高等机器学习

# 计算机视觉

李亚利

Acknowledge: 郑书新 (微软亚洲研究院)

Microsoft

清華大學 Tsinghua University

# Computer Vision

Make computers understand images and video or any visual data.

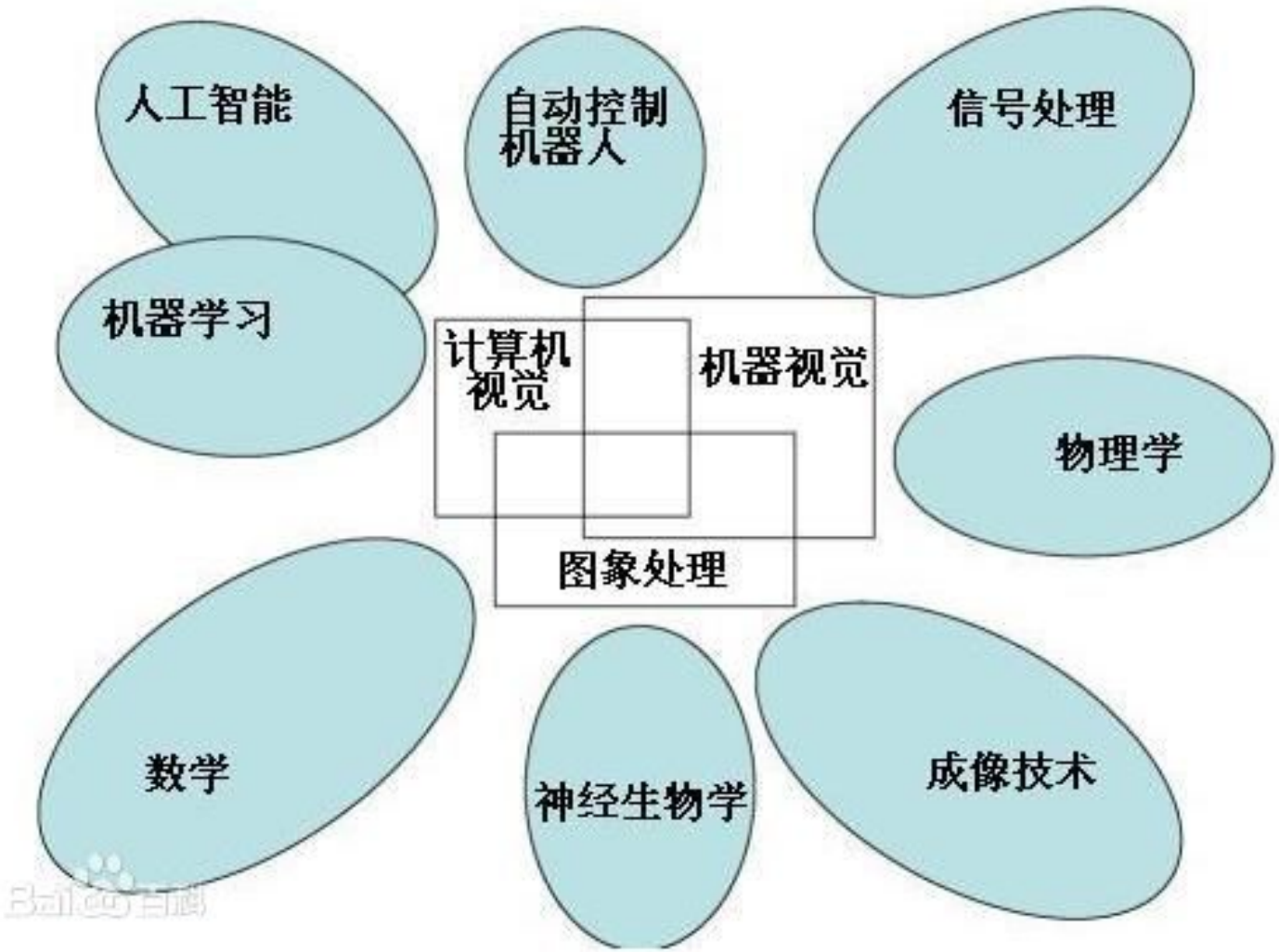
What kind of scene?

Where are the cars?

How far is the building?

…

Credit: James Hays


人工智能
自动控制
机器人
信号处理
机器学习
计算机
视觉
机器视觉
物理学
图象处理
数学
神经生物学
成像技术

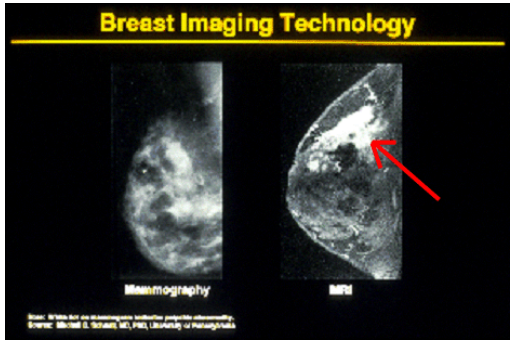
# Vision is really hard

- Vision is an amazing feat of natural intelligence
  - Visual cortex occupies about 50% of Macaque brain
  - One third of human brain devoted to vision (more than anything else)


Credit: James Hays

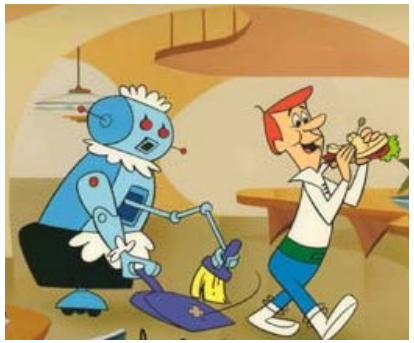
# Why computer vision matters

Safety



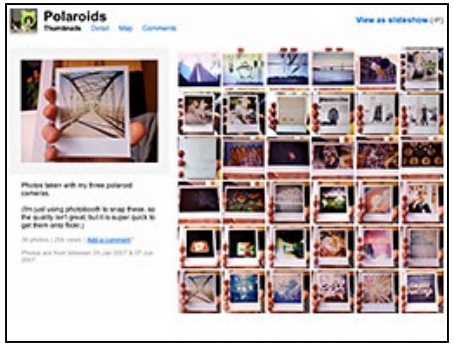

Health

Security



Comfort

Fun

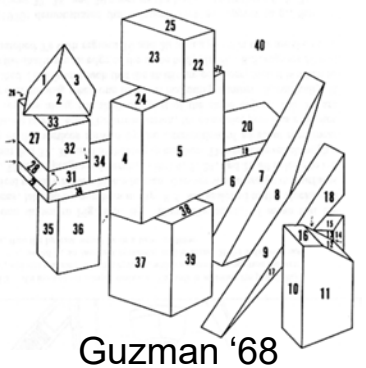



Access

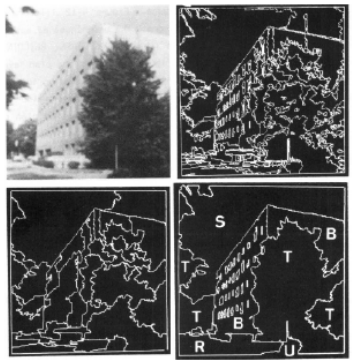
Credit: James Hays

# Ridiculously brief history of computer visior

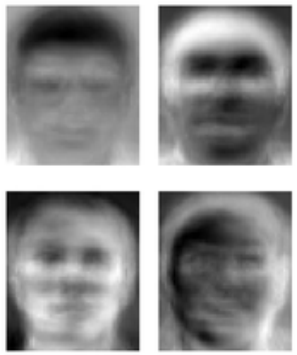
- 1966: Minsky assigns computer vision as an undergrad summer project
- 1960's: interpretation of synthetic worlds
- 1970's: some progress on interpreting selected images
- 1980's: ANNs come and go; shift toward geometry and increased mathematical rigor
- 1990's: face recognition; statistical analysis in vogue
- 2000's: broader recognition; large annotated datasets available; video processing starts
- 2010's: Deep learning with ConvNets
- 2020's: Widespread autonomous vehicles?
- 2030's: robot uprising?



Guzman '68

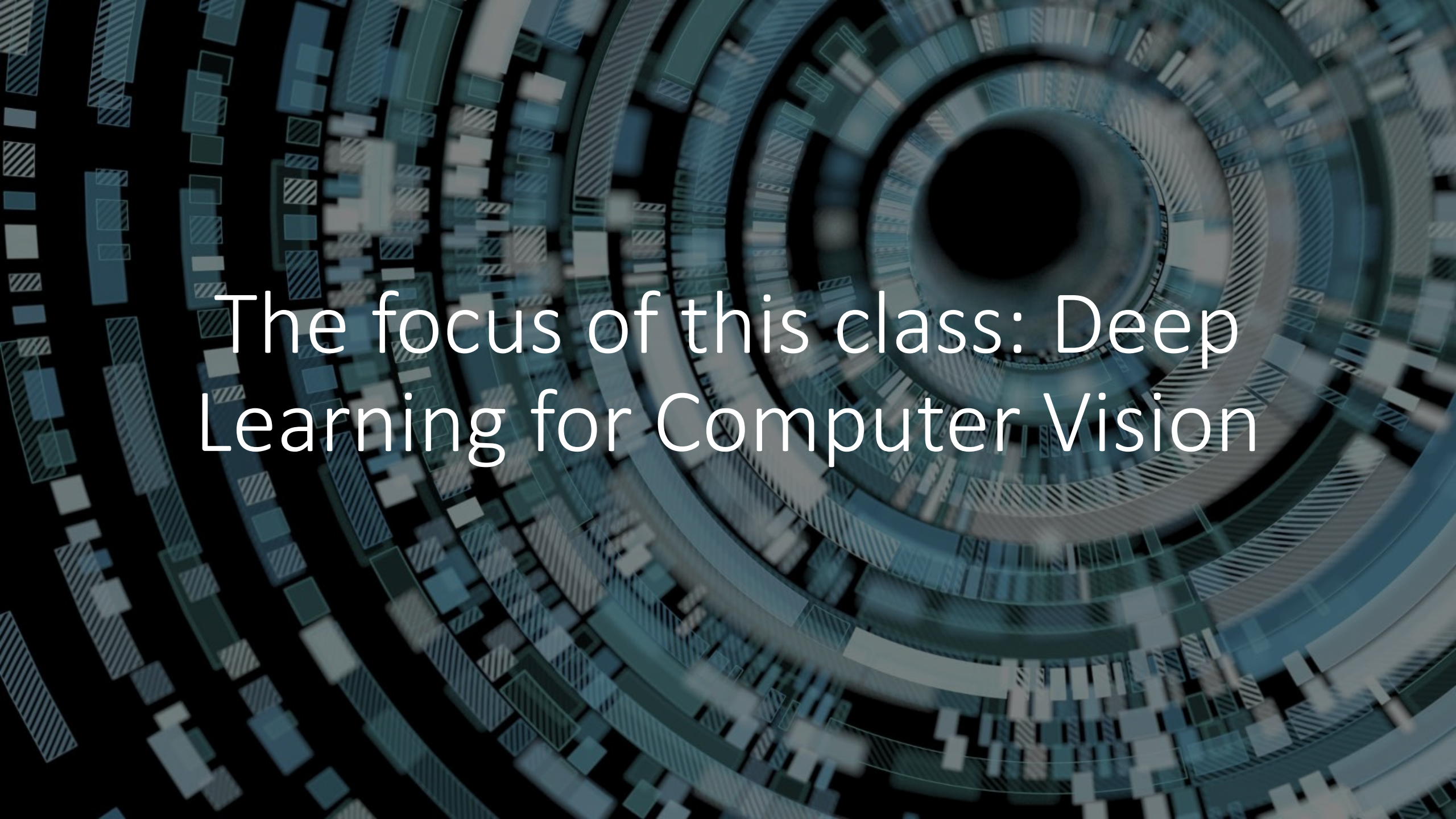


Ohta Kanade '78



Turk and Pentland '91

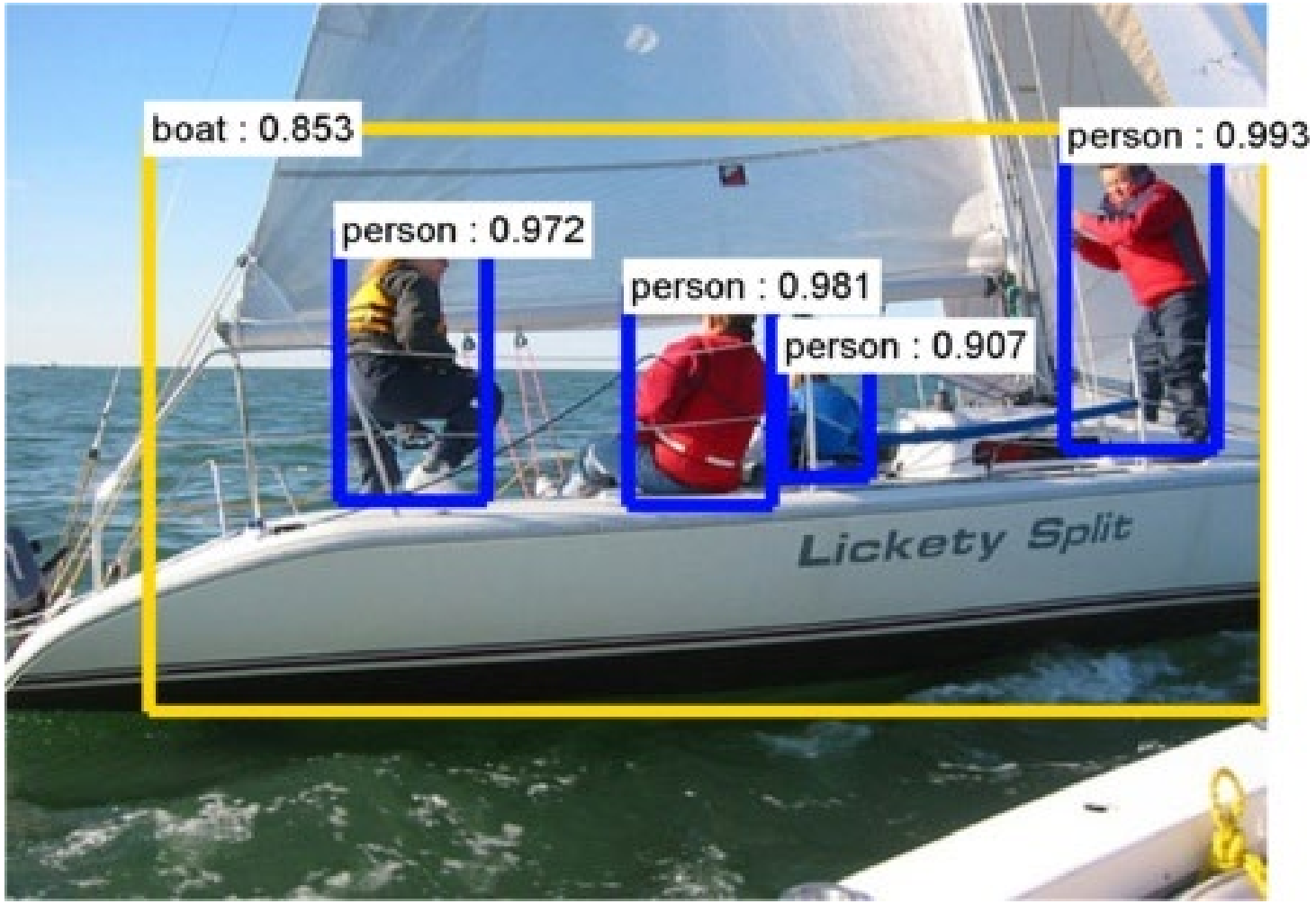
Credit: James Hays


The focus of this class: Deep Learning for Computer Vision

# Outline

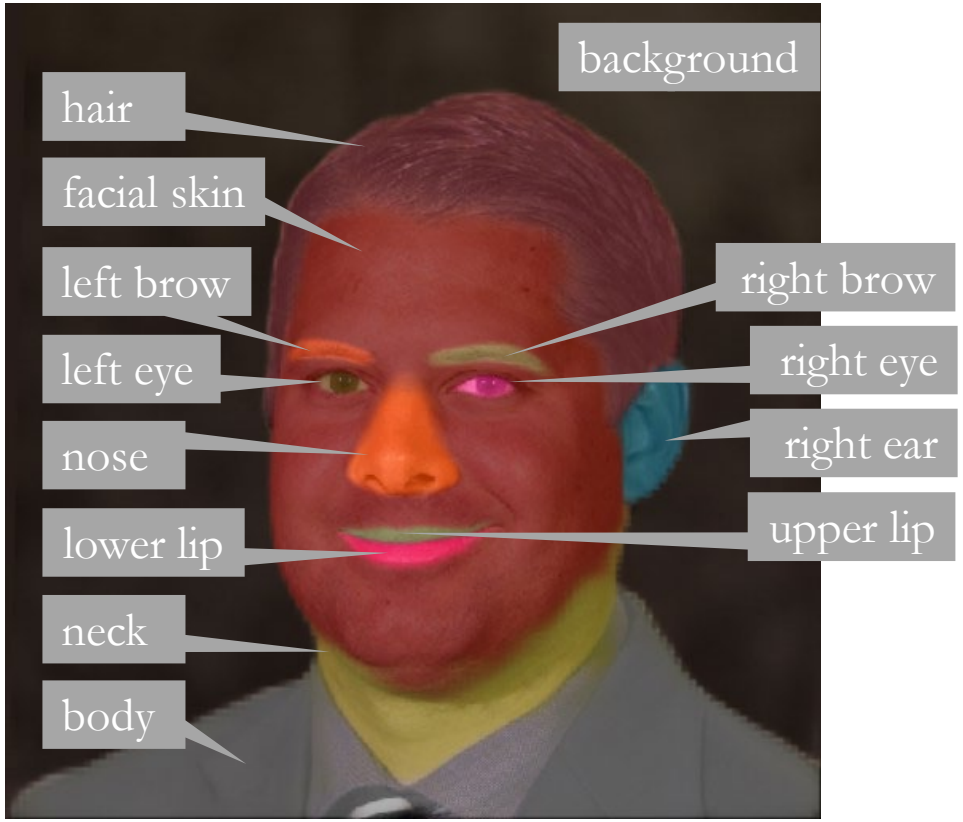
- What can CV do
- CNN Basics and architectures
- Vision Transformer
- Selected tasks
- Self-supervised learning
- Recent New Trends

# Object Detection




# Image Segmentation

# Face Parsing




# Face Aging

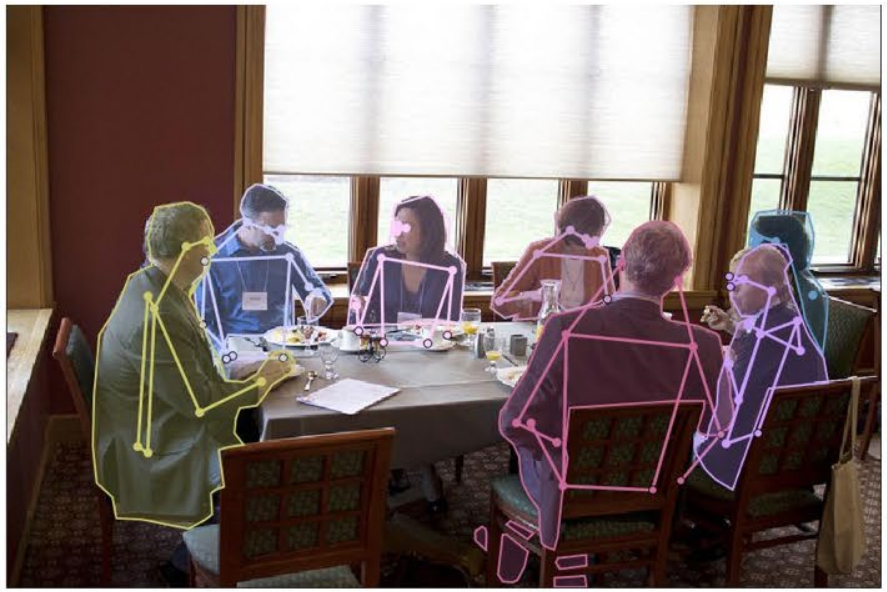
Input

Generated images

21 31 41 51

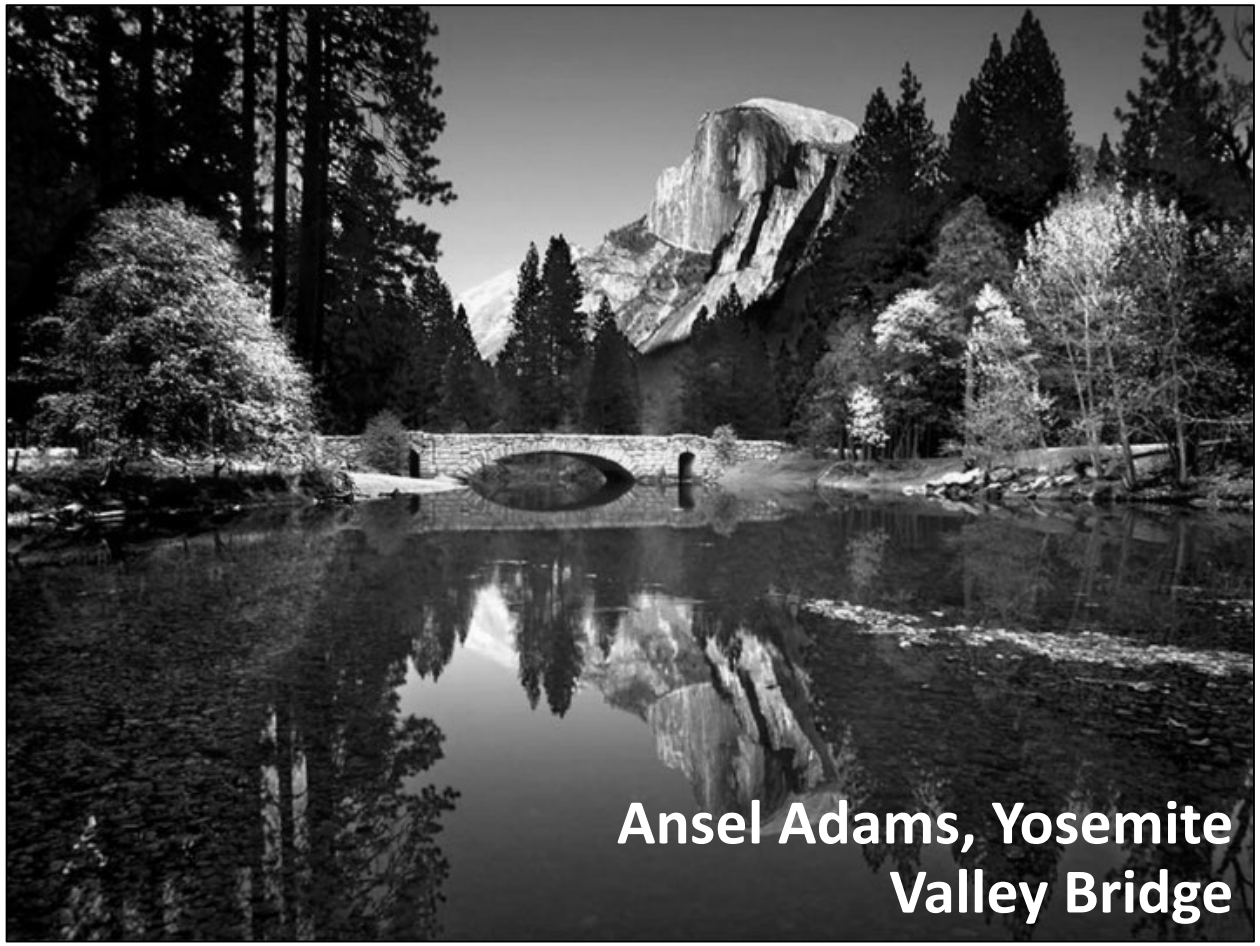
27 37 47 57

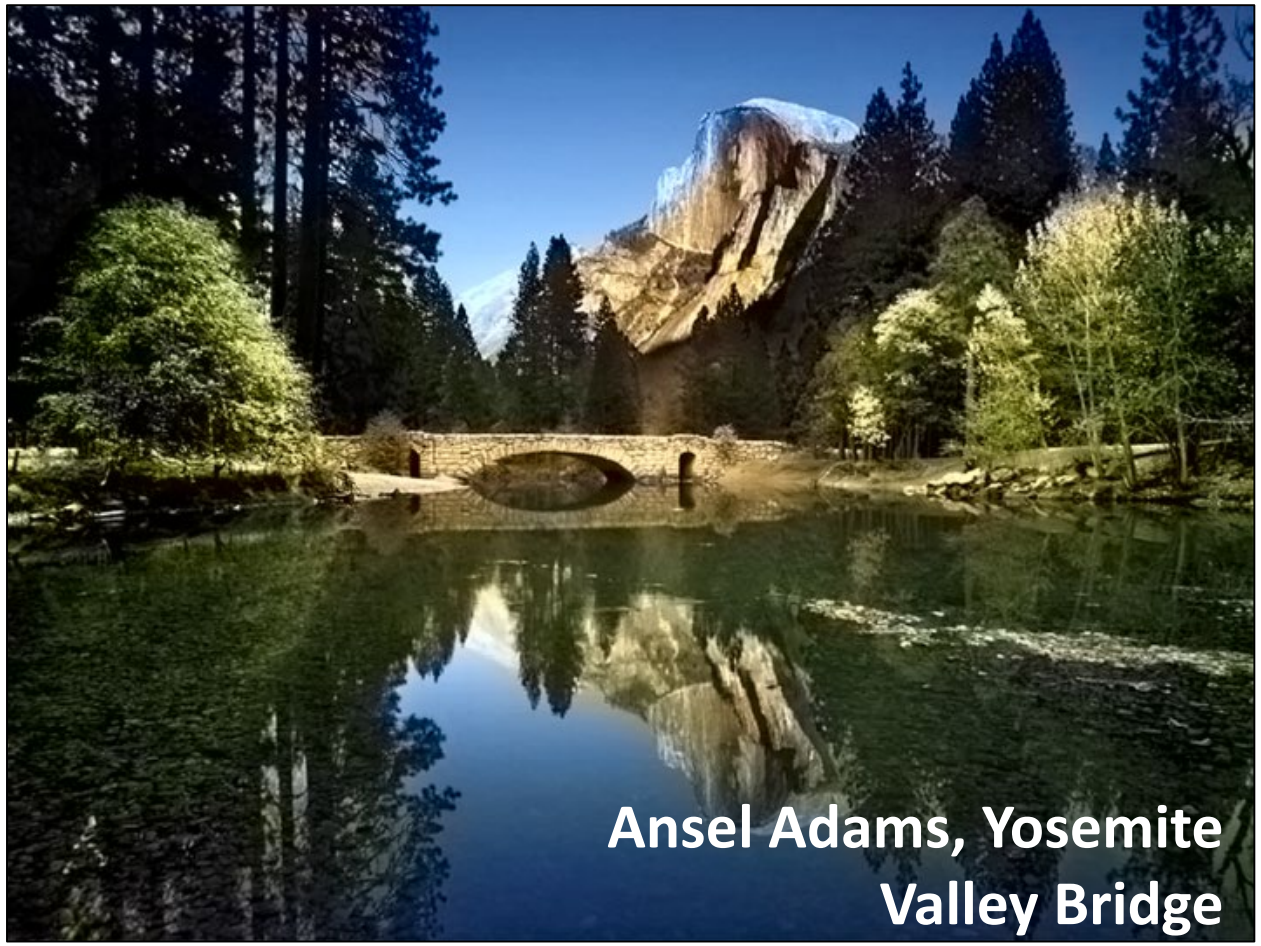
# Keypoint Detection



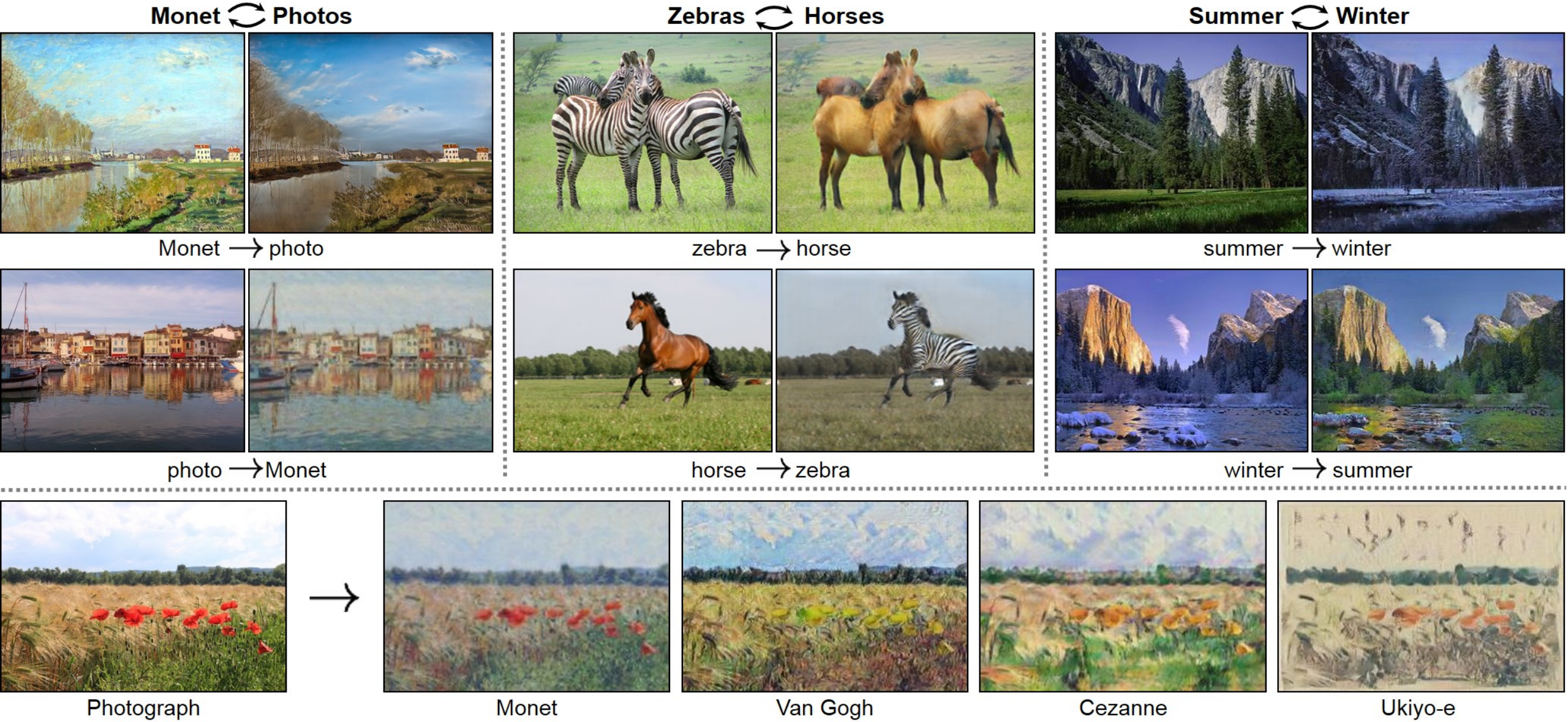



# Colorization



# Style Transfer



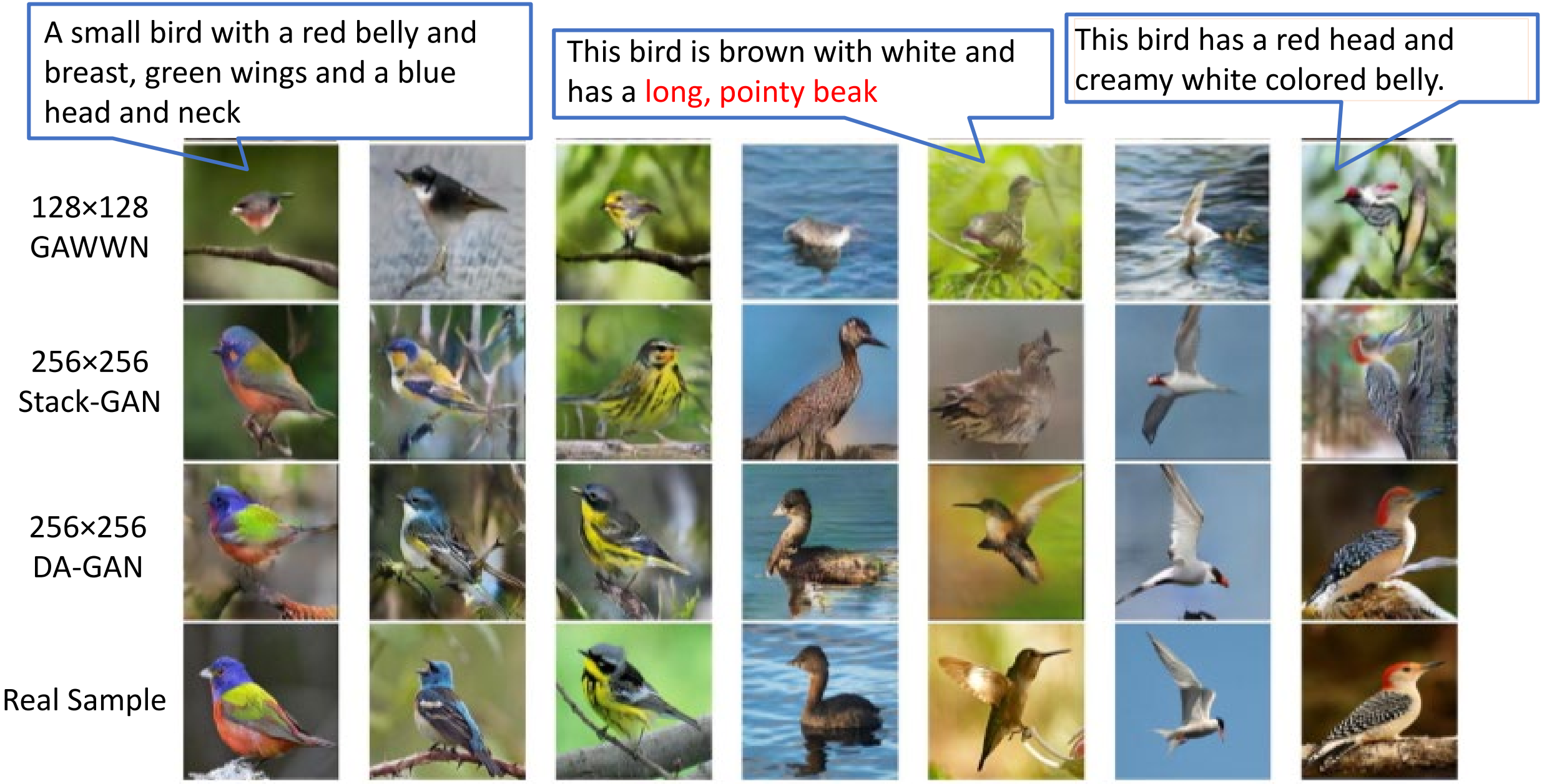
# Optical Character Recognition


# Image Captioning

The man at bat readies to swing at the pitch while the umpire looks on.

A large bus sitting next to a very tall building.

# Text to Image Synthesis



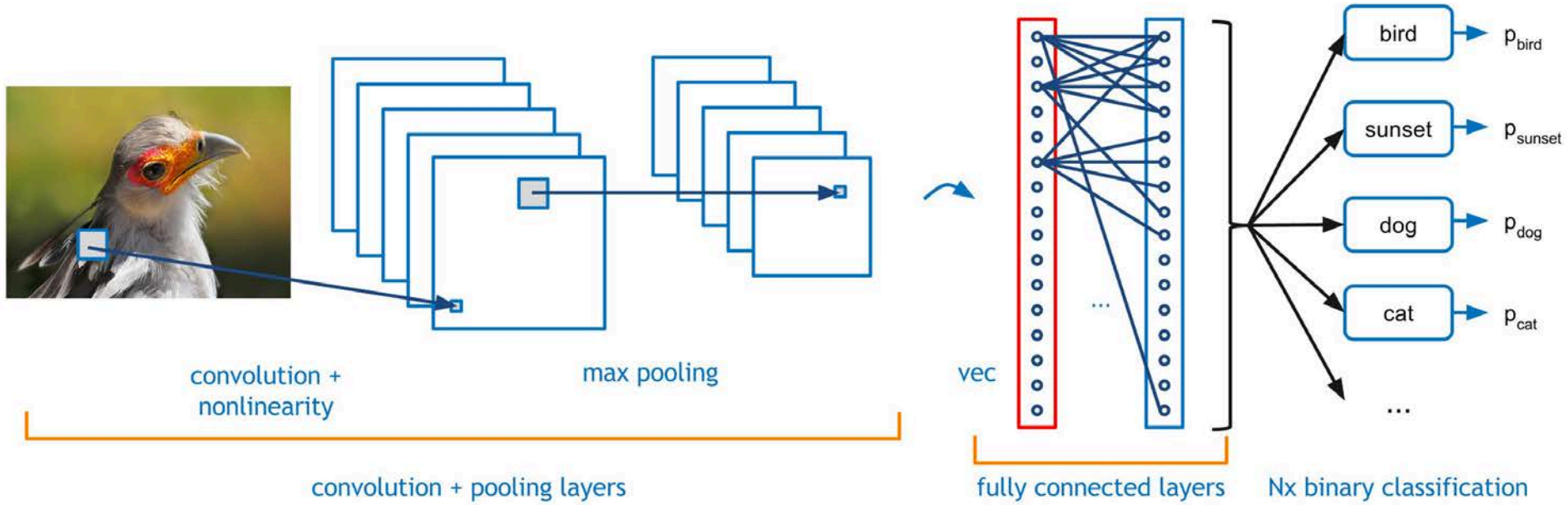
# Part 1: CNN: Basics & Architectures

# Convolutional Neural Networks (CNN)

- Inspired by biology:
  - The visual cortex contains cells that are sensitive to small sub-regions, tiled to cover the entire visual field. These cells act as local filters over the input space and are well-suited to exploit the strong spatially local correlation present in natural images.

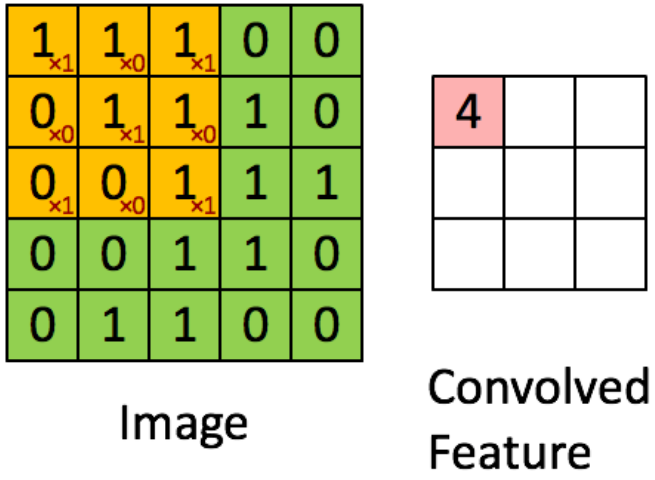
# Convolutional Neural Networks (CNN)

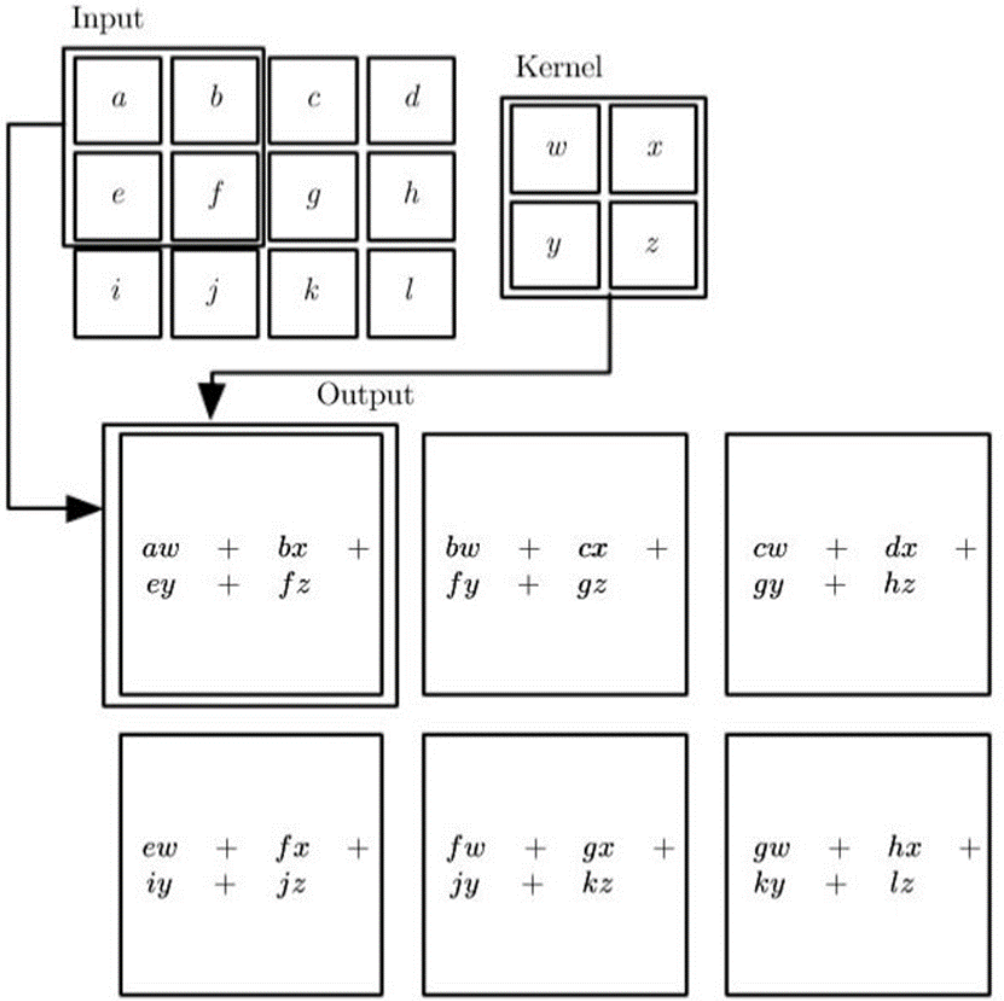
- Key ideas
  - Convolution
  - Weight sharing
  - Pooling
- Layers
  - Convolutional layers
  - Pooling layers
  - Fully connected layers



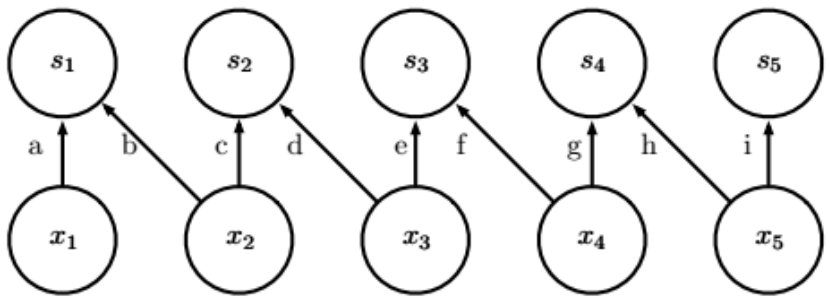
# 2D convolution

- Local connection
- Sparse connection
- Parameter sharing

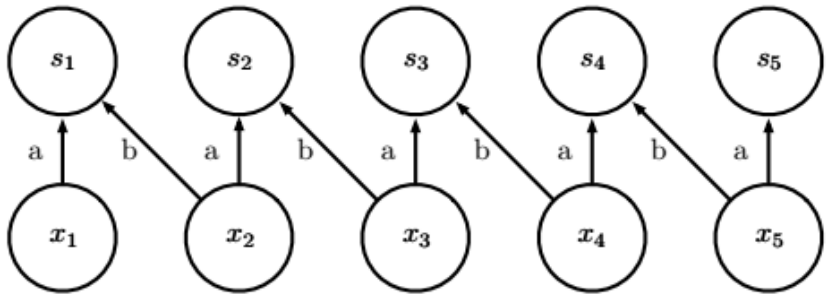


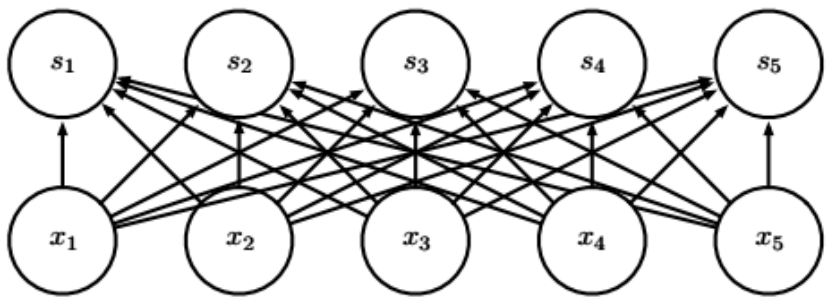

(Goodfellow 2016)




Local connection: like convolution, but no sharing

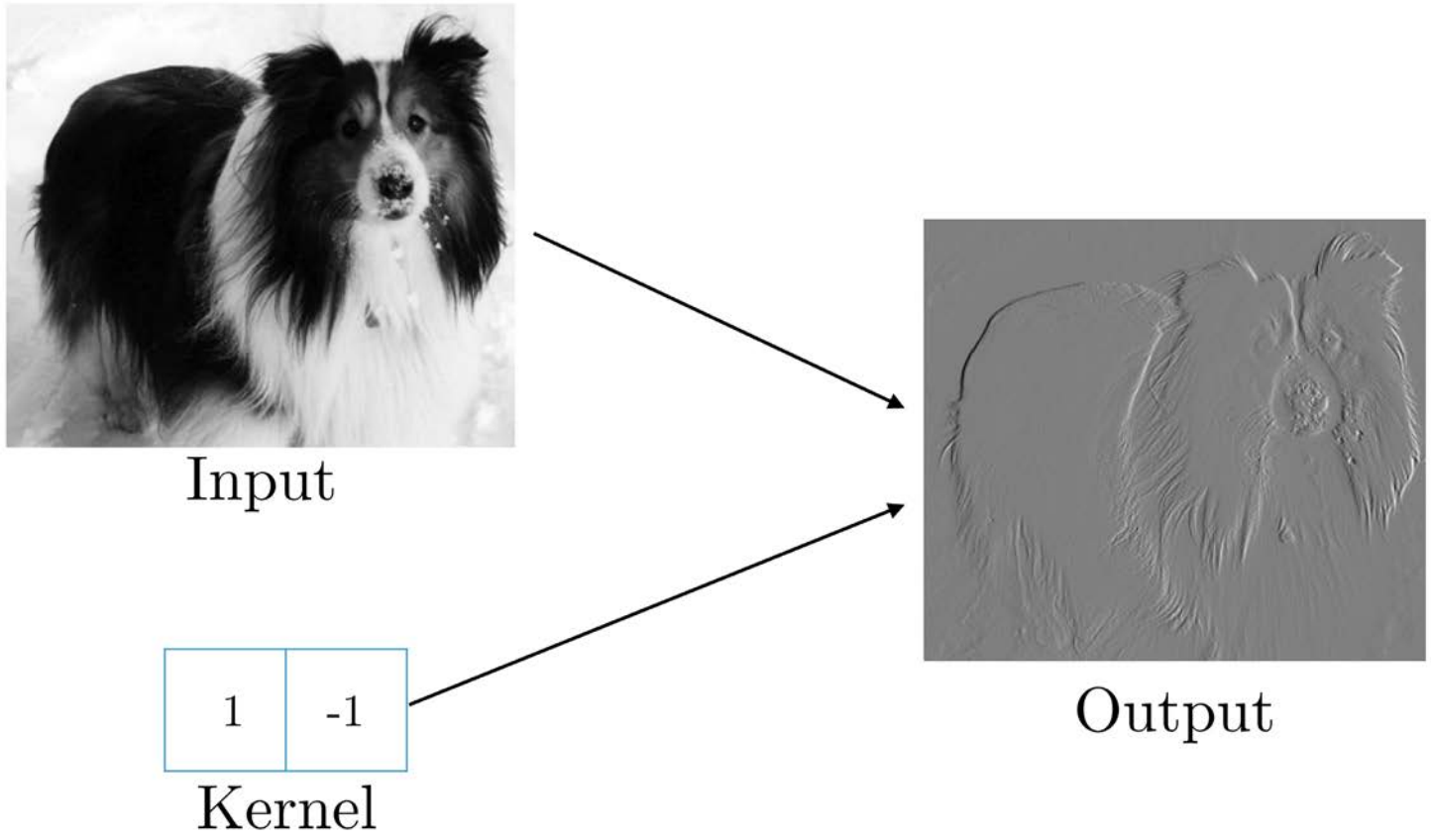



Convolution

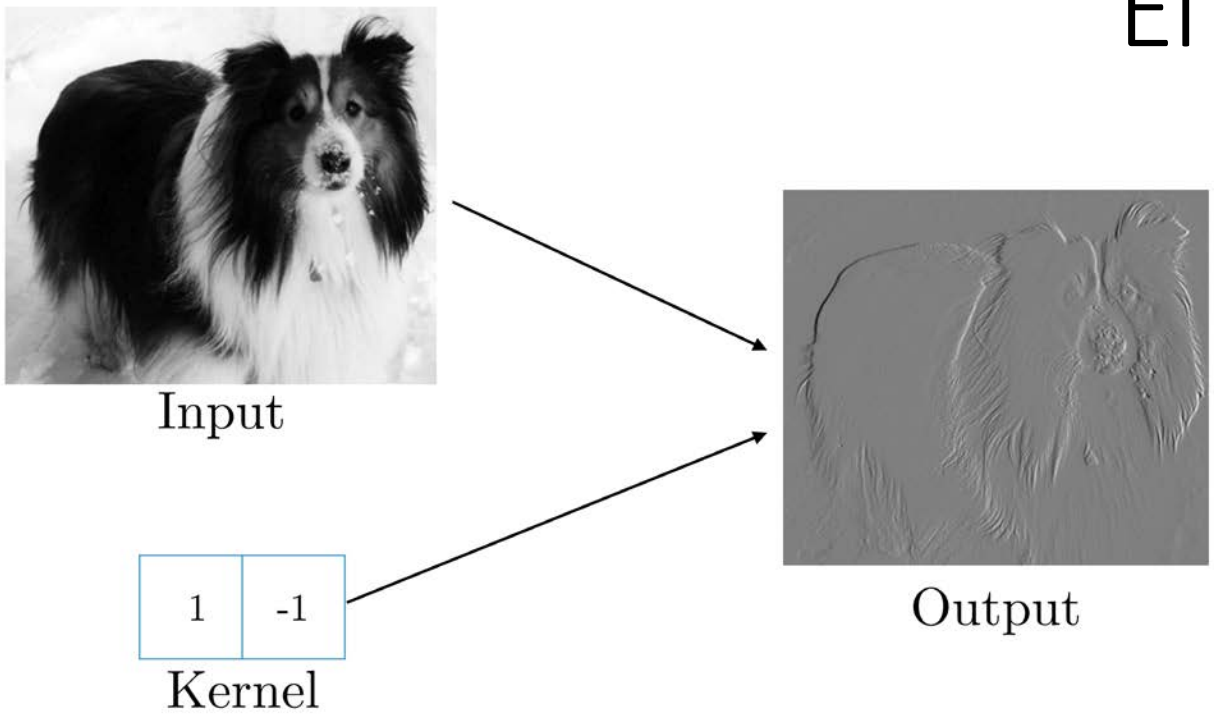



Fully connected

# Edge detection by convolution



# Efficiency of convolution



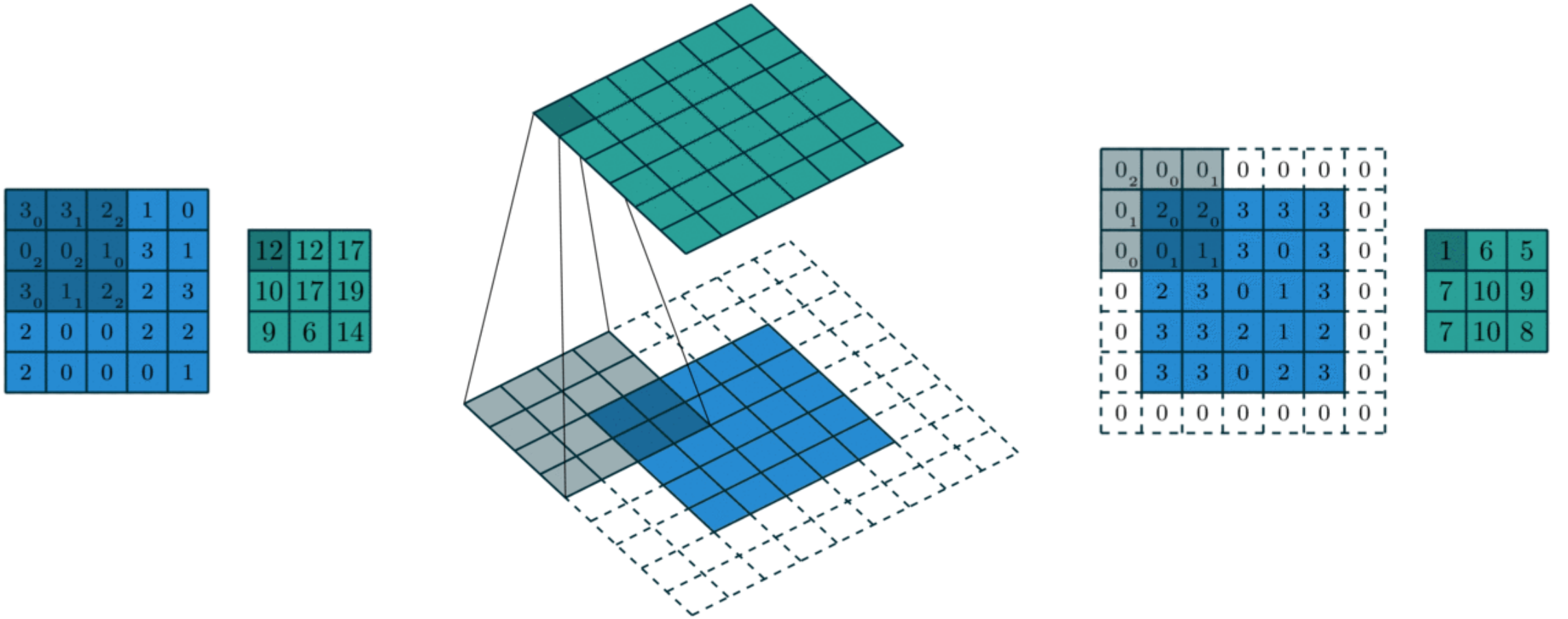

Input size: 320*280

Kernel size: 2*1

Output size: 319*280

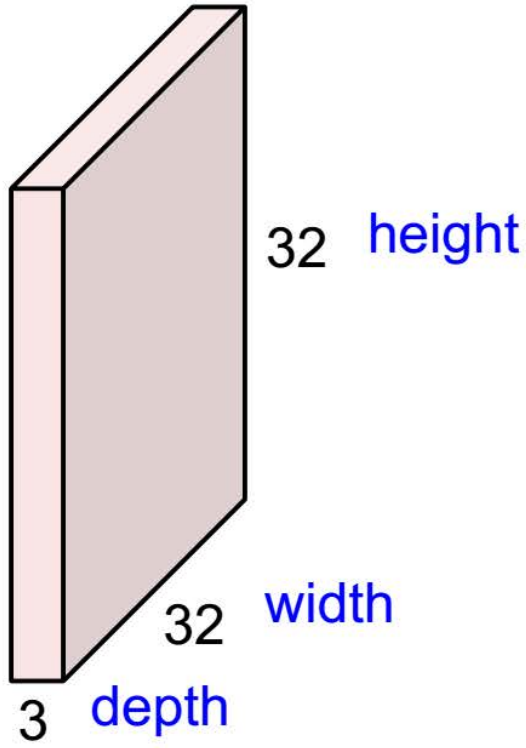
| | Full connection | Sparse connection | Convolution |
|---|---|---|---|
| Model size | 319*280*320*280 >8e9 | 2*319*280 =178,640 | 2 |
| Float operation | >8e9 | 319*280*3 = 267,960 | 319*280*3 = 267,960 |

# Convolution, Padding, Stride



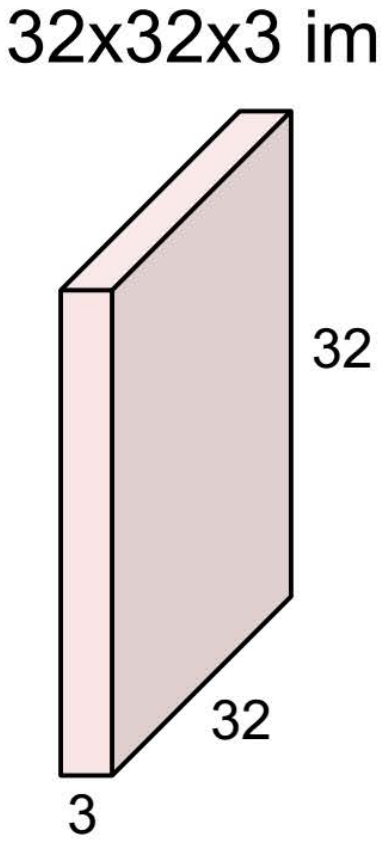
# 3D convolution

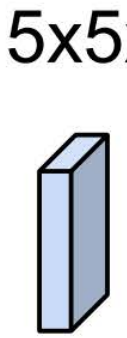
32x32x3 image -> preserve spatial structure




Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

# 3D convolution

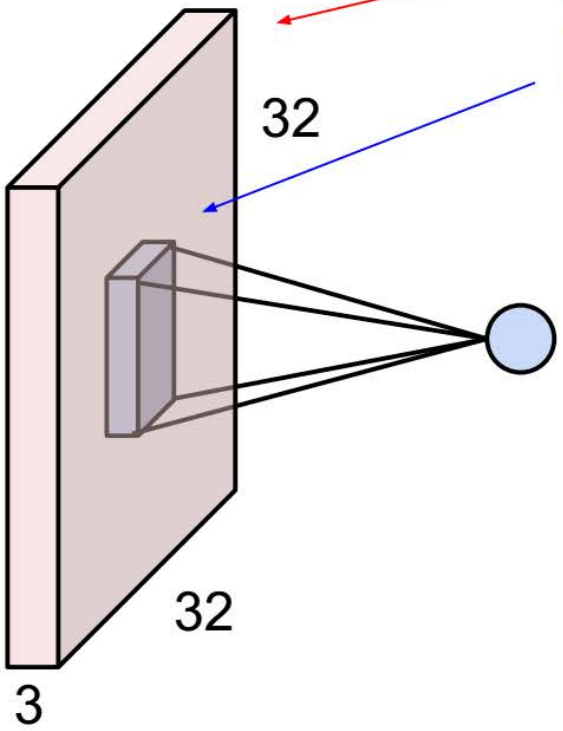



**Convolve** the filter with the image
i.e. “slide over the image spatially,
computing dot products”

Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

# 3D convolution




32x32x3 image

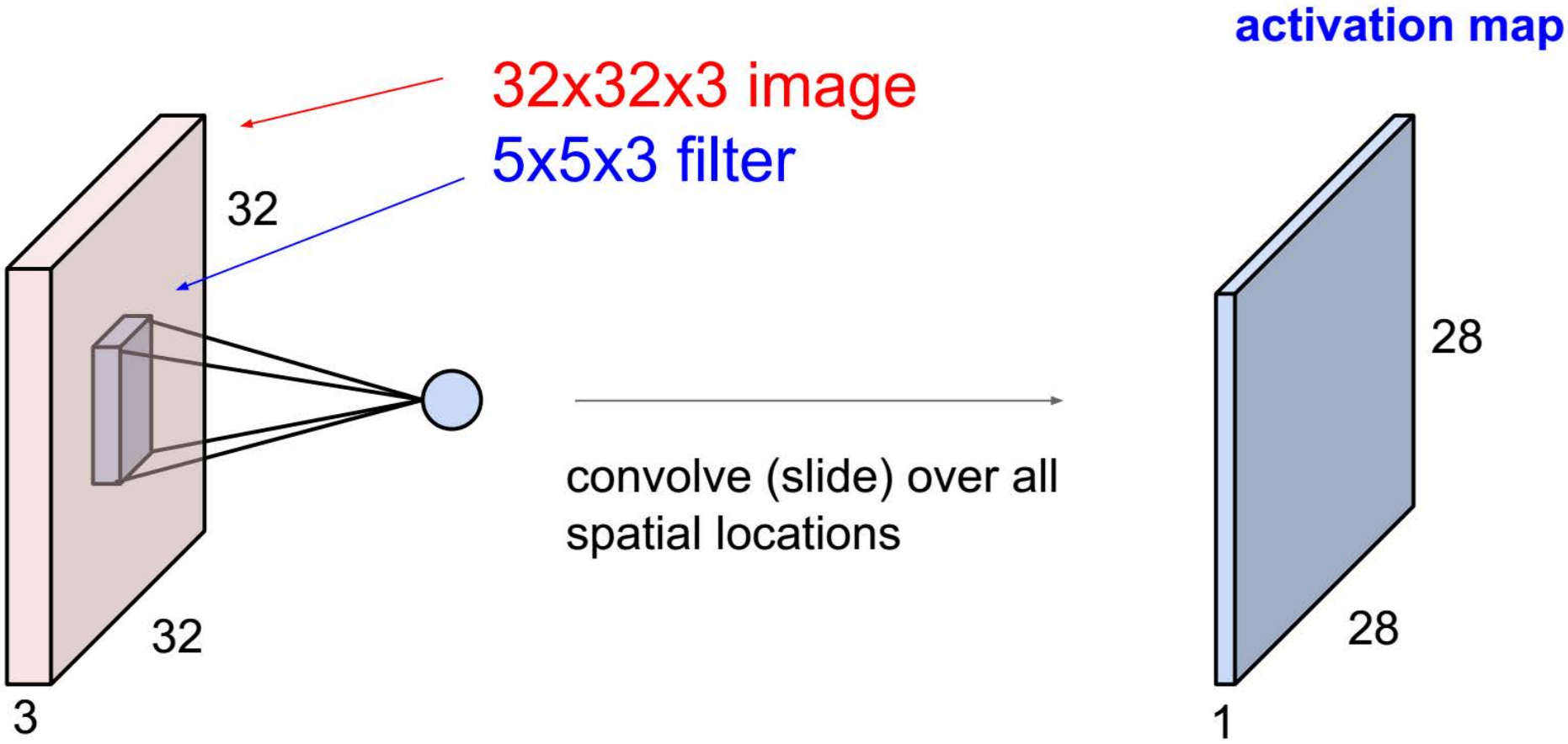
5x5x3 filter $w$

**1 number:**
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

# 3D convolution




Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

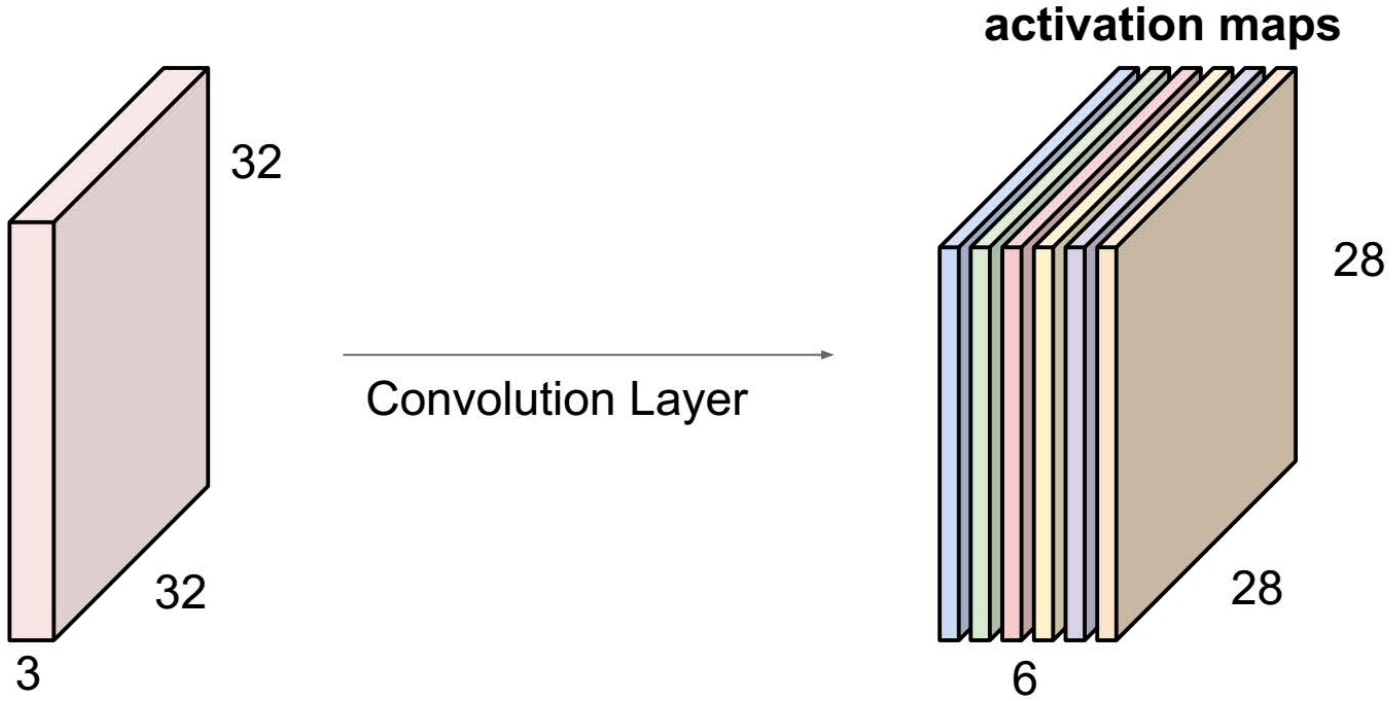
# 3D convolution

Second filter -> Second layer output


Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

# 3D convolution

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:




We stack these up to get a "new image" of size 28x28x6!

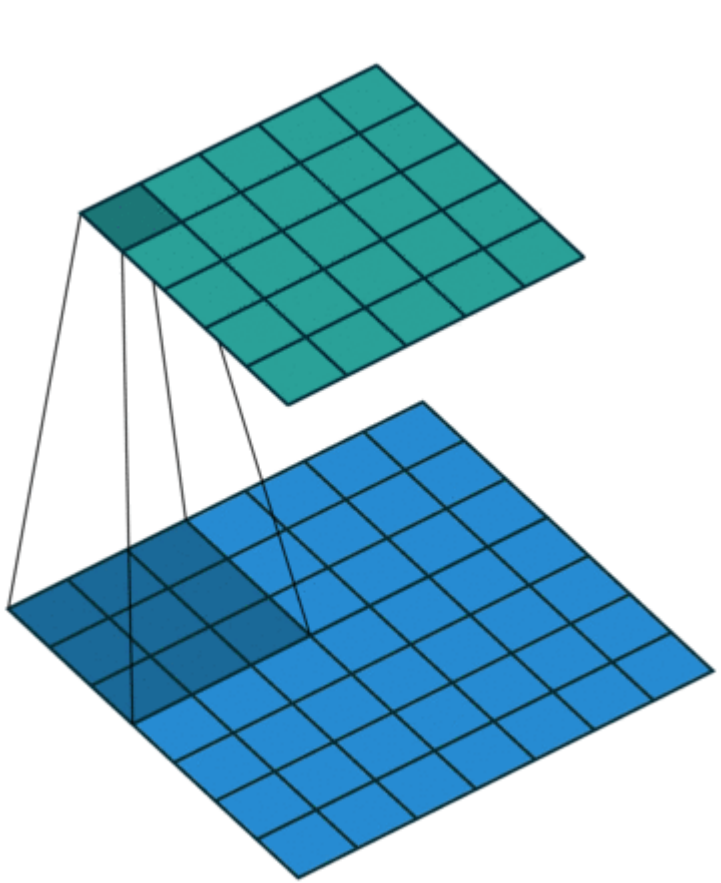
Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf

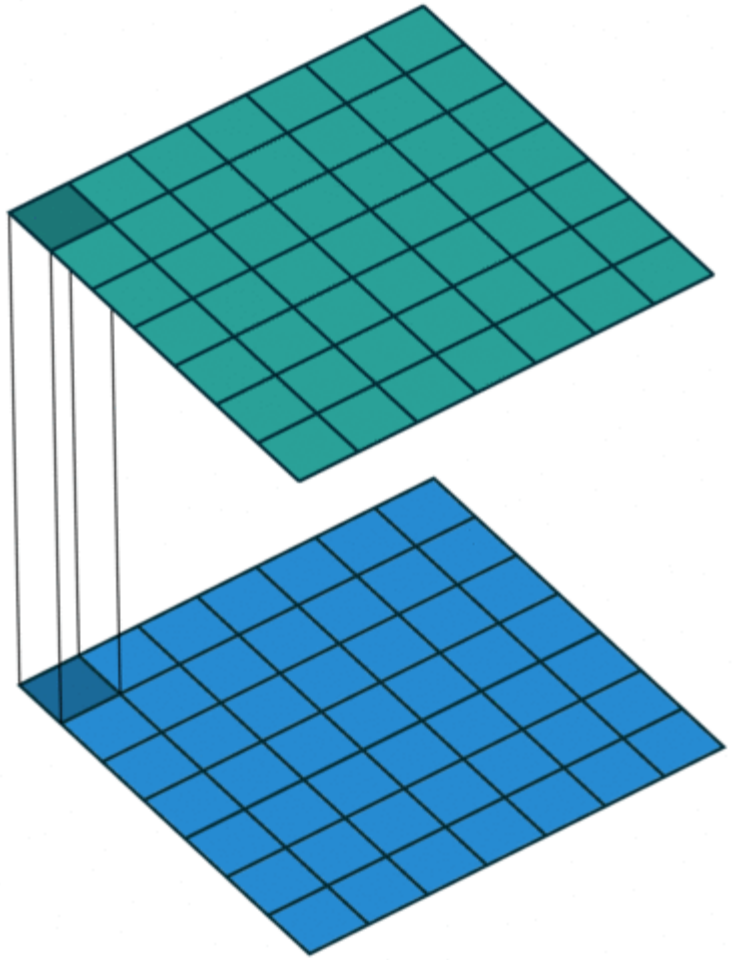
# Summary of Conv Layer

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.

# 1x1 Convolution

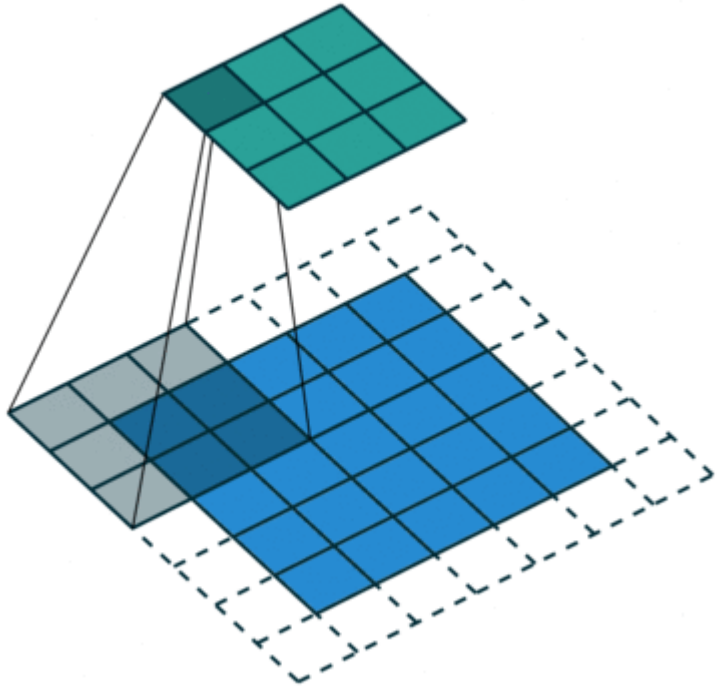
Feature dimension reduction, e.g., input 100x100x50 + 20 1x1 filters ➔ 100x100x20

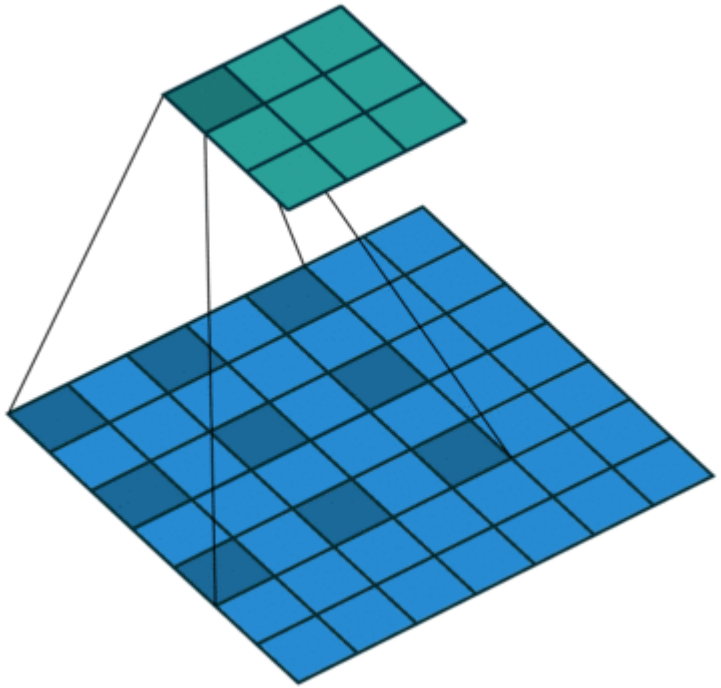




# Dilated Convolution

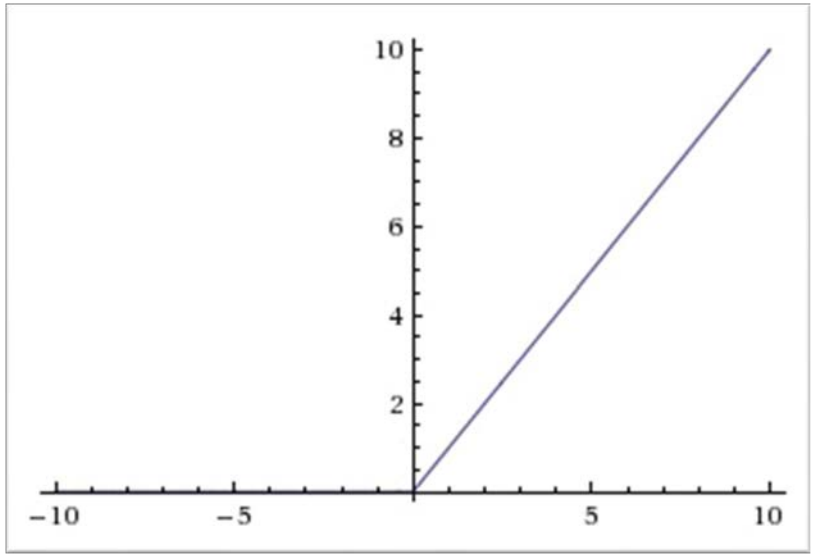
Larger receptive field with less layers





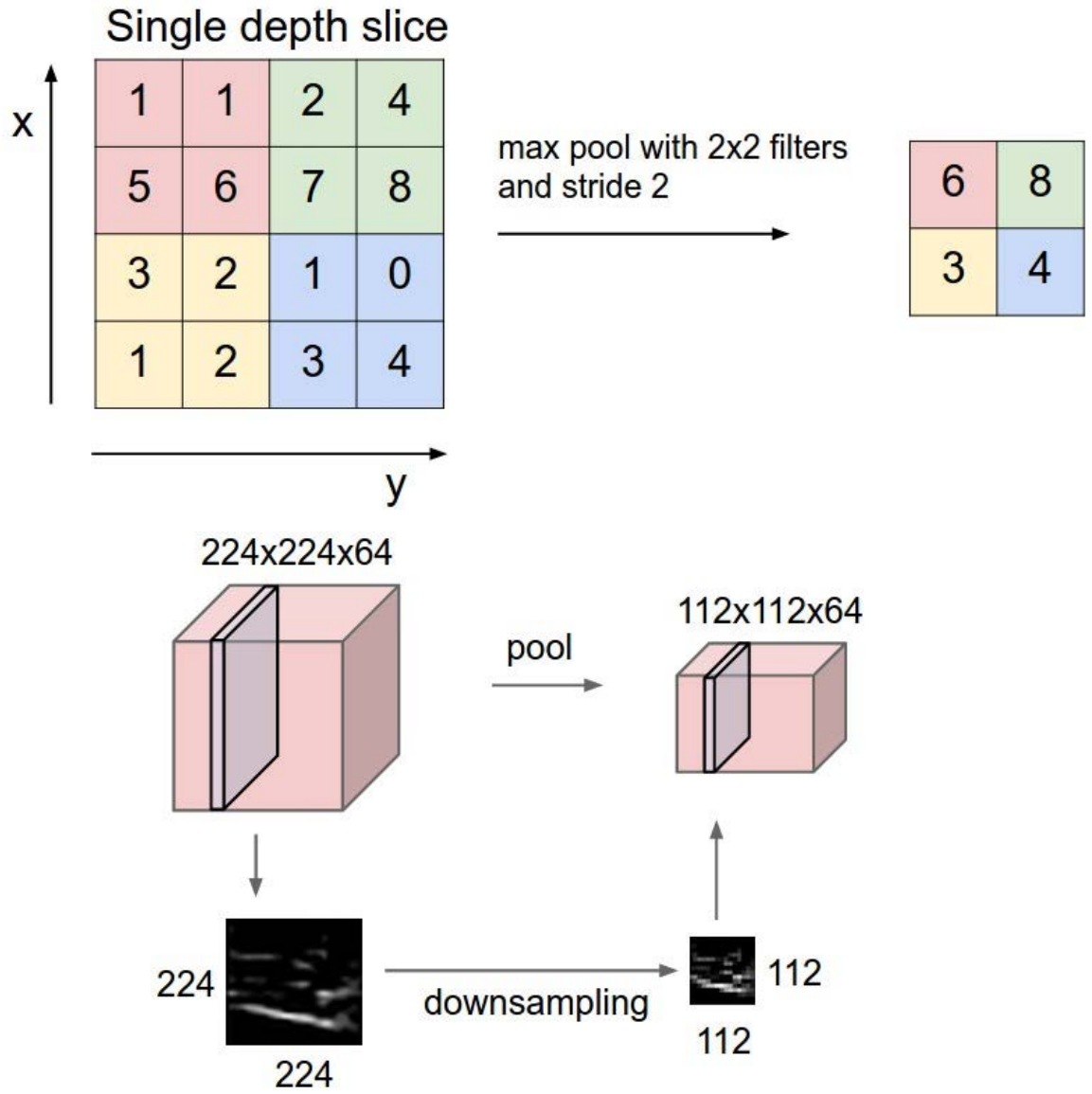
# Activation

- Applied after convolution
- Nonlinear transformation
- ReLU as a popular choice $\max\{0, x\}$

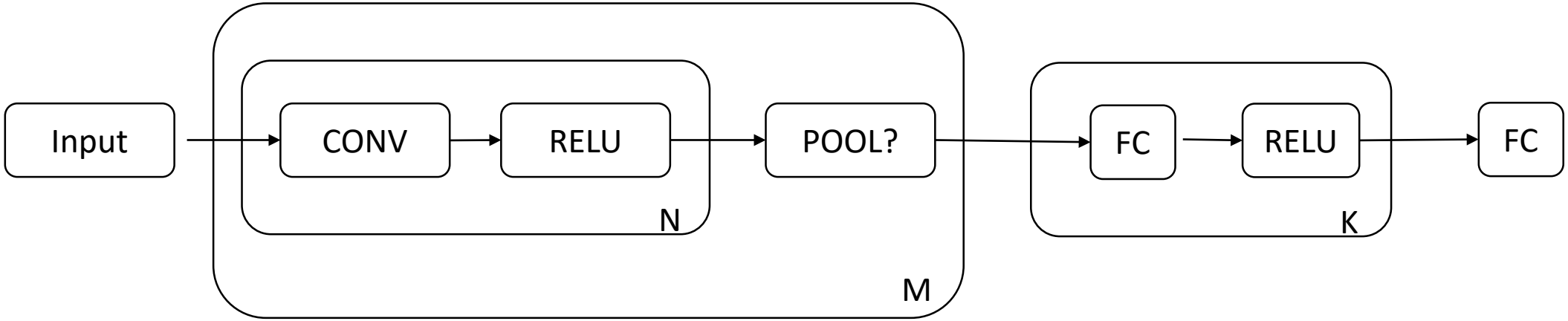


# Pooling

- Reduce dimension
  - Control overfitting to some extent
- Achieve translation invariance
- Popular choice: MAX pooling



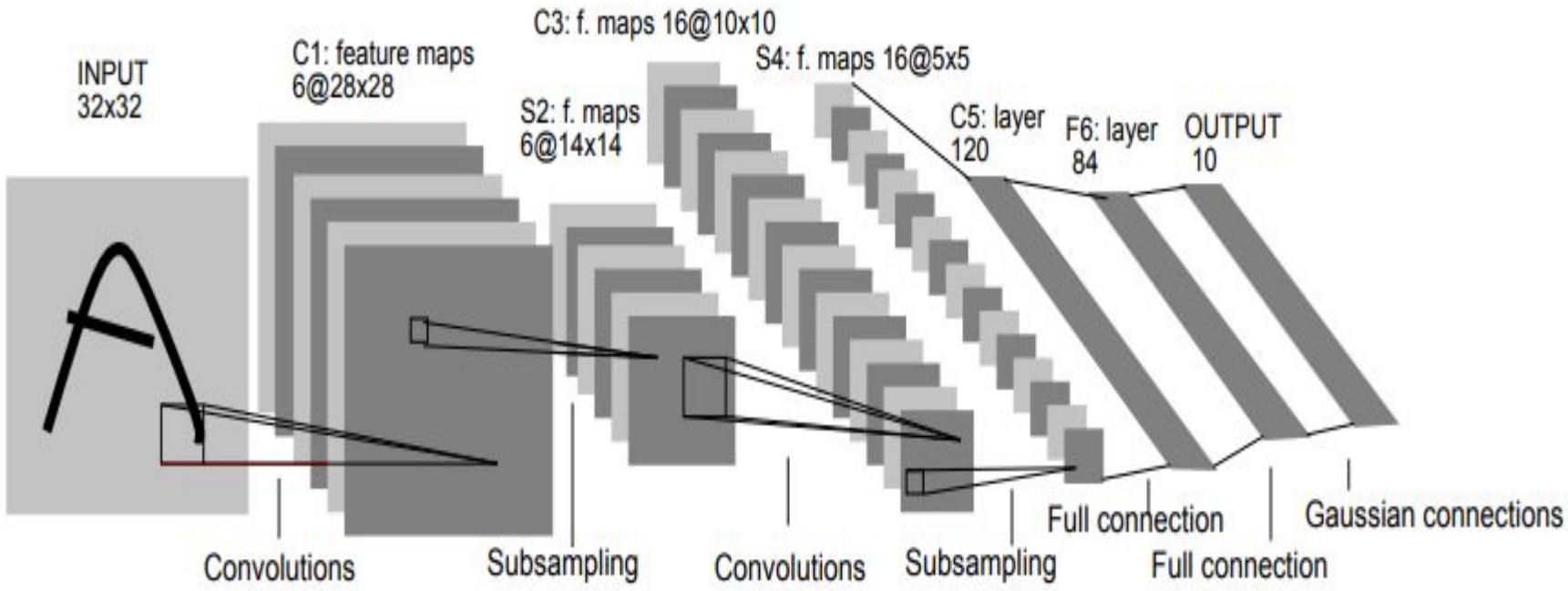
# ConvNet architectures




- Exceptions
  - Google's Inception architectures
  - (state of the art) Residual Networks from Microsoft Research Asia

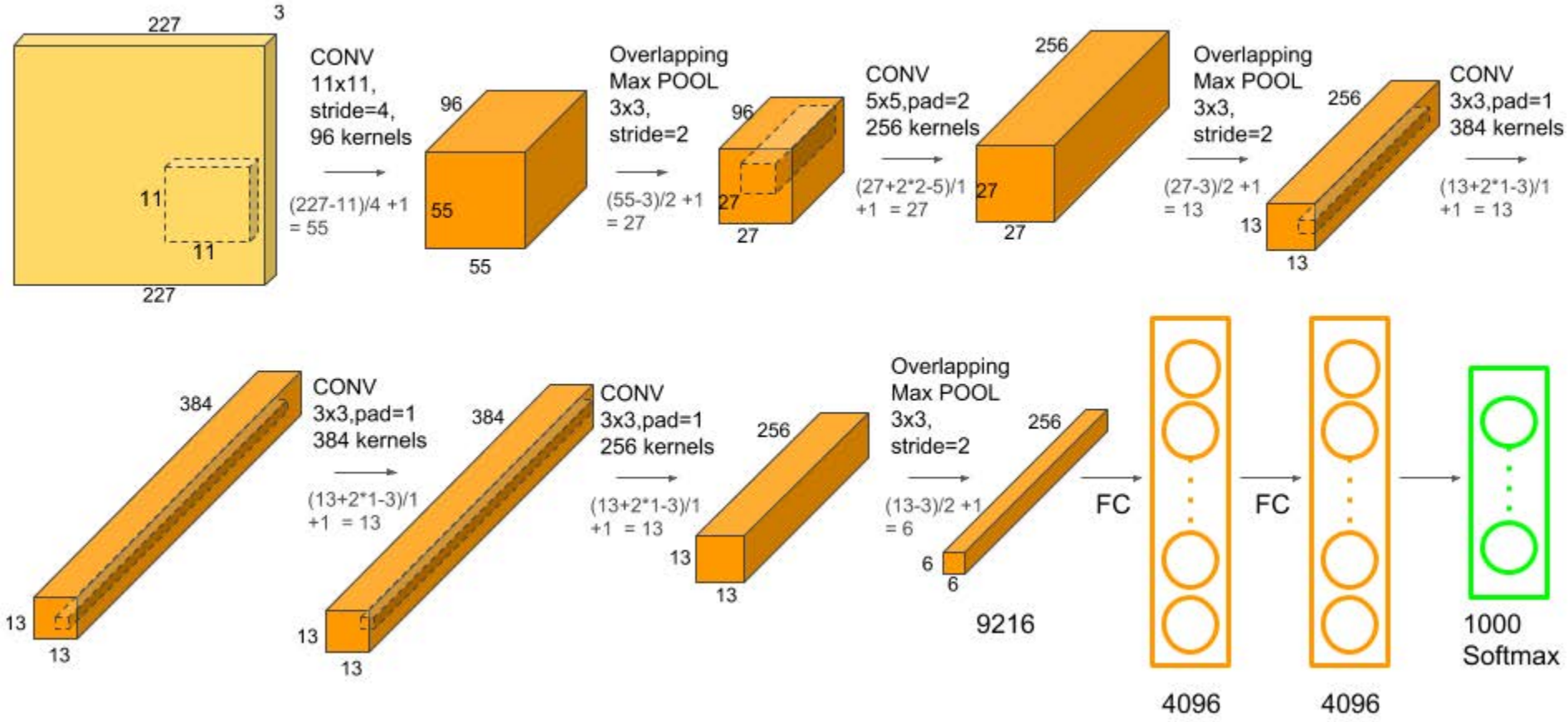
# LeNet

- Convolution
- Pooling
- FC




Gradient-based learning applied to document recognition, LeCun et al. 1998

# AlexNet

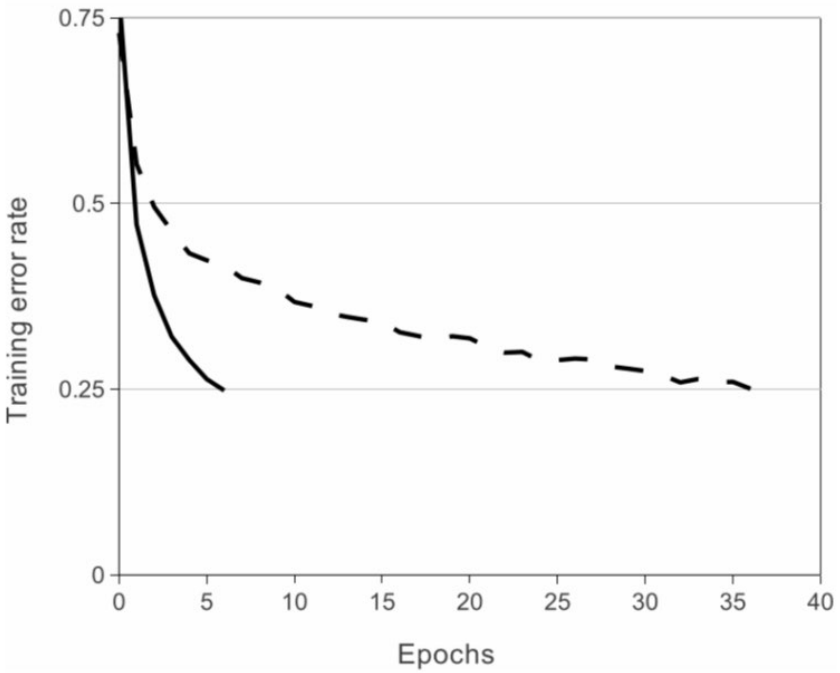


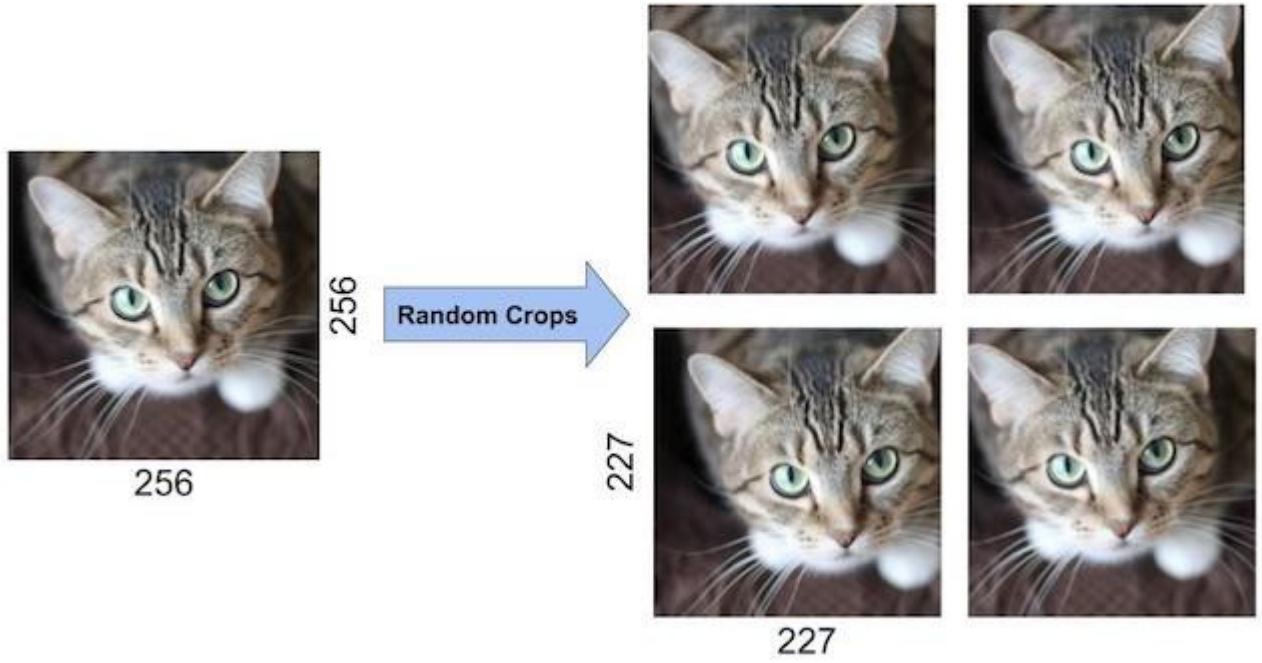

ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky, Sutskever, Hinton. NIPS 2012

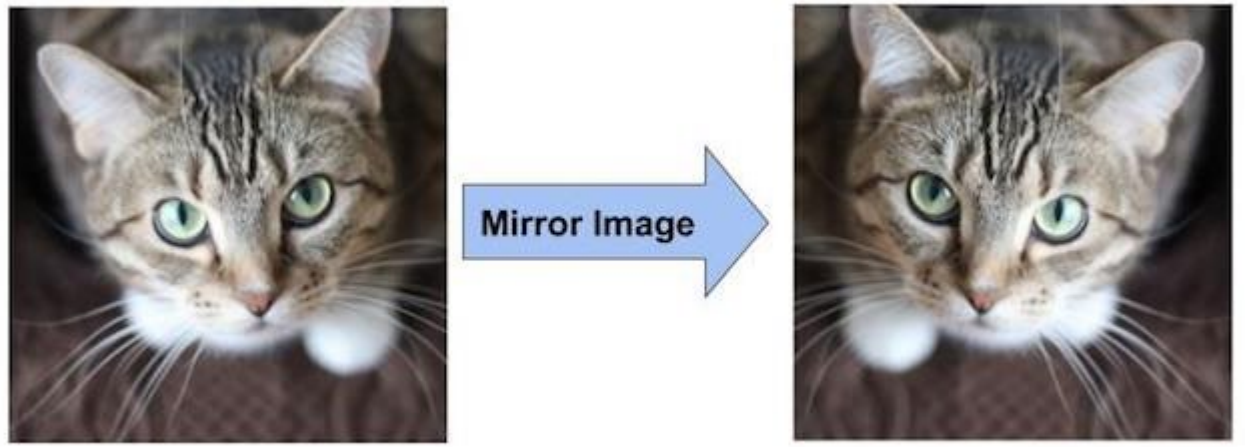
Photo credit: https://www.learnopencv.com/understanding-alexnet/

# AlexNet

- Overlapping max pooling
- ReLU
- Data augmentation

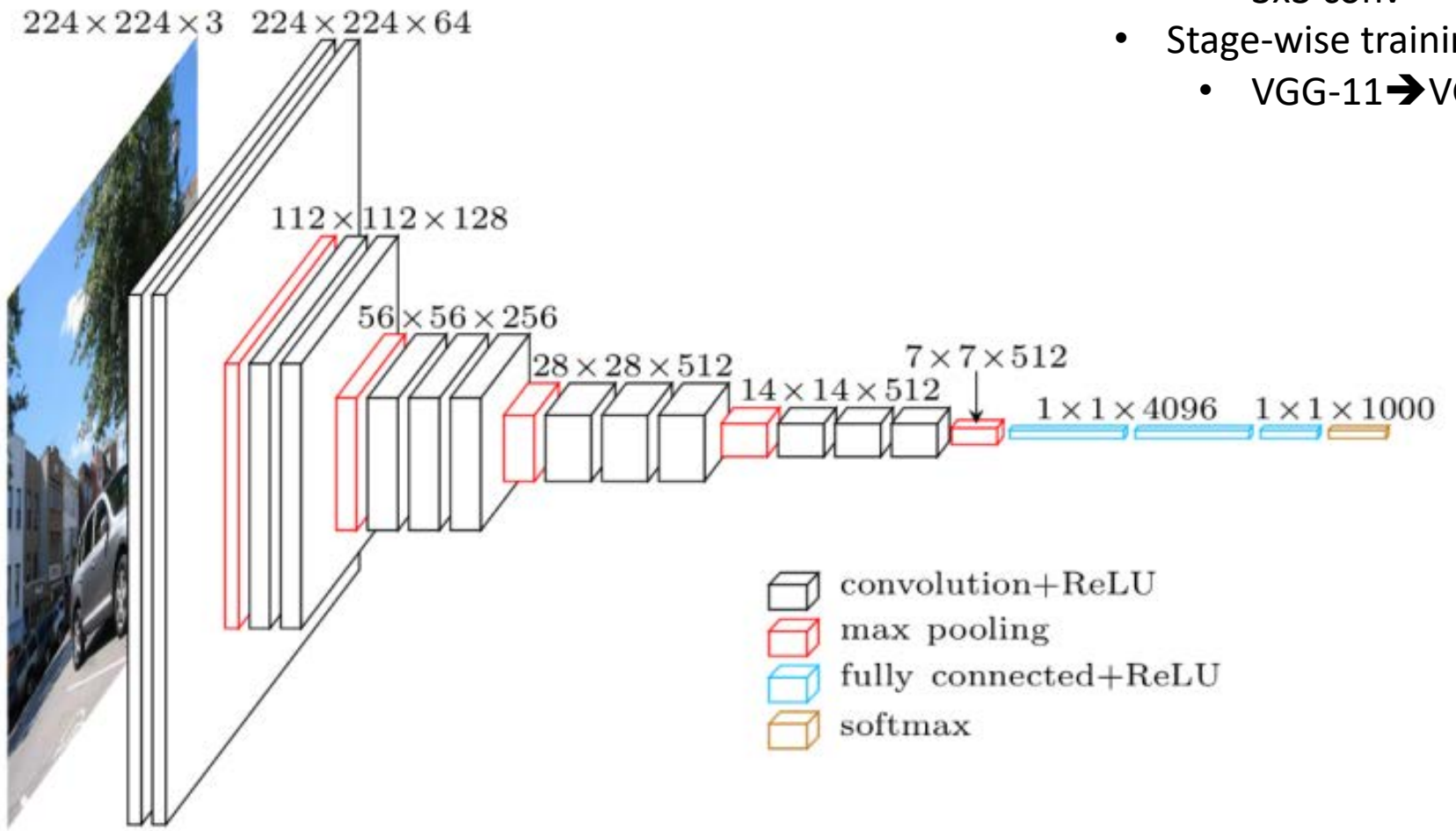


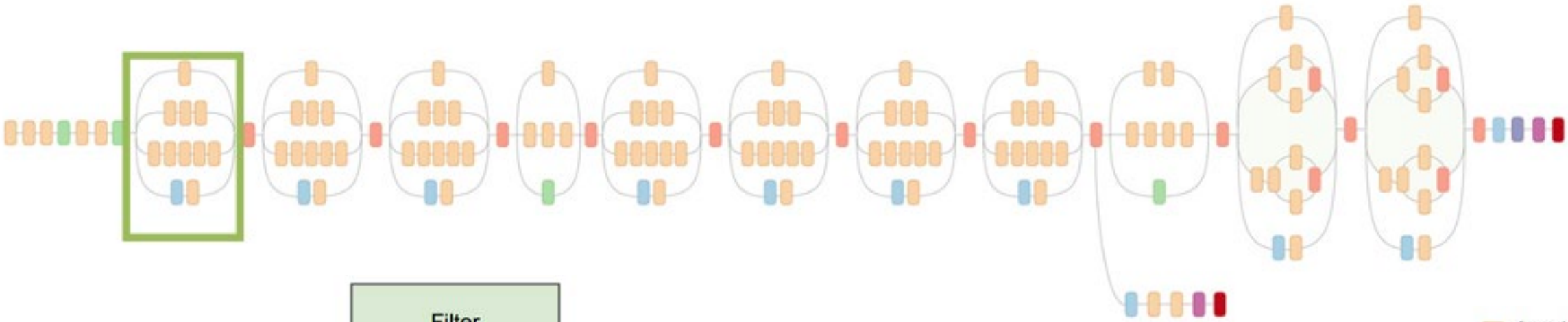
# VGG-16/19

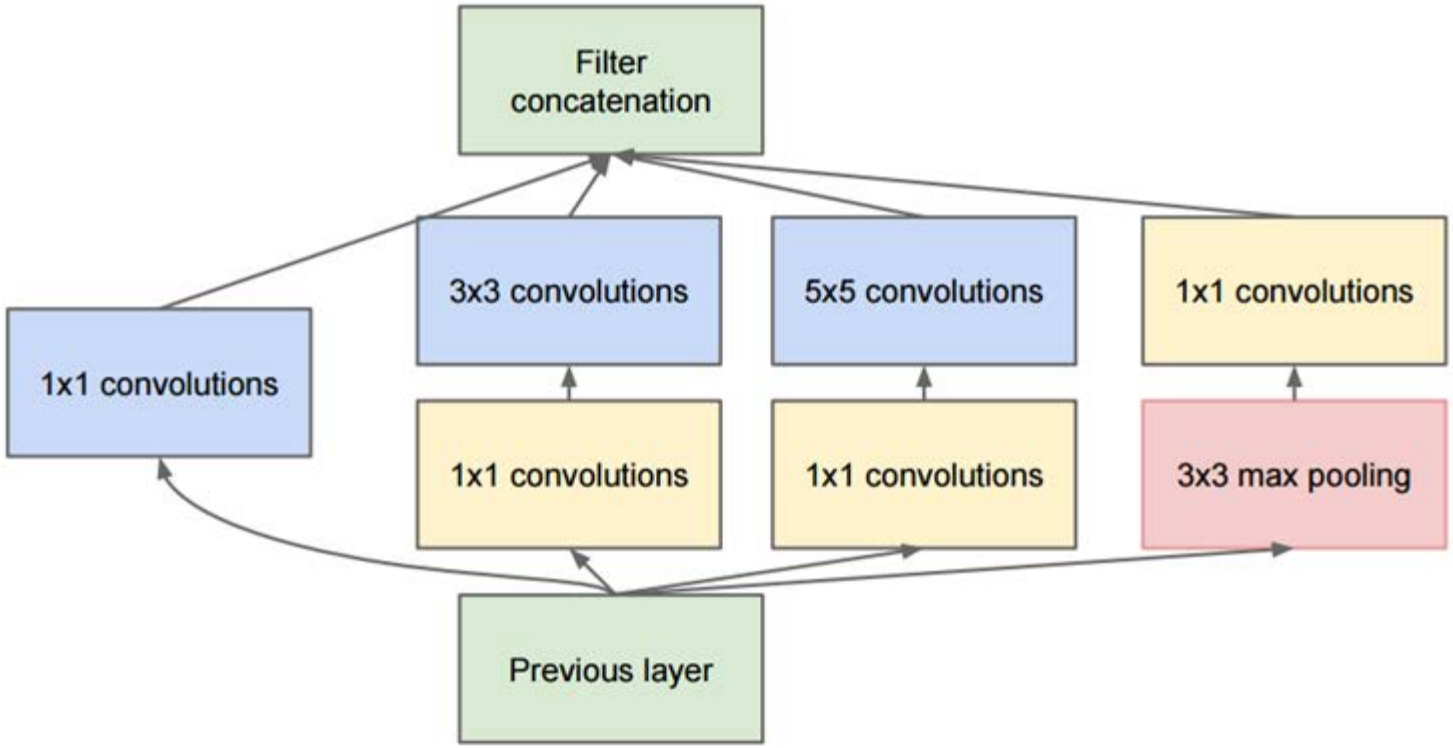
- Modularized design
  - 3x3 conv
- Stage-wise training
  - VGG-11➔VGG-13➔VGG-16




Very Deep Convolutional Networks for Large-Scale Image Recognition, Simonyan & Zisserman. arXiv 2014 (ICLR 2015)

# GoogleNet/Inception




Full Inception module

- Multiple branches
- Dimension reduction by 1x1 conv

# Batch Normalization

**Input:** Values of $x$ over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: $\gamma$, $\beta$
**Output:** $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

**Algorithm 1:** Batch Normalizing Transform, applied to activation $x$ over a mini-batch.

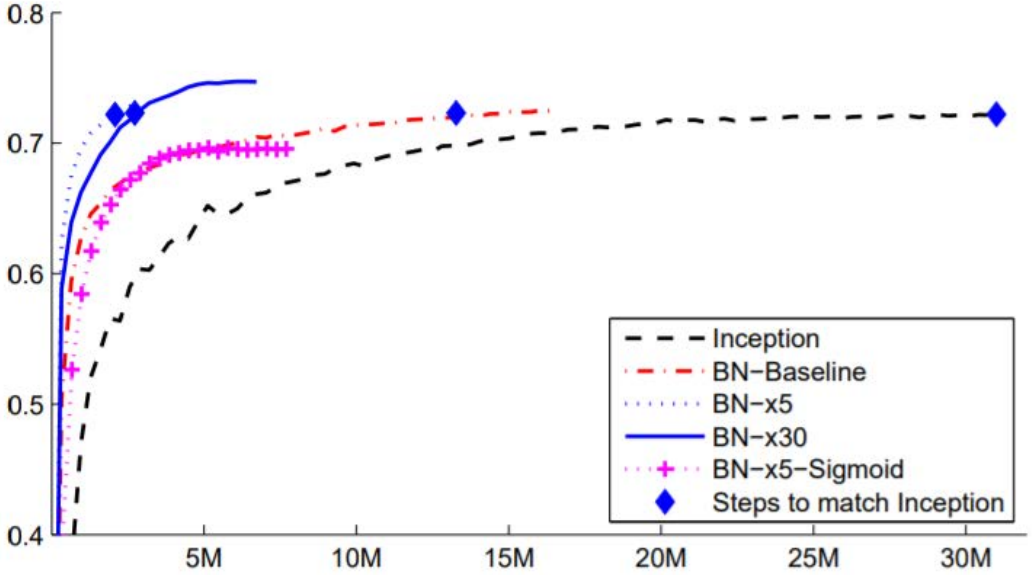
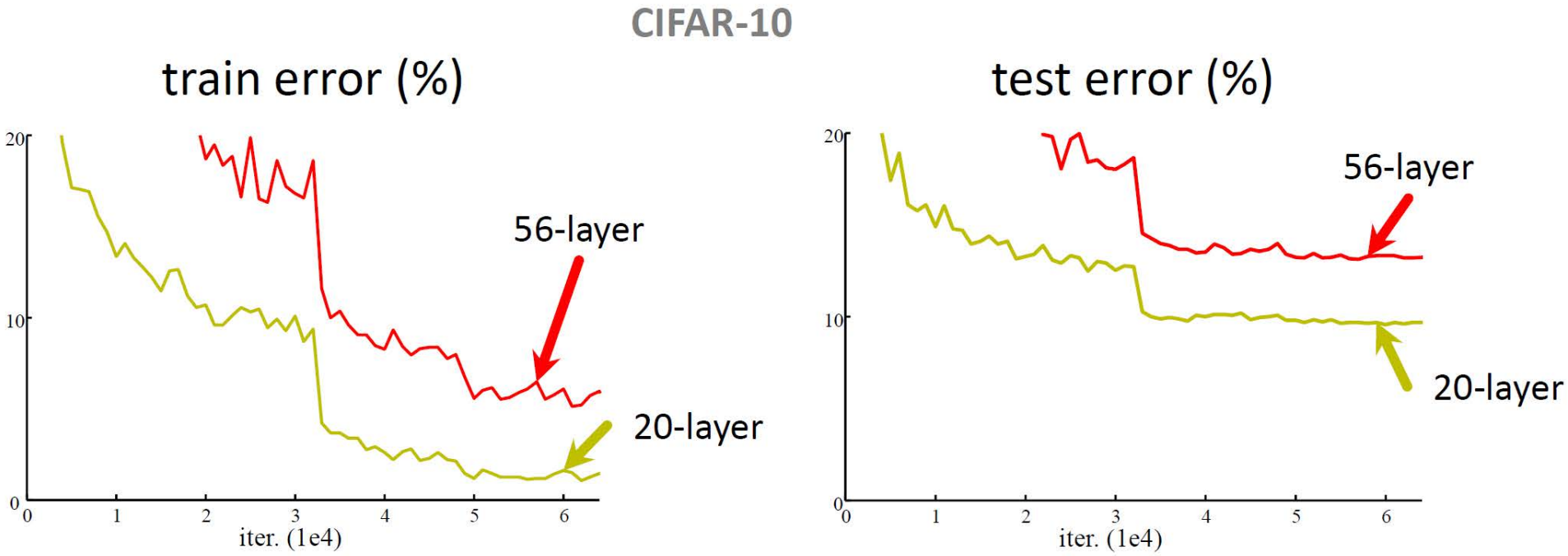


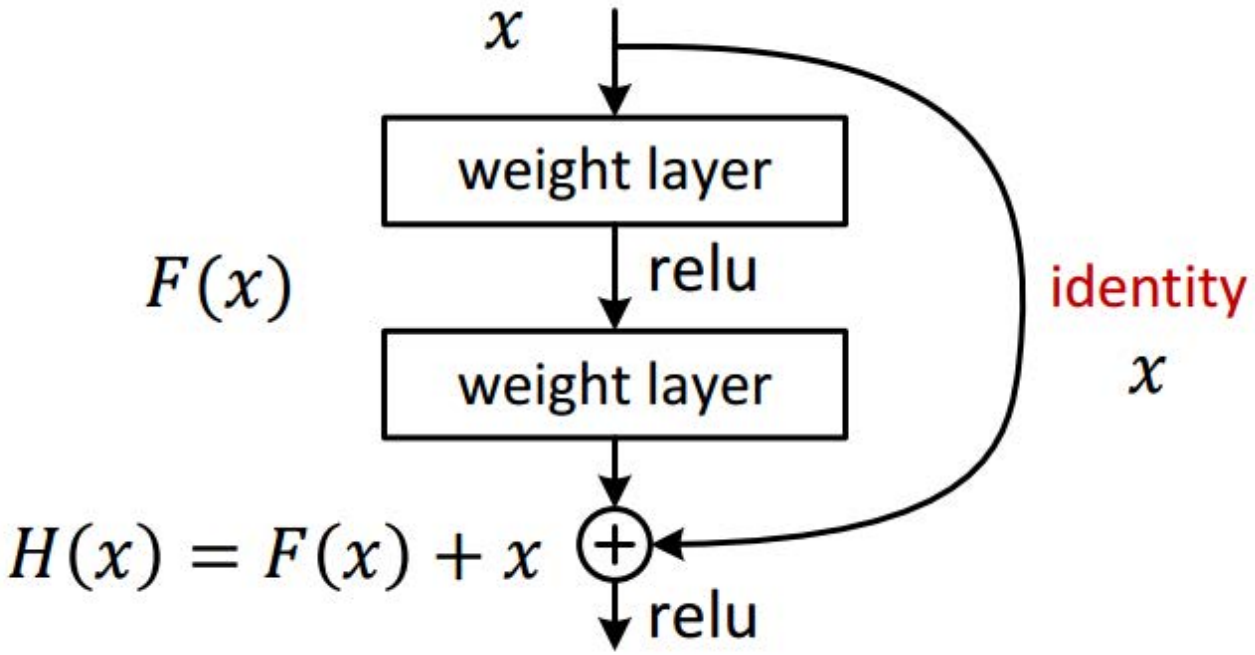

Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

# Going Deeper?




- Simply stacking more layers does not work
  - 56-layer net has higher training error than a 20-layer net

# ResNet



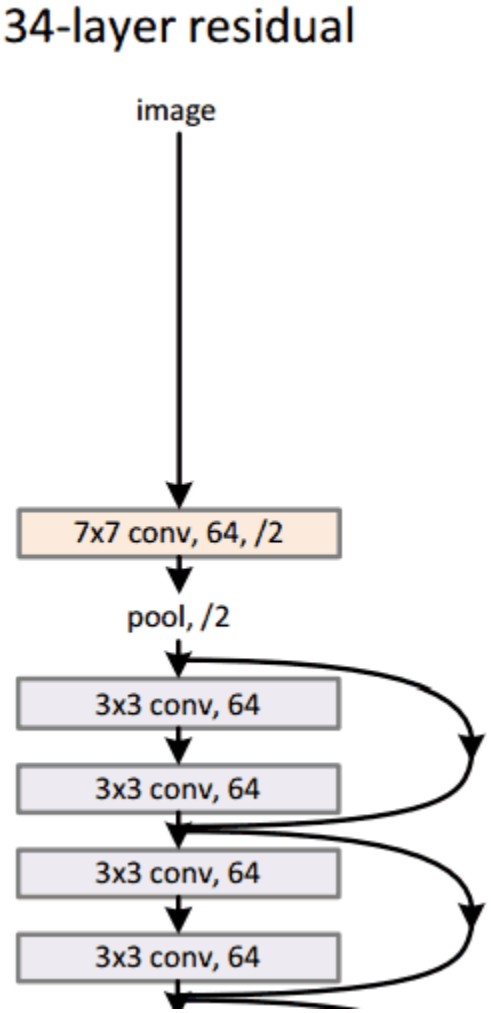
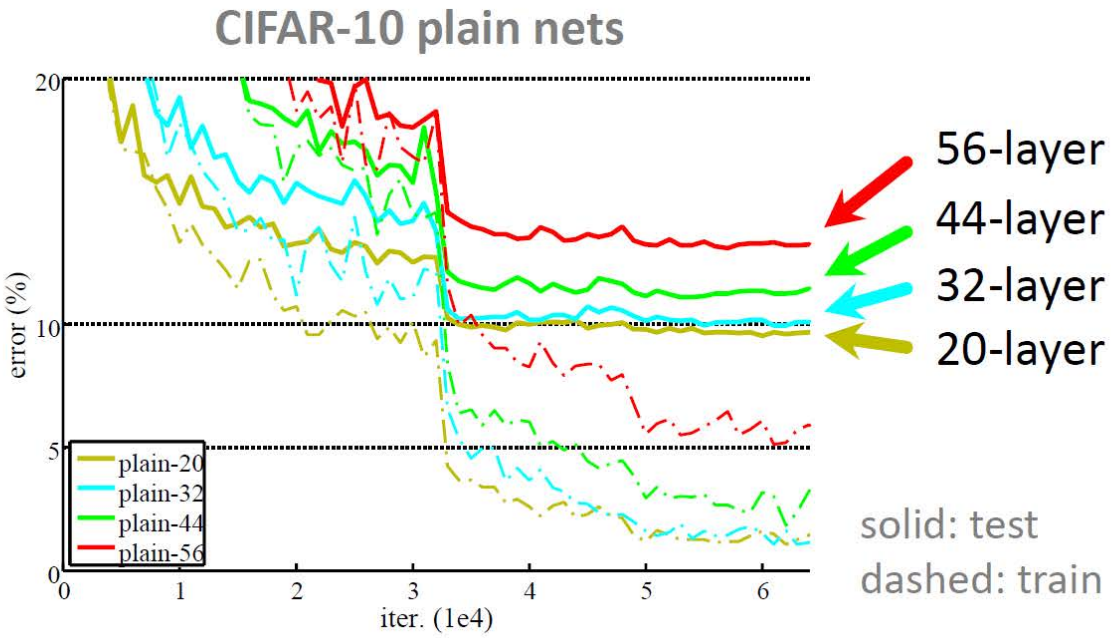
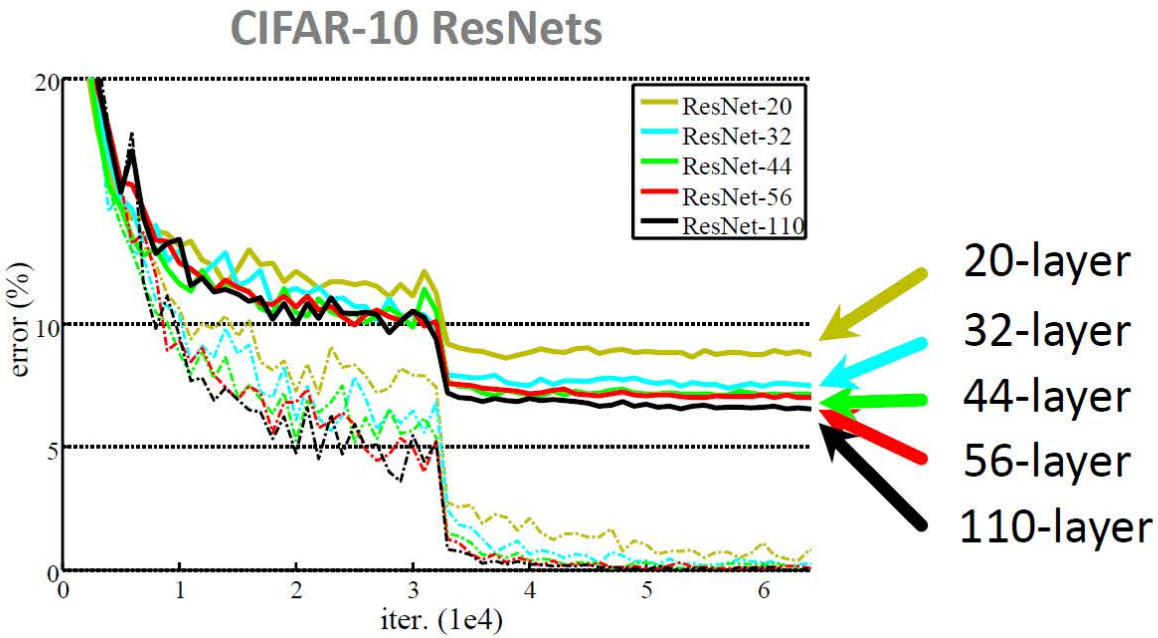




Photo credit: Kaiming He

Deep Residual Learning for Image Recognition, He et al. CVPR 2016

# ResNets on CIFRA-10



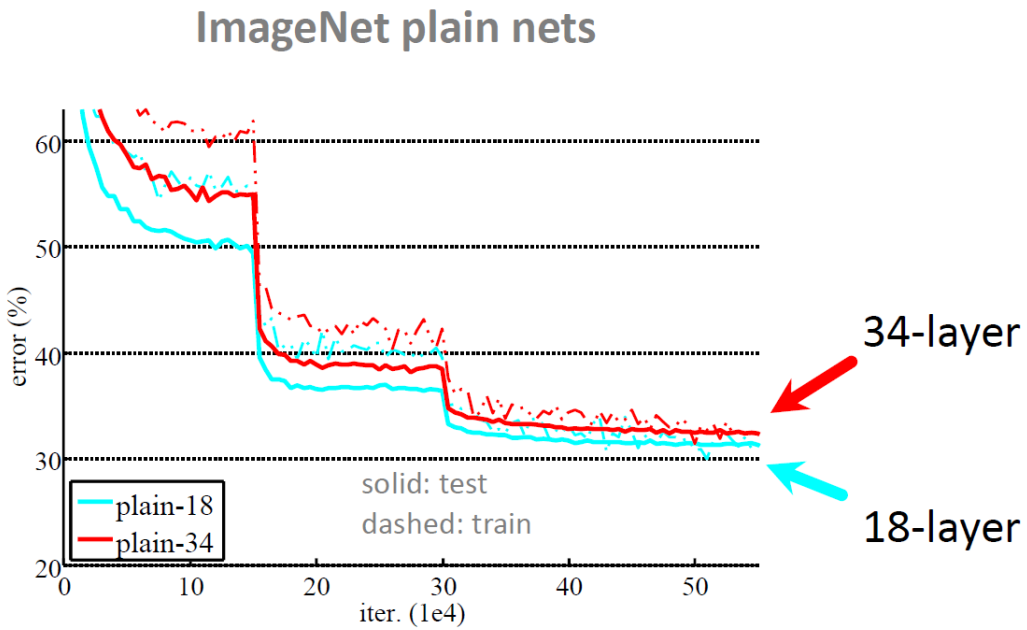



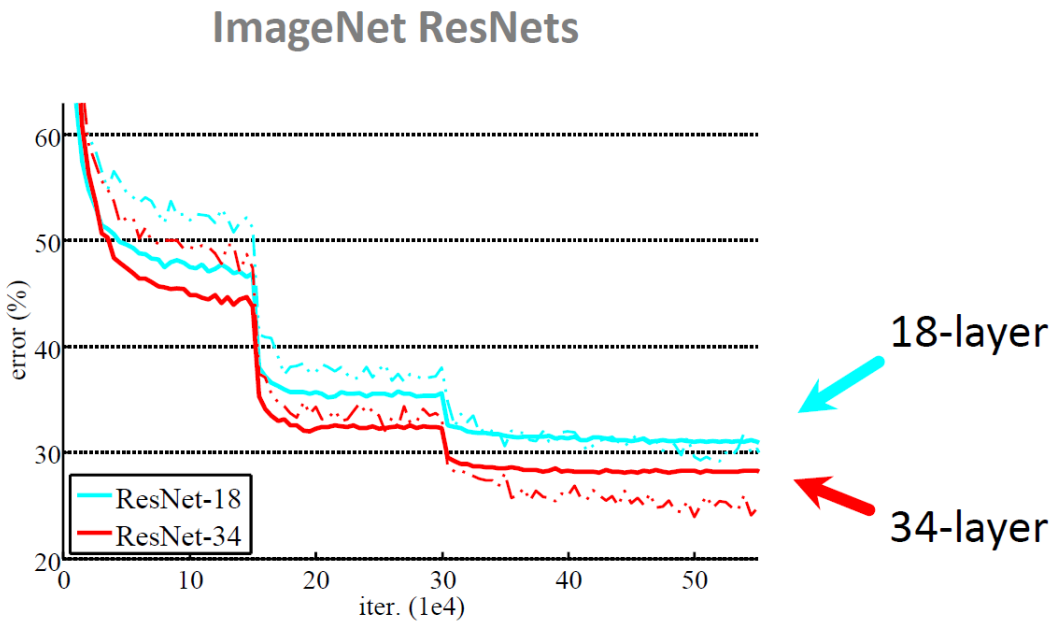

- Deep ResNets can be trained without difficulties
- Deeper ResNets have lower training error, and lower test error

Photo credit: Kaiming He

Deep Residual Learning for Image Recognition, He et al. CVPR 2016

# ResNets on ImageNet



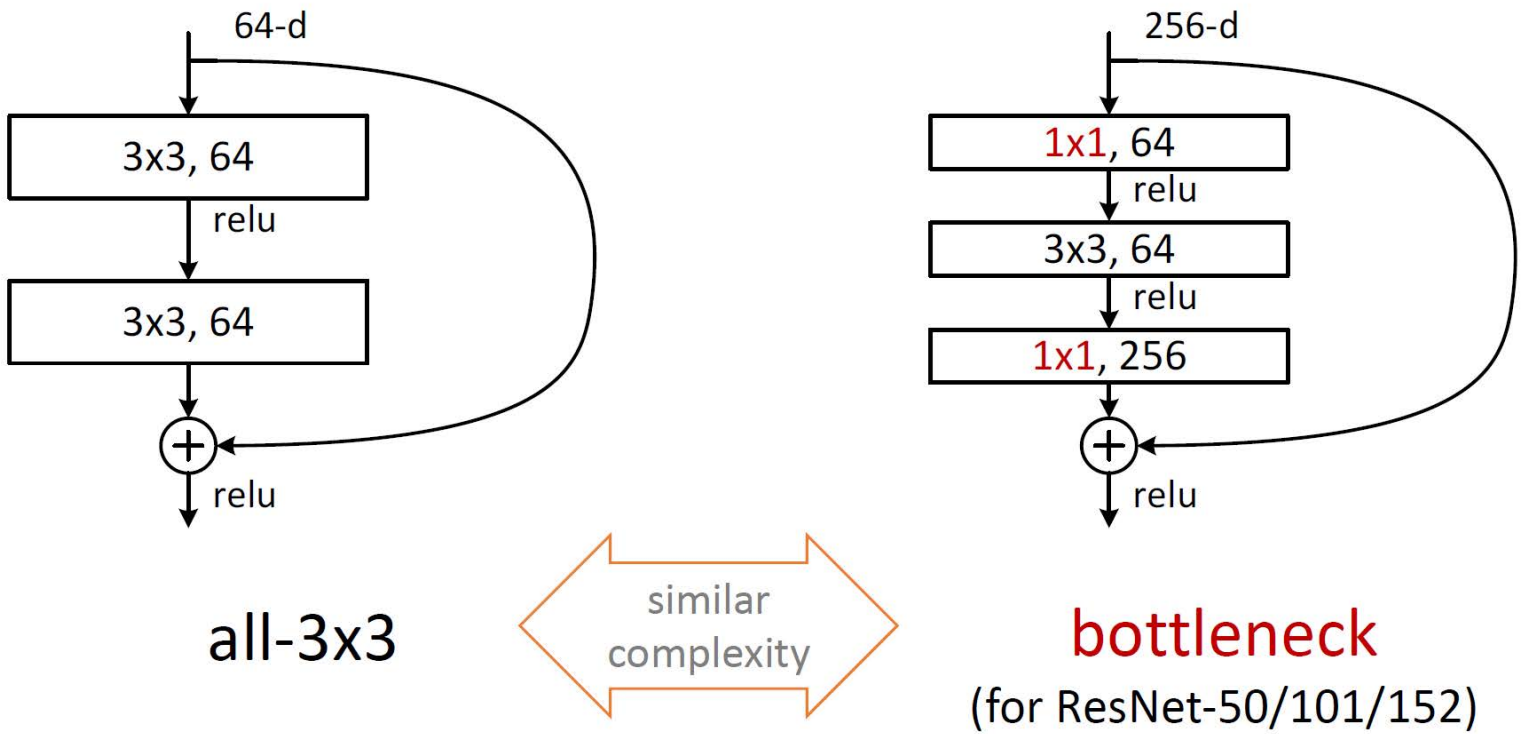

- Deep ResNets can be trained without difficulties
- Deeper ResNets have lower training error, and lower test error

Photo credit: Kaiming He

Deep Residual Learning for Image Recognition, He et al. CVPR 2016

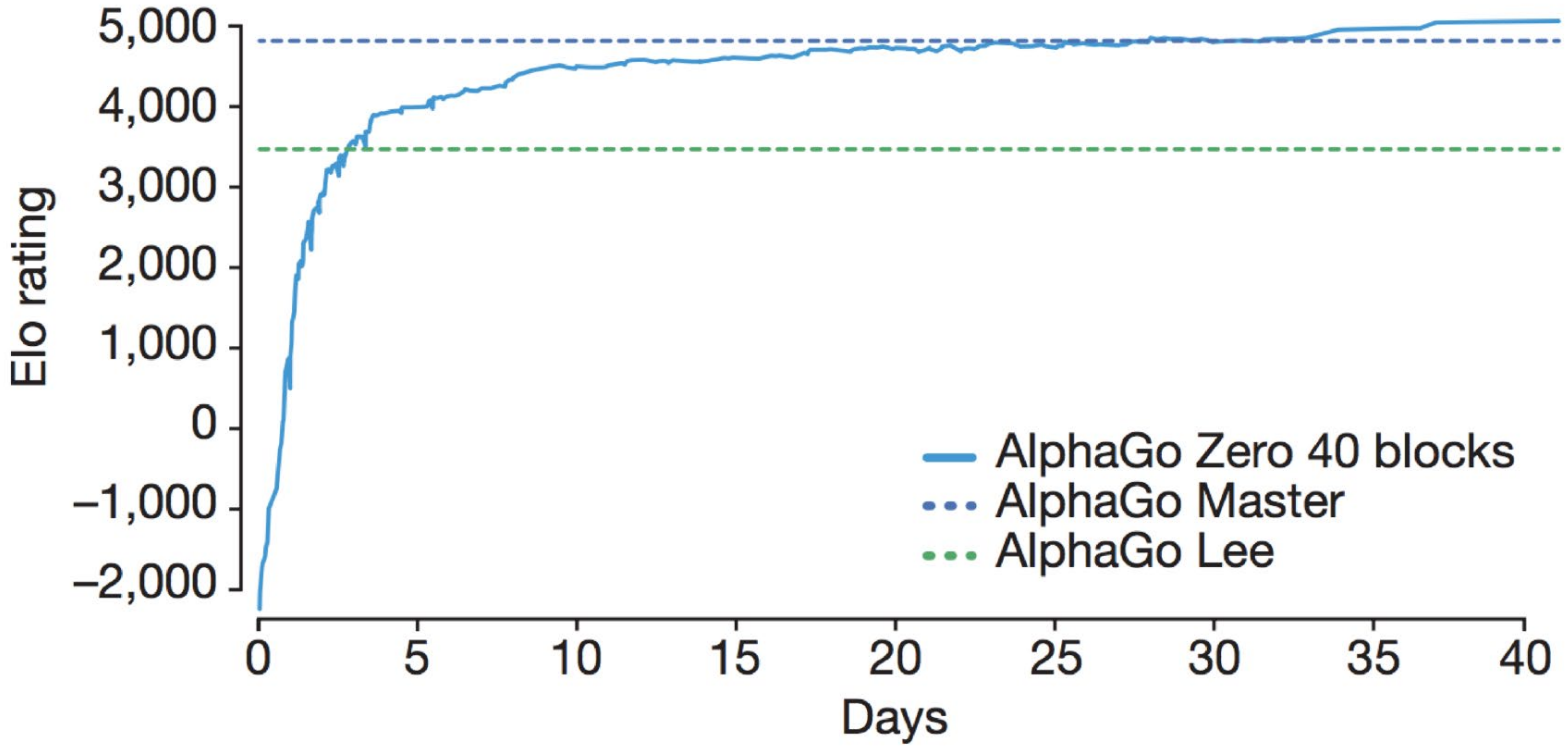
# Going Deeper

- A practical design for going deeper



# ResNet in AlphaGo Zero

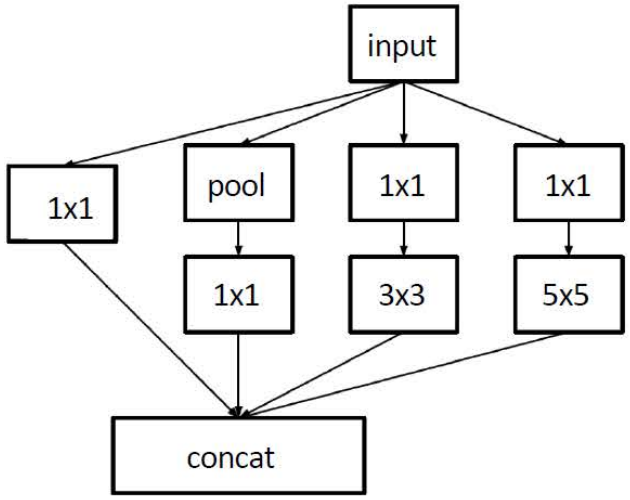
- AlphaGo Zero: 40 Residual Blocks



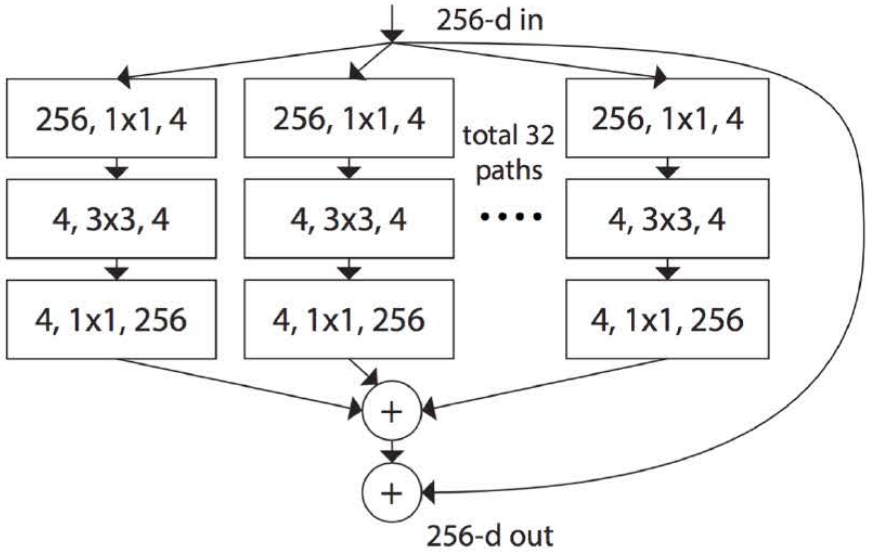

Mastering the game of Go without human knowledge, Silver et al. Nature 2017

# ResNeXt

- Introduce multiple branches into ResNet



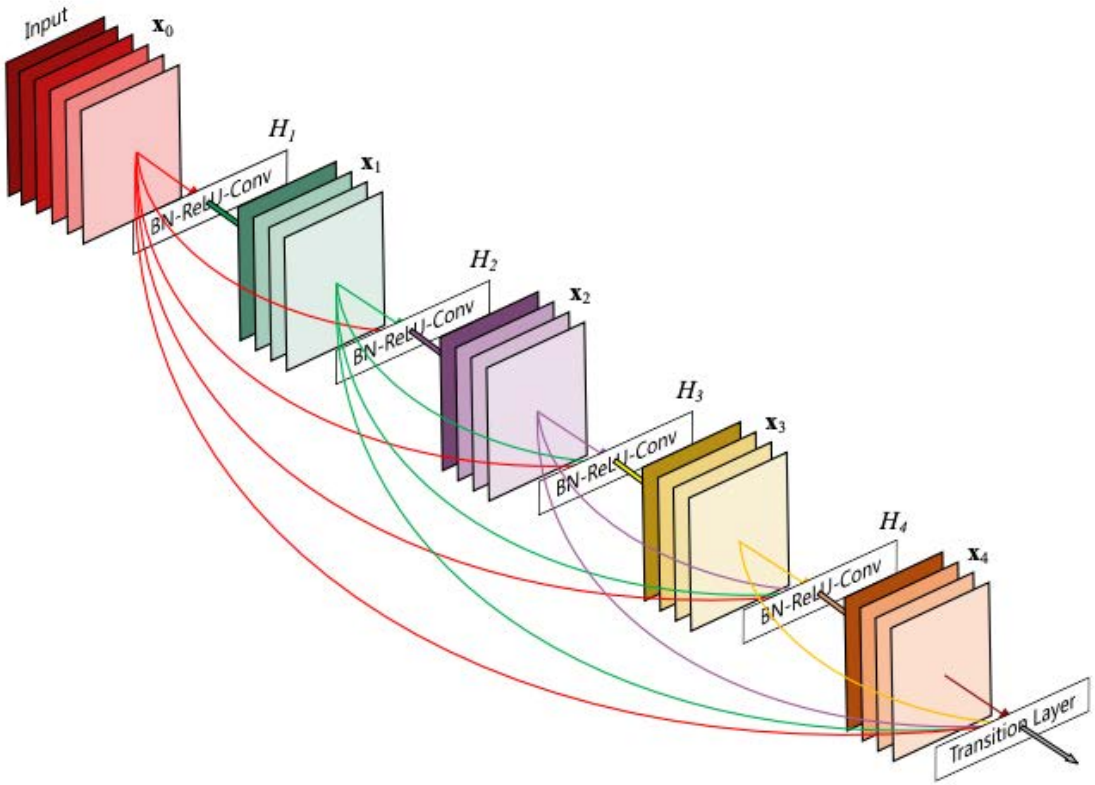

**Inception**:
heterogeneous multi-branch

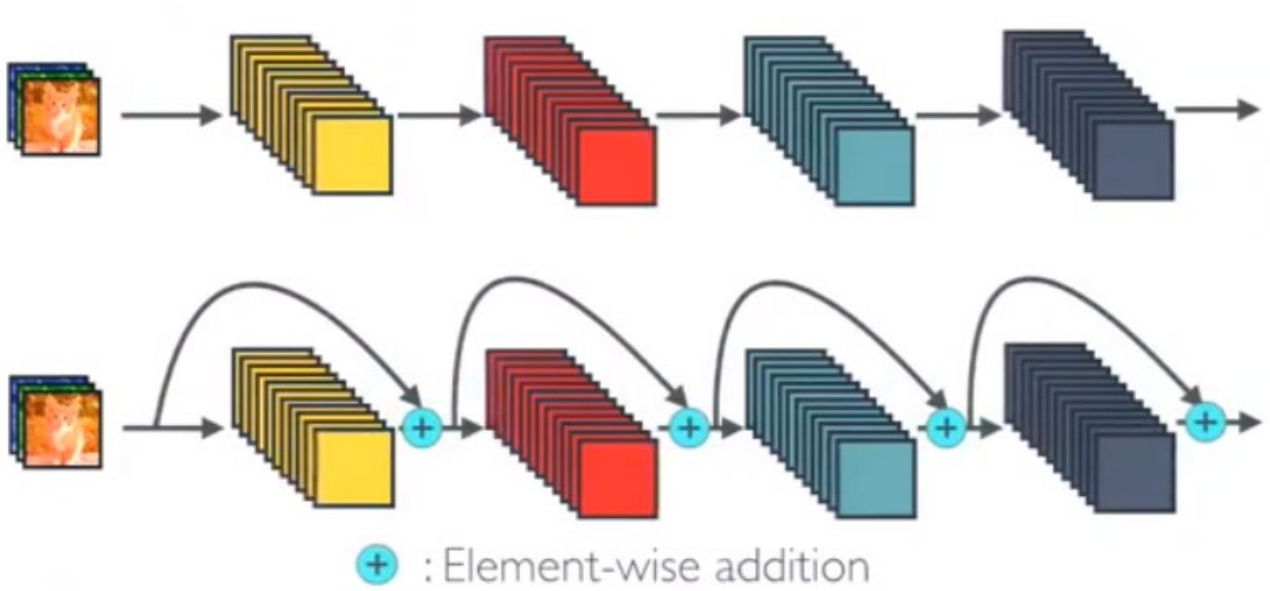


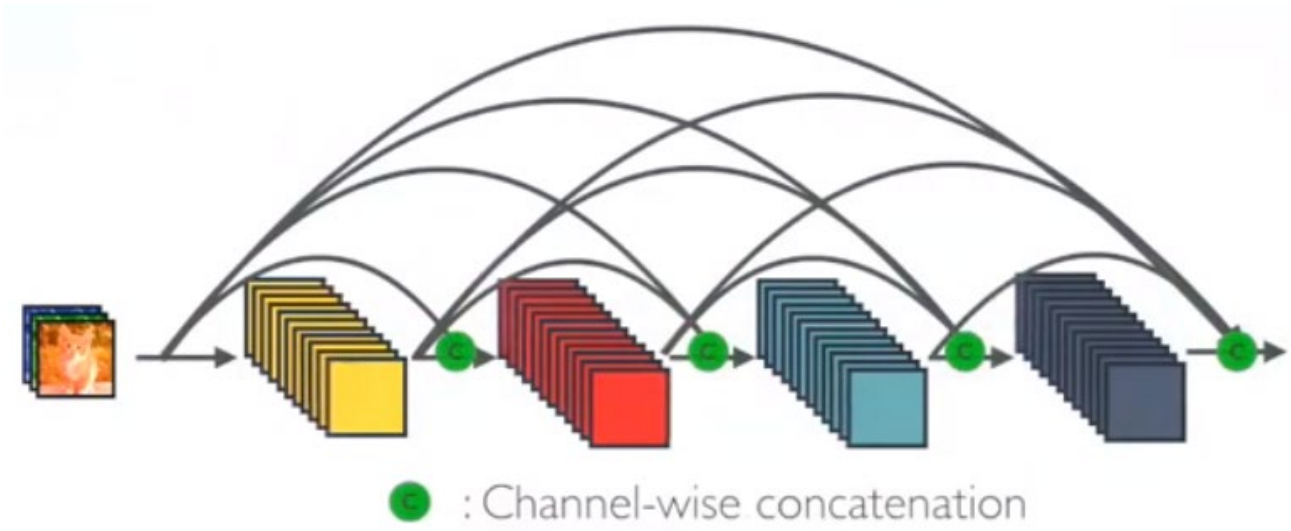

**ResNeXt**:
uniform multi-branch

Aggregated Residual Transformations for Deep Neural Networks, Xie et al. CVPR 2017

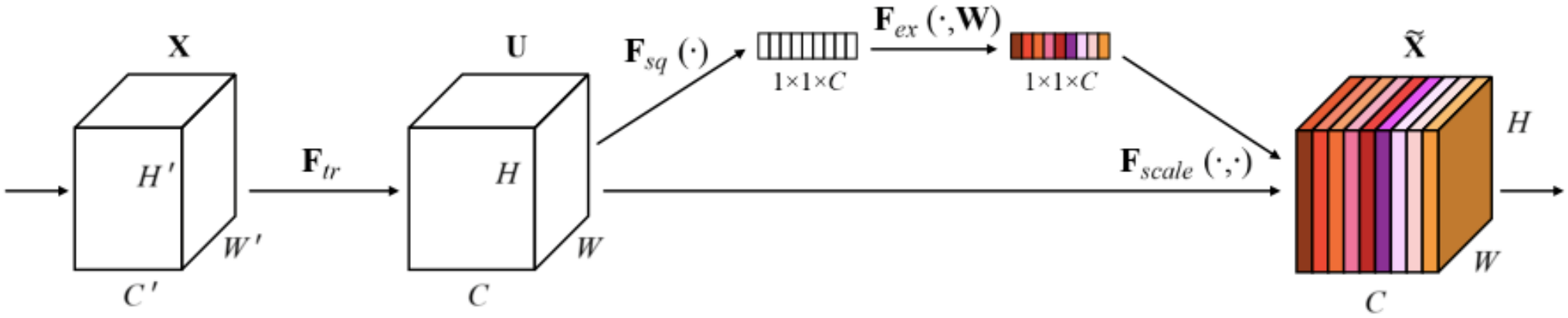
# DenseNet


Input
$\mathbf{x}_0$
$H_1$
BN-ReLU-Conv
$\mathbf{x}_1$
$H_2$
BN-ReLU-Conv
$\mathbf{x}_2$
$H_3$
BN-ReLU-Conv
$\mathbf{x}_3$
$H_4$
BN-ReLU-Conv
$\mathbf{x}_4$
Transition Layer


: Element-wise addition


C
: Channel-wise concatenation

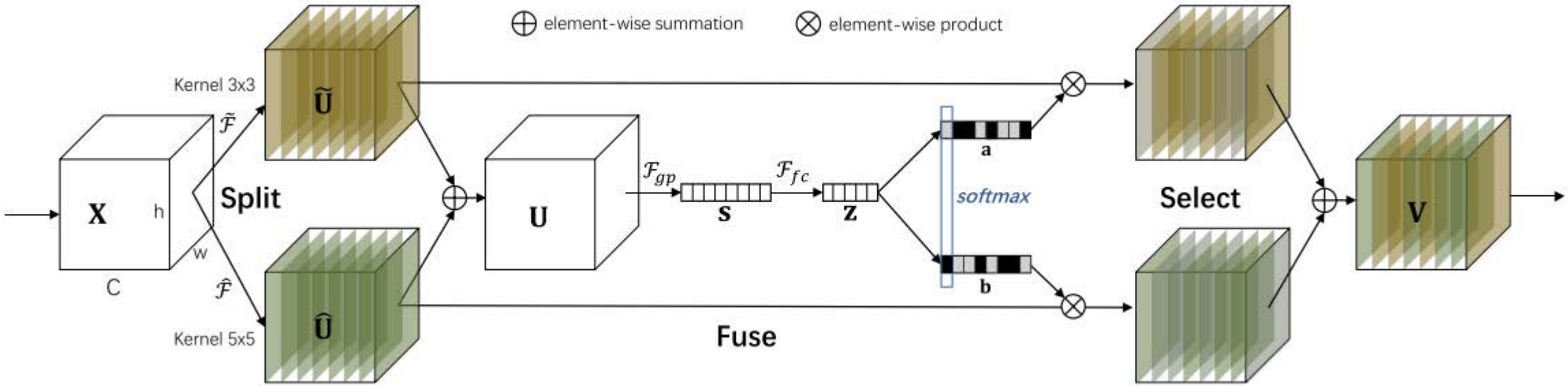
# SENet (attention in CNN)




- Reweight channels according to the input, or Channel Attention
- Capture global context information via GAP

*Hu J, Shen L, Sun G. Squeeze-and-Excitation Networks. CVPR, 2018.*

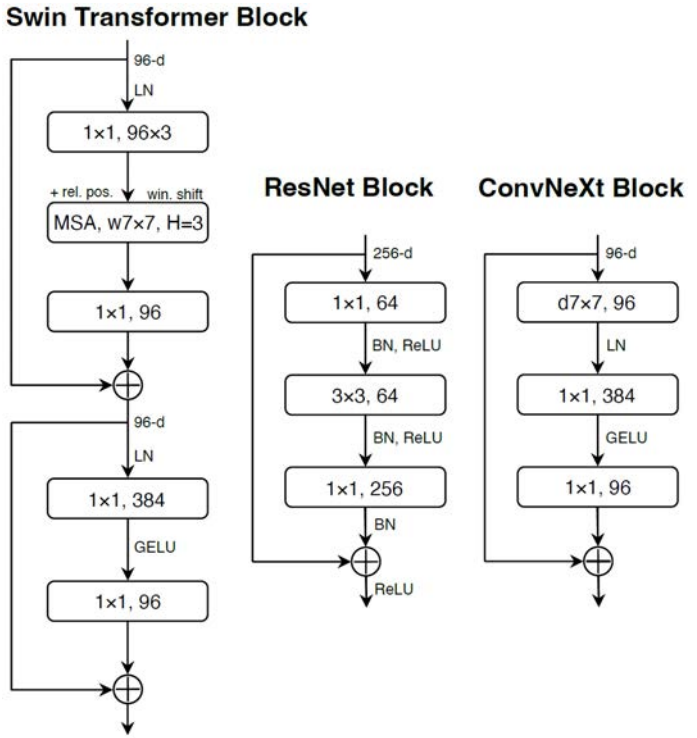
# SKNet (selective kernels)

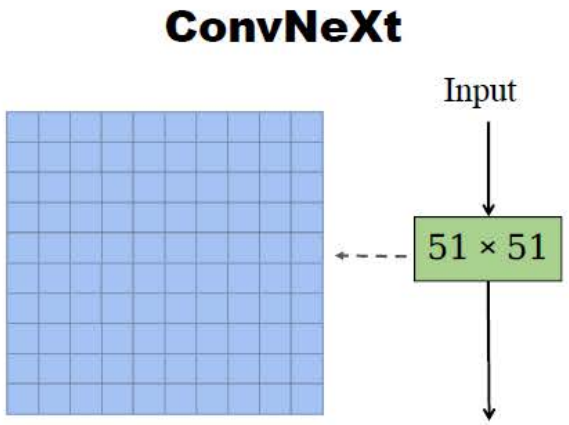


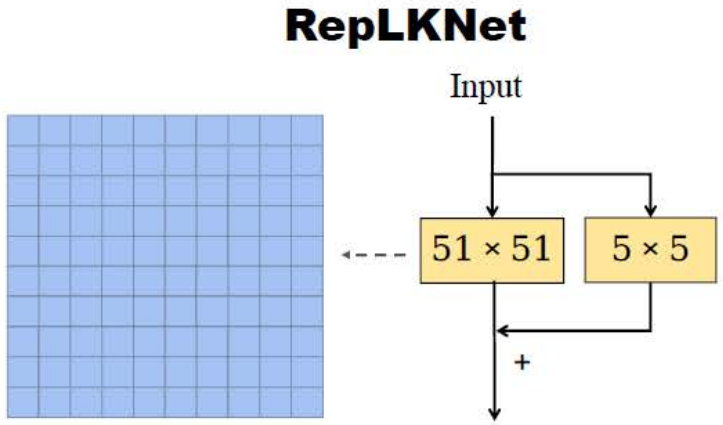

*Li X, Wang W, Hu X, et al. Selective Kernel Networks. CVPR, 2019.*

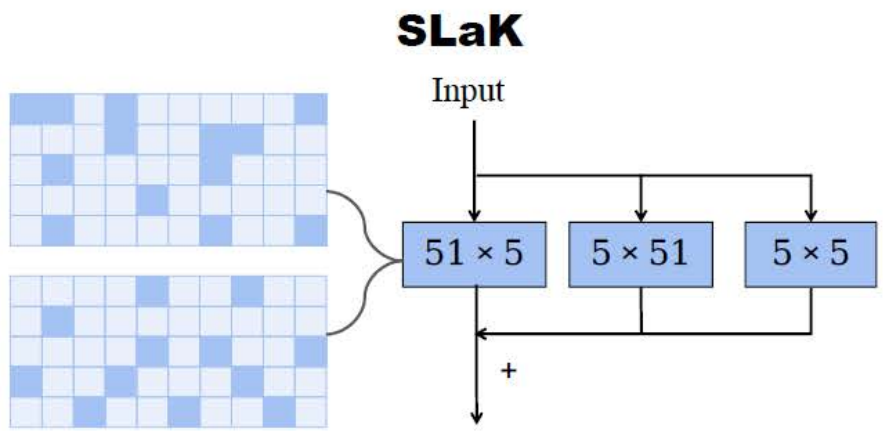
# Large Kernels

- ConvNeXt
- RepLKNet
- SLaK



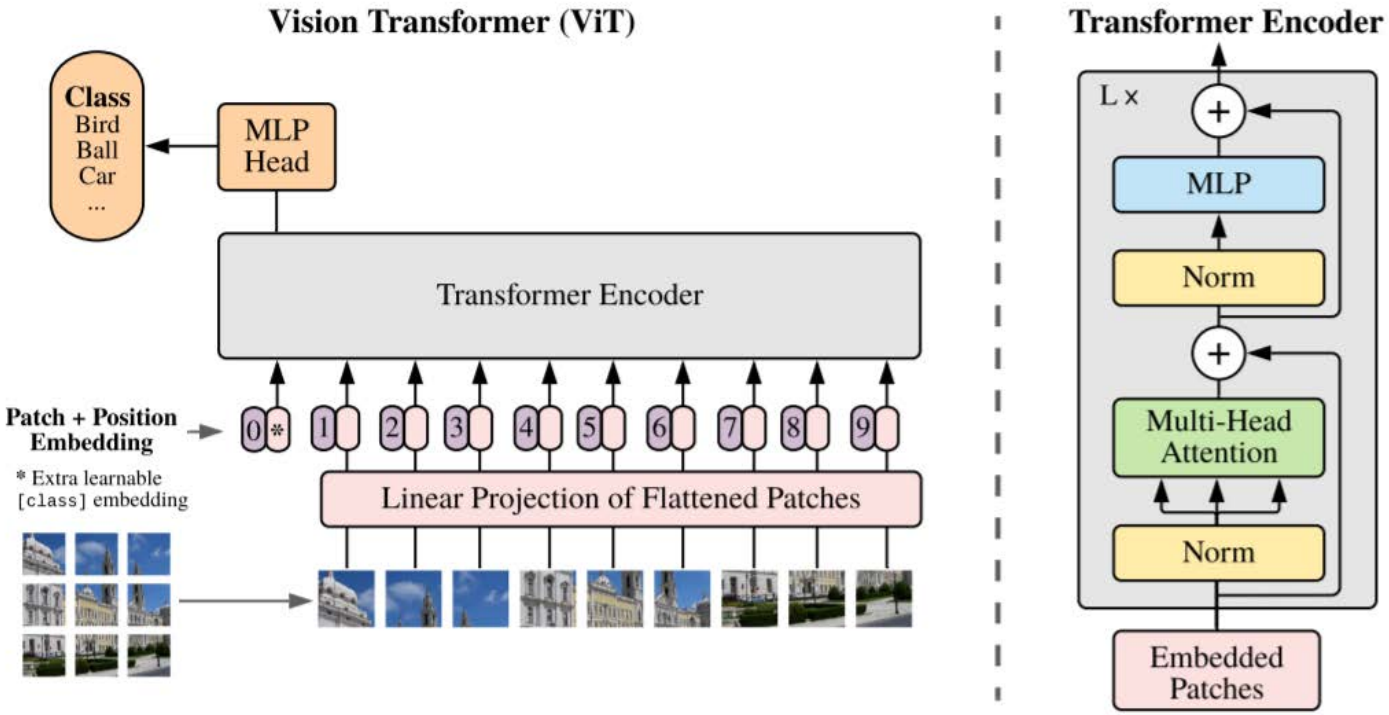




*Liu Z, et al. A ConvNet for the 2020s. CVPR, 2022.*

# Part 2: Vision Transformer

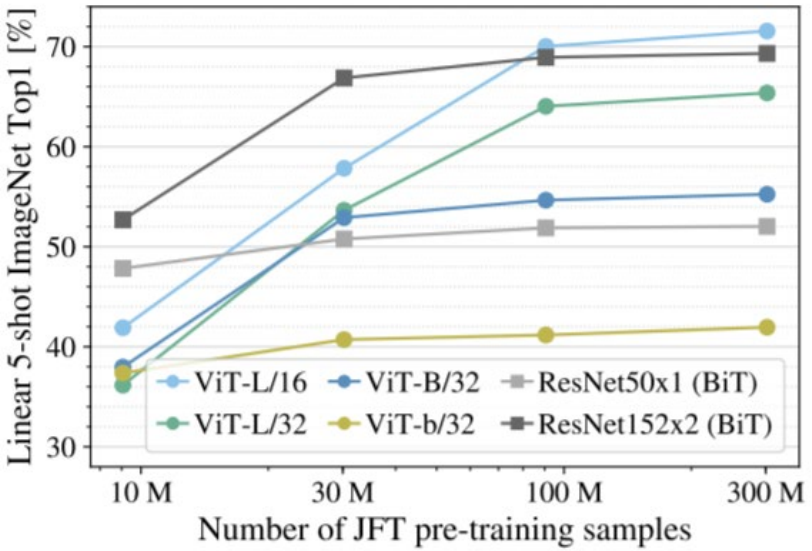
# Vision Transformer

- ViT (An Image is Worth 16x16 Words) (ICLR 2021, Google Brain)



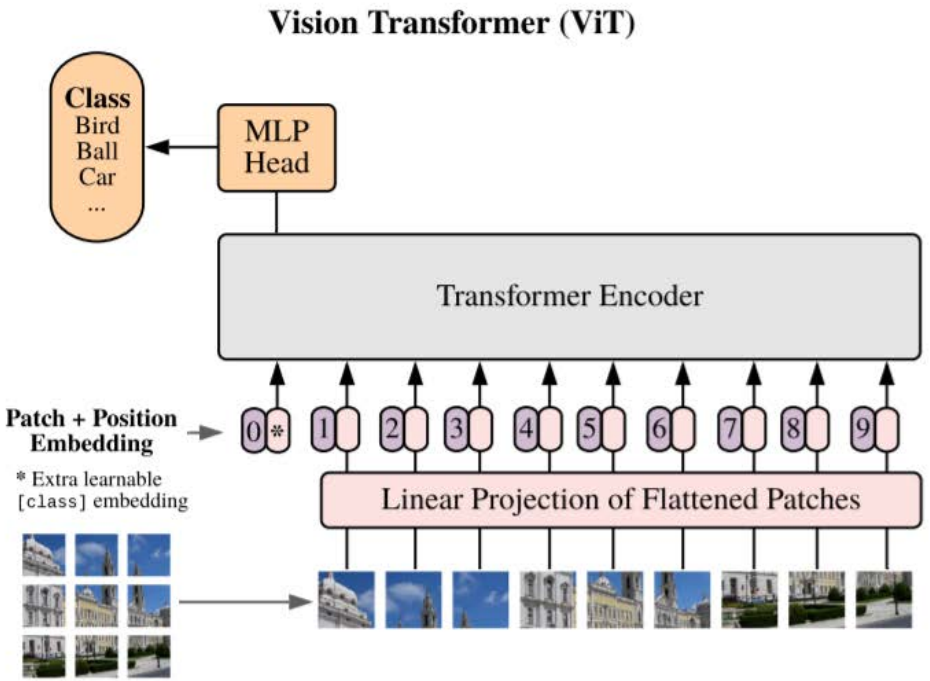


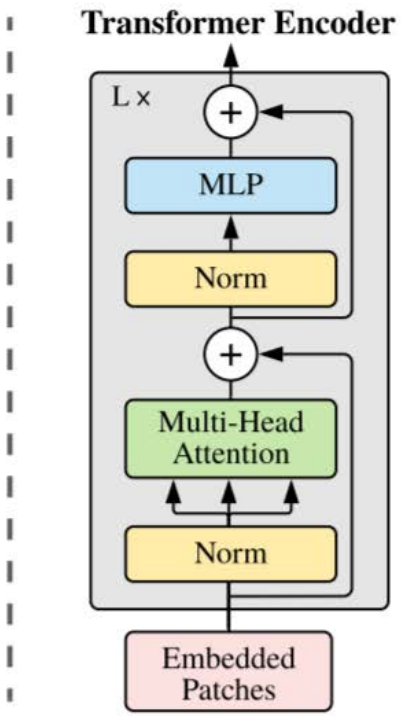




# Vision Transformer

- ViT (An Image is Worth 16x16 Words) (ICLR 2021, Google Brain)




$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\ \mathbf{x}_p^1\mathbf{E};\ \mathbf{x}_p^2\mathbf{E};\ \cdots\ ;\ \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos},$$

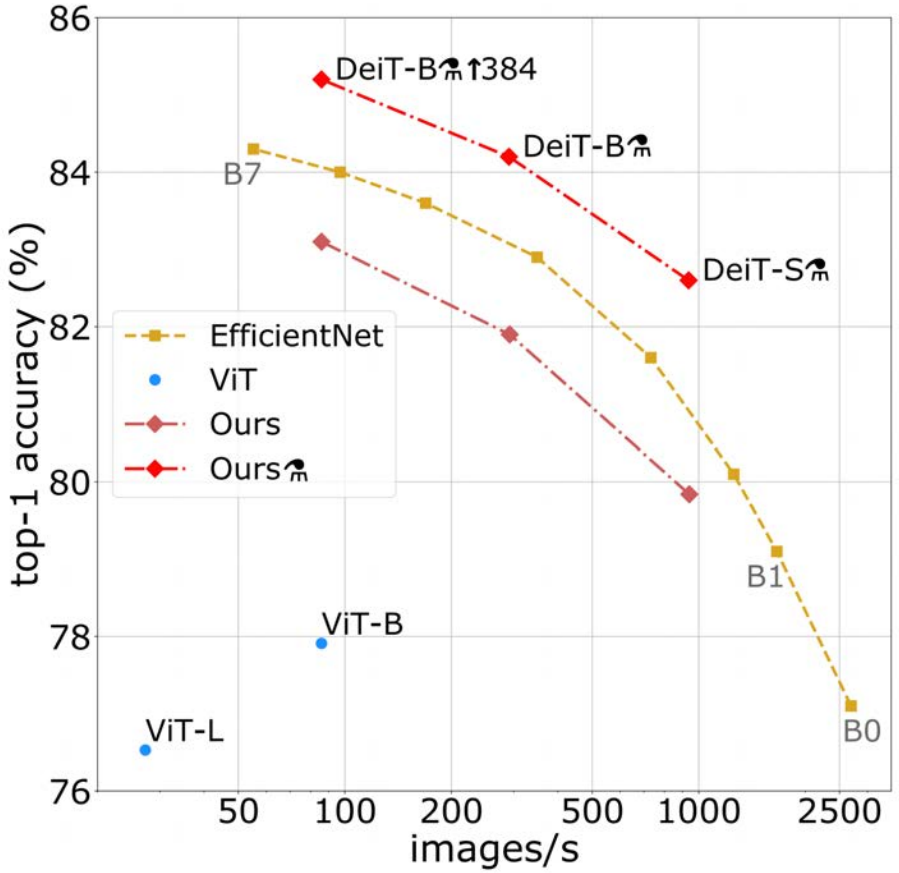
$$\mathbf{z'}_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1},$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z'}_\ell)) + \mathbf{z'}_\ell,$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0)$$

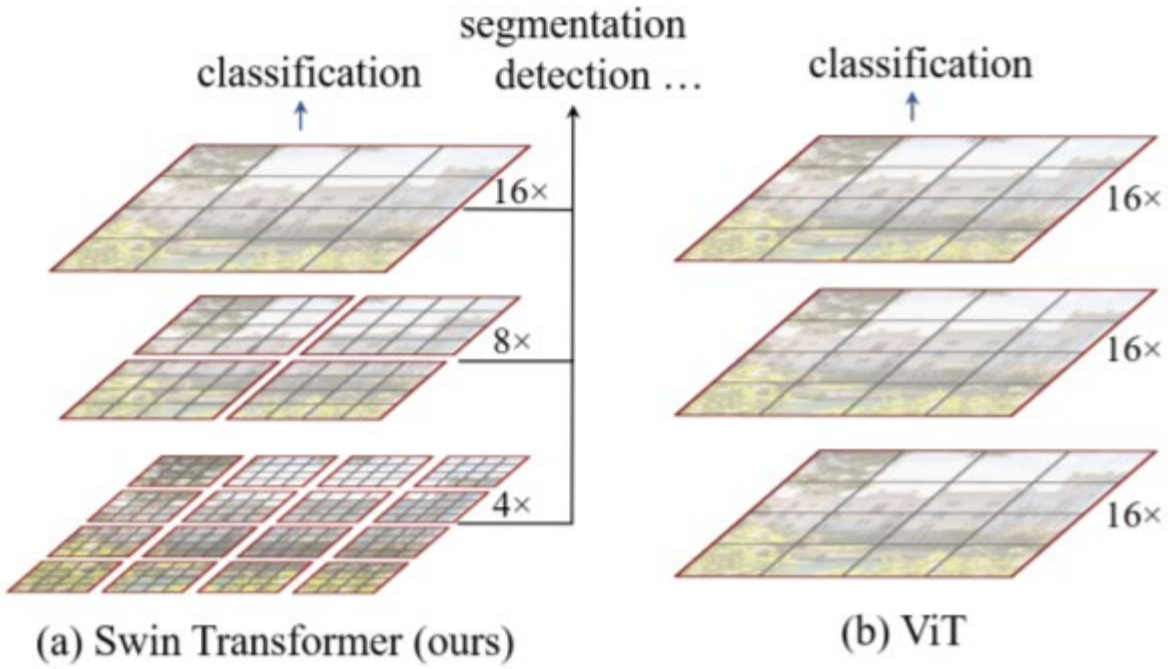
# Vision Transformer

- DEiT ->  engineering improvement of ViT (FaceBook)

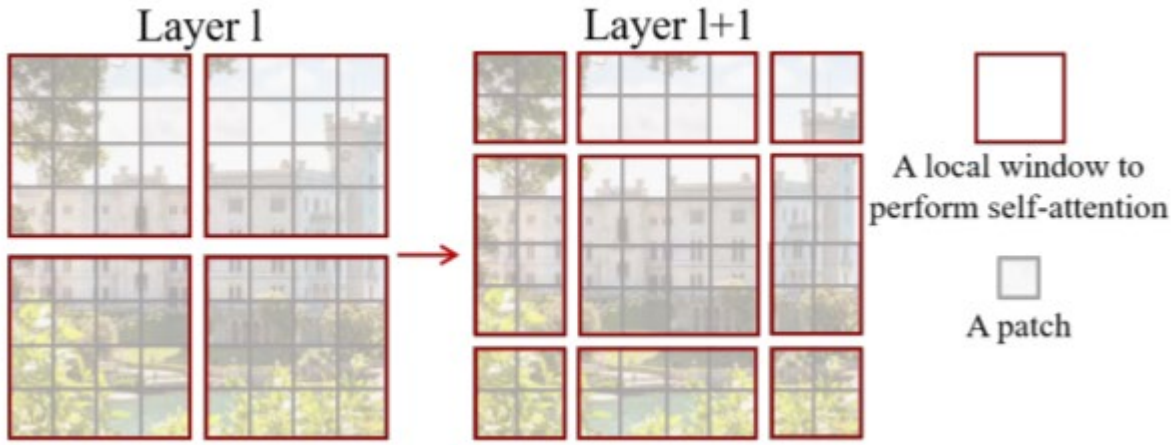


# Vision Transformer

- Swin Transformer (ICCV 2021, , Maar Prize, MSRA)



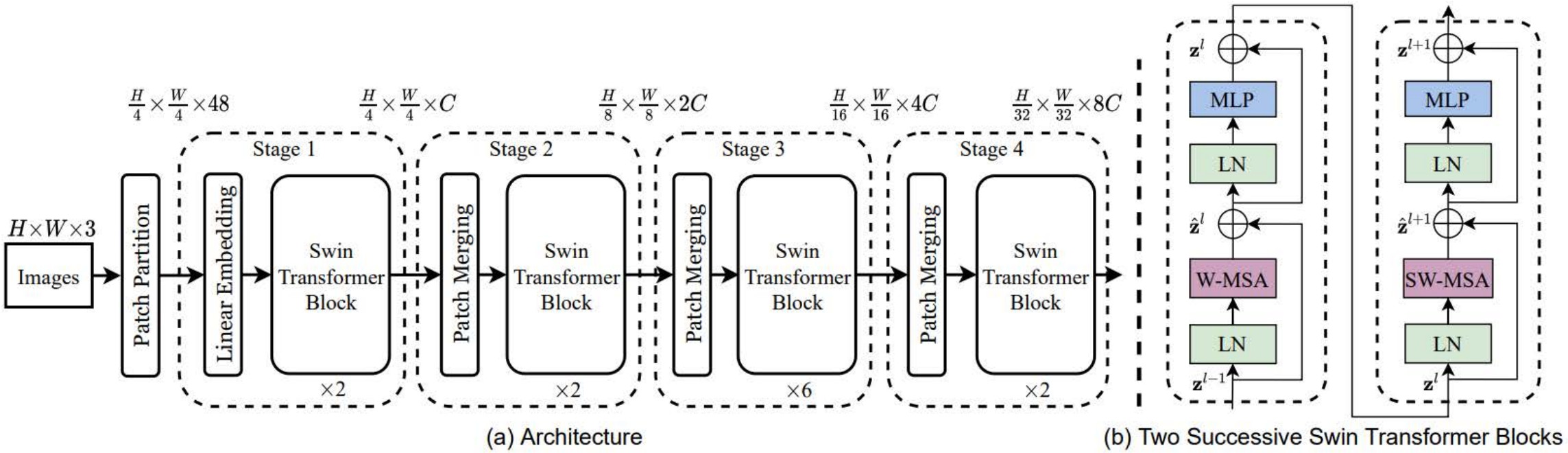

Hierarchical Architecture




Shifted Window

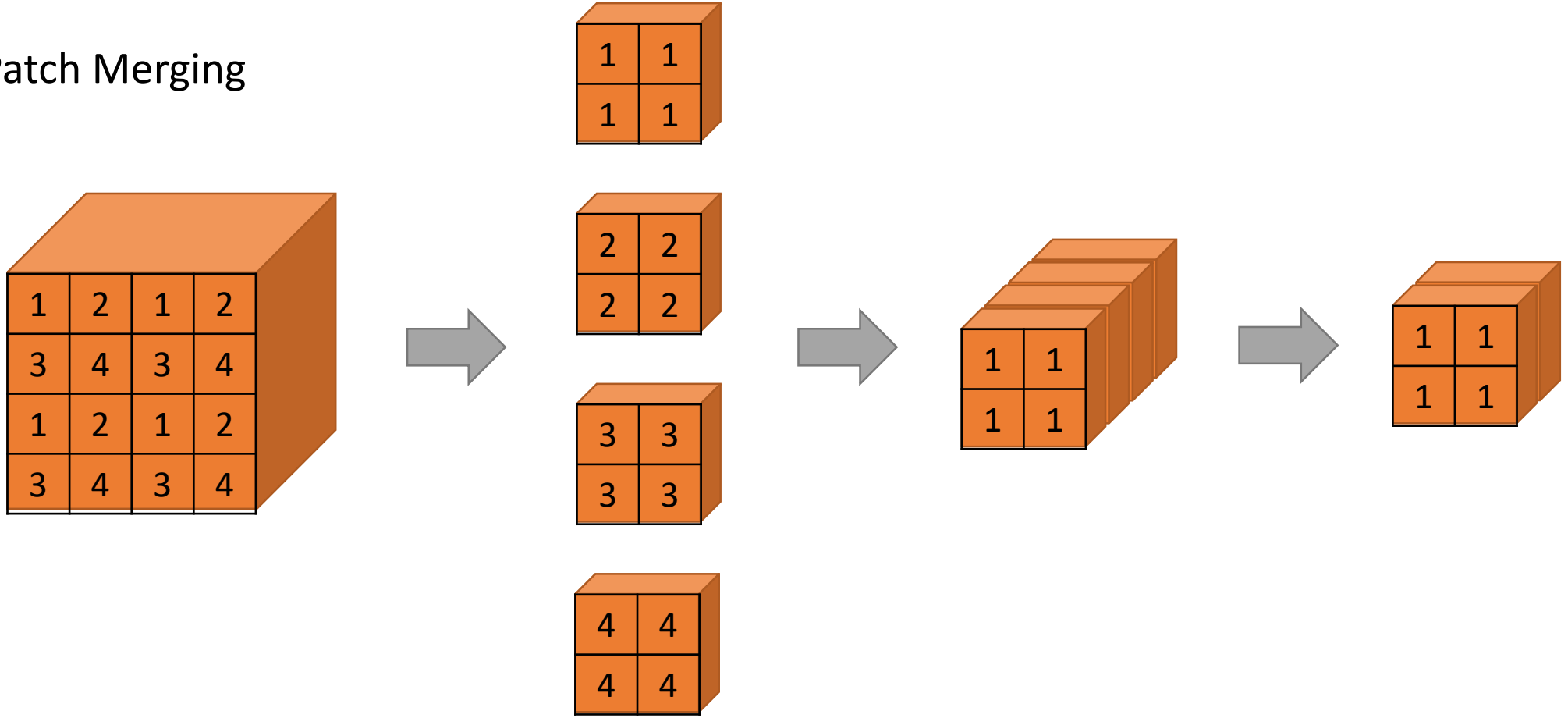
# Vision Transformer

- Swin Transformer (ICCV 2021, , Maar Prize, MSRA)



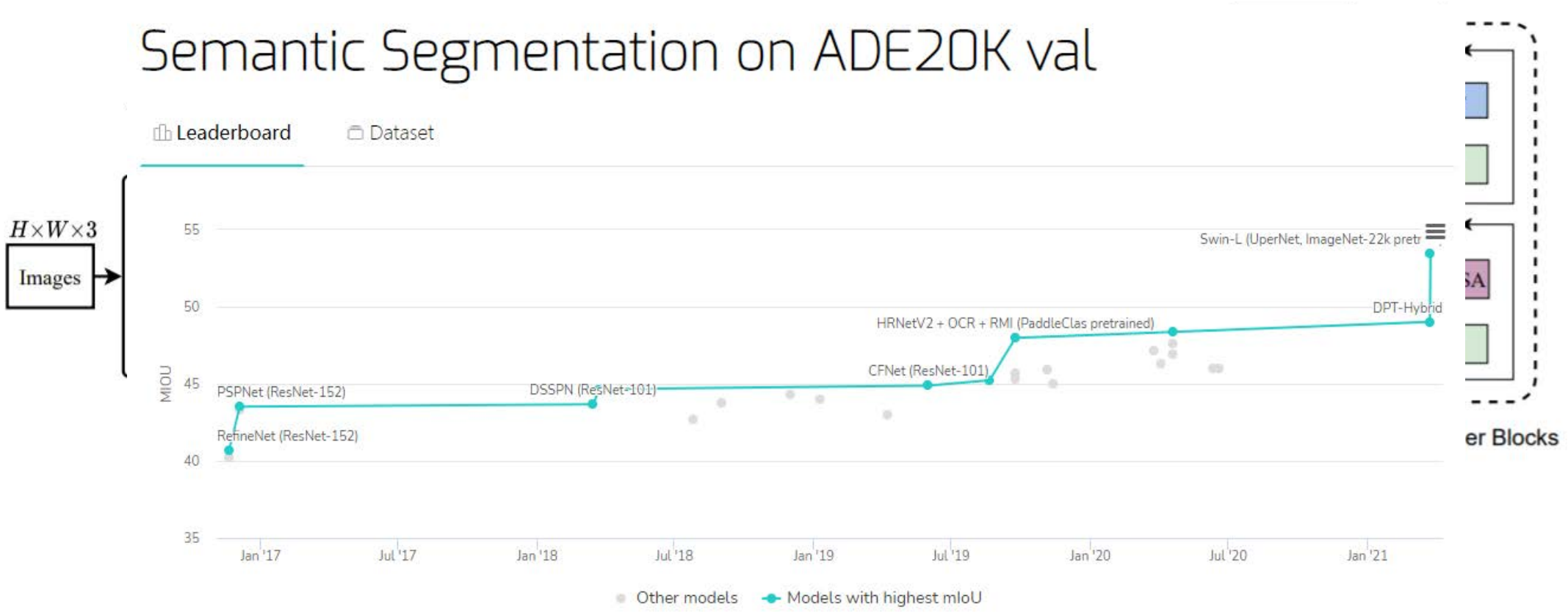
# Vision Transformer

- Swin Transformer (ICCV 2021, , Maar Prize, MSRA)
  - Patch Merging



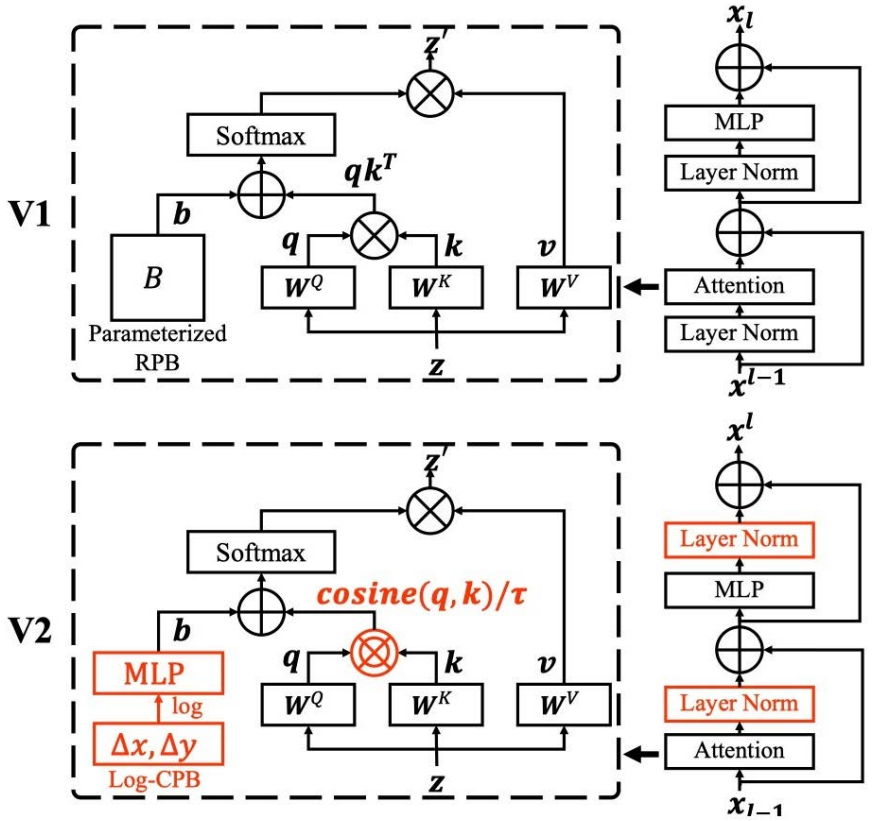
# Vision Transformer

- Swin Transformer (ICCV 2021, Maar Prize, MSRA)



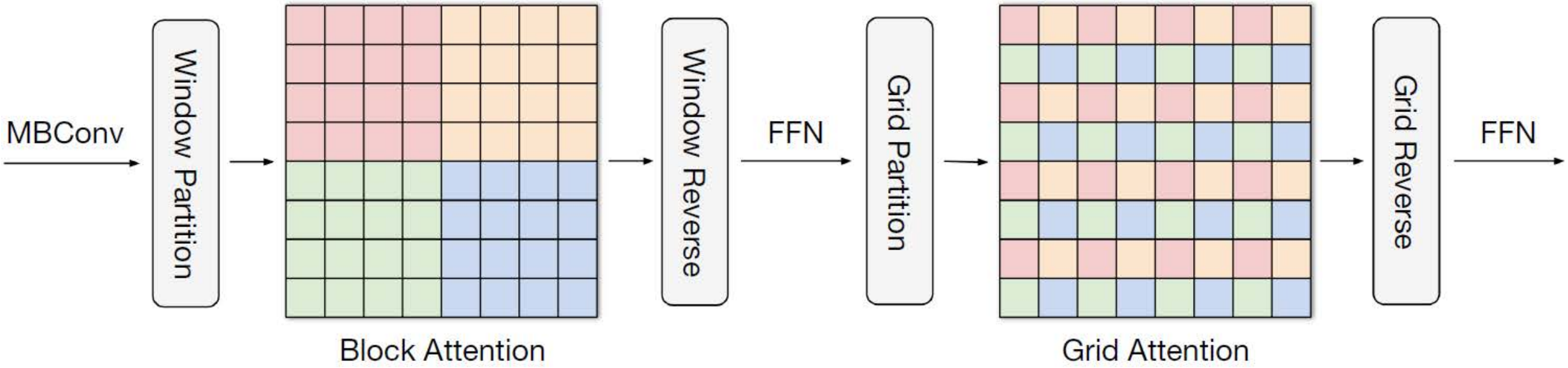
# Vision Transformer

- Swin Transformer v2 (Nov 2021, MSRA)
  - scaling Swin Transformer up to 3 billion parameters
  - making it capable of training with images of up to 1,536×1,536 resolution
  - 84.0% top-1 accuracy on ImageNet-V2 image classification,
  - 63.1 / 54.4 box / mask mAP on COCO object detection,
  - 59.9 mIoU on ADE20K semantic segmentation,
  - 86.8% top-1 accuracy on Kinetics-400 video action classification



# Vision Transformer

- MaxViT (2022)




**Block attention**

窗口内自注意机制

$$(H, W, C) \rightarrow \left(\frac{H}{P} \times P, \frac{W}{P} \times P, C\right) \rightarrow \left(\frac{HW}{P^2}, P^2, C\right)$$

**Grid attention**

$$(H, W, C) \rightarrow \left(G \times \frac{H}{G}, G \times \frac{W}{G}, C\right) \rightarrow \left(G^2, \frac{HW}{G^2}, C\right)$$

# Vision Transformer

- MaxViT (2022)

$$\mathbf{x} \leftarrow \mathbf{x} + \text{Proj}(\text{SE}(\text{DWConv}(\text{Conv}(\text{Norm}(\mathbf{x})))))$$

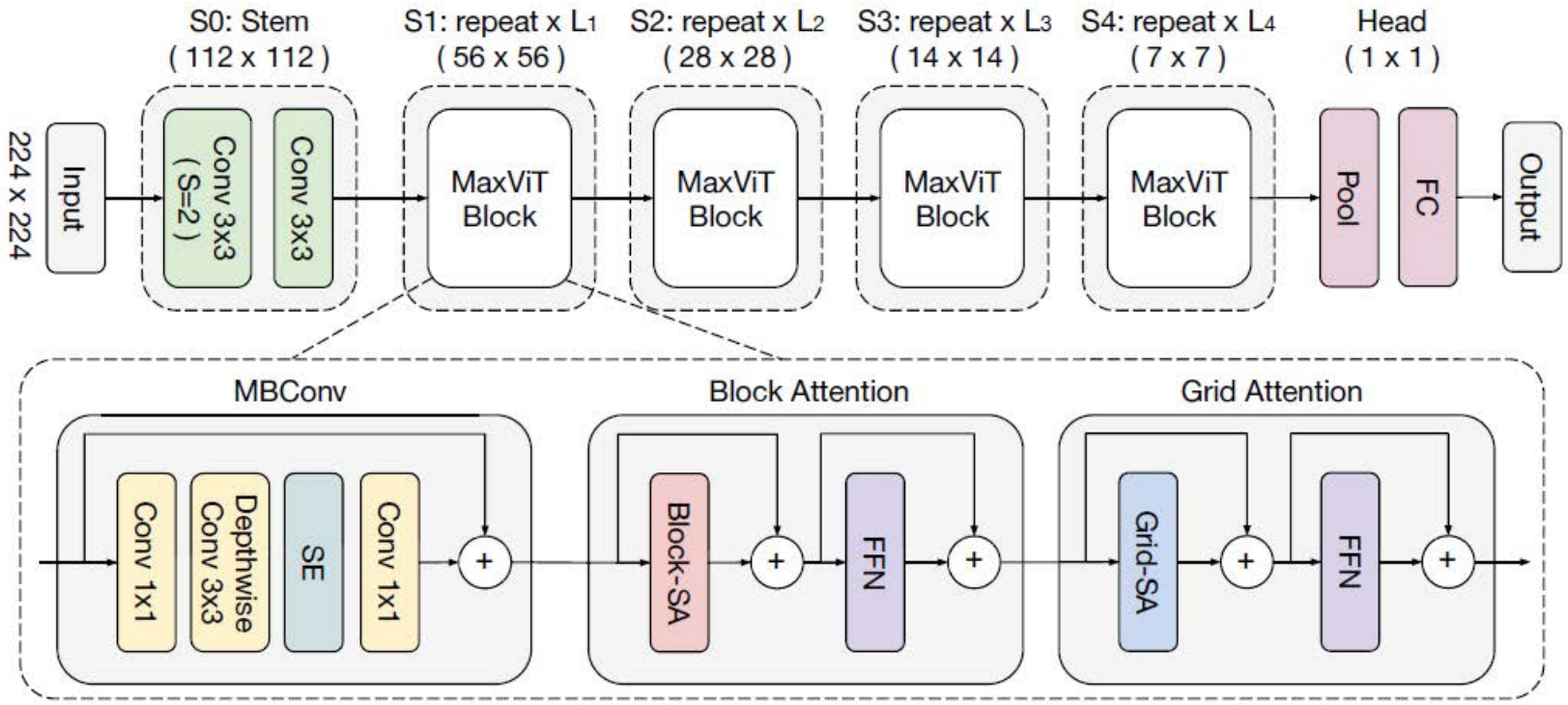
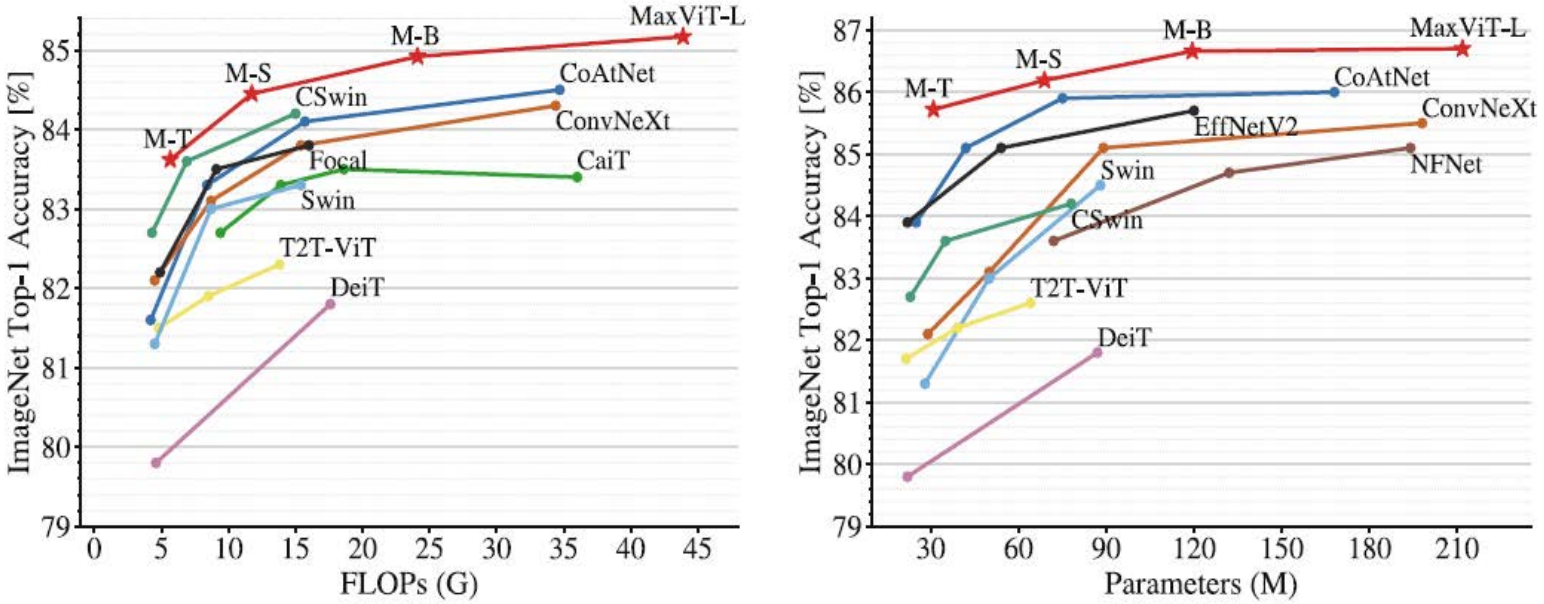



**Fig. 2. MaxViT architecture.** We follow a typical hierarchical design of ConvNet practices (e.g., ResNet) but instead build a new type of basic building block that unifies MBConv, block, and grid attention layers. Normalization and activation layers are omitted for simplicity.

*Zhengzhong Tu, et al. MaxViT: Multi-axis Vision Transformer.*
*European Conference on Computer Vision, ECCV 2022*

# Vision Transformer

- MaxViT (2022)




(a) Accuracy vs. FLOPs performance scaling curve under ImageNet-1K training setting at input resolution 224×224.

(b) Accuracy vs. Parameters scaling curve under ImageNet-1K fine-tuning setting allowing for higher sizes (384/512).

**Fig. 1. Performance comparison of MaxViT with state-of-the-art vision Transformers on ImageNet-1K.** Our model shows superior performance in terms of both accuracy *vs.* computation and accuracy *vs.* parameters tradeoff.

*Zhengzhong Tu, et al. MaxViT: Multi-axis Vision Transformer.*
*European Conference on Computer Vision, ECCV 2022*

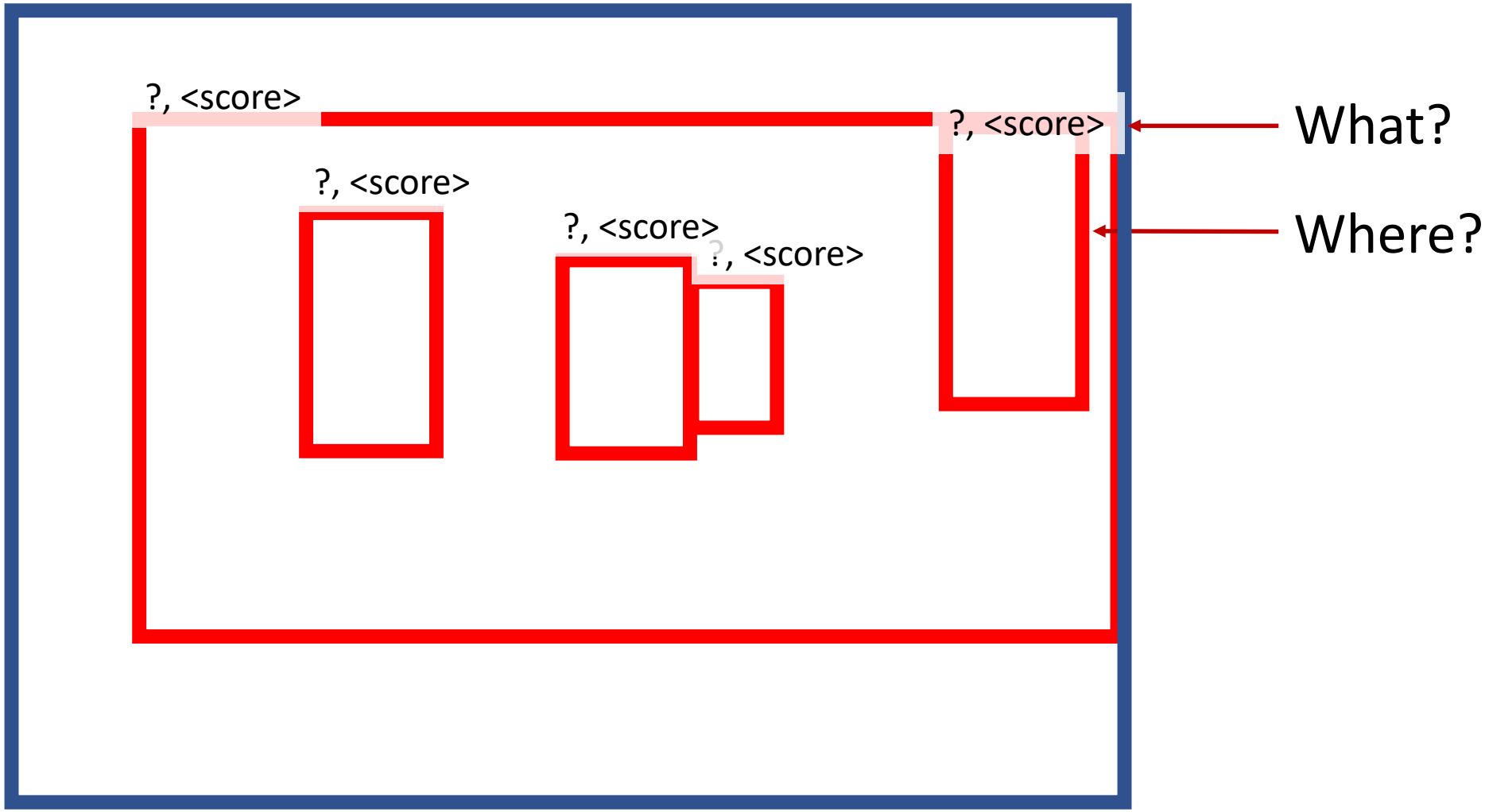
# Part 3: Selected Tasks

Object detection

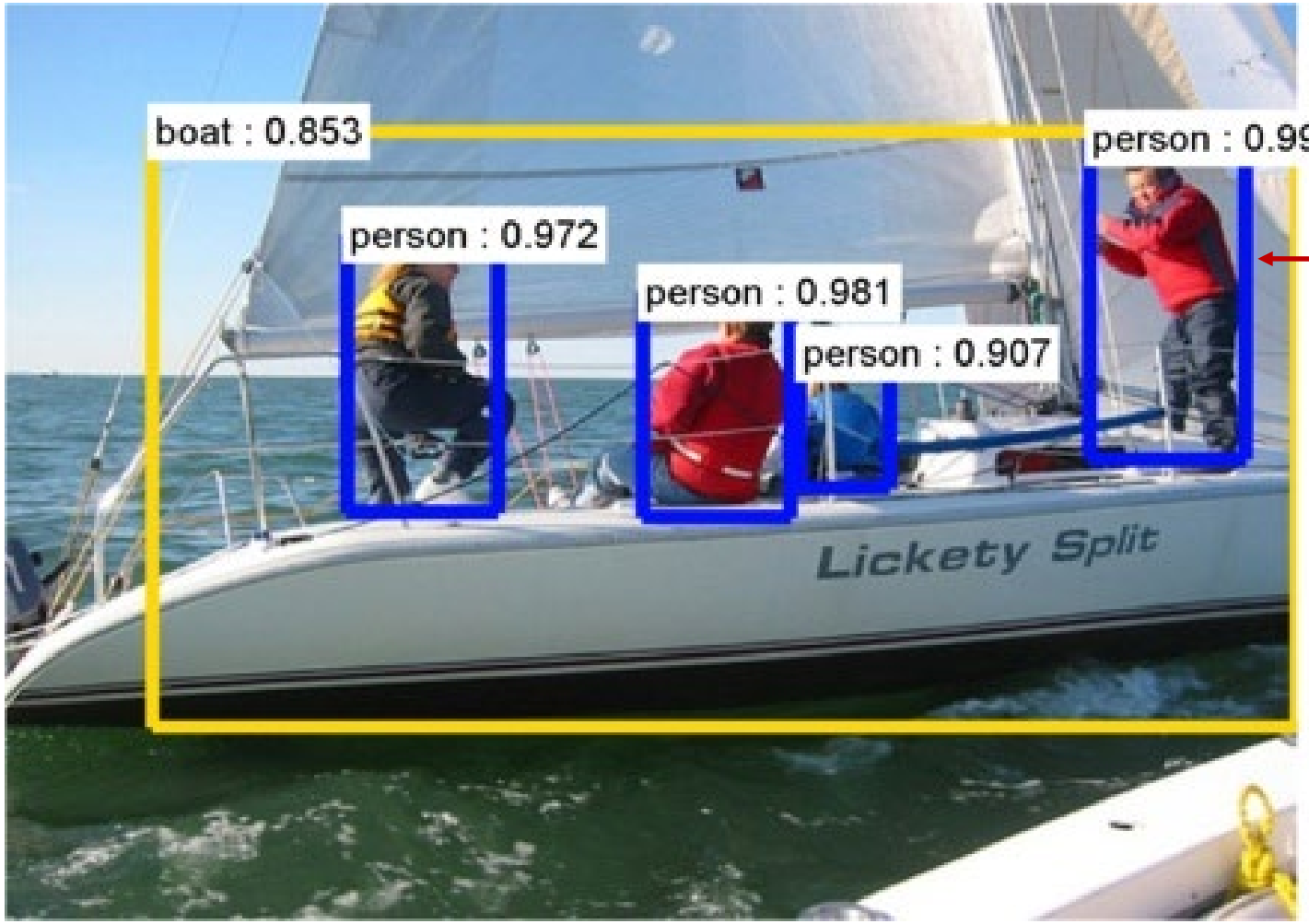
Image segmentation

Image captioning

# Object Detection



# Object Detection with Bounding Boxes

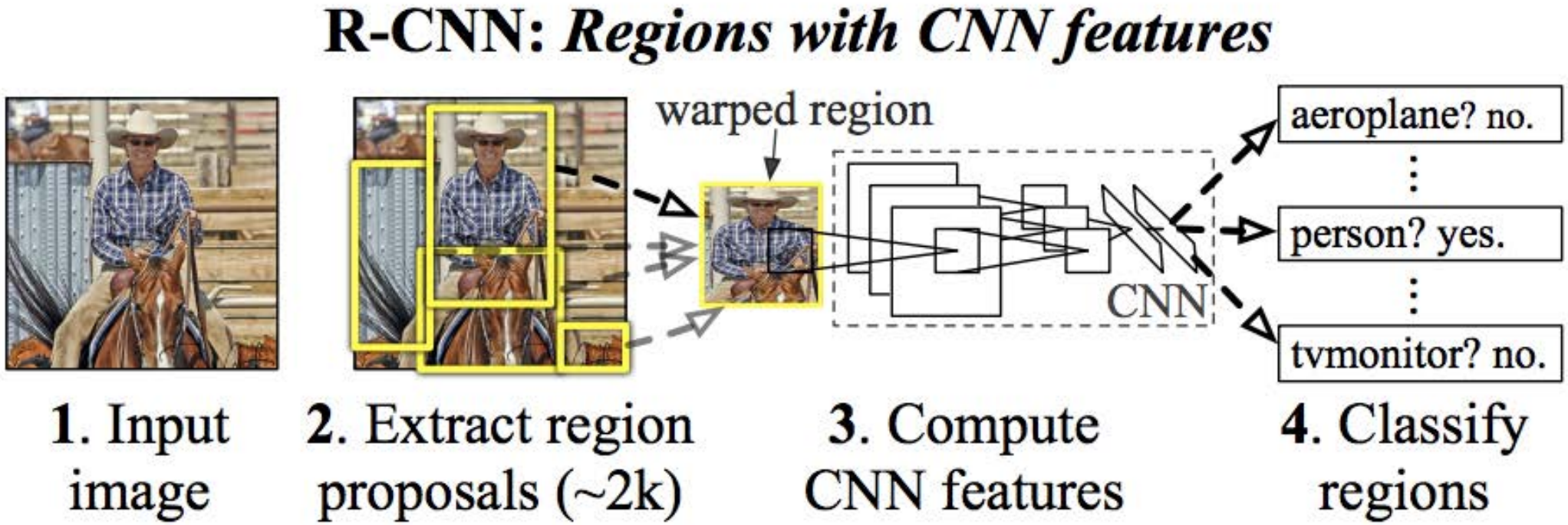



What?

Where?

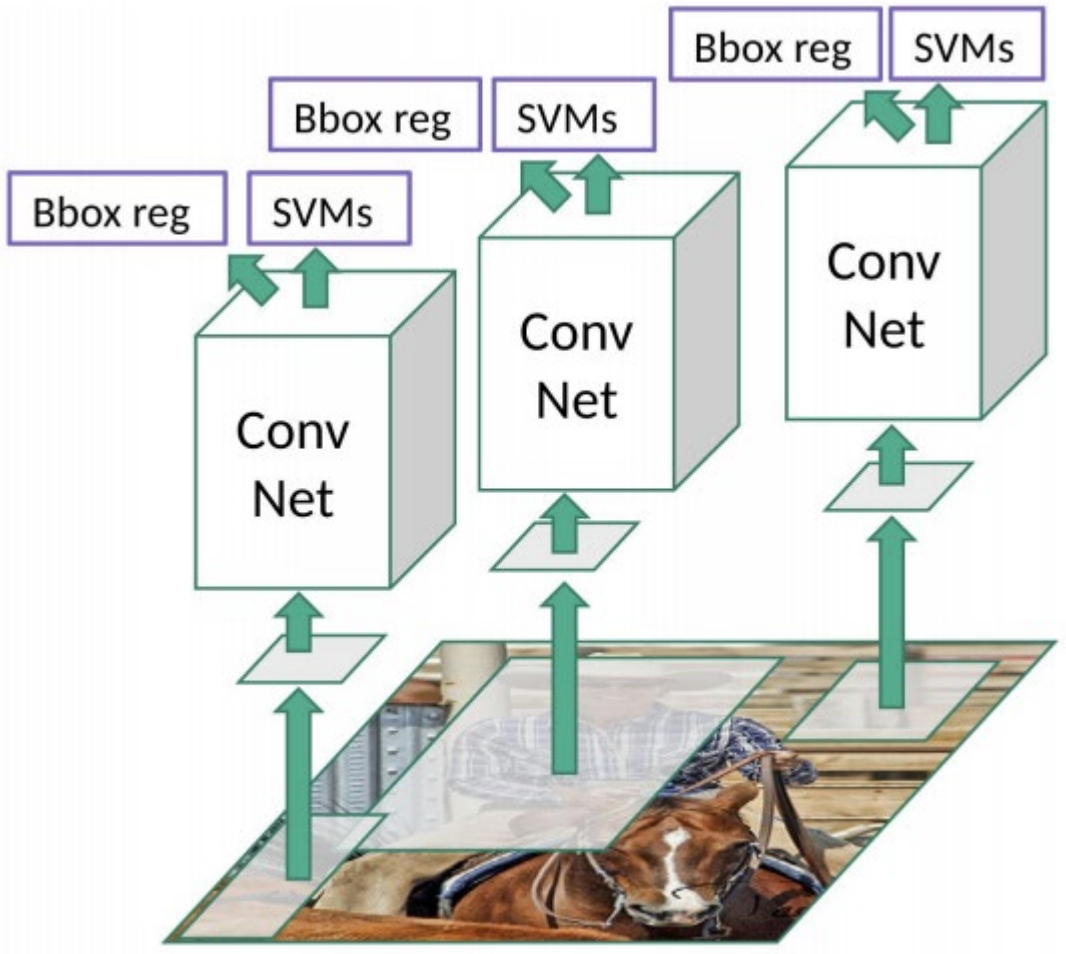
"Object detection"

# R-CNN

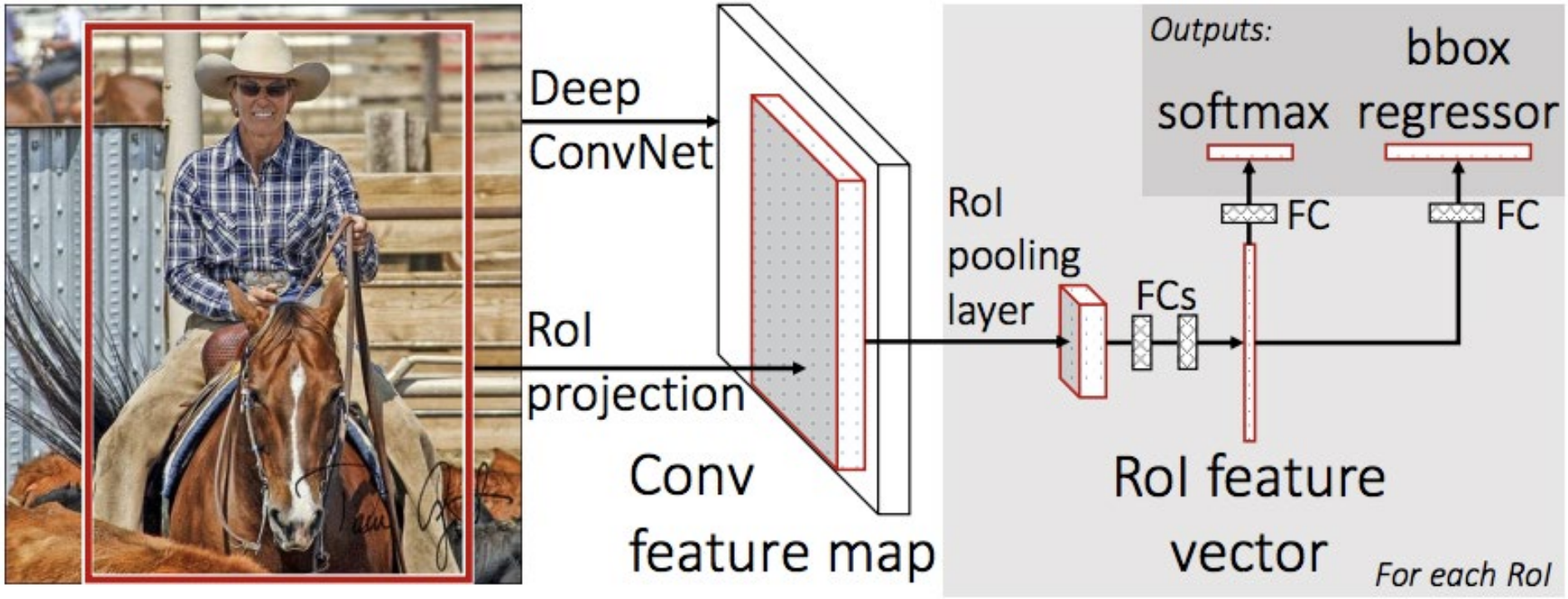

R-CNN: Regions with CNN features
warped region
CNN
aeroplane? no.
person? yes.
tvmonitor? no.
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

# R-CNN

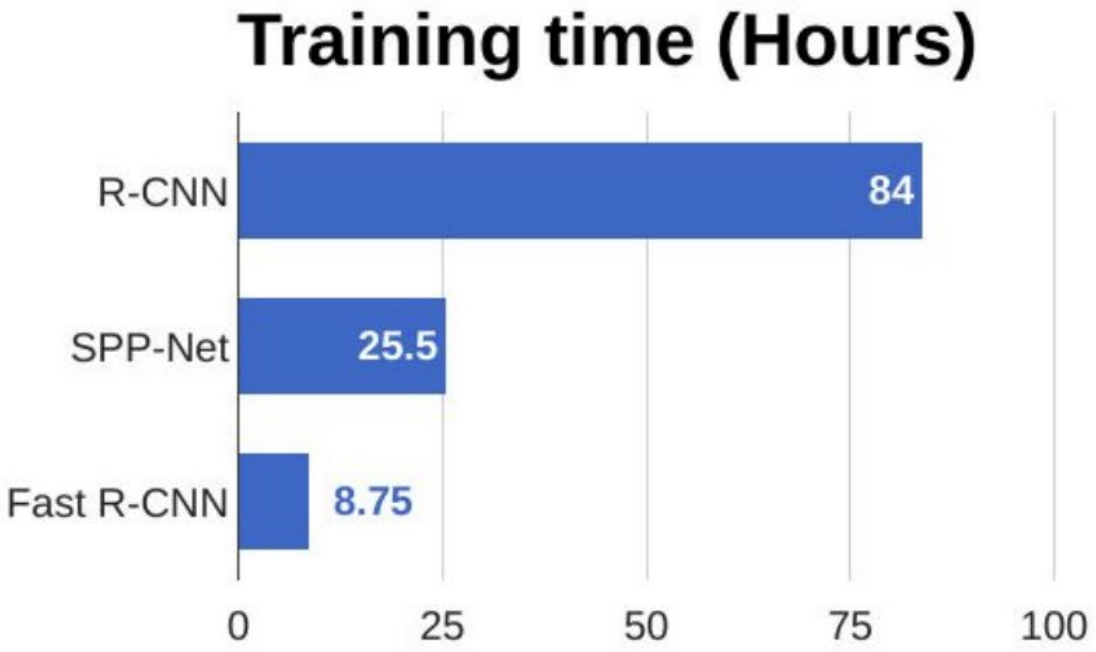
- Huge amount of training time: have to classify 2000 region proposals per image.
- Cannot be real time as: ~47 seconds for each test image.
- The selective search algorithm is a fixed algorithm, no learning

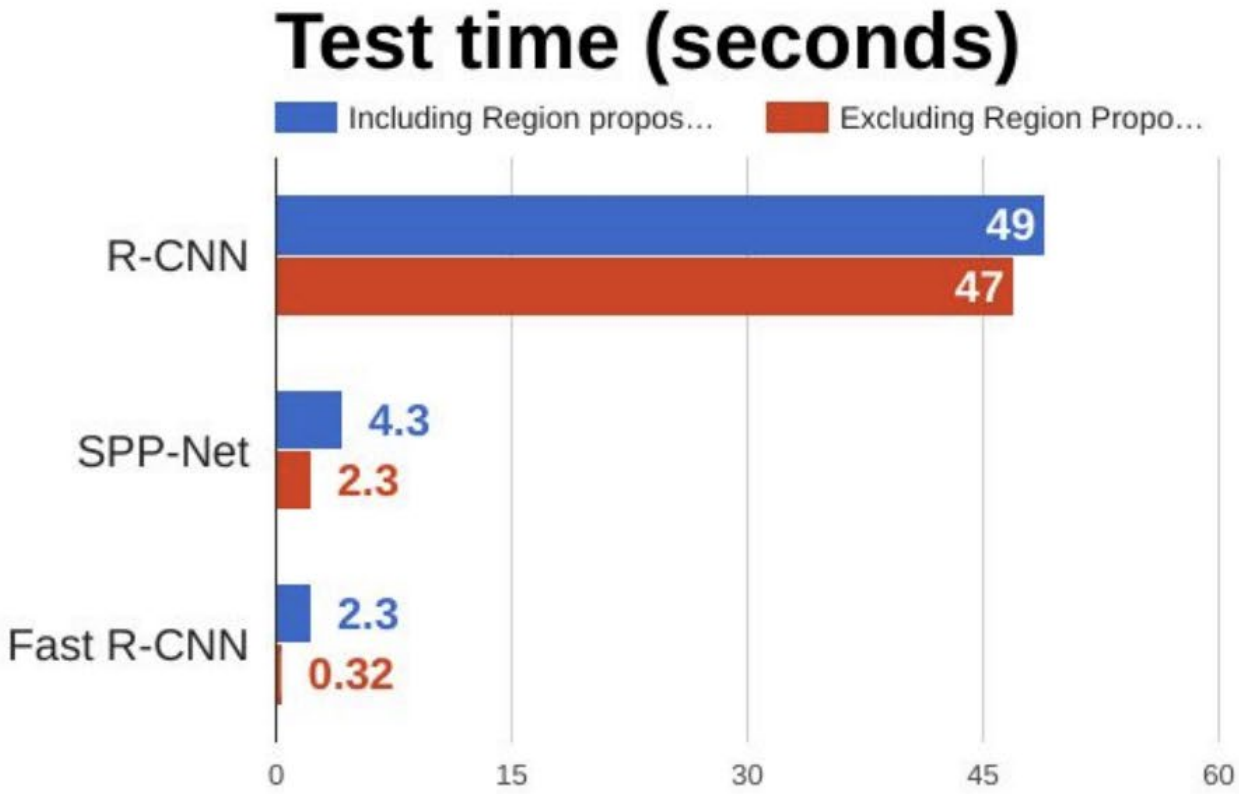


# Fast R-CNN

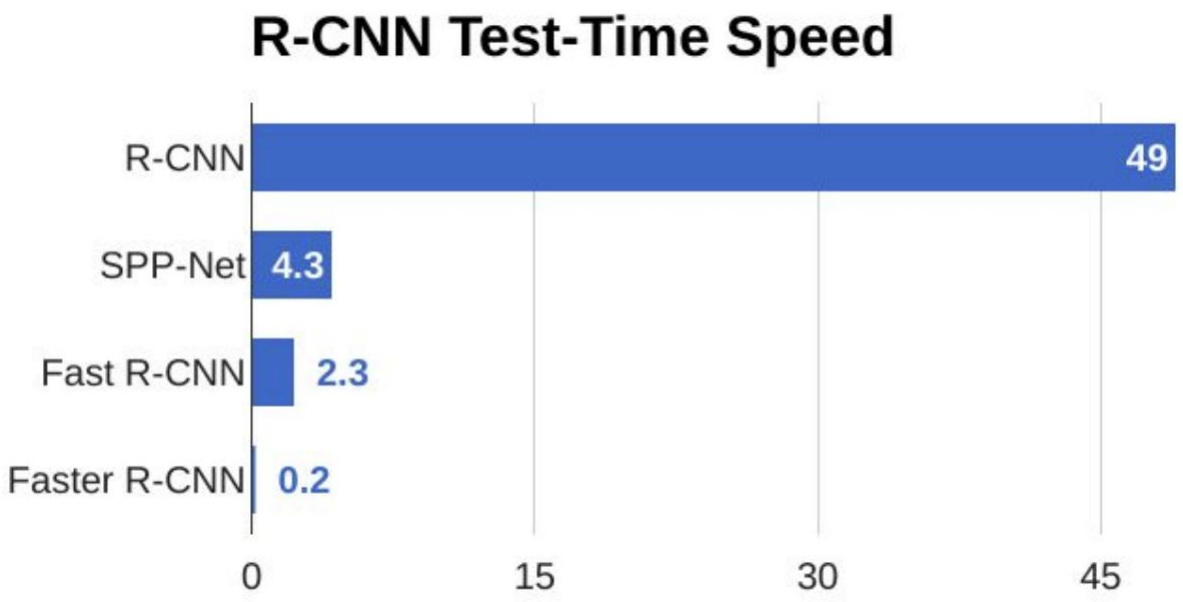

Deep
ConvNet
RoI
projection
Conv
feature map
RoI
pooling
layer
FCs
Outputs:
softmax
bbox
regressor
FC
FC
RoI feature
vector
For each RoI

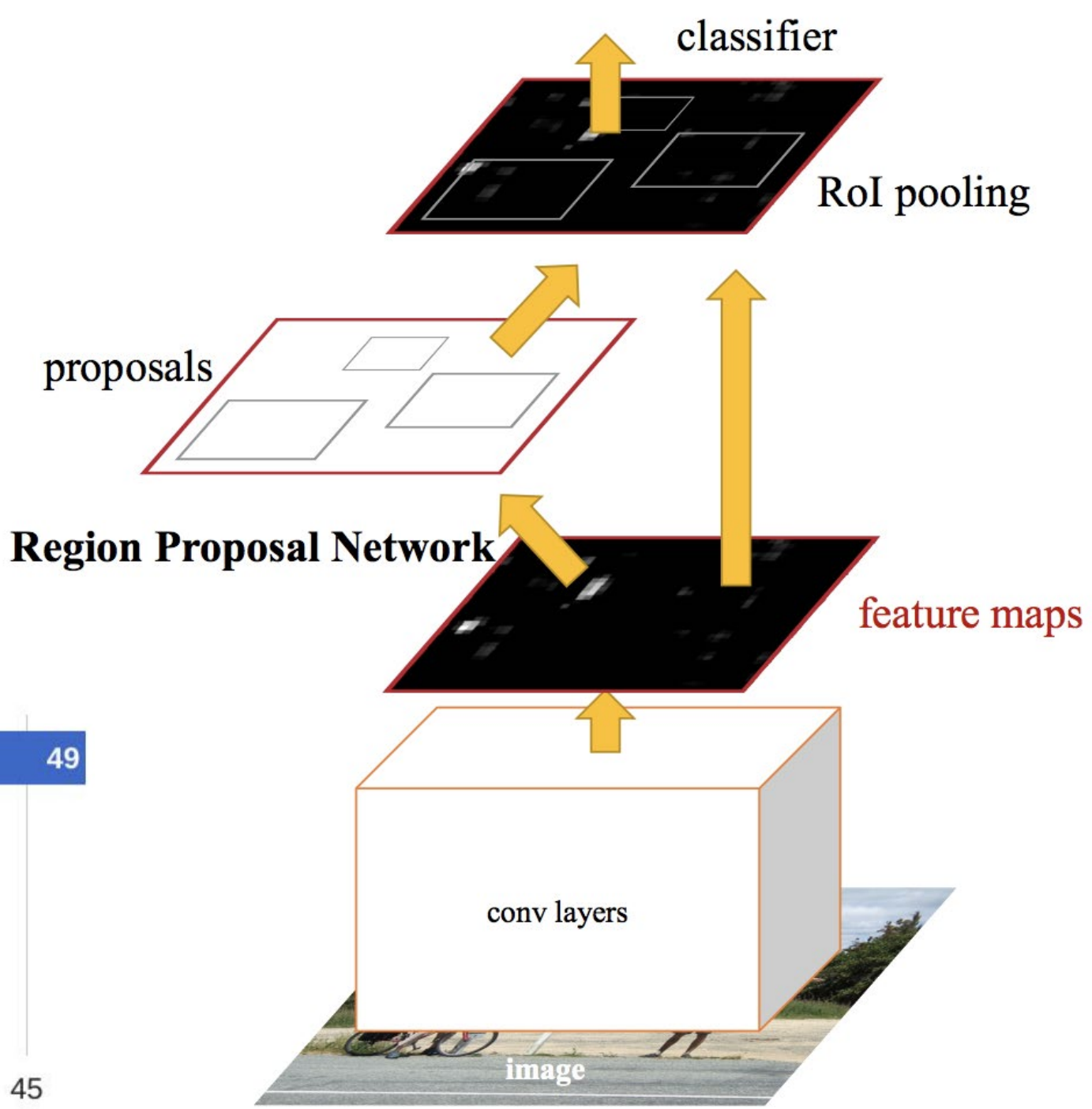
# Fast R-CNN


Training time (Hours)
R-CNN
84
SPP-Net
25.5
Fast R-CNN
8.75
0
25
50
75
100


Test time (seconds)
Including Region propos…
Excluding Region Propo…
R-CNN
49
47
SPP-Net
4.3
2.3
Fast R-CNN
2.3
0.32
0
15
30
45
60

# Faster R-CNN


R-CNN Test-Time Speed
R-CNN
49
SPP-Net
4.3
Fast R-CNN
2.3
Faster R-CNN
0.2
0
15
30
45


classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image

# DETR

- End-to-End Object Detection with Transformers
  - **DE**tection + **TR**ansformer
  - **Streamline the detection pipeline.**
  - **Bipartite matching**
- Eliminate hand-crafted components
  - No anchor mechanism
  - Output detection results directly, without NMS

*Carion, Nicolas, et al. End-to-End Object Detection with Transformers. In ECCV 2020.*

# DETR

- DEtection + TRansformer

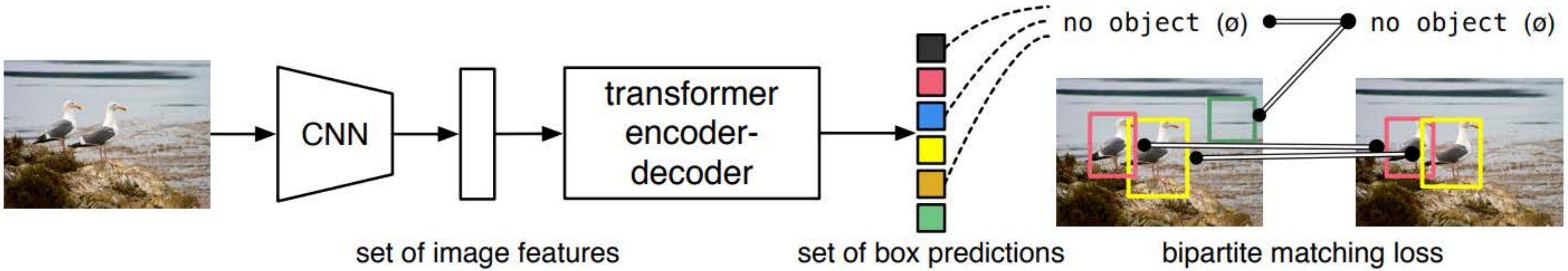


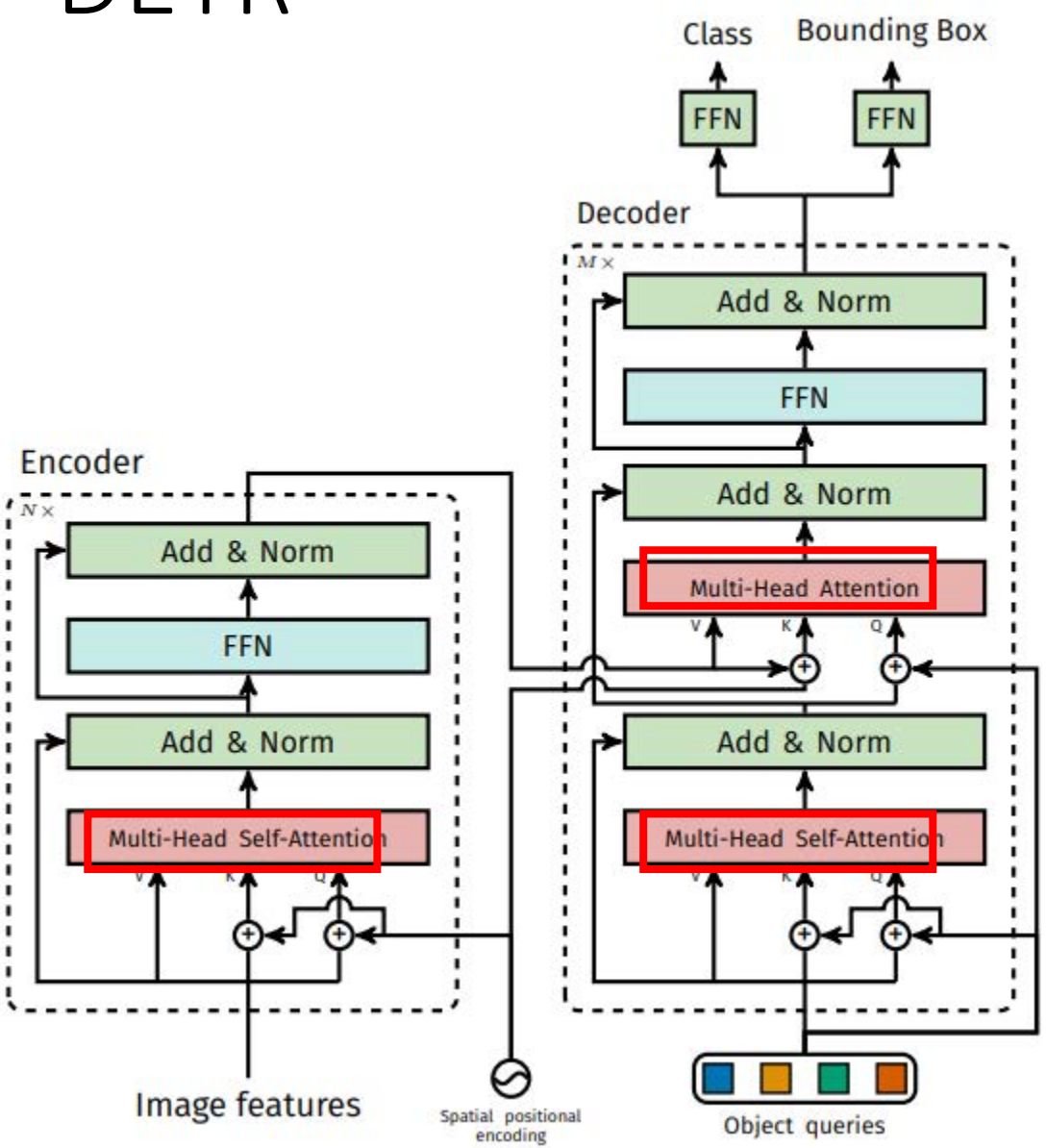

Fig. 1: DETR directly predicts (in parallel) the final set of detections by combining a common CNN with a transformer architecture. During training, bipartite matching uniquely assigns predictions with ground truth boxes. Prediction with no match should yield a "`no object`" ($\varnothing$) class prediction.

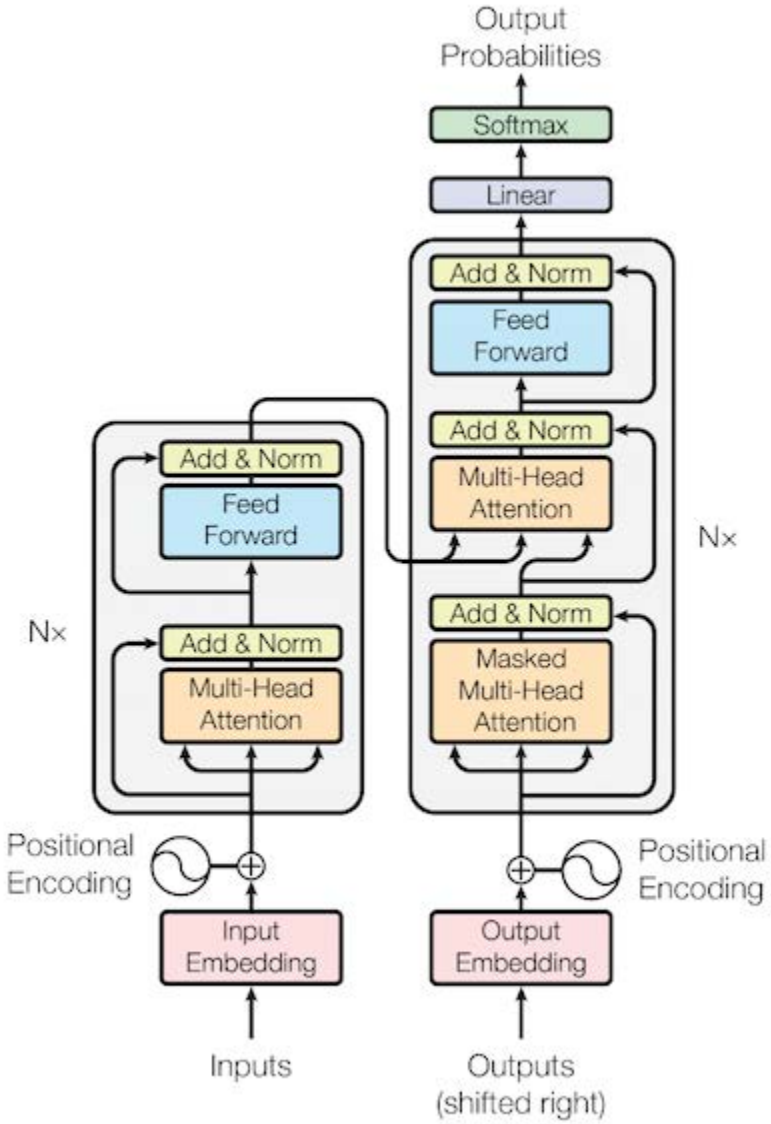
# DETR

- DEtection + TRansformer

  - a set prediction loss that forces unique matching between predicted and ground truth boxes.
  - an architecture that predicts (in a single pass) a set of objects and models their relation.

# DETR


Class
Bounding Box
FFN
FFN
Decoder
M×
Add & Norm
FFN
Add & Norm
Multi-Head Attention
Add & Norm
Multi-Head Self-Attention
Encoder
N×
Add & Norm
FFN
Add & Norm
Multi-Head Self-Attention
Image features
Spatial positional encoding
Object queries


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
N×
N×
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

# DETR

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

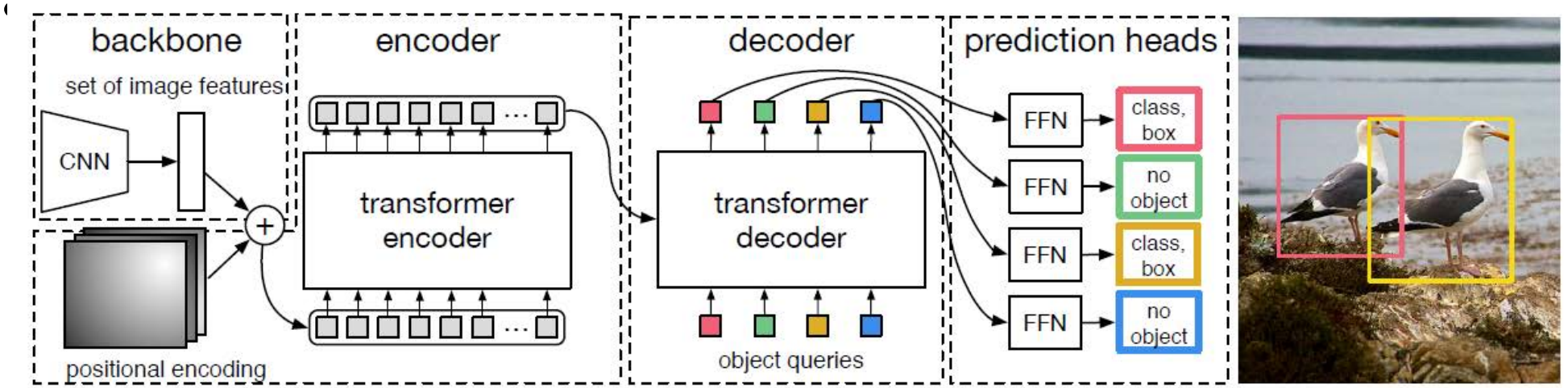



Fig. 2: DETR uses a conventional CNN backbone to learn a 2D representation of an input image. The model flattens it and supplements it with a positional encoding before

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[ -\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

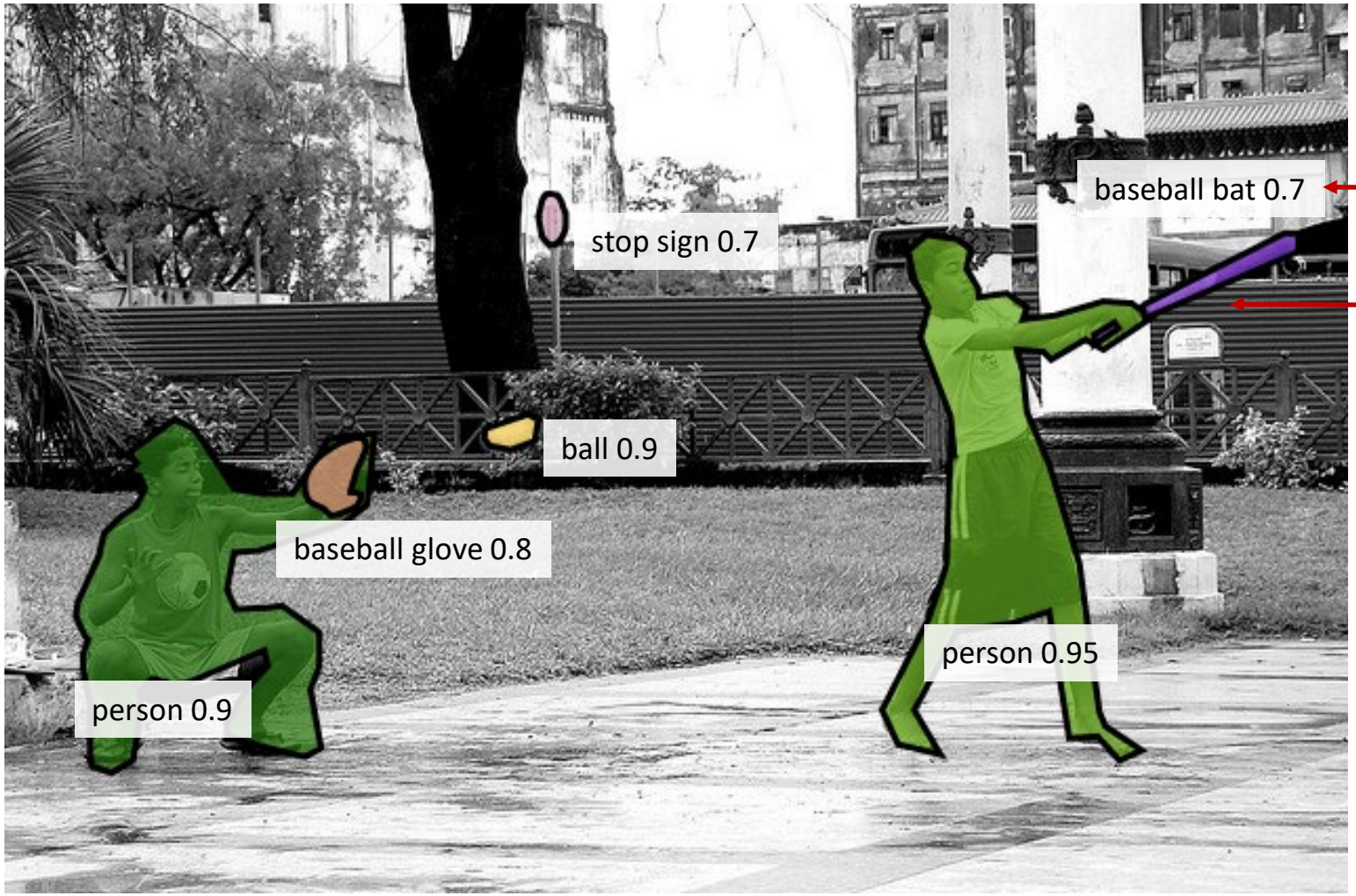
*Nicolas Carion et al. End-to-End Object Detection with Transformers. ECCV 2020.*

# DETR - Experiments

Table 1: Comparison with Faster R-CNN with a ResNet-50 and ResNet-101 backbones on the COCO validation set. The top section shows results for Faster R-CNN models in Detectron2 [50], the middle section shows results for Faster R-CNN models with GIoU [38], random crops train-time augmentation, and the long `9x` training schedule. DETR models achieve comparable results to heavily tuned Faster R-CNN baselines, having lower $AP_S$ but greatly improved $AP_L$. We use torchscript Faster R-CNN and DETR models to measure FLOPS and FPS. Results without R101 in the name correspond to ResNet-50.

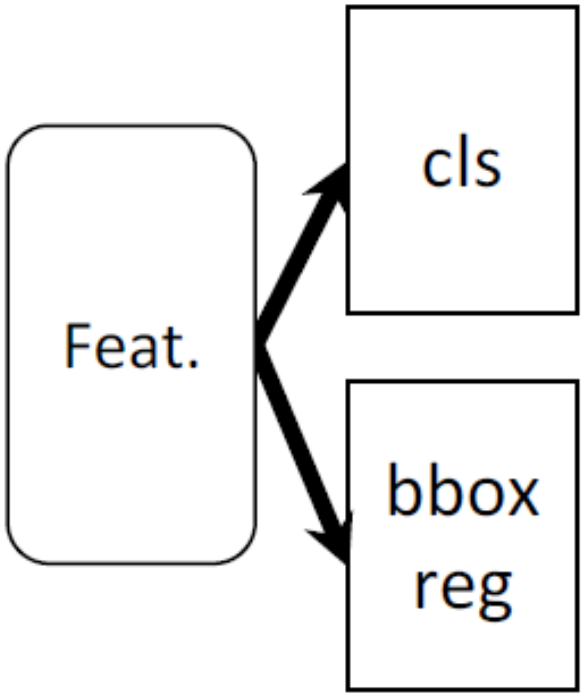
| Model | GFLOPS/FPS | #params | AP | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|---|---|
| Faster RCNN-DC5 | 320/16 | 166M | 39.0 | 60.5 | 42.3 | 21.4 | 43.5 | 52.5 |
| Faster RCNN-FPN | 180/26 | 42M | 40.2 | 61.0 | 43.8 | 24.2 | 43.5 | 52.0 |
| Faster RCNN-R101-FPN | 246/20 | 60M | 42.0 | 62.5 | 45.9 | 25.2 | 45.6 | 54.6 |
| Faster RCNN-DC5+ | 320/16 | 166M | 41.1 | 61.4 | 44.3 | 22.9 | 45.9 | 55.0 |
| Faster RCNN-FPN+ | 180/26 | 42M | 42.0 | 62.1 | 45.5 | 26.6 | 45.4 | 53.4 |
| Faster RCNN-R101-FPN+ | 246/20 | 60M | 44.0 | 63.9 | **47.8** | **27.2** | 48.1 | 56.0 |
| DETR | 86/28 | 41M | 42.0 | 62.4 | 44.2 | 20.5 | 45.8 | 61.1 |
| DETR-DC5 | 187/12 | 41M | 43.3 | 63.1 | 45.9 | 22.5 | 47.3 | 61.1 |
| DETR-R101 | 152/20 | 60M | 43.5 | 63.8 | 46.4 | 21.9 | 48.0 | 61.8 |
| DETR-DC5-R101 | 253/10 | 60M | **44.9** | **64.7** | 47.7 | 23.7 | **49.5** | **62.3** |

# Object Detection with Segmentation Masks

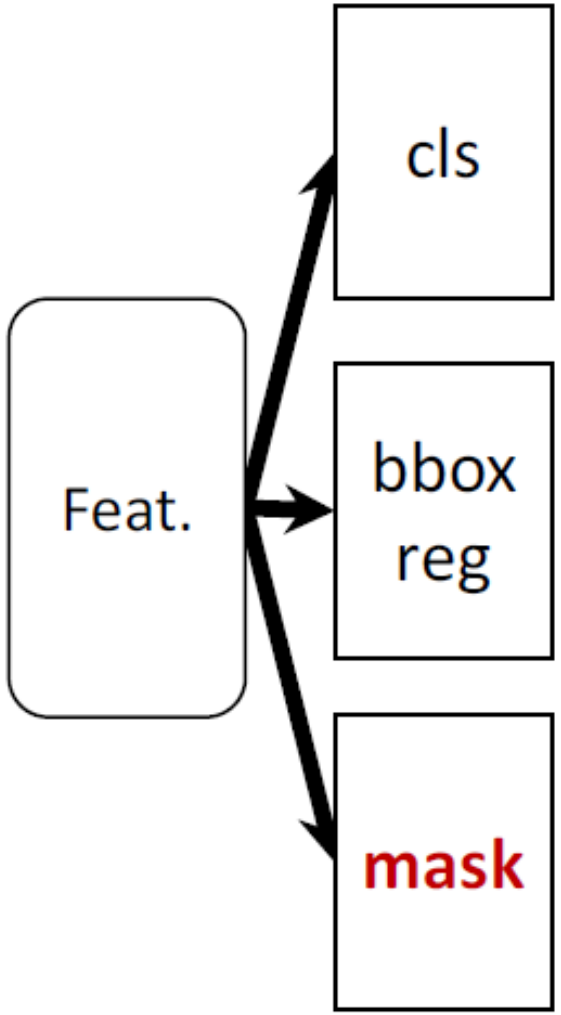



"Instance segmentation"

# Mask R-CNN

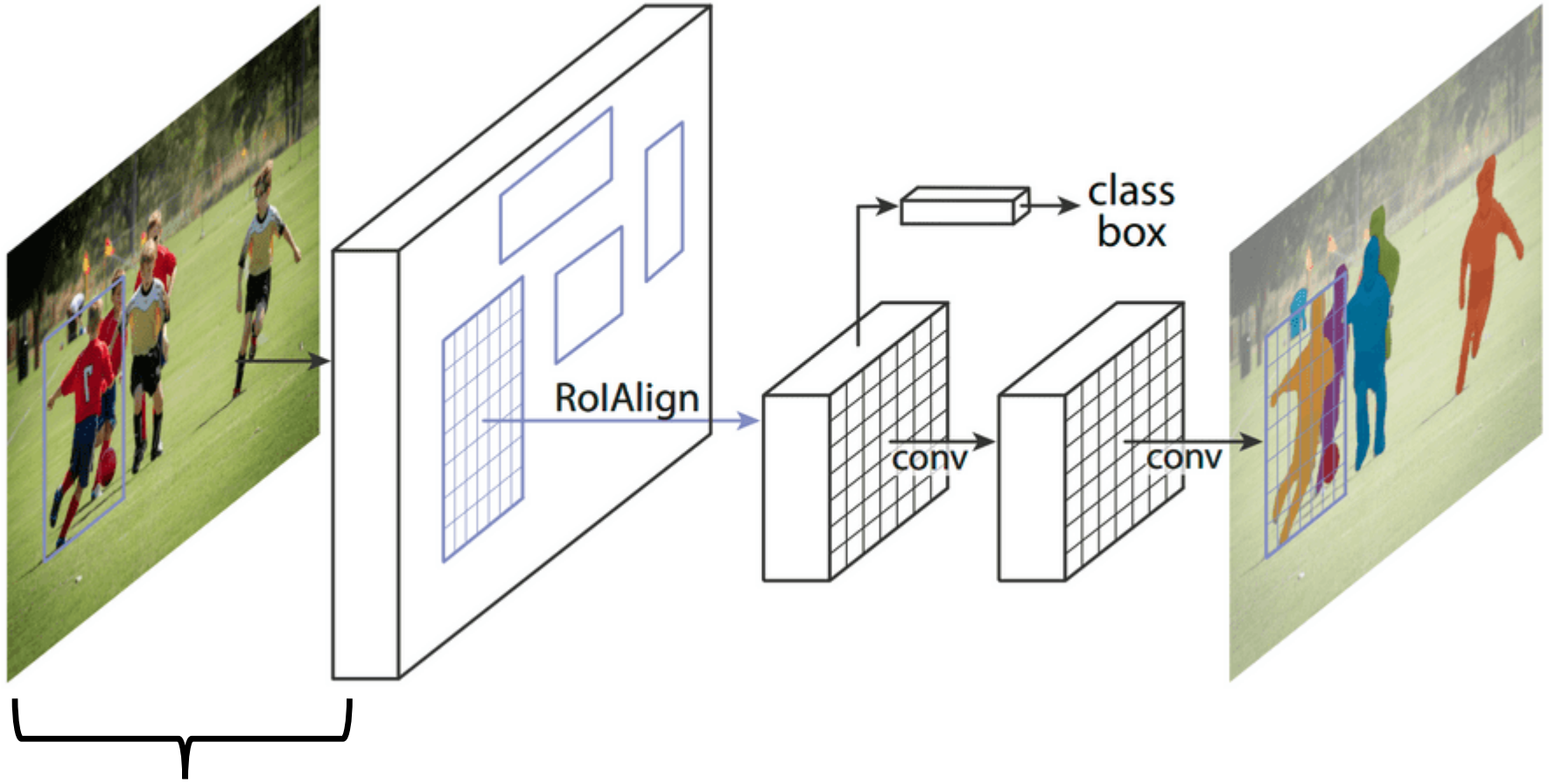



Faster R-CNN




Mask R-CNN

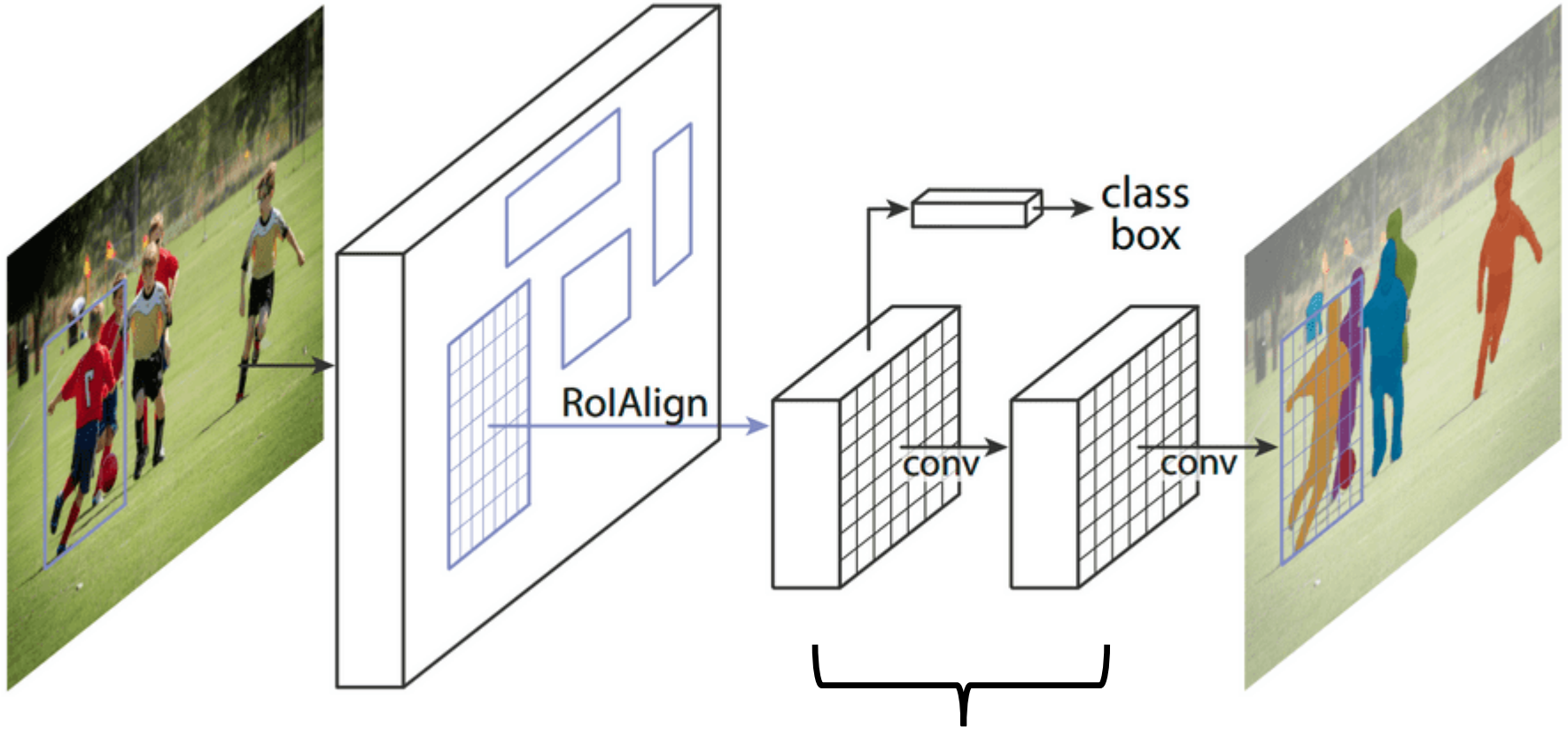
# Mask R-CNN




Slide credit:
Kaiming He

1. Fully-Conv Features:
equivariant to global (image) translation

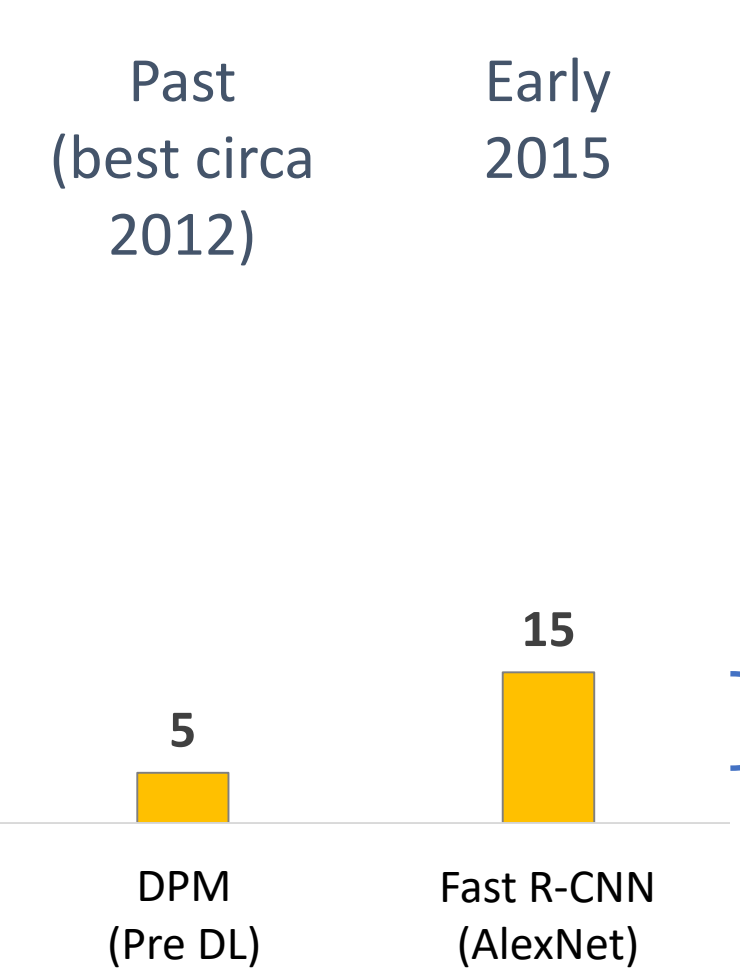
# Mask R-CNN




2. Fully-Conv on RoI:
equivariant to translation within RoI

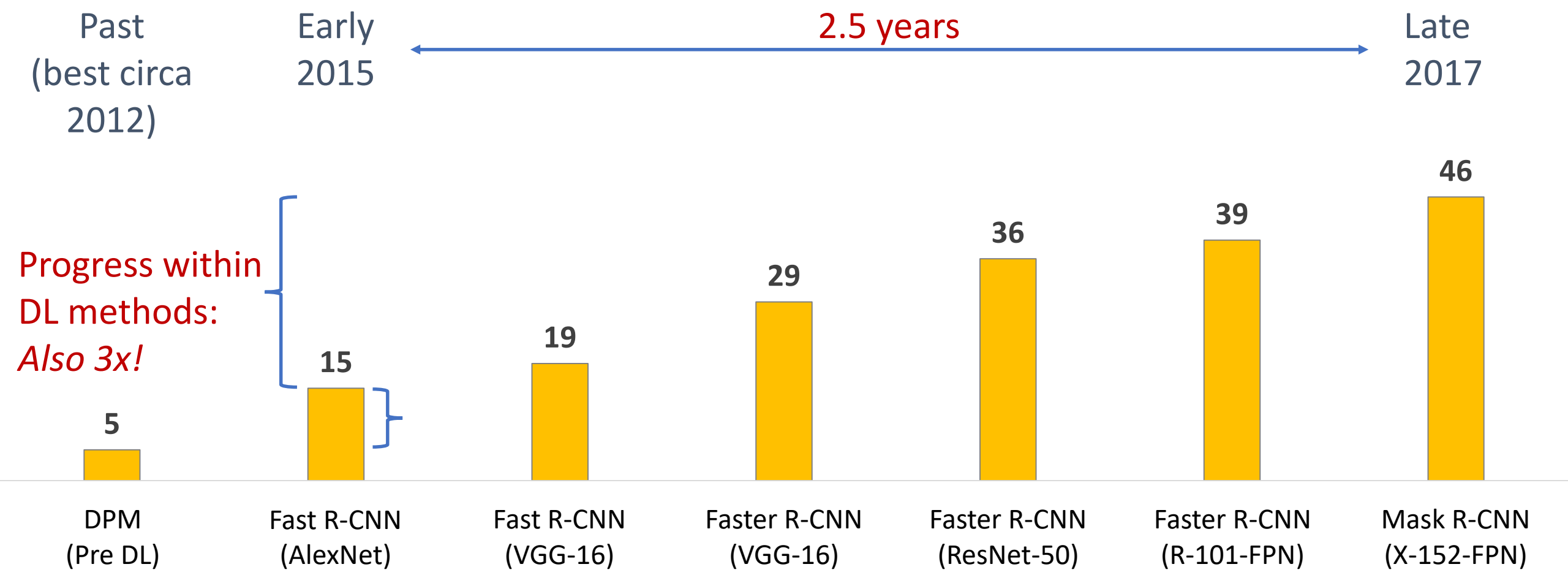
Slide credit:
Kaiming He

# COCO Object Detection Average Precision (%)




Movement to
Deep Learning methods:
*3x improvement in AP*

# COCO Object Detection Average Precision (%)



# Part 4: Unsupervised Visual Learning

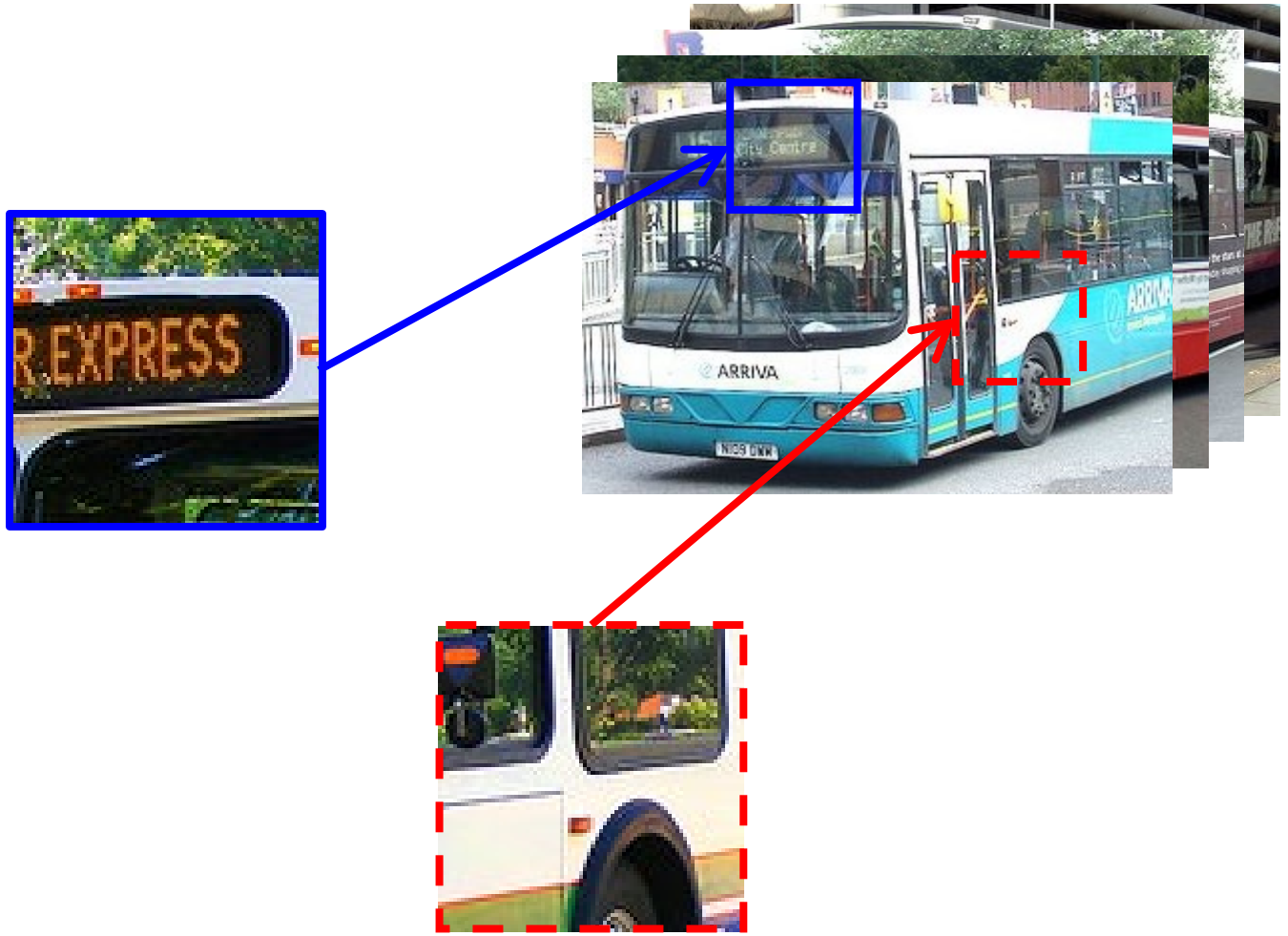
# Why Unsupervised Learning?

- Unlabeled data: virtually unlimited

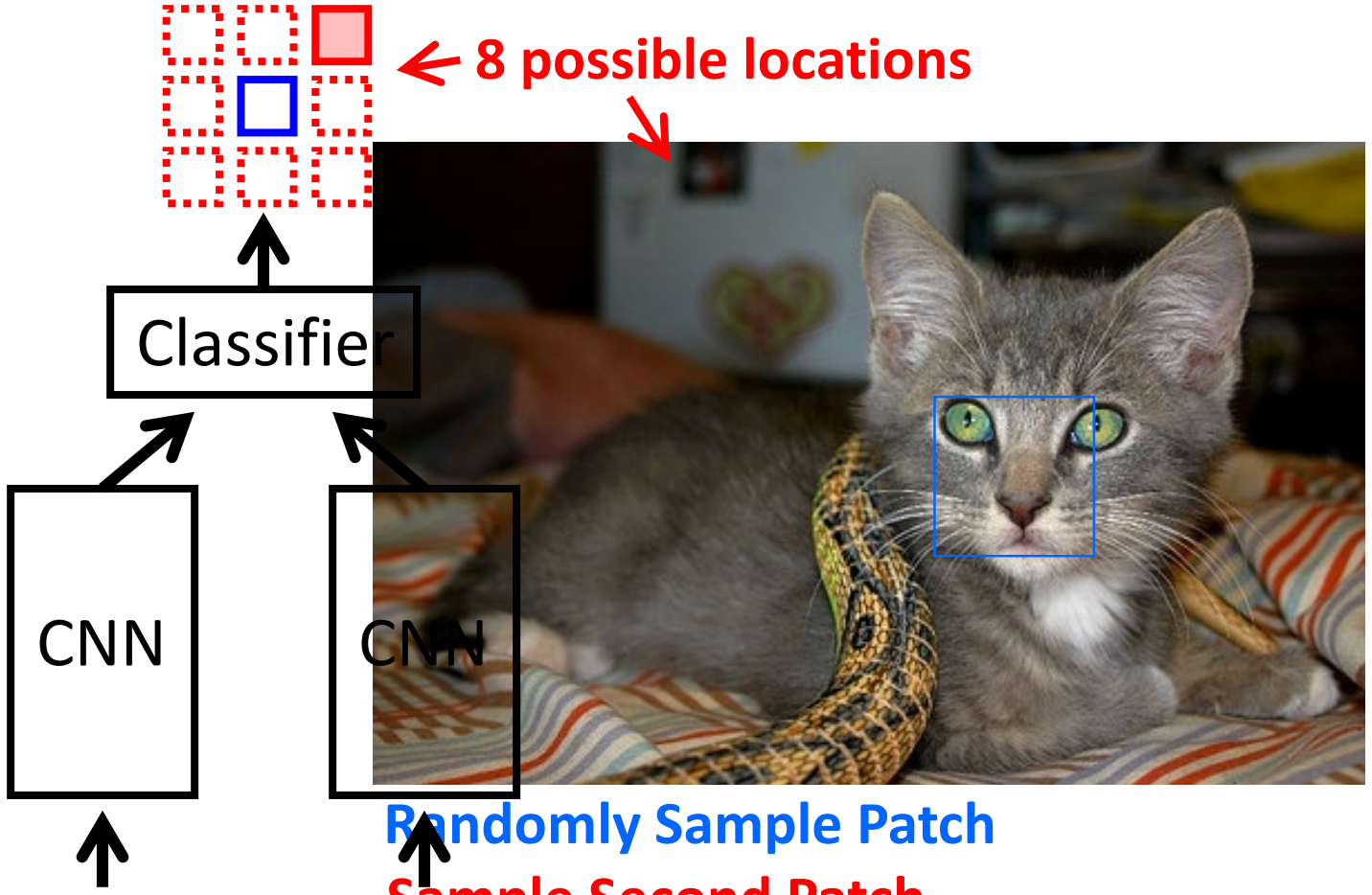
- Supervision is expensive and slow
  - May take years to obtain a large labeled dataset
  - Labeling is noisy; takes iterations; proper instructions are hard to define; redundancy and checks on label quality is needed
  - Labeling is expensive

# Context Prediction for Images

# Semantics from a non-semantic task

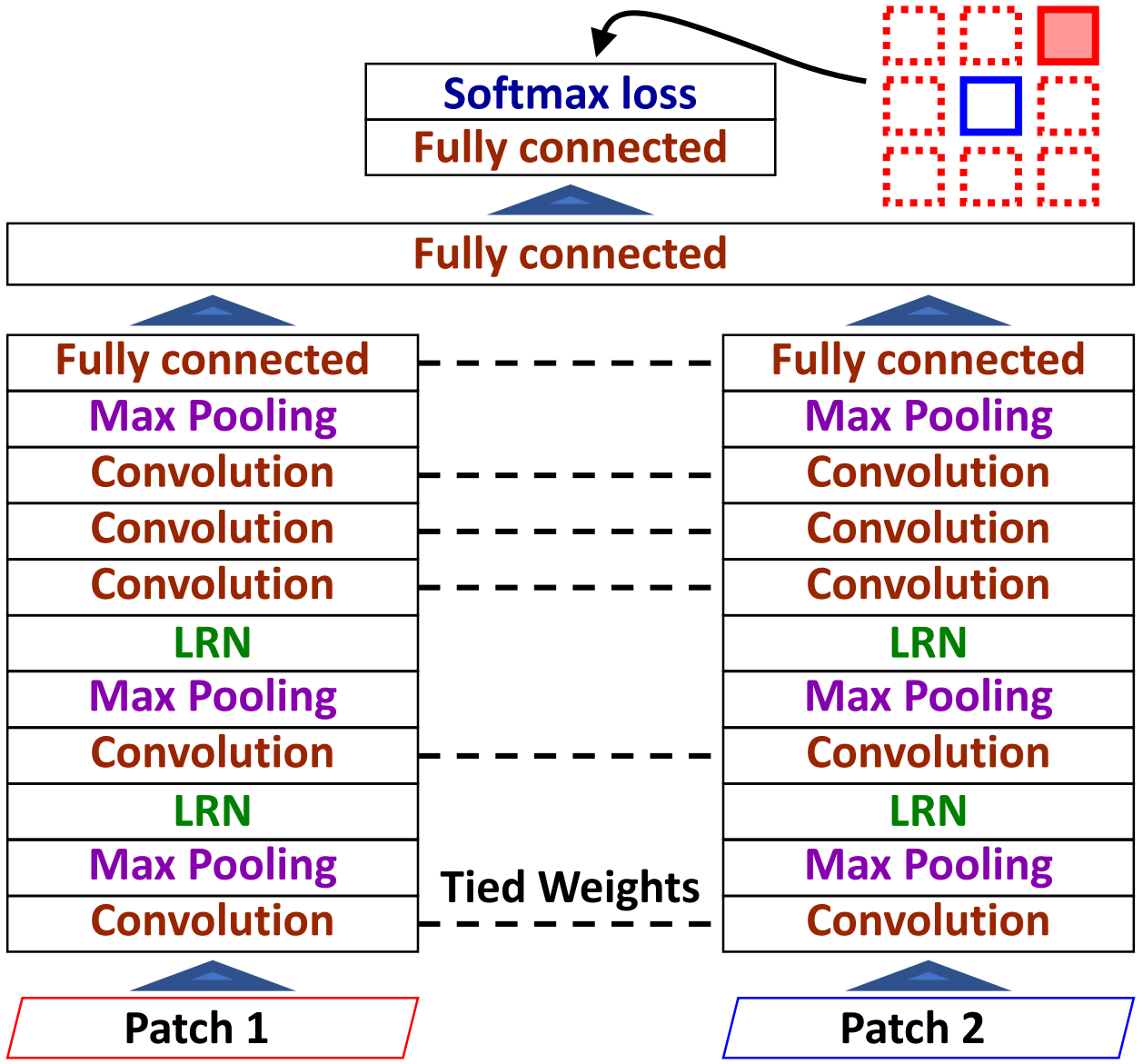


# Relative Position Task

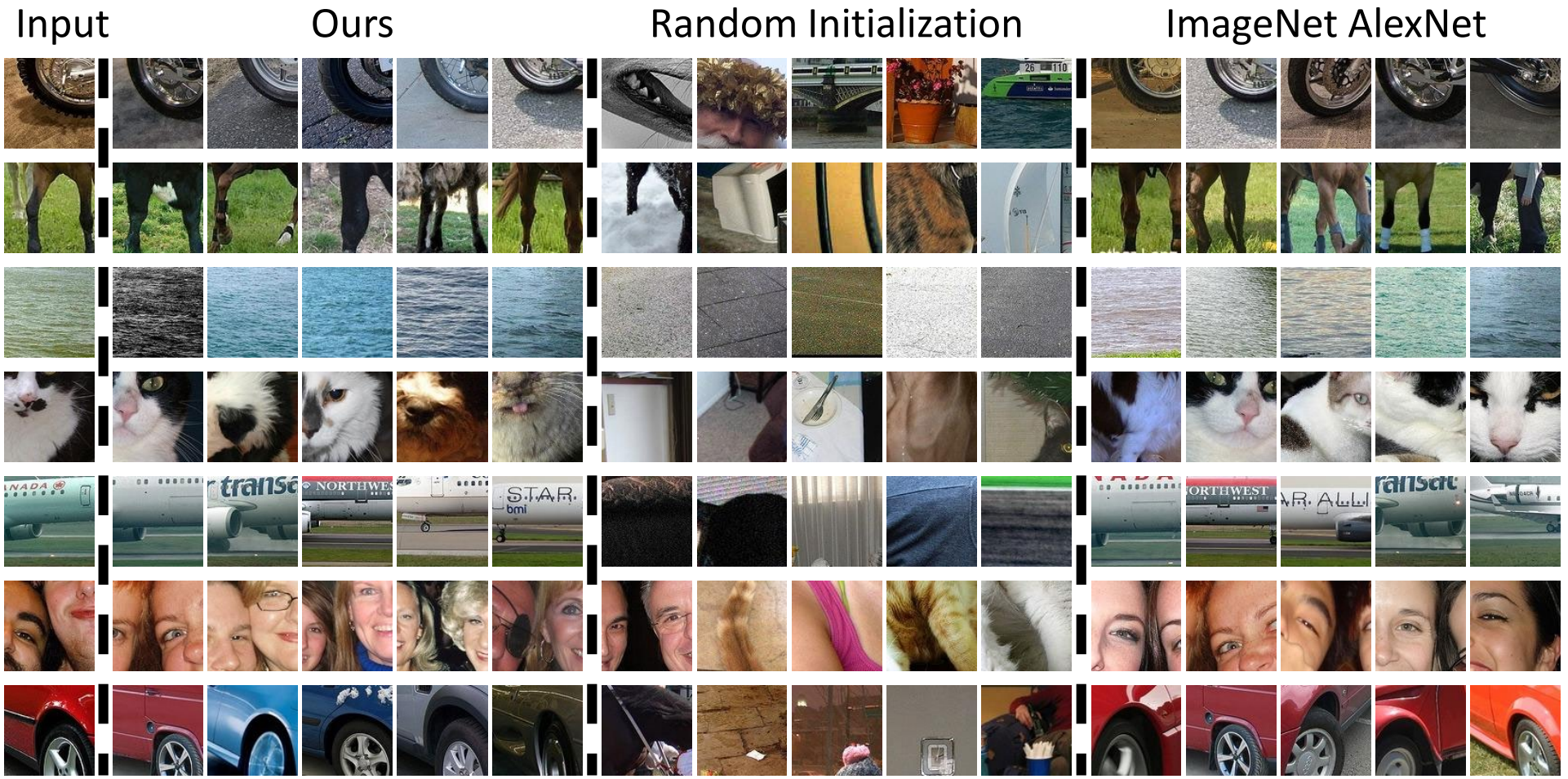



Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised Visual Representation Learning by Context Prediction. ICCV 2015

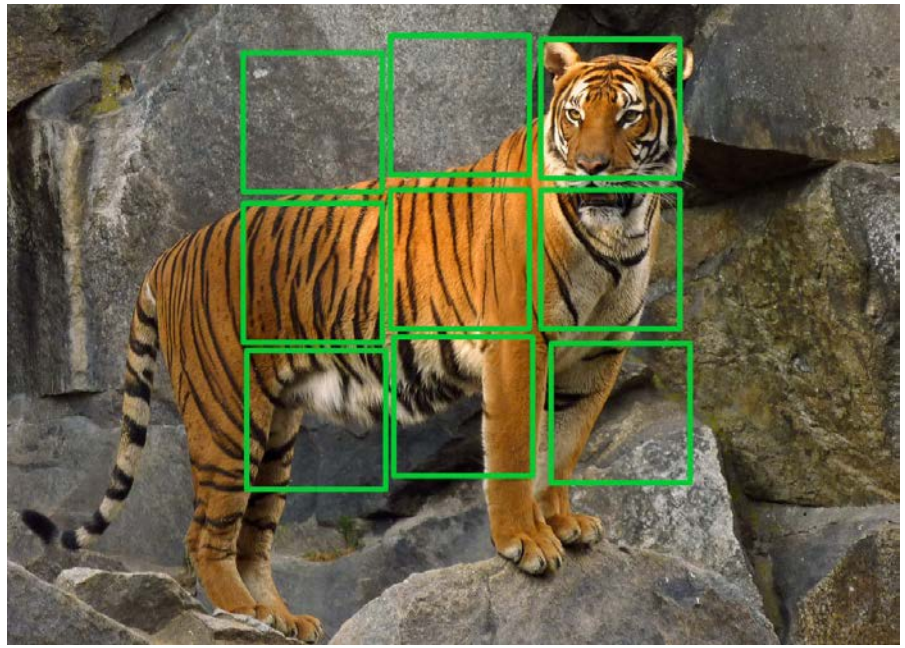
# Architecture

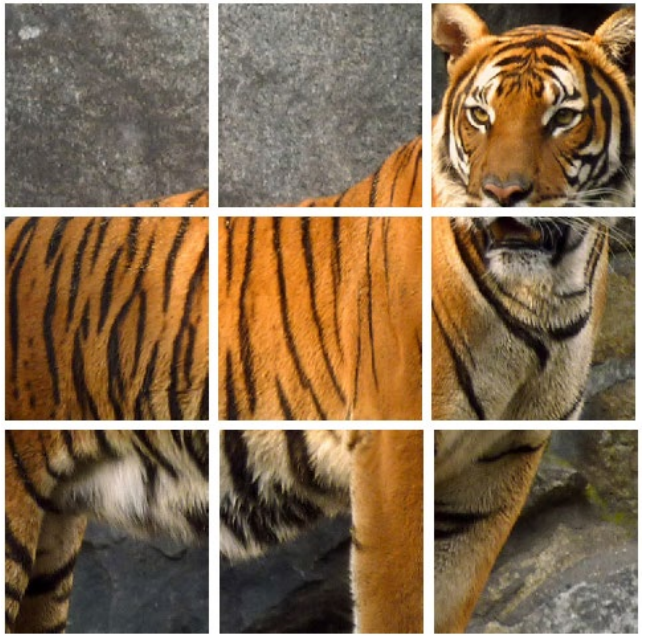


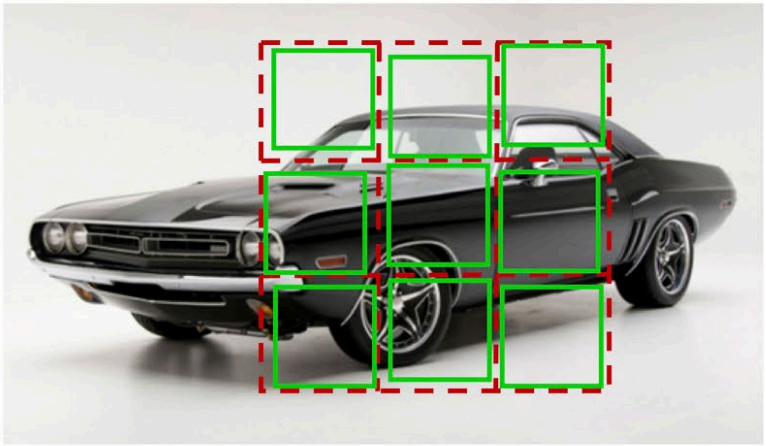
# What is learned?



# Jigsaw Puzzles



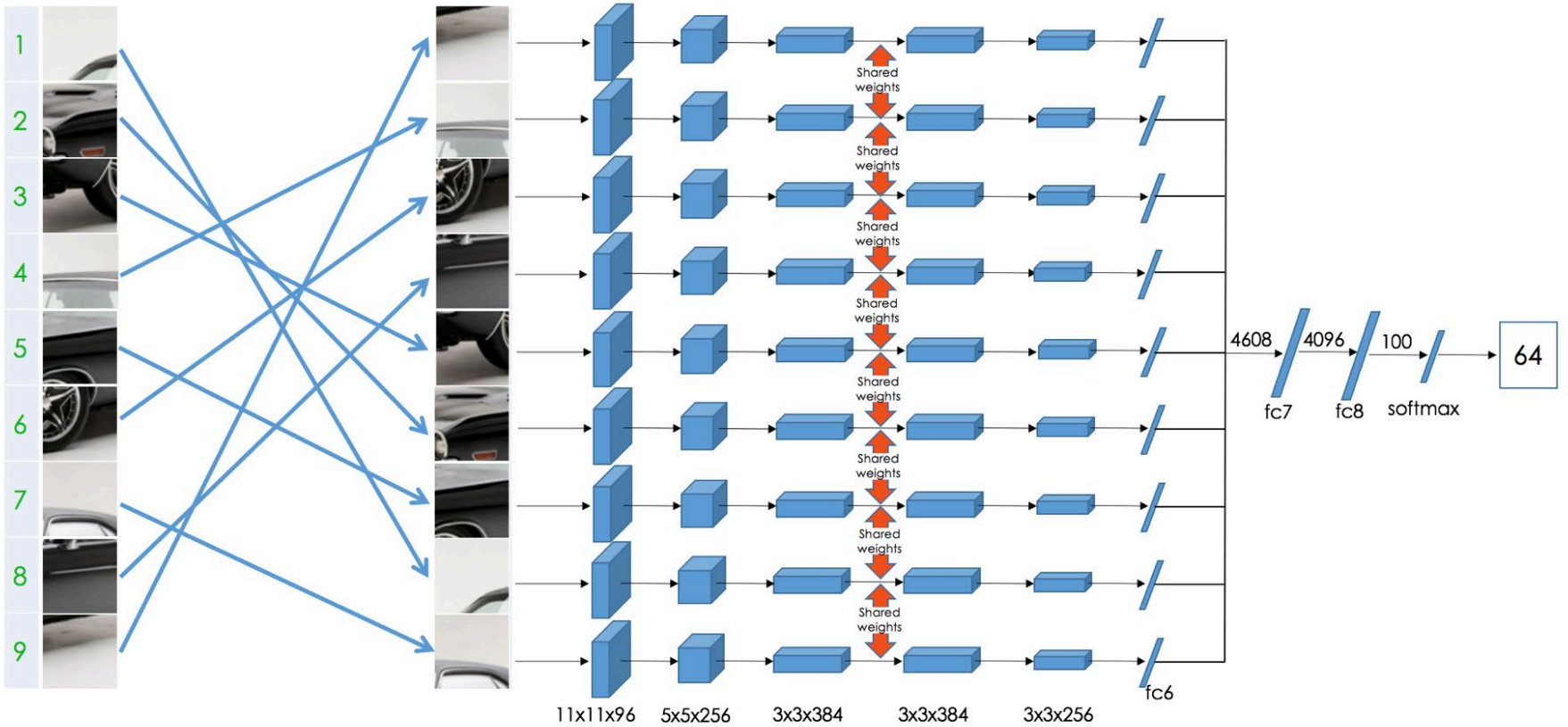




Permutation Set

| index | permutation |
|---|---|
| | |
| 64 | 9,4,6,8,3,2,5,1,7 |
| | |
| | |

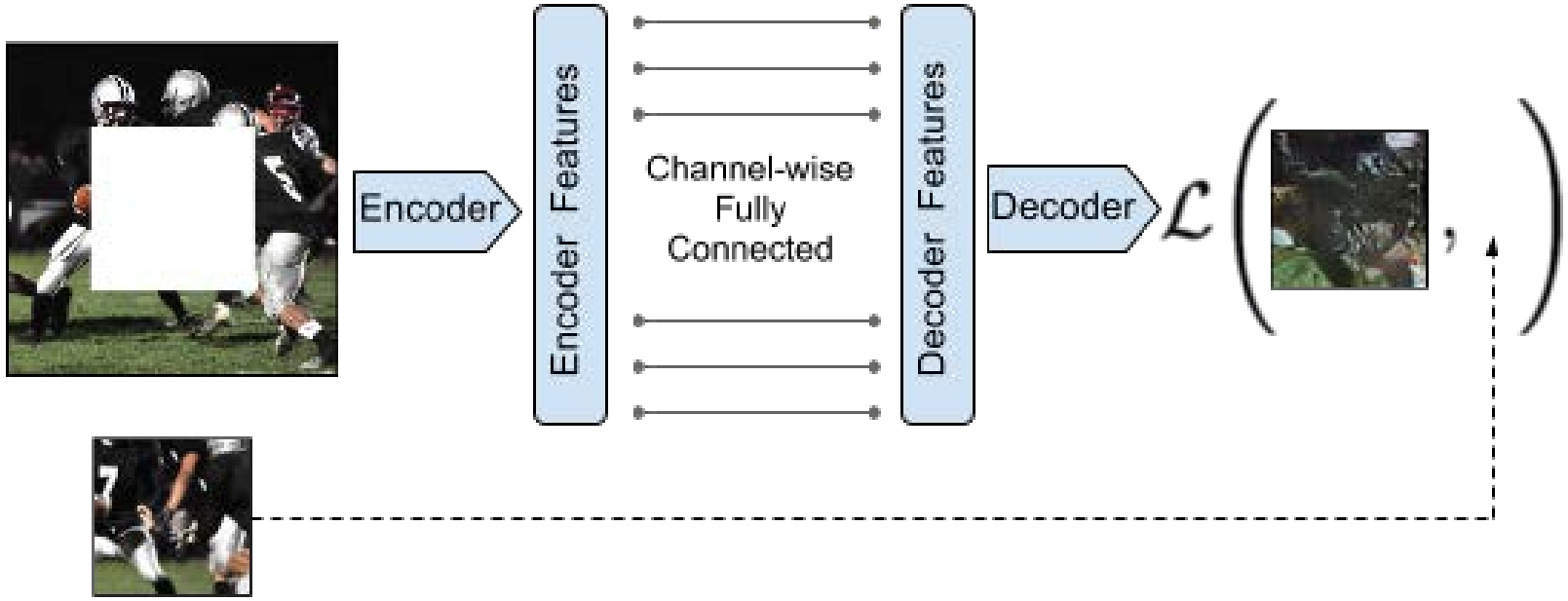
Reorder patches according to the selected permutation




Noroozi, Mehdi, and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. ECCV 2016.

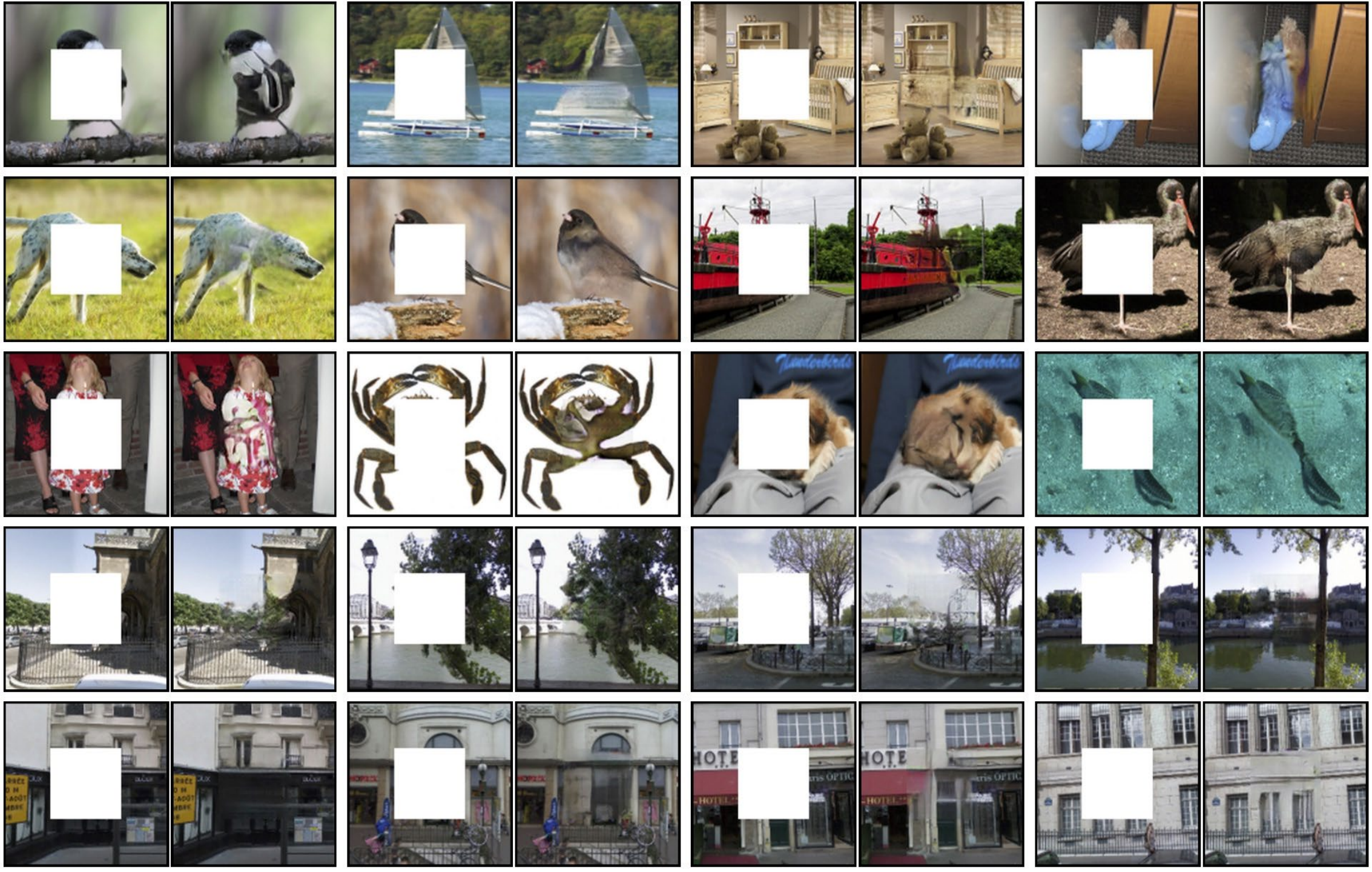
# Feature Learning by **Inpainting**

# Inpainting




- Encoder can be substituted with any network architecture like AlexNet etc.
- Decoder is a set of UpConv/deconv/frac-strided-conv layers

Pathak, Deepak, et al. Context encoders: Feature learning by inpainting. CVPR 2016.








# Colorful Image Colorization

Richard Zhang, Phillip Isola, Alexei (Alyosha) Efros

richzhang.github.io/colorization

# Contrastive Learning (Unsupervised Learning)

- 1. Contrastive Predictive Coding (DeepMind, 1807)
- Core Idea:
  - 1. Generative Model as Unsupervised Learning?
  - 2. How to evaluate an encoder?

- InfoNCE Loss:

$$\mathcal{L}_{\mathrm{N}} = -\underset{X}{\mathbb{E}}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

# Contrastive Learning (Unsupervised Learning)

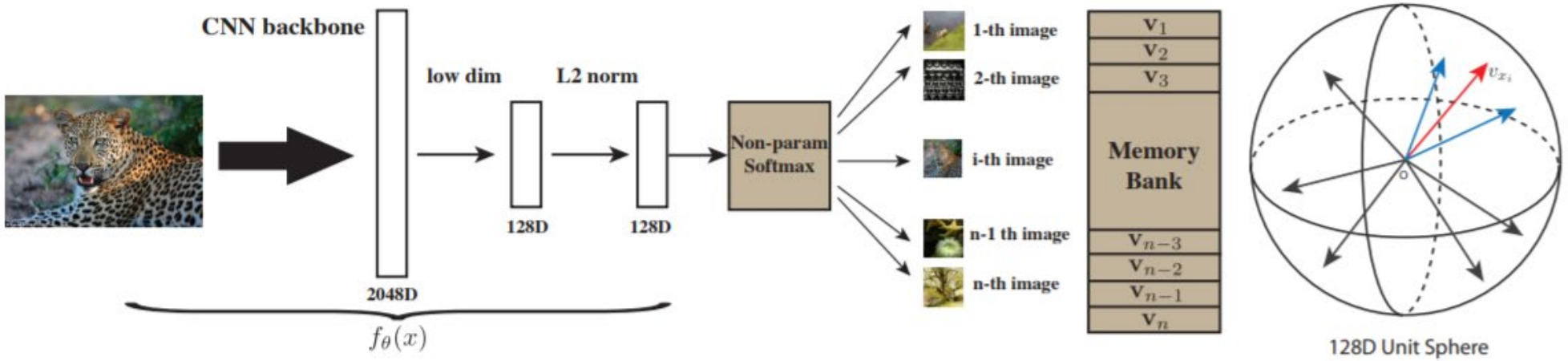
- 2. Non-Parametric Instance Discrimination (Zhirong Wu, CVPR2018)

$$P(i|\mathbf{v}) = \frac{\exp\left(\mathbf{w}_i^T \mathbf{v}\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{w}_j^T \mathbf{v}\right)}.$$

Parametric Classifier

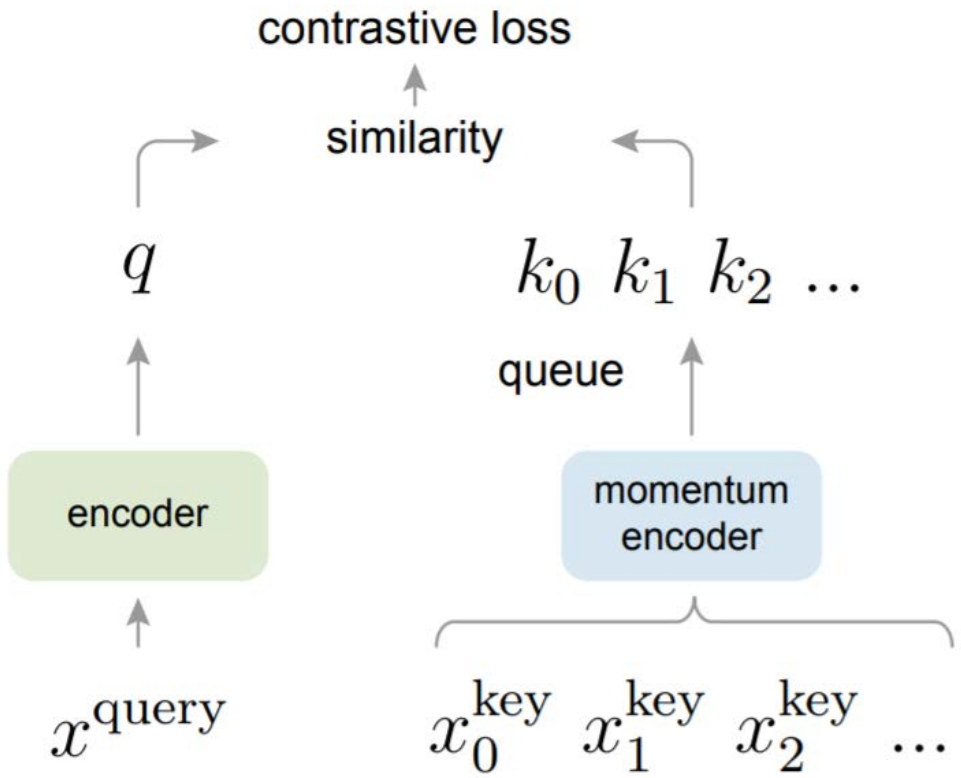
$$P(i|\mathbf{v}) = \frac{\exp\left(\mathbf{v}_i^T \mathbf{v}/\tau\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{v}_j^T \mathbf{v}/\tau\right)},$$

Non-Parametric Classifier



# Contrastive Learning (Unsupervised Learning)

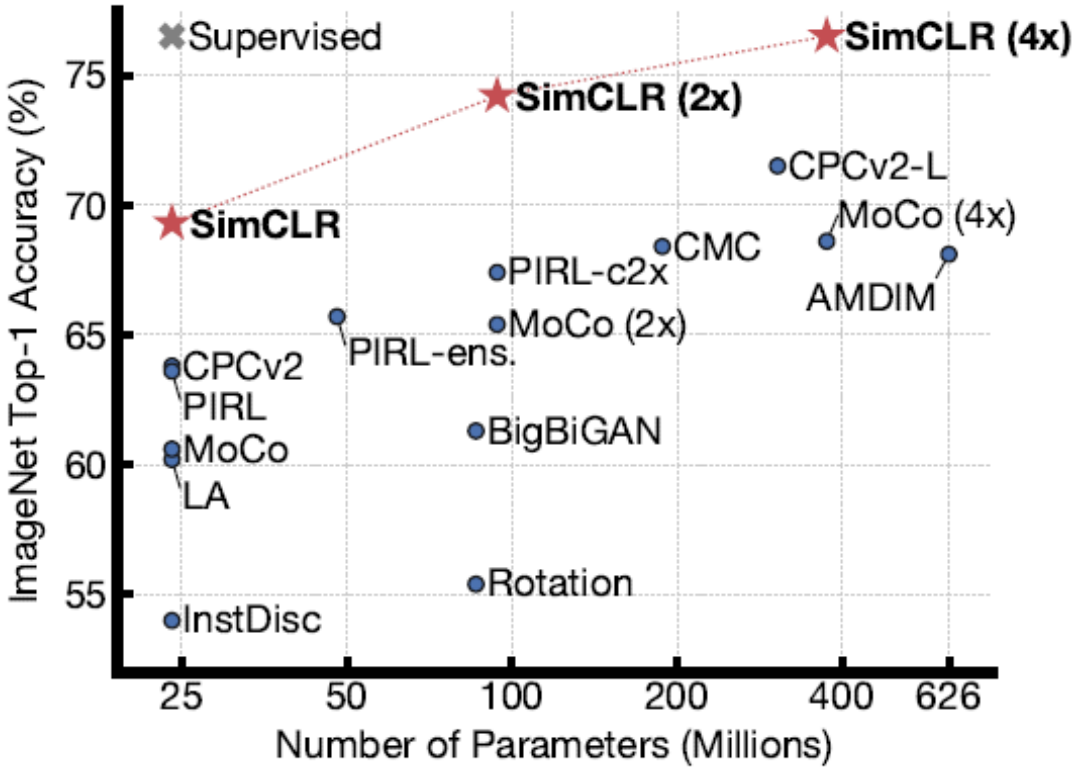
- 3. Momentum Contrastive (MOCO) (Kaiming He, 2019)

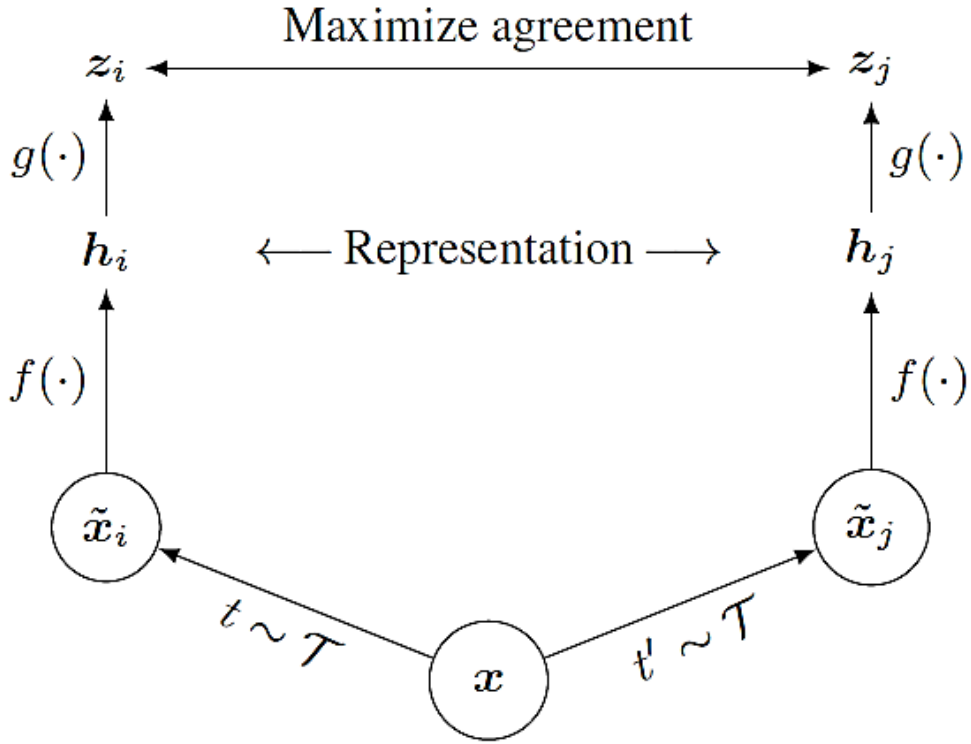



Important Milestone of Contrastive Learning:
Competitive or Better Performance on
7 Downstream Tasks

# Contrastive Learning (Unsupervised Learning)

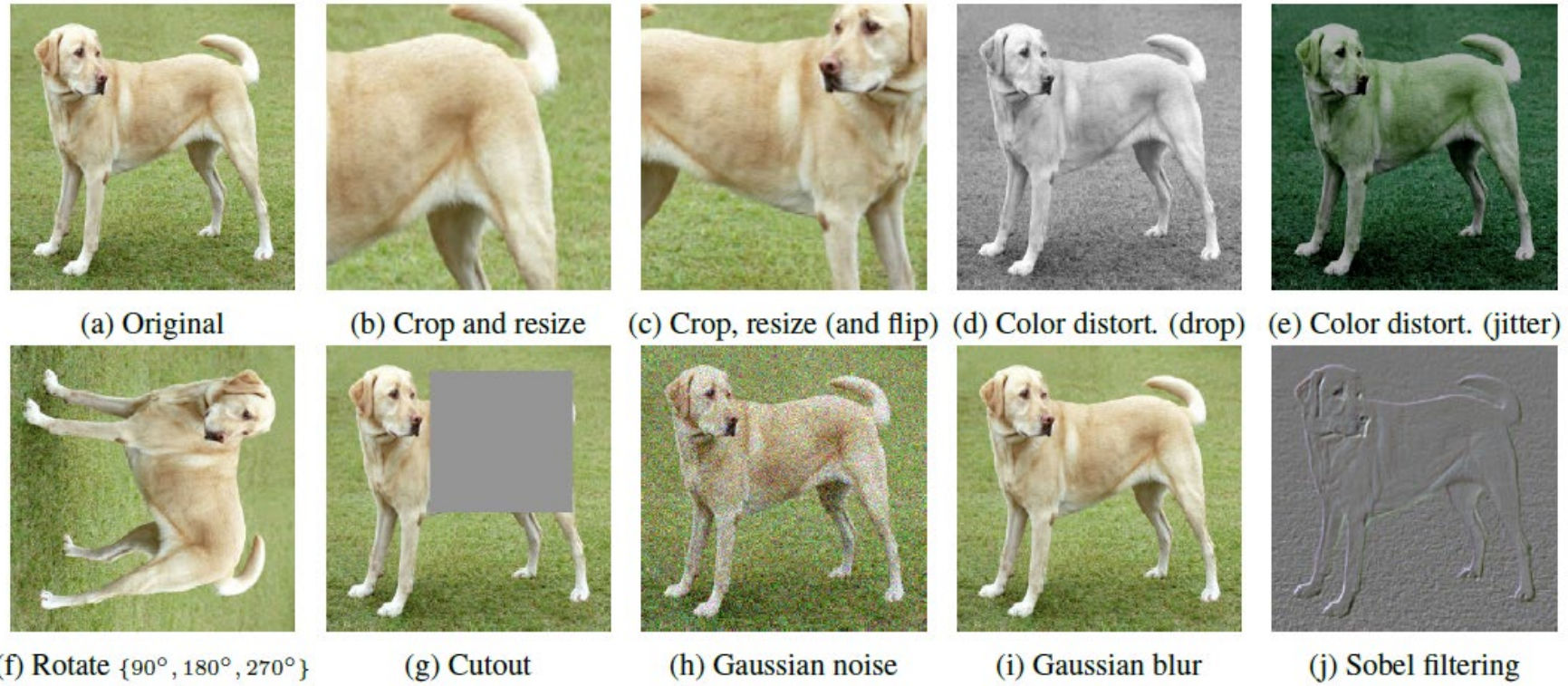
- 4. SimCLR (Chen, ICML20)
  - (1) Memory bank is unnecessary. Negative sample can come from Large Batch.
  - (2) Data Augmentation is extremely important.



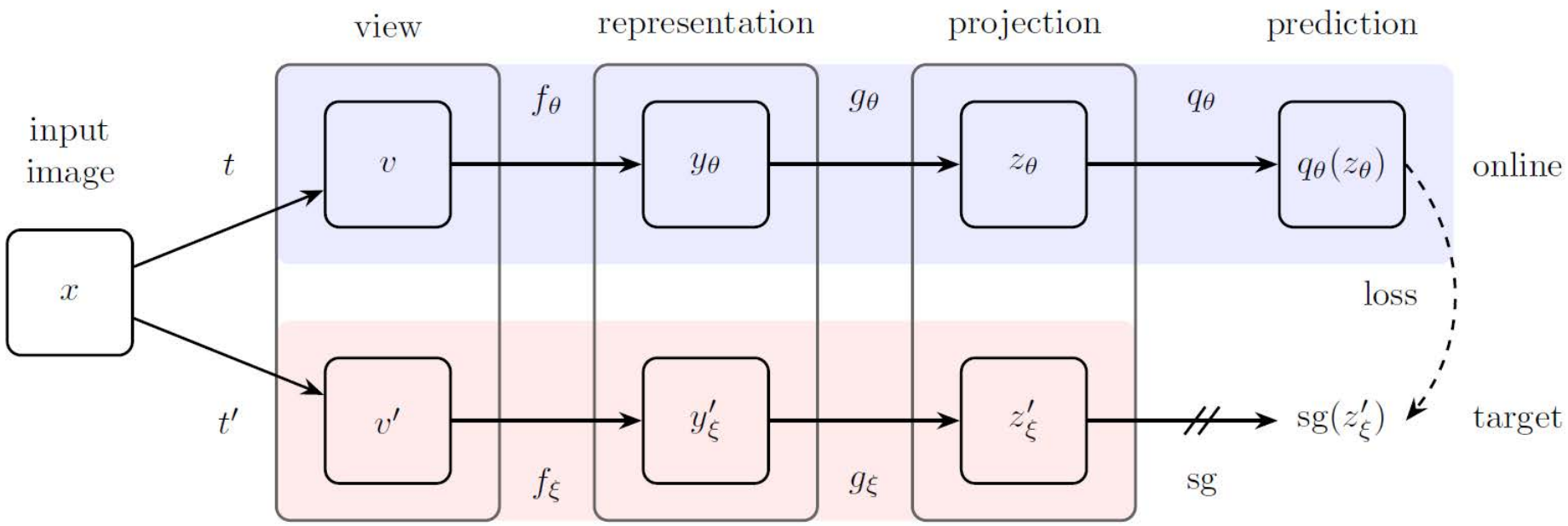
# Contrastive Learning (Unsupervised Learning)

- 4. SimCLR (Chen, ICML20)
  - (1) Memory bank is unnecessary. Negative sample can come from Large Batch.
  - (2) Data Augmentation is extremely important.



# Contrastive Learning (Unsupervised Learning)

- 5. BYOL (Grill, NeurIPS20)
  - Self-distillation without negative pairs

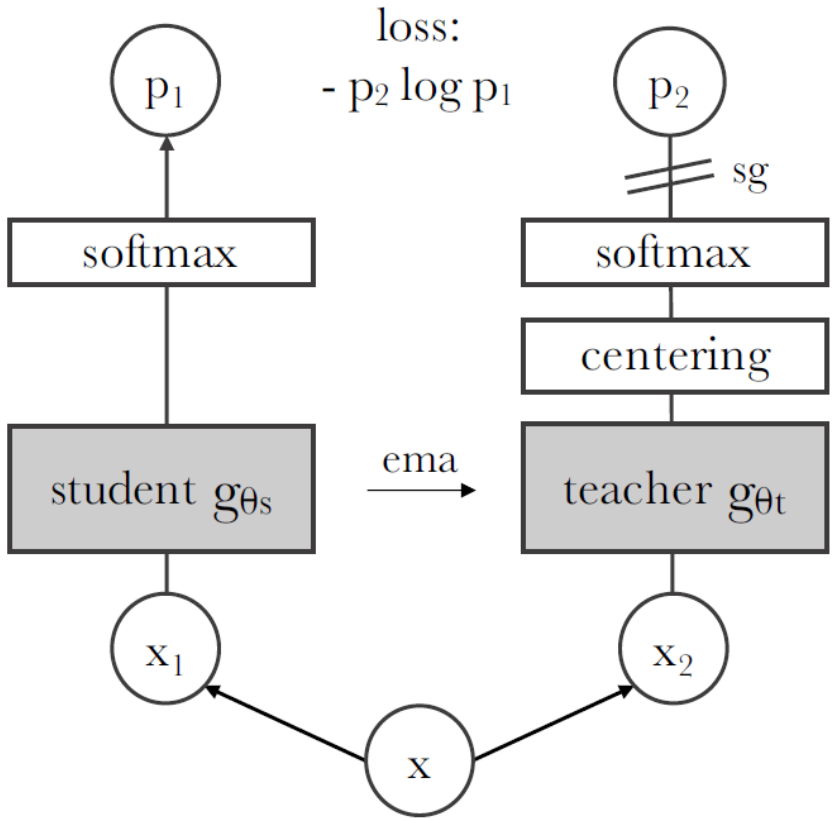



J.-B. Grill et al., "Bootstrap your own latent: A new approach to self-supervised earning," *NeurIPS* , 2020

# Contrastive Learning (Unsupervised Learning)

- 6. DINO (Caron, ICCV21)

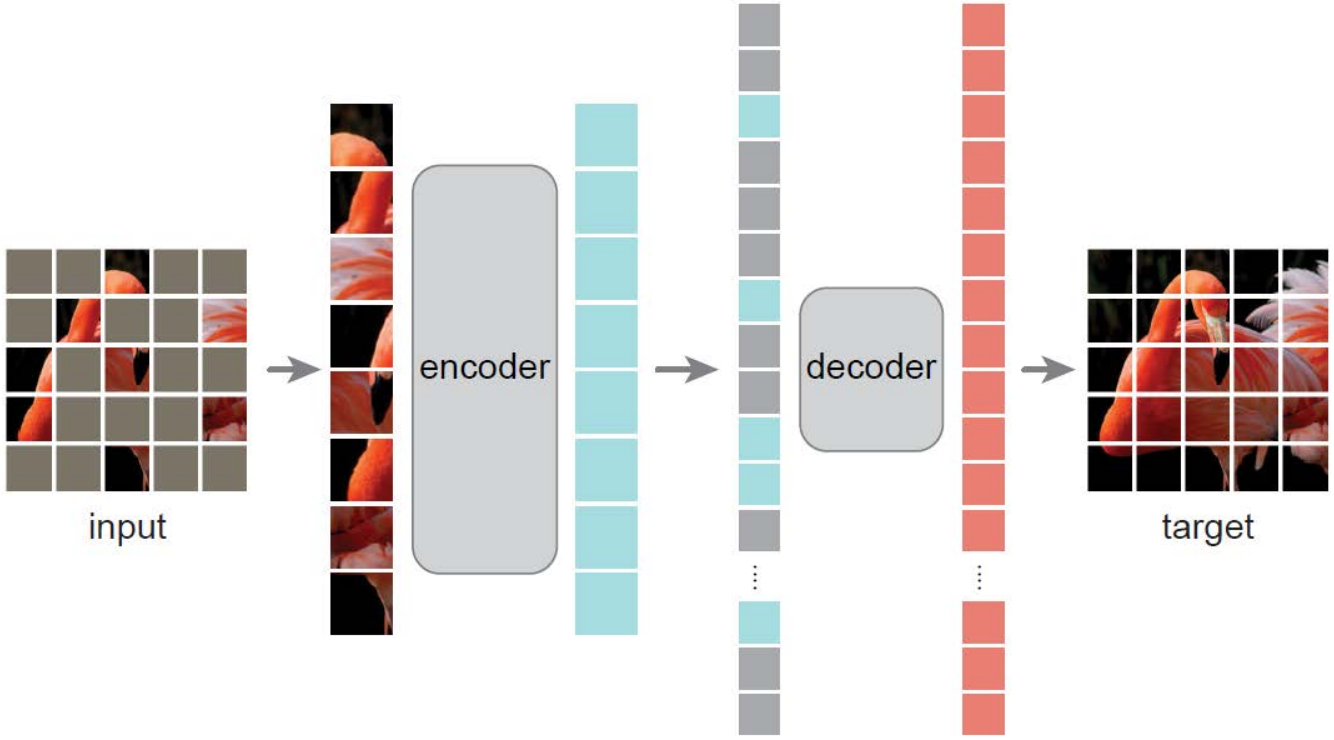
$$P_s(x)^{(i)} = \frac{\exp(g_{\theta_s}(x)^{(i)}/\tau_s)}{\sum_{k=1}^{K}\exp(g_{\theta_s}(x)^{(k)}/\tau_s)}$$




M. Caron et al., "Emerging properties in self-supervised vision transformers," ICCV , 2021

# Generative Learning (Unsupervised Learning)

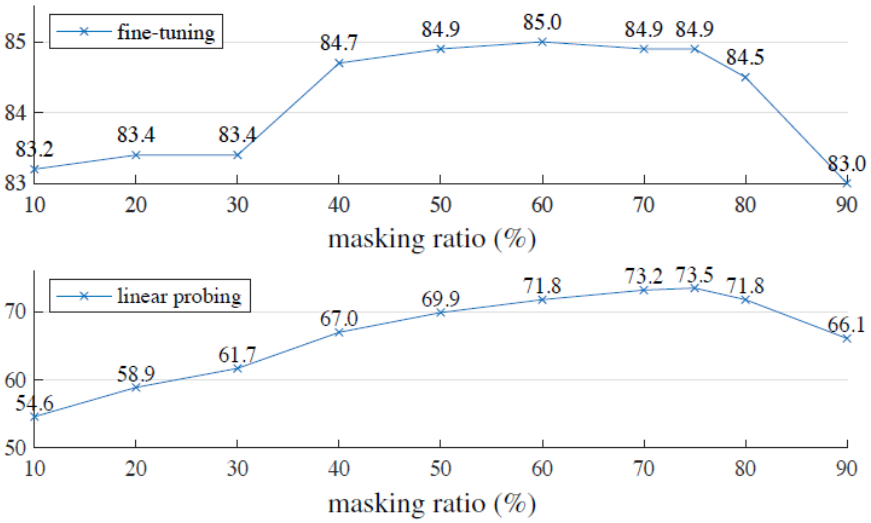
- 1. MAE (Kaiming He, 2021)

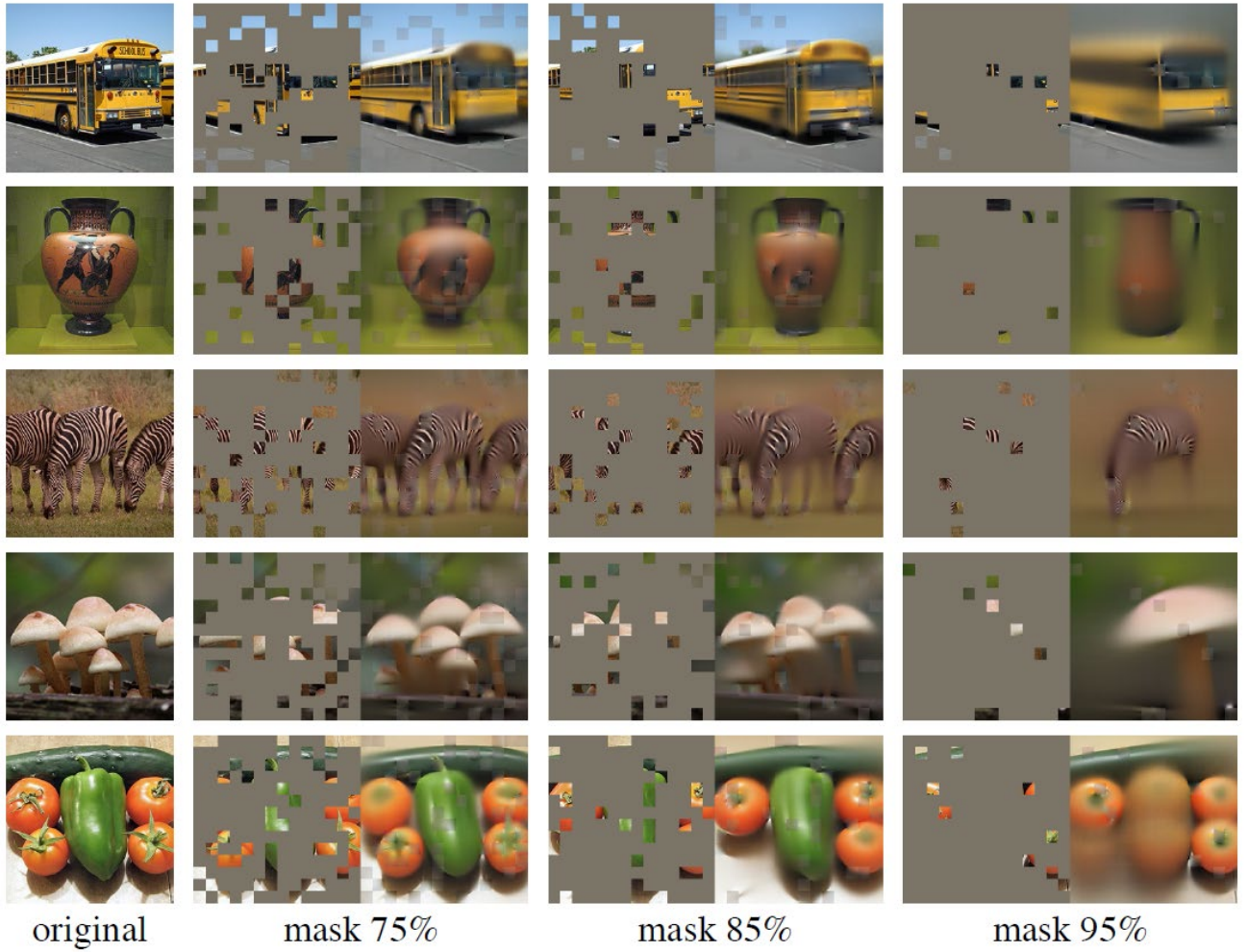



MAE:
(1) mask random patches of the input image and reconstruct the missing pixels
(2) asymmetric encoder-decoder architecture
(3) masking a high proportion of the input image, e.g., 75%, yields a nontrivial and meaningful self-supervisory task

# Generative Learning (Unsupervised Learning)

- 1. MAE (Kaiming He, 2021)



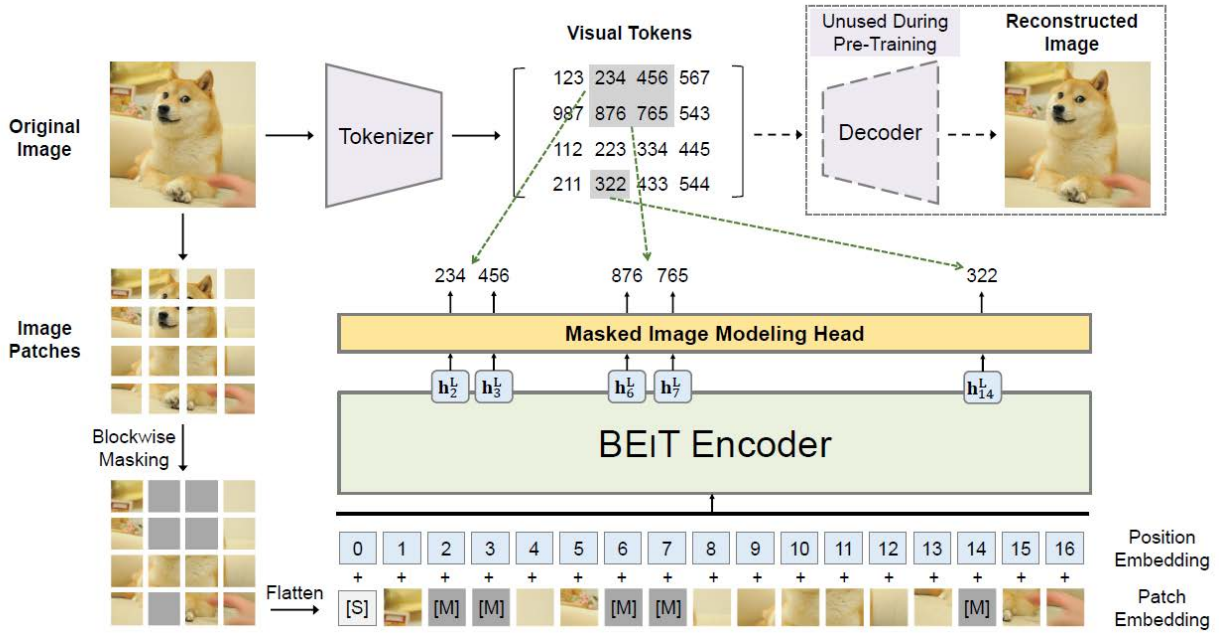



# Generative Learning (Unsupervised Learning)

- 2. BEiT (Bao, ICLR 2022)
  - Bert pre-training of image transformers

**Algorithm 1** Blockwise Masking

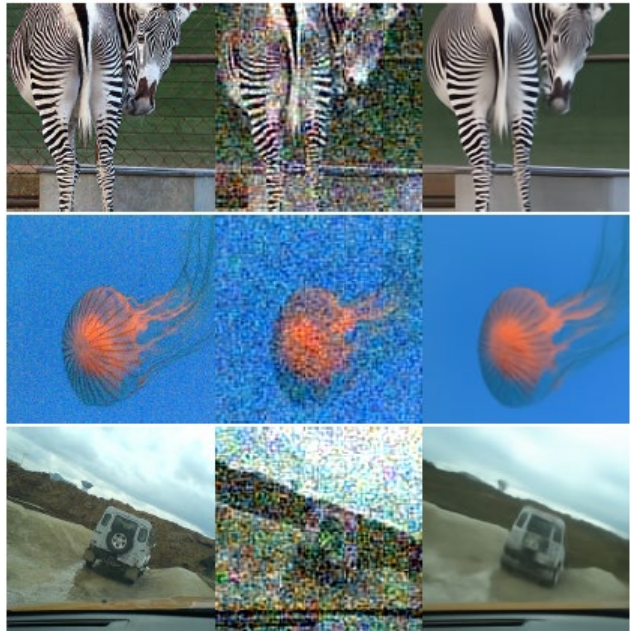
**Input:** $N(=h \times w)$ image patches
**Output:** Masked positions $\mathcal{M}$
$\mathcal{M} \leftarrow \{\}$
**repeat**
  $s \leftarrow \text{Rand}(16, 0.4N - |\mathcal{M}|)$ ▷ *Block size*
  $r \leftarrow \text{Rand}(0.3, \frac{1}{0.3})$ ▷ *Aspect ratio of block*
  $a \leftarrow \sqrt{s \cdot r}; b \leftarrow \sqrt{s/r}$
  $t \leftarrow \text{Rand}(0, h-a)\ ; l \leftarrow \text{Rand}(0, w-b)$
  $\mathcal{M} \leftarrow \mathcal{M} \bigcup \{(i,j) : i \in [t, t+a), j \in [l, l+b)\}$
**until** $|\mathcal{M}| > 0.4N$ ▷ *Masking ratio is 40%*
**return** $\mathcal{M}$

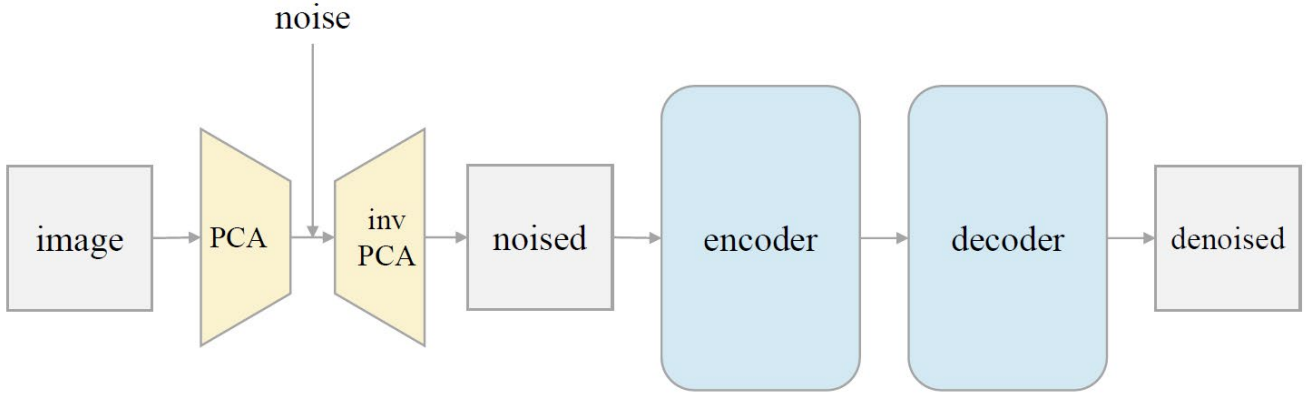



H. Bao, L. Dong, S. Piao, and F. Wei, "BEiT: Bert pre-training of image transformers," ICLR, 2022,

# Generative Learning (Unsupervised Learning)

- 3. DAE (Chen, 2024)
  - Denoising Autoencoder

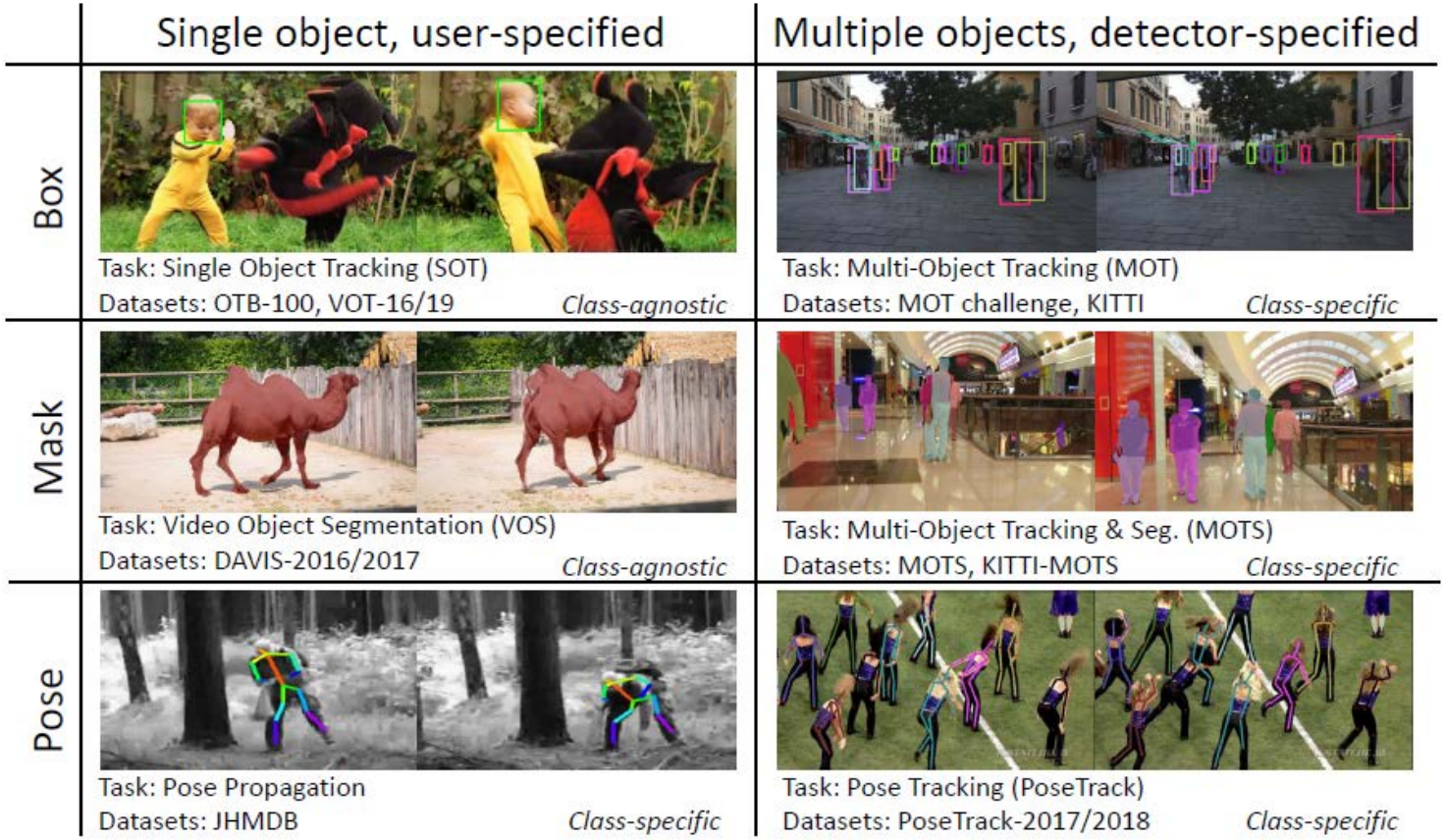





clean image noised denoised

*Chen X, et al. Deconstructing denoising diffusion models for self-supervised learning[J]., 2024.*

# UniTrack



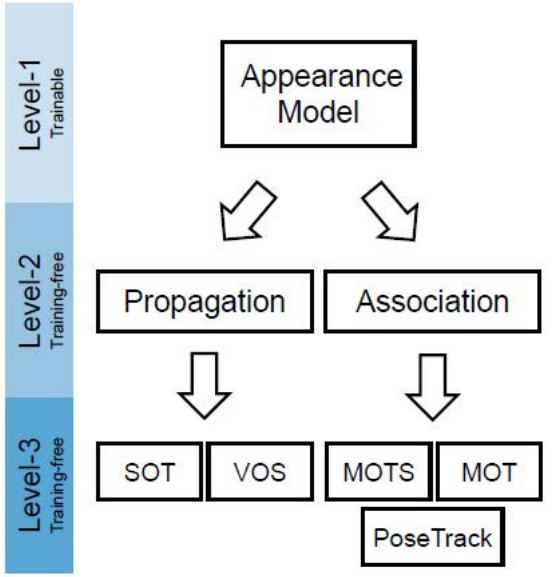




**Figure 3:** Overview of UniTrack. The framework can be divided in three levels. Level-1: a trainable appearance model. Level-2: the fundamental primitives of *propagation* and *association*. Level-3: task-specific heads.

Wang, Zhongdao et al. Do Different Tracking Tasks Require Different Appearance Models?. NeurIPS 2021.

# Part 5: Recent New Trends

# UniDetector

- Towards Universal Object Detection
  - a universal detection framework to utilize images of heterogeneous label spaces and generalize to the open world

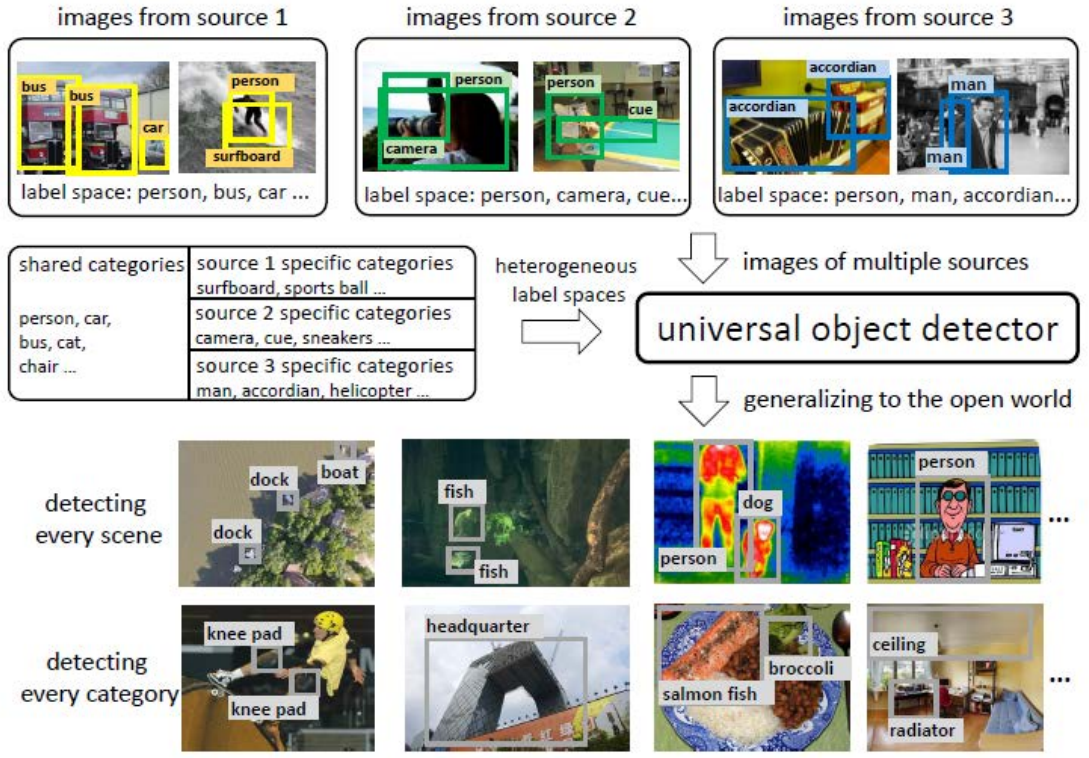
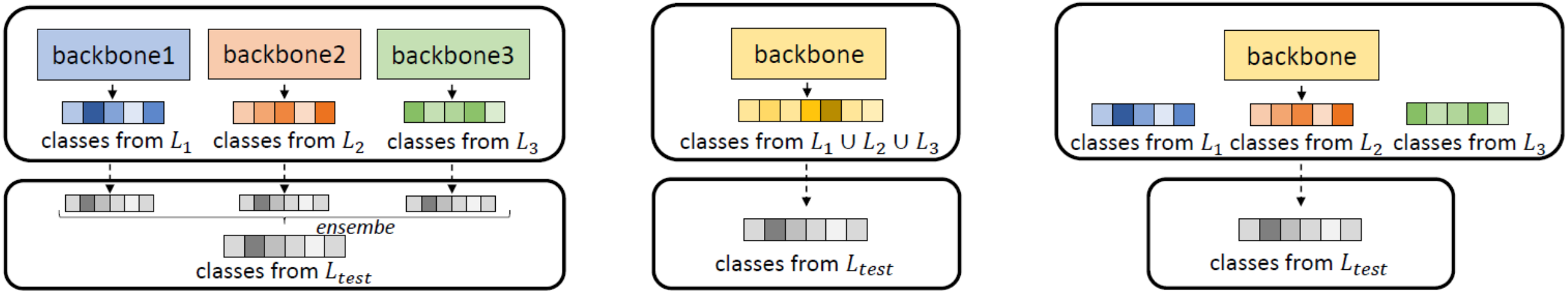



Figure 1. **Illustration for the universal object detector.** It aims to detect every category in every scene and should have the ability to utilize images of multiple sources with heterogeneous label spaces for training and generalize to the open world for inference.

*Zhenyu Wang et al, Detecting Everything in the Open World: Towards Universal Object Detection. CVPR 2023.*

# UniDetector

- Towards Universal Object Detection




(a) seperate label spaces (b) unified label space (c) partitioned label space

Figure 3. **Possible structures to utilize images from heterogeneous label spaces for training.** The above boxes denote the structure for training, and the below boxes denote the inference process. All the classification head here adopts the similarity between the region features and language embeddings. The separate structure trains individual networks and ensembles them for inference, the unified structure unifies the multiple datasets into one dataset, and the partitioned structure shares the same backbone but different classification heads.

*Zhenyu Wang et al, Detecting Everything in the Open World: Towards Universal Object Detection. CVPR 2023.*

# UniDetector

- Towards Universal Object Detection

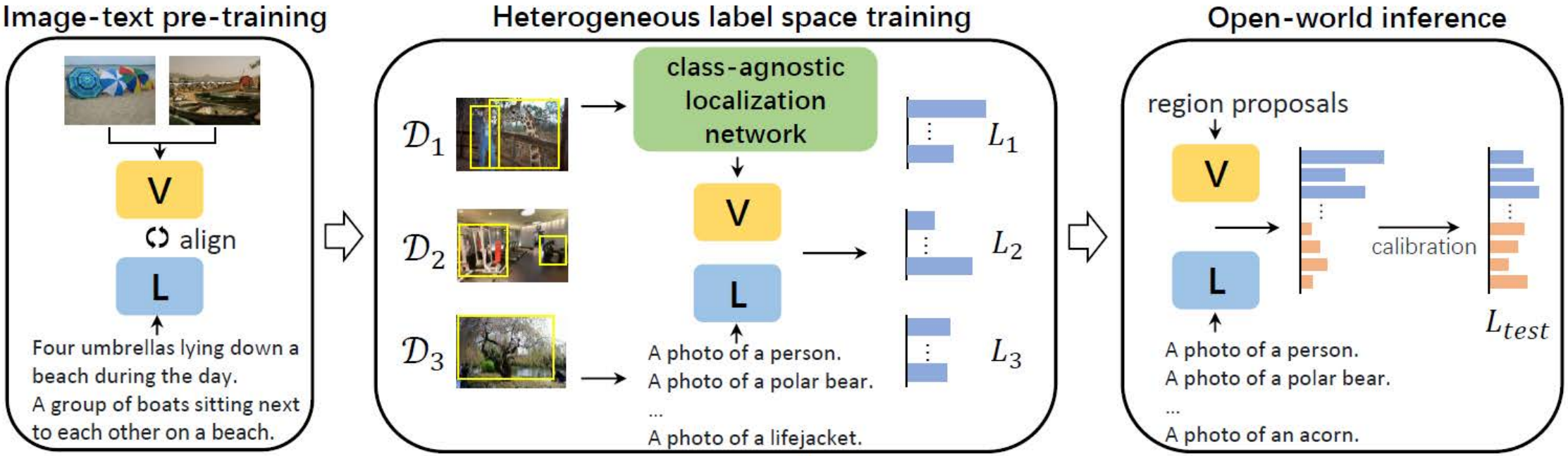



Figure 2. **Overview of UniDetector.** It consists of three steps. With the image-text pre-training parameters, UniDetector is trained with images of different sources with multiple label spaces. In this way, it can directly detect in the open world for inference. 'V' denotes the module to process visual information, and 'L' denotes the language encoder. The first stage conducts image-text pre-training to align the two spaces, the second stage trains with images of heterogeneous label spaces in the decoupling manner, and the third stage applies probability calibration to maintain the balance.

*Zhenyu Wang et al, Detecting Everything in the Open World: Towards Universal Object Detection. CVPR 2023.*

# UniDetector

- Towards Universal Object Detection

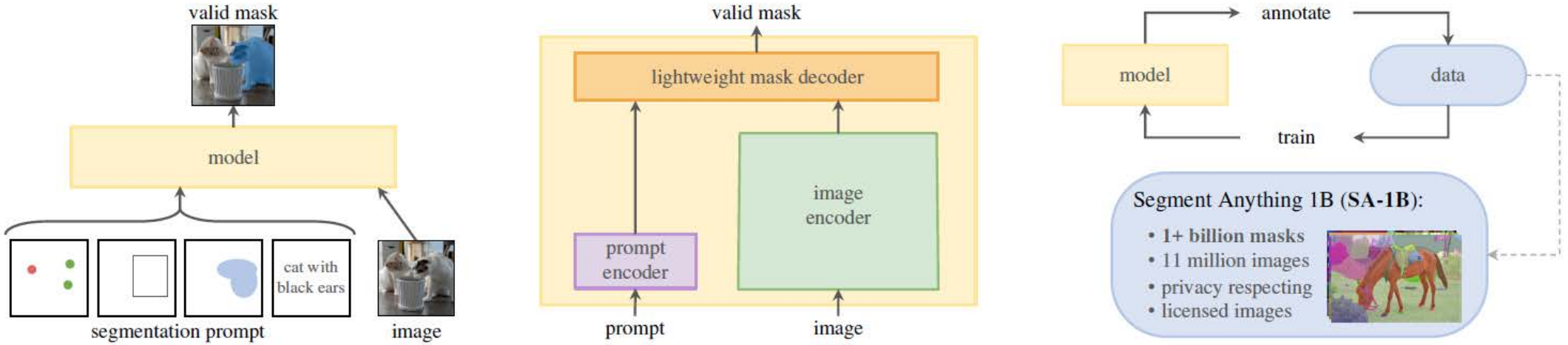
Table 1. **The performance of UniDetector in the open world.** We evaluate it on LVIS, ImageNetBoxes and VisualGenome. S, U, P denote treating heterogeneous label spaces as separate spaces, a unified one or a partitioned one. The Faster RCNN (closed world) row is from the traditional supervised Faster RCNN C4 baseline trained on the corresponding dataset with the same random data sampler. We select 35k, 60k, 78k images from COCO, Objects365 and OpenImages respectively for training.

| Training data | Structure | LVIS v0.5 (1,230) | | | | LVIS v1 (1,203) | | | | ImageNetBoxes (3,622) | | | VisualGenome (7,605) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AP | $AP_r$ | $AP_c$ | $AP_f$ | AP | $AP_r$ | $AP_c$ | $AP_f$ | AP | $AP_{50}$ | Loc. Acc | $AR_1$ | $AR_{10}$ | $AR_{100}$ |
| Faster RCNN (closed world) | | 17.7 | 1.9 | 16.5 | 25.4 | 16.2 | 0.9 | 13.1 | 26.4 | 3.9 | 6.1 | 15.3 | 3.5 | 4.3 | 4.3 |
| COCO | - | 16.4 | 18.7 | 17.1 | 14.5 | 13.7 | 13.5 | 13.6 | 13.9 | 4.8 | 6.8 | 8.3 | 4.3 | 5.9 | 5.9 |
| O365 | - | 20.2 | 21.3 | 20.2 | 19.8 | 16.8 | 14.7 | 16.2 | 18.3 | 3.8 | 5.5 | 8.4 | 5.4 | 7.3 | 7.3 |
| OImg | - | 16.8 | 21.8 | 17.6 | 13.8 | 13.9 | 14.7 | 14.2 | 13.2 | 7.9 | 10.8 | 16.0 | 5.9 | 8.1 | 8.2 |
| COCO + O365 | S | 21.0 | 22.2 | 21.8 | 19.4 | 17.5 | 16.0 | 17.2 | 18.4 | 4.5 | 6.5 | 8.9 | 6.2 | 8.5 | 8.6 |
| COCO + O365 | U | 20.9 | 19.6 | 21.0 | 21.3 | 17.6 | 14.6 | 17.0 | 19.6 | 3.6 | 5.1 | 8.0 | 5.3 | 7.1 | 7.2 |
| COCO + O365 (+mosaic) | U | 21.4 | 22.3 | 21.5 | 21.0 | 16.8 | 13.5 | 16.2 | 18.9 | 3.6 | 5.1 | 7.7 | 5.0 | 6.8 | 6.9 |
| COCO + O365 (+pseudo [62]) | U | 20.8 | 22.5 | 22.7 | 19.7 | 16.6 | 13.4 | 16.1 | 18.7 | 3.6 | 5.1 | 7.6 | 5.0 | 6.6 | 6.7 |
| COCO + O365 | P | 22.2 | 23.7 | 22.5 | 21.2 | 18.2 | 15.5 | 17.6 | 20.1 | 4.7 | 6.6 | 10.1 | 5.9 | 8.0 | 8.1 |
| COCO + OImg | P | 19.9 | 22.1 | 20.7 | 17.9 | 16.8 | 16.0 | 16.8 | 17.1 | 6.9 | 9.5 | 14.7 | 5.7 | 7.7 | 7.8 |
| COCO + O365 + OImg | P | **23.5** | **23.6** | **24.3** | **22.4** | **19.8** | **18.0** | **19.2** | **21.2** | **8.2** | **11.4** | **16.9** | **6.5** | **8.7** | **8.8** |

*Zhenyu Wang et al, Detecting Everything in the Open World: Towards Universal Object Detection. CVPR 2023.*

# SAM: Segment anything

- ✓ Segment anything project, a new AI model from Meta AI that can "cut out" any object, in any image, with a single click (MetaAI, Apr 5, 2023)



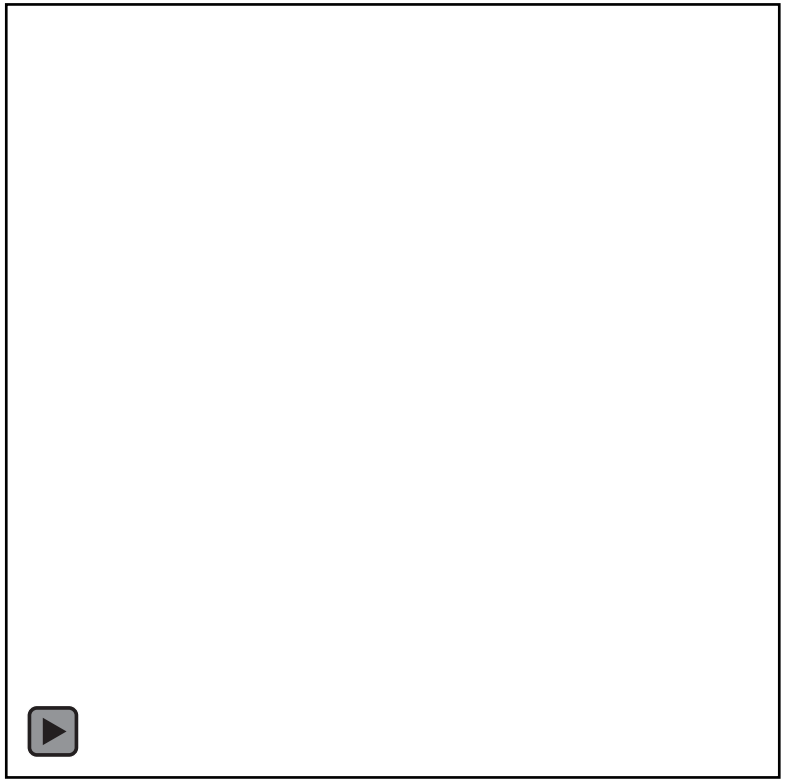

(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

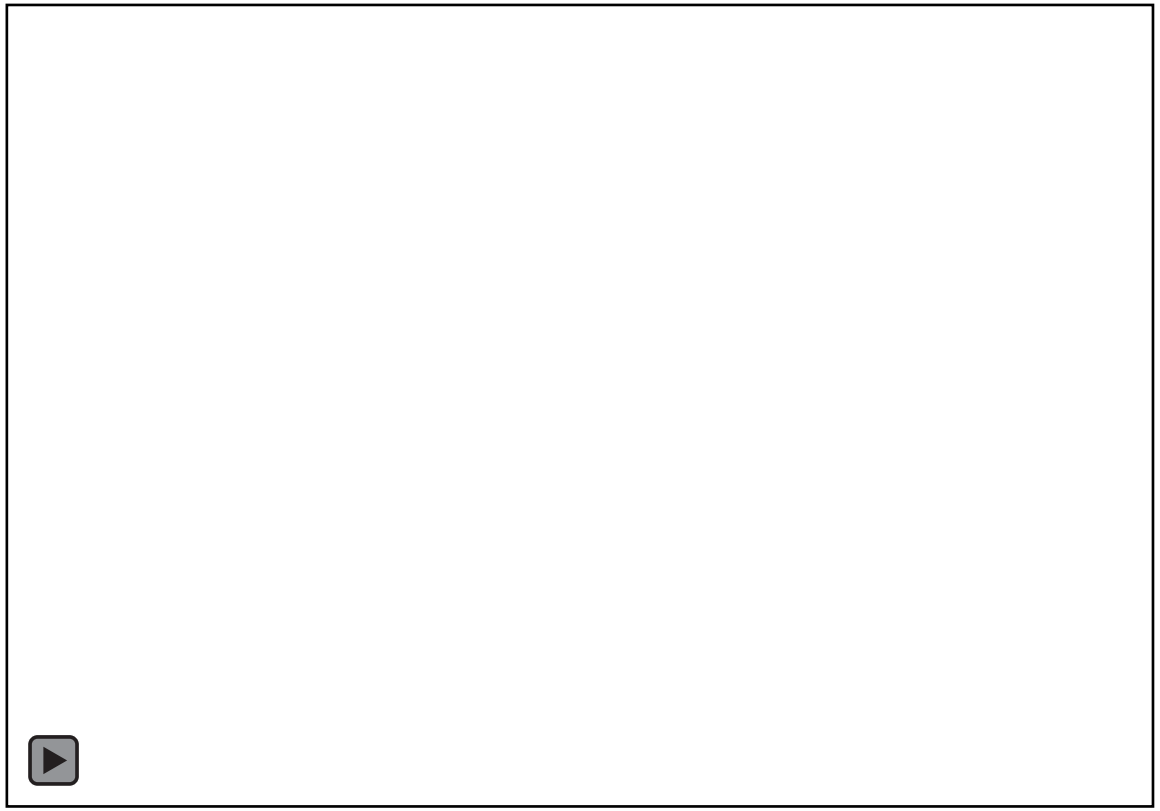
Figure 1: We aim to build a foundation model for segmentation by introducing three interconnected components: a prompt-able segmentation *task*, a segmentation *model* (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a *data* engine for collecting SA-1B, our dataset of over 1 billion masks.

*https://segment-anything.com/*

*Alexander Kirillov et al. Segment Anything. ICCV 2023.*

# SAM: Segment anything





*https://segment-anything.com/*

*Alexander Kirillov et al. Segment Anything. ICCV 2023.*

# SAM: Segment anything

- Dataset
  - Model-in-the-loop Data engine: using SAM and its data to interactively annotate images and update the model. (repeated many times).
  - SA-1B: 1.1B mask dataset.
    - 11M images, 1B+ masks

*Alexander Kirillov et al. Segment Anything. ICCV 2023.*

# SAM: Segment anything

Image encoder. MAE pre-trained Vision Transformer (ViT)

Mask encoder. A modification of a Transformer decoder block followed by a dynamic mask prediction head.

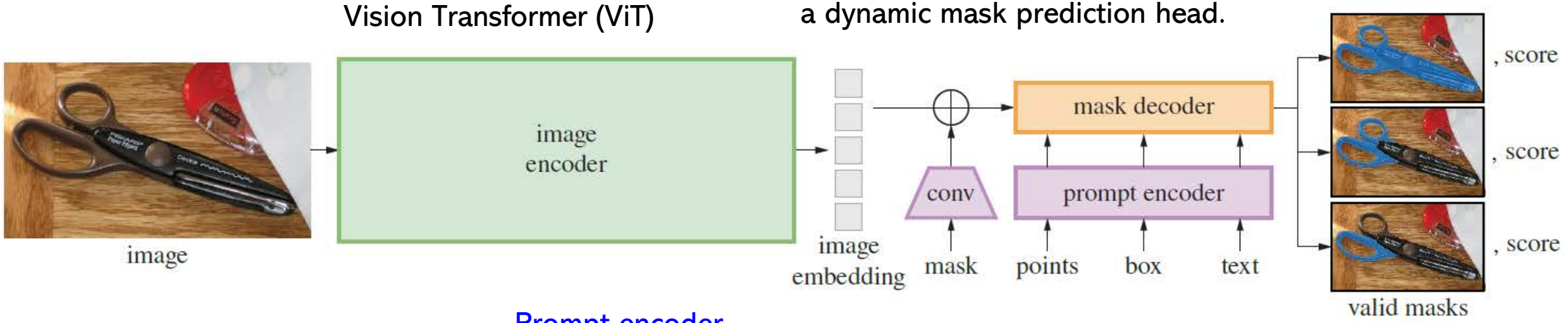



Figure 4: Segment Anything Model (SA … embedding that can then be efficiently queried by a variety of … ed. For ambiguous prompts corresponding to more than one … nfidence scores.

Prompt encoder.

Sparse (points, boxes, text):

- points and boxes by positional encodings summed with learned embeddings for each prompt type
- text with an off-the-shelf text encoder from CLIP

Dense (masks):

- embedded using convolutions and summed element-wise with the image embedding.

*https://segment-anything.com/*

*Alexander Kirillov et al. Segment Anything. ICCV 2023.*

# SAM: Segment anything

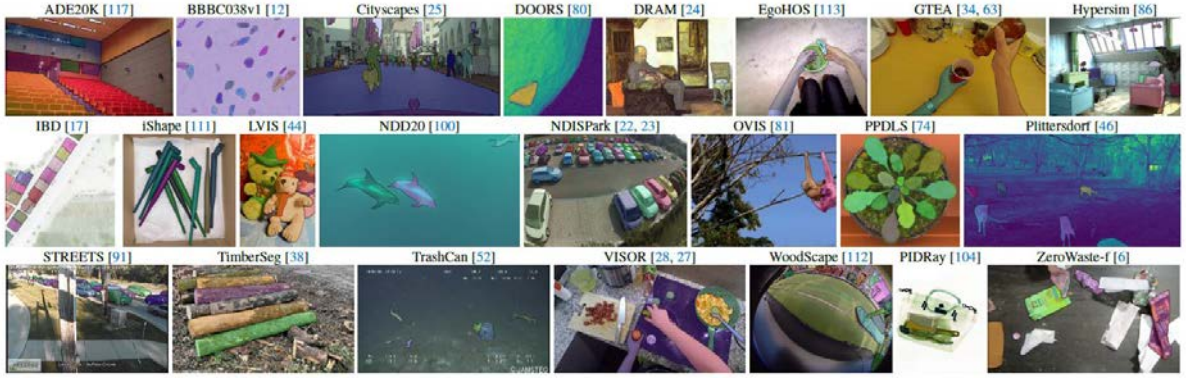
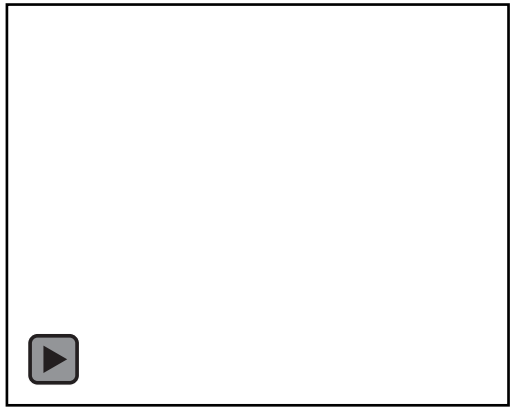
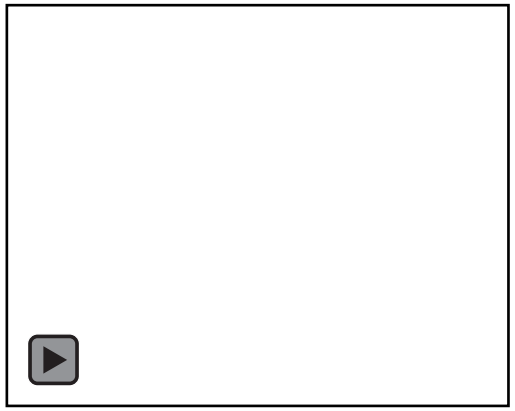
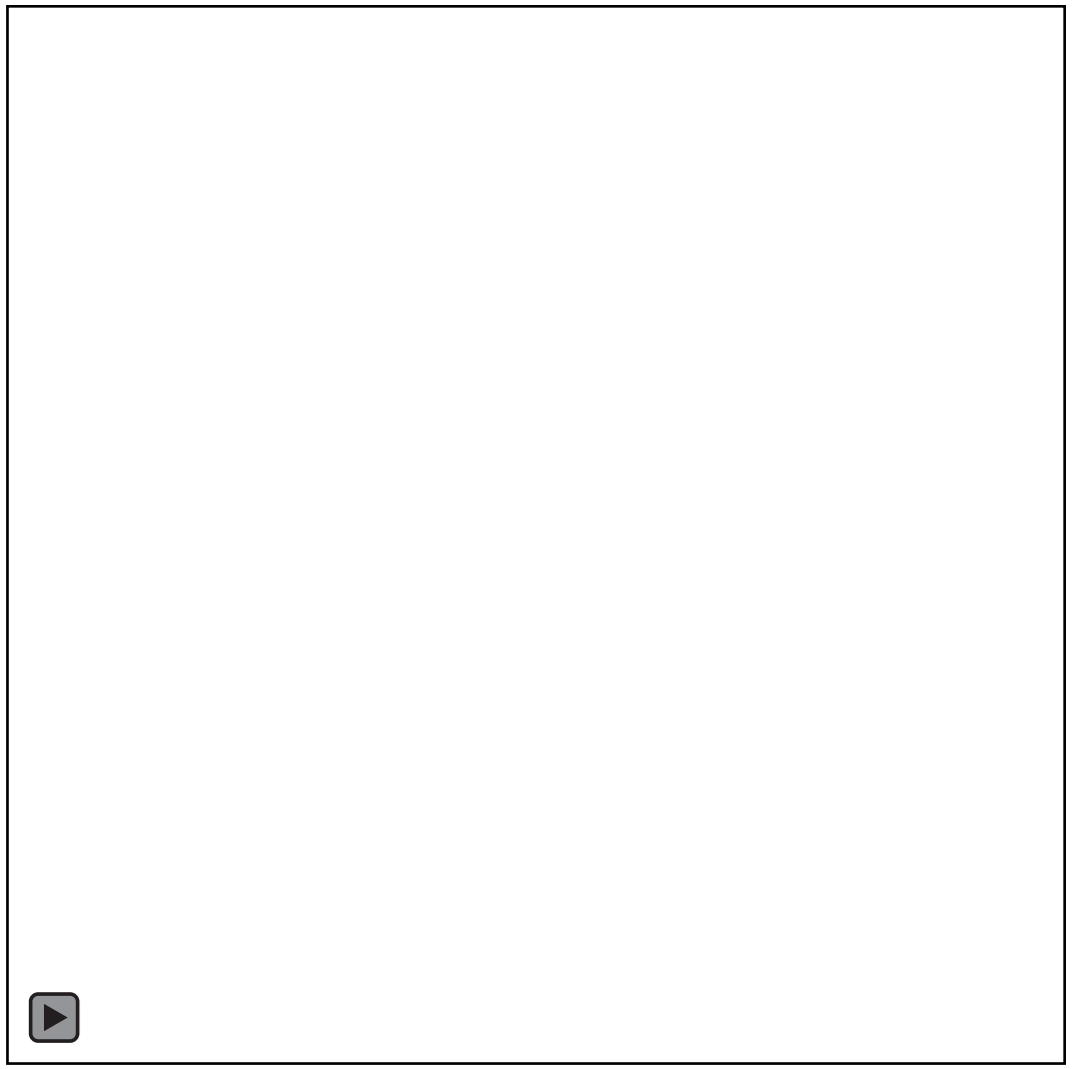



Figure 8: Samples from the 23 diverse segmentation datasets used to evaluate SAM's zero-shot transfer capabilities.





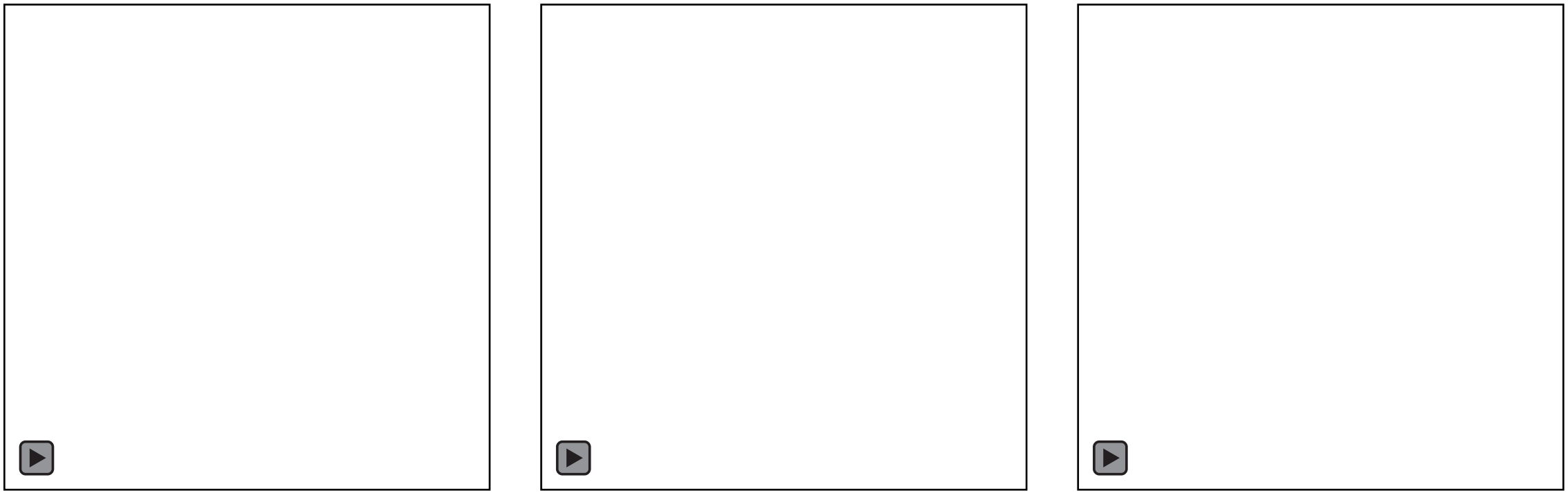


*https://segment-anything.com/*

*Alexander Kirillov et al. Segment Anything. ICCV 2023.*

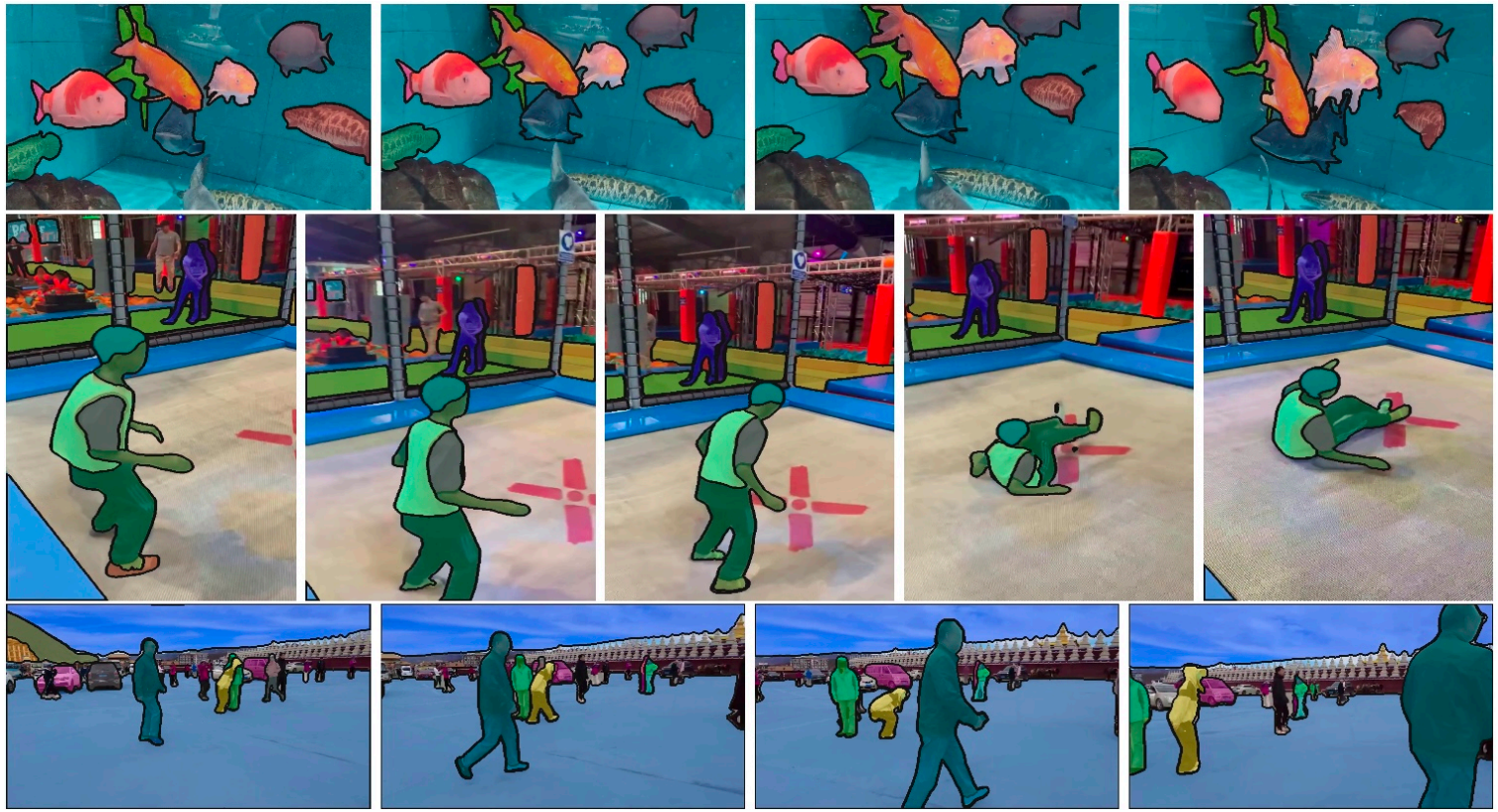
# SAM 2 - Segment Anything Model 2

A unified model for segmenting objects across **images** and **videos**. You can use a click, box, or mask as the input.



# SAM 2 - Segment Anything Model 2

- Largest Video Segmentation Dataset **SA-V**



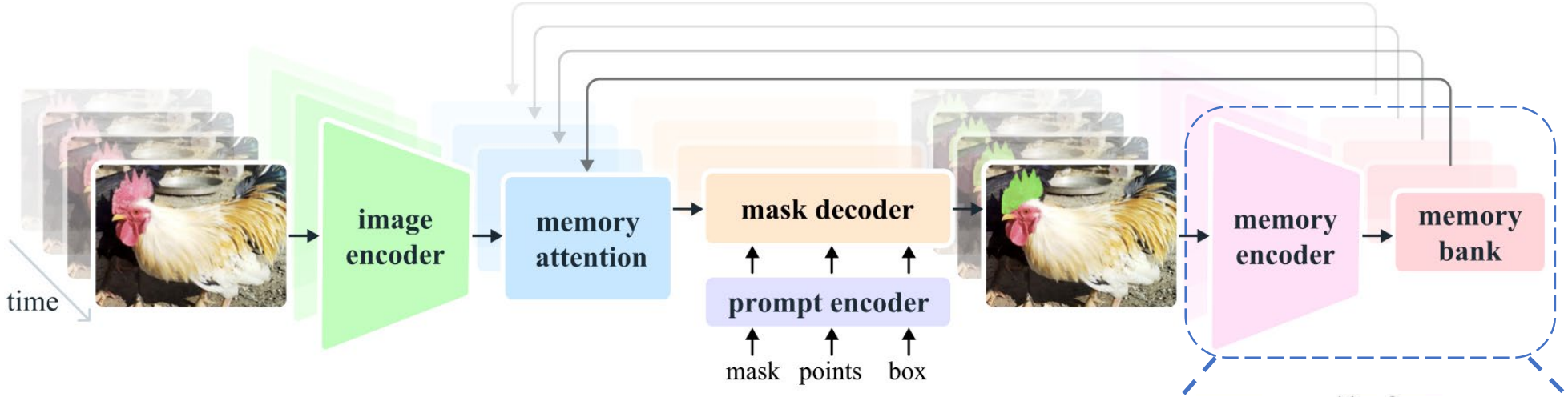
**Large Data**

51K diverse videos

643K masks

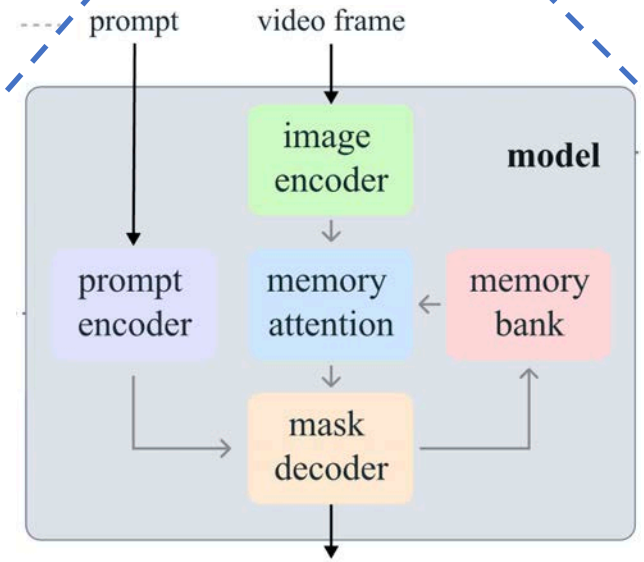
cover diverse subjects

# SAM 2 - Segment Anything Model 2



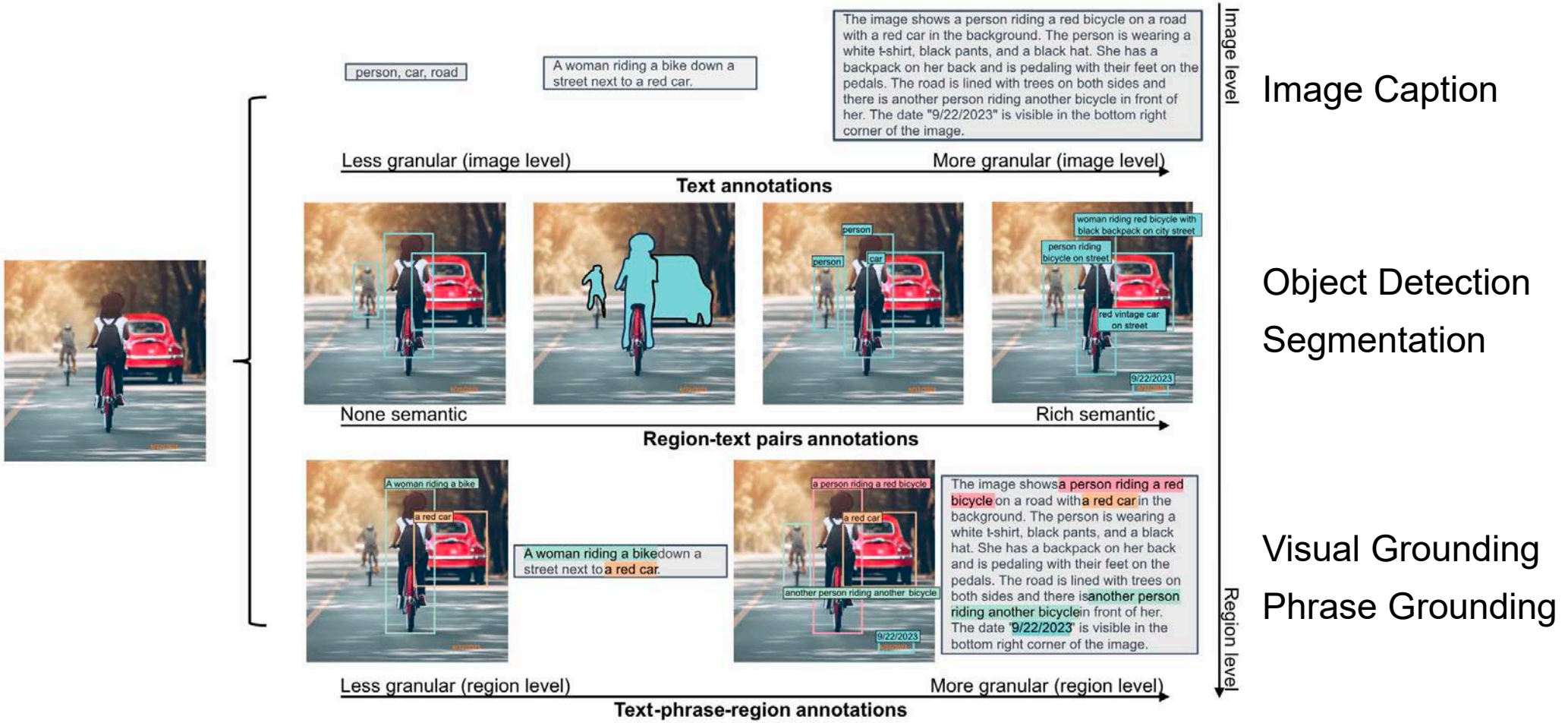

- **Memory attention:**

Conditioning current frame features on past frame features and predictions through **cross-attention**.



# Florence-2: Advancing a Unified Representation for a Variety of Vision Tasks

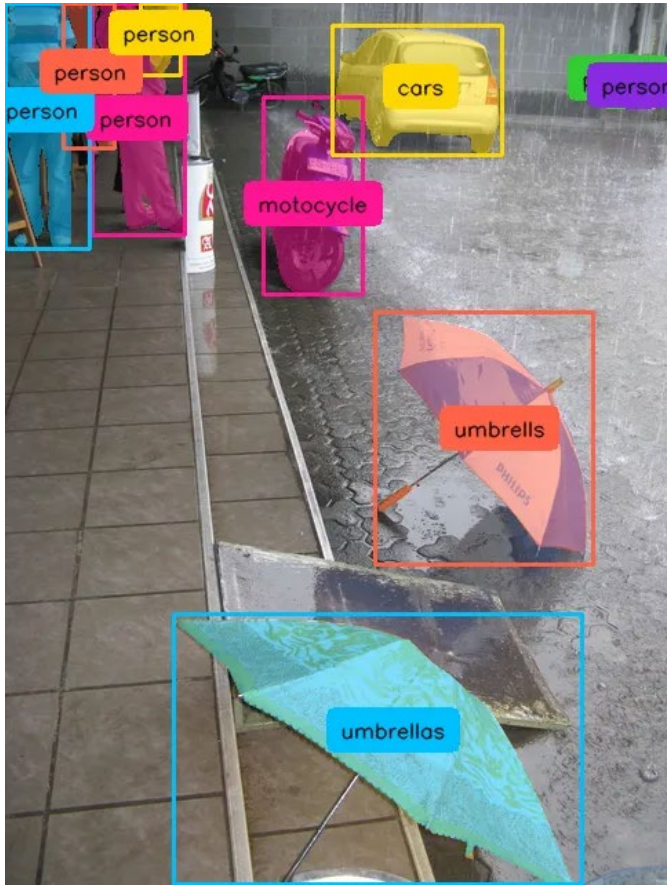
- Built on **large-scale data** from both **visual and textual sources** to understand complex relationships.



# Florence-2: Advancing a Unified Representation for a Variety of Vision Tasks

Input Image

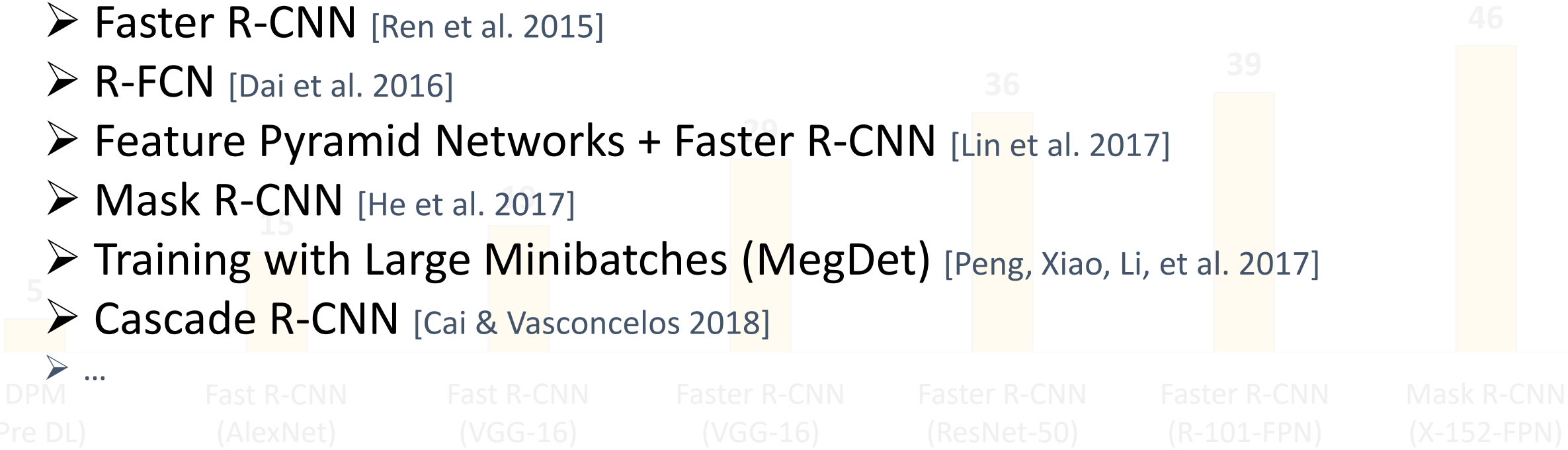
Object Detection + Segmentation




Phrase Grounding

# Steady Progress on Boxes and Masks

- R-CNN [Girshick et al. 2014]
- SPP-net [He et al. 2014]
- Fast R-CNN [Girshick. 2015]
- Faster R-CNN [Ren et al. 2015]
- R-FCN [Dai et al. 2016]
- Feature Pyramid Networks + Faster R-CNN [Lin et al. 2017]
- Mask R-CNN [He et al. 2017]
- Training with Large Minibatches (MegDet) [Peng, Xiao, Li, et al. 2017]
- Cascade R-CNN [Cai & Vasconcelos 2018]
- …



# Thanks!

Q&A

# Key References


- Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press. Chapter 9. http://www.deeplearningbook.org/contents/convnets.html
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. Imagenet classification with deep convolutional neural networks. NIPS 2012.
- Simonyan, K., & Zisserman, A. Very deep convolutional networks for large-scale image recognition. ICLR 2015.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. Going deeper with convolutions. CVPR 2015.
- He, K., Zhang, X., Ren, S., & Sun, J. Deep residual learning for image recognition. CVPR 2016.
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. Densely connected convolutional networks. CVPR 2017.
- Ren, S., He, K., Girshick, R., & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. NIPS 2015.
- He, K., Gkioxari, G., Dollár, P., & Girshick, R. Mask r-cnn. ICCV 2017.

# End of the 6th Week