 7

Saturdays.AI
Kigali

Recap: regular convolution layer



Depthwise separable convolution



Depthwise separable: $4 \cdot 3 \cdot 3 + 4 \cdot 2 = 44$ parameters $\quad O(MK^2 + MN)$ parameters

Regular convolution: $4 \cdot 3 \cdot 3 \cdot 2 = 72$ parameters $\quad O(MK^2N)$ parameters

Saturdays.AI
Kigali

Depthwise separable convolution

Depthwise separable: $512 \cdot 3 \cdot 3 + 512 \cdot 512 = 267\text{k parameters}$

Regular convolution: $512 \cdot 3 \cdot 3 \cdot 512 = 2.4\text{m parameters}$

Practice

Understand how depthwise separable convolutions improve efficiency and save parameters in CNNs.

Coding Exercise 6.1

Break

Transfer learning

Building upon pre-trained CNNs

Saturdays.AI
Kigali

Standing on the shoulders of giants

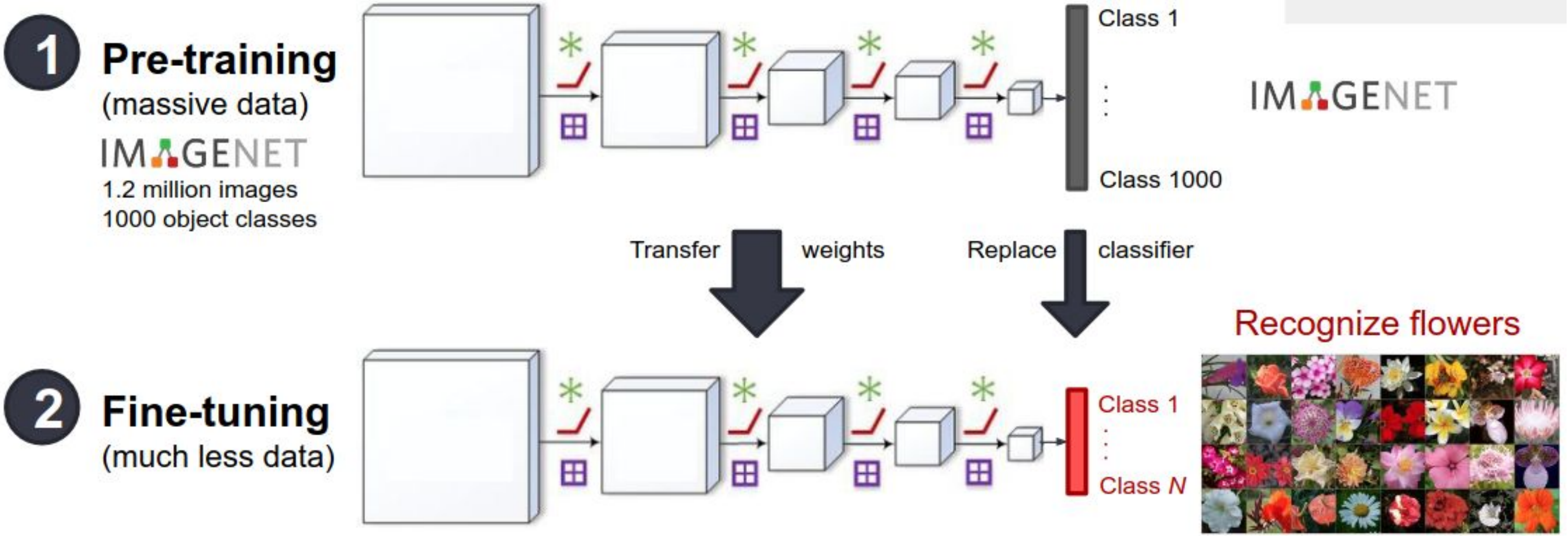
Image source: https://www.cgtrader.com/3d-models/character/man/cartoon-giant

Transfer learning: motivation

Suppose you want to build an app to recognize flowers…

Saturdays.AI
Kigali

Transfer learning: the idea



Saturdays.AI
Kigali

Transfer learning: what should we train?

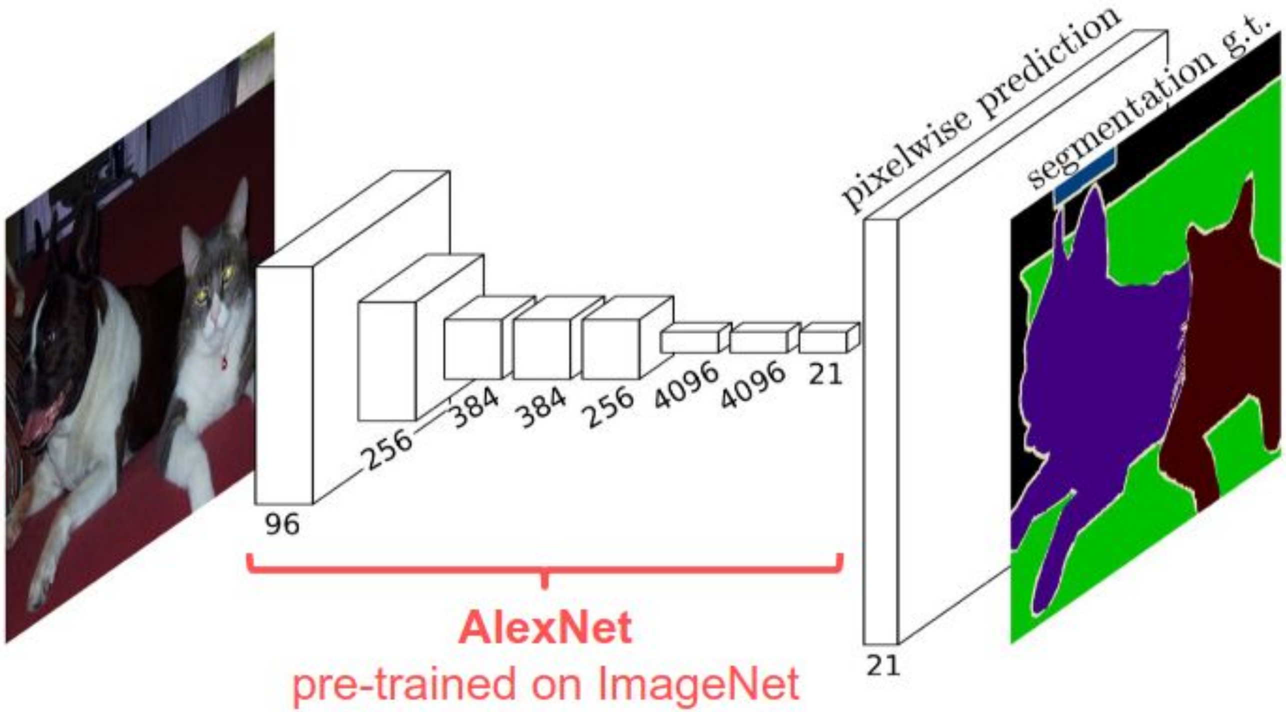
Option 1: Train only classification layer, freeze backbone
(sometimes referred to as the “linear evaluation protocol”)

- Fast & simple

Option 2: Train classification layer, fine-tune backbone at the same time

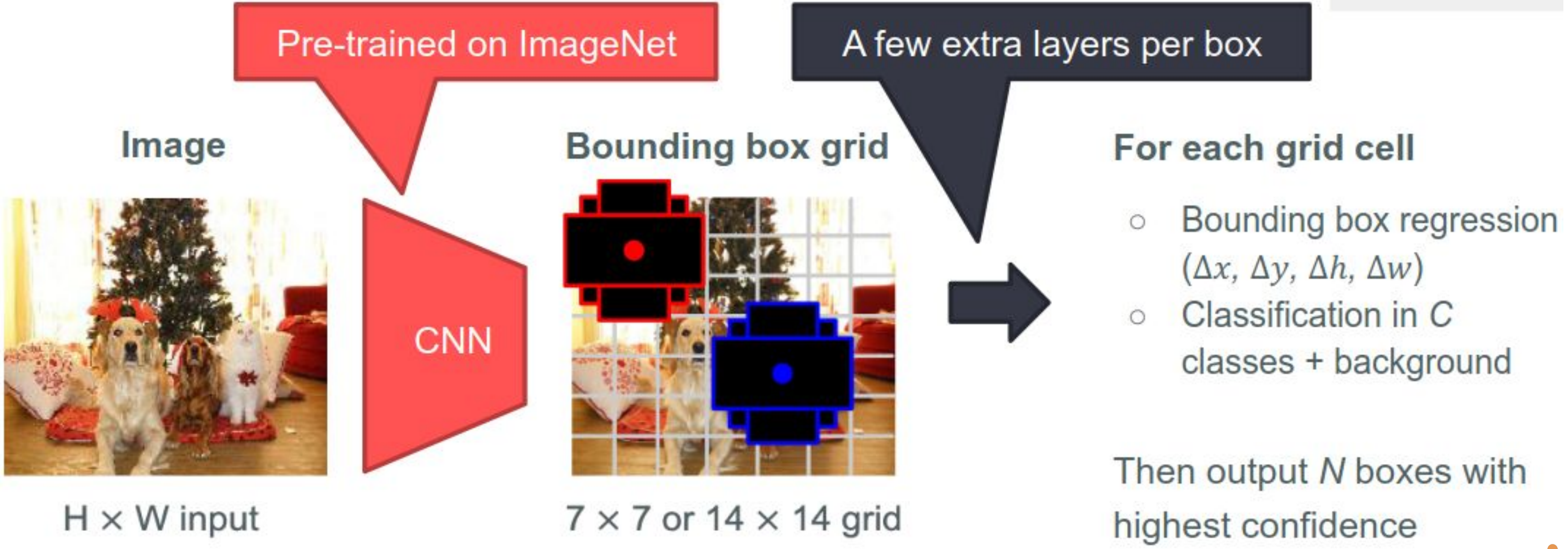
- Slower, but can adapt feature extraction to dataset statistics

Example: semantic segmentation



Saturdays.AI
Kigali

Single-stage object detectors: SSD, YOLO, …



Saturdays.AI
Kigali

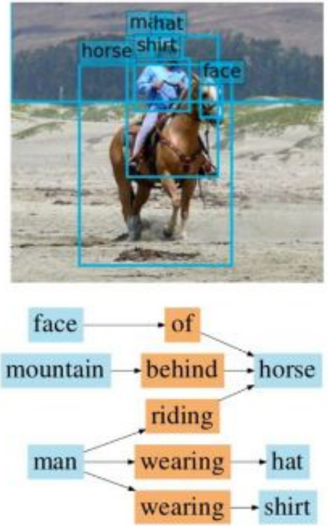
Pre-training on ImageNet is everywhere ...

Pose estimation

Image captioning

"two young girls are playing with lego toy."

Scene graph prediction



...

Many, many more

Saturdays.AI
Kigali

Practice

Implement transfer learning for Pokemon classification

Which approach do you think will perform best?

- Training from scratch
- Training only the classification layer
- Fine-tuning entire network

Section 7.4

Saturdays.AI
Kigali

TYU: Convolution

Assume that the number of filters is less than number of input channel, Which of the following will reduce the number of dimensions of input X

A. 1x1 CONV, 3 filters, no padding
B. 3x3 CONV, 3 filters, no padding
C. 3x3 POOL, 3 filters, no padding
D. All of the above

Challenges & Next steps!

Saturdays.AI
Kigali

Kahoot!

Saturdays.AI
Kigali

Any questions?

Saturdays.AI
Kigali

THANKS

Saturdays.AI
Kigali

kigali@saturdays.ai
coming soon
coming soon