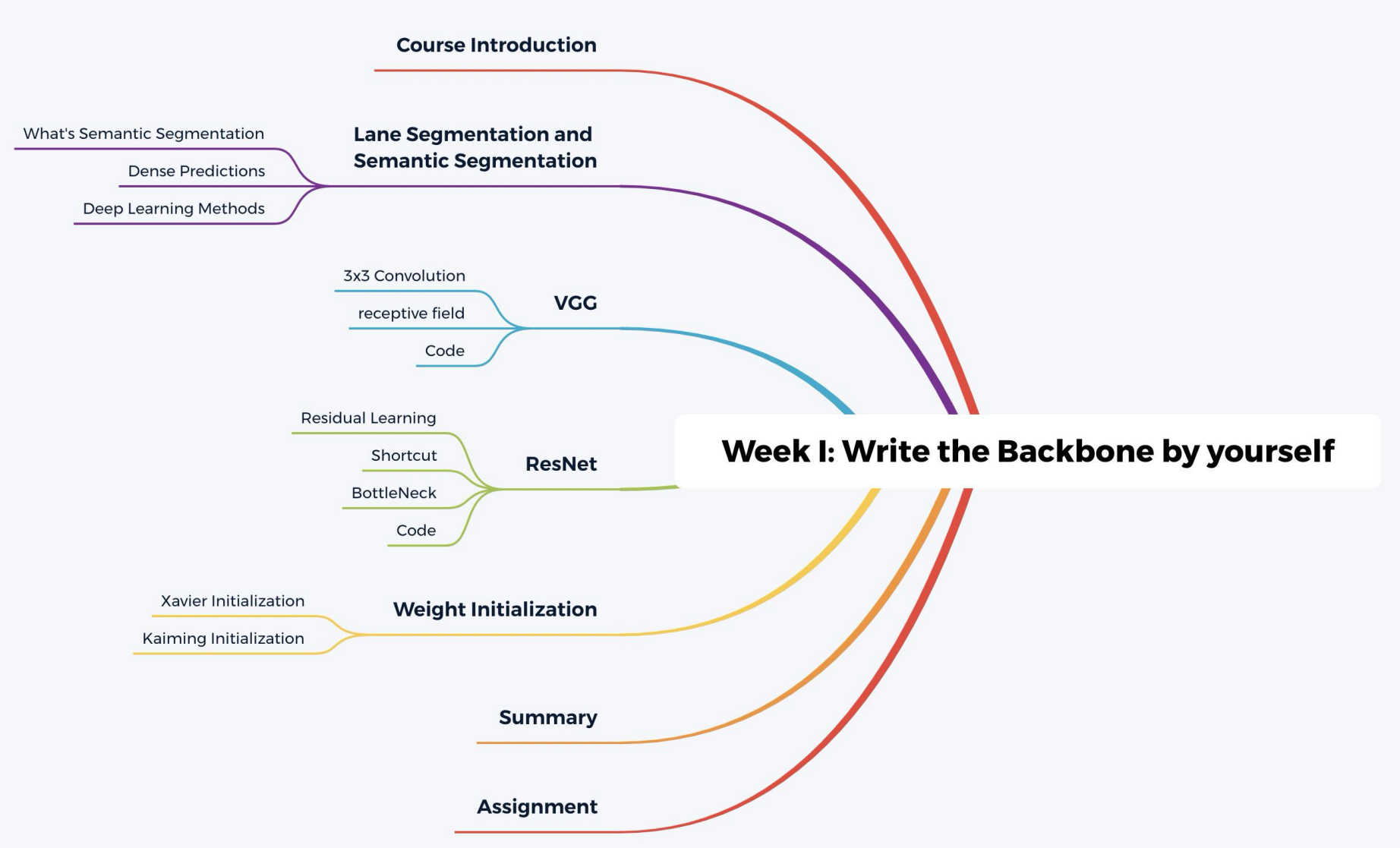
后厂理工学院
houchangtech.com

# Lane Segmentation Week 1

HCT CV Class

# 主要内容





# 学习目标

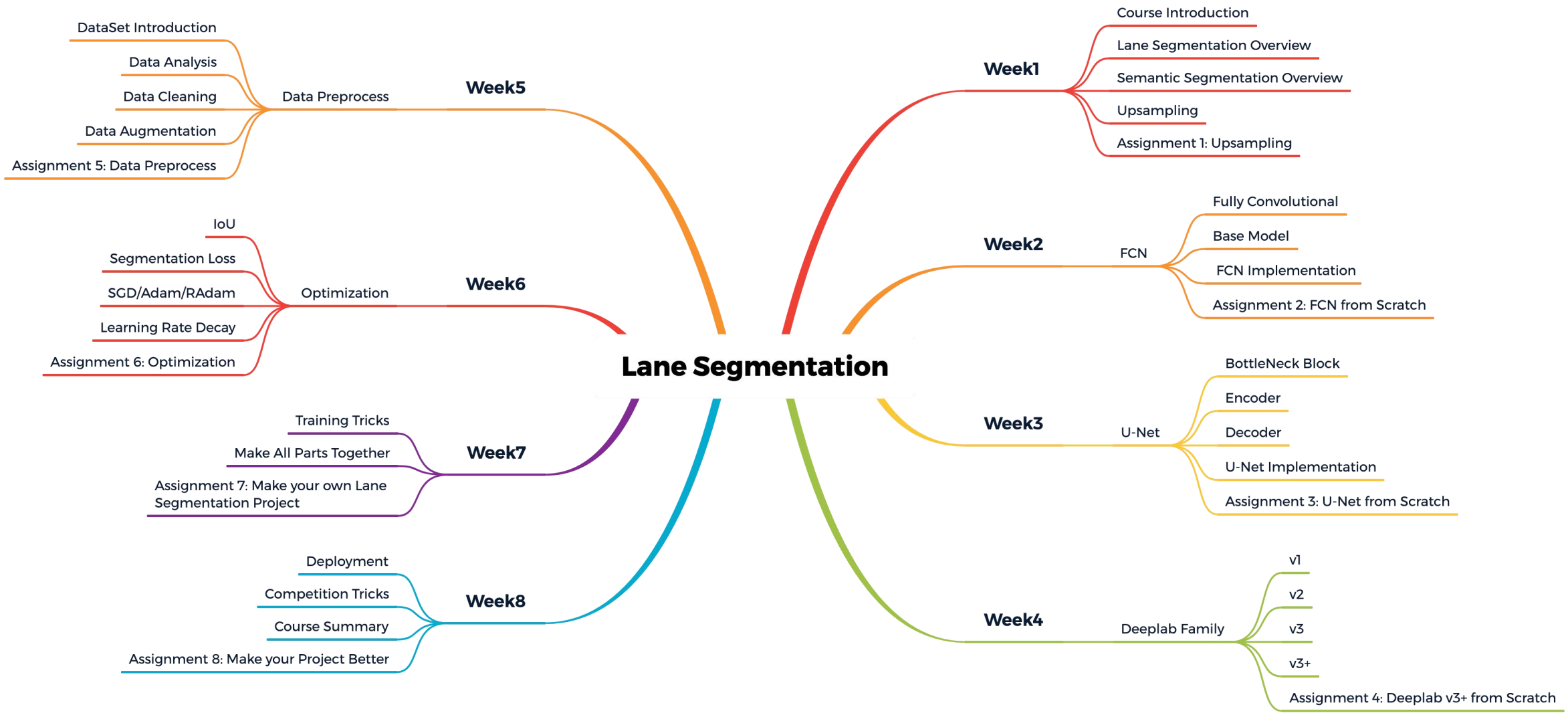
- 了解Semantic Segmentation
- 掌握VGG
- 掌握ResNet
- 掌握Weight Initialization

# 课程介绍

- 详细讲解Lane Segmentation的原理和代码实现。通过课程学习可以掌握Semantic Segmentation的原理和经典模型的构建，掌握百度无人驾驶车道线数据集的预处理方法和优化训练技巧，不仅可以达到百度车道线挑战赛一流成绩，而且可以广泛应用于各种图像语义分割任务，对图像识别和目标检测等其他深度学习计算机视觉任务的水平也将会有很大提高。

# Syllabus





# 预备知识

- Python
- Machine Learning
- CNN and Image Classification
- Tensorflow or Pytorch or PaddlePaddle...
- English
- Math

# Project



Bai大脑 | AI Studio　项目　数据集　比赛　课程　社区　教育合作　文档　访问飞桨官网　中 | En　登录

## 无人车车道线检测挑战赛 结束

奖池：￥99000

报名人数: 743

本次PaddlePaddle-无人车车道线检测挑战赛旨在为参赛者提供一定数量的准确的车道线标注数据，让更多的研究者参与并设计出高效、准确的检测算法，以此来共同推动无人车的发展，从而造福整个社会。

已结束

比赛介绍　赛题说明　提交结果　我的团队　排行榜　讨论区

### 竞赛简介

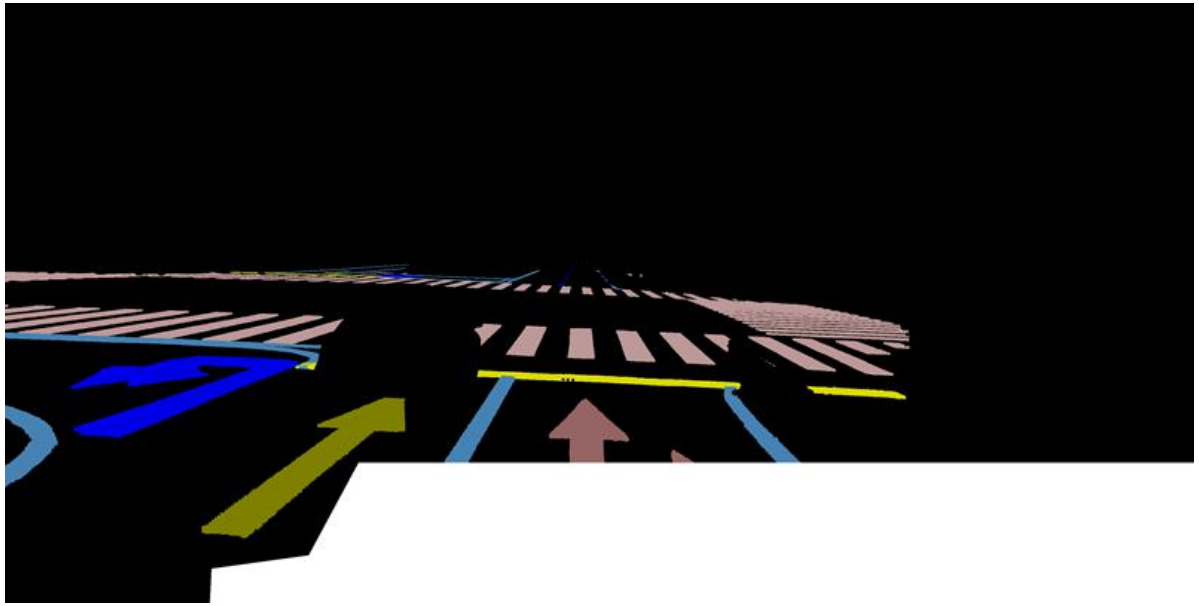
作为下一个改变人类生活方式的技术蓝海，无人驾驶车辆研究日益受到学术界和产业界的高度关注，其技术涉及认知科学、人工智能、控制科学、机械工程等交叉学科，是各种新兴技术的最佳验证平台，也是未来汽车发展的必然趋势。在无人驾驶汽车研究领域，带有车道线属性的高精地图是商业无人驾驶的一个非常关键的环节。截止目前，大多数的高精地图都是靠人工标注来完成的。人工标注不仅效率低，而且成本高不适宜大规模的商业应用。在高精地图的生产过程中，如何从图像中分割出不同类型的车道线是非常重要的一个环节。同时，准确分割车道线也会为将来高精地图的更新提供帮助。本次无人车车道线检测挑战赛旨在为参赛者提供一定数量的准确的车道线标注数据，让更多的研究者参与并设计出高效、准确的检测算法，以此来共同推动无人车的发展，从而造福整个社会。

### 赛程赛制

竞赛分为初赛、复赛两个阶段，各阶段参赛队伍必须按照要求按时、合规地提交参赛作品。

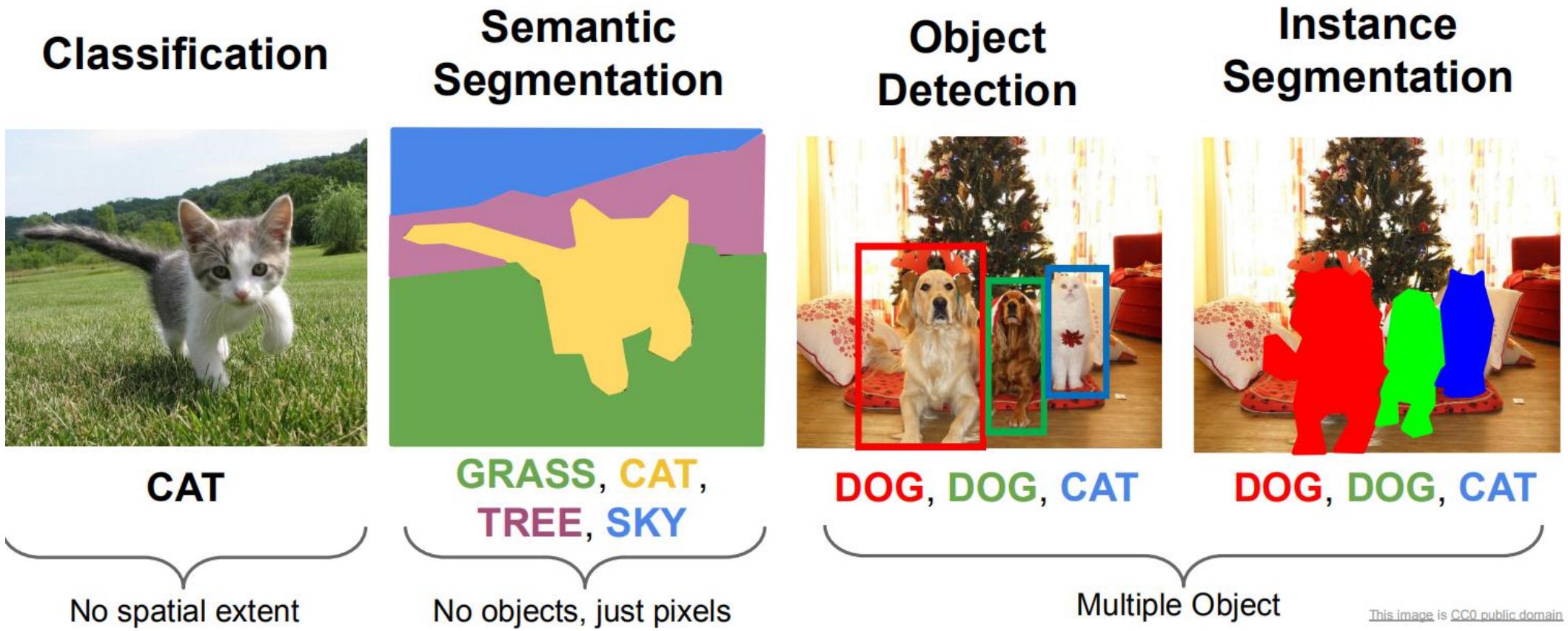
https://aistudio.baidu.com/aistudio/competition/detail/5

# Lane Segmentation



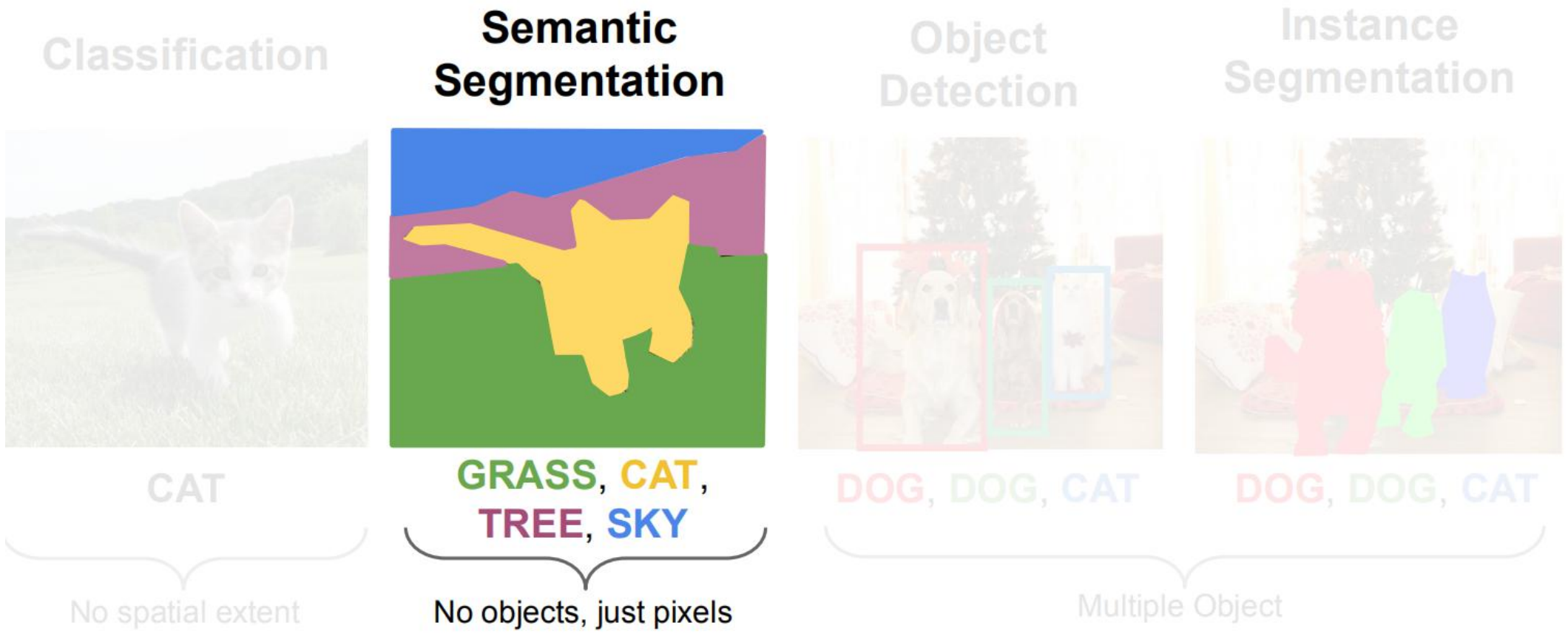
https://aistudio.baidu.com/aistudio/competition/detail/5

# CV Tasks




From Stanford CS231n

# Semantic Segmentation




From Stanford CS231n

# Semantic Segmentation

- Label each pixel in the image with a category label
- Don't differentiate instances, only care about pixels
- We can think of semantic segmentation as image classification at a pixel level

# Semantic Labels

Input

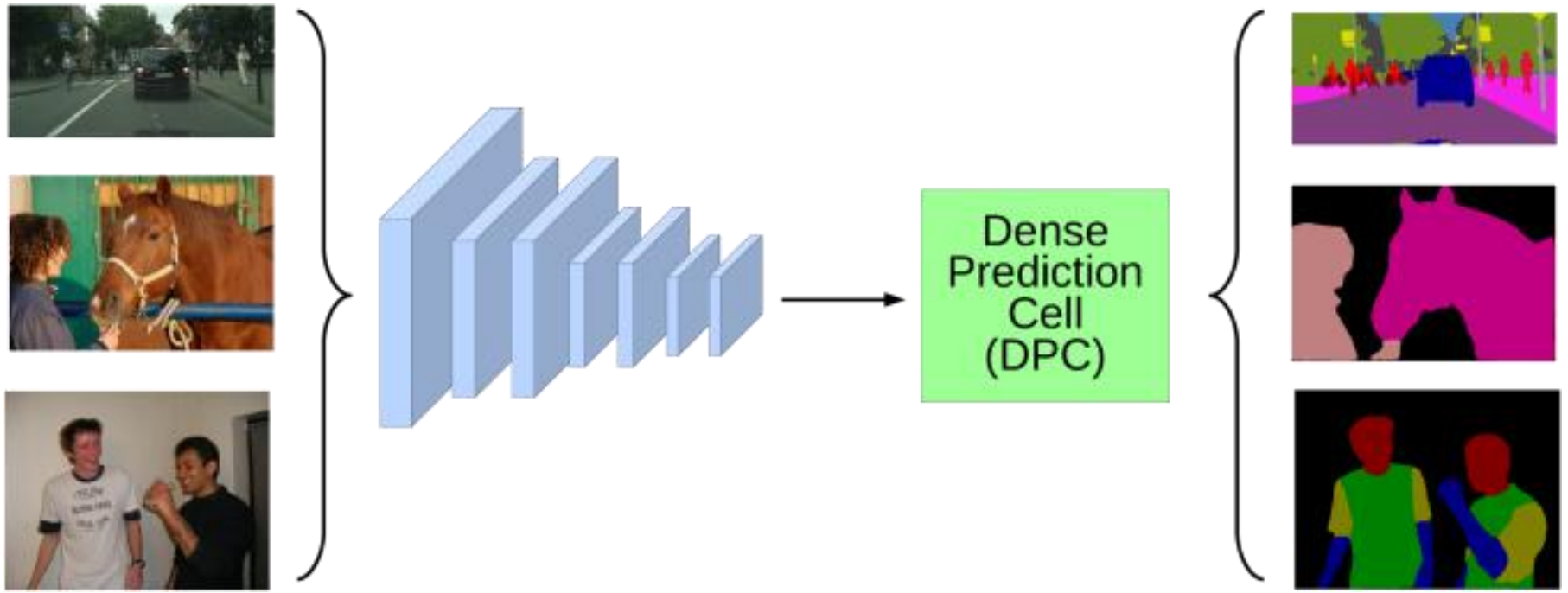
segmented

1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

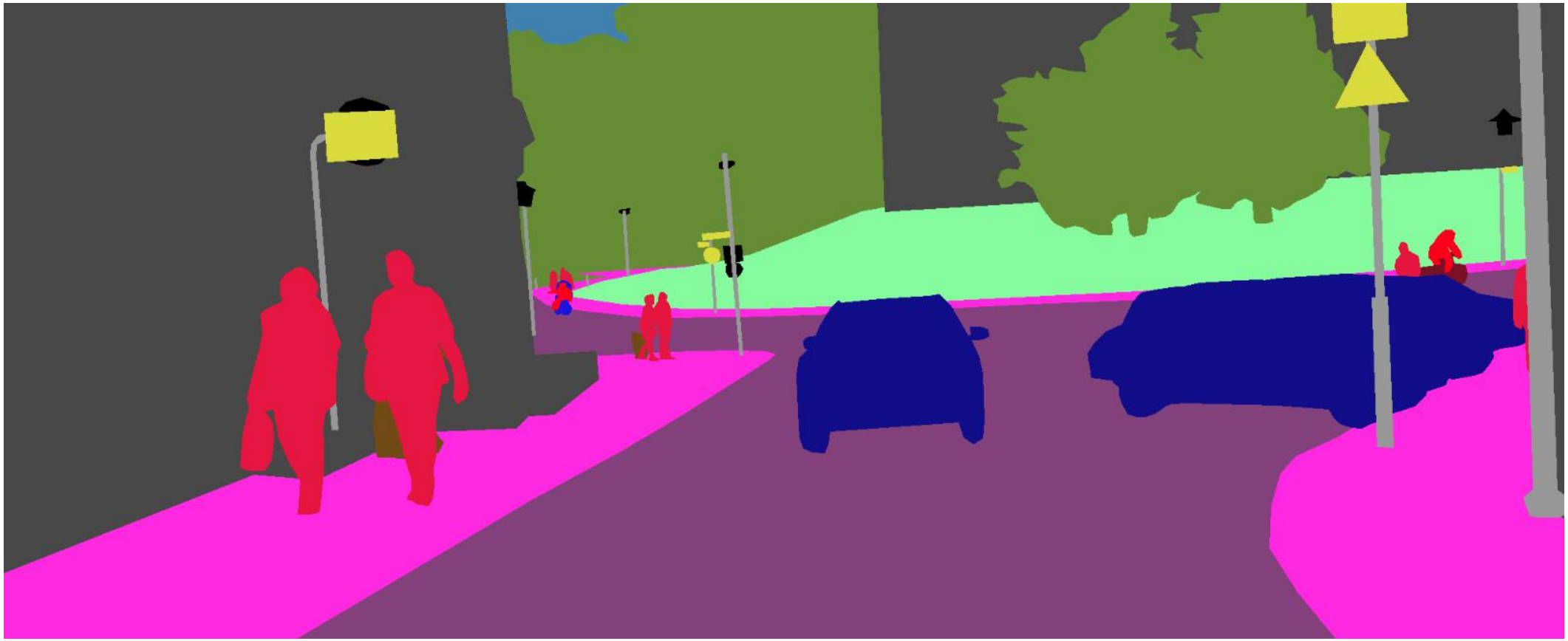
```
3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 1 1 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 1 1 1 1 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 1 1 3 3 3 5 5 5 5 5 5 5
5 5 3 3 3 3 1 1 3 3 5 5 5 5 5 5 5 5
4 4 3 4 1 1 1 1 1 1 4 4 4 5 5 5 5 5
4 4 3 4 1 1 1 1 1 1 4 4 4 4 4 5 5 5
4 4 4 1 1 1 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 1 1 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 2 2 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 2 2 1 1 1 1 1 4 4 4 4 4 4 4
```

Semantic Labels

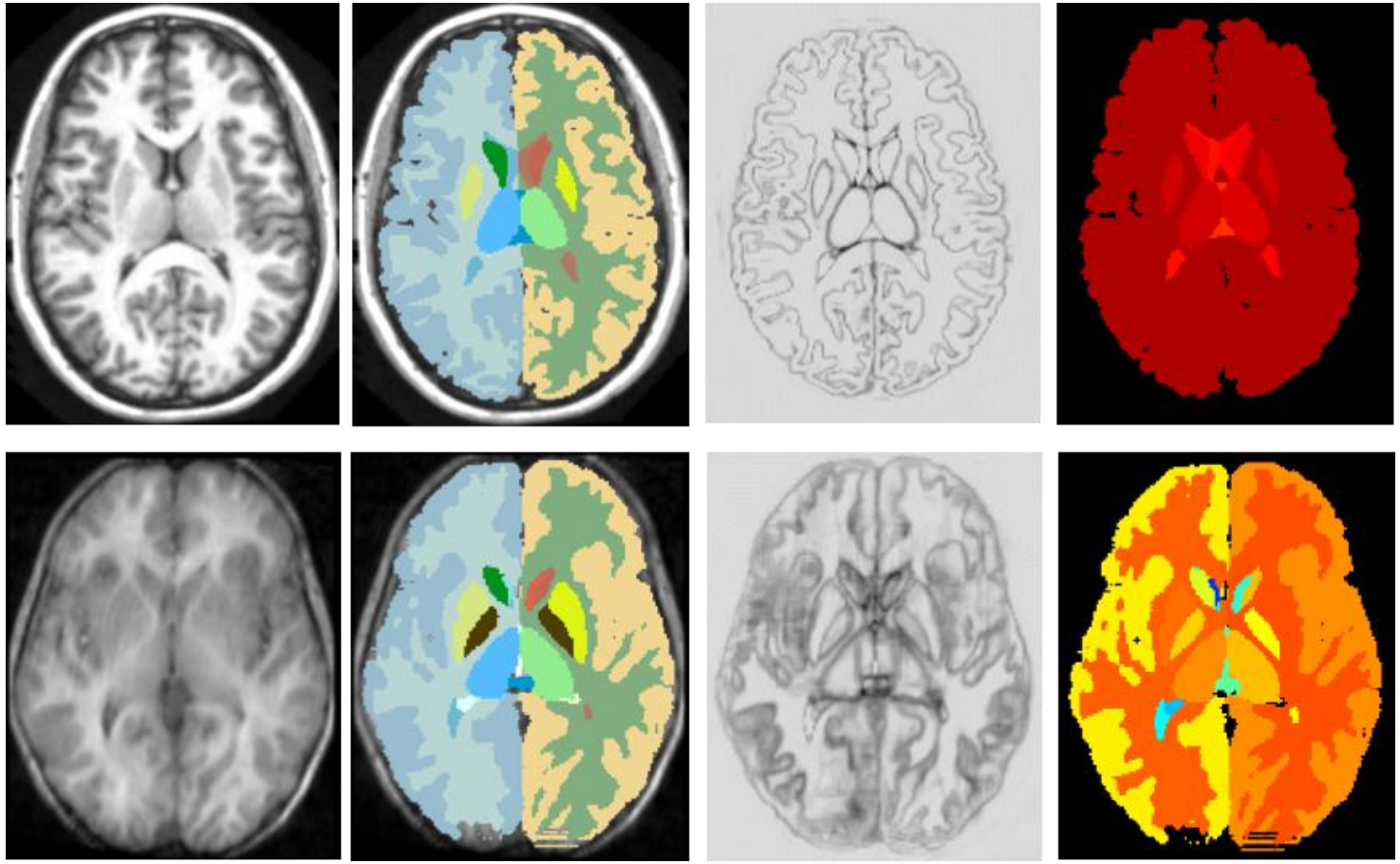
# Dense Predictions



# Auto Driving



# medical image segmentation

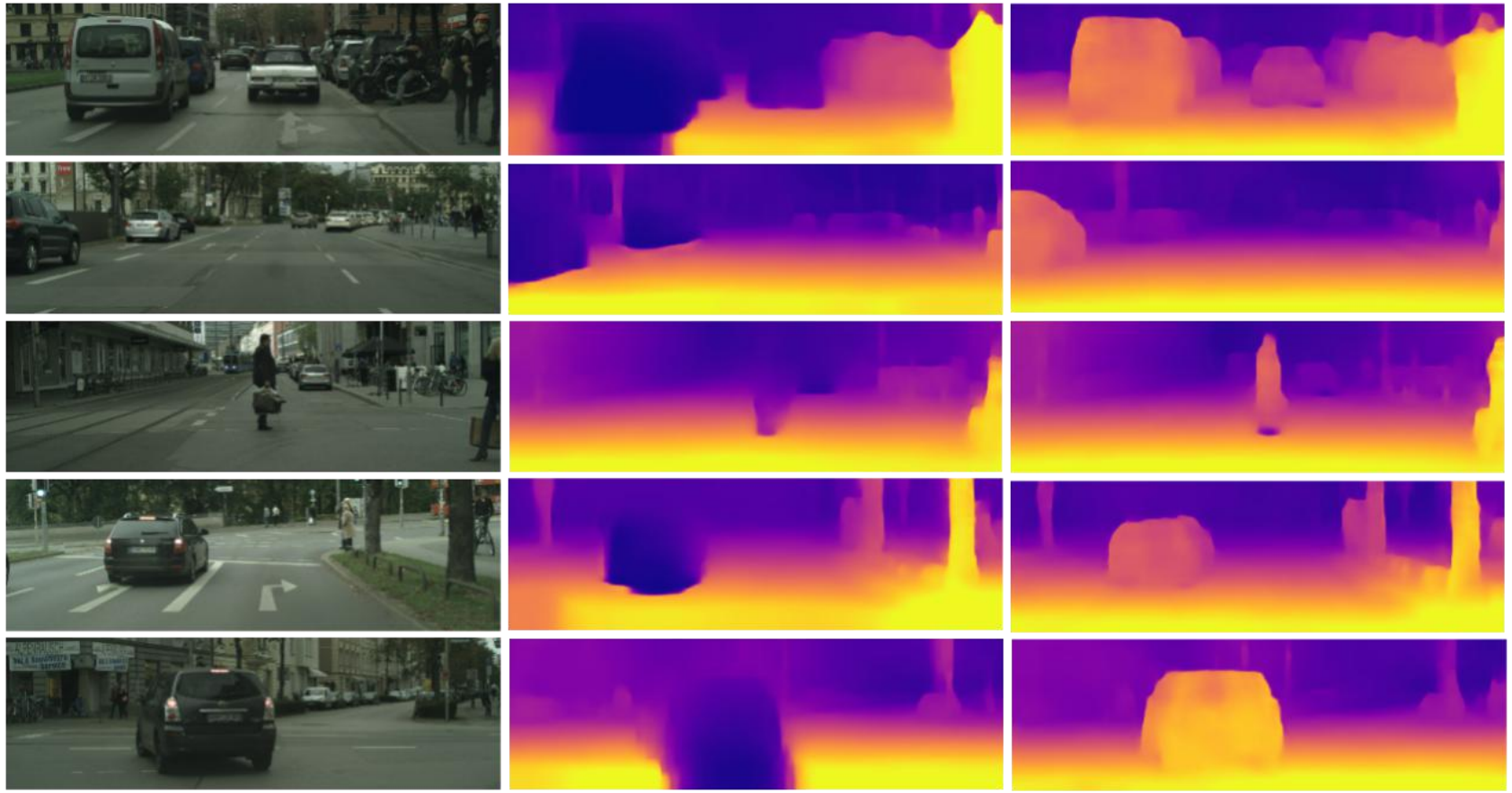


# shallow depth-of-field image

# Human Pose Estimation

# Bodypix

# Depth Predictions



# FCOS



## FCOS: Fully Convolutional One-Stage Object Detection

Zhi Tian Chunhua Shen* Hao Chen Tong He
The University of Adelaide, Australia



### Abstract

*We propose a fully convolutional one-stage object detector (FCOS) to solve object detection in a per-pixel prediction fashion, analogue to semantic segmentation. Almost all state-of-the-art object detectors such as RetinaNet, SSD, YOLOv3, and Faster R-CNN rely on pre-defined anchor boxes. In contrast, our proposed detector FCOS is anchor-box free, as well as proposal free. By eliminating the pre-defined set of anchor boxes, FCOS completely avoids the complicated computation related to anchor boxes such as calculating overlapping during training and significantly reduces the training memory footprint. More importantly, we also avoid all hyper-parameters related to anchor boxes, which are often very sensitive to the final detection perfor-*

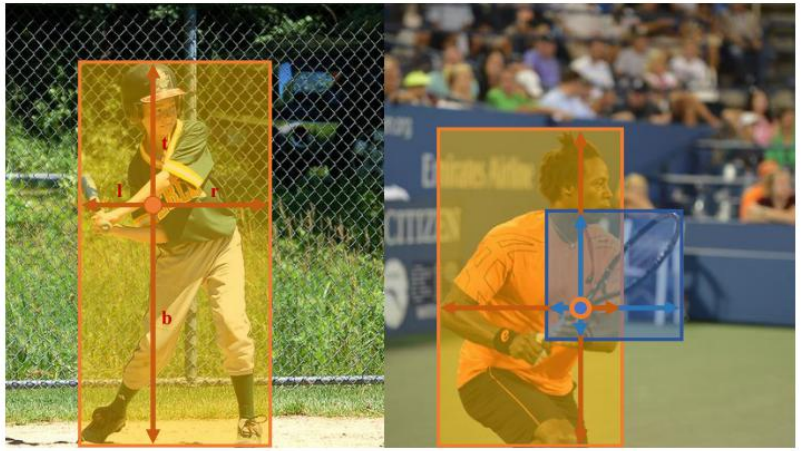
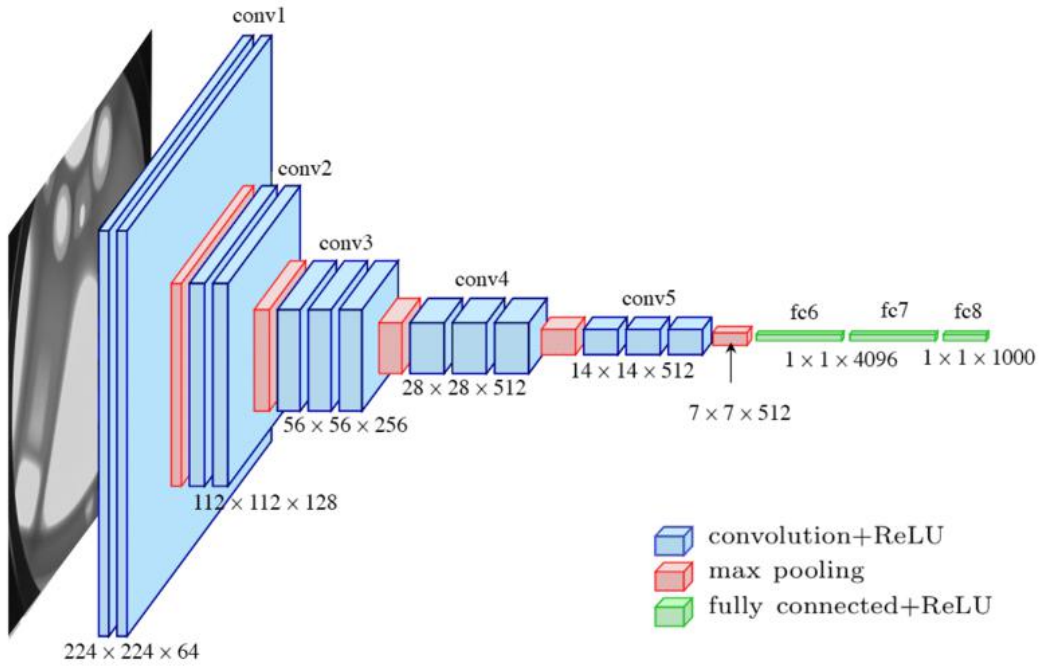


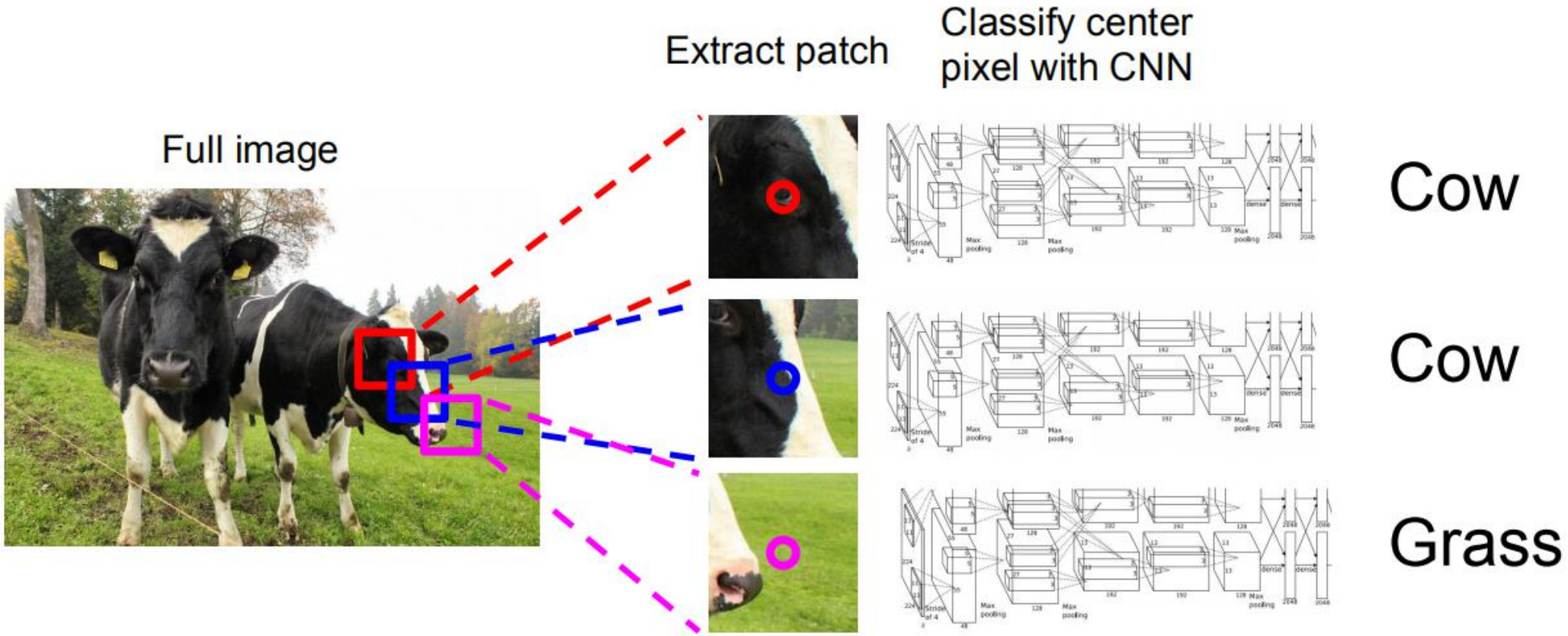

**Figure 1** – As shown in the left image, FCOS works by predicting a 4D vector $(l, t, r, b)$ encoding the location of a bounding box at each foreground pixel (supervised by ground-truth bounding box information during training). The right plot shows

# Dense Predictions

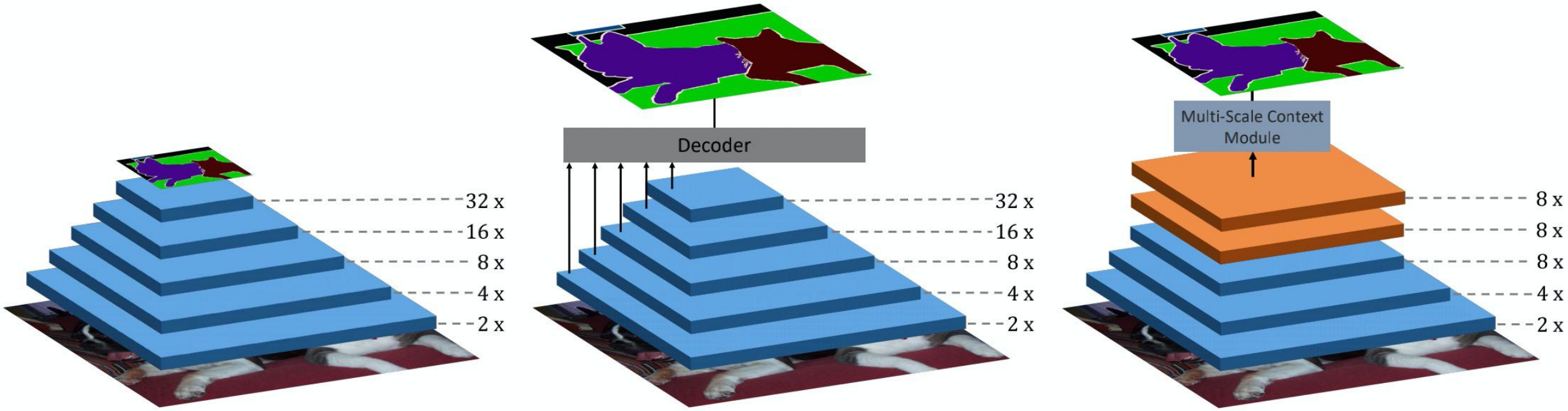
- How To do Dense Predictions from Classification?



# Sliding Window



# Semantic Segmentation




(a) FCN [22] (b) EncoderDecoder (c) DilatedFCN

# HRNet

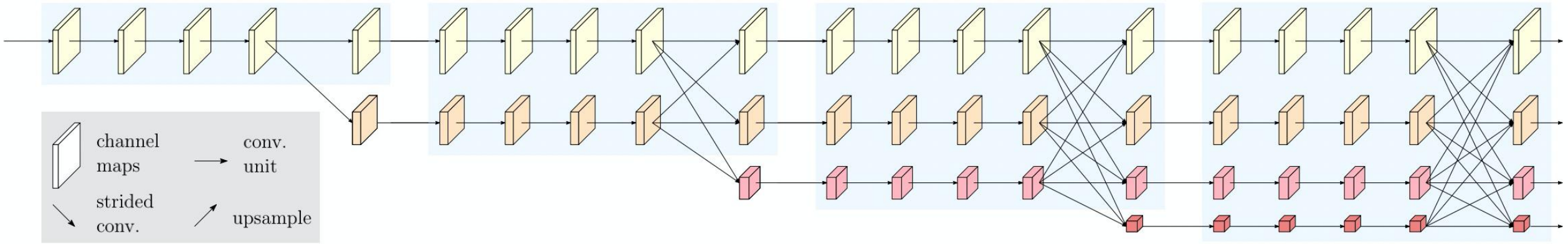
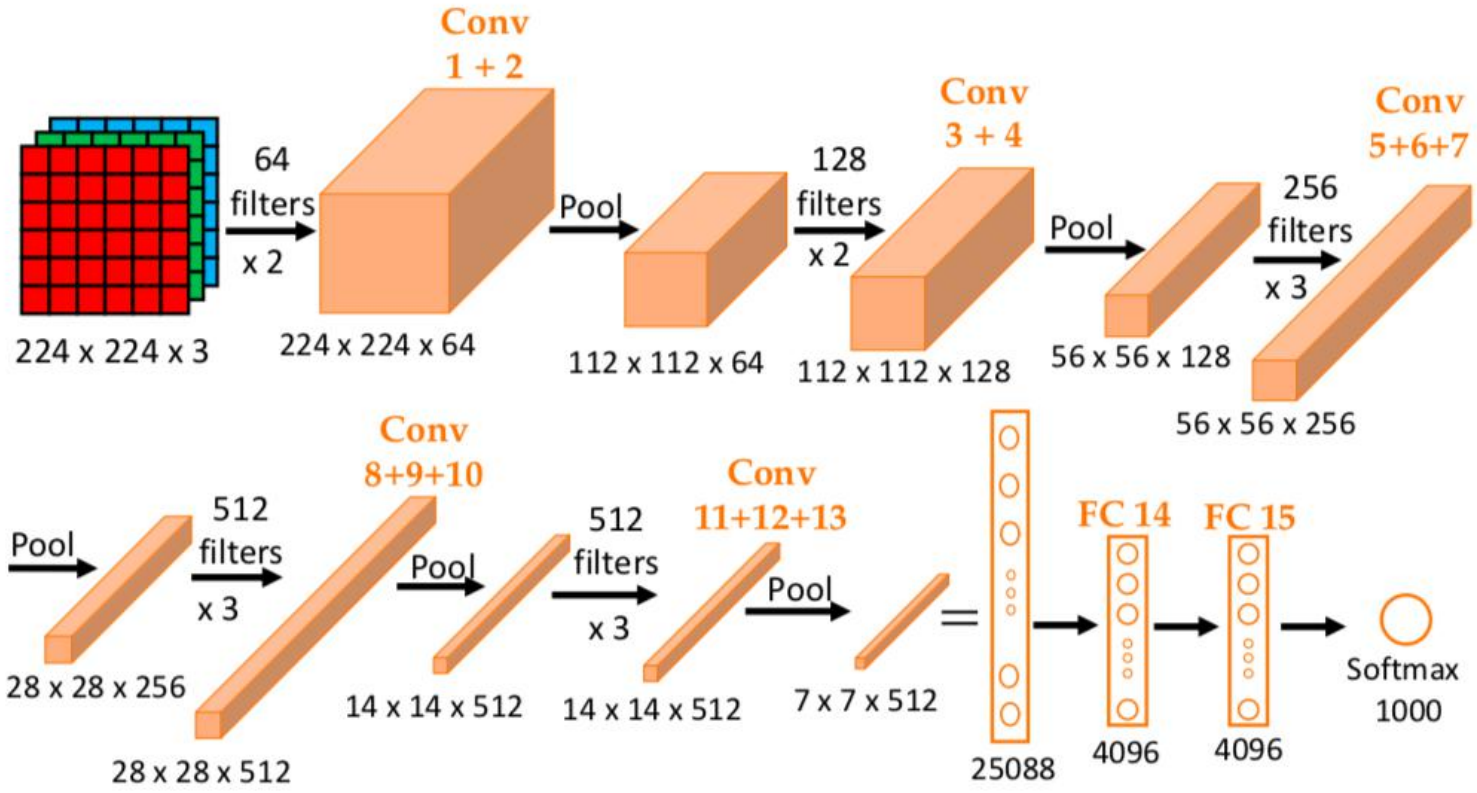



Fig. 2. An example of a high-resolution network. There are four stages. The 1st stage consists of high-resolution convolutions. The 2nd (3rd, 4th) stage repeats two-resolution (three-resolution, four-resolution) blocks. The detail is given in Section 3.

# Note

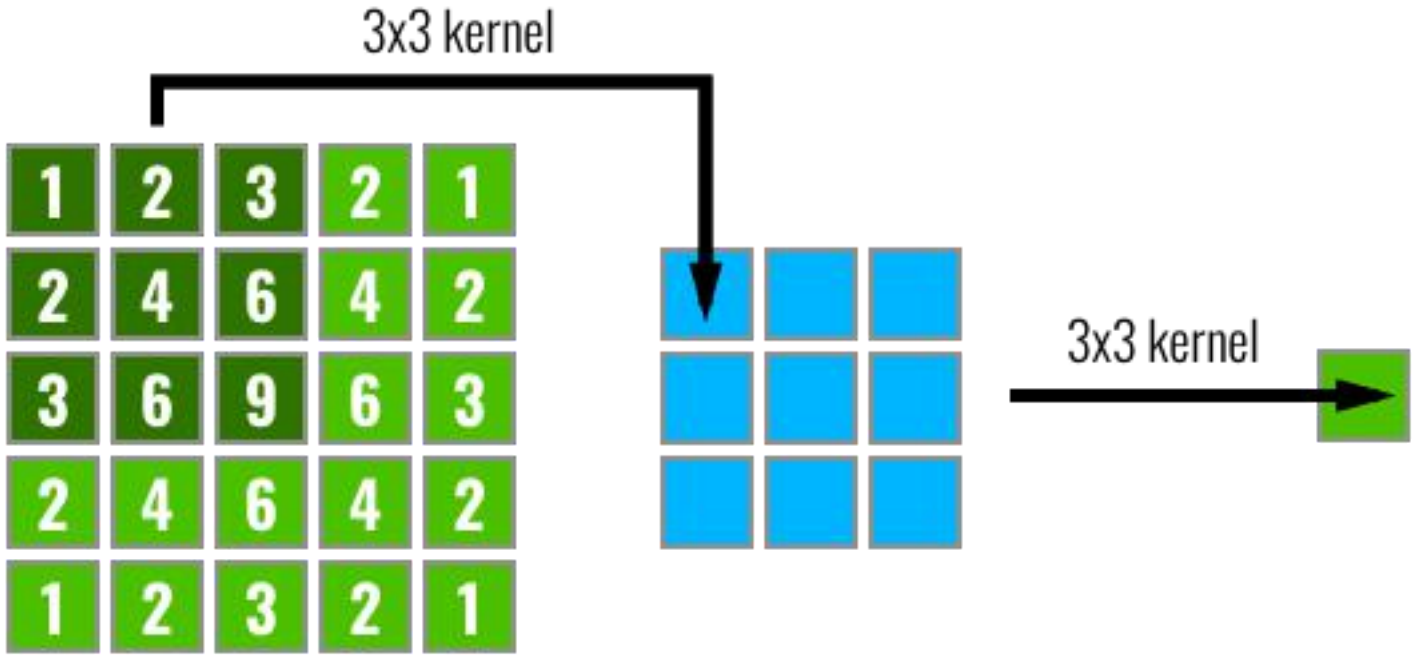
# VGG



# 3x3 Convolution

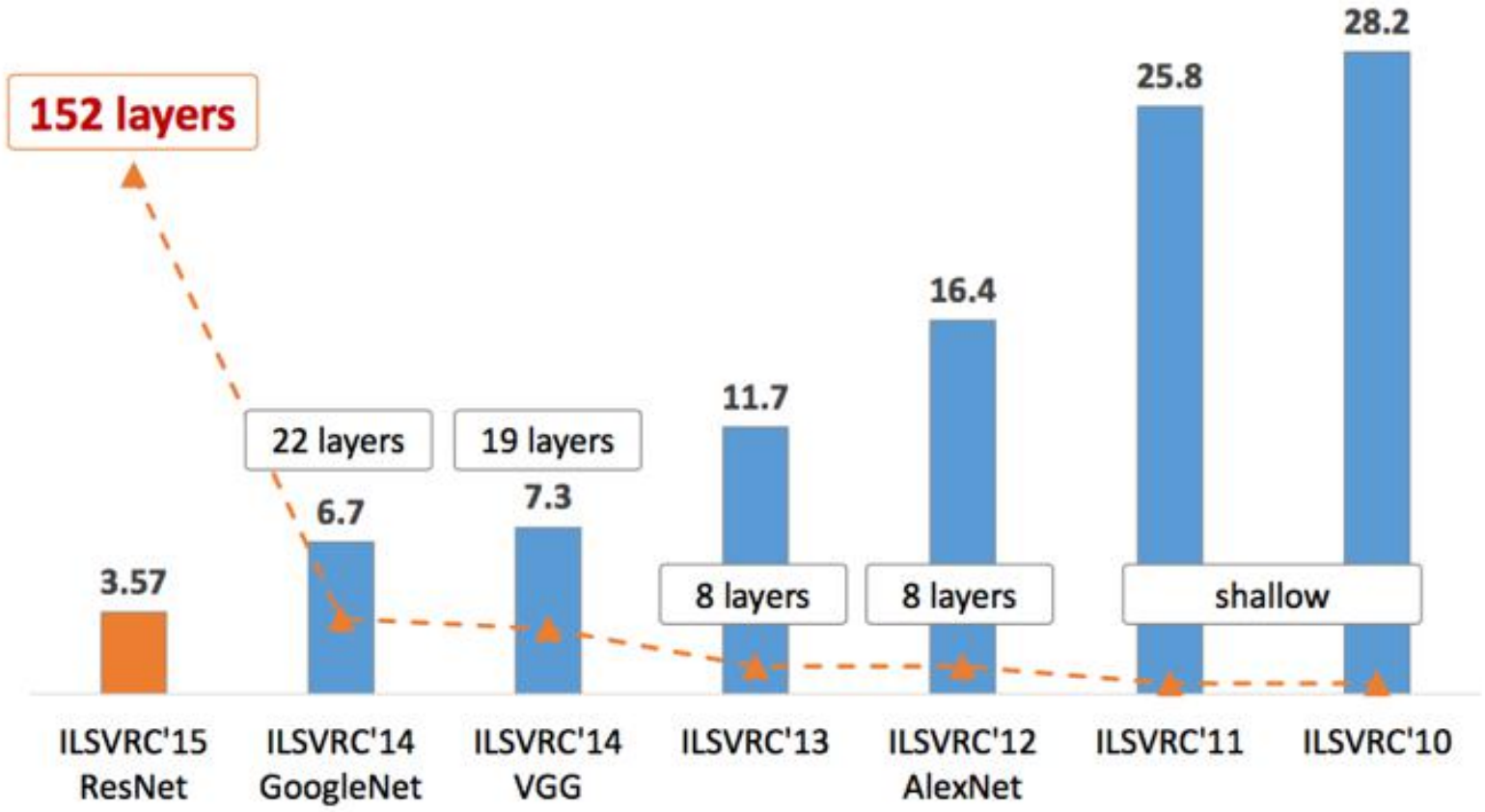
- the smallest size to capture the notion of left/right, up/down, center

# receptive field

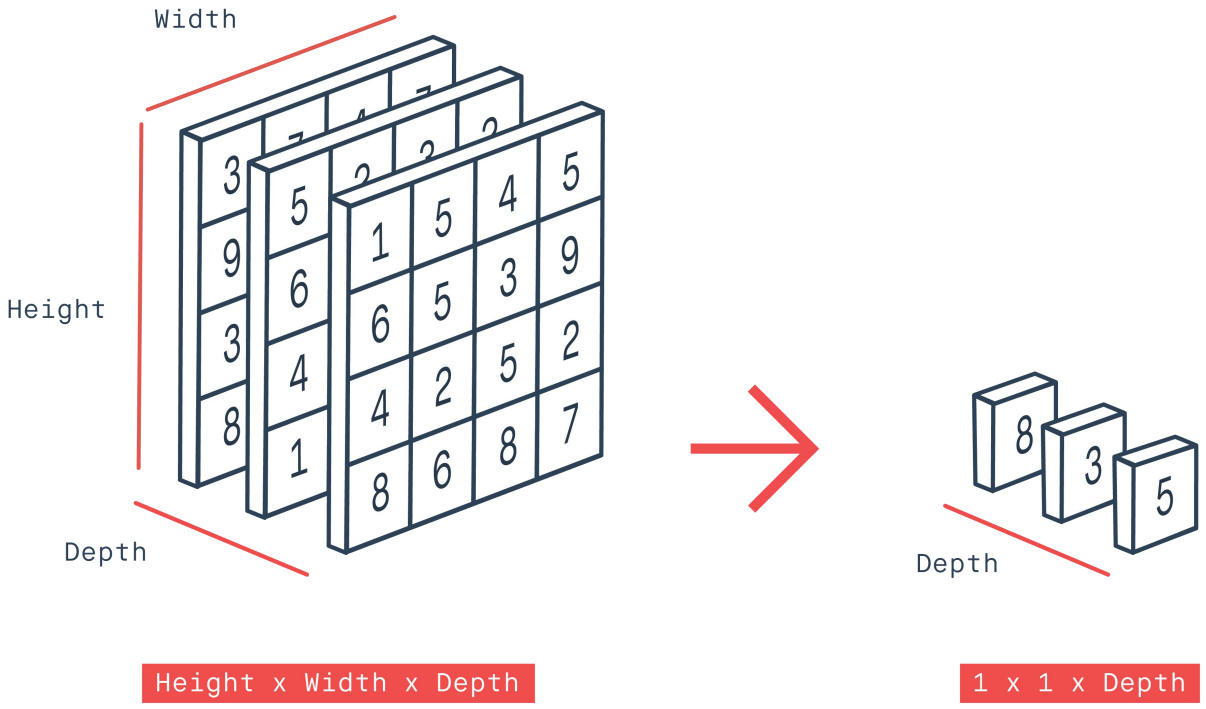


# VGG

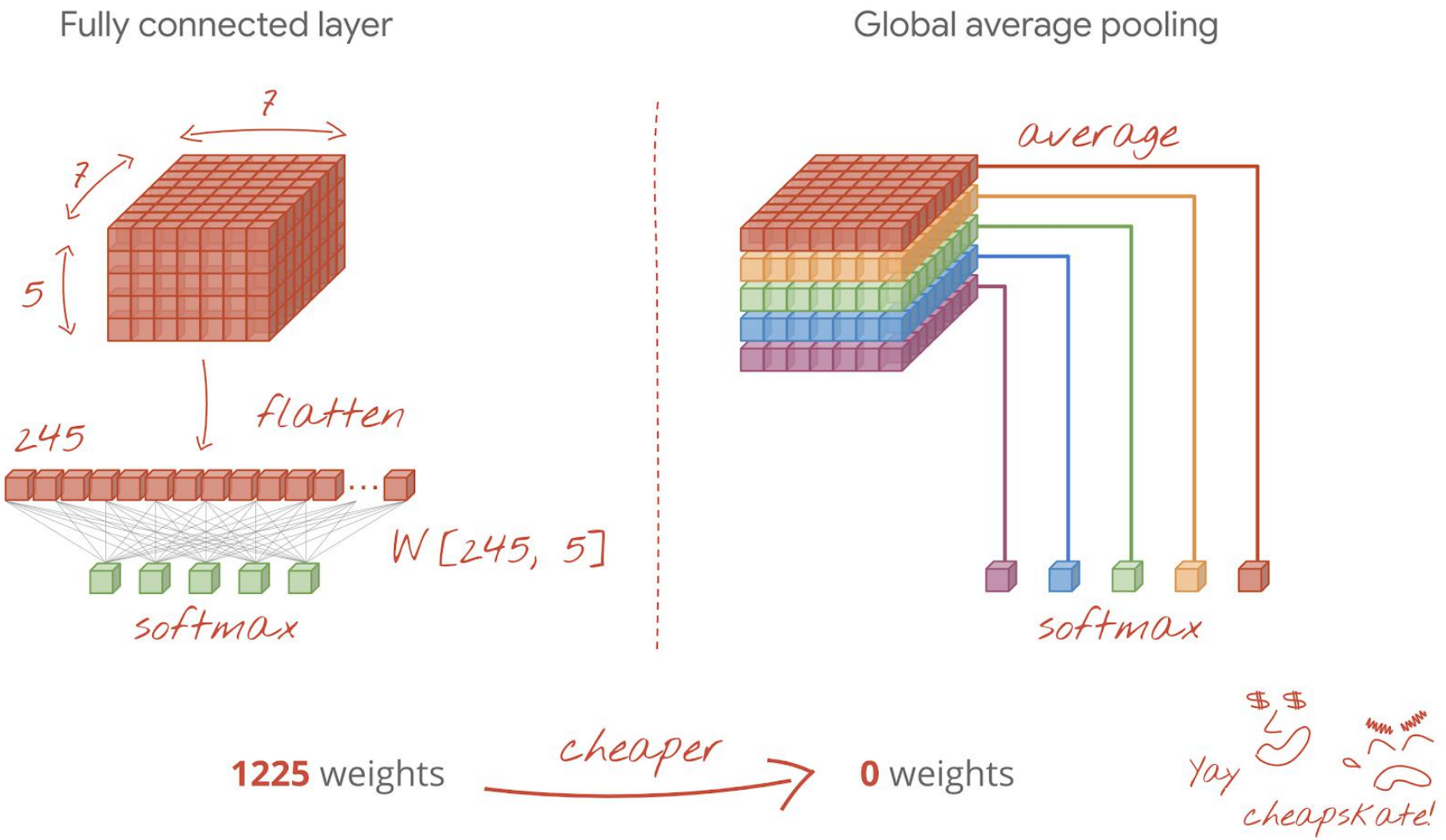
# ResNet



# Global Average Pooling

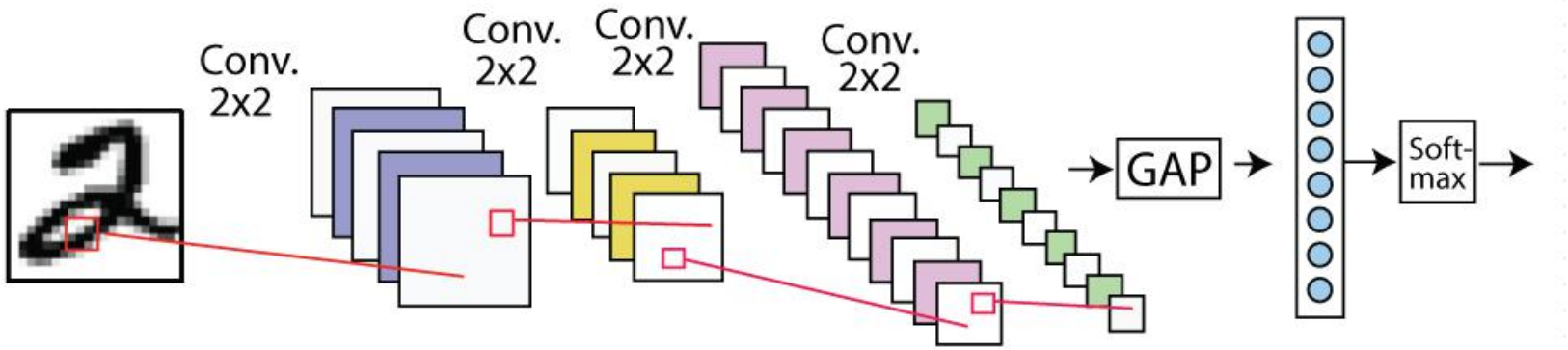


# Global Average Pooling

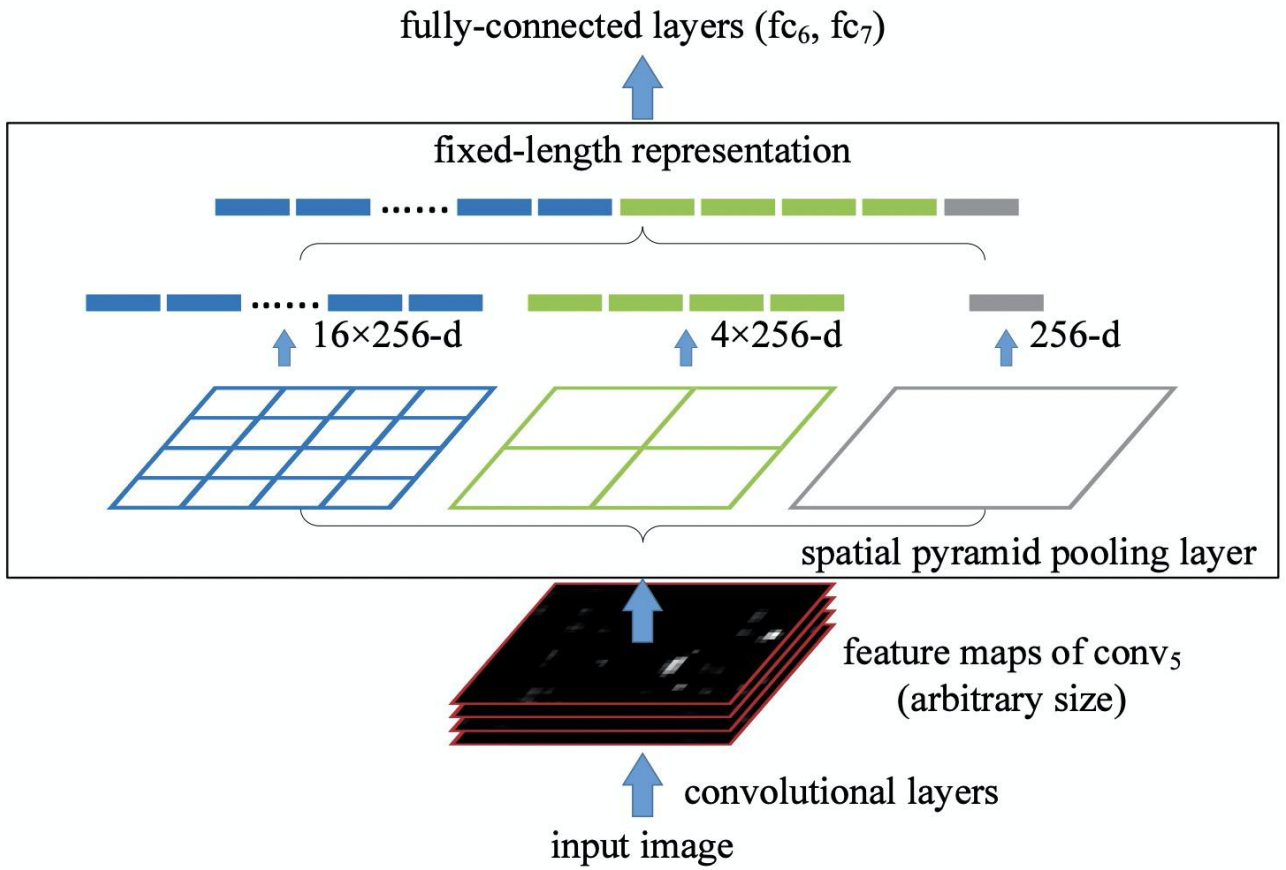


# Global Average Pooling

- torch.nn.AdaptiveAvgPool2d(output_size)
- tf.keras.layers.GlobalAvgPool2D

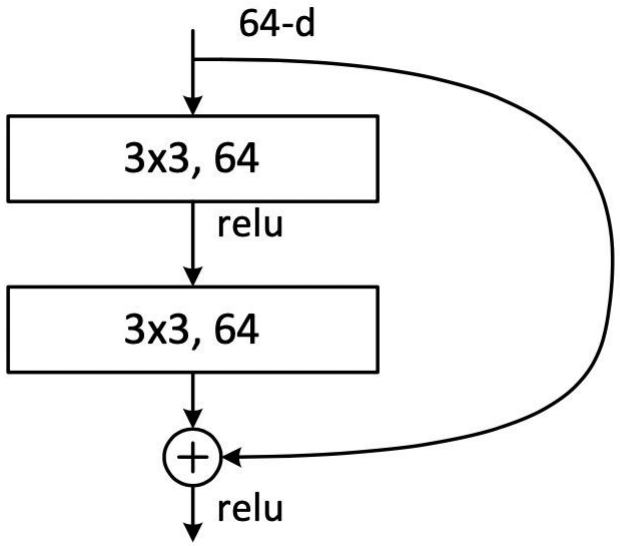


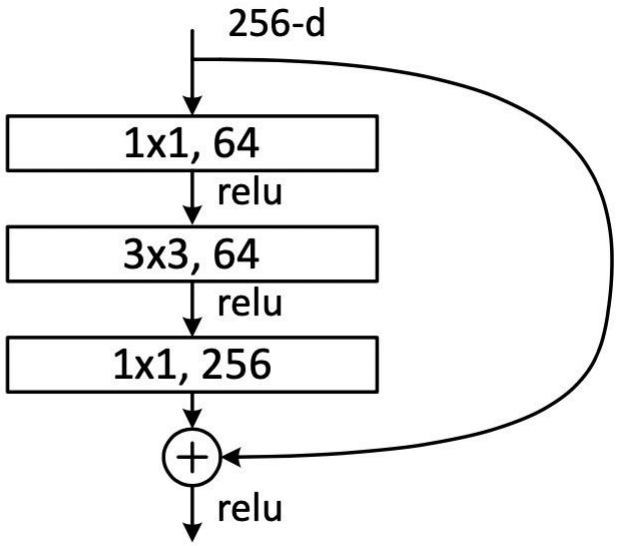
# Spatial Pyramid Pooling



# Notes

# BottleNeck



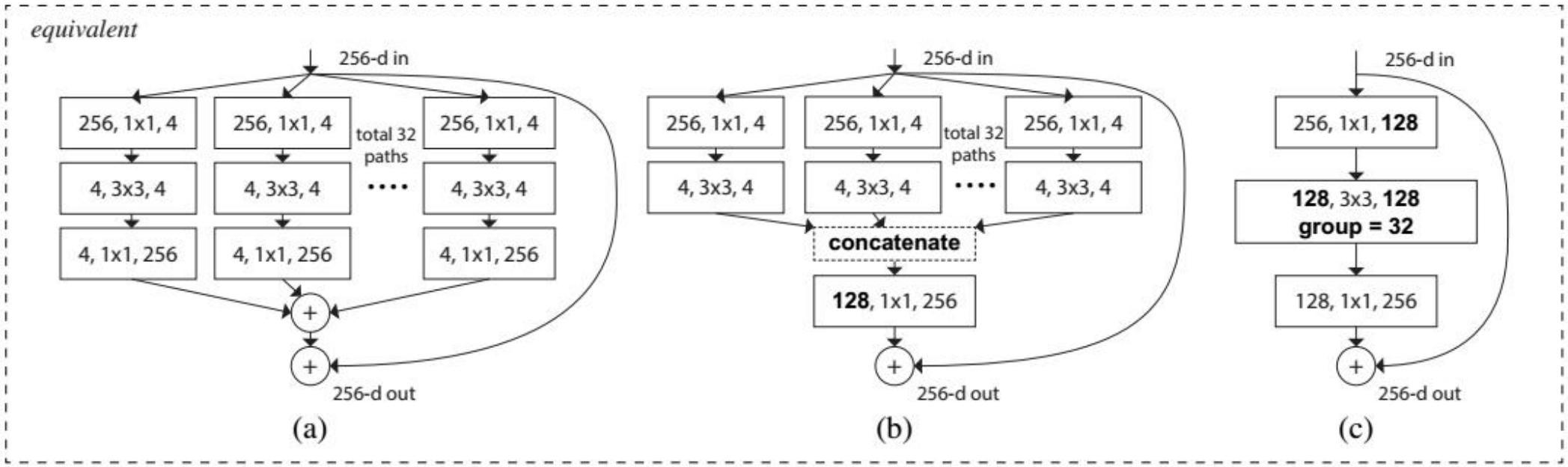



# Code

# ResNet新发展

- ResNeXt
- ResNeXt-Attention
- ResNeXt WSL

# ResNeXt



# SENet

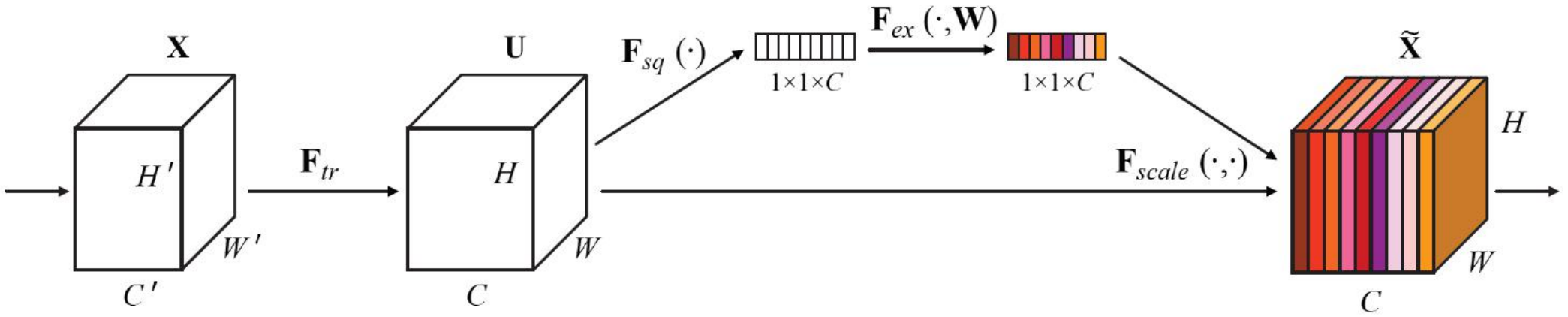
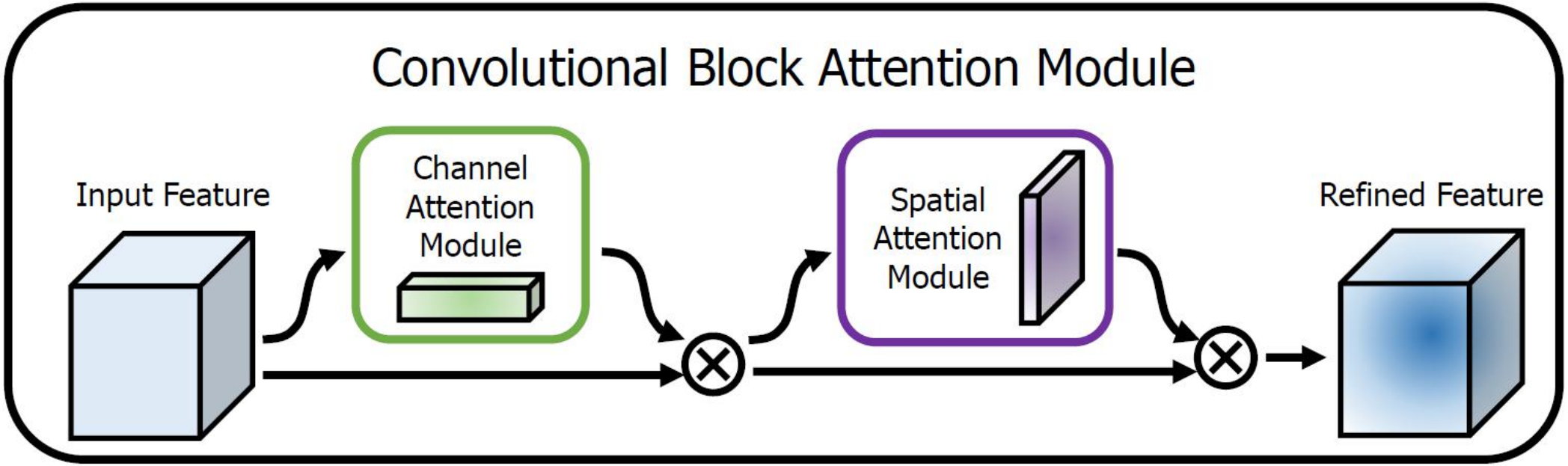



Figure 1: A Squeeze-and-Excitation block.

# ResNeXt-Attention



# ResNeXt WSL



View on Github › Open on Google Colab ›

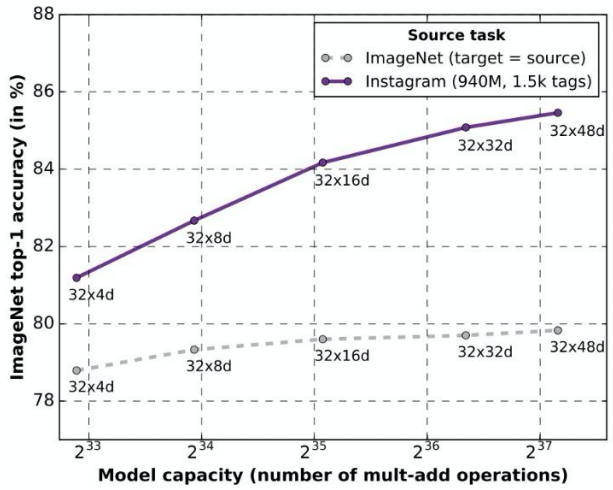



Fig. 5: Classification accuracy on val-IN-1k using ResNeXt-101 32×{4, 8 16, 32, 48}d with and without pretraining on the IG-940M-1.5k dataset.

```
import torch
model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x8d_wsl')
# or
# model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x16d_wsl')
# or
# model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x32d_wsl')
# or
#model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x48d_wsl')
model.eval()
```

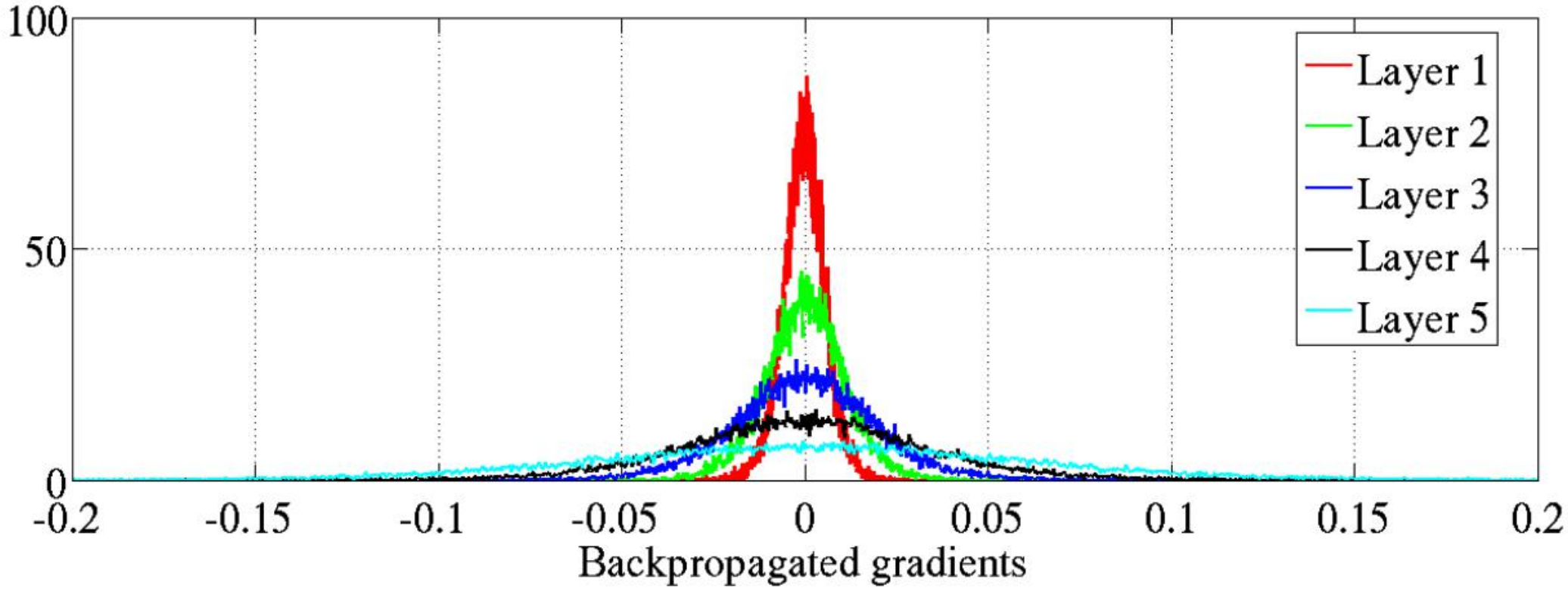
All pre-trained models expect input images normalized in the same way, i.e. mini-batches of 3-channel RGB images of shape `(3 x H x W)`, where `H` and `W` are expected to be at least `224`. The images have to be loaded in to a range of `[0, 1]` and then normalized using `mean = [0.485, 0.456, 0.406]` and `std = [0.229, 0.224, 0.225]`.

# Notes

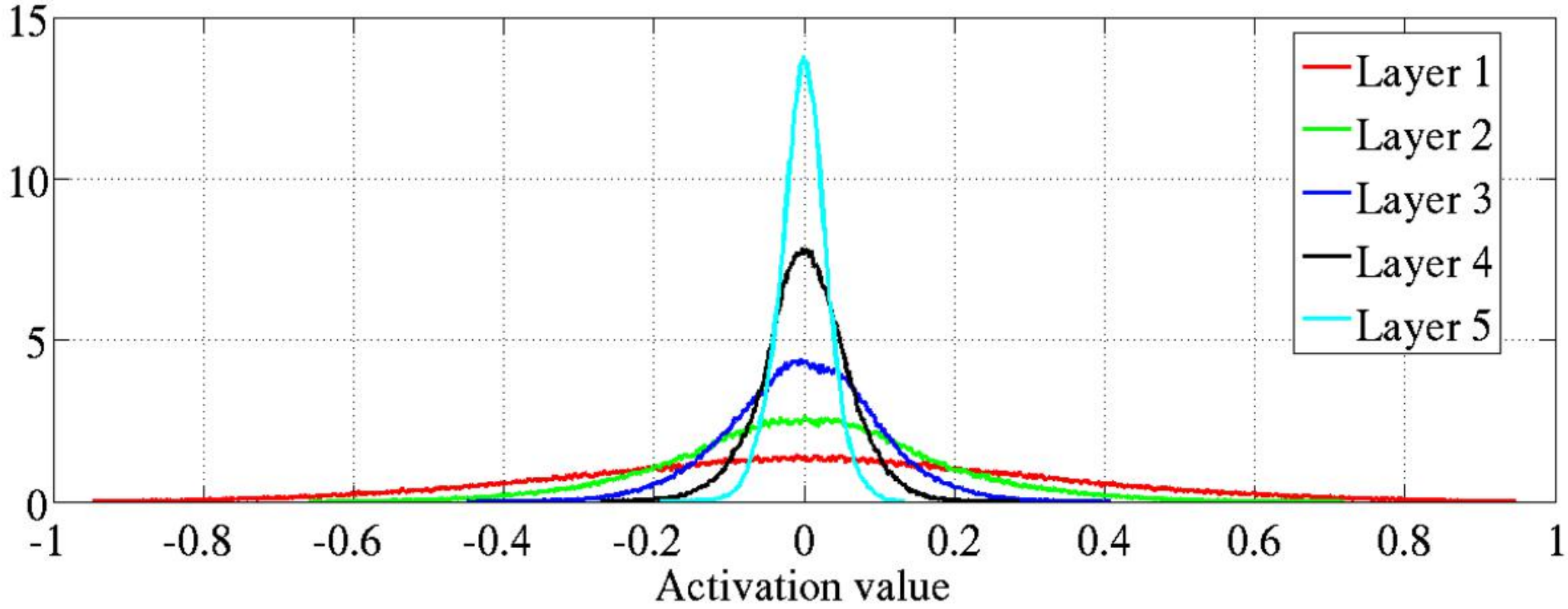
# Weight Initialization

- Initialization of neural networks isn’t something we think a lot about nowadays. It’s all hidden behind the different Deep Learning frameworks we use, like TensorFlow or PyTorch. However, it’s at the heart of why and how we can make neural networks as deep as they are today, and it was a significant bottleneck just a few years ago.

# Weight Initialization



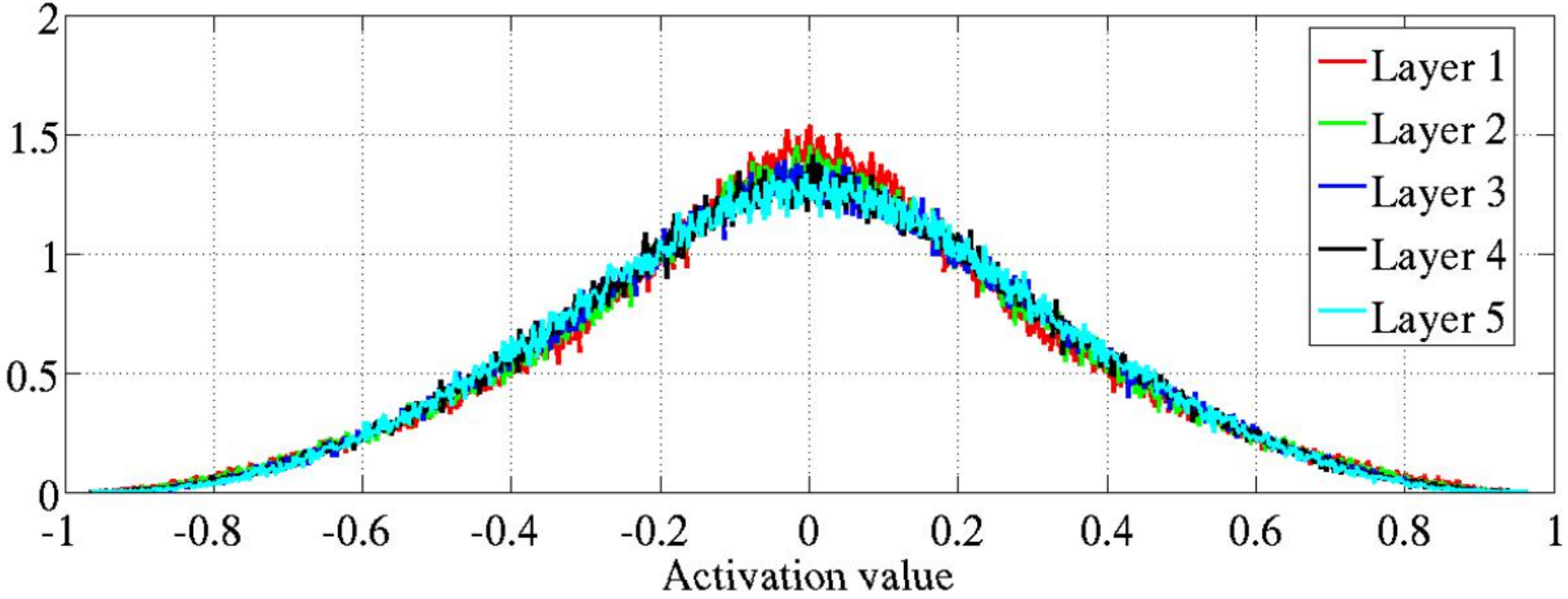
# Weight Initialization



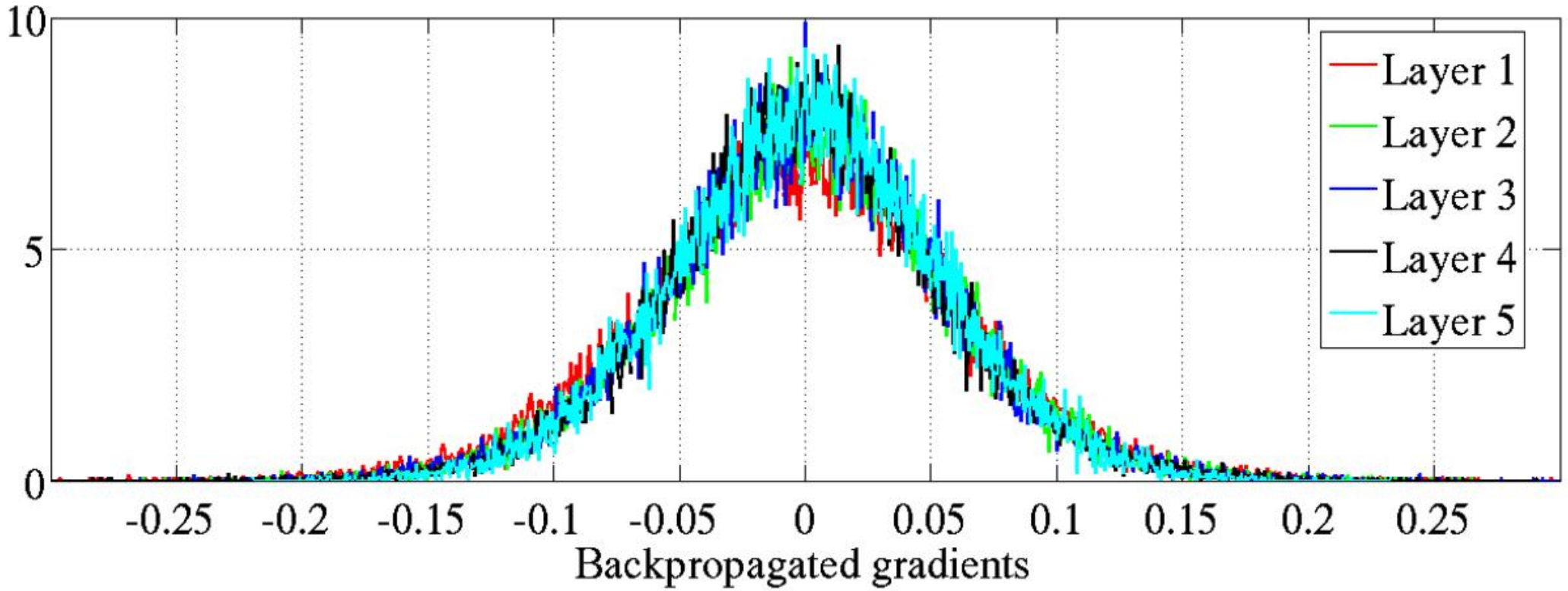
# Xavier Initialization

- Understanding the difficulty of training deep feedforward neural networks
- http://proceedings.mlr.press/v9/glorot10a.html

# Xavier Initialization



# Xavier Initialization



# Kaiming Initialization

- Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification
- https://arxiv.org/abs/1502.01852

# Kaiming Initialization

$$\text{std} = \sqrt{\frac{2}{(1 + a^2) \times \text{fan\_in}}}$$

# Training

- Metrics
- Loss
- Optimization
- Hyperparameters

# Note

# 课程总结

- Semantic Segmentation
- VGG
- ResNet
- Weight Initialization

# 重难点

- Semantic Segmentation
- VGG的实现方法
- ResNet的实现方法
- Weight Initialization的实现方法

# 扩展资料

- Very Deep Convolutional Networks for Large-Scale Image Recognition
- https://arxiv.org/abs/1409.1556
- Deep Residual Learning for Image Recognition
- https://arxiv.org/abs/1512.03385

# 扩展资料

- Convolutional Neural Networks (CNNs / ConvNets)

https://cs231n.github.io/convolutional-networks/

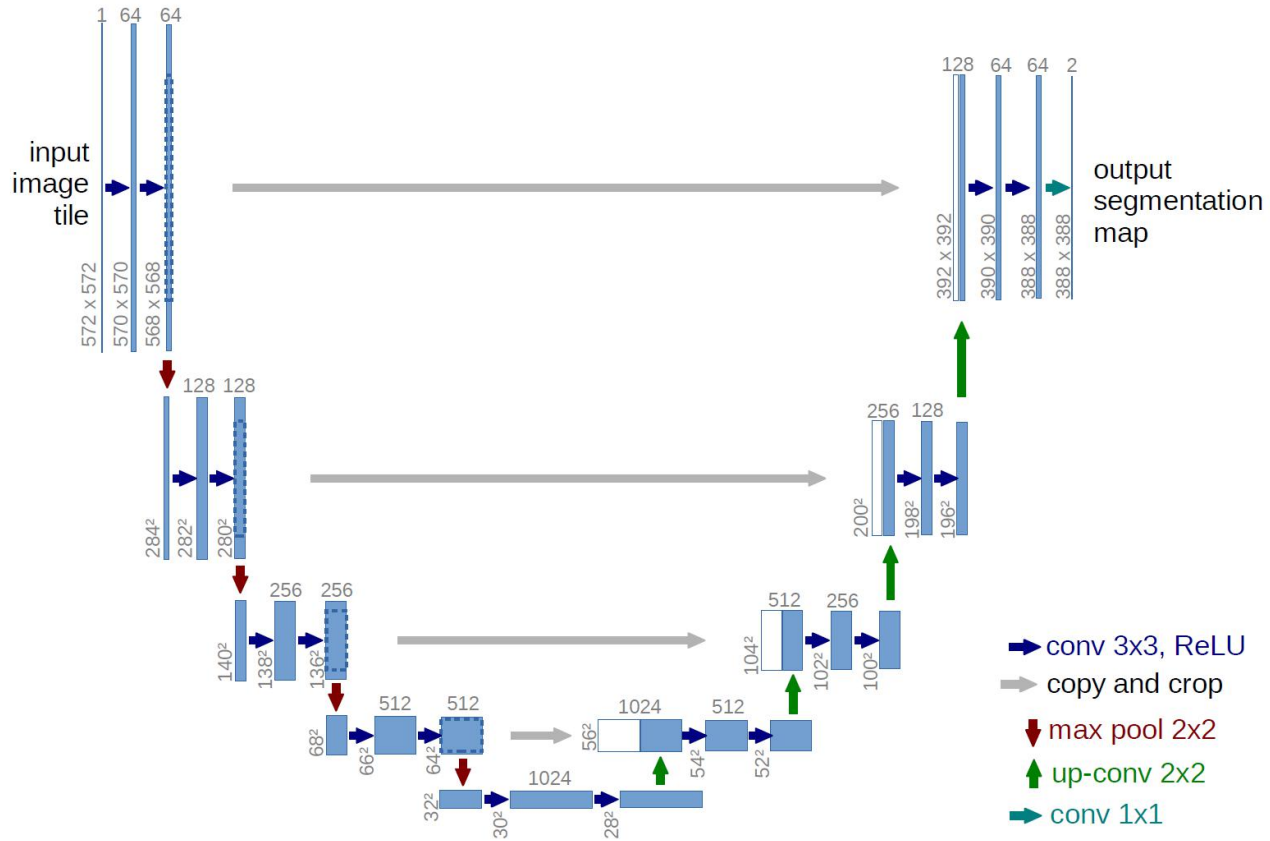
- Convolution arithmetic

https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

# 课程作业

- 安装开发环境
  - Framework (Tensorflow 2.0/Pytorch 1.3/PaddlePaddle 1.6)
- 实现ResNet50加载预训练参数实现图片分类

# Week 2：U-Net



后厂理工学院

houchangtech.com

一所专注前沿互联网技术领域的创新实战大学