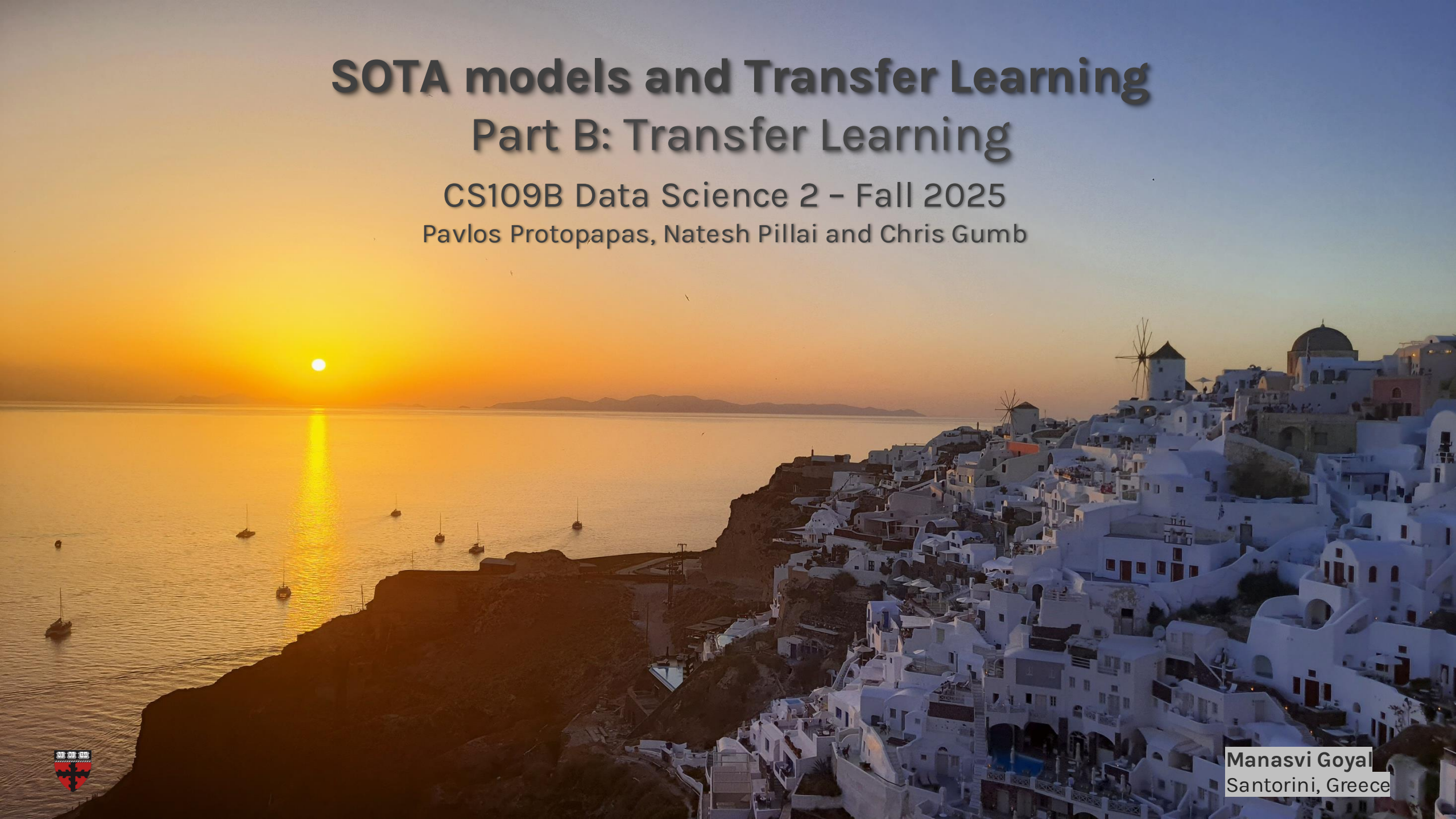
SOTA models and Transfer Learning

Part B: Transfer Learning

CS109B Data Science 2 – Fall 2025

Pavlos Protopapas, Natesh Pillai and Chris Gumb



Manasvi Goyal
Santorini, Greece


TRANSFER LEARNING
WHAT SOCIETY THINKS I DO
WHAT MY FRIENDS THINK I DO
WHAT INVESTORS THINK I DO
Transfer Learning using Mobilenet and Keras
K?
I'm going to learn jujitsu?
WHAT MY MOM THINKS I DO
WHAT I THOUGHT I'LL DO
WHAT I ACTUALLY DO
imgflip.com

Outline

- Introduction to Transfer Learning
- Transfer Learning Strategies

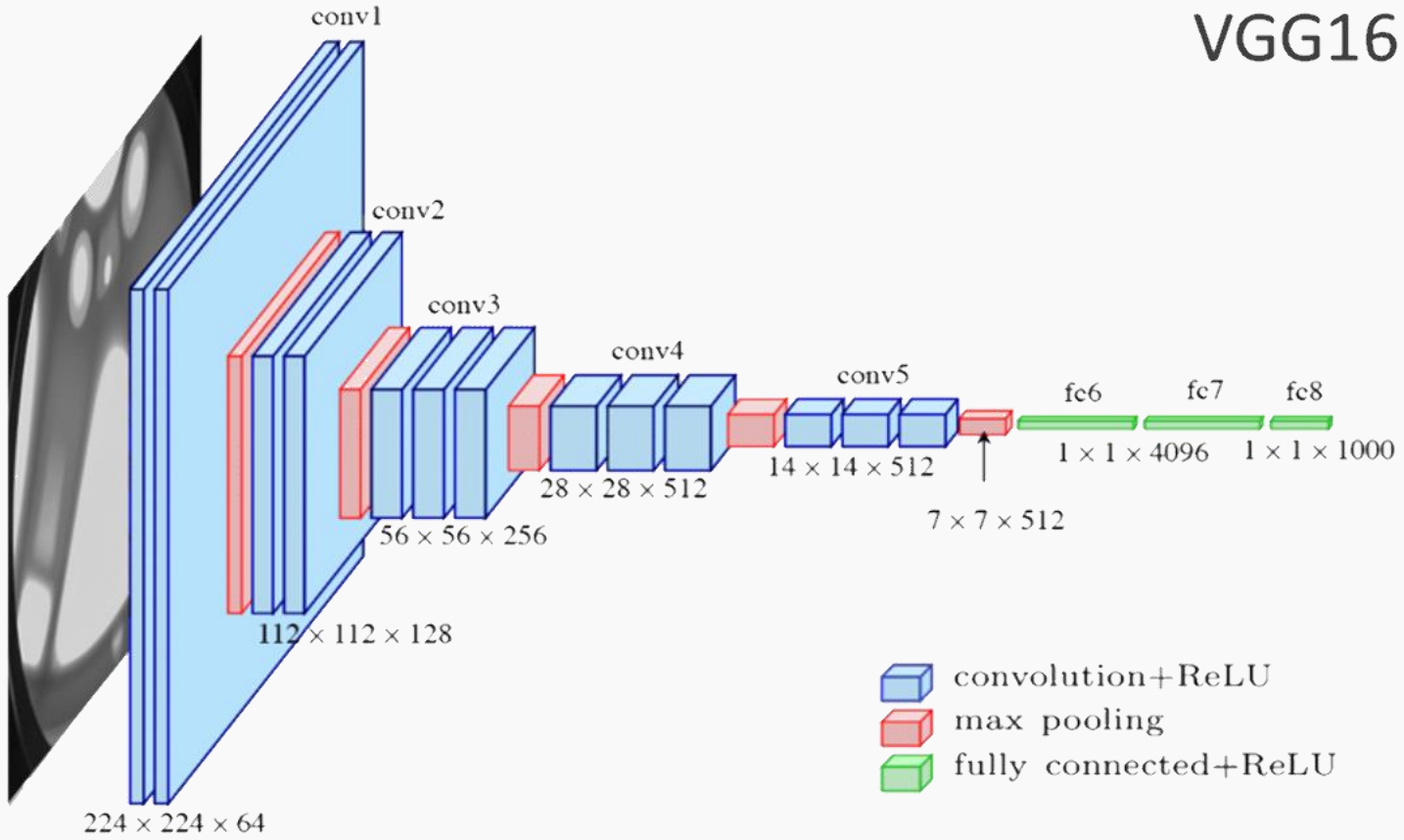
Outline

- **Introduction to Transfer Learning**
- Transfer Learning Strategies

Motivation for Transfer Learning

- ImageNet has more than 14 million labeled images and more than 1000 categories and SOTAs networks perform amazingly well.
- The ImageNet challenge is only a very tiny subset of all possible categories for which we may not have a lot of training data. For example, can you guess the animals in the images below?

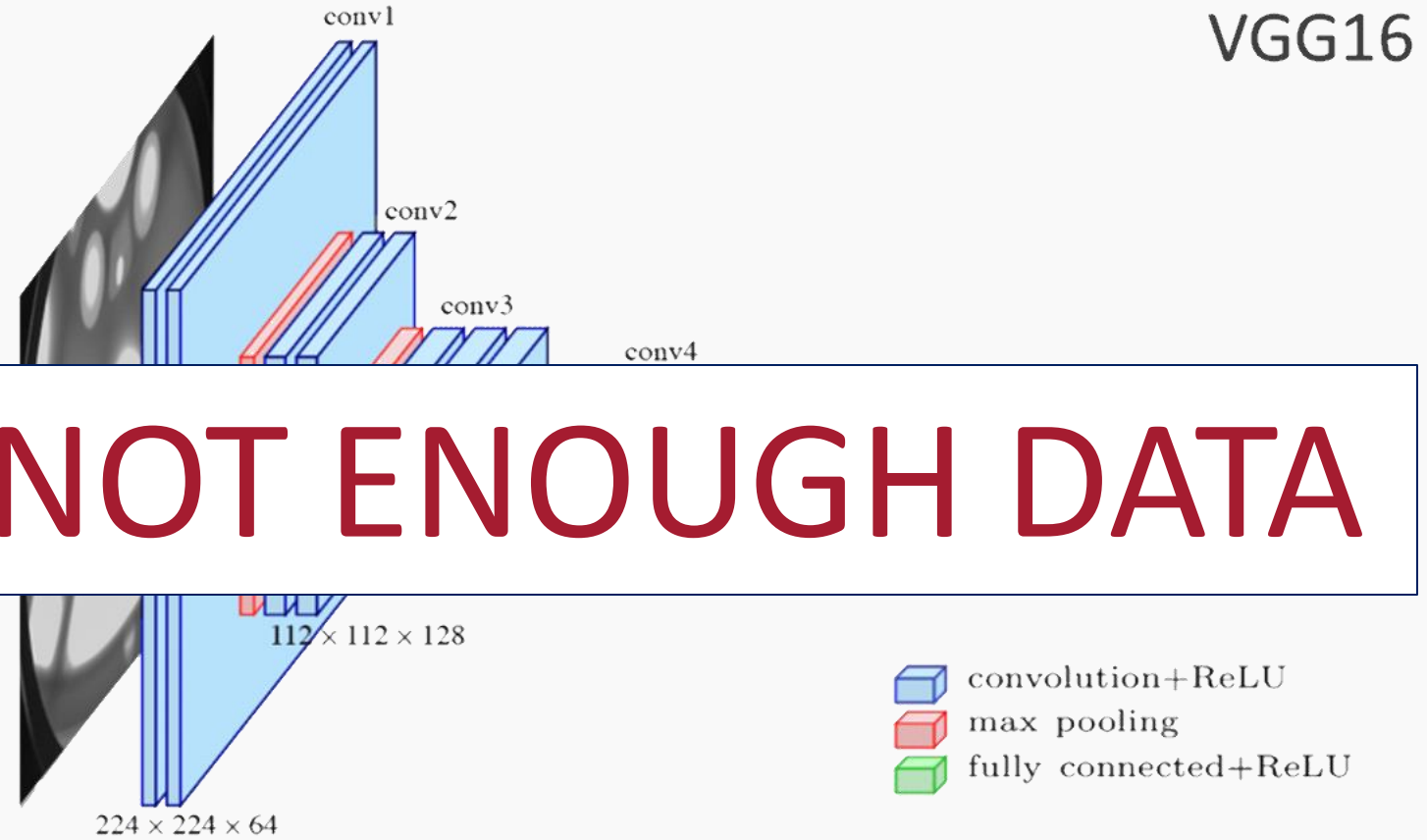
Classify Rarest Animals



Number of parameters: 134,268,737

Data Set: Few hundred images

Classify Rarest Animals

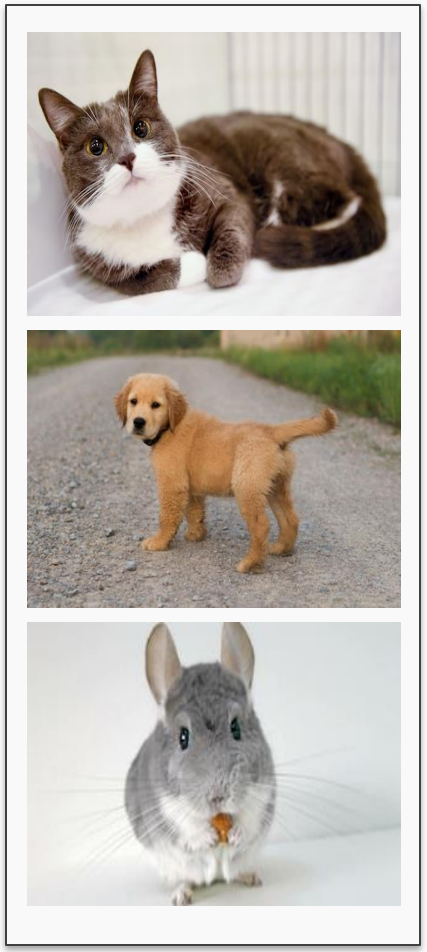


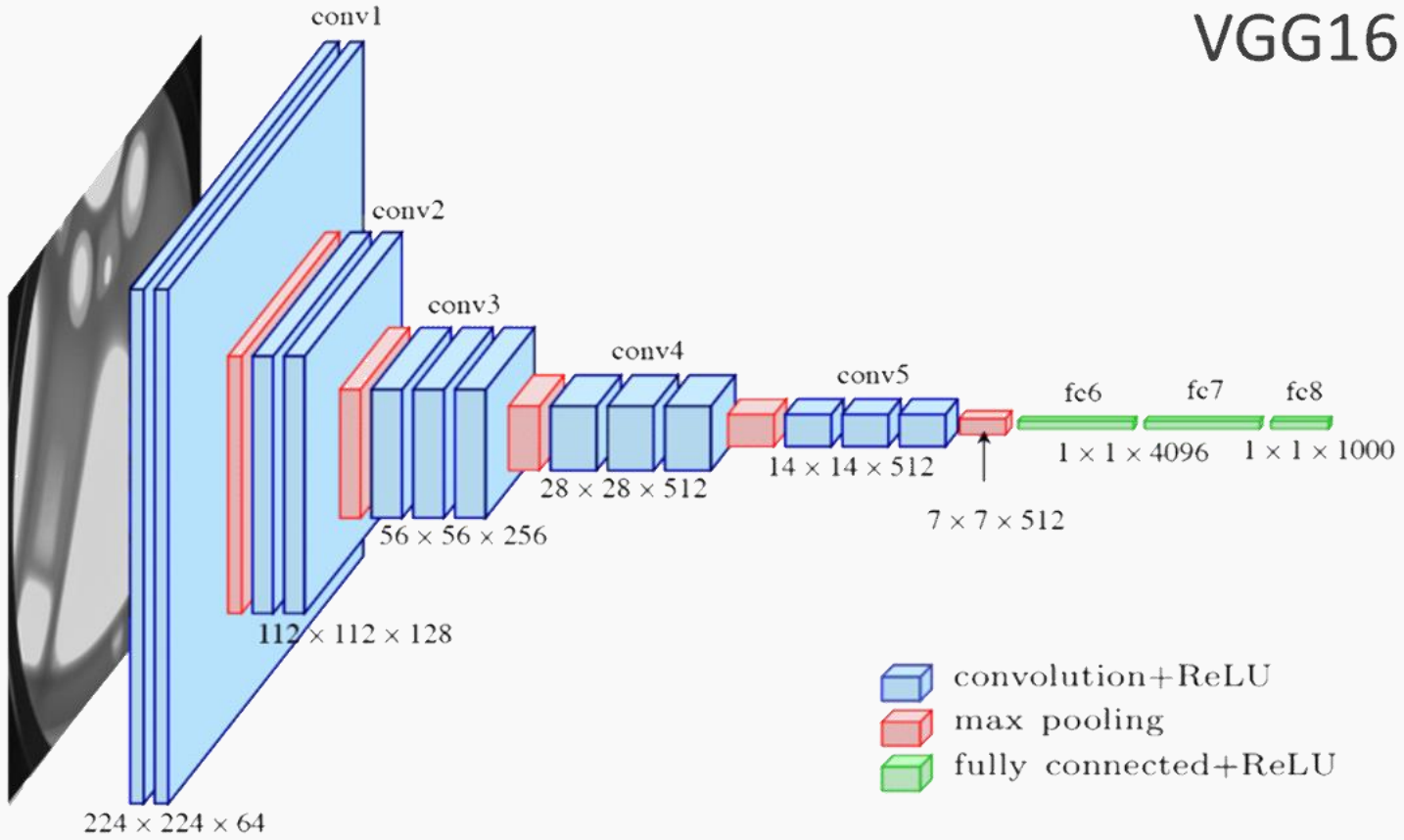
NOT ENOUGH DATA

Number of parameters: 134,268,737

Data Set: Few hundred images

Classify Cats, Dogs, Chinchillas etc

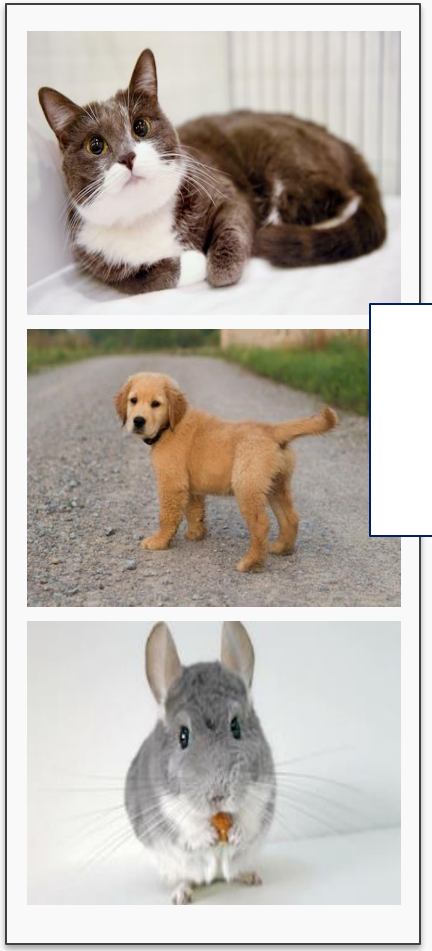


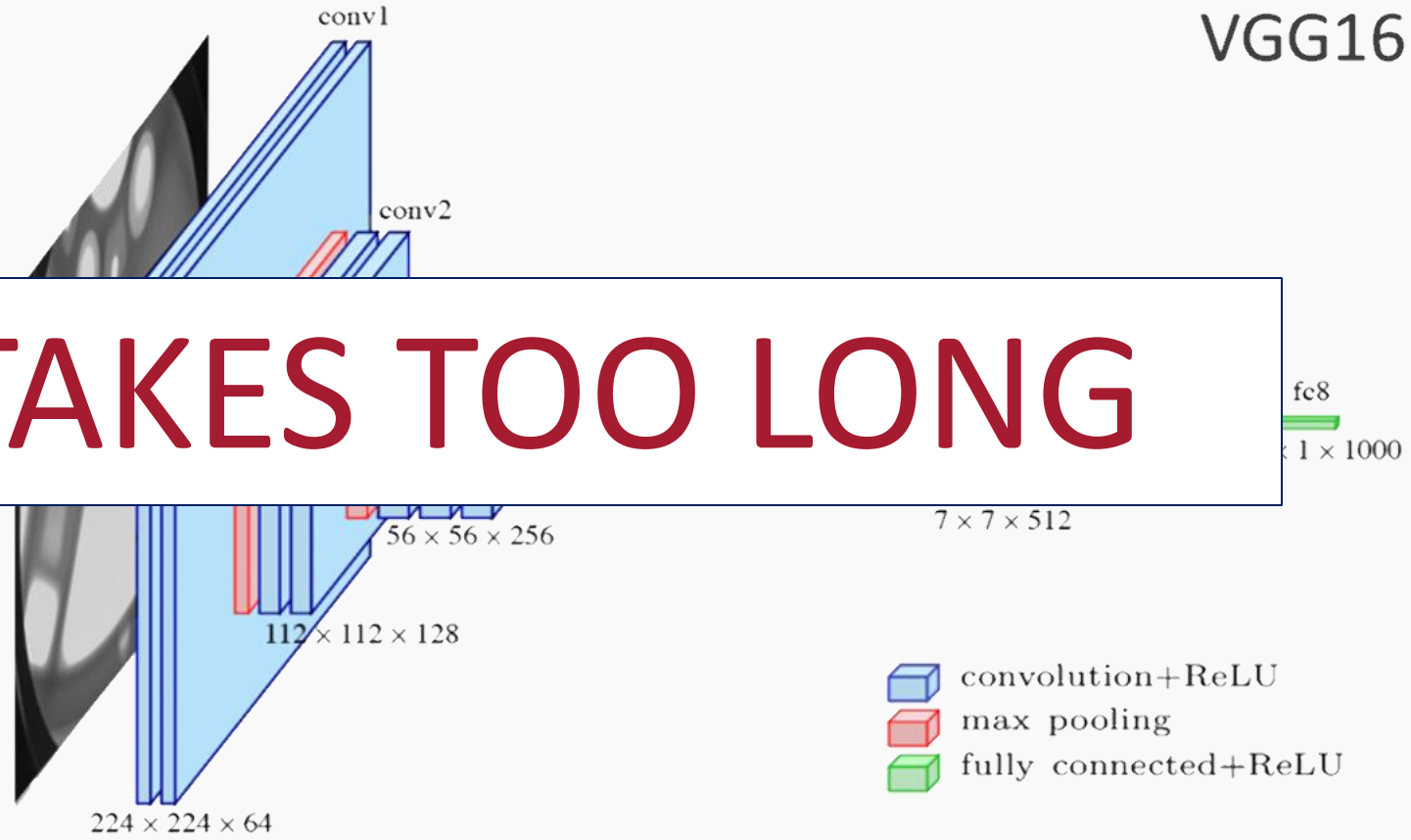


Number of parameters: 134,268,737

Enough training data. ImageNet approximate 1.2M

Classify Cats, Dogs, Chinchillas etc



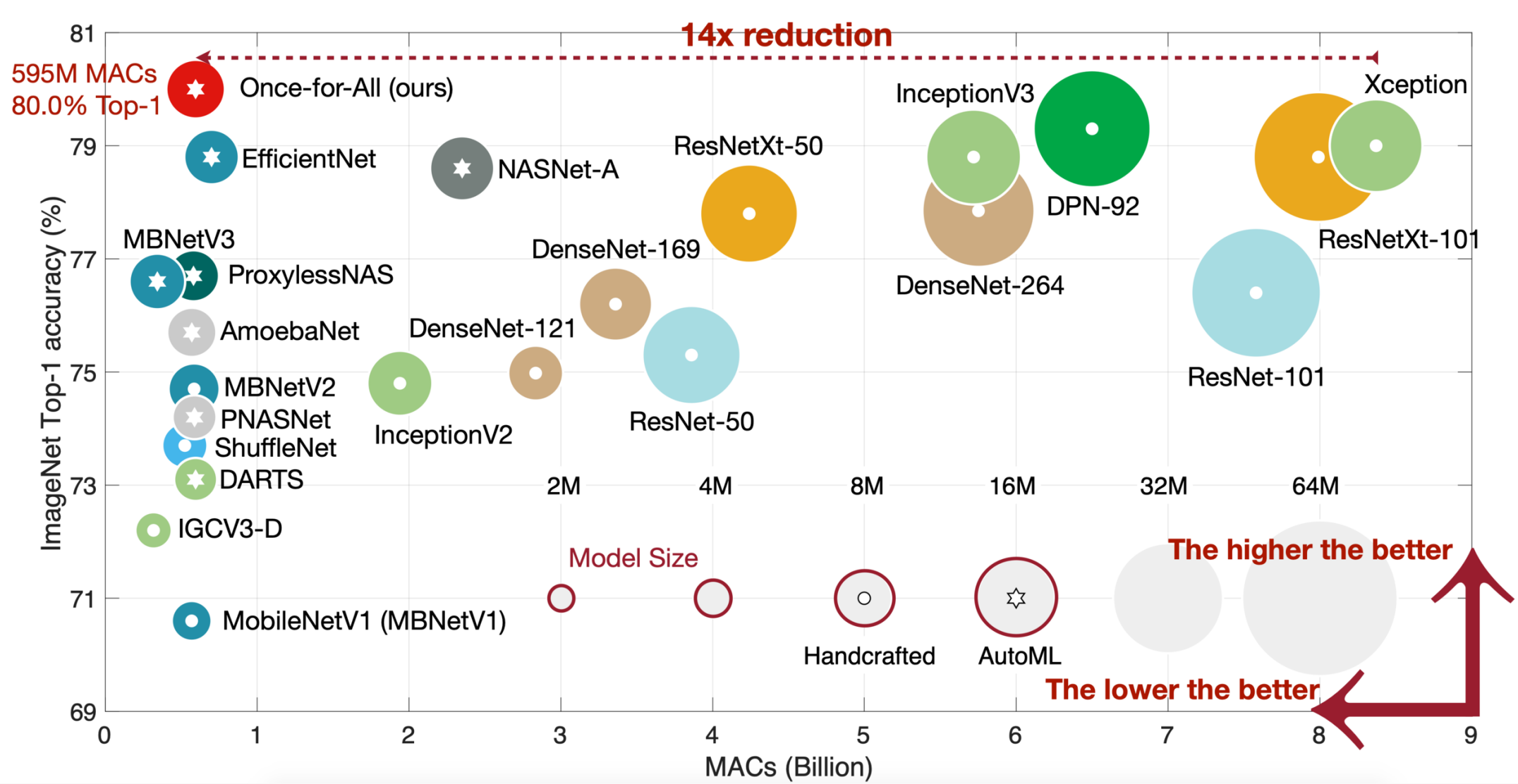


TAKES TOO LONG

Number of parameters: 134,268,737

Enough training data. ImageNet approximate 1.2M

Motivation for Transfer Learning



[source]

Motivation for Transfer Learning

Not only does it take too long to keep training models from scratch but is also harmful to the environment.

Do you know?

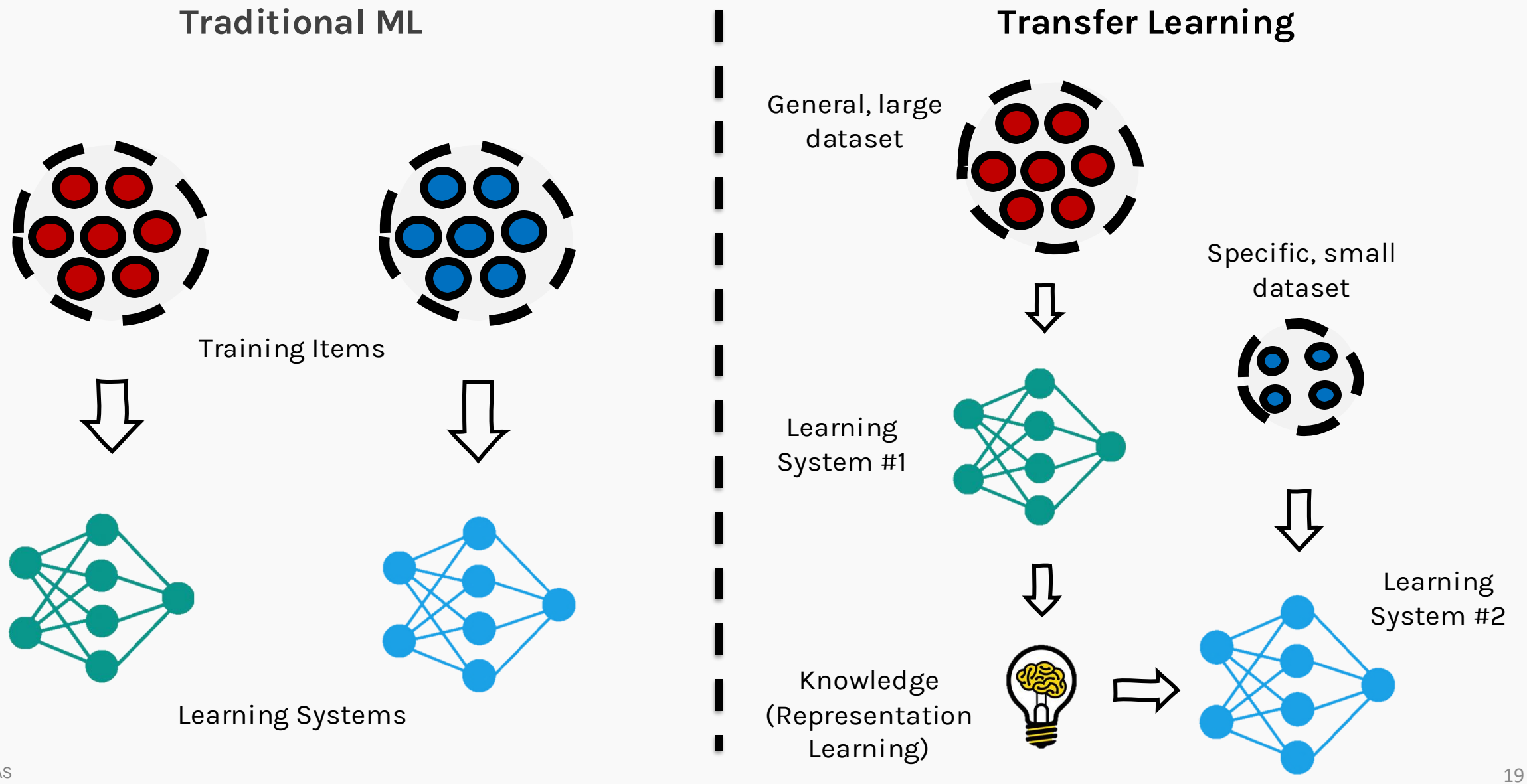
A recent research paper — Energy and Policy Considerations for Deep Learning in NLP — notes that an inefficiently trained NLP model using Neural Architecture Search can emit **more than 626,000 pounds of CO_2**. That's about five times the lifetime emissions of an average American car!

Motivation for Transfer Learning

So how do you build an image classifier that can be trained in a few minutes with very little data?

Wikipedia:
Transfer learning (TL) focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.

Basic idea of Transfer Learning

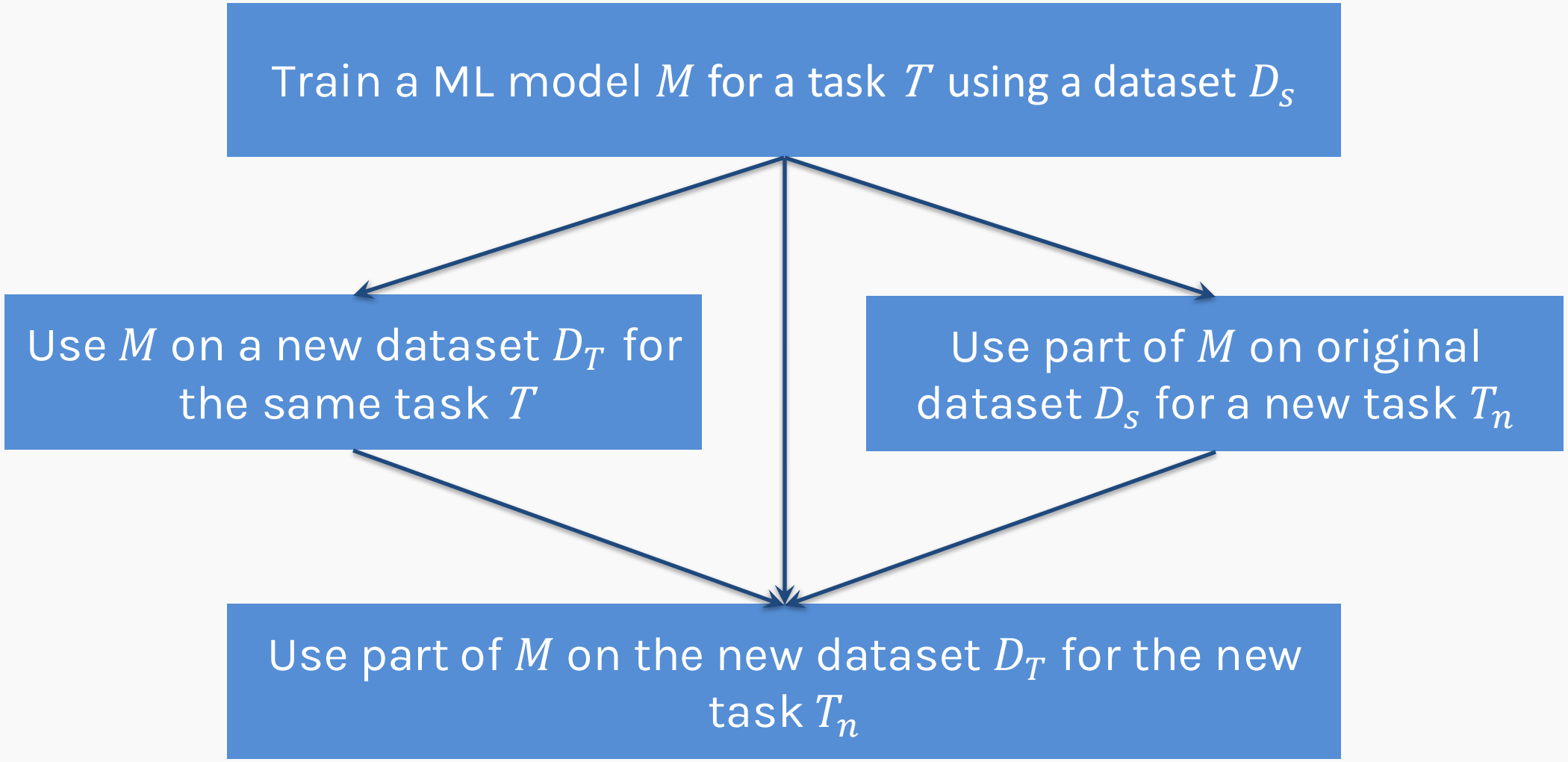


Basic idea of Transfer Learning

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

The model M can create representations of our data, similar to how self-supervised models like autoencoders and language models do.

Basic idea of Transfer Learning

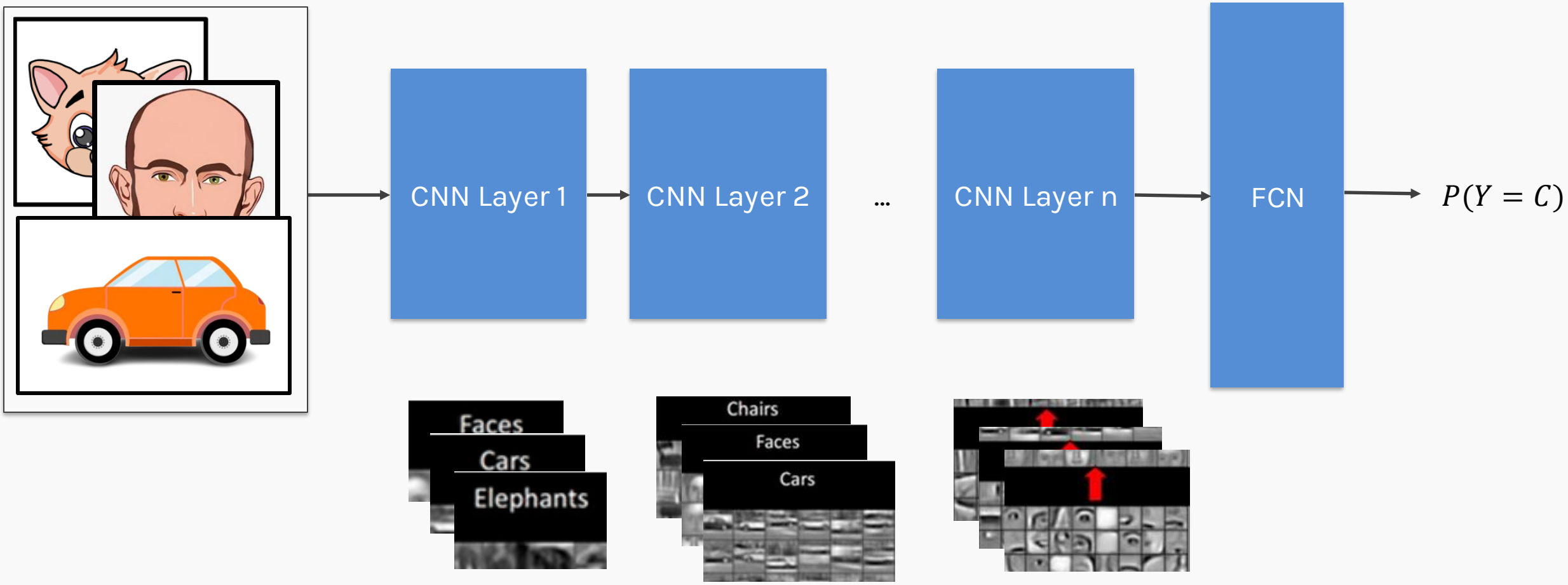


Outline

- Introduction to Transfer Learning
- **Transfer Learning Strategies**

Transfer Learning Strategies

Assume we want to classify cars, people, animals and other objects

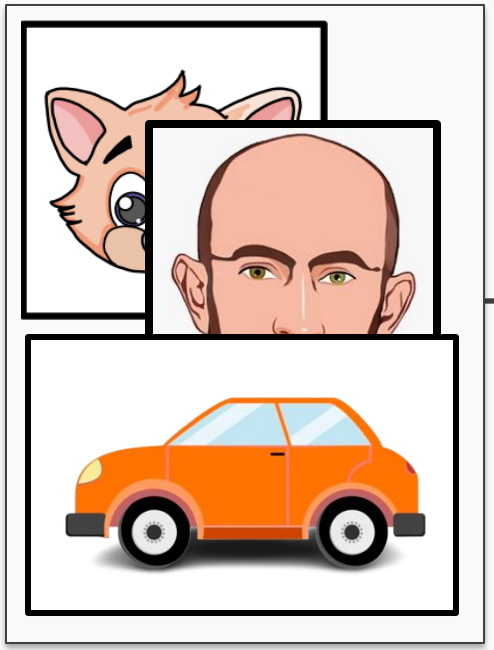


Transfer Learning Strategies

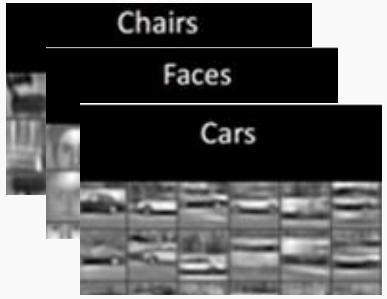
We don't change the weights for this (i.e. weights are frozen).

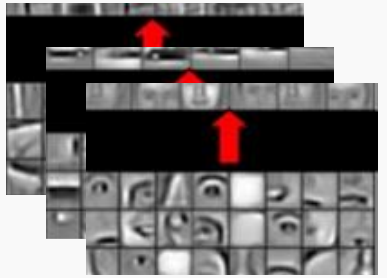
Train only this

Pre-train network such as VGG, etc.

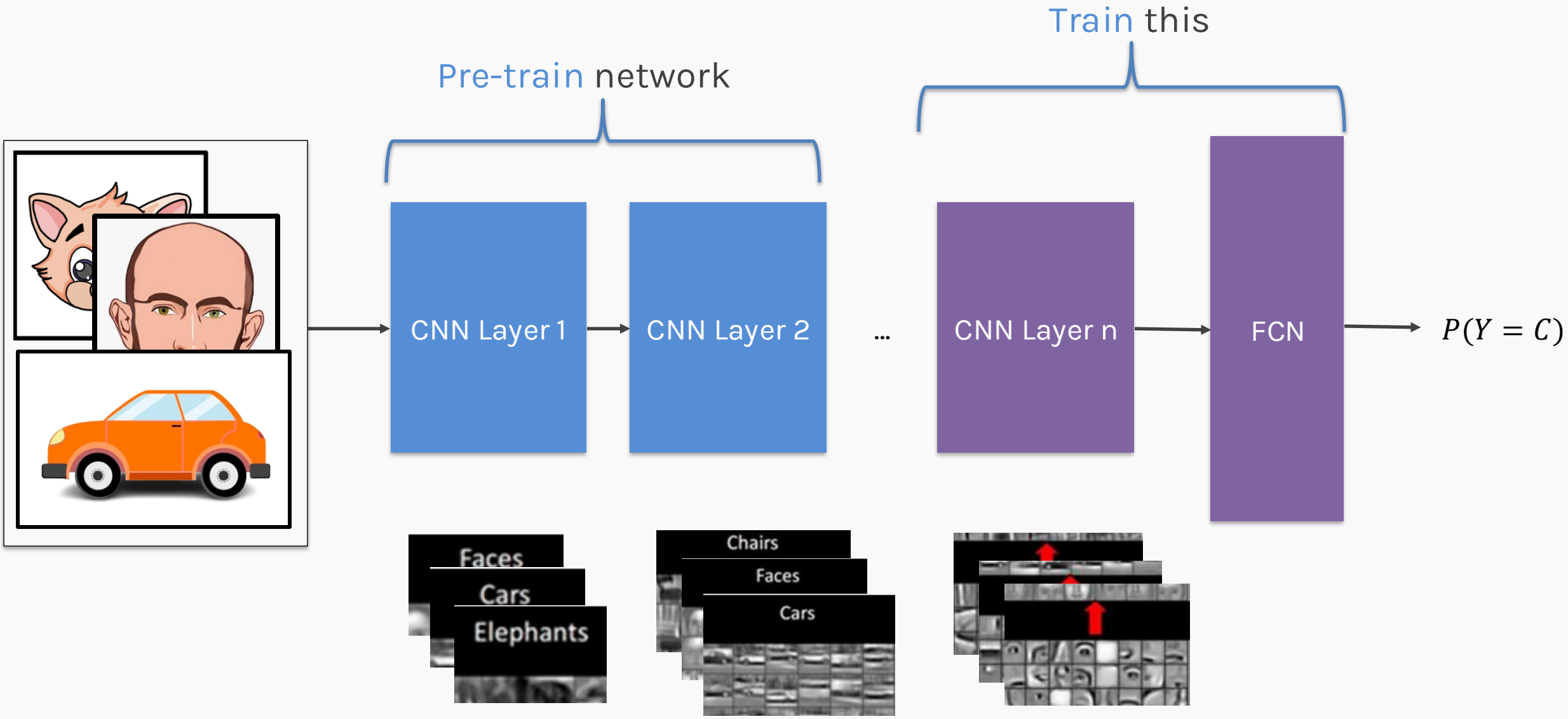


CNN Layer 1 → CNN Layer 2 → ... → CNN Layer n → FCN → $P(Y = C)$



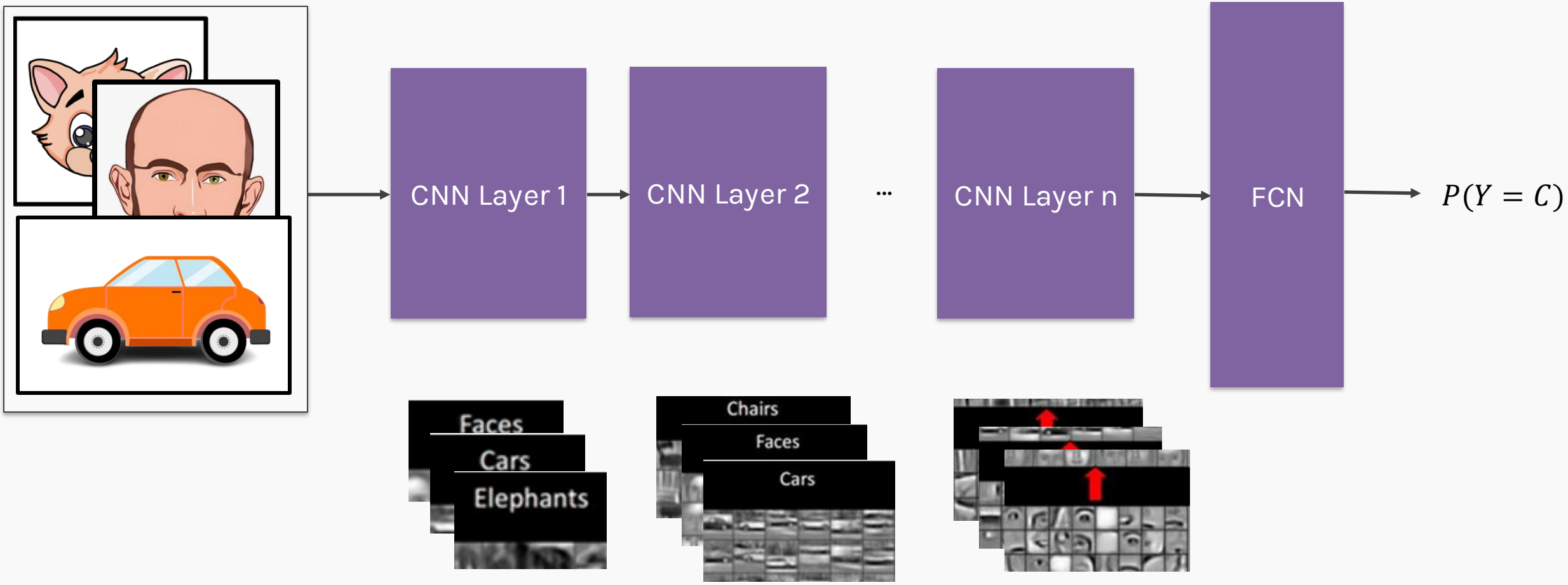


Transfer Learning Strategies



Transfer Learning Strategies: FINE TUNING

Train everything but start with weights that are trained already



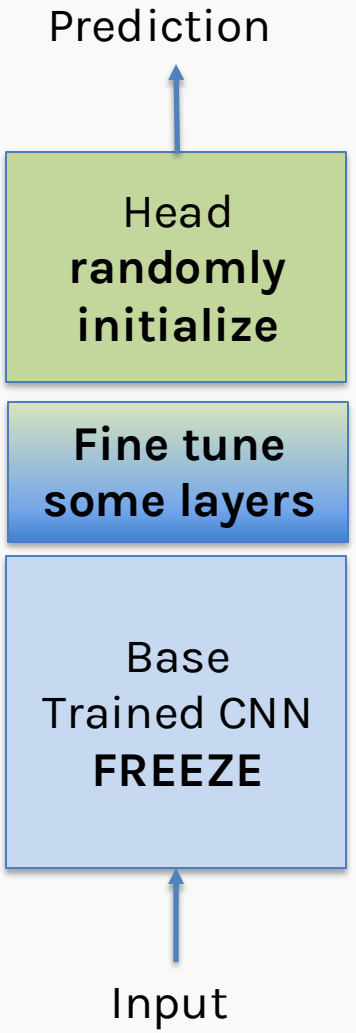
Procedure for Fine-tuning

1. Freeze the convolutional base.

2. First train the fully connected head, keeping the convolutional base fixed.

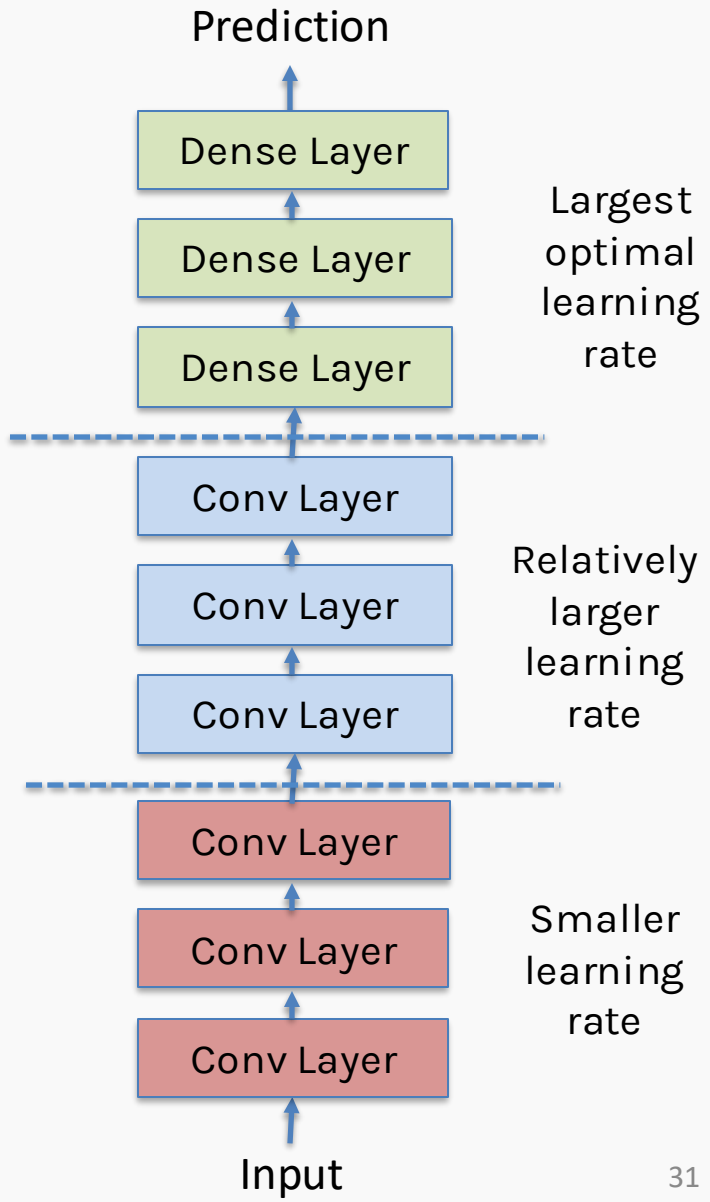
3. Unfreeze some "later" layers in the base net and now train the base net and FC net together.

Since you are now in a better part of the loss surface already, gradients won't be terribly high, but we still need to be careful. Often we use a **very low learning rate**.



Procedure for Fine-tuning: Differential Learning Rates

- The general idea to fine tune is to train different layers at different rates.
- In the pretrained model, the layers closer to the input are more likely to have learned more general features. Thus, we don't want to change them much. Therefore, each "earlier" layer or layer group can be trained at 3x-10x smaller learning rate than the next "later" one.
- Moreover, a low learning rate can take a lot of time to train on the "later" layers as they are learning more complex features.
- One could even train the entire network again this way until we overfit and then step back some epochs.



Thank you