GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Deep Learning

Lecture 8: Transfer learning & perceptual losses

Alexander Ecker
Institut für Informatik, Uni Göttingen

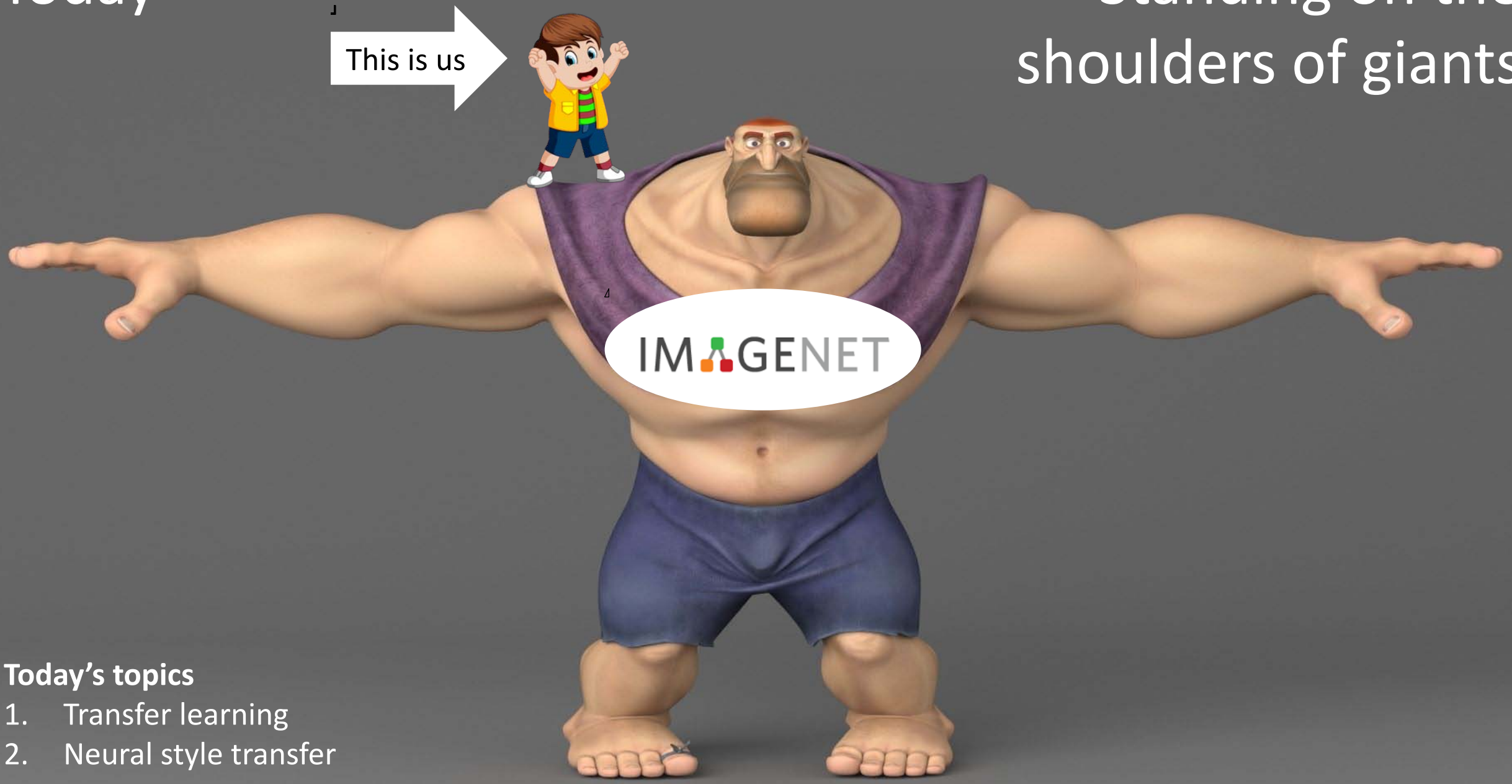
alxecker

https://alexanderecker.wordpress.com

– Credit: some of the slides based on Fei Fei Li, Justin Johnson and Serena Yeung's slides –

Today

Standing on the shoulders of giants



Today's topics

1. Transfer learning
2. Neural style transfer

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

Transfer learning

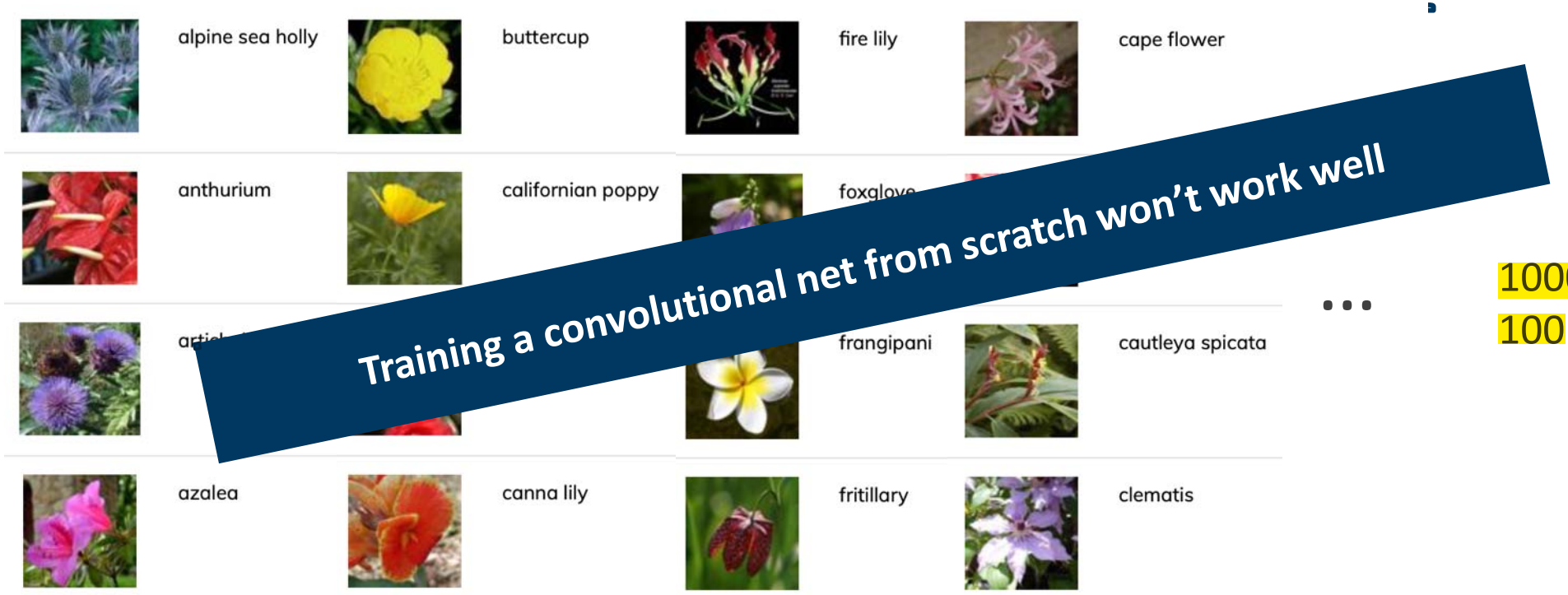
the accurancy will much lower than 99% or anything else

even if 1000 images is small

Transfer learning: motivation

Illustrative example

Suppose you want to build an app to recognize flowers…



1000 images
100 classes

Image source: https://www.robots.ox.ac.uk/~vgg/data/flowers/102/

Transfer learning: the idea

to solve the problem:
- train the classifier on much more images or reload it
- afterwards fine tune it
-> should converge

1

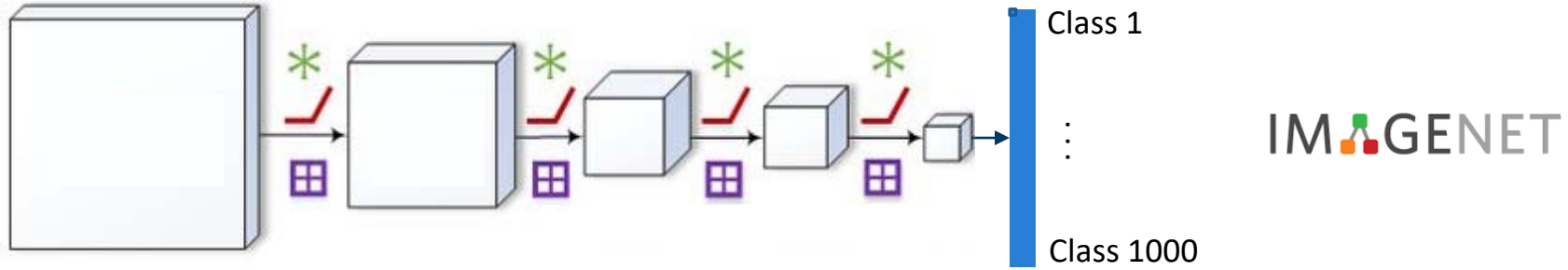
Pre-training

(massive data)

IMAGENET

1.2 million images

1000 object classes



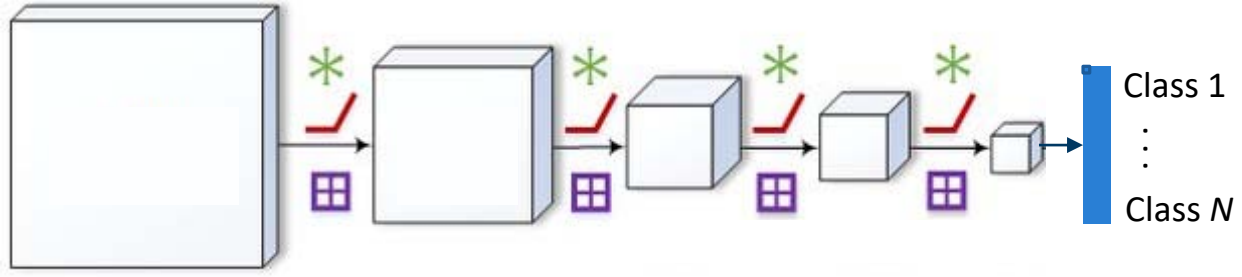
Transfer weights

2

Fine-tuning

(much less data)

Add some extra layers, to classify other things as well



Recognize flowers

Transfer learning: the idea

1

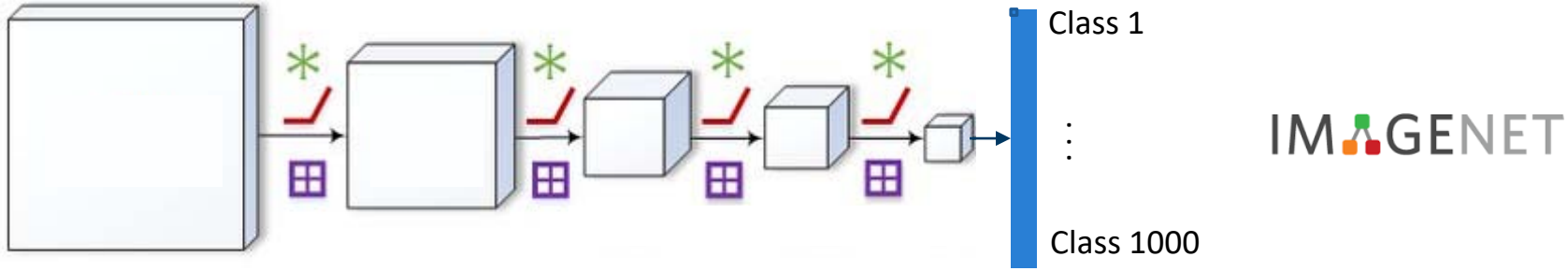
Pre-training

(massive data)

IMAGENET

1.2 million images

1000 object classes

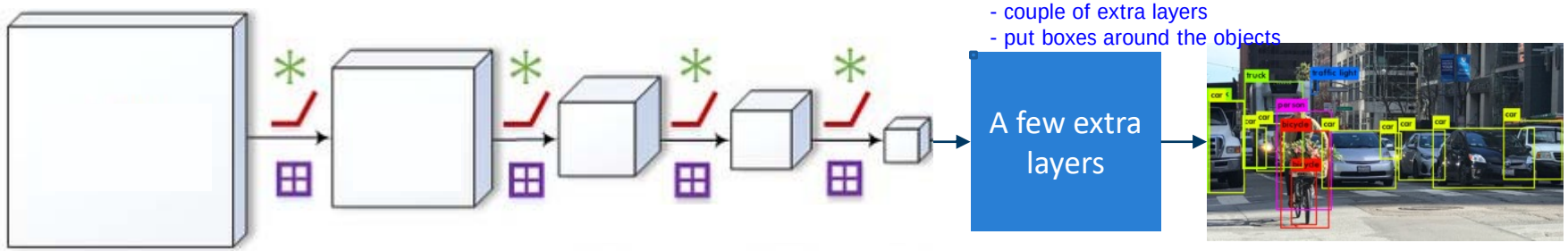


Transfer weights

2

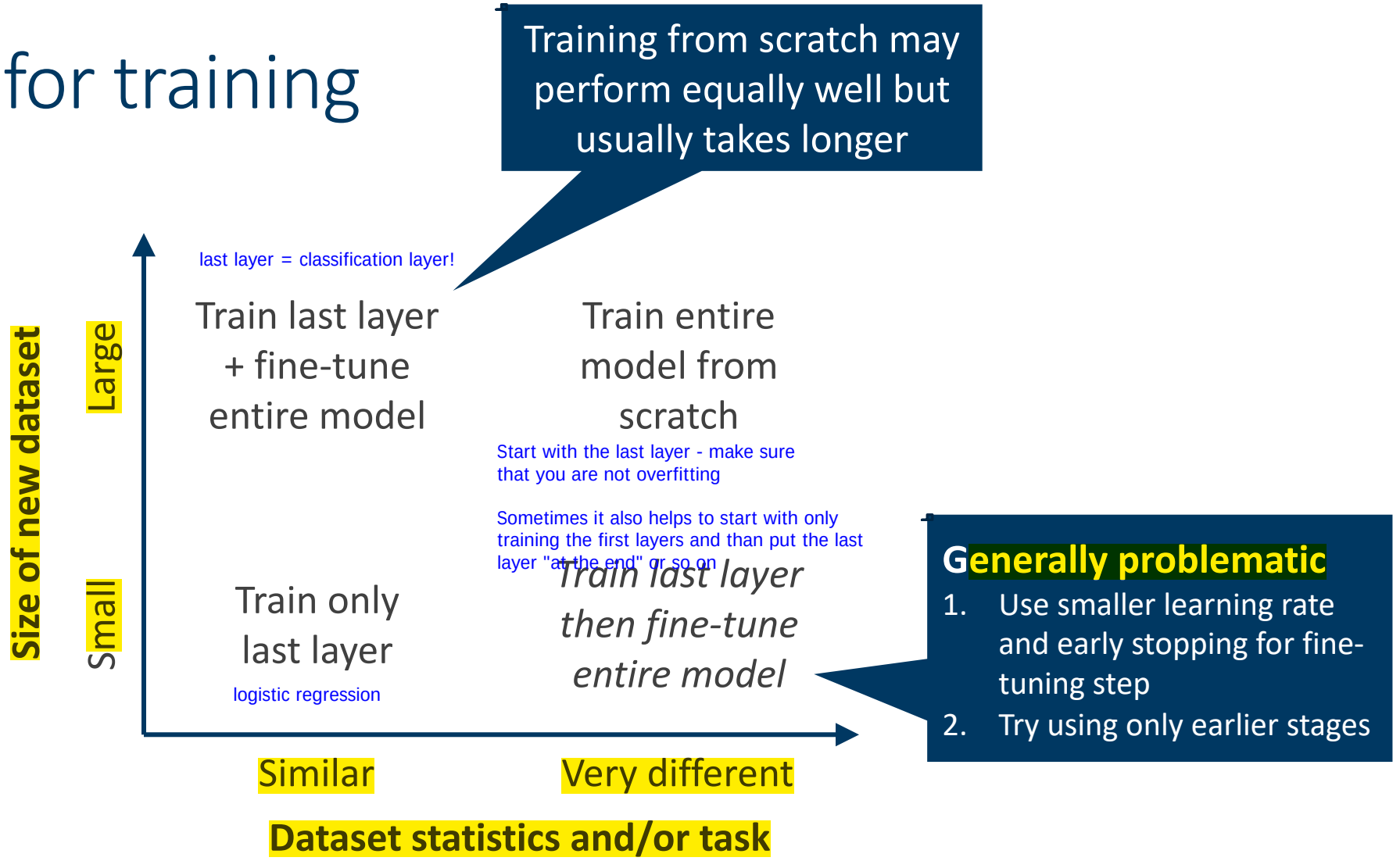
Fine-tuning

(much less data)



Object detection

Strategies for training



Better ImageNet lead to improvements on downstream tasks

the used the differnet architectures from the last lectures

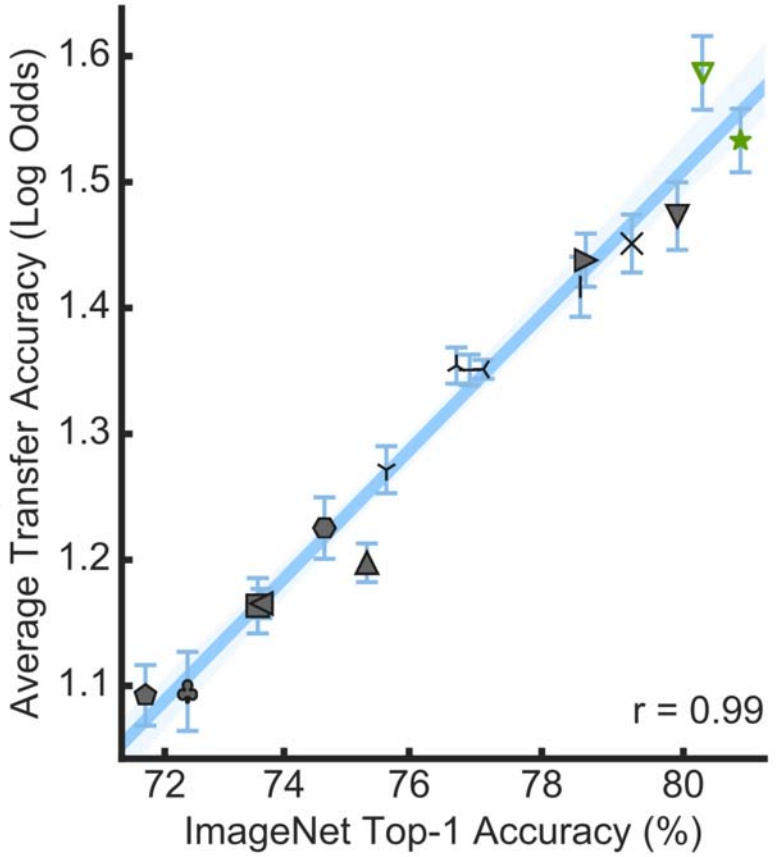
improvement by updating our classifier, no effort on the complexity

Do Better ImageNet Models Transfer Better?

Yes!

Simon Kornblith[illegible] and Quoc V. Le
[illegible]oogle Brain
{skornblith,shlens,qvl}@google.com

Tested on 12 different image classification sets



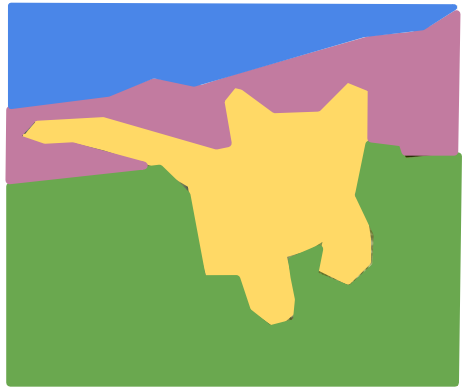
Computer vision tasks

Image Classification

Cat

No spatial extent

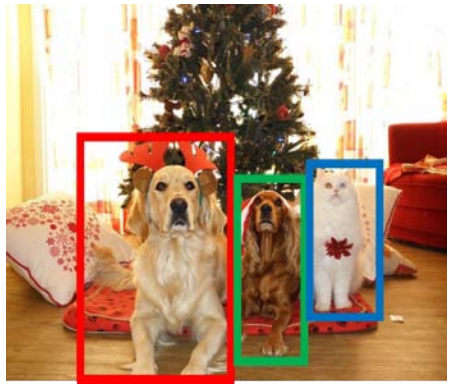
Semantic Segmentation



Sky, Trees, Grass, Cat

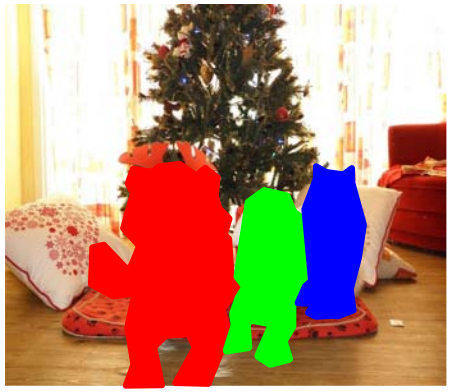
No objects, just pixels

Object Detection



Dog, Dog, Cat

Instance Segmentation

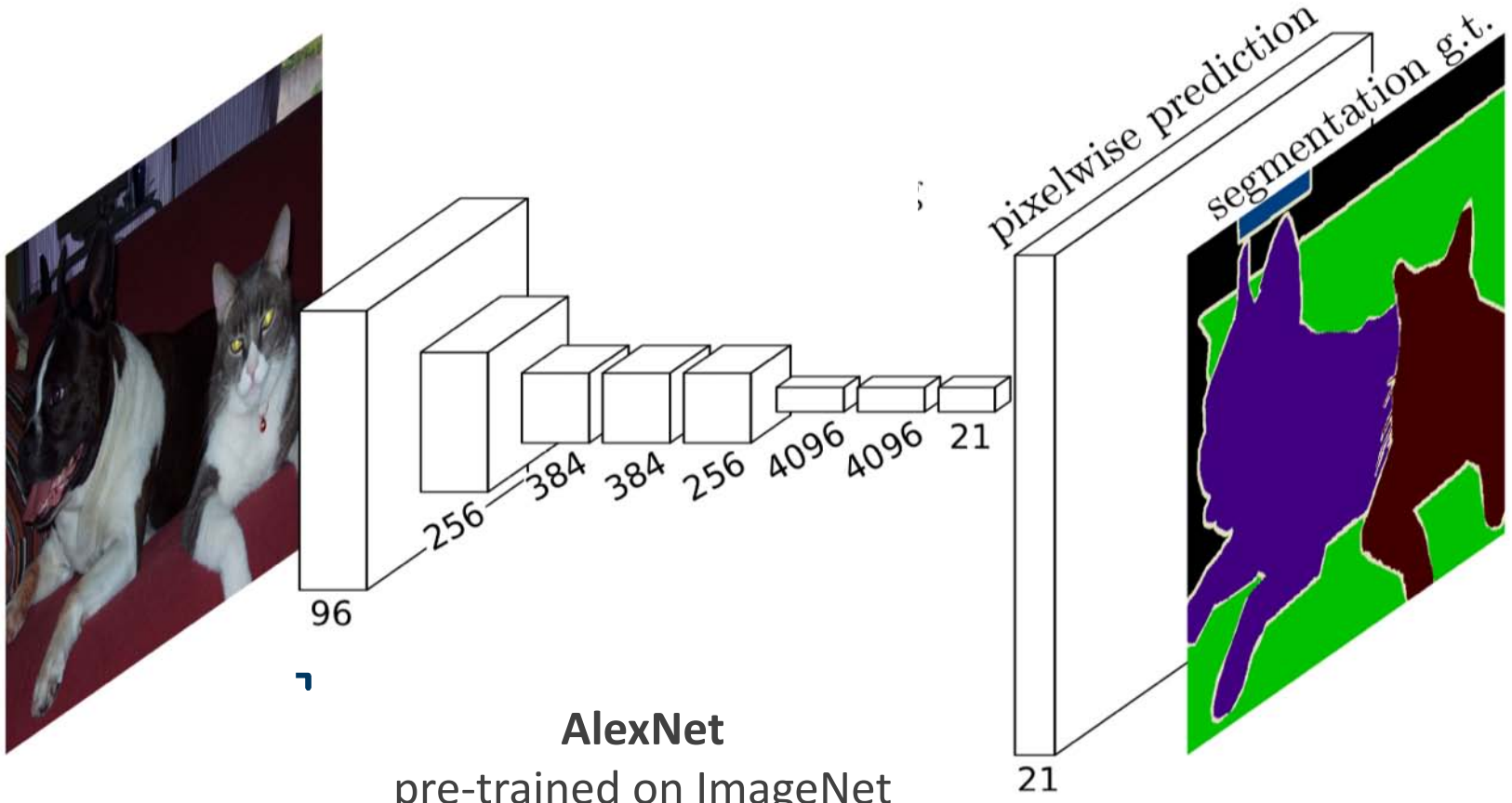


Dog, Dog, Cat

Multiple objects

Source: Fei Fei Li, Justin Johnson, Serena Yeung

Semantic segmentation

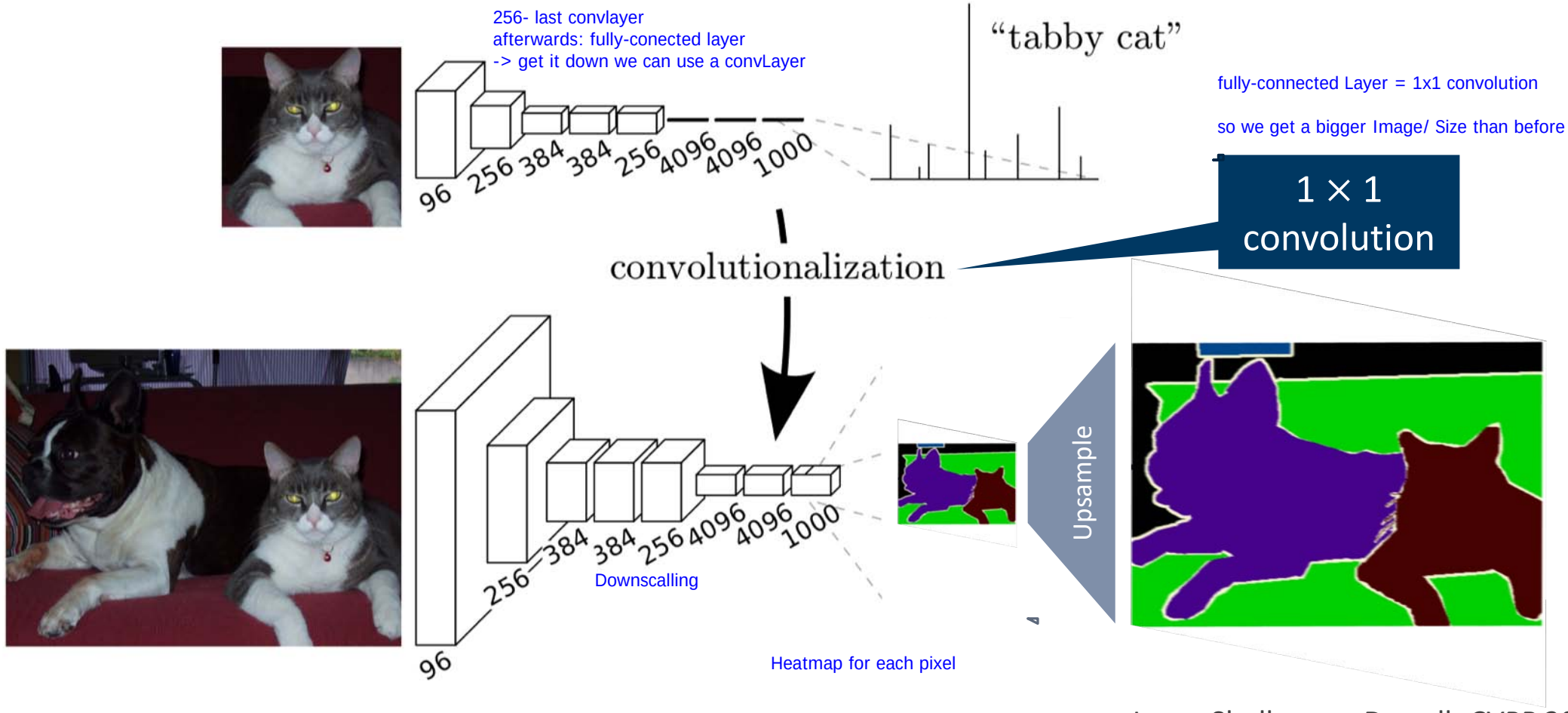


AlexNet
pre-trained on ImageNet

Long, Shelhamer, Darrell, *CVPR* 2015

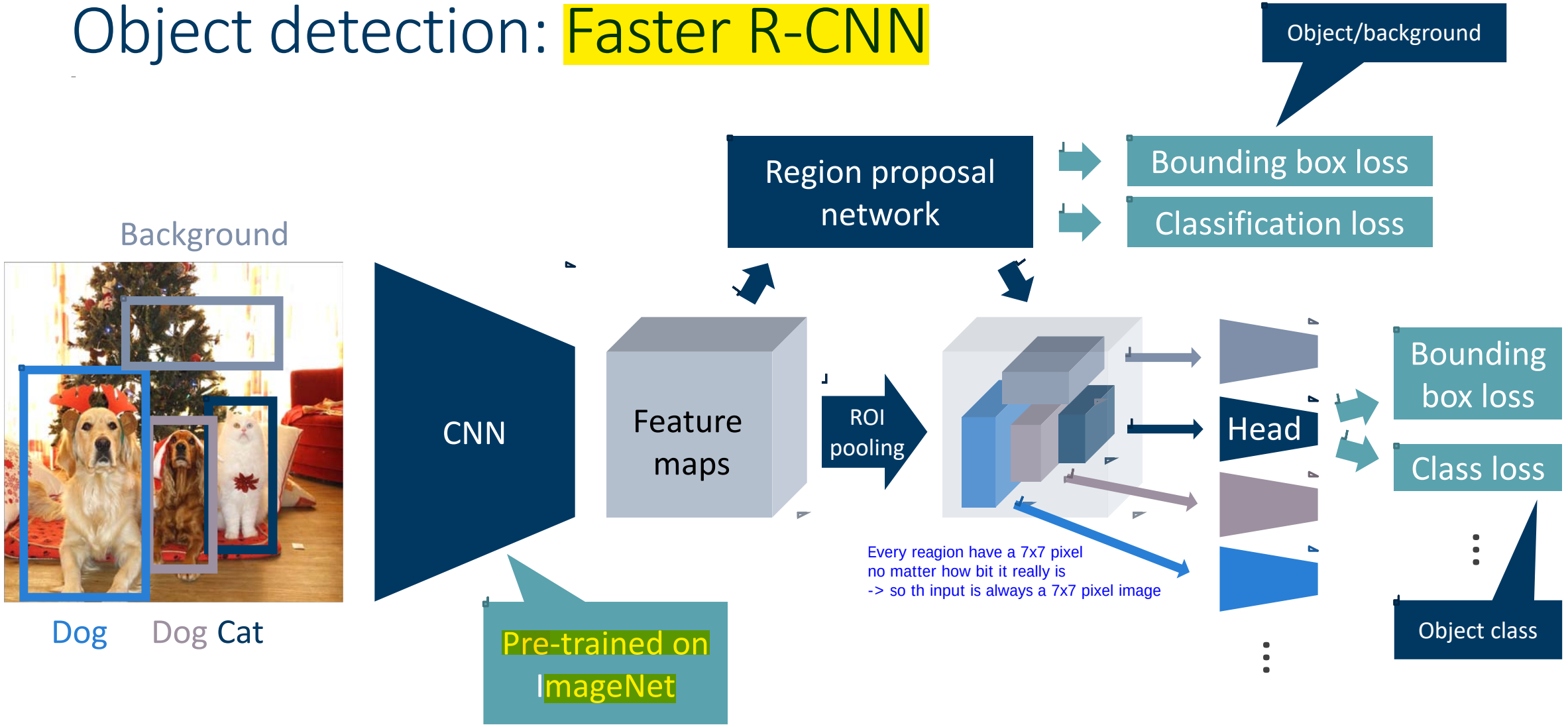
Turning fully-connected layers into conv layers

Image -> feature map -> ... -> image -> feature map -> fully-connected layer



Long, Shelhamer, Darrell, *CVPR* 2015

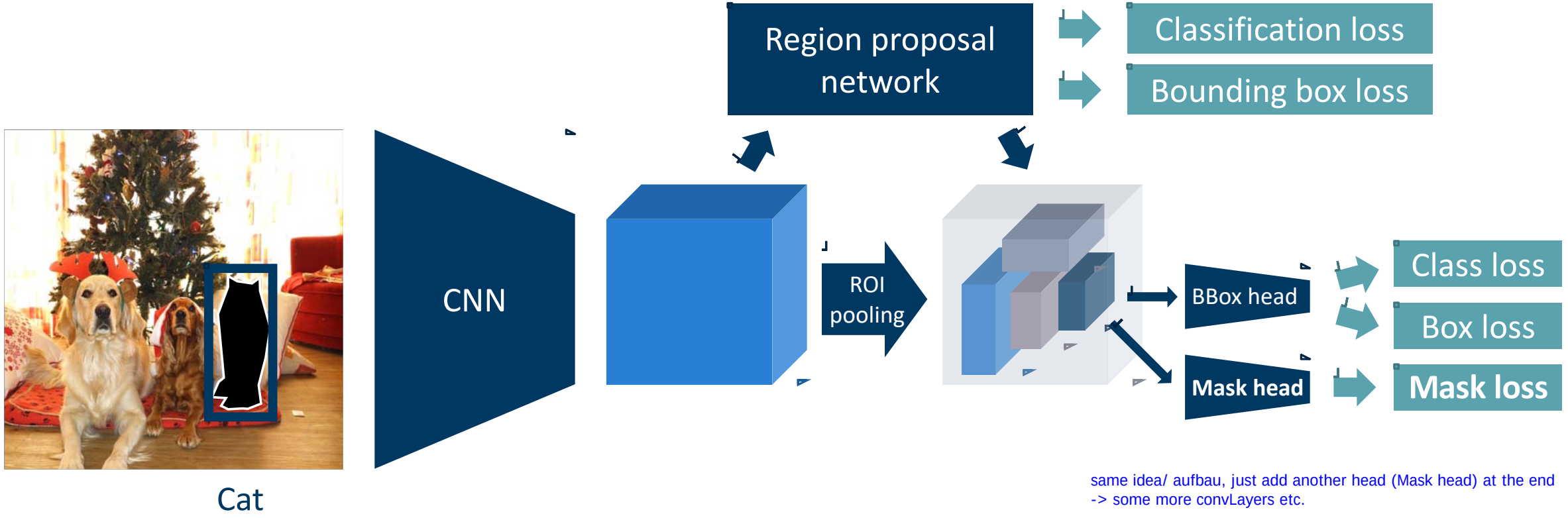
Object detection: Faster R-CNN



Ren et al., *NeurIPS* 2015

which pixel is one object and which pixel is another object?!

Instance segmentation: Mask R-CNN



same idea/ aufbau, just add another head (Mask head) at the end
-> some more convLayers etc.

He et al., *CVPR* 2017

Single-stage detectors: SSD, YOLO, RetinaNet

Advantage: this systems are faster
Disadvantage: but not so accurate

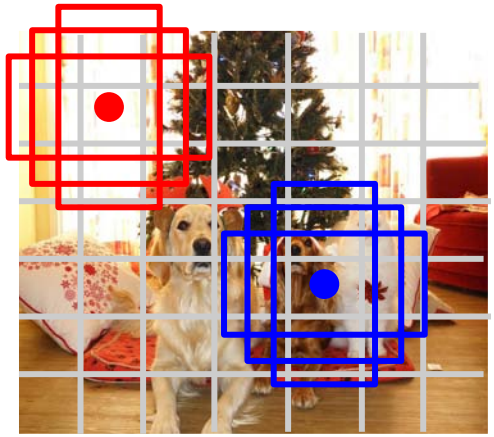
Image

H × W input

CNN

Pre-trained on ImageNet

Bounding box grid



7 × 7 or 14 × 14 grid

For each grid cell

Bounding box regression $(\Delta x, \Delta y, \Delta h, \Delta w)$

Classification in C classes + background

Output N boxes with highest confidence

A few extra layers per bbox

Liu et al., *ECCV* 2016 | Redmon et al., *CVPR* 2016 | Lin et al., *CVPR* 2017

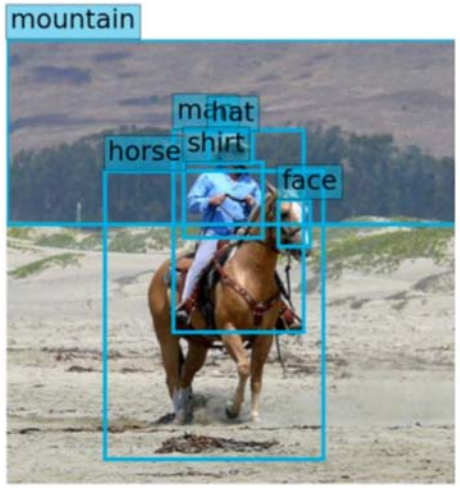
Pre-training on ImageNet is everywhere …

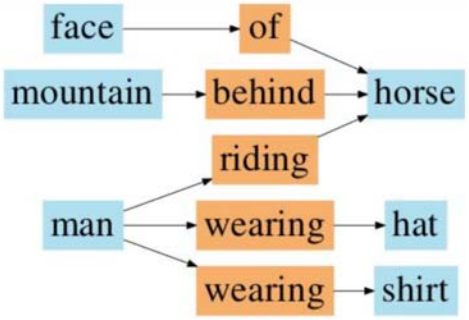
Human pose estimation

Image captioning

"two young girls are playing with lego toy."

Scene graph prediction





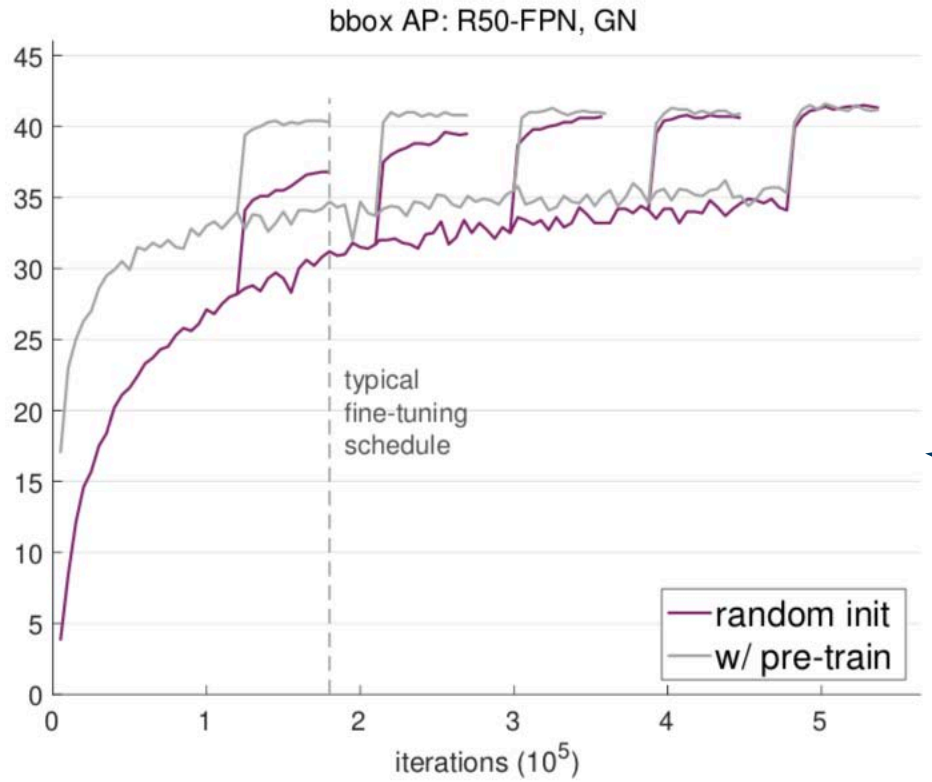
…

Many, many more

Transfer learning from ImageNet is the standard…

… but may not always be necessary

Not always necessary to pre-train on image net, but is faster



Object detection
on MS COCO dataset
123,287 images
886,284 instances

He, Girshick, Dollar: “Rethinking ImageNet pre-training”, *ICCV* 2019

Visual saliency

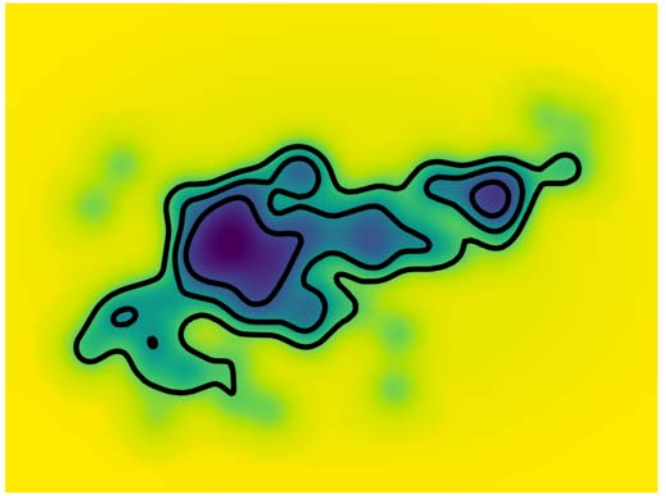
PREDICTING WHERE PEOPLE LOOK

Image saliency: predicting where people look

Image shown

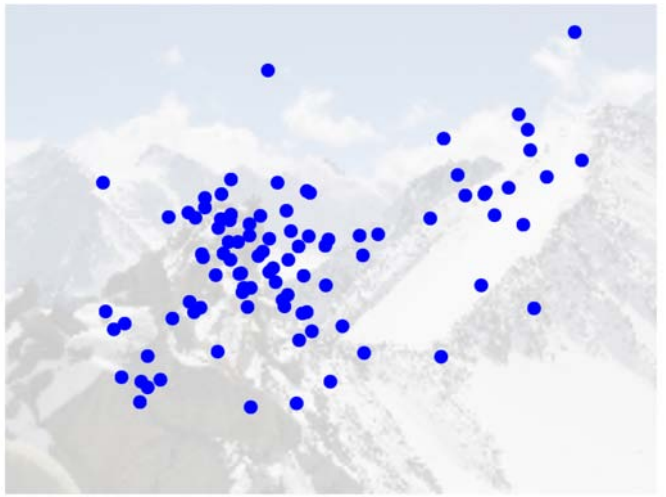
Predict

Goal:
Saliency map



$P(x, y \mid \text{image})$

Measured eye fixations



Loss

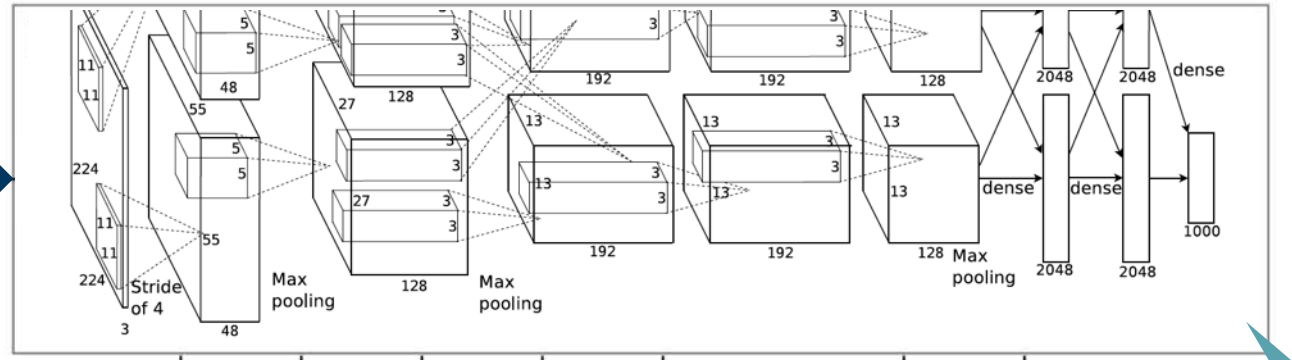
One slide is missing!
Deep Gaze: loss fct!!

its a pipeline

Deep Gaze I

engl. gaze = Blick

Kümmerer, Theis, Bethge,
ICLR Workshops 2015



for the homework we use instead of AlexNet: VGG

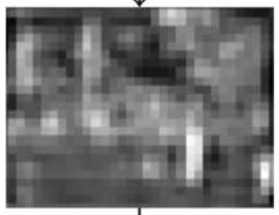
AlexNet
pre-trained on ImageNet

$r_k(x, y)$

Large filterbank

just linear comb., no normilization $\sum_k w_k r_k(x, y)$

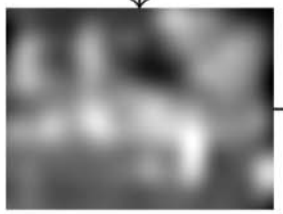
Linear prediction



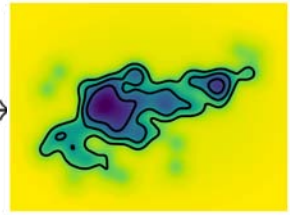
A-priori saliency map (center bias)

it is an additional step
-> when people look at image the look firstly at the center, because of the pos of the head

Blurring (as regularization)



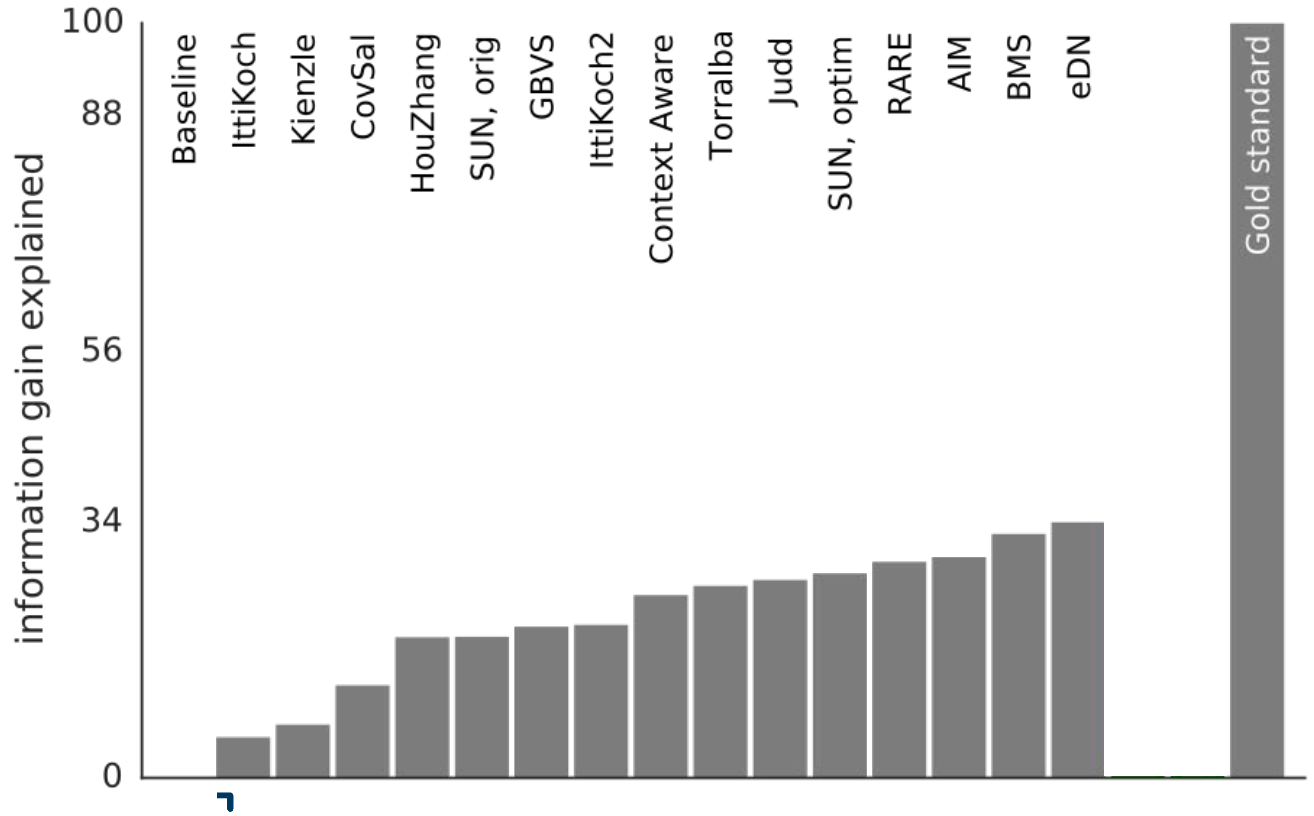
softmax



Saliency map (probability distribution)

like a logLikeliehood

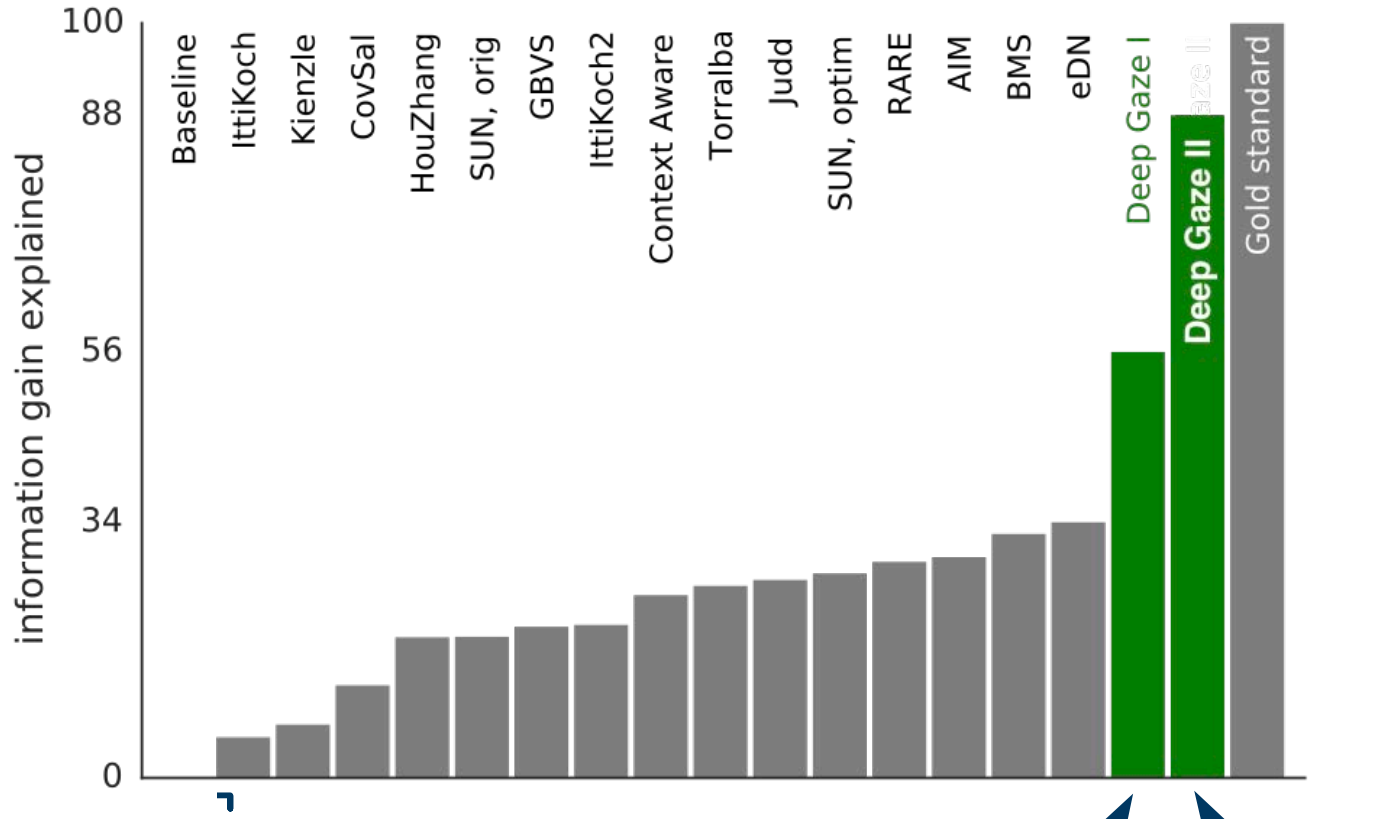
Deep Gaze I + II



20 years
of research…

Kümmerer, Theis, Bethge, *ICLR Workshops* 2015

Deep Gaze I + II



Kümmerer, Theis, Bethge,
ICLR Workshops 2015
Kümmerer, Wallis, Bethge, *arXiv* 2016

Neural style transfer

TEXTURE SYNTHESIS AND PERCEPTUAL LOSS FUNCTIONS

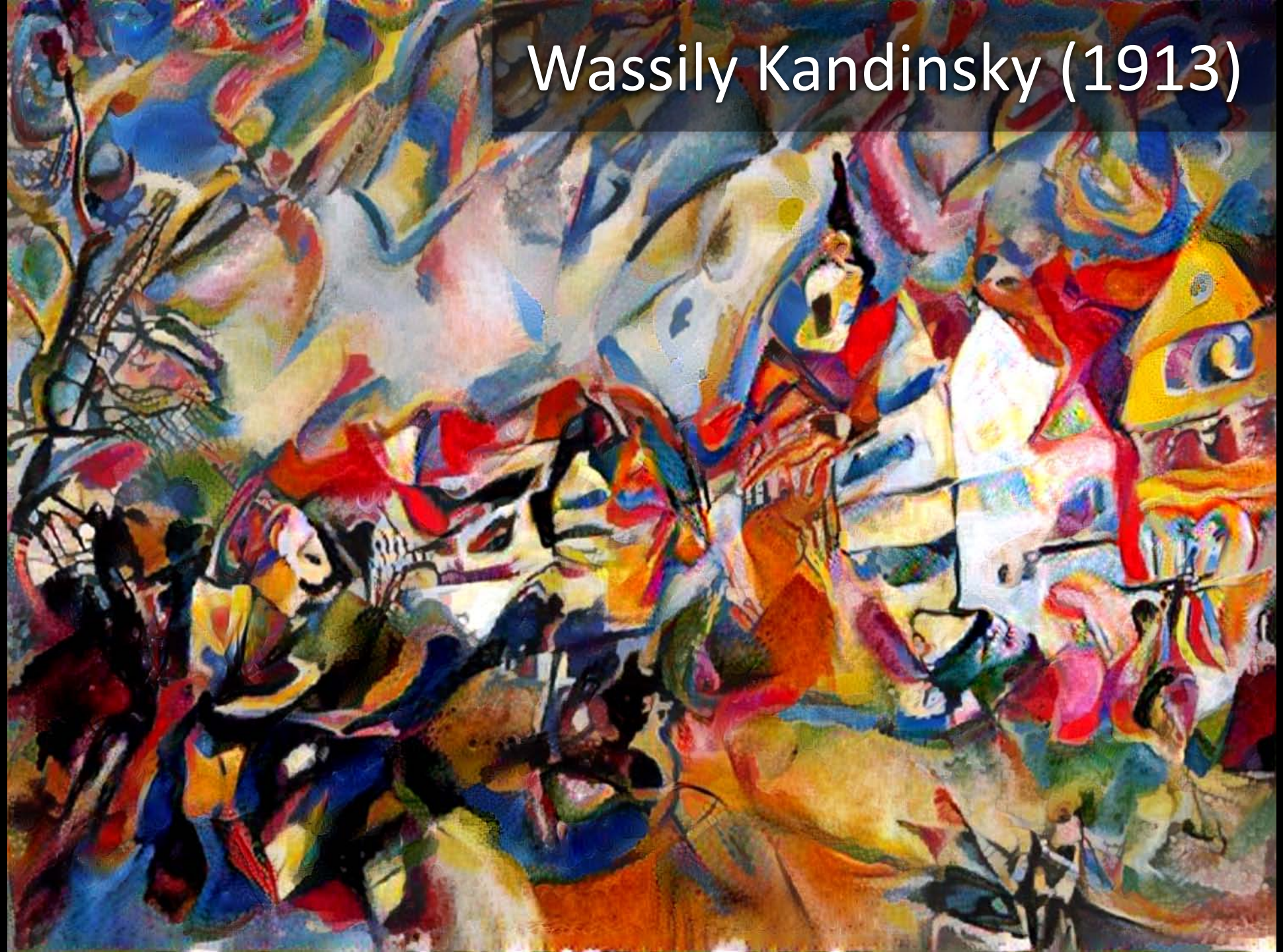
History of art – Tübingen


Vincent van Gogh (1889)


William Turner (1805)

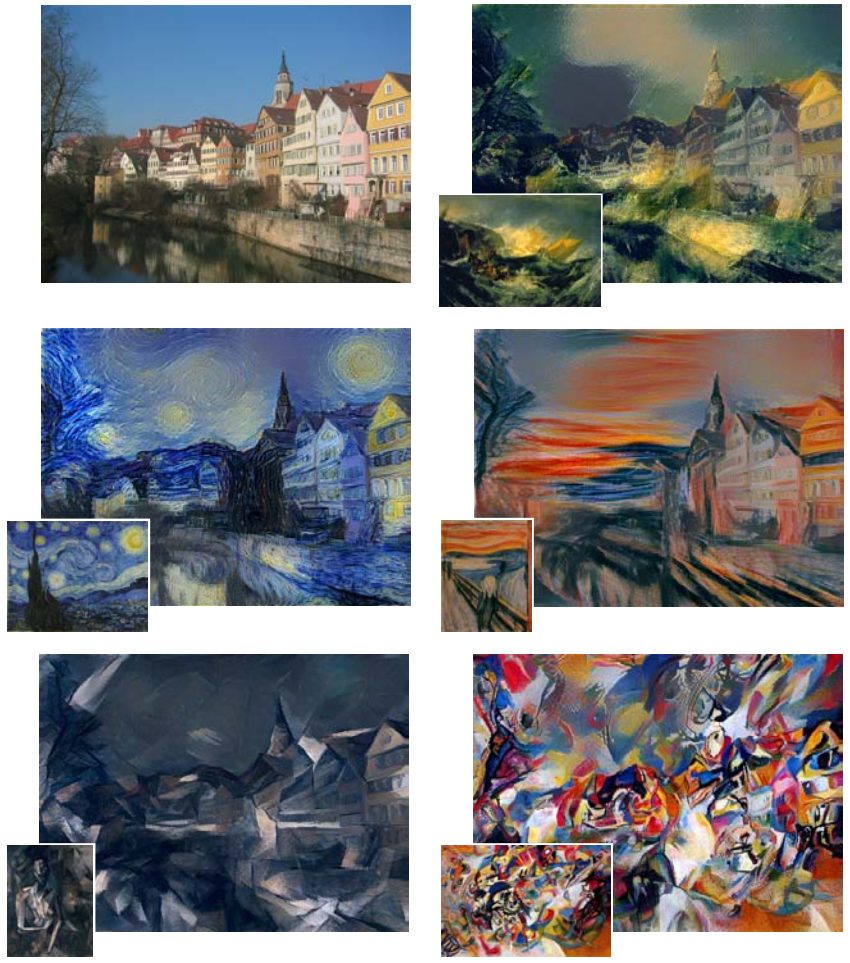

Pablo Picasso (1910)


Edvard Munch (1893)


Wassily Kandinsky (1913)

A Neural Algorithm of Artistic Style

Gatys, Ecker, Bethge (CVPR 2016)



Neural Style Transfer

Content

Style (→ texture)

Gatys, Ecker Bethge, CVPR 2016

A brief history of parametric texture modeling

Julesz (1962)

- Conjecture: N^{th} order summary statistics determine texture perception

Portilla & Simoncelli (2000)

- Summary statistics of features from early visual system (steerable pyramid)

Gatys, Ecker Bethge (NIPS 2015)

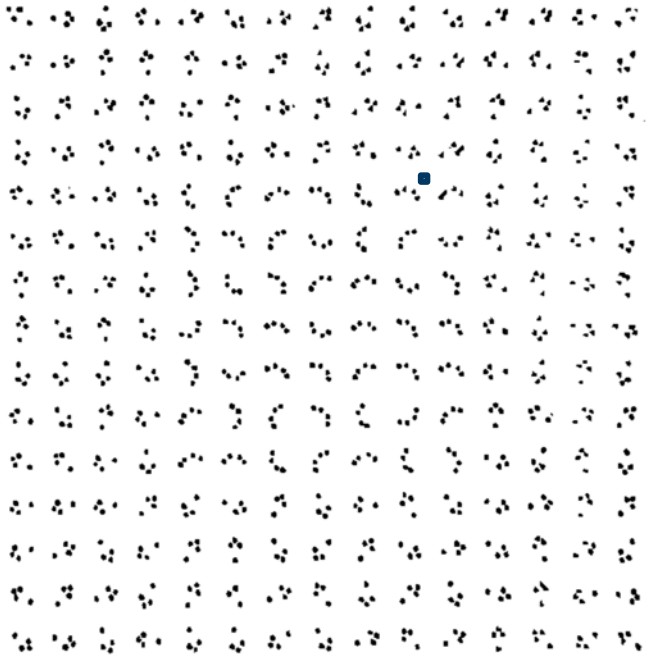
- Summary statistics of features from CNNs trained on ImageNet

Julesz' conjecture (1962)

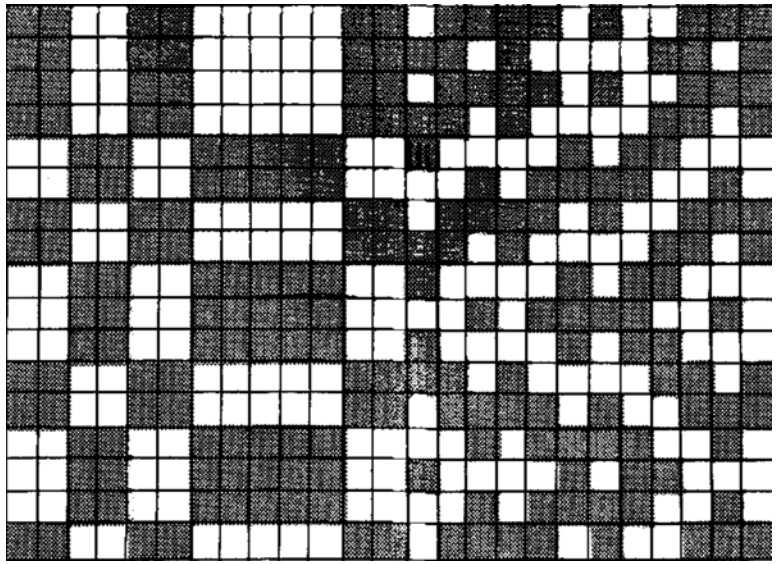
2nd and 3rd order statistic is not enough, as we can see
-> so it was a nice idea, but not really works

N^{th} order summary statistics determine texture perception.

Uniform 2nd order statistics (power spectrum)



Uniform 3rd order statistics



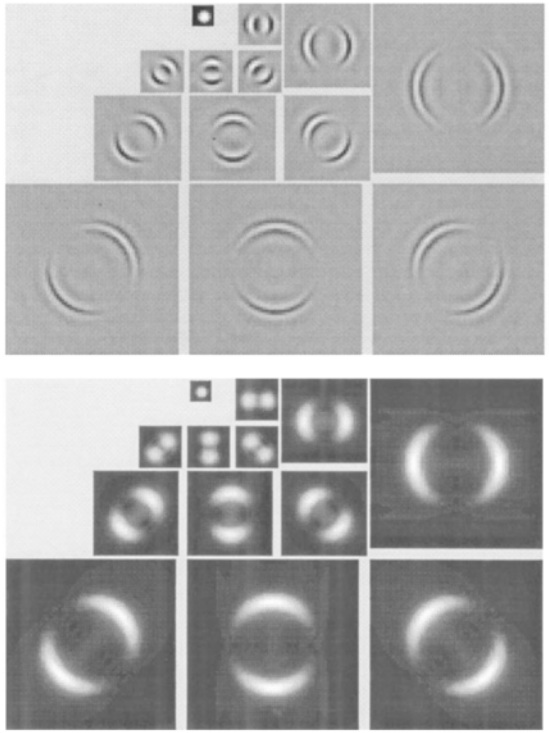
Julesz 1962, Julesz et al. 1978

Portilla & Simoncelli texture model

Image

Steerable pyramid

(Simoncelli & Freeman 1995)



multiple filters and feature maps

Summary statistics

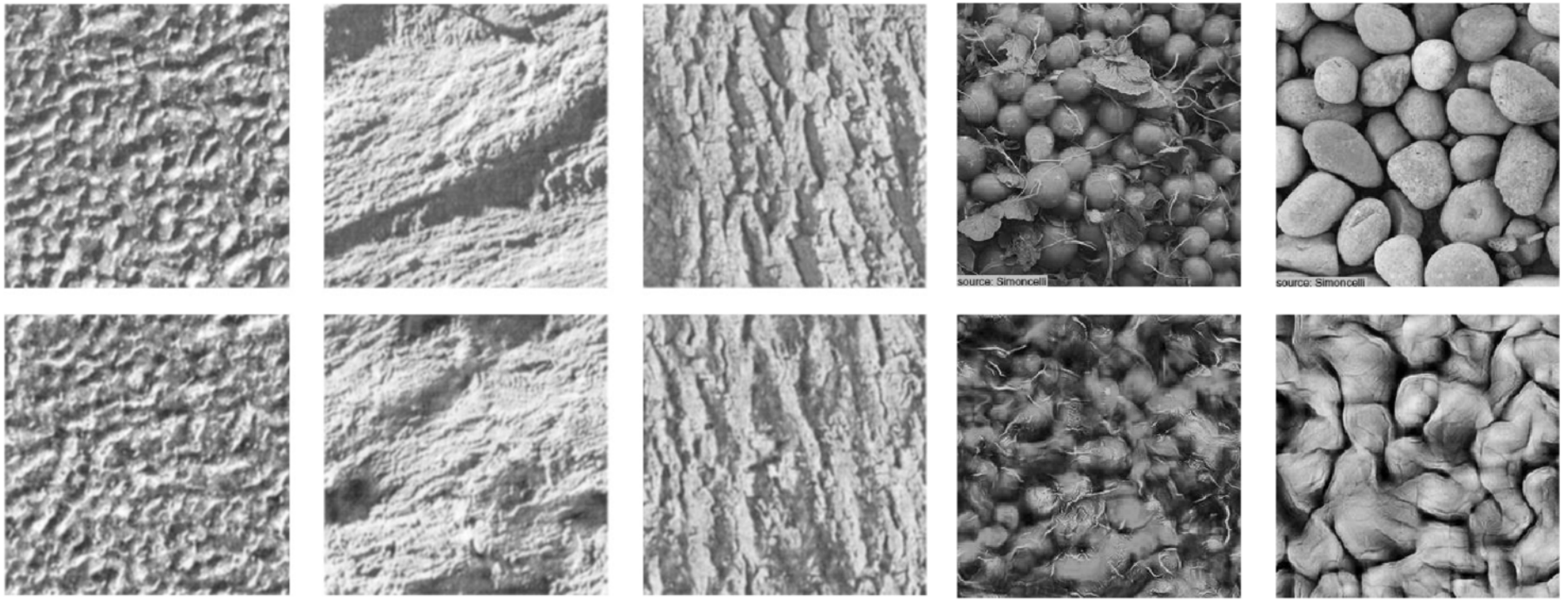
- Mean
- Variance
- Correlations across
  - Space
  - Features

Portilla & Simoncelli (2000)

Portilla & Simoncelli: results

not the same in terms of pixels, but the same in terms of texture
-> works for some well, but not for all

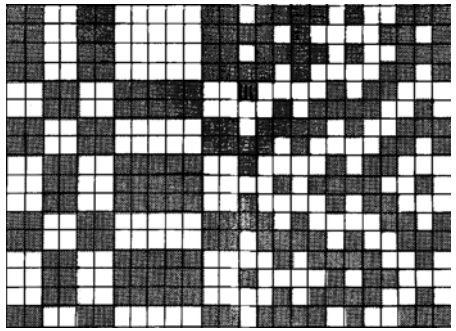
Original images



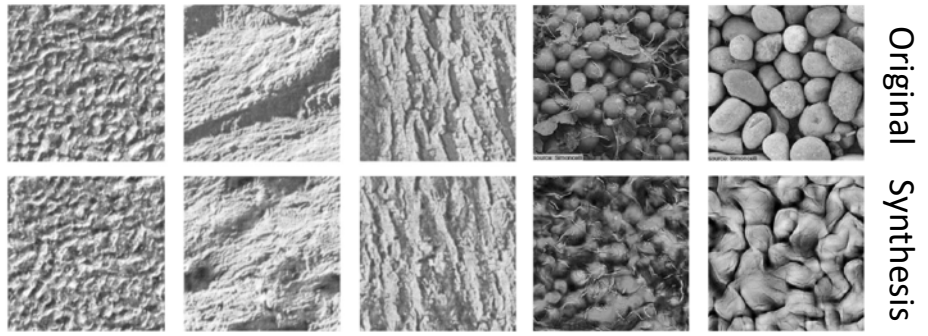
Synthesized by Portilla & Simoncelli

Portilla & Simoncelli (2000)

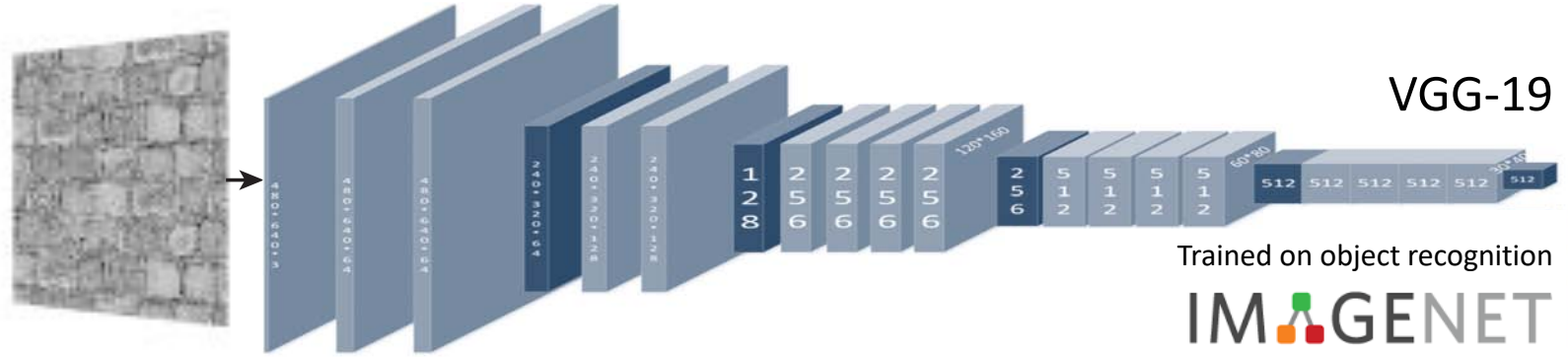
Texture models: summary statistics



Julesz (1962):
N^{th} order pixel stats

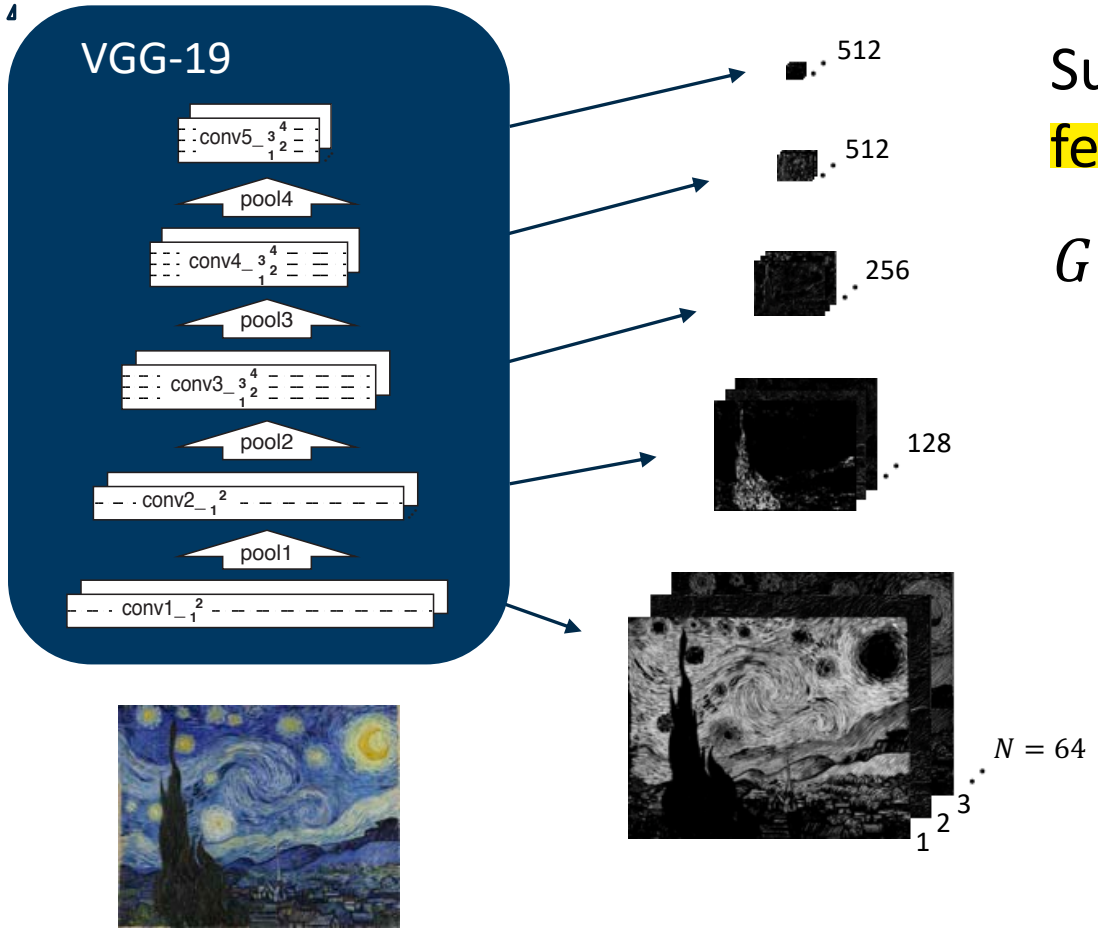


Portilla & Simoncelli (2000):
V1 summary statistics → Steerable Pyramid



Gatys, Ecker, Bethge (2015): Entire visual pathway → CNN features

CNN: Multiscale nonlinear filter bank



Summary statistic:

feature correlations

$$G = \Phi^{\mathrm{T}}\Phi$$

Gatys, Ecker Bethge, *NeurIPS* 2015

Gram matrix computation

conv1_1 feature maps

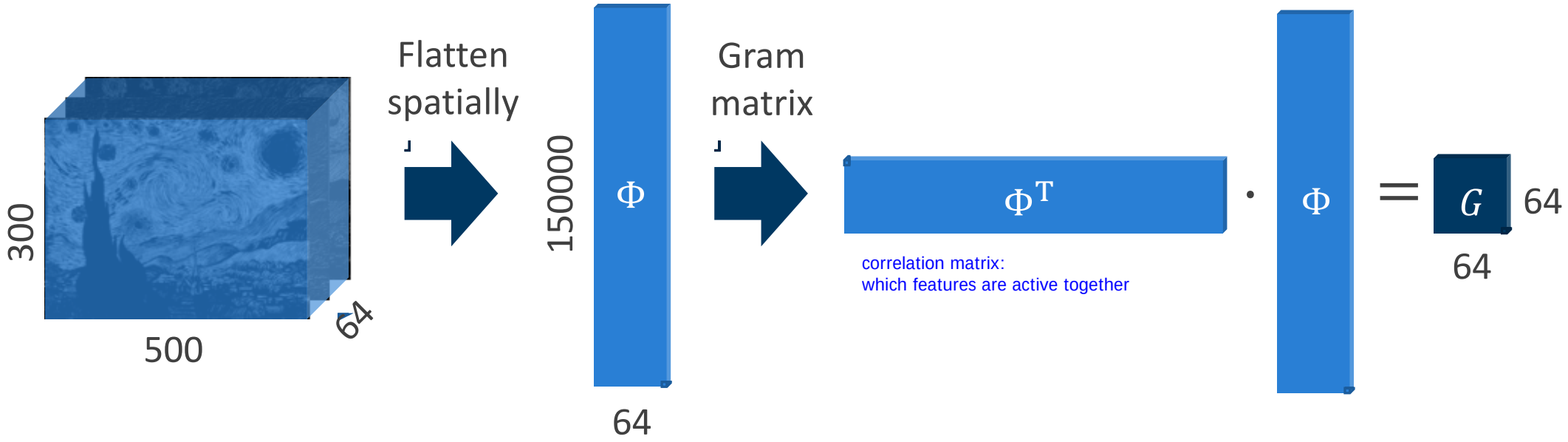
$300 \times 500 \times 64$

Flattened spatially

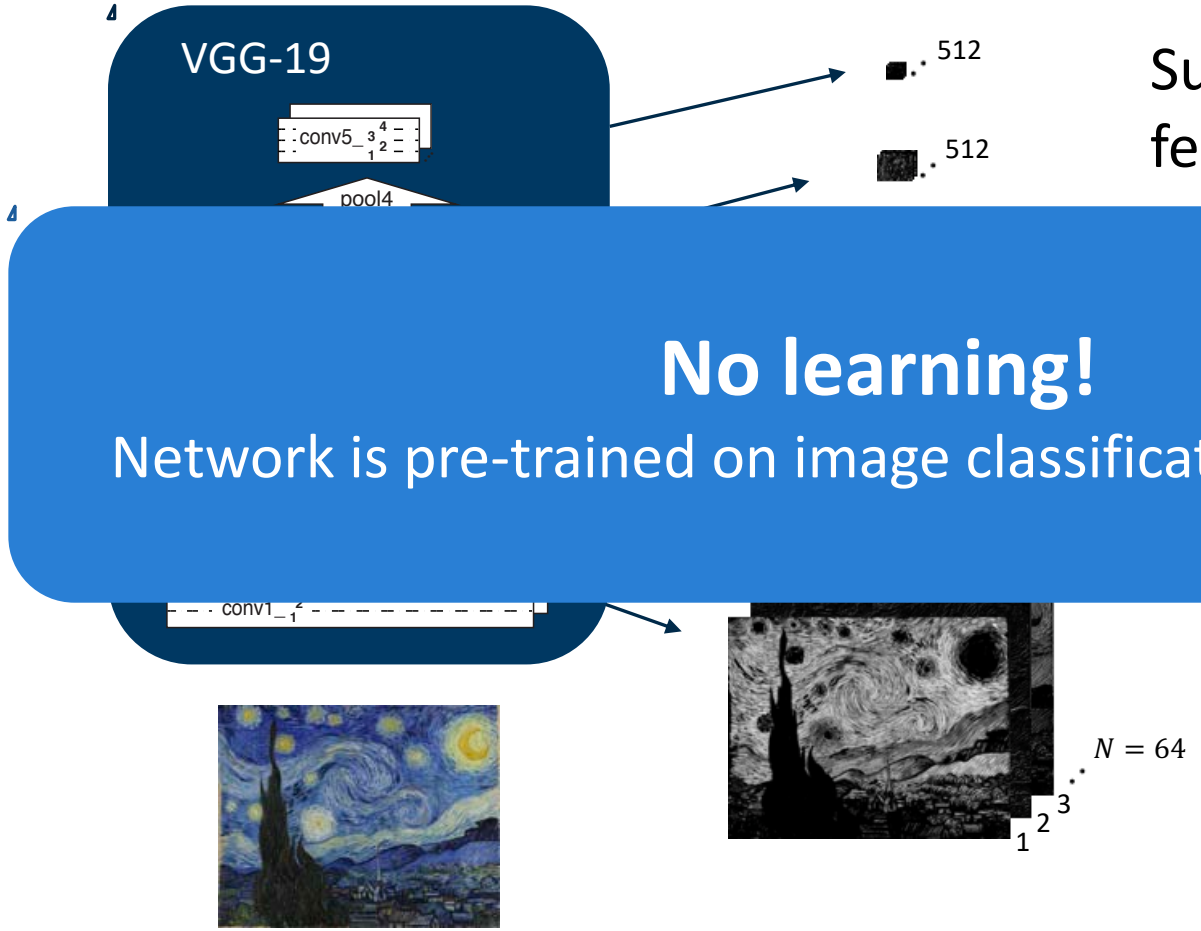
150000×64

Gram matrix

64×64



CNN: Multiscale nonlinear filter bank



Summary statistic: feature correlations

No learning!

Network is pre-trained on image classification on ImageNet

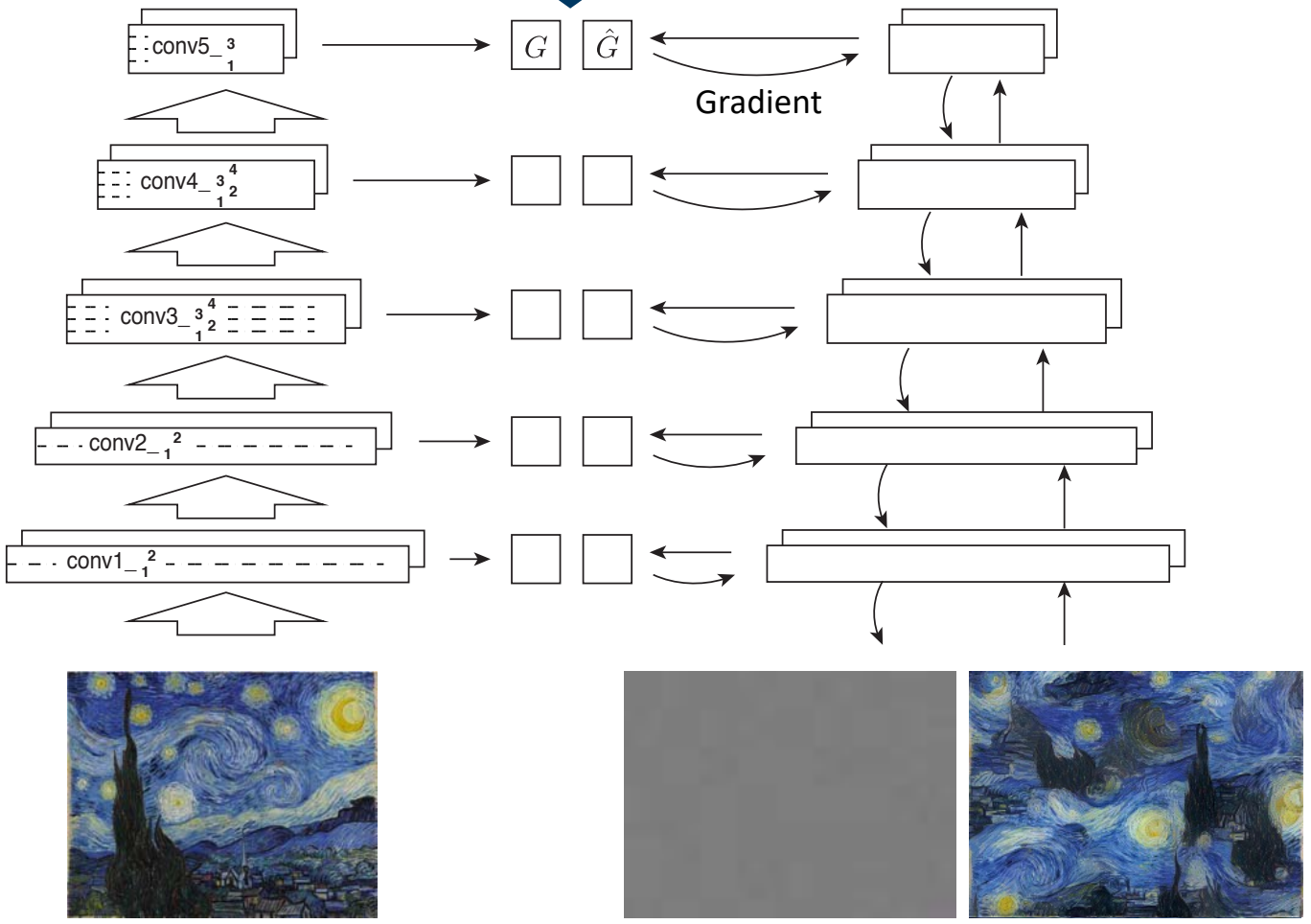
Gatys, Ecker Bethge, *NeurIPS* 2015

Texture loss

$$\mathcal{L}_{\text{style}} = \sum_{ij} \left(G_{ij} - \hat{G}_{ij}\right)^2$$

MSE between the target image and the real image

-> no updating the weights
-> updating the images



Gatys, Ecker Bethge, *NeurIPS* 2015

Textures based on deep features

CNN-based model

Original image

Portilla & Simoncelli

Gatys, Ecker Bethge, *NeurIPS* 2015

Neural Style Transfer

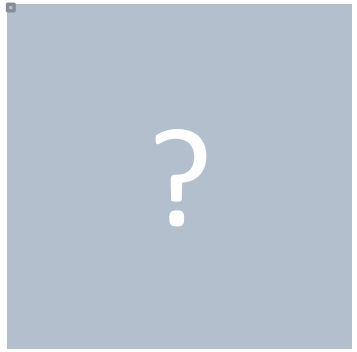
Content

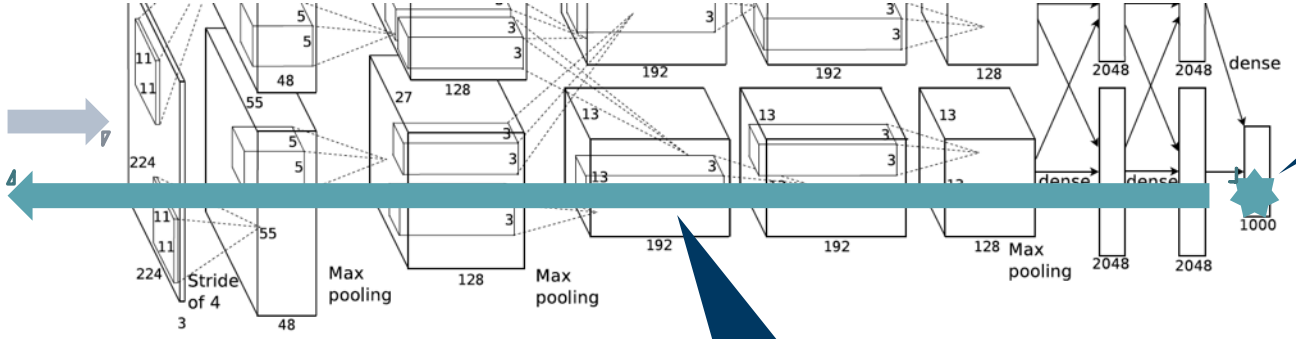
Style (→ texture)

Gatys, Ecker Bethge, *CVPR* 2016

Visualizing CNN features by activity maximization

Image





Goal: Maximize activation

Bell pepper

$$I^* = \arg\max f(I) - \lambda \|I\|_2^2$$

therefore the backprop gradient!

Found by gradient ascent

Backpropagate gradient

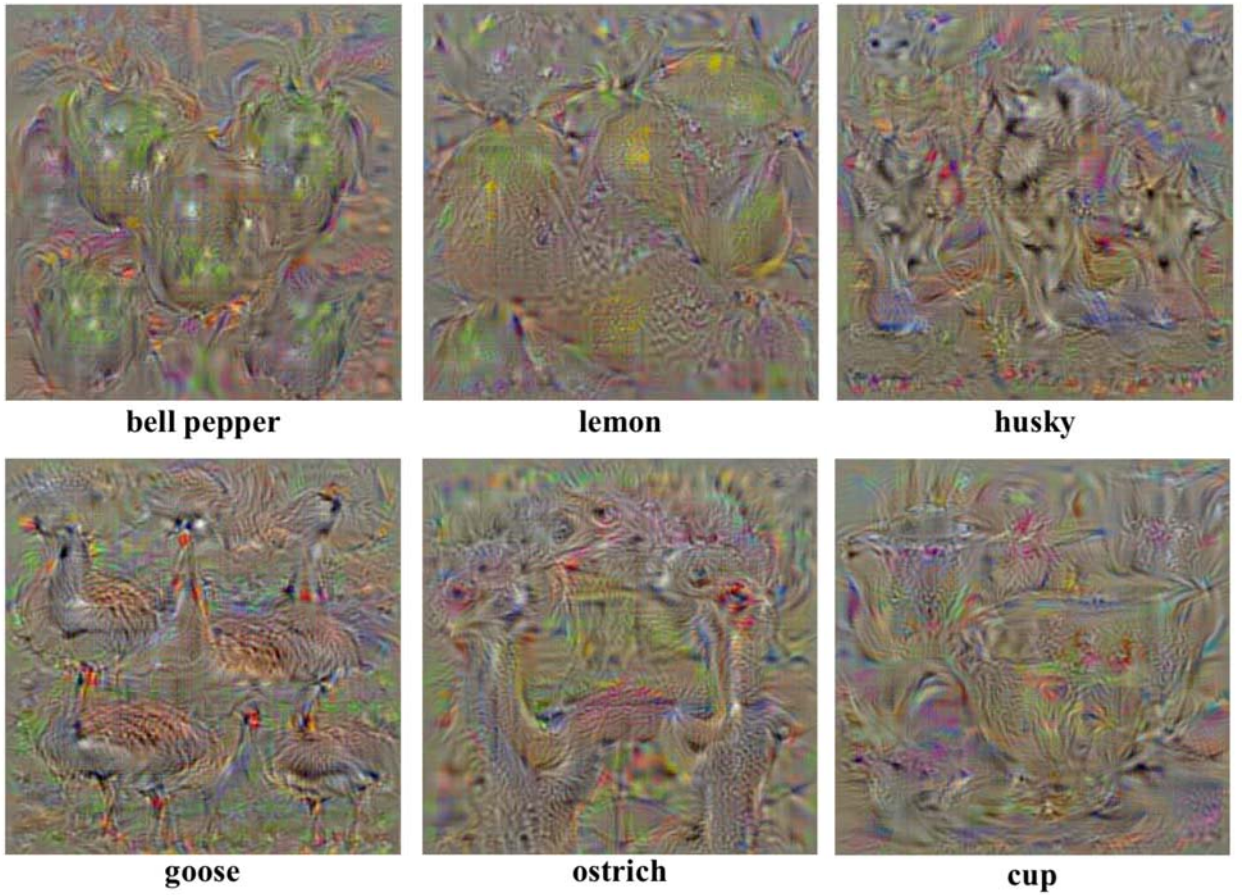
Iterate

1. Forward pass image
2. Backprop gradient
3. Update image

Visualizing classification layer

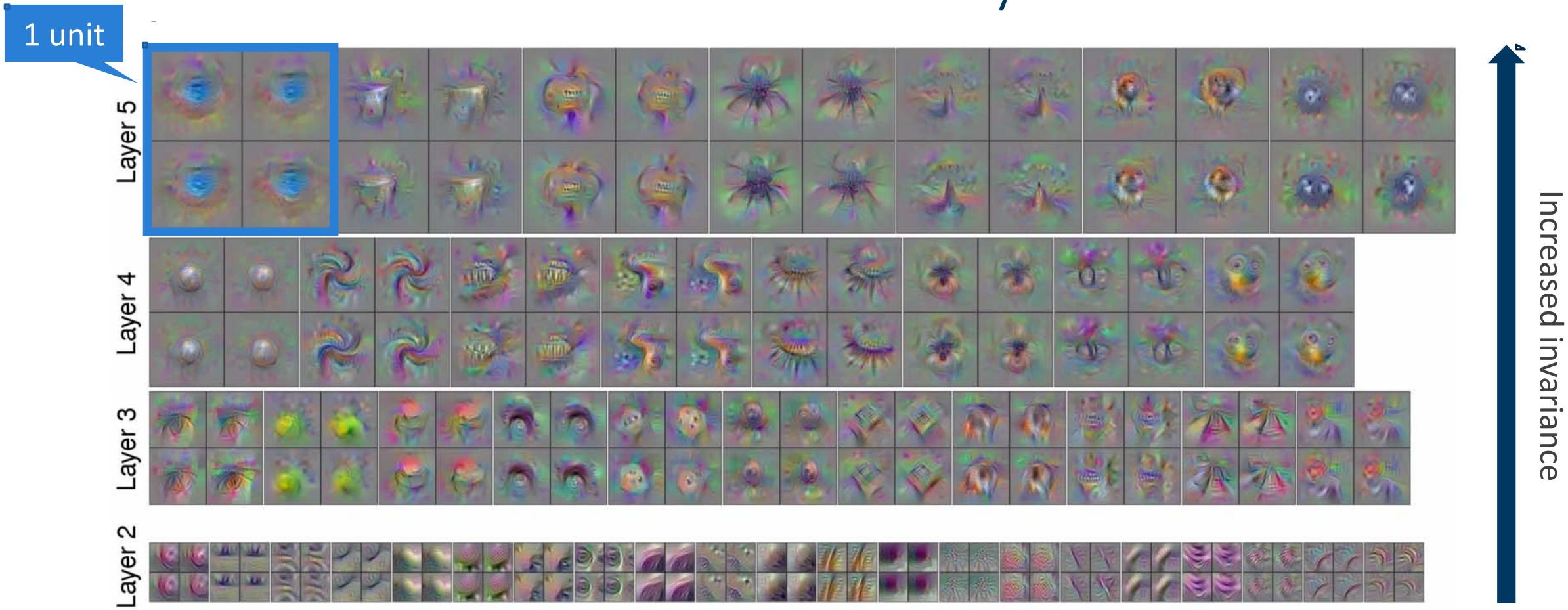
$$I^* = \arg\max f(I) - \lambda \|I\|_2^2$$

Regularizer to get "natural" images



Simonyan, Vedaldi, Zisserman, *ICLR Workshops* 2014

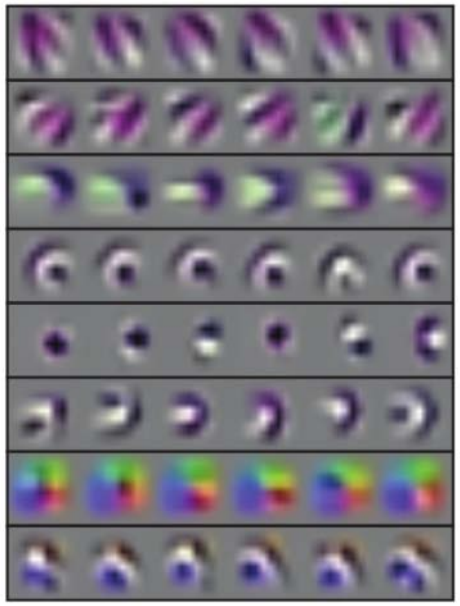
Visualization of intermediate layers of AlexNet



Yosinski et al., *ICML Deep Learning Workshop* 2015

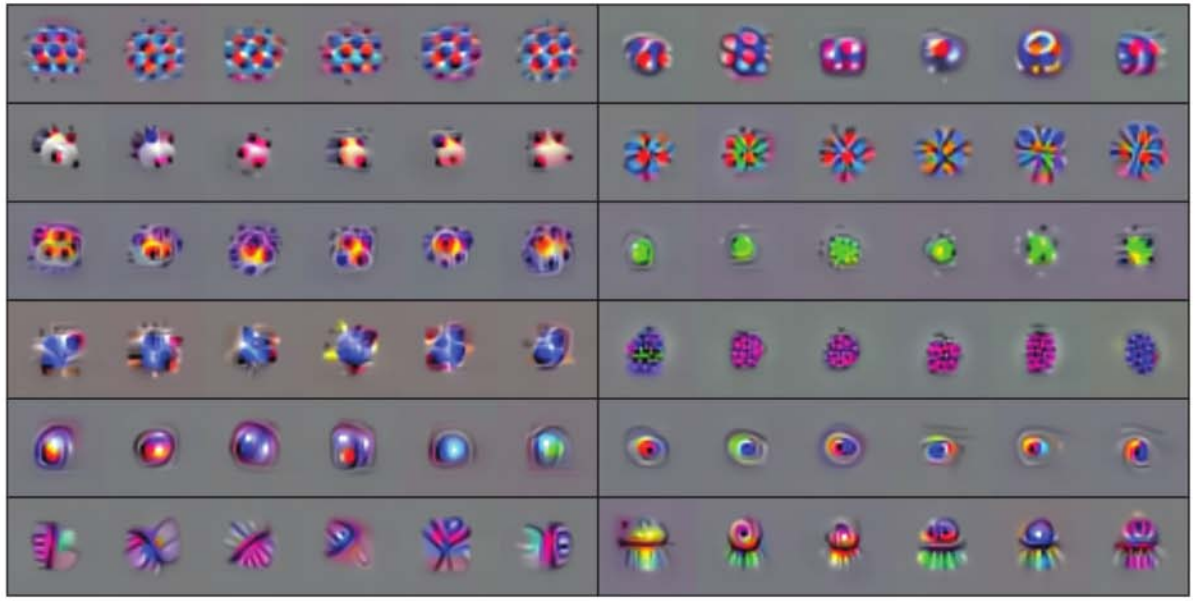
Visualization of intermediate layers of VGG-19

conv2_1



Increased invariance

conv3_3



Early: selective

Intermediate/late: Increased invariance

Optimization target here: obtain diverse set of images that maximize overall activity

Cadena et al., *ECCV* 2018

Style loss

$$\mathcal{L}_{\text{style}} = \sum_{ij} \left(G_{ij} - \hat{G}_{ij}\right)^2$$

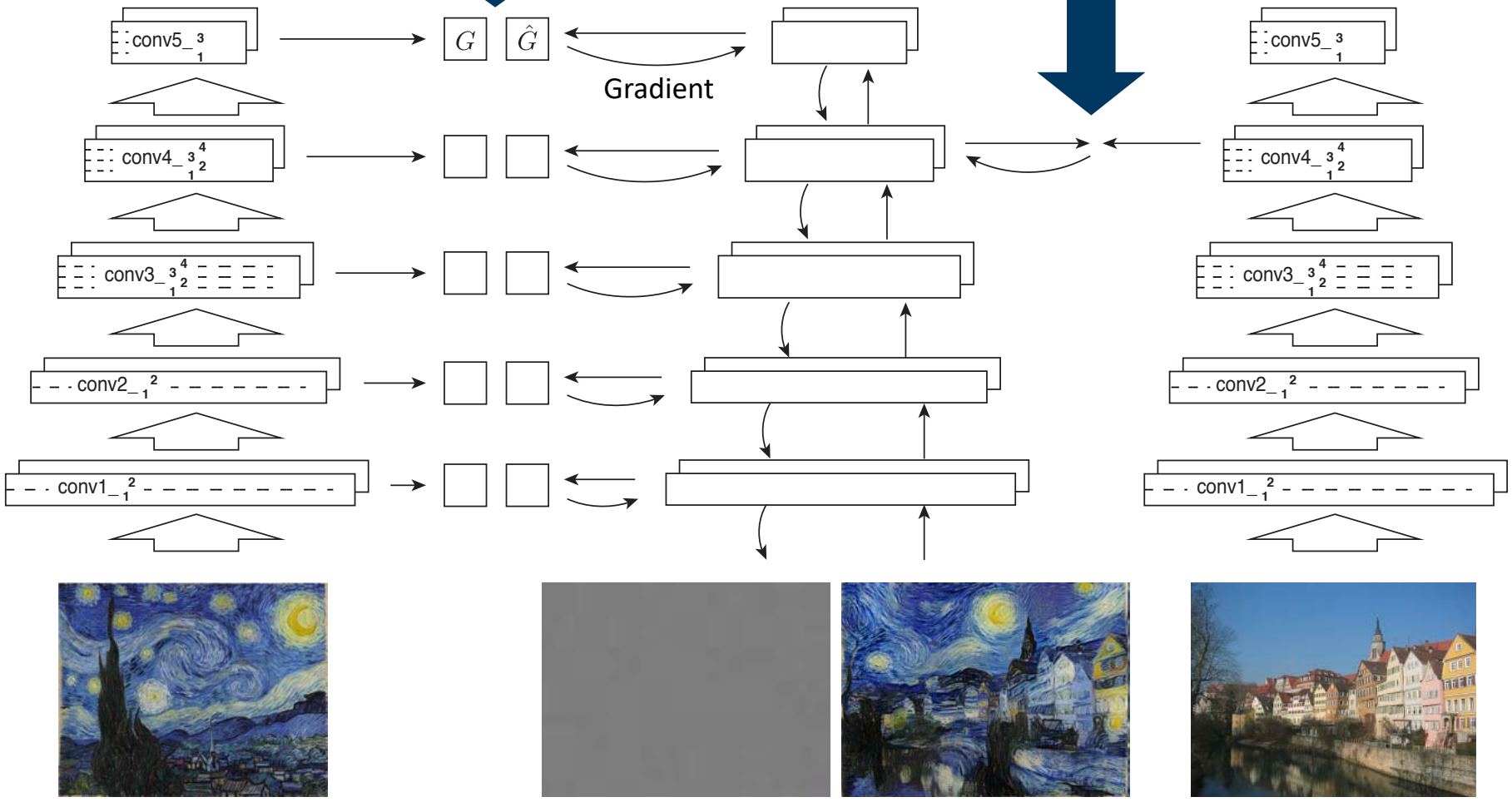
+

Content loss

$$\mathcal{L}_{\text{content}} = \sum_{ijk} \left(\Phi_{ijk} - \hat{\Phi}_{ijk}\right)^2$$

Matching the individual activations

Content Loss:
MSE between feature activations



Gatys, Ecker Bethge, *CVPR* 2016

Timelapse

Gatys, Ecker Bethge, *CVPR* 2016

deepart.io

DEEPART.io Latest artworks **CREATE YOUR OWN** Videos Offer About Register Sign in

Create your own artwork
https://deepart.io

TURN ANY PHOTO INTO AN ARTWORK – FOR FREE!

We use an algorithm inspired by the human brain. It uses the stylistic elements of one image to draw the content of another. Get your own artwork in just three steps.

1 Upload photo

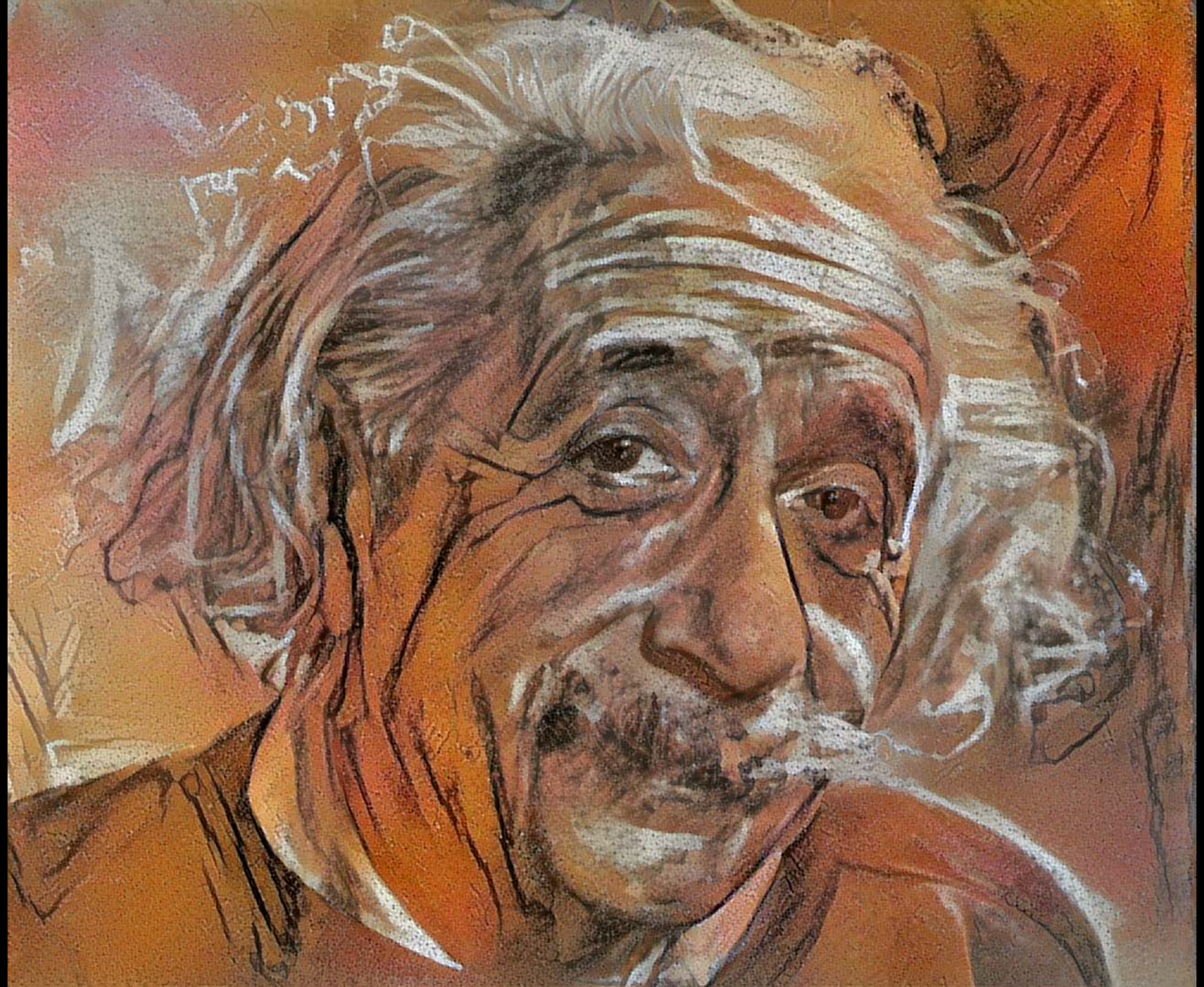
The first picture defines the scene you would like to have painted.

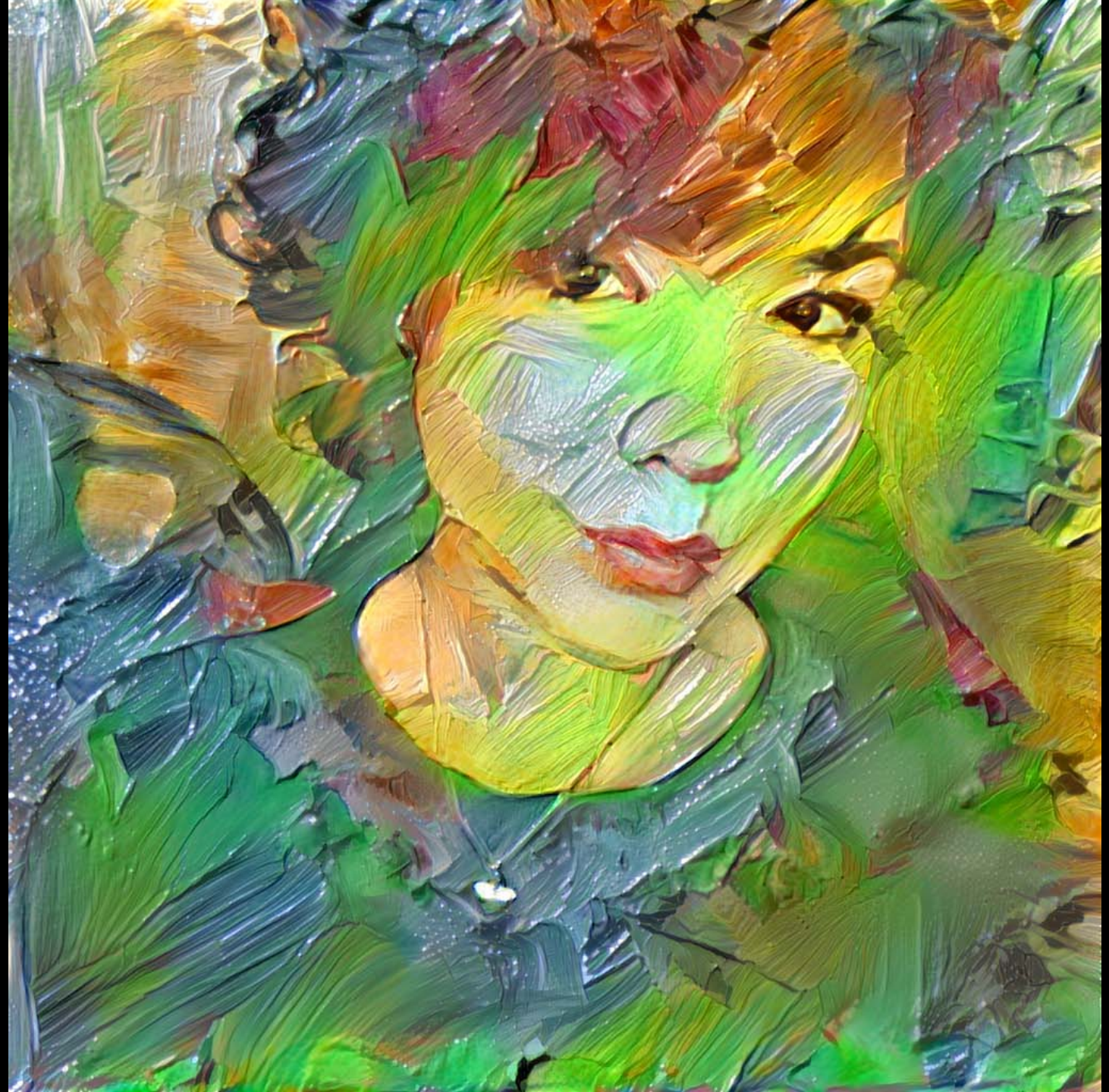
2 Choose style

Choose among predefined styles or upload your own style image.

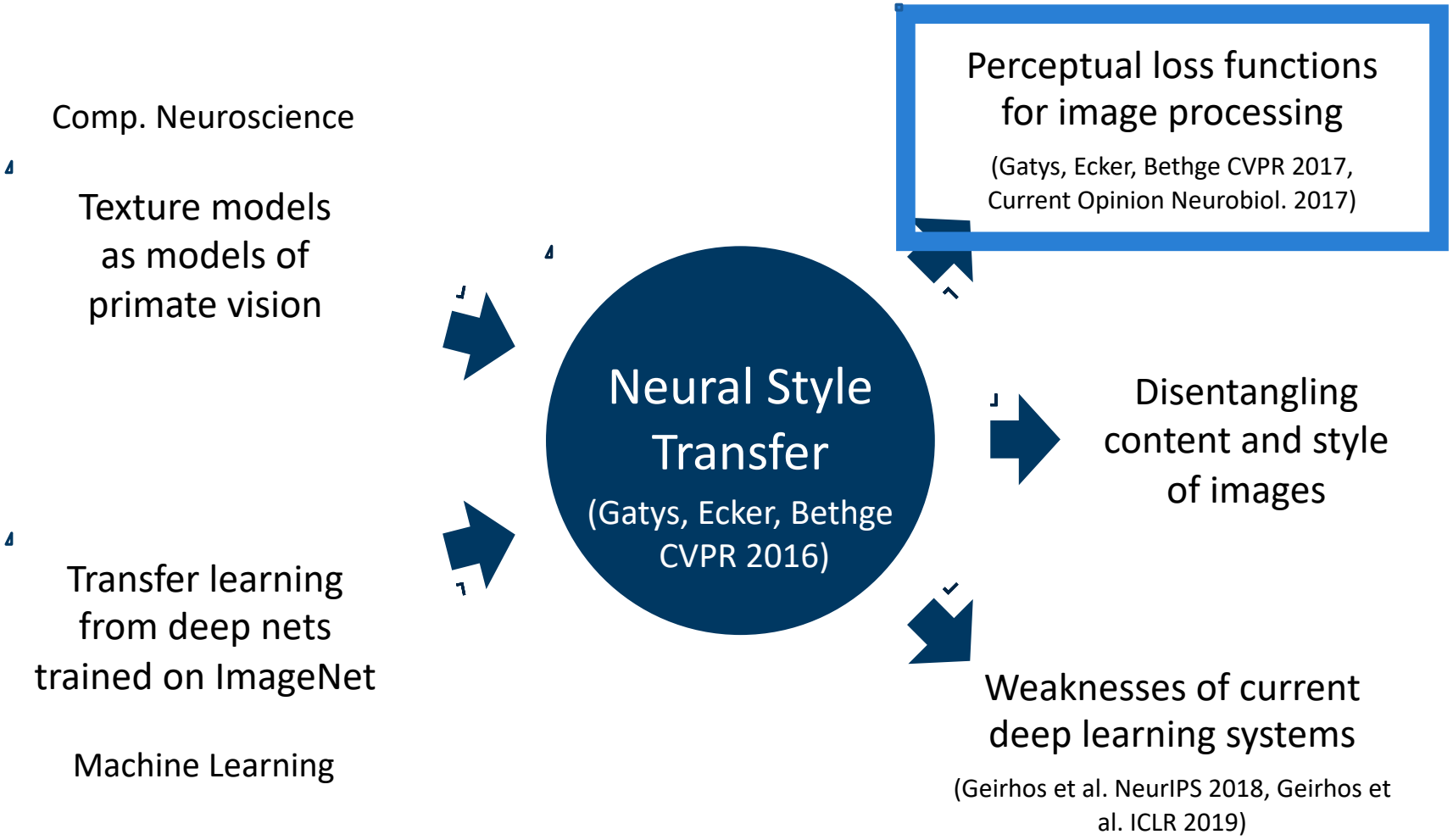
3 Submit

Our servers paint the image for you. You get an email when it's done.



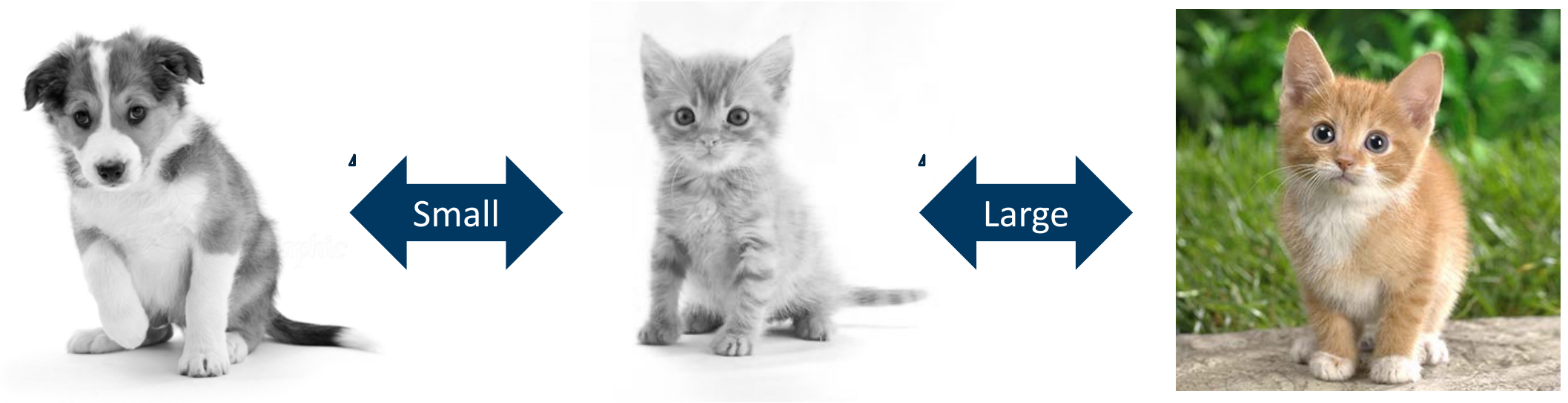


Significance of neural style transfer



Perceptual loss functions

Euclidean distance in pixel space does not cover semantics of images



Perceptual loss functions

Euclidean distance in pixel space does not cover perceptual differences

Blurred

Small

Original

Large

Shifted by 20 pixels

Compute distances in high-level feature space of deep neural network trained on ImageNet

Neural style transfer implementations

Leon Gatys' PyTorch implementation:
https://github.com/leongatys/PytorchNeuralStyleTransfer

Justin Johnson's excellent and very popular Lua/Torch implementation:
(lots of features, but may be hard to run these days)
https://github.com/jcjohnson/neural-style

Neural Style in official PyTorch tutorials:
https://pytorch.org/tutorials/advanced/neural_style_tutorial.html

Questions!