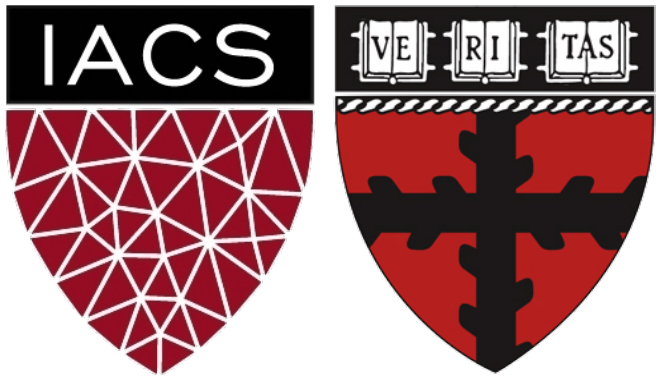
Transfer Learning

Pavlos Protopapas
Institute for Applied Computational Science
Harvard



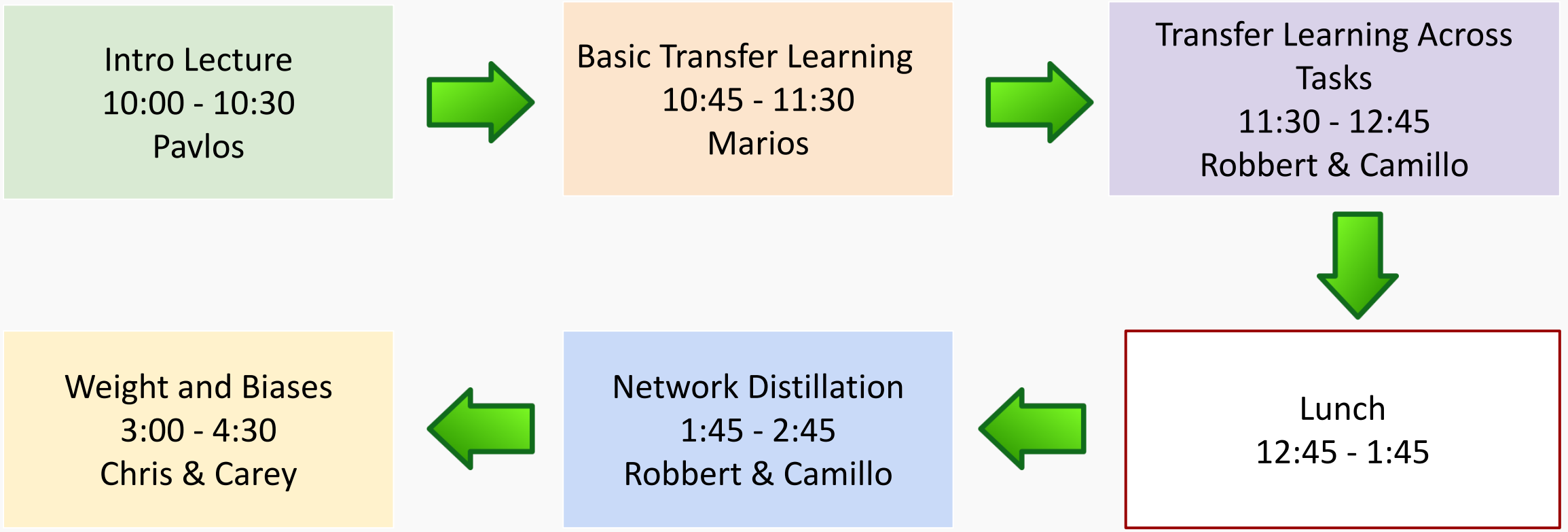
Marios Mattheakis

Robbert Struyven

Camilo Fosco

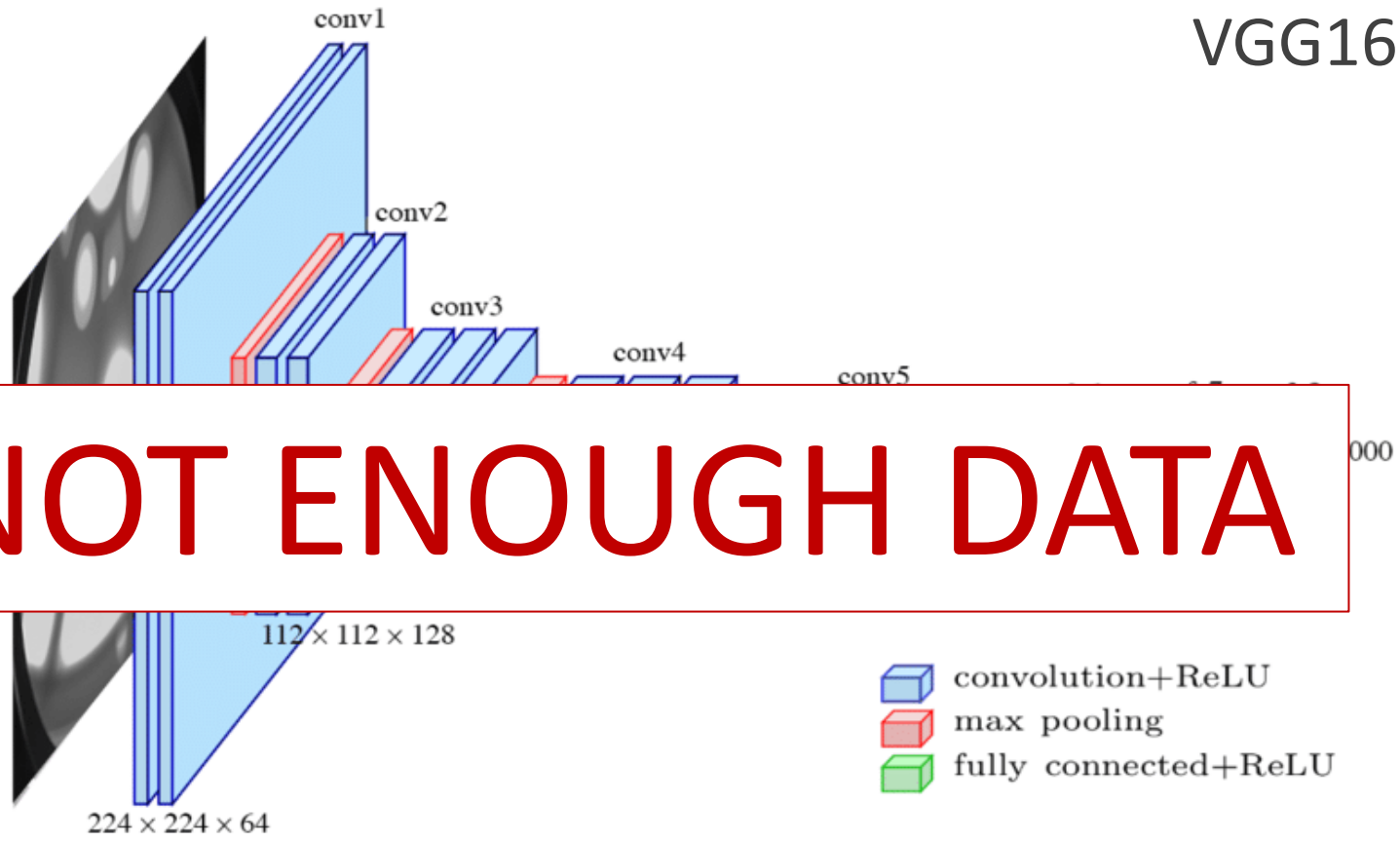
Vincent Casser

Workshop Overview for Session 2



Classify Rarest Animals

VGG16

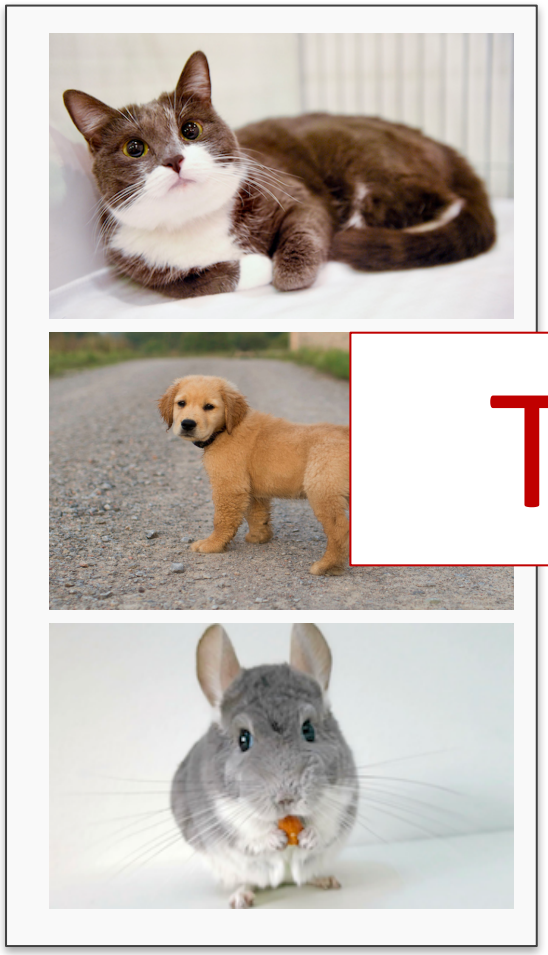


NOT ENOUGH DATA

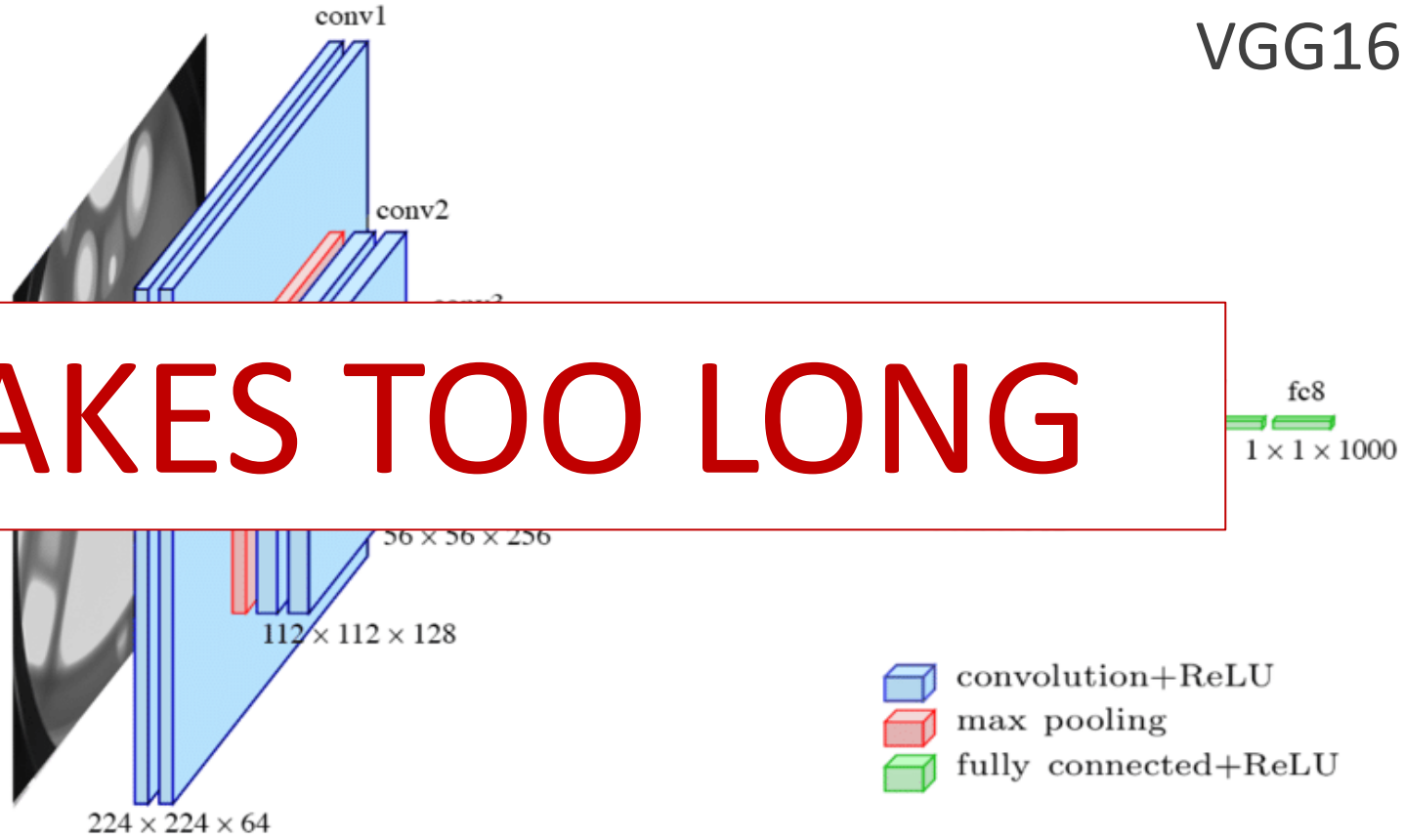
Number of parameters: 134,268,737

Data Set: Few hundred images

Classify Cats, Dogs, Chinchillas etc

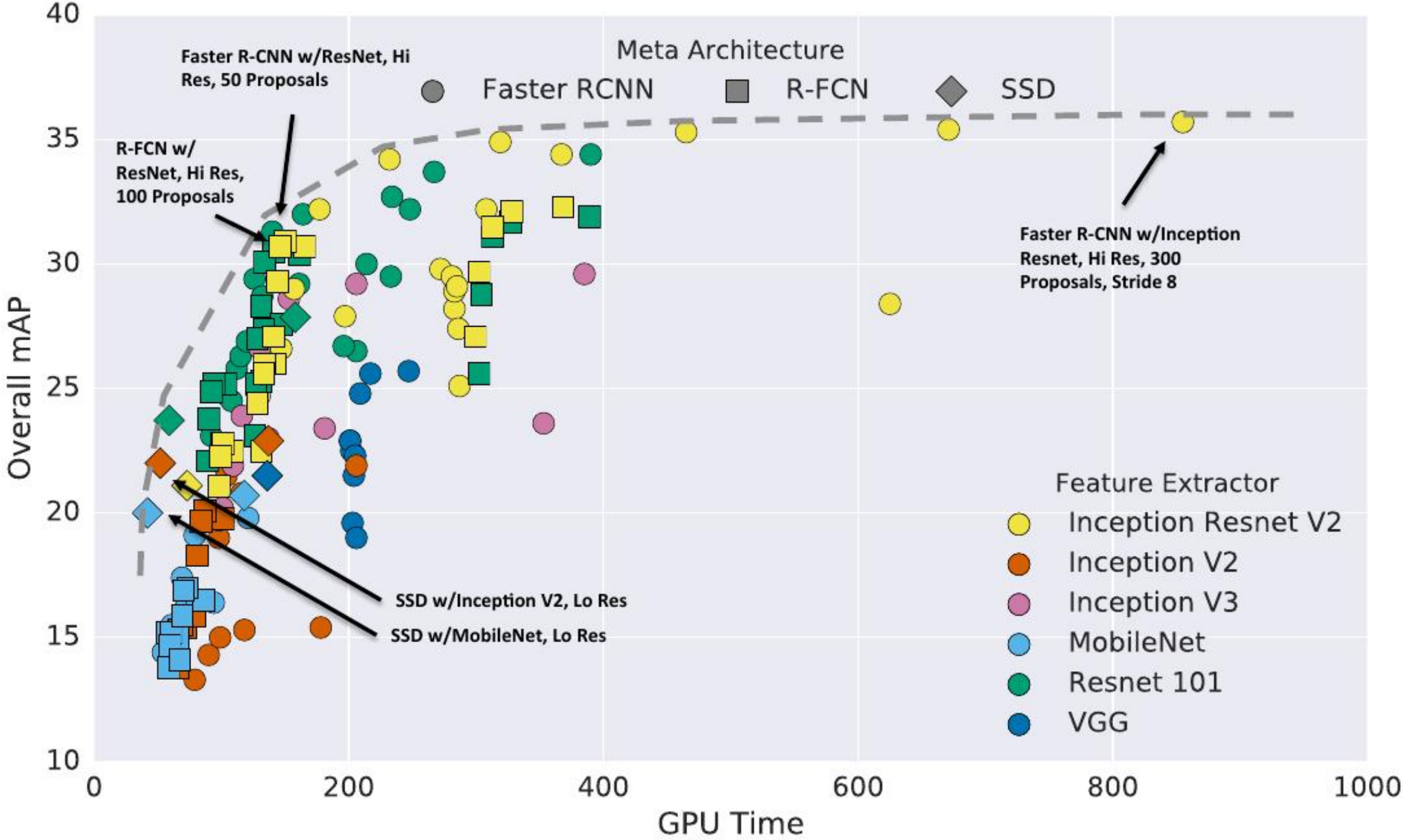


VGG16



TAKES TOO LONG

Number of parameters: 134,268,737
Enough training data. ImageNet approximate 1.2M


Meta Architecture
Faster RCNN
R-FCN
SSD
Faster R-CNN w/ResNet, Hi Res, 50 Proposals
R-FCN w/ ResNet, Hi Res, 100 Proposals
Faster R-CNN w/Inception Resnet, Hi Res, 300 Proposals, Stride 8
SSD w/Inception V2, Lo Res
SSD w/MobileNet, Lo Res
Feature Extractor
Inception Resnet V2
Inception V2
Inception V3
MobileNet
Resnet 101
VGG
Overall mAP
GPU Time
40
35
30
25
20
15
10
0
200
400
600
800
1000

Transfer Learning To The Rescue

How do you build an image classifier that can be trained in a few minutes on a CPU with very little data?

Basic idea of Transfer Learning

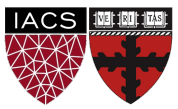
Train a ML model M for a task T using a dataset D_s

Use M on a new dataset D_T for the same task T

Use part of M on original dataset D_s for a new task T_n

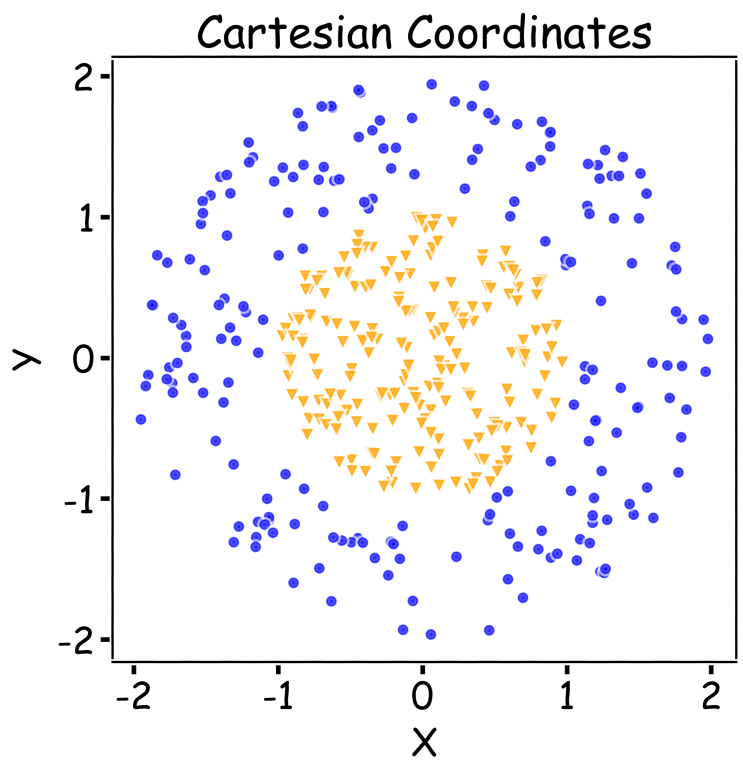
Use part of M on a new dataset D_T for a new task T_n

Wikipedia:
Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.[1]



Key Idea: Representation Learning

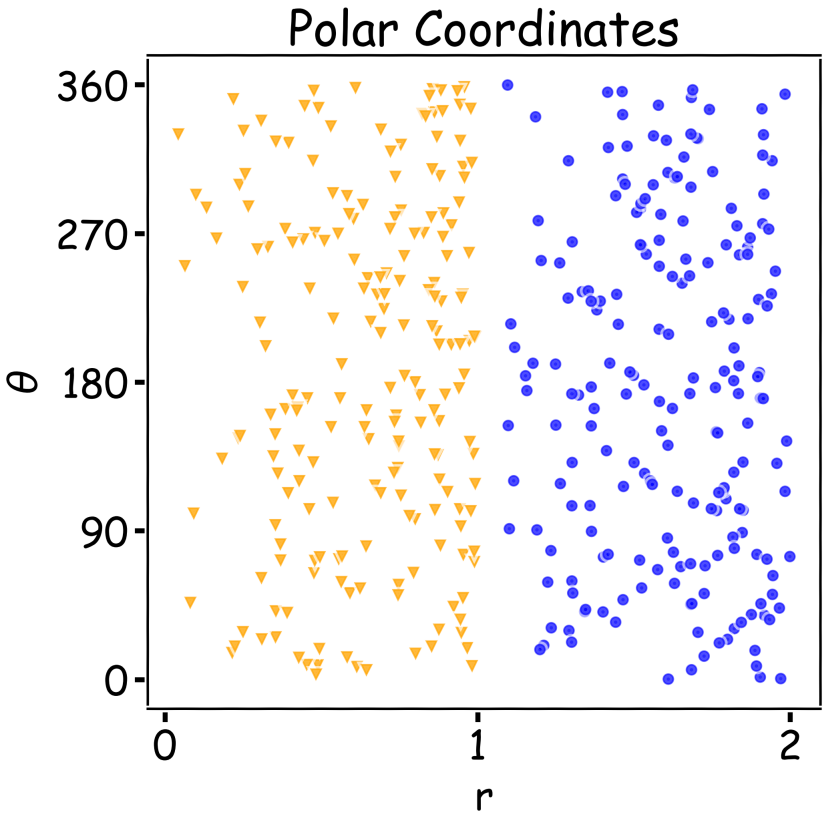
Relatively difficult task



Tranform:

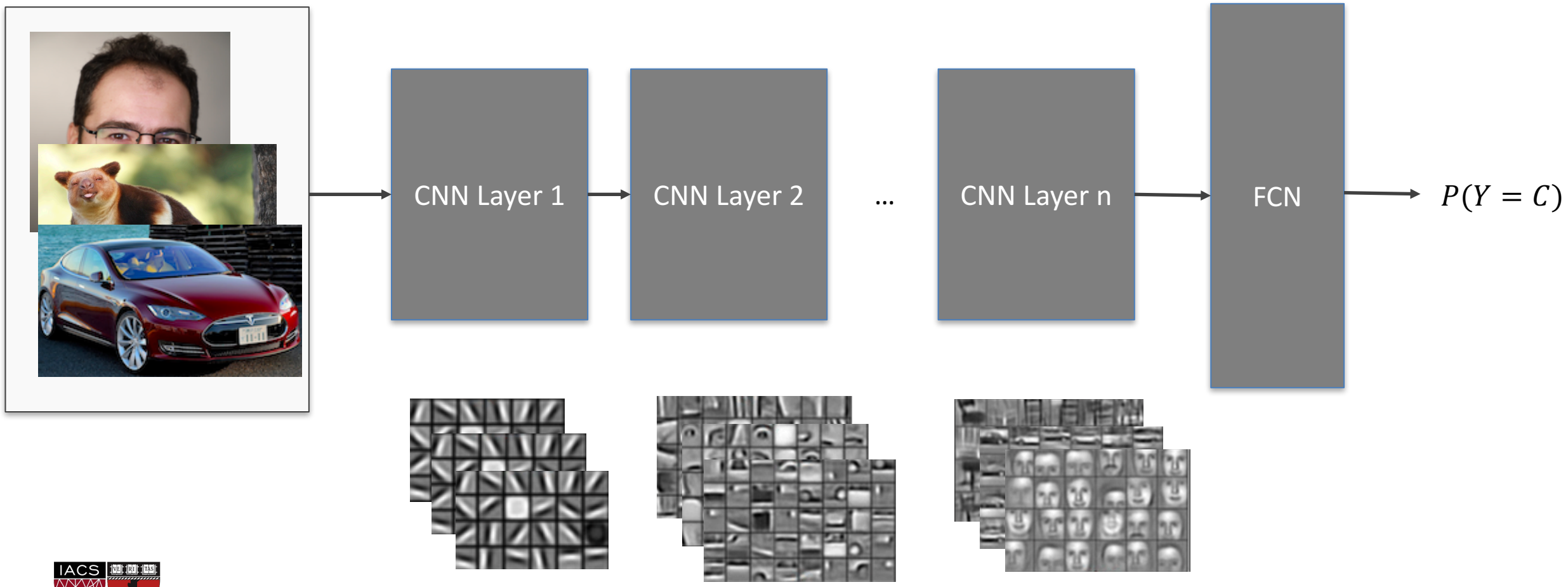
$(X, Y) \rightarrow (r, \theta)$

Easier task



Representation Learning

Task: classify cars, people, animals and objects



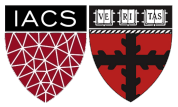
Transfer Learning To The Rescue

How do you make an image classifier that can be trained in a few minutes on a CPU with very little data?

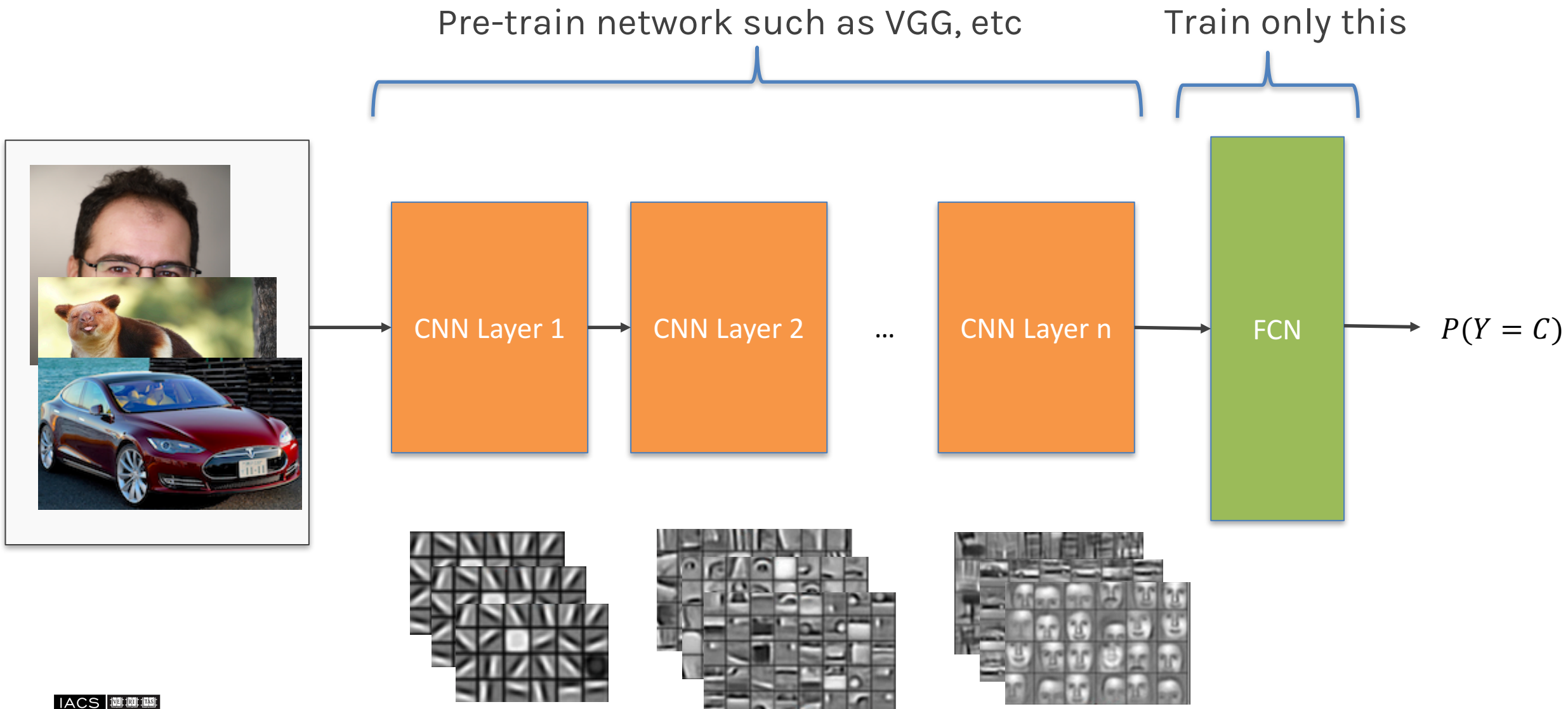
Use pre-trained models, i.e., models with known weights.

Main Idea: earlier layers of a network learn low level features, which can be adapted to new domains by changing weights at later and fully-connected layers.

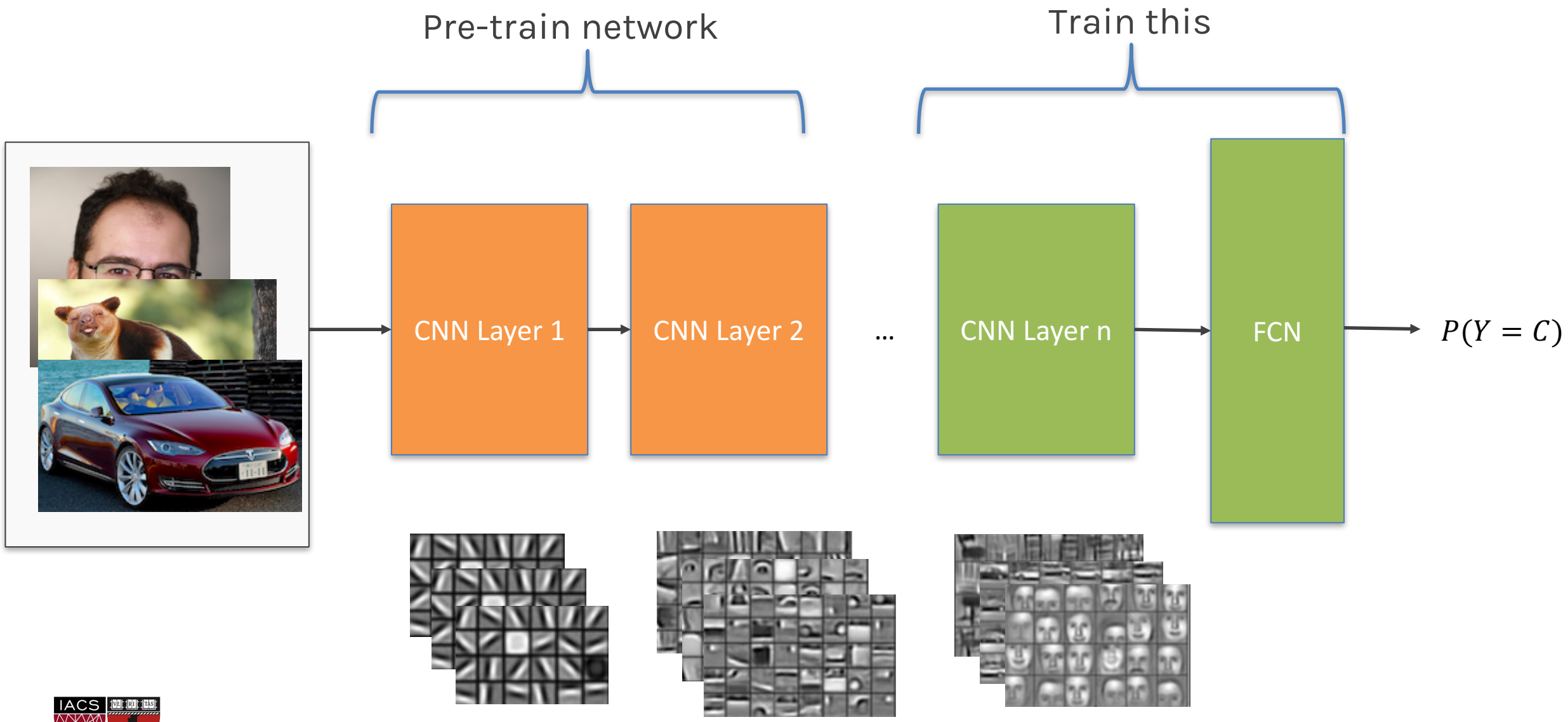
Example: use ImageNet trained with any sophisticated huge network. Then retrain it on a few images



Transfer Learning

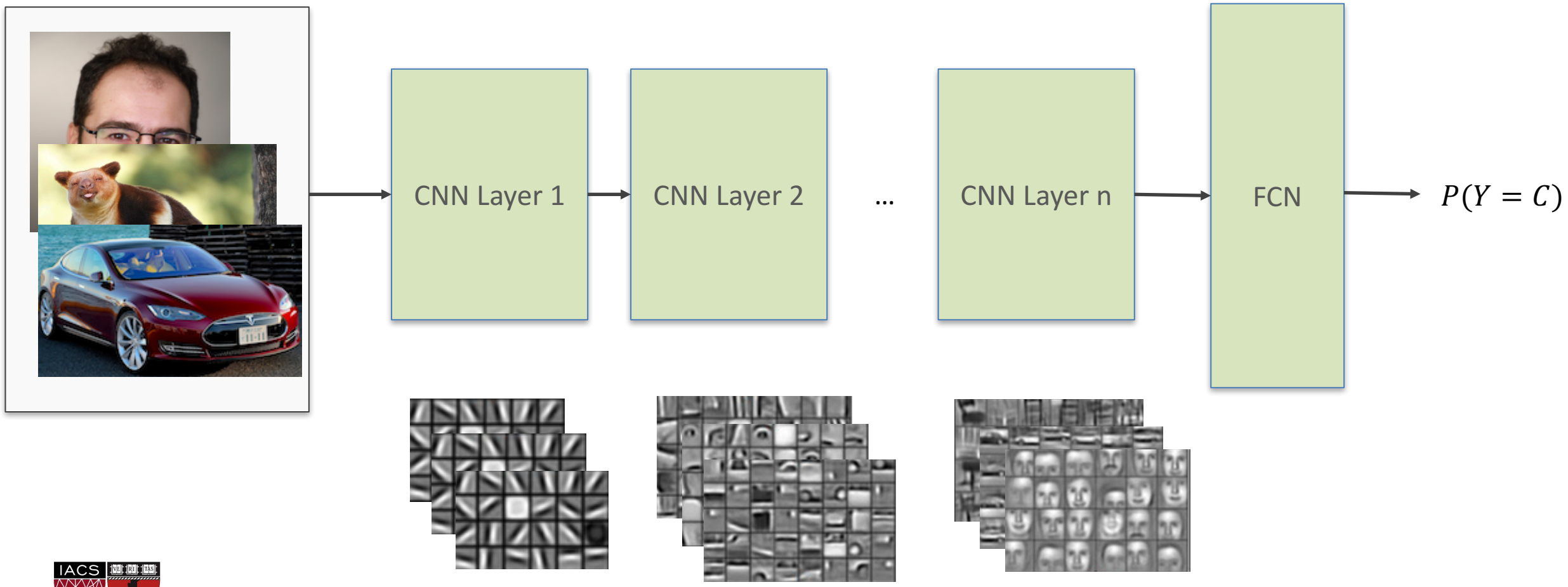


Transfer Learning



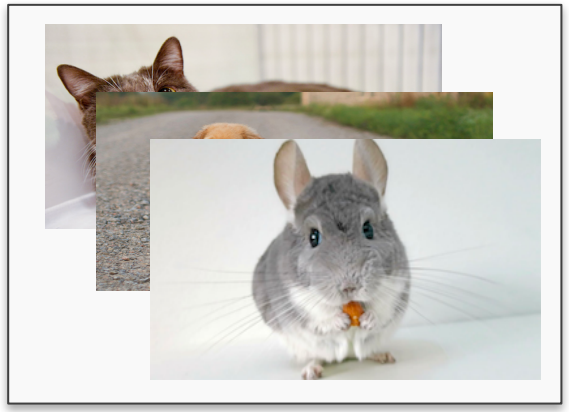
Transfer Learning – fine tuning

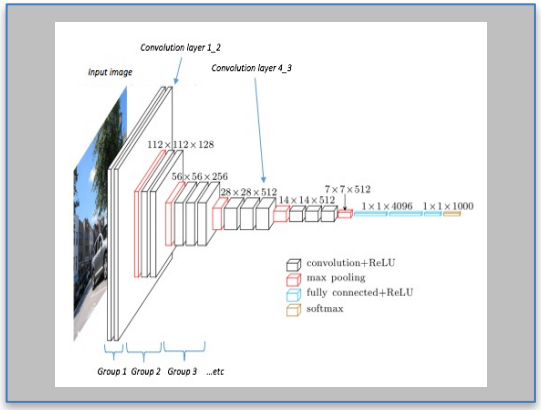
Train everything but start with weights that are trained already



Transfer Learning Across Tasks

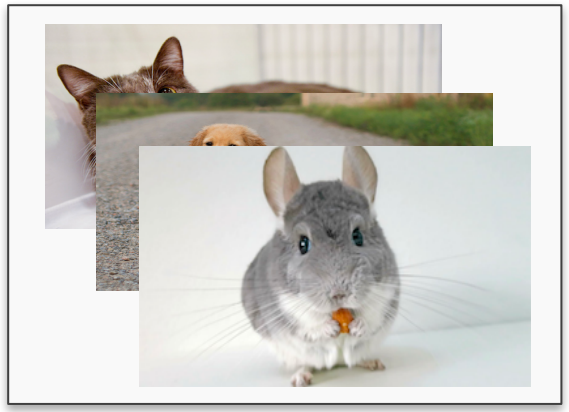
Classification:

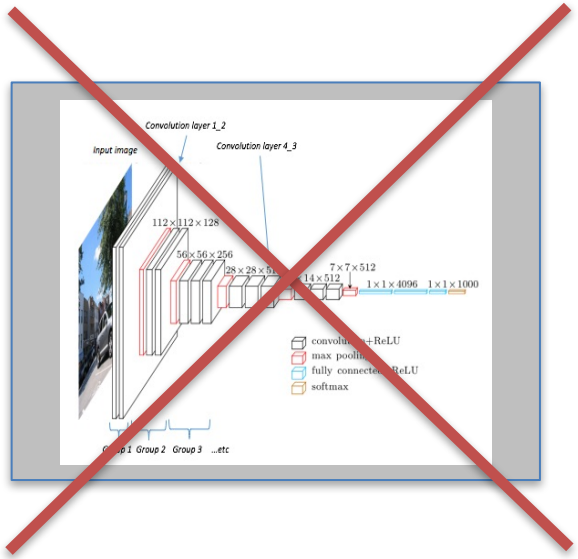


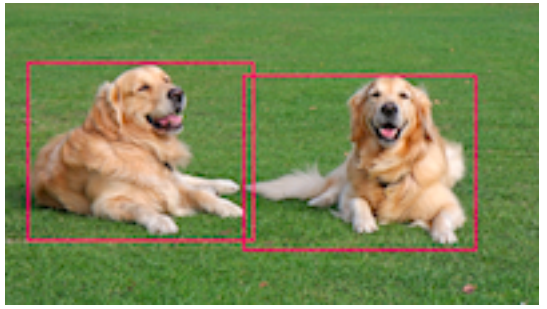


P(cat)
P(chinchilla)
P(dog)

Object Detection:

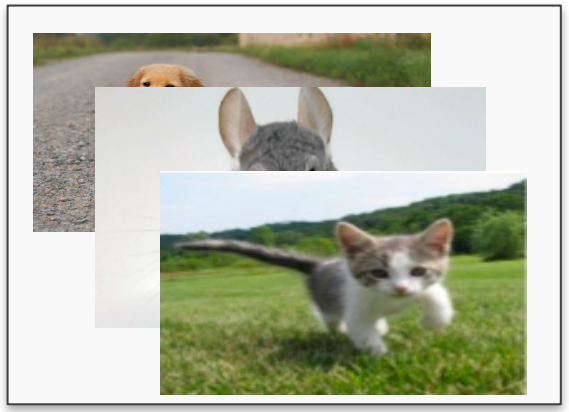


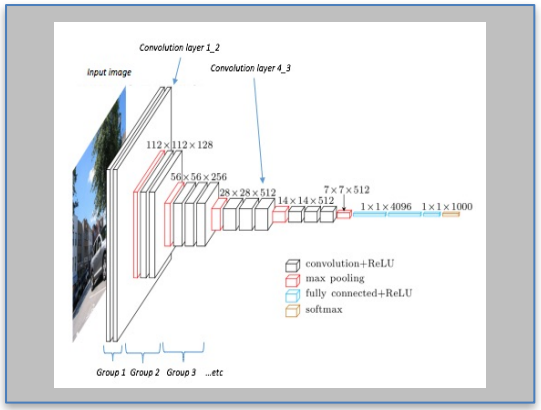




Transfer Learning Across Tasks (cont)

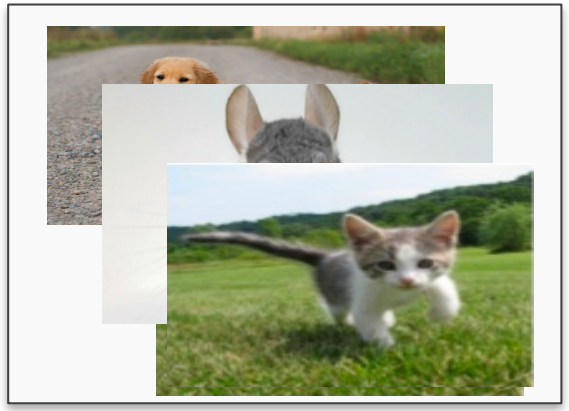
Classification:

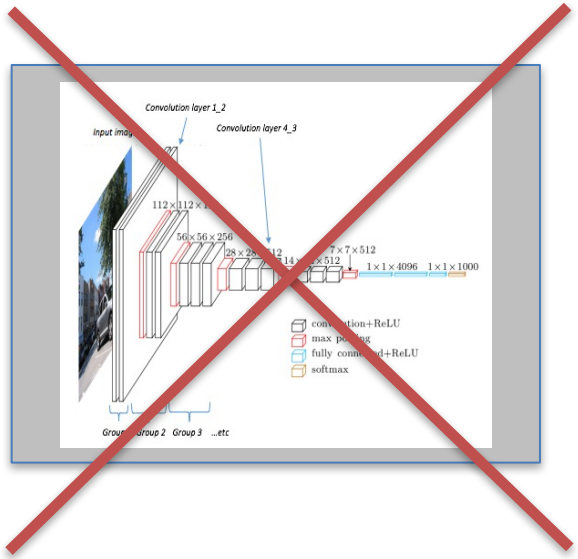




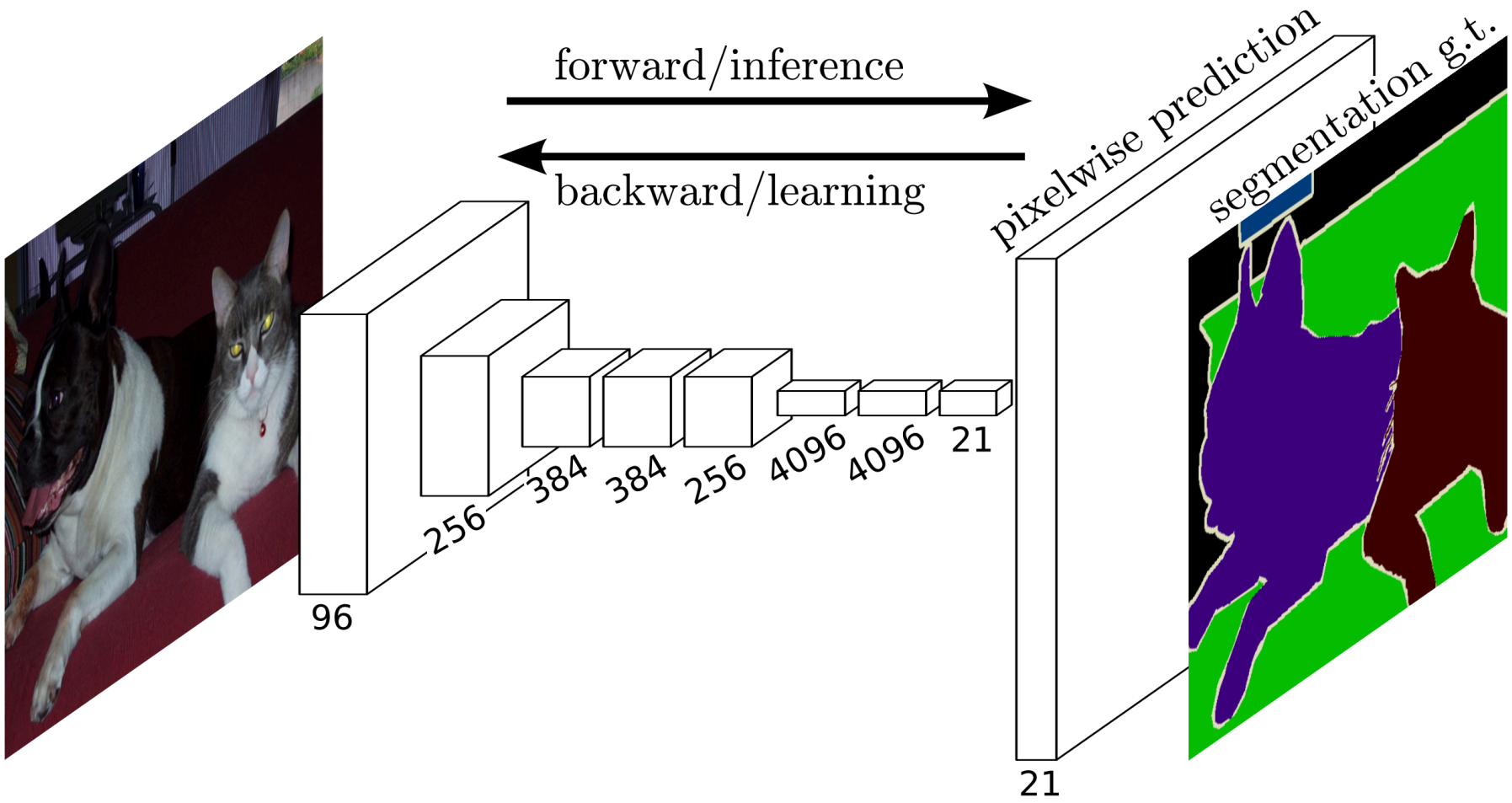
P(cat)
P(chinchilla)
P(dog)

Segmentation:

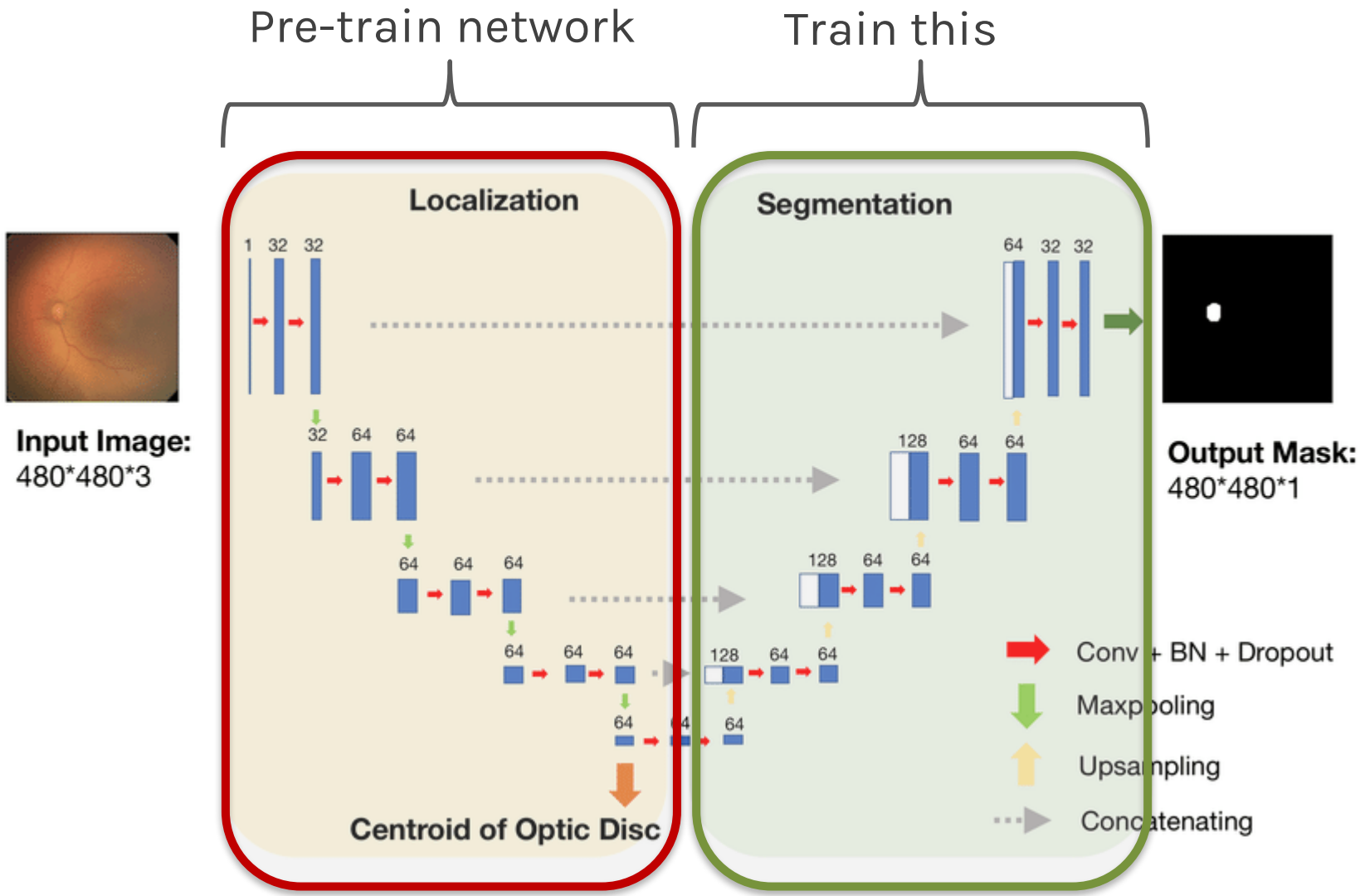




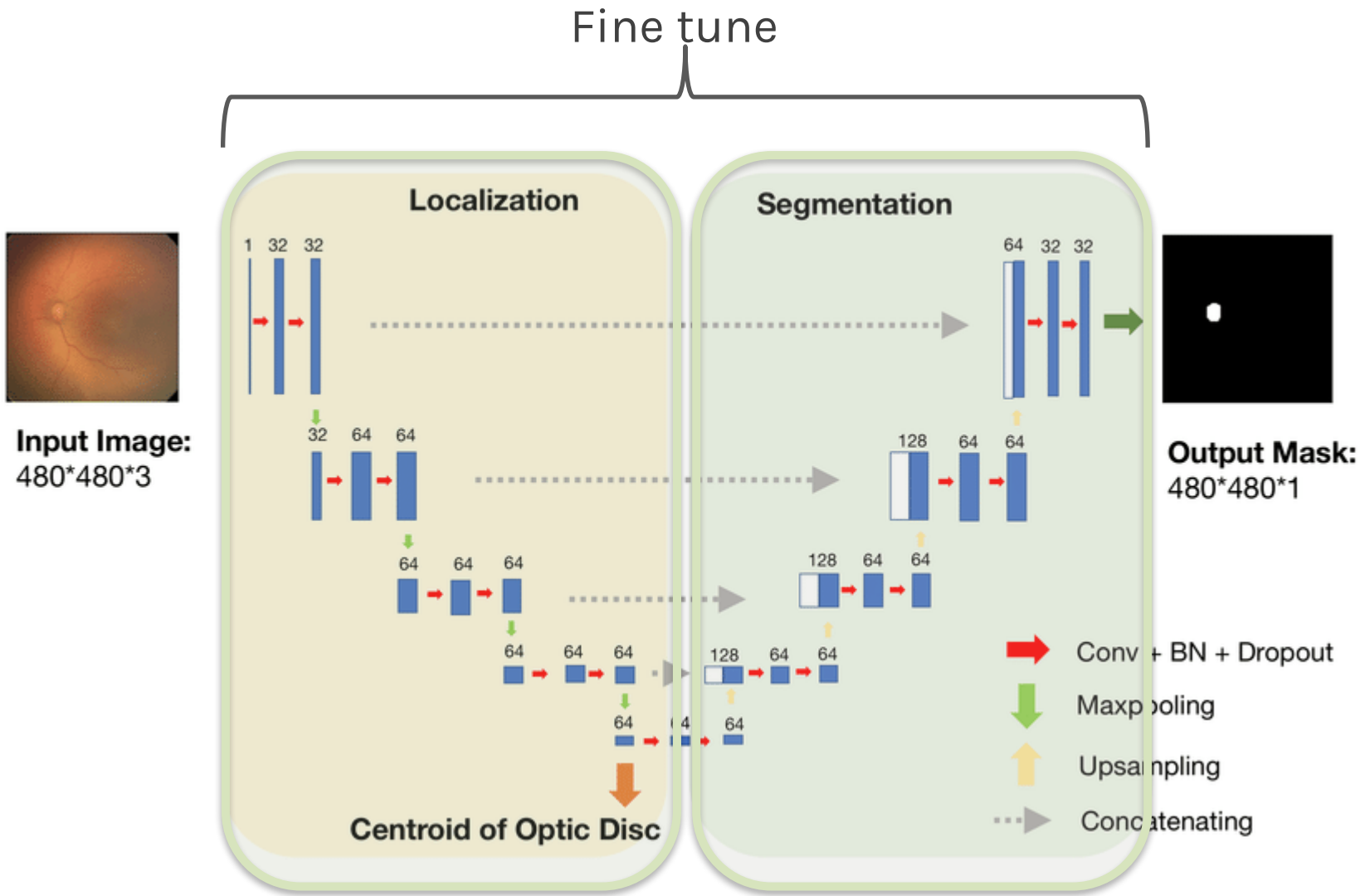
Segmentation



Segmentation: U-net architecture

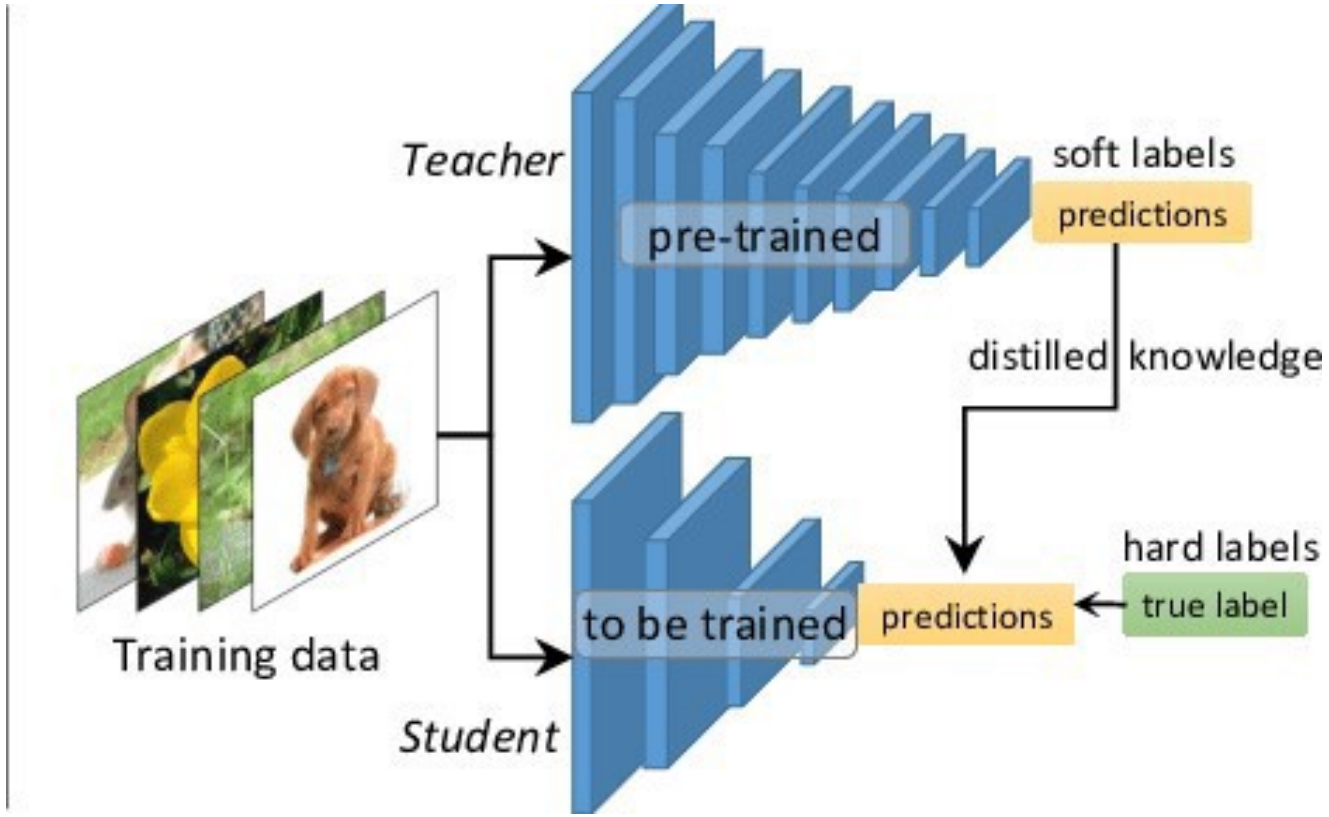


Segmentation (fine tuning)



Transfer Learning through Network Distillation

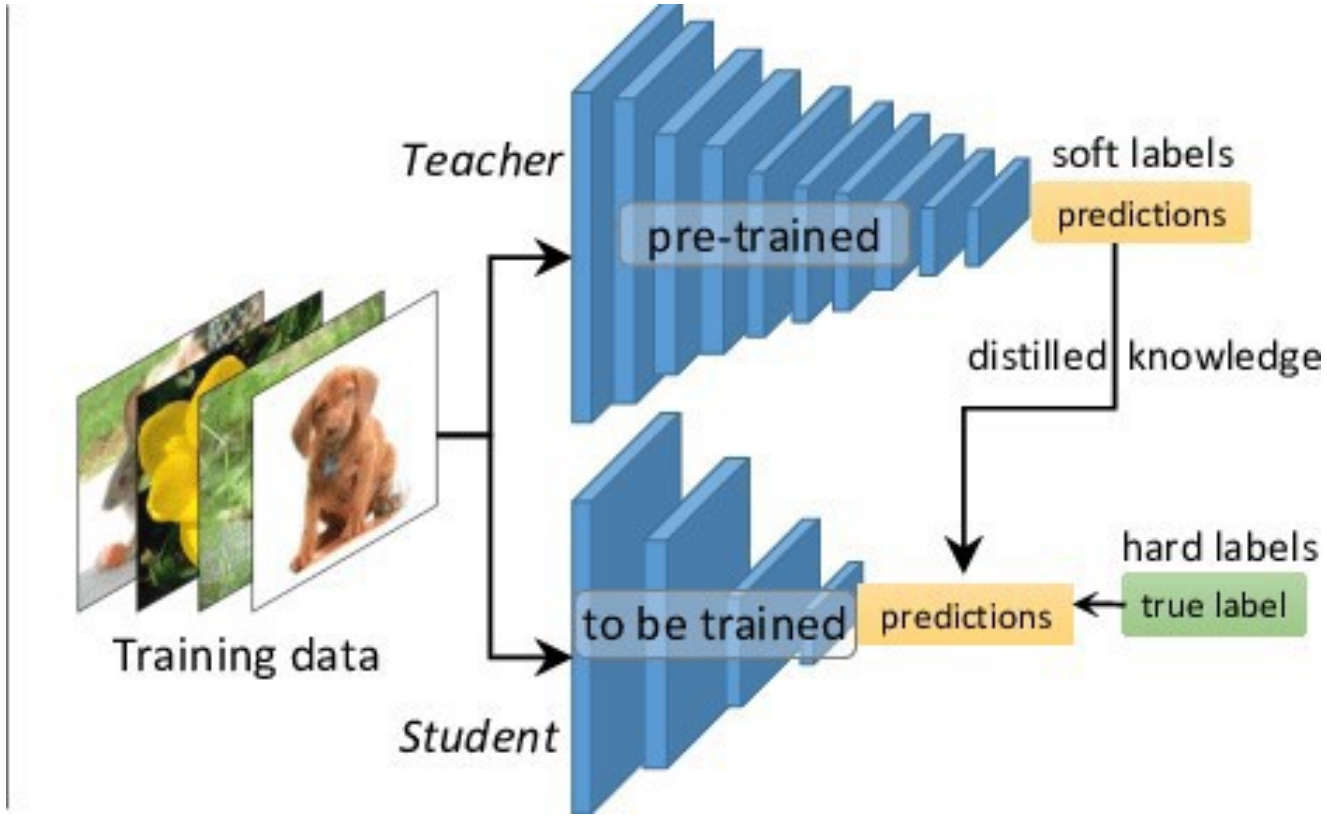
Question: is it possible to **distill** and **compress** the *knowledge* of the large and complex training model (**the teacher**) into a small and simple deployment model (**the student**)?



Standard cross entropy with correct labels

Cross entropy with the output from the teacher network predictions

Transfer Learning through Network Distillation (temperature)



Standard cross entropy with correct labels

Soften labels with temperature

$$q_i = \frac{e^{z_i/T}}{\sum e^{z_i/T}}$$

Cross entropy with the soft targets from the teacher network predictions

Workshop Overview for Session 2