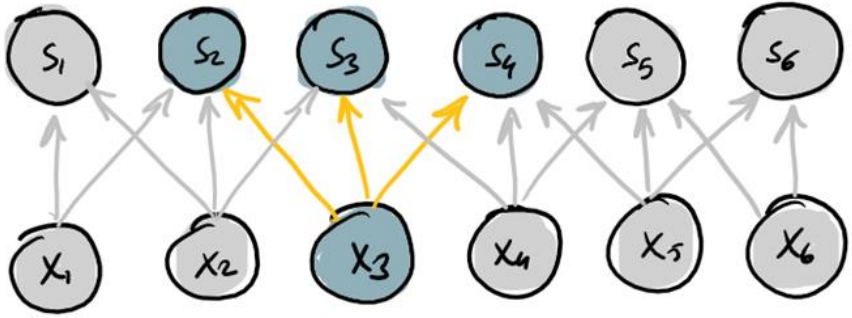
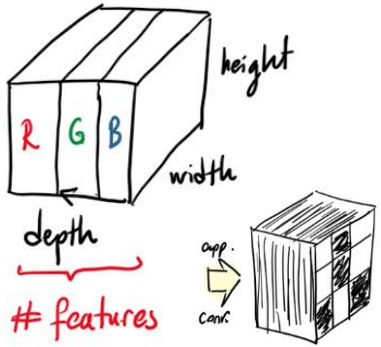
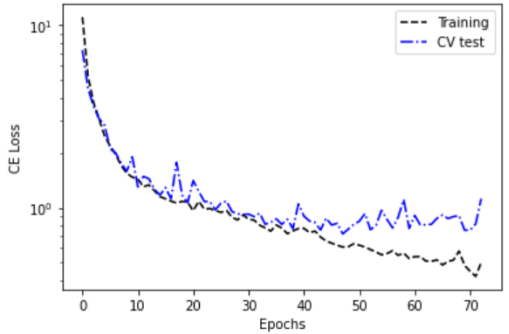
Data Driven Engineering II: Advaced Topics

Image processing and analysis

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer







Summary of last week

- [x] What is convolution
- [x] Convolution operation in ANN
  - [x] What is a filter?
  - [x] What do we learn in CNN?
  - [x] Advantages compared to MLPs
- [x] Padding, Stride, Pooling
- [x] Rule of thumbs

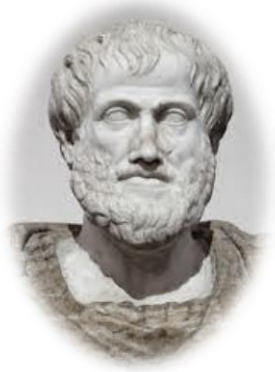
Case Study

- Image classification
  - augmentation

Outline of the week :

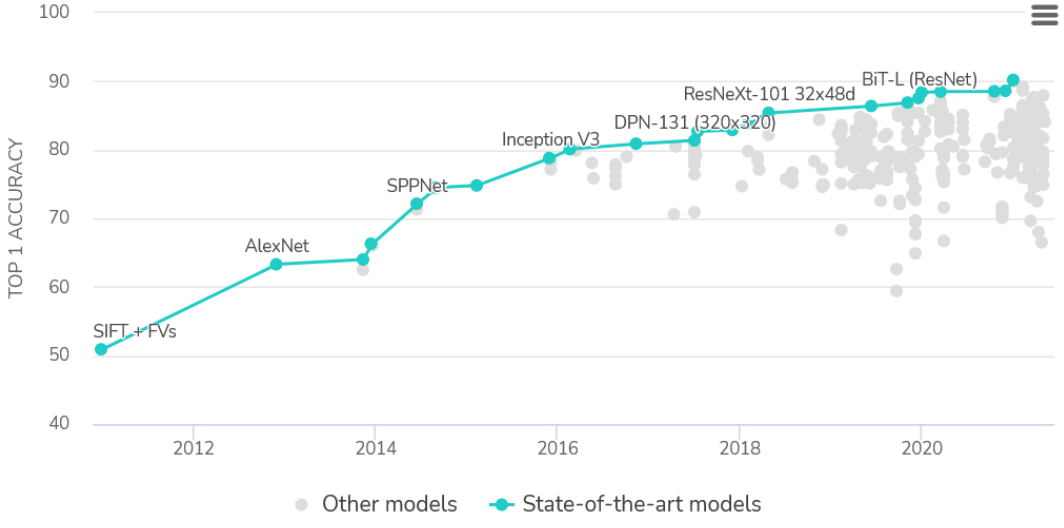
Conv. Neural Networks

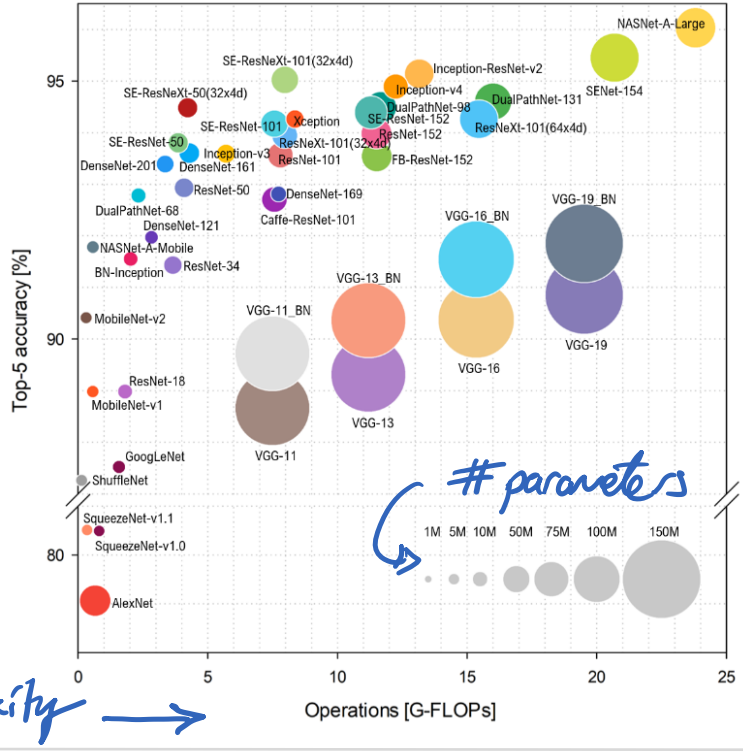
* What is CNN ?
* Why convolution is useful ?
* Where is it useful ?
* CNN – How does it work ?
* "Hall of Fame" : Popular Arch.
* Transfer Learning with CNN



"The soul never thinks without a picture."

where to begin ...



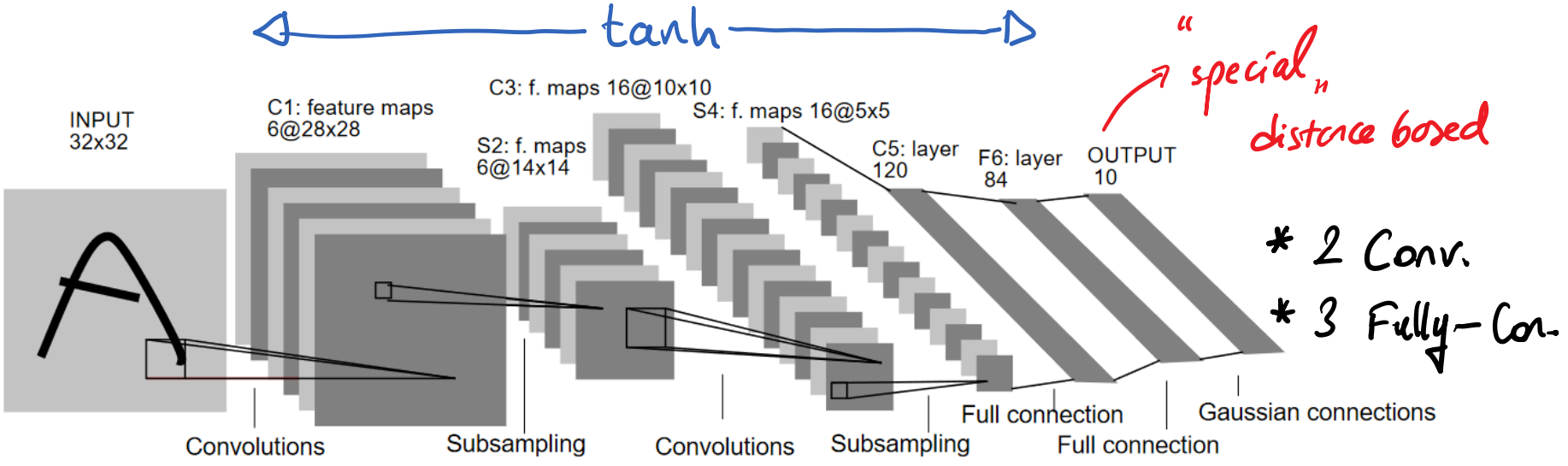


What is different?

* Increase in depth
* increase in width ⇨ more filters ⇨ filter hacks
* Regularization & fine tuning
* Training hacks
* Data augmentation & tr. learning
* Increased Image resolution

Who does not like classics?

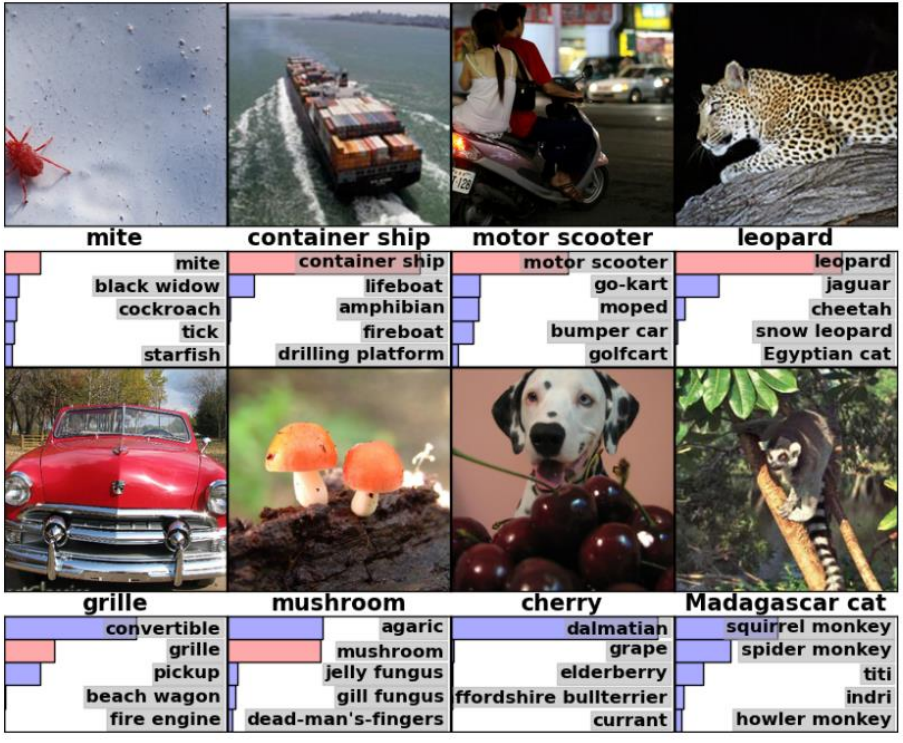
LeNet-5 (1998)



"special" distance based

* 2 Conv.
* 3 Fully-Con.

Image recognition – AlexNet (2012)



- Cited 128,948 times
- 1.2 Million images
- **5 layer network**:
  - 60 million parameters
  - 650k **point neurons**

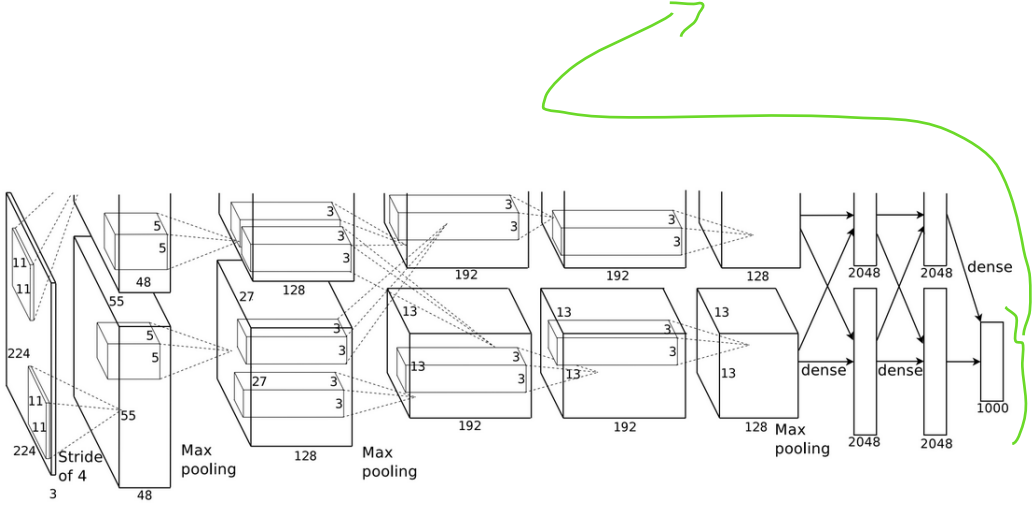
One small step for men, one giant leap for CNN

AlexNet (2012)

* Deeper := 5 Conv. + 3 Fully Con.
  ↳ 60 M parameters
* Stack conv. layers
* Use ReLu + Dropouts
* Data agumentation

11x11 kernel makes so many parameters.
This brings the problem of the vanishing gradients.

Thousands classes, thats why it has thousands neurons.

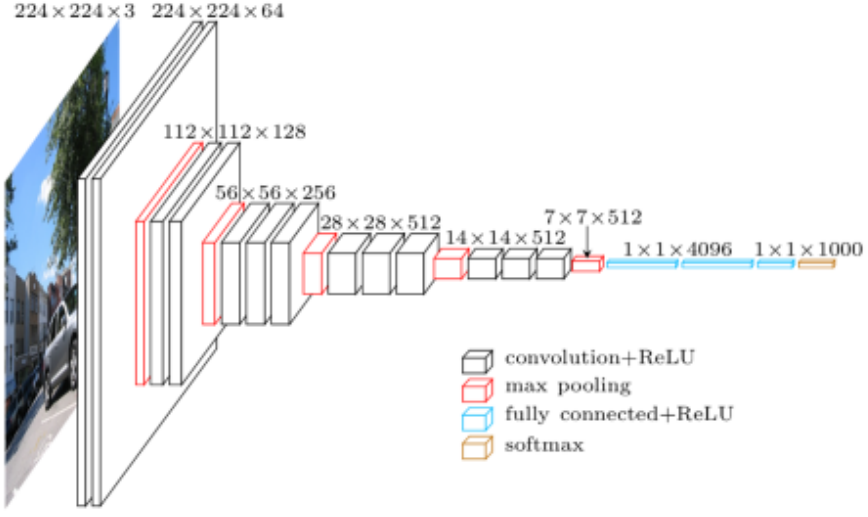


Deep Learning ...

VGG-16 (2014)

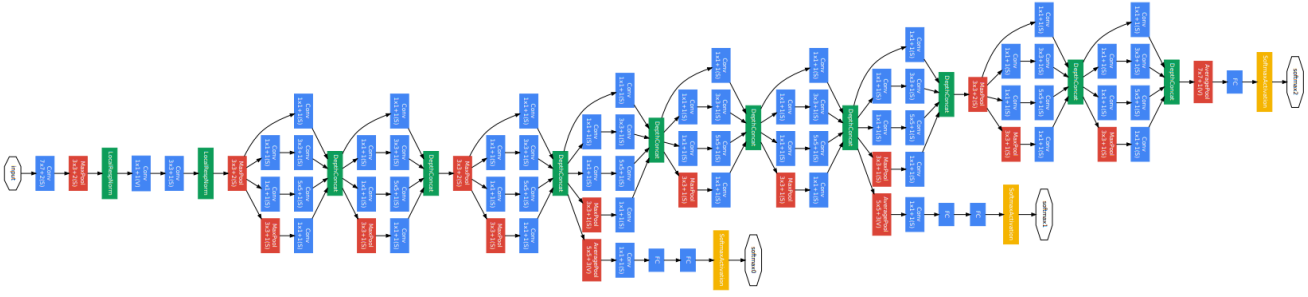
* 13 Conv. + 3 Fully-connected.
* 138 M parameters
* ReLu + Smaller Kernels (2×2, 3×3)

↳ Deeper Variant "VGG-19"



Things get multi-layered:

inception-v1 // GoogLeNet (2014)

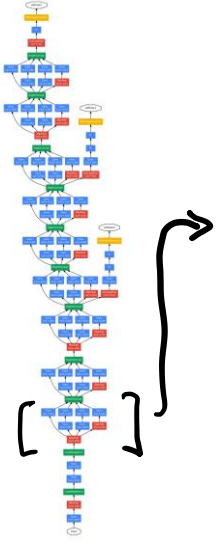


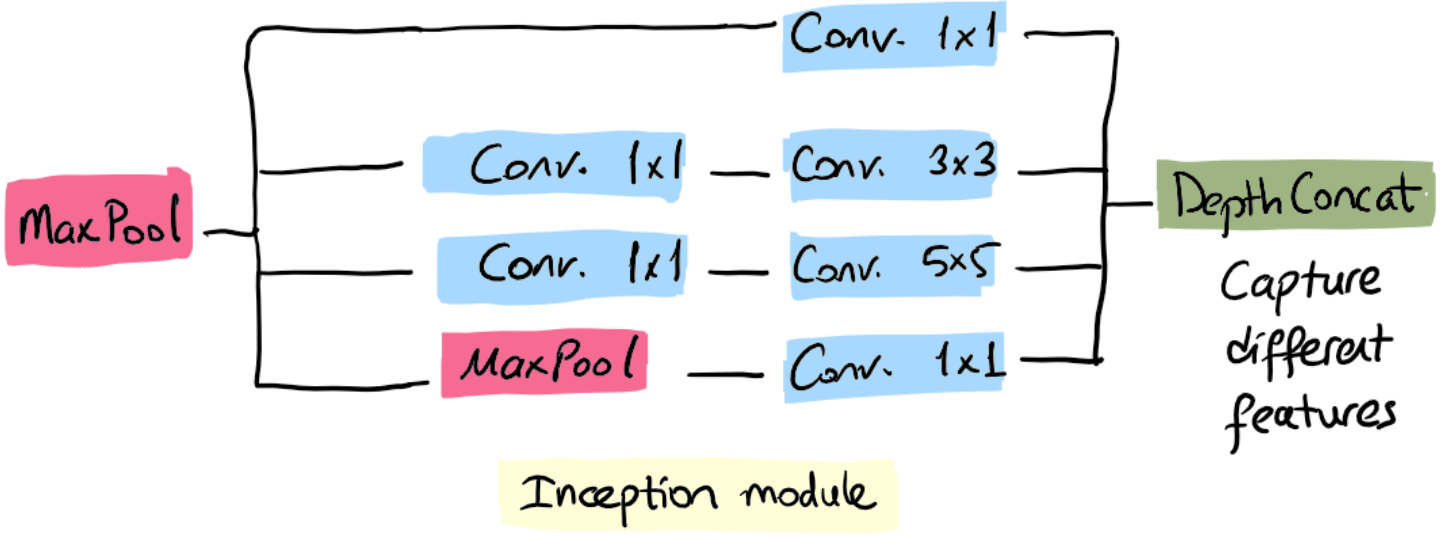
* 22 Layers ⇒ 5M parameters
* inception module ⇨ "network in network"
  ↳ enables much deeper network

Things get multi-layered:

inception-v1 // GoogLeNet (2014)

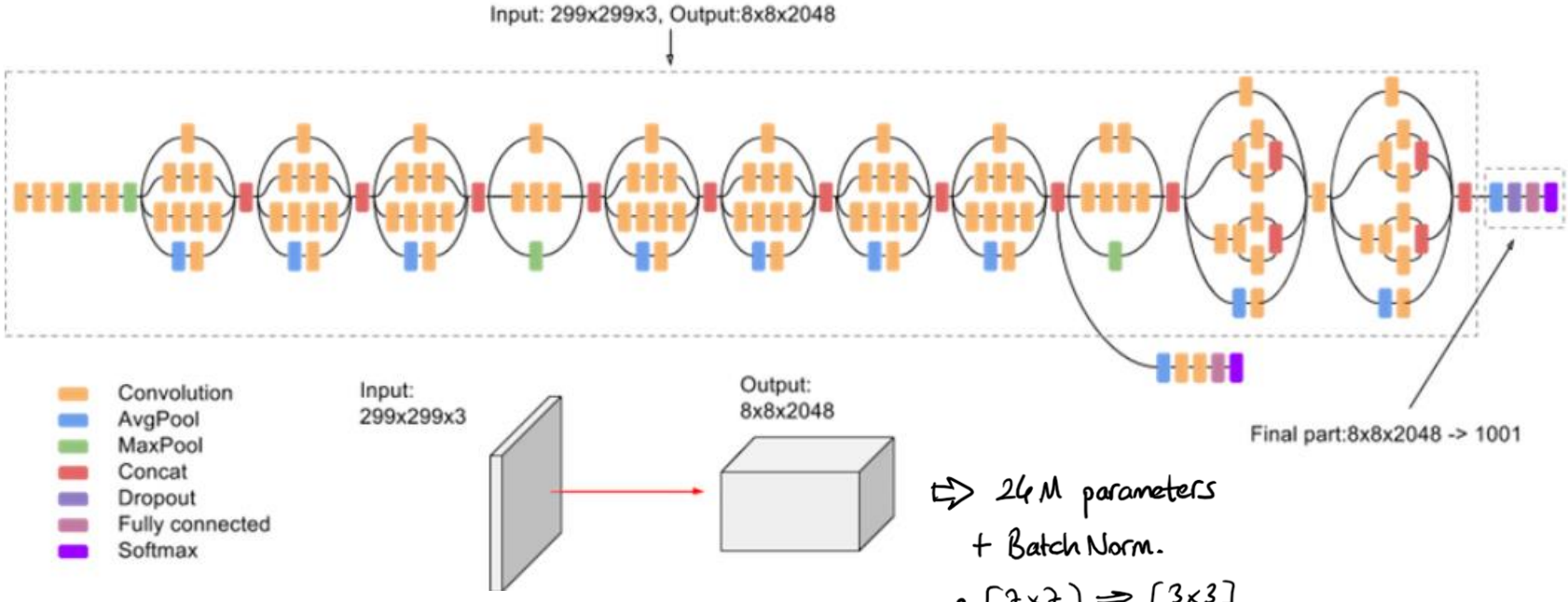
Now we have a multiple pipeline compared with the previous class.





Things get multi-layered: v3 (2015)

Instead of using the 3x3 filter size, they modified here.
Te models are getting bigger and bigger.



⇨ 24 M parameters
+ Batch Norm.

- [7×7] ⇒ [3×3]
- [5×5] ⇒ [3×3]
- [n×n] ⇒ [1×n, n×1]

(v4 → 2016)

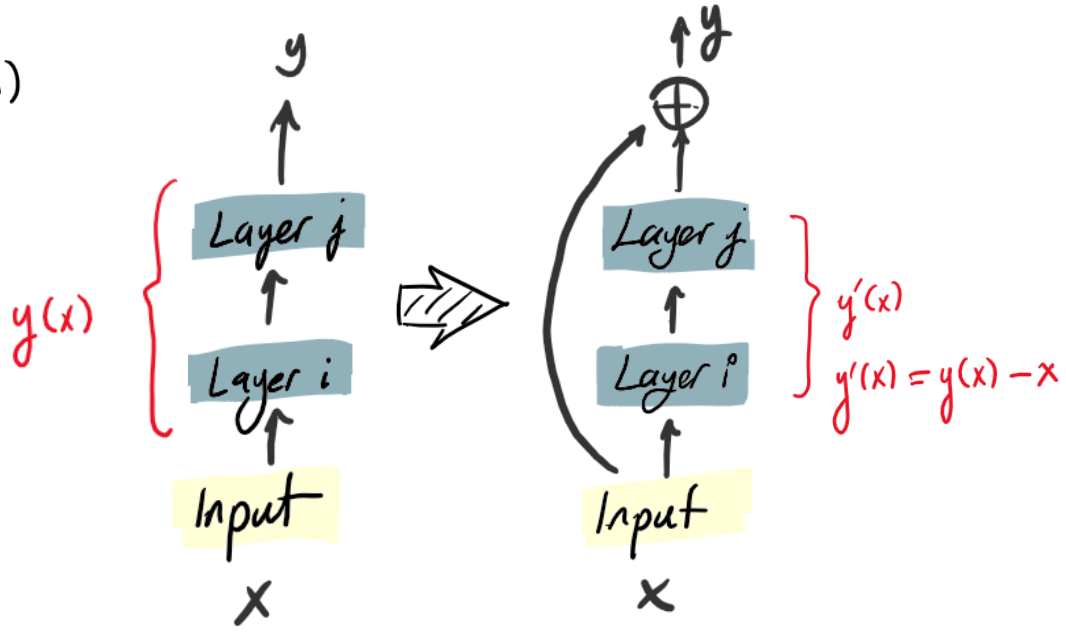
A smart touch: ResNet (2015)

* much deeper network → 152 layers
  ↳ fewer parameters (26M)

Exp./Van. Gradient problem

Skip Connections
+ BatchNorm

* Residual learning
* LSTM ?

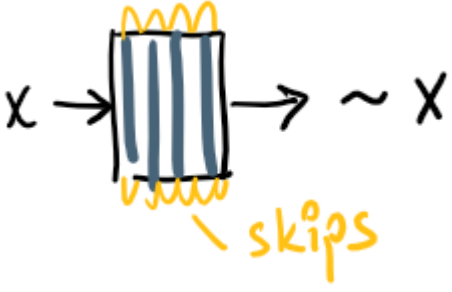


A smart touch: ResNet (2015)

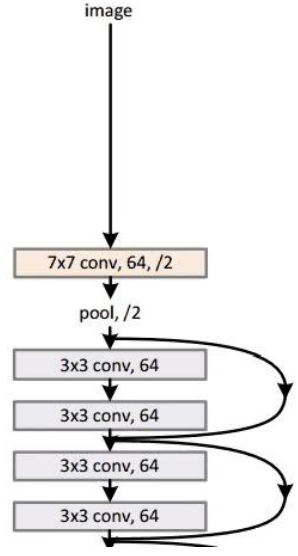
How does it help training?

① When network is initialized;

$\vec{w} \rightarrow \emptyset$



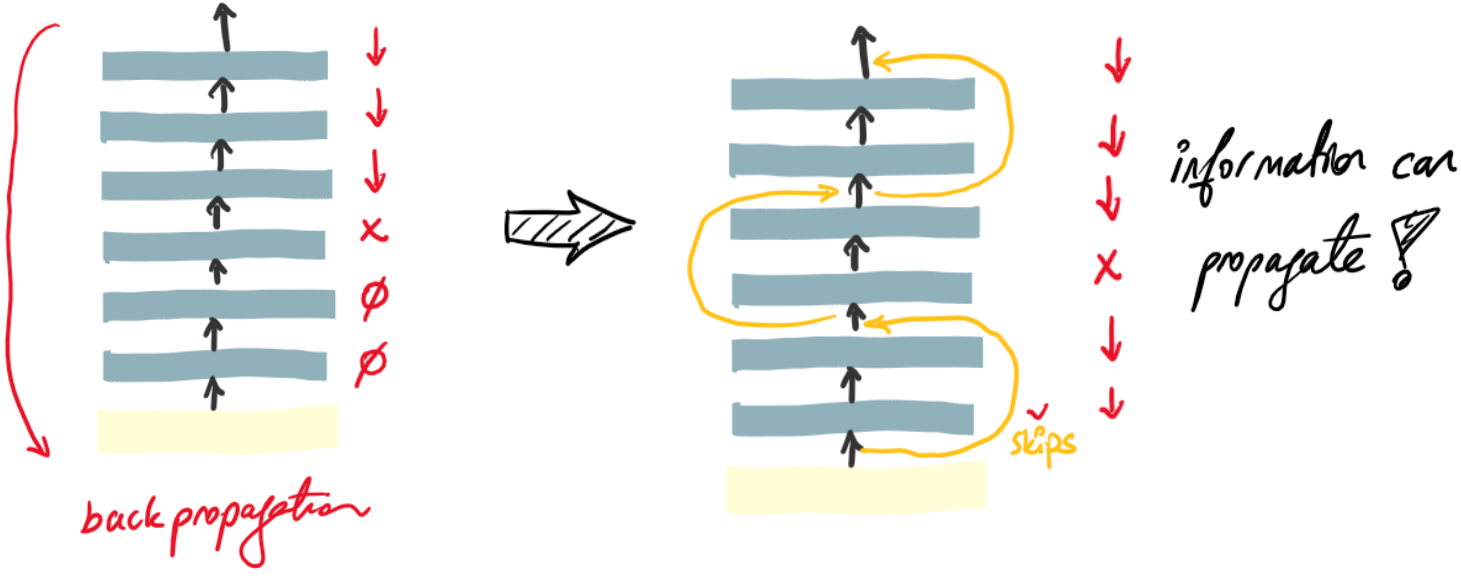
} identity function ↑↑↑ training



A smart touch: ResNet (2015)

How does it help training?

② Learning bottlenecks



It looks nice . Lets use it everywhere !

Karlsruhe Institute of Technology

DenseNet 2017

* use skip connections in dense blocks → Re-use features with less parameters
* use "concatenate" rather than adding ! Needs proper padding

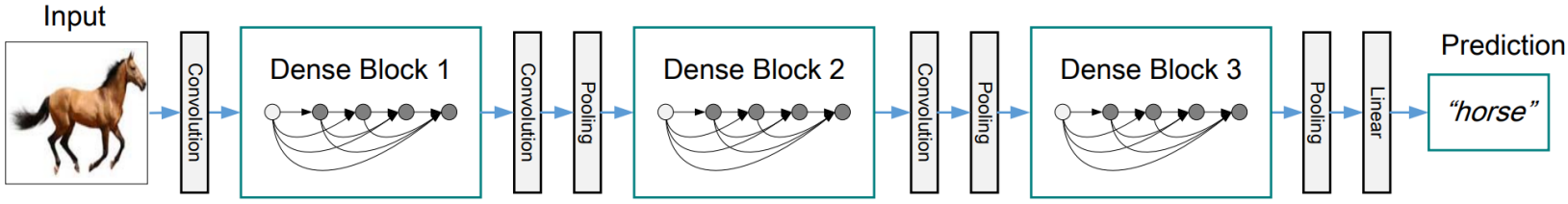
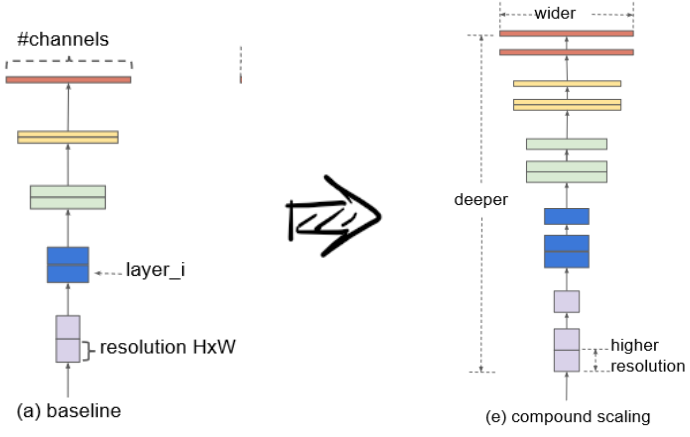
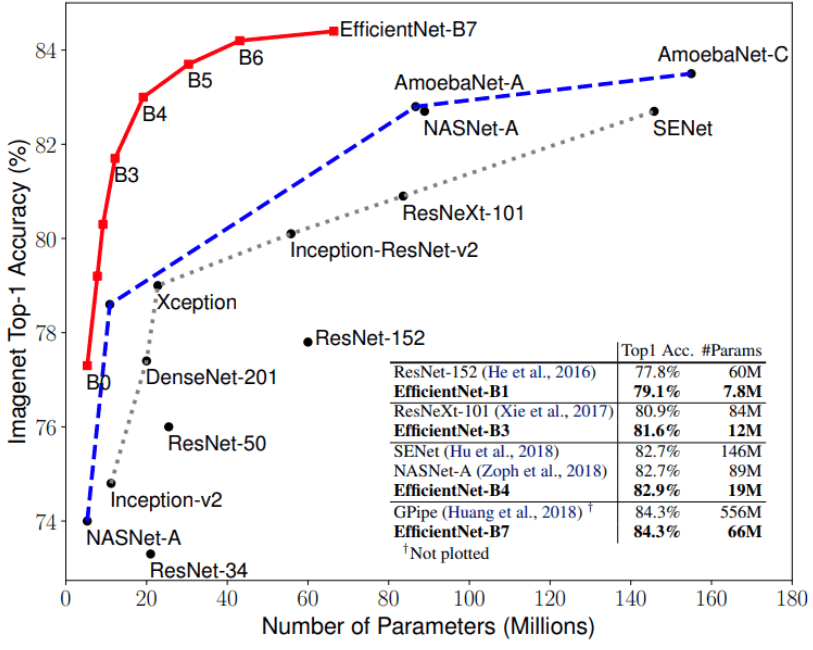


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

EfficientNet: "Rethinking model scaling"

* balancing network → scaling method { depth, width, resolution





	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

EfficientNet: "Rethinking model scaling"

* balancing network → scaling method { depth, width, resolution }

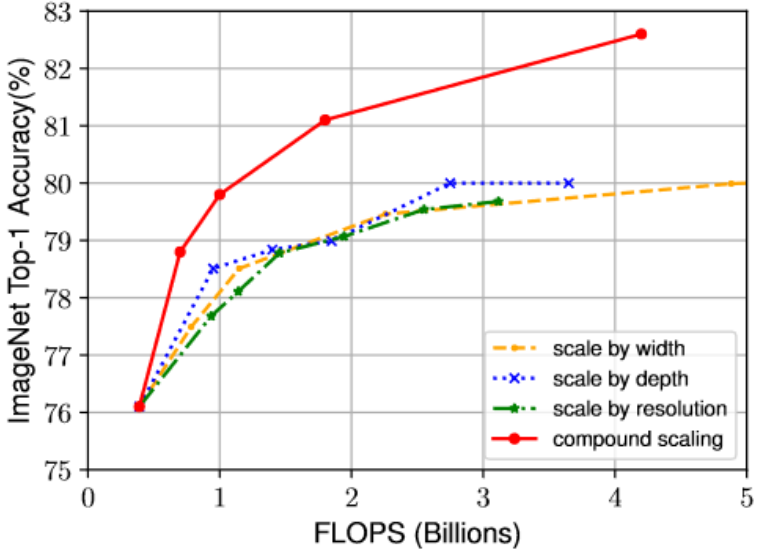
⇒ Start small; gradually scale it

- depth = $d = \alpha^{\phi}$; $\alpha \beta^2 \gamma^2 \approx 2$ (FLOPs)
- width = $w = \beta^{\phi}$
- res. = $r = \gamma^{\phi}$; $\alpha, \beta, \gamma > 1$

(i) $\phi = 1$; do grid search on α, β, γ

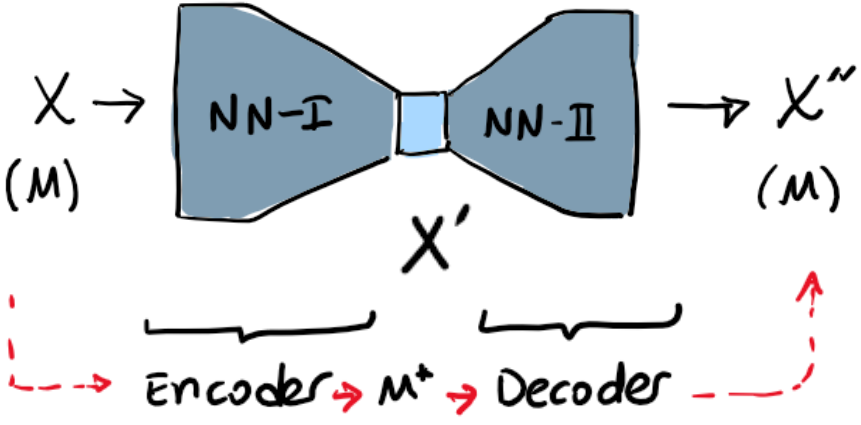
eg. (1.2, 1.2, 1.15)

(ii) Fix α, β, γ; scale ϕ with respect to hardware



A useful Tool: Conv. Autoencoders

* DDE-1 ⇨ "Autoencoders" ⇨ "PIV & GAN-Representation Projects"

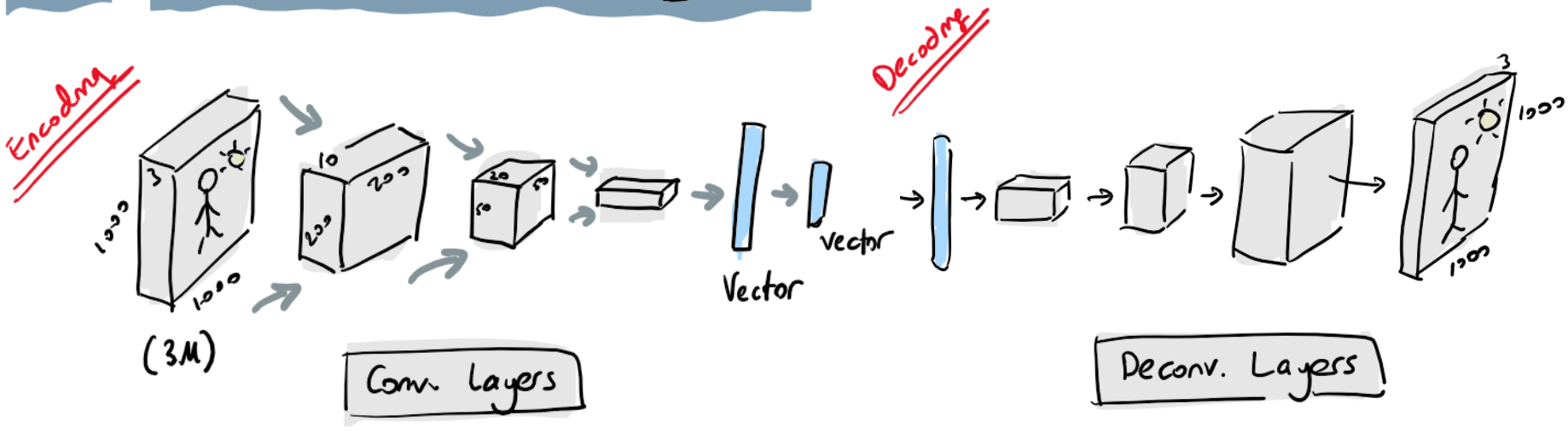


☑ $(X - X'') :=$ Reconst. Error

☑ $M^* << M$

Representation Learning

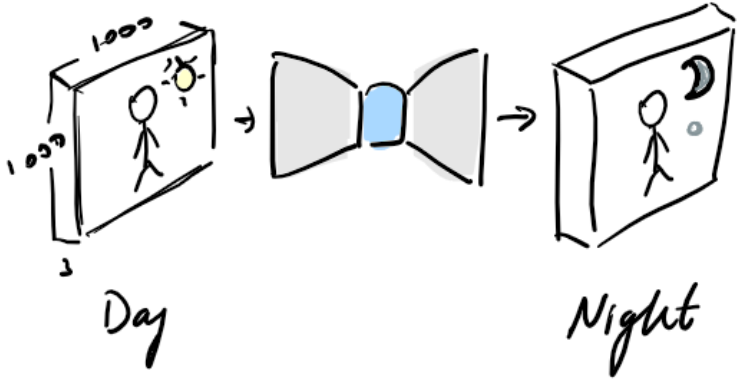
A useful Tool: Conv. Autoencoders



How this is useful ?

A useful Tool: Conv. Autoencoders

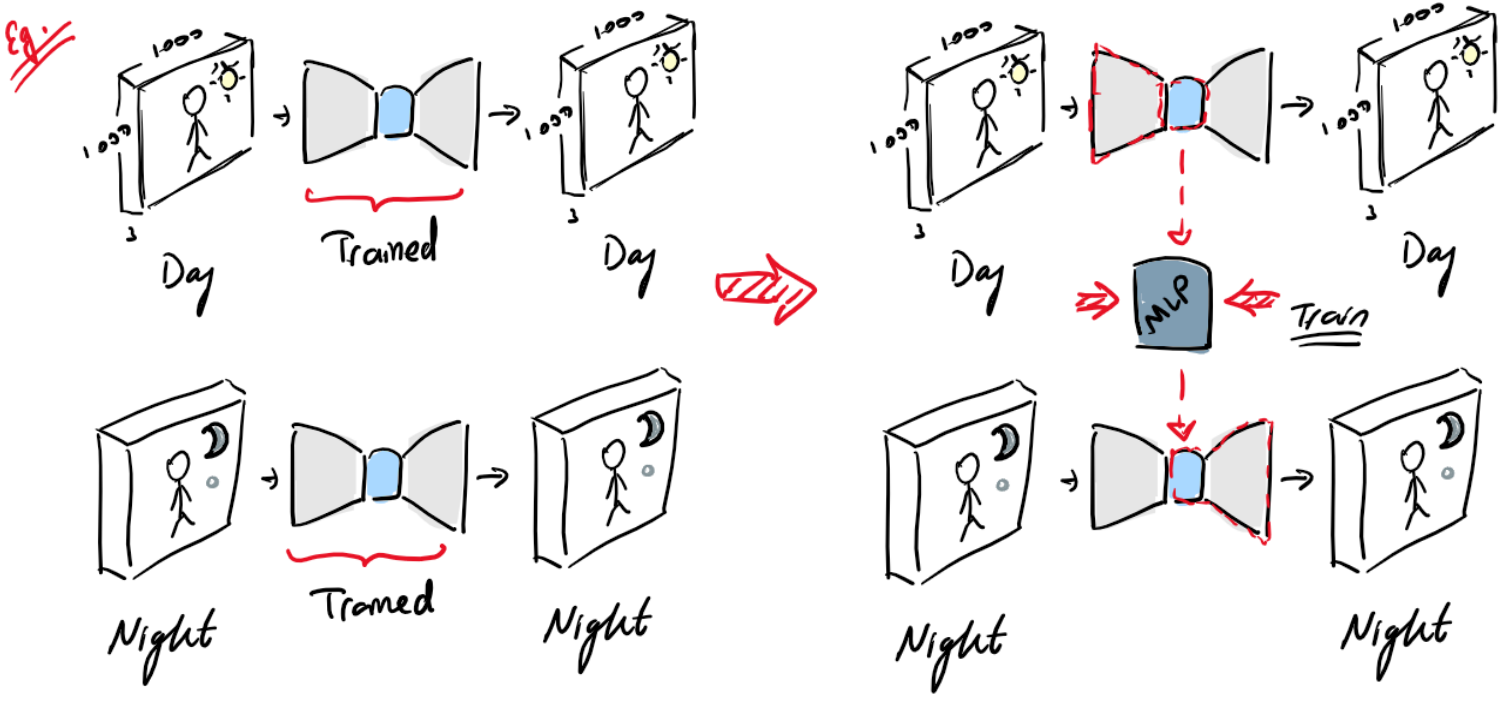
Eg.:



(*) Image translation; (*) CNN pretraining

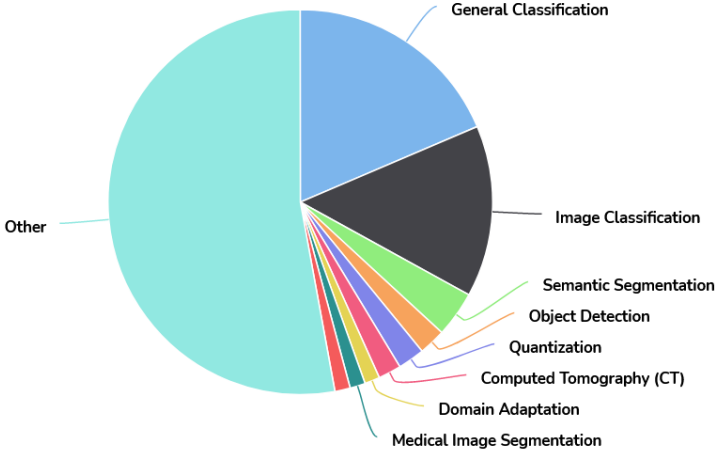
- Day → Night
- B&W → Color
- Noisy → Clean
- ...

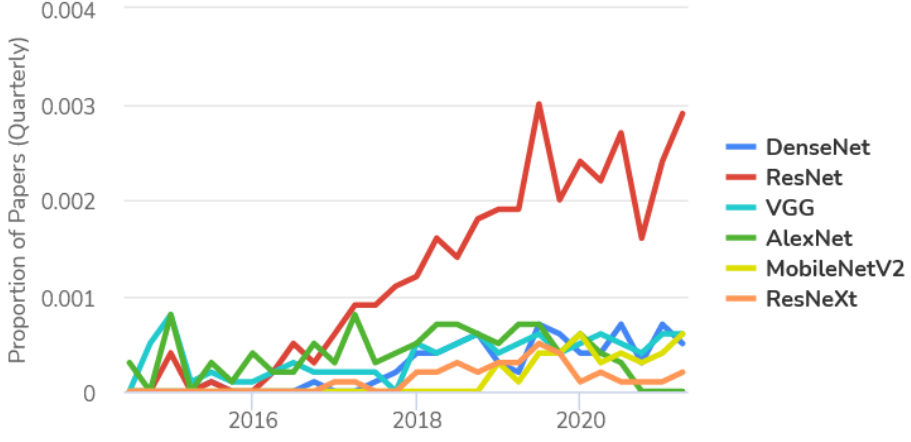
A useful Tool: Conv. Autoencoders



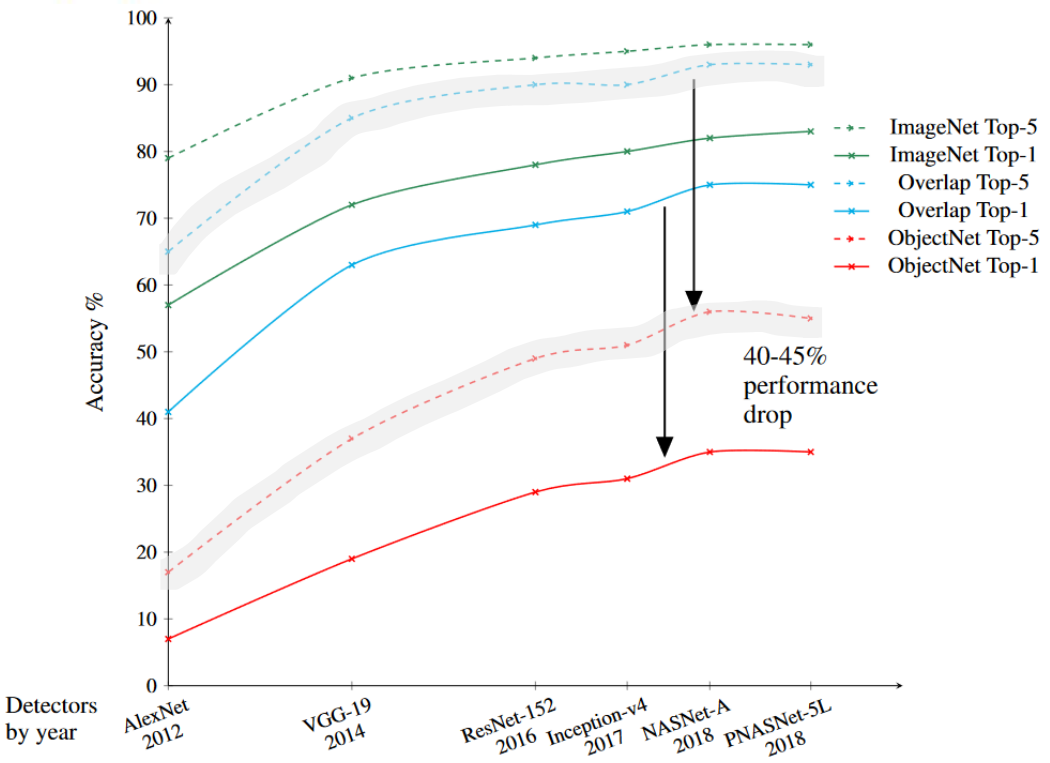
Too many options ...

* Models including CNN ~ 100 accessible
* Utilized in various fields

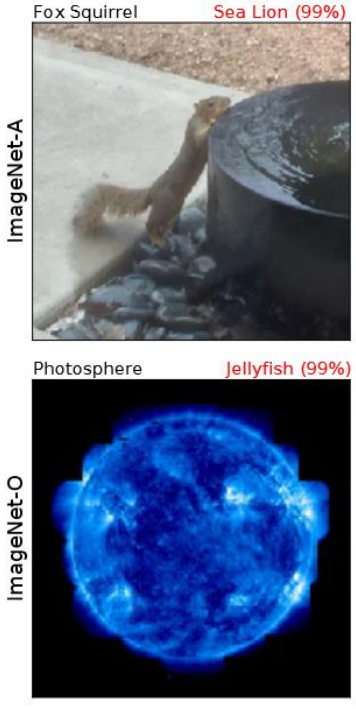


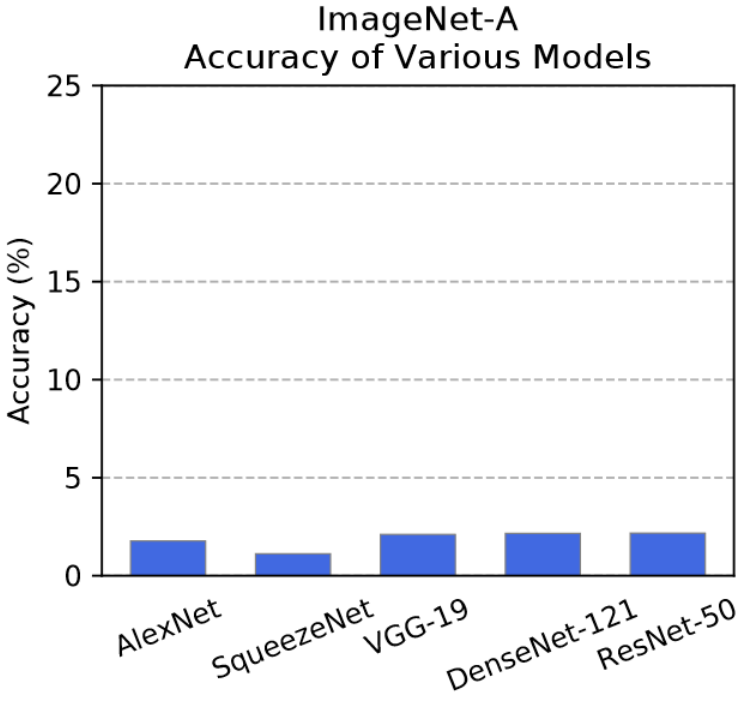


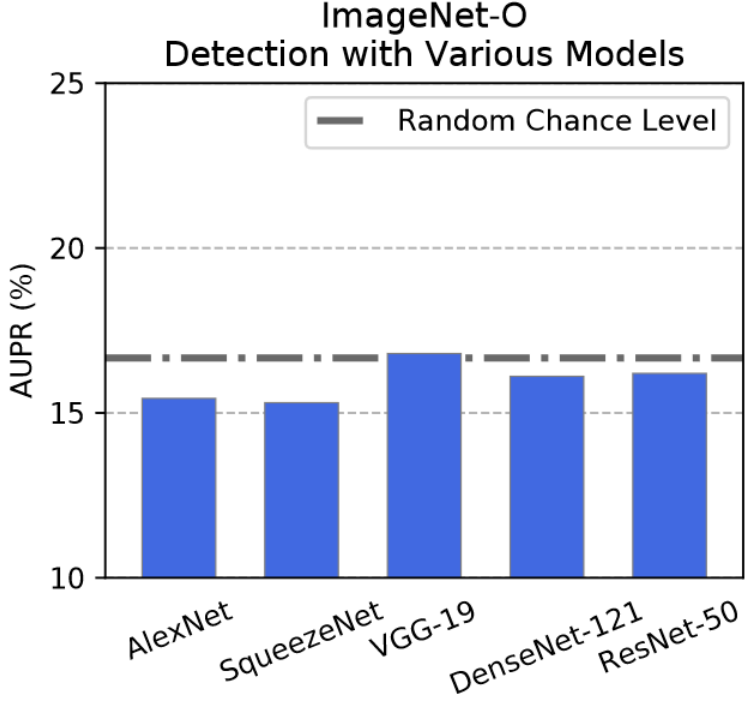
Shallow & Brittle: no hierarchical perception



Shallow & Brittle: no hierarchical perception

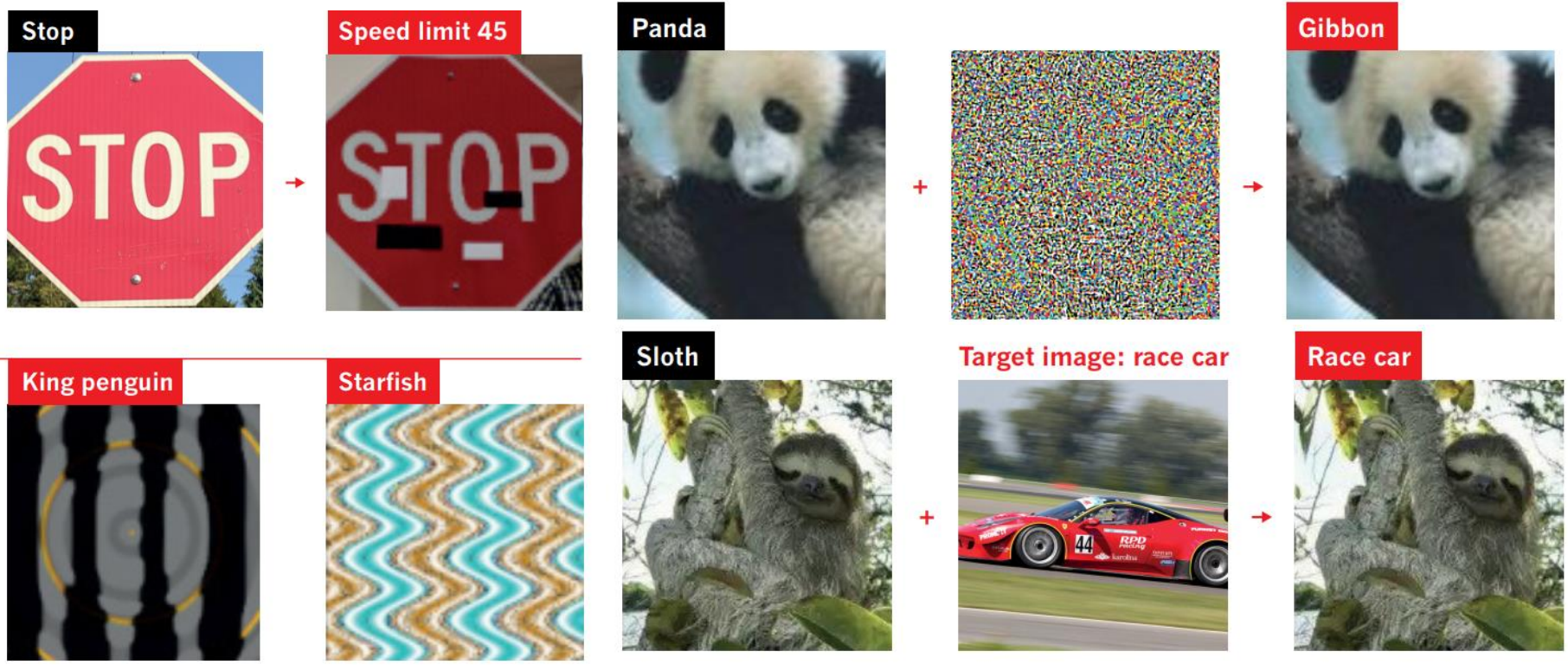




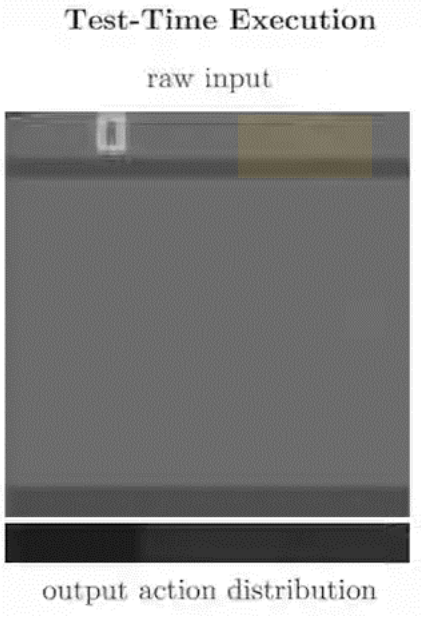


https://arxiv.org/pdf/1907.07174.pdf

Lack of hierarchical perception: easy to trick



Lack of hierarchical perception: easy to trick



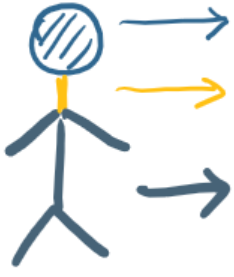
https://arxiv.org/abs/1702.02284

Case study: Object Detection:

* {Classify + Location + many objects}

Model anatomy reflects this aim

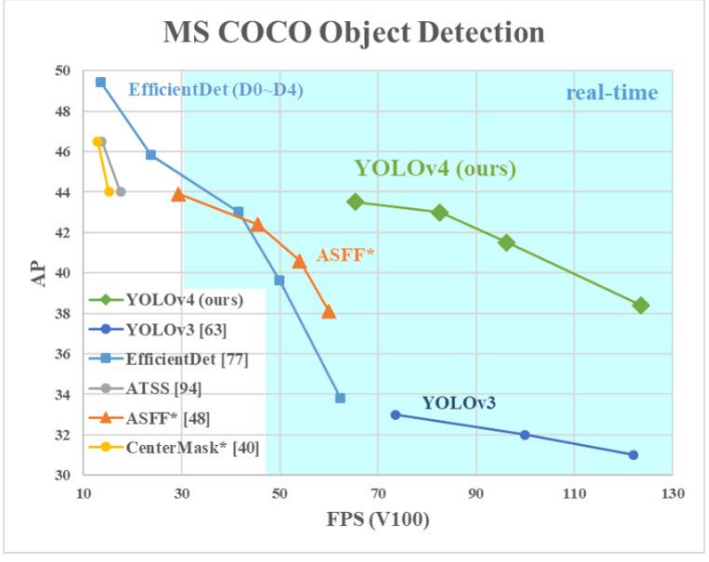
* YOLO := "You only look once"



→ Head : Decision-unit

→ Neck : collect features

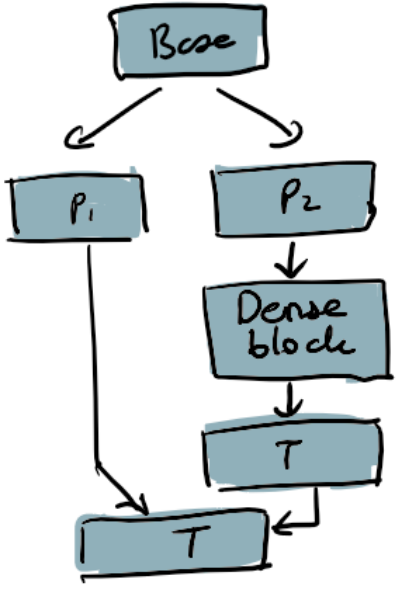
→ Backbone : extracts features



Case study: Object Detection:

Backbone: Dense block + Cross-Stage-Partial connections + Darknet 53

higher acc. > Resnet



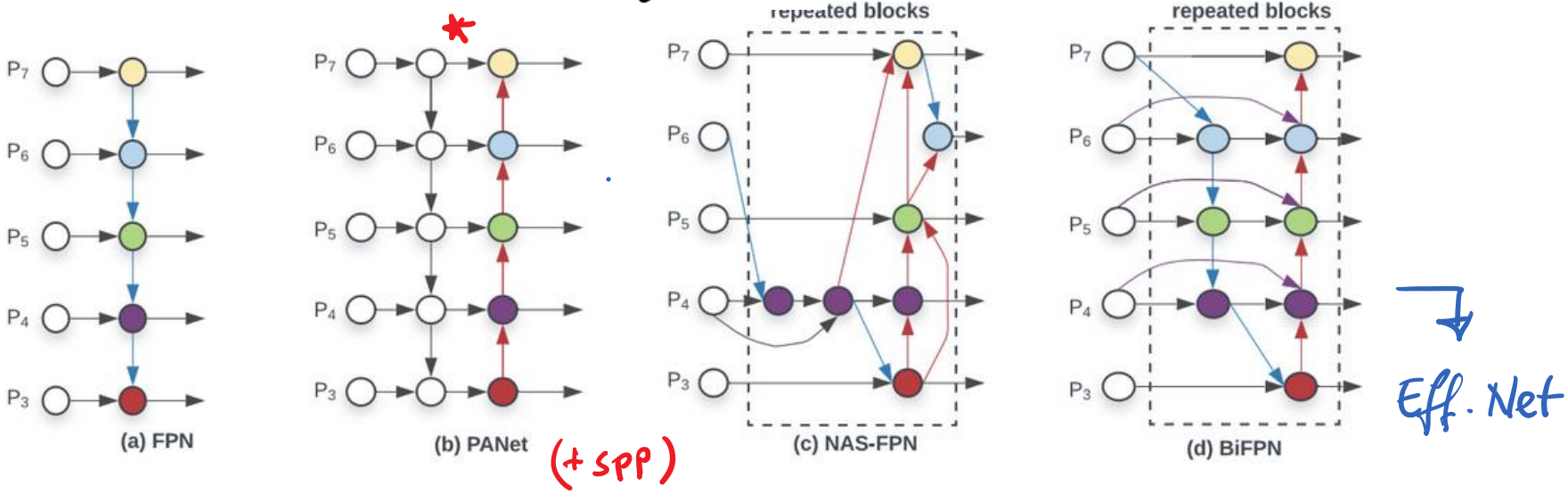
	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Case study: Object Detection:

Neck

* Combination of features managed here

P → feature layer



Eff. Net

Case study: Object Detection:

Head

* YOLOv4 ← YOLOv3
* anchor-based detection
* 3 levels of granularity

Bag of Freebies

* DropBlock regularization
* Class label smoothing
* CutMix & Mosaic data aug.

Bag of Specials

* Mish activation
* Cross Stage Partial connections
* Multi-input-weighted residual connections

Case study: Object Detection:

Anchor Boxes

(i) Create many boxes for each predictor

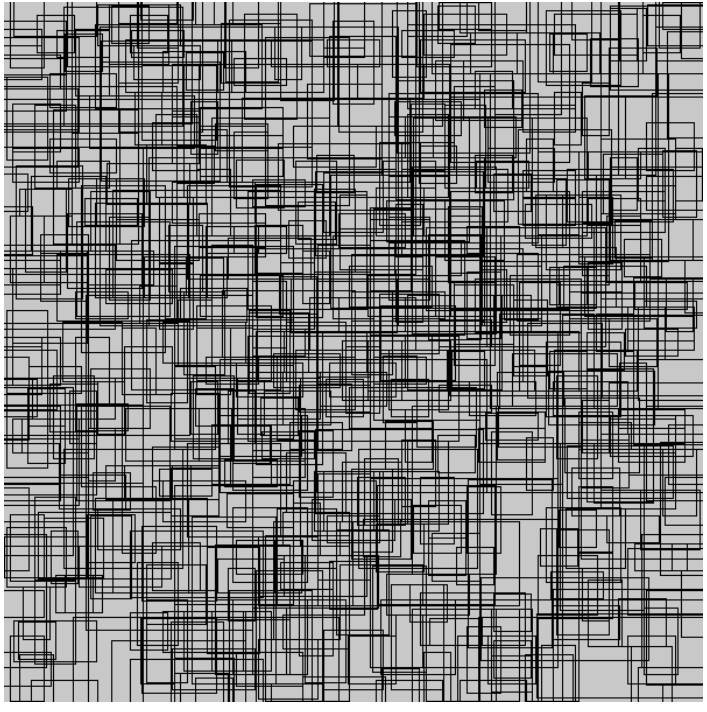
(ii) For each box: calculate which objects bounding box has the highest overlap/non-overlap ratio. (IOU)

(iii) If highest (IOU) > 50% ⇒ "Detect object"

(iv) ~40% – 50% ⇒ ambiguous ⇒ "not learn from this case"

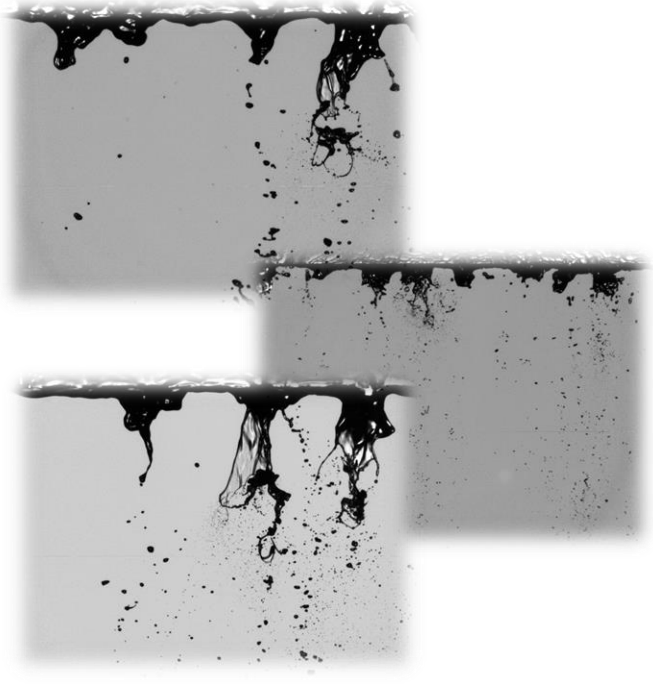
(v) < 40% ⇒ No object here

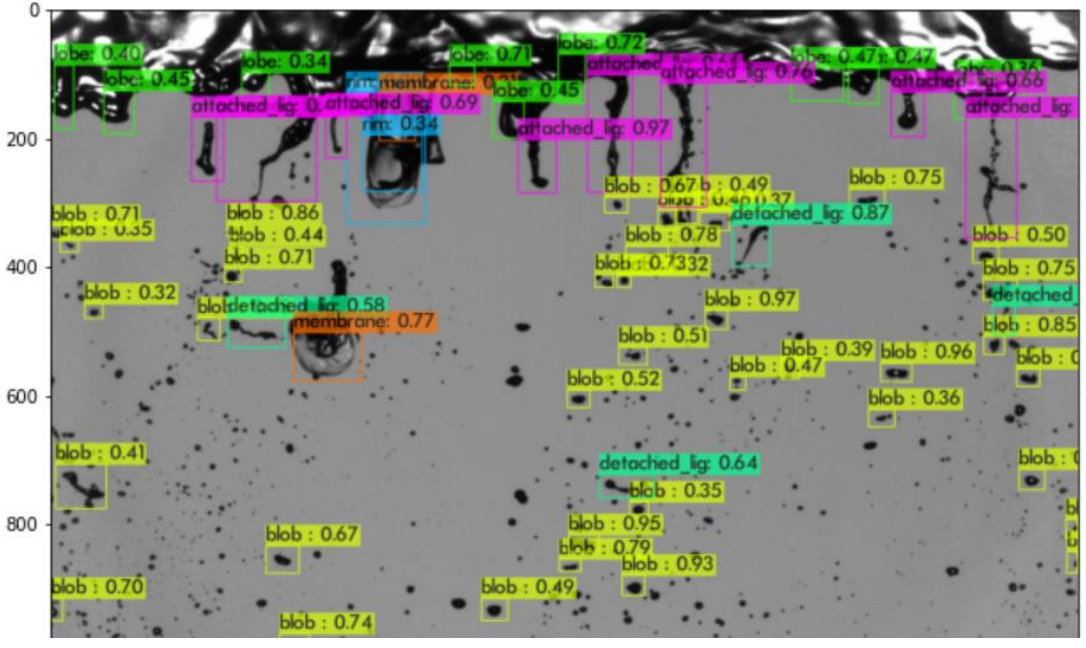
? Box dimensions ⇒ YOLO: K-means clustering on training data



Case study: Object Detection:

Get the pictures of each part of the video as frame per seconds, and use it for training our model.





Ateliers & Saveurs in Montreal

colab

Methods > Computer Vision

Computer Vision

544 methods • 43959 papers with code

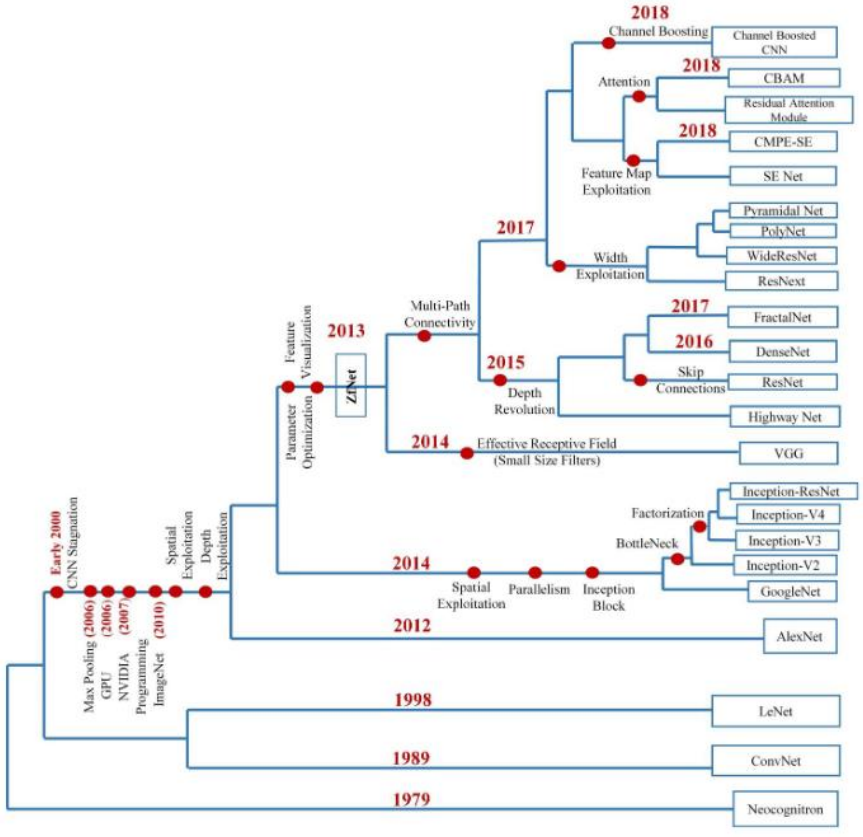
Image Feature Extractors

See all 39 methods

Convolutions

Convolutional Neural Networks

METHOD	YEAR	PAPERS
 ResNet	2015	1139
 VGG	2014	293
 AlexNet	2012	278
 DenseNet	2016	248
 MobileNetV2	2018	144
 ResNeXt	2016	107
 GoogLeNet	2014	106
 EfficientNet	2019	101



Artificial Intelligence Review
A Survey of the Recent Architectures of Deep Convolutional Neural Networks

ImageNet Challenge :