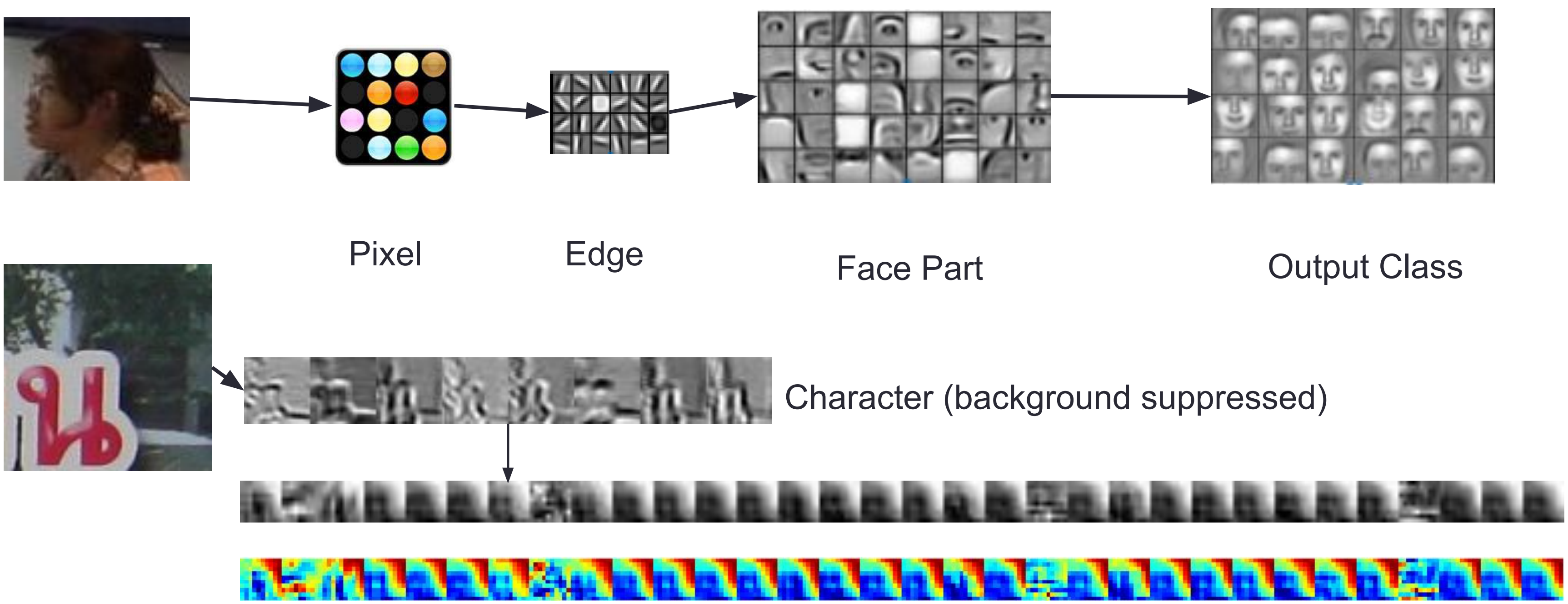
Transfer learning

Deep learning as a feature extractor

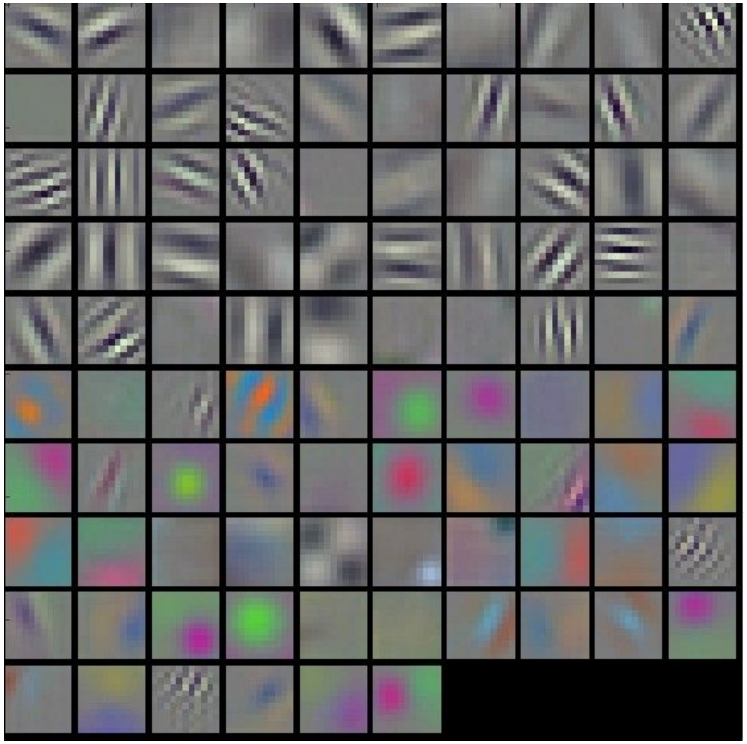
Convolution Neural Network (CNN)

- Hierarchy of representations with increasing level of abstraction
- Each stage is a kind of trainable feature transform
- Image recognition: Pixel →edge → texture → part → object



What does the convolution learn

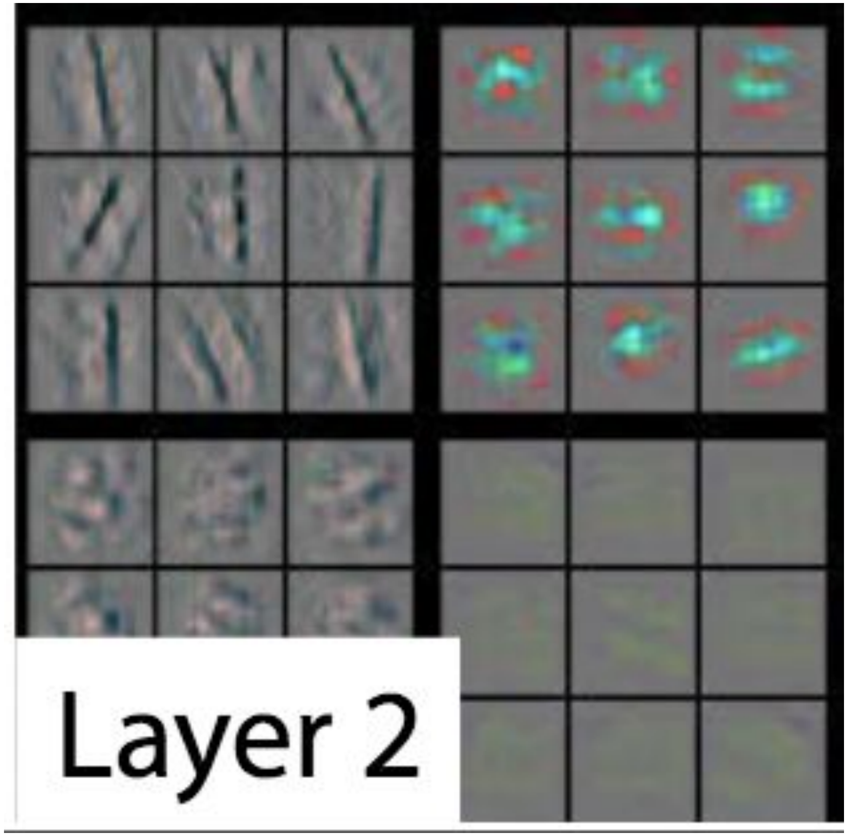
- Learning patterns



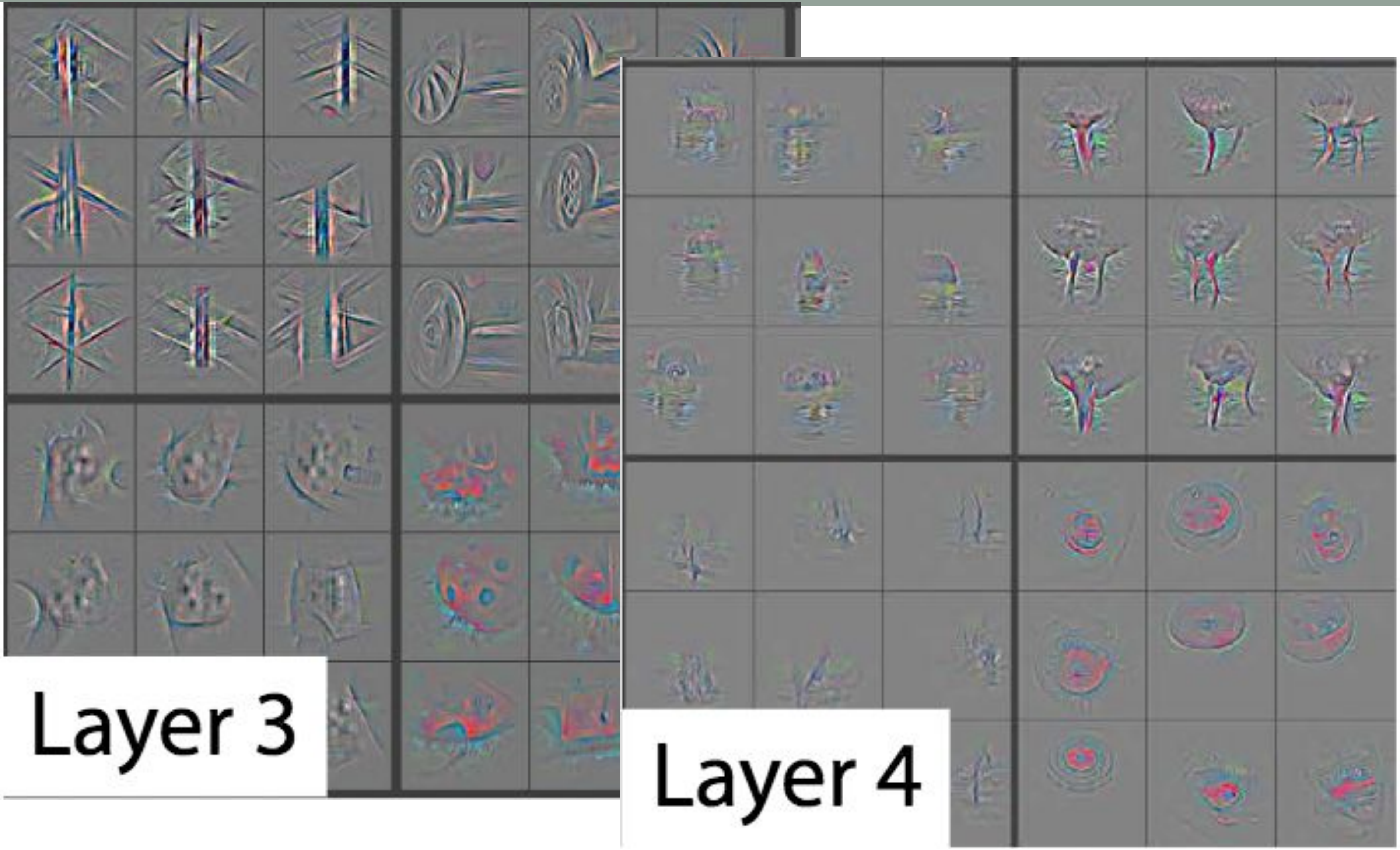
http://cs231n.github.io/understanding-cnn/

Higher layer captures higher-level concepts

Layer 1



https://arxiv.org/pdf/1311.2901.pdf


Layer 3
Layer 4

Have you ever seen this creature before?
Can you guess whether it is land or water animal?

You can transfer your knowledge in the past

Transfer learning

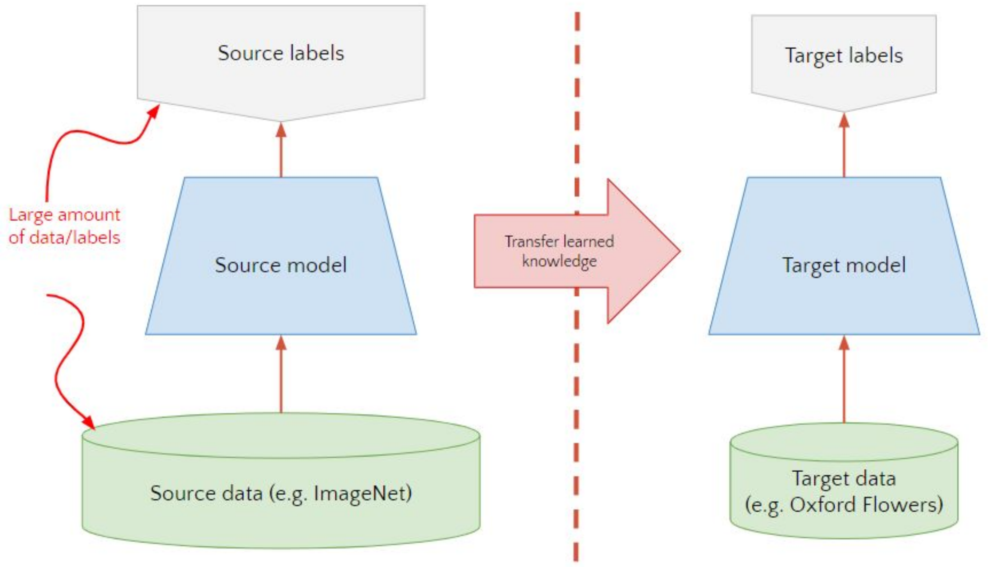
Myth: you can't do deep learning unless you have a million labelled examples for your problem.

Reality:

- *You can **transfer** learned representations from **a related task***
- You can train on a nearby **surrogate objective** for which it is easy to generate labels

Transfer learning (basics)

- We know networks captures good representations
- Can we use it for other tasks?
- Use trained networks to initialize a new network for a different task.
- Re-train the network using SGD on new data.



For CV tasks, we call the pre-trained network backbones

Transfer learning idea

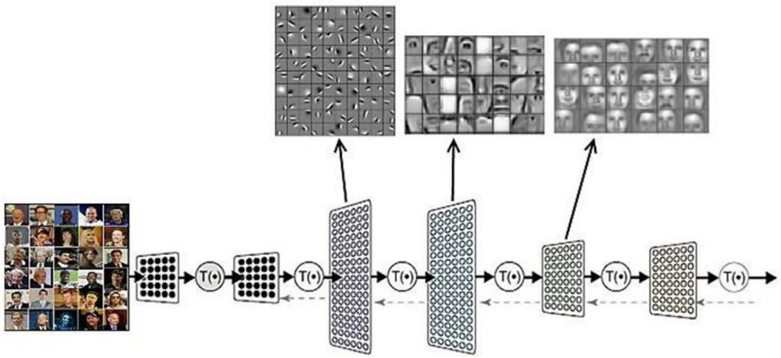
Instead of training a deep network from scratch for your task, you can

- Take a network trained on **a different domain** for a **different source task**
- **Adapt (fine-tune)** it for your domain and your **target task**

This lecture will talk about how to do this.

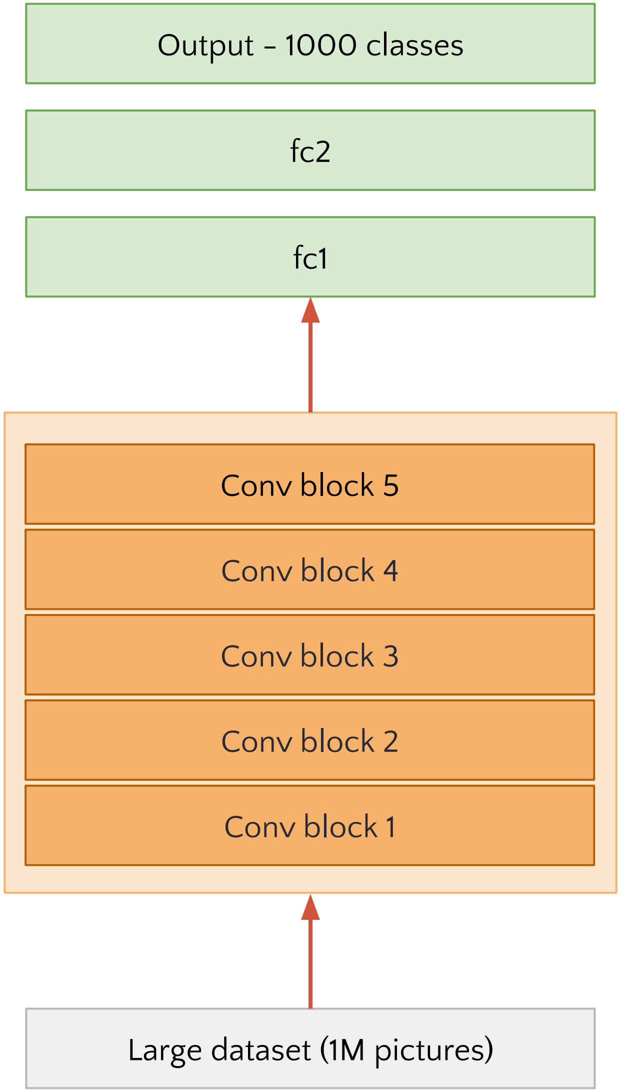
Variations:

- Different domain, same task
- Different domain, different task

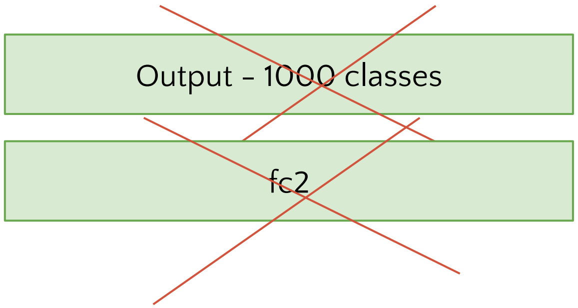


Transfer learning

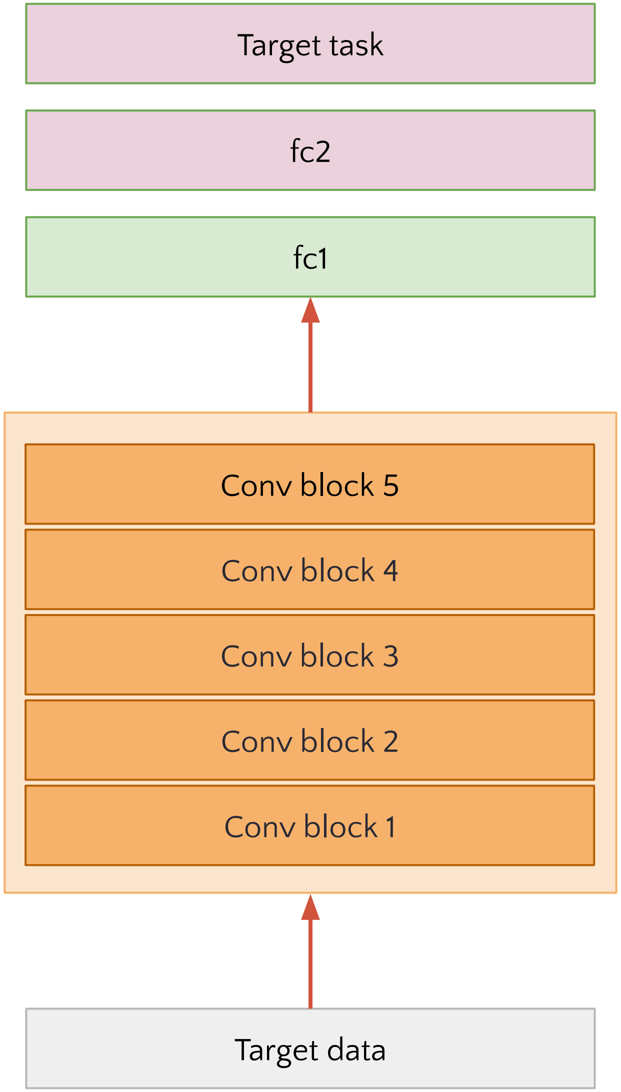
1. A model is trained on large dataset
   Ex ImageNet



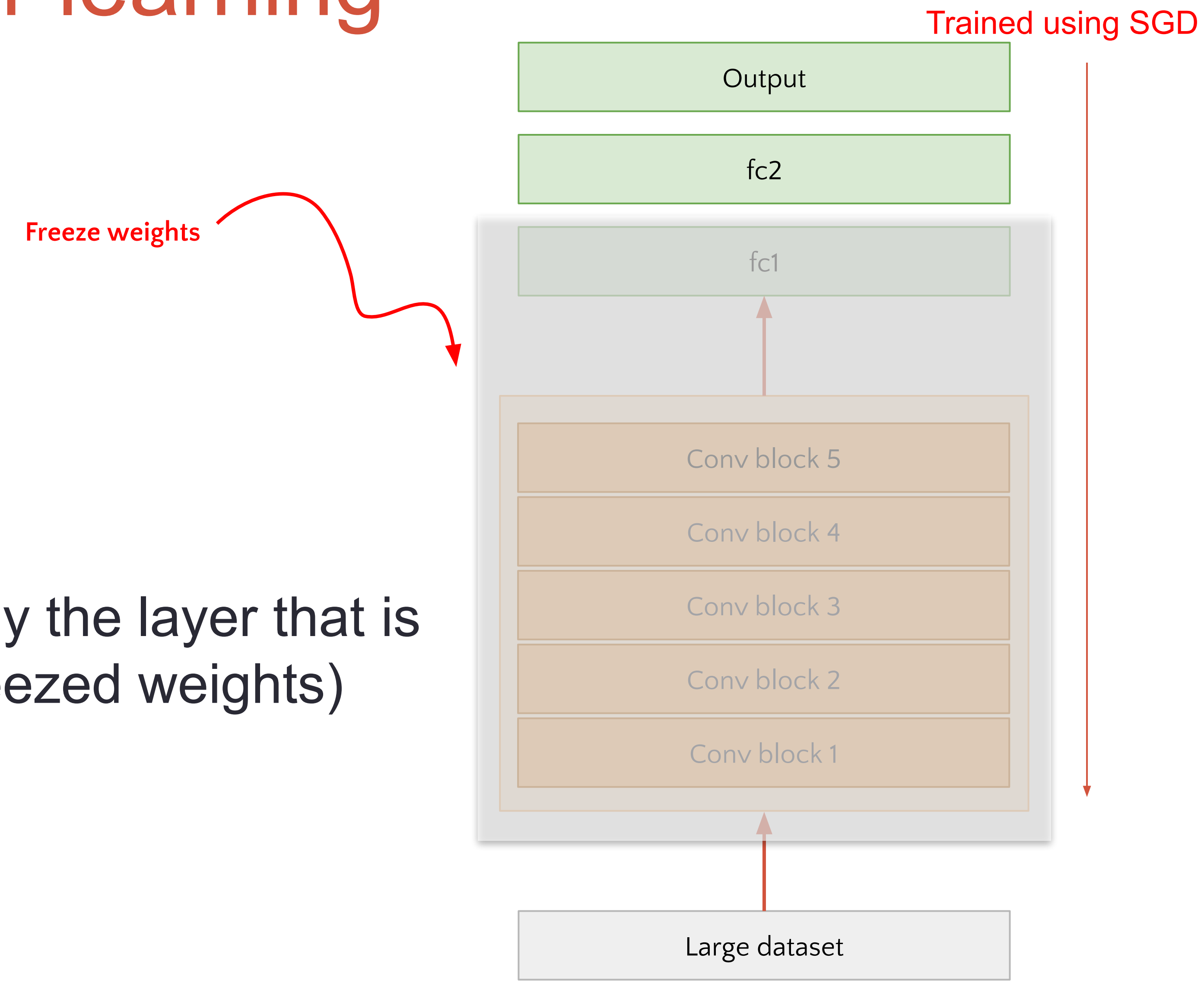
Transfer learning



2. Replace the top layers with target task



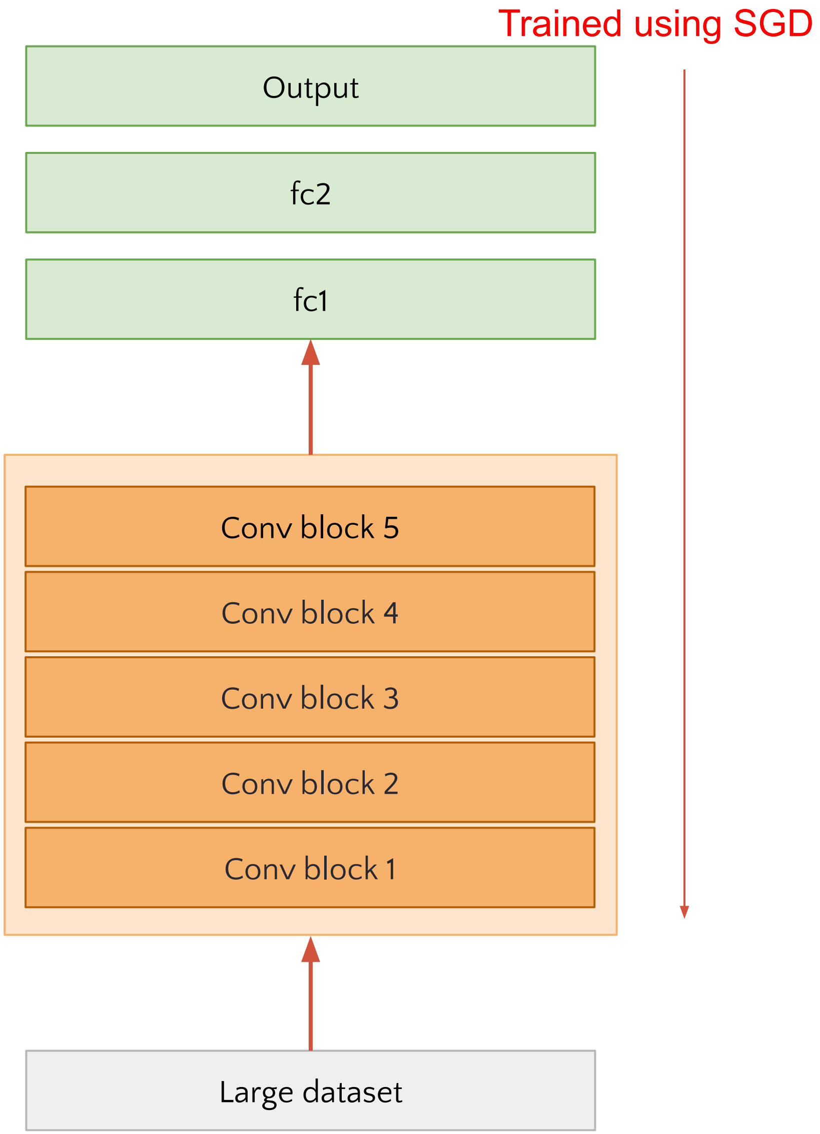
Transfer learning



3A. Train only the layer that is replaced (freezed weights)

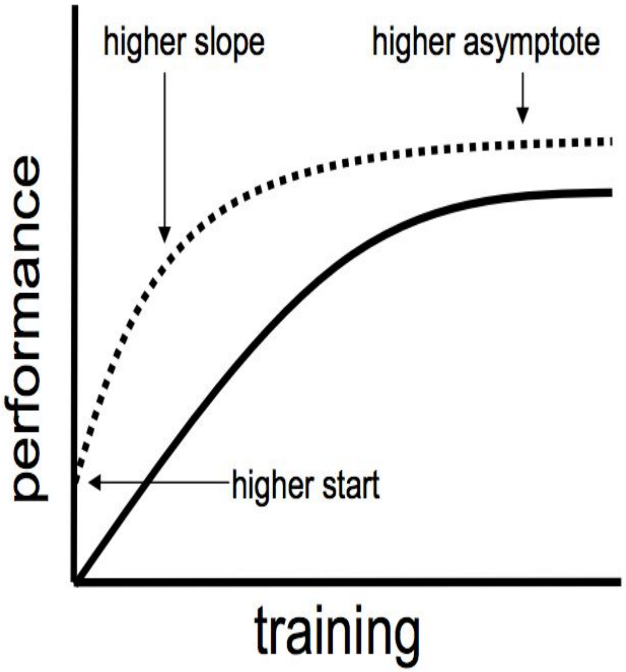
Transfer learning

3B. Train all layers (unfreezed weights)



Benefits to transfer learning

1. Higher start
2. Higher slope (converge faster)
3. Higher asymptote (if small data)

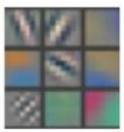


Chapter 11: Transfer Learning, Handbook of Research on Machine Learning Applications, 2009.

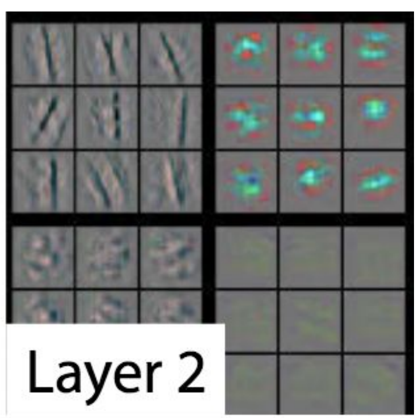
Layers

Lower layers: more general

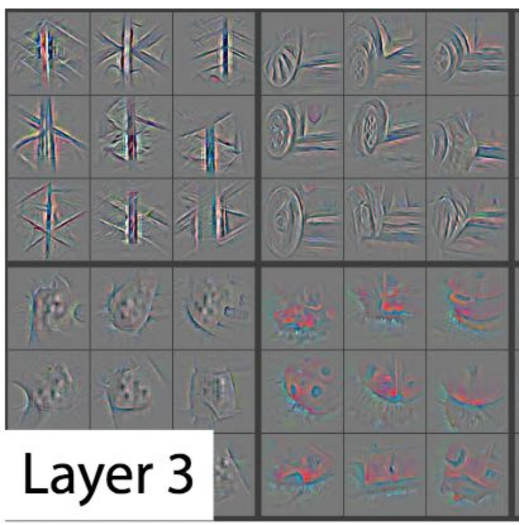
Higher layers: more specific



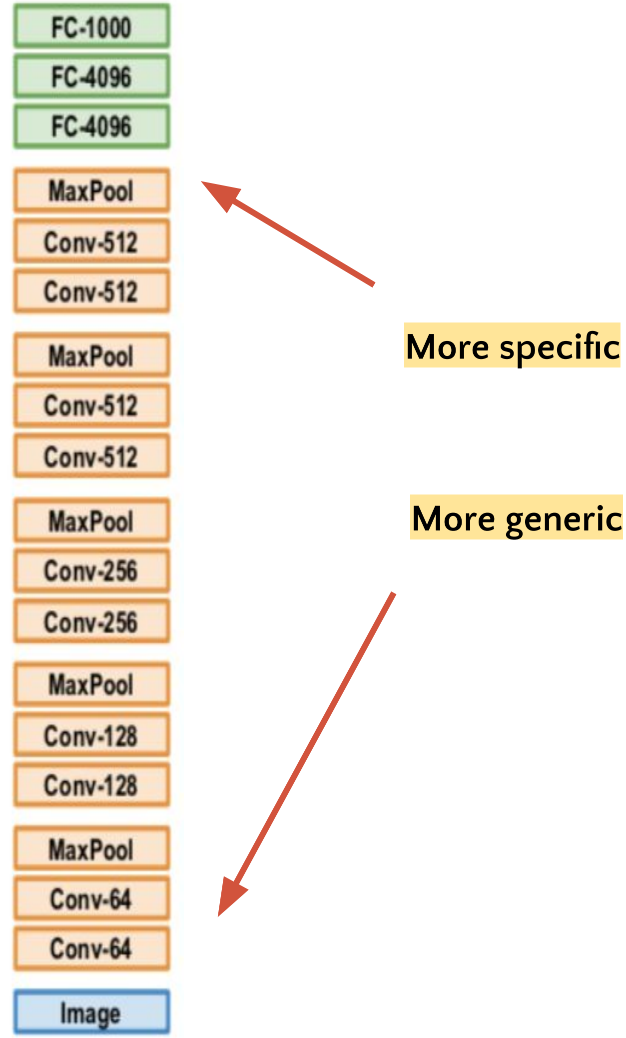
Layer 1



Layer 2



Layer 3



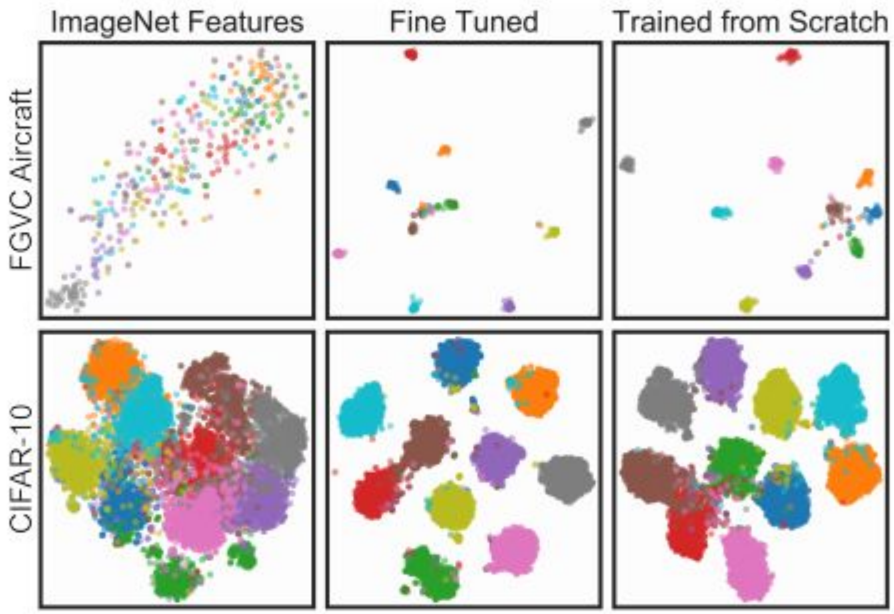
Domains

Similar domain can transfer easier

Fine-tuning (adaptation) is crucial

Aircraft images
Different from ImageNet

Natural images
Similar to ImageNet

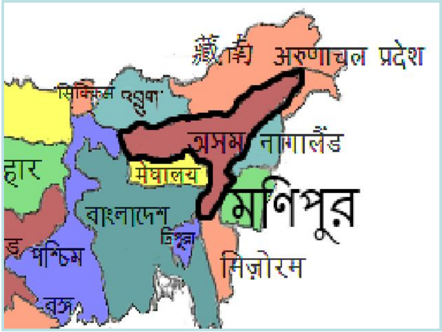


Kornblith et al., Do Better ImageNet Models Transfer Better?, arxiv 2018 https://arxiv.org/abs/1805.08974

Transfer learning in speech recognition

Assamese and Bengali language

Bengali



Assamese

Bengali	**WER**
No pre-training (10 hr Bengali)	71.8%
No pre-training (60 hr Bengali)	64.5%
Transfer learning (10 hr Assamese -> 10 hr Bengali)	66.0%
Transfer learning (60 hr Assamese -> 10 hr Bengali)	64.6%
+ Adaptation	63.7%
Transfer learning (60 hr Bengali -> 10 hr Bengali)	61.6%

Closer domain is better

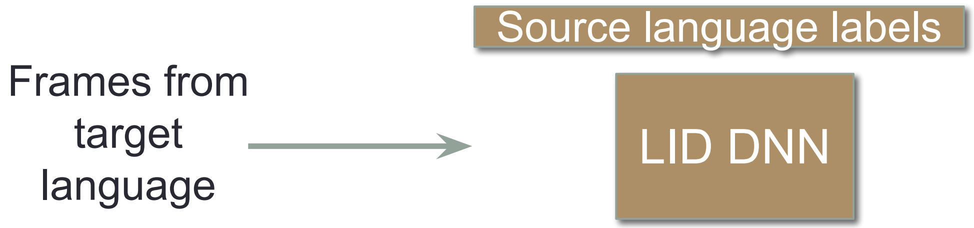
- Pre-train on 60 hours of data, and adapt on languages with 10 hours of data
- Numbers in blue is for 10->10 hours (baseline)

		Pretrain on			
		Bengali	Assamese	Lao	Turkish
Adapt to	Bengali	66.0	63.8	65.1	64.2
	Assamese	61.2	65.2	62.9	62.1
	Lao	59.8	60.1	62.3	60.0
	Turkish	61.8	63.1	63.3	63.9

- Transfer learning always improves performance.
- Similar languages perform better on transfer learning

Language Identification for data selection

- Train a classifier (LID) on source languages to predict the language given input frames
- Compute posteriors of the target language data using that classifier
- The best language for the target language should have the highest average posterior score



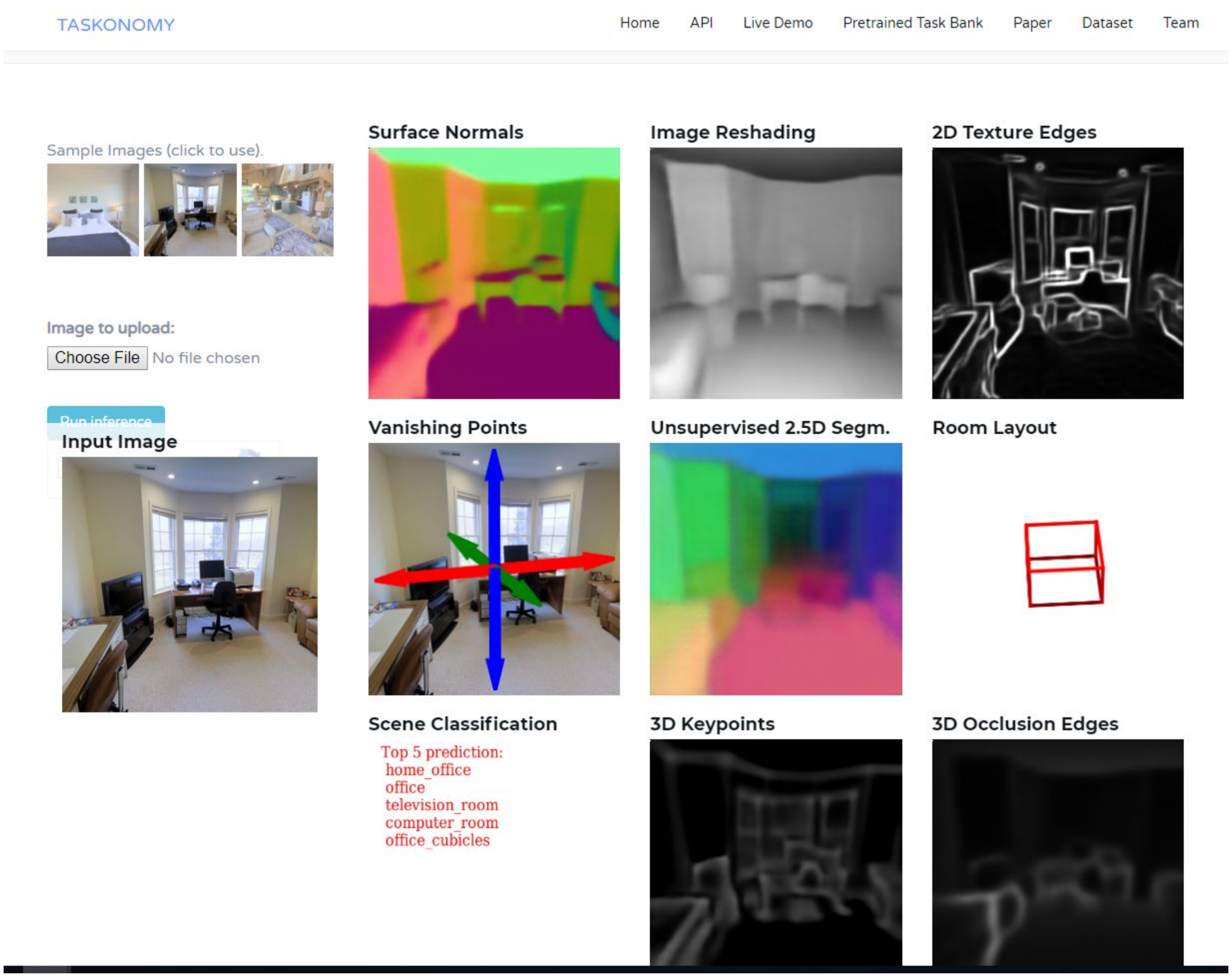
Predicting the best language

		Pretrain on			
		Bengali	Assamese	Lao	Turkish
Adapt to	Bengali	66.0	63.8	65.1	64.2
	Assamese	61.2	65.2	62.9	62.1
	Lao	59.8	60.1	62.3	60.0
	Turkish	61.8	63.1	63.3	63.9

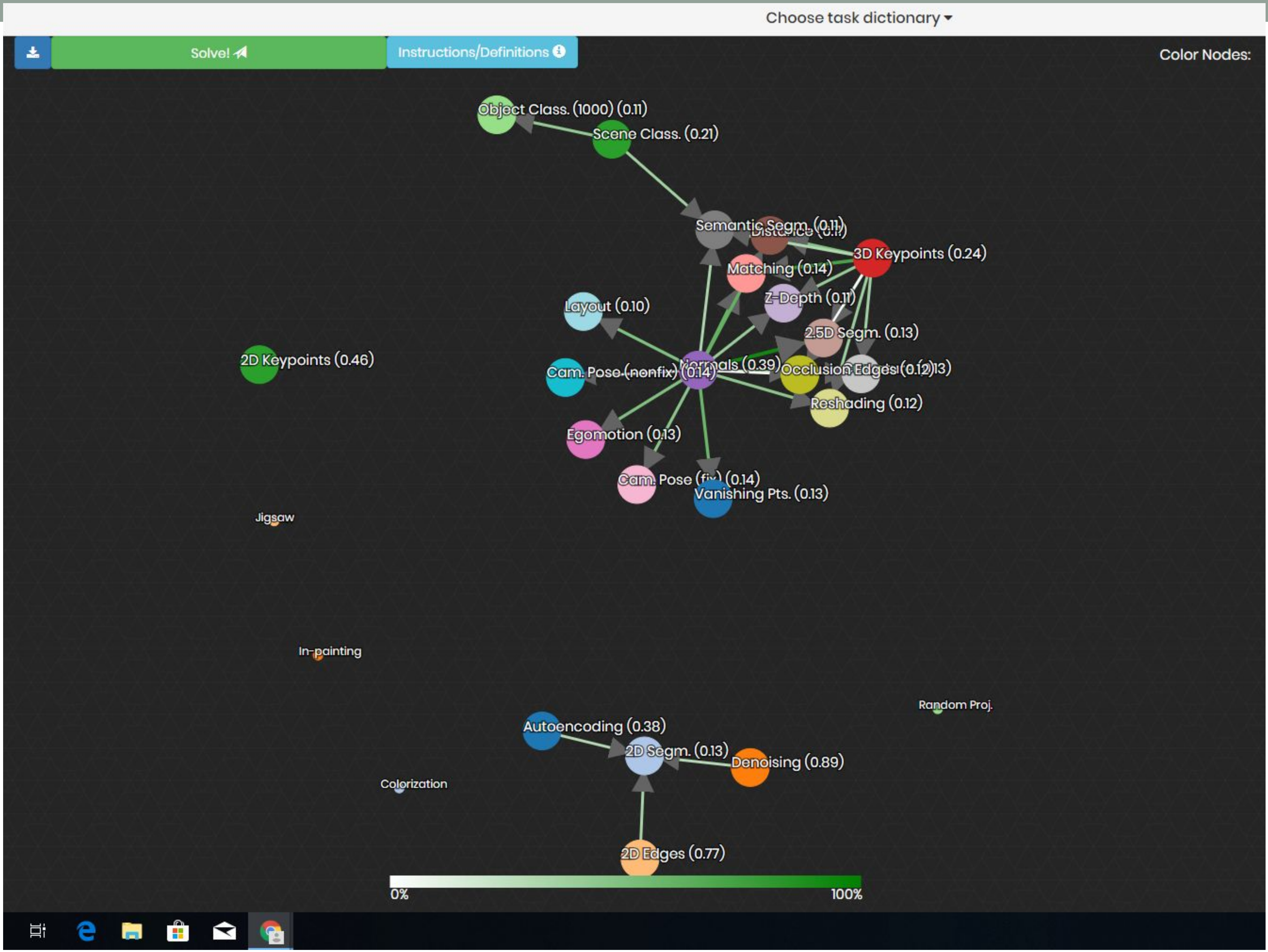
		Language prediction score			
		Bengali	Assamese	Lao	Turkish
Input frames	Bengali	0.57	0.21	0.09	0.13
	Assamese	0.21	0.57	0.11	0.11
	Lao	0.08	0.11	0.71	0.10
	Turkish	0.13	0.12	0.10	0.65

The LID scores correspond to the best language to use most of the time.

Which task to transfer?



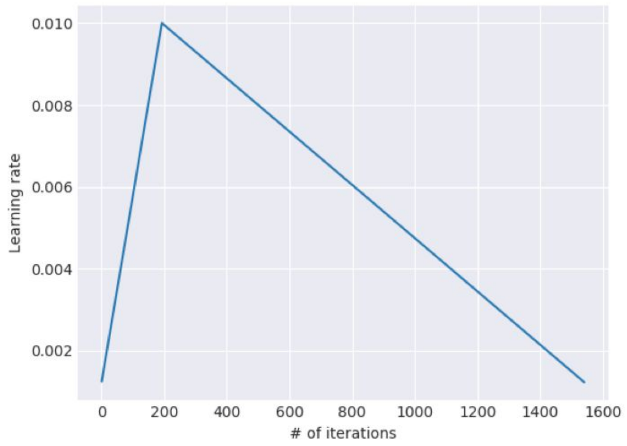
http://taskonomy.stanford.edu/tasks/


Choose task dictionary
Solve!
Instructions/Definitions
Color Nodes:
Object Class. (1000) (0.11)
Scene Class. (0.21)
Semantic Segm. (0.11)
3D Keypoints (0.24)
Matching (0.14)
Z-Depth (0.11)
Layout (0.10)
2.5D Segm. (0.13)
2D Keypoints (0.46)
Cam. Pose (nonfix) (0.14)
Occlusion Edges (0.12)
Reshading (0.12)
Egomotion (0.13)
Vanishing Pts. (0.13)
Jigsaw
In-painting
Random Proj.
Autoencoding (0.38)
2D Segm. (0.13)
Denoising (0.89)
Colorization
2D Edges (0.77)
0%
100%

Adaptation tricks

Triangle learning rate

At the start some of the model is randomly initialized. Cannot trust the gradient



Discriminative fine-tuning

Instead of using the same learning rate for all layers of the model, discriminative fine-tuning allows us to tune each layer with different learning rates.

- Layers that are closer to inputs → large learning rate
- Layers that are closer to outputs → small learning rate

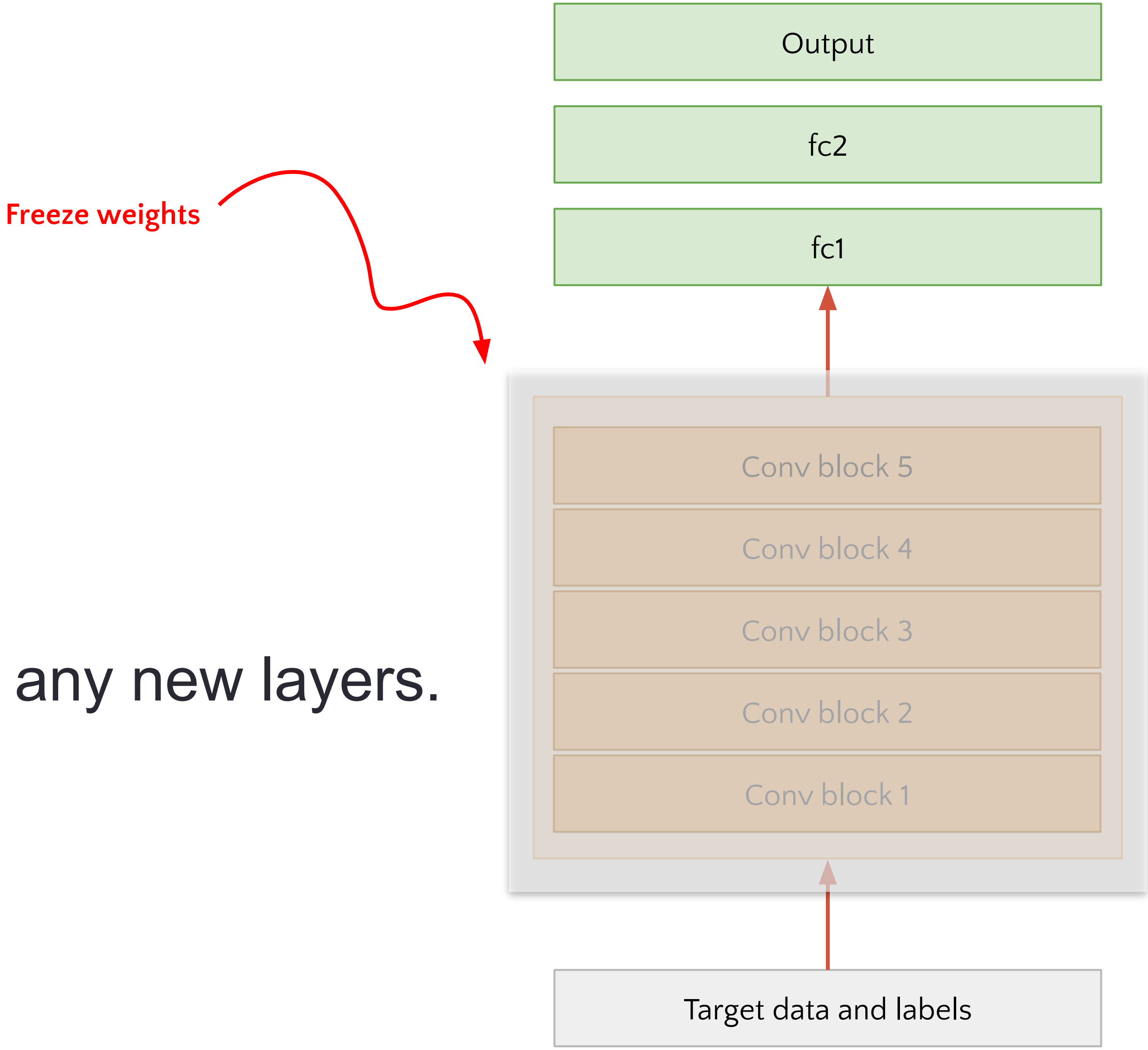
Advance adaptation ideas

- Chain-taw
  - Freeze the old layers, train the new weights for a bit

1. fine-tunes any new layers.
2. fine-tunes each layer individually of base model starting from the first to the last.
3. the entire model is trained with all layers.

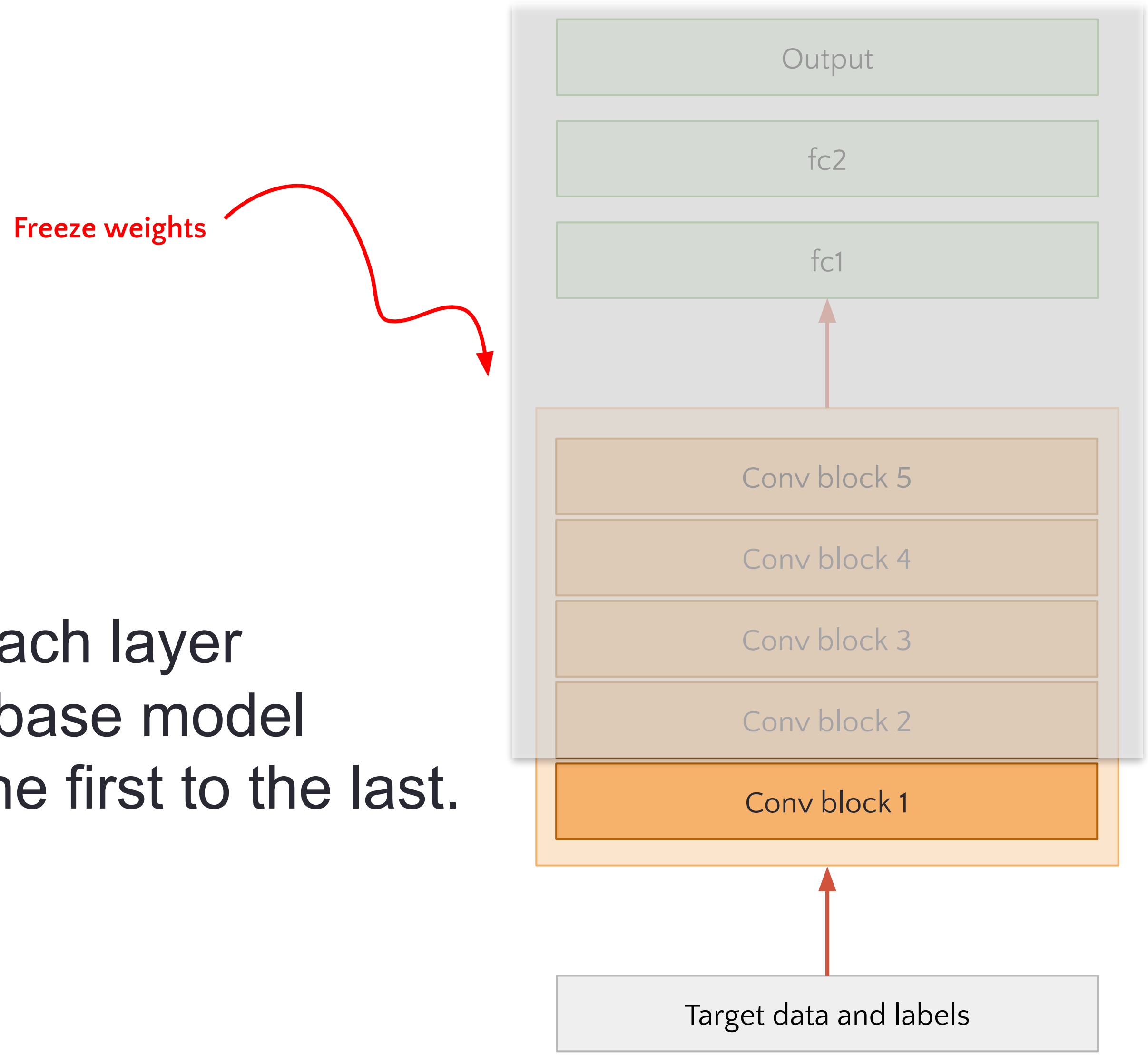
Felbo et al., **Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm,** arxiv 2017, https://arxiv.org/pdf/1708.00524.pdf

Chain-taw



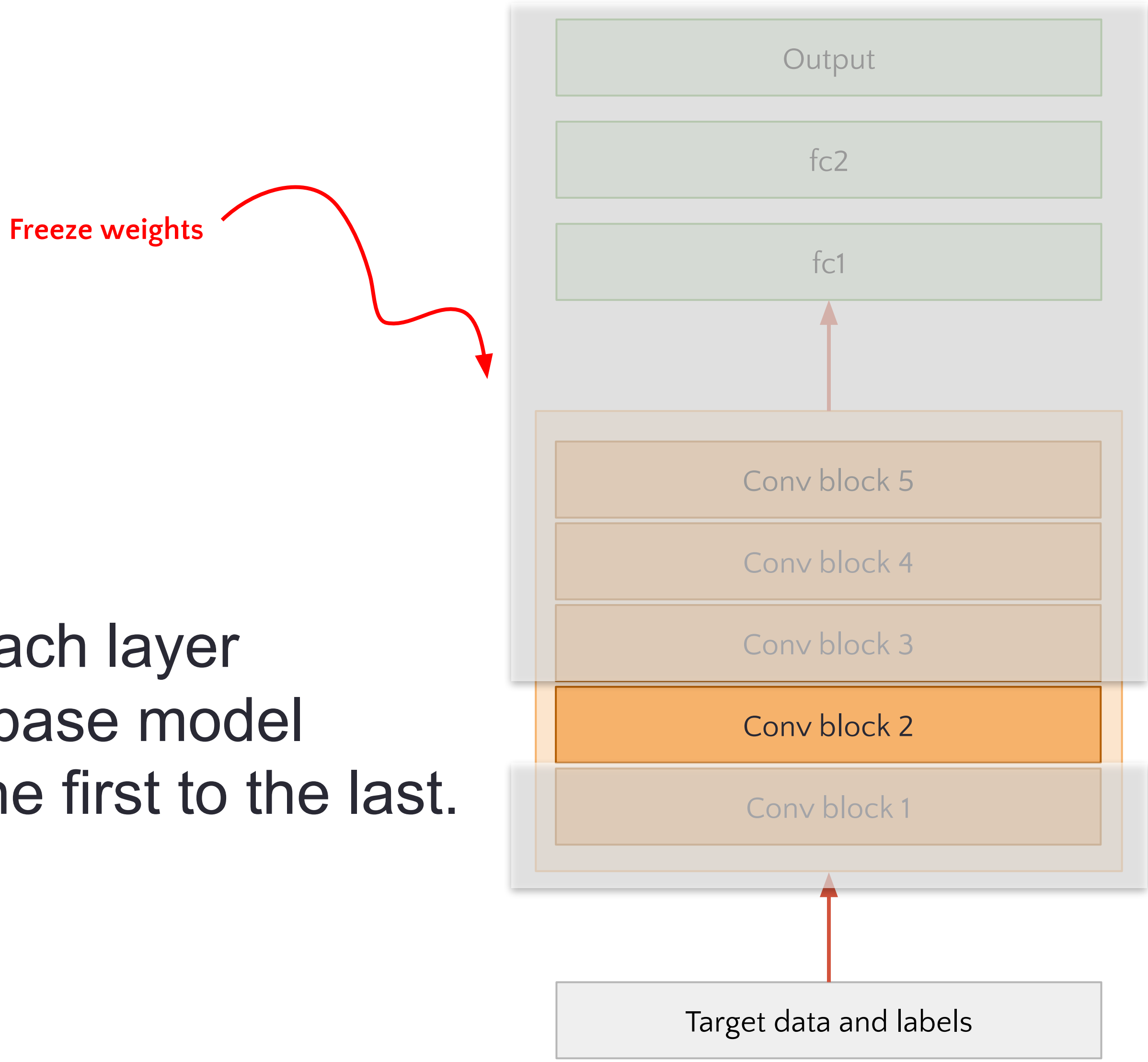
1. fine-tunes any new layers.

Chain-taw



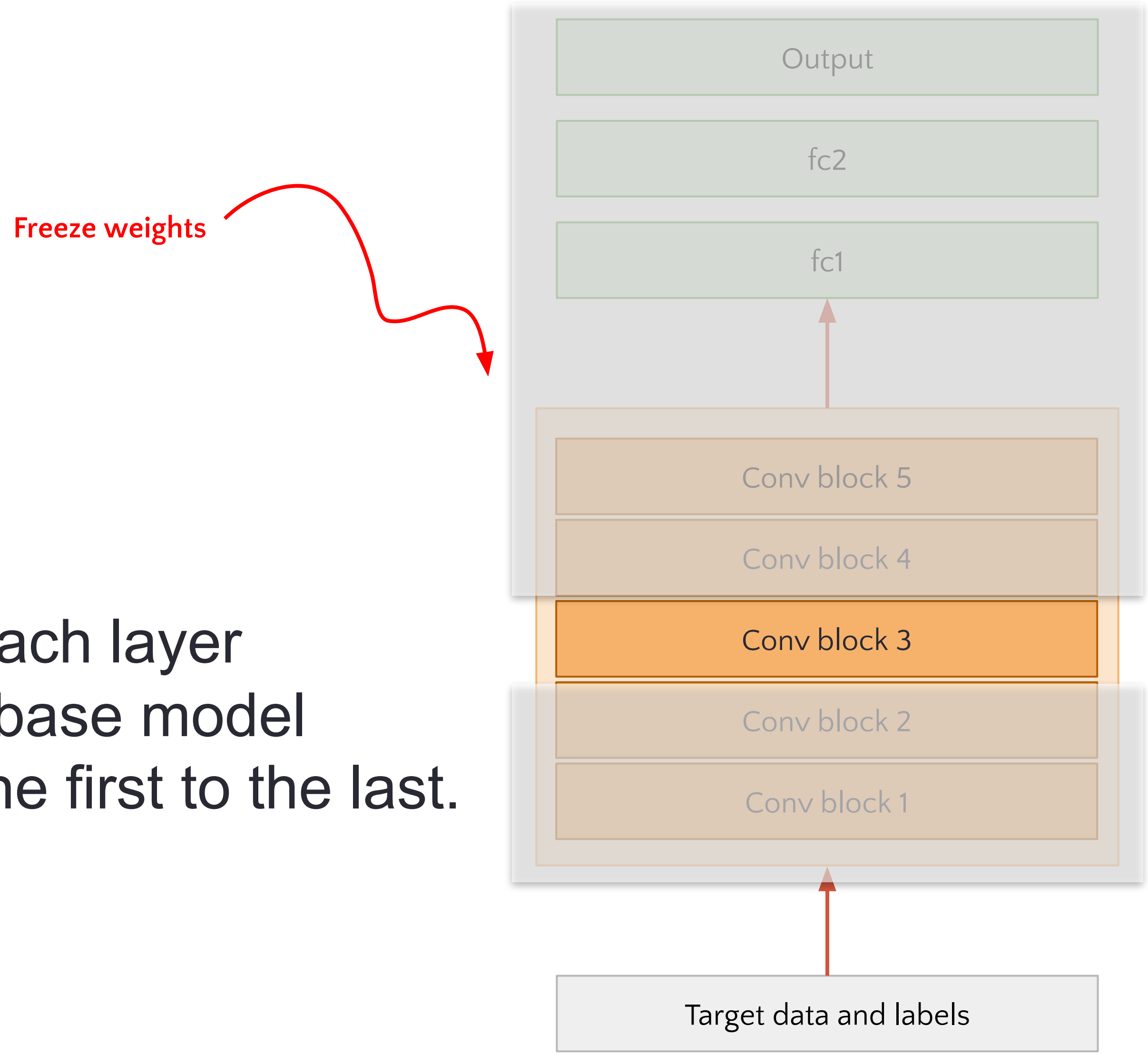
2. fine-tunes each layer individually of base model starting from the first to the last.

Chain-taw



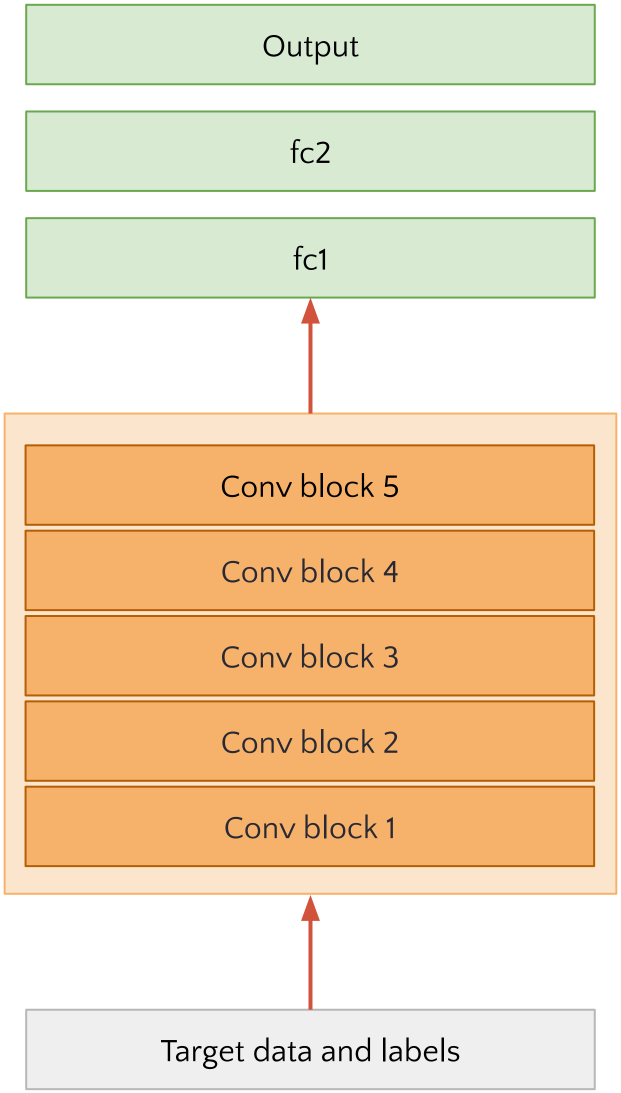
2. fine-tunes each layer individually of base model starting from the first to the last.

Chain-taw



2. fine-tunes each layer individually of base model starting from the first to the last.

Chain-taw



3. the entire model is trained with all layers.

Lab

Housing price prediction

Structured vs unstructured data

Trainset_Homedottech_Hackathon.csv (1.18 MB)   20 of 36 columns   Views

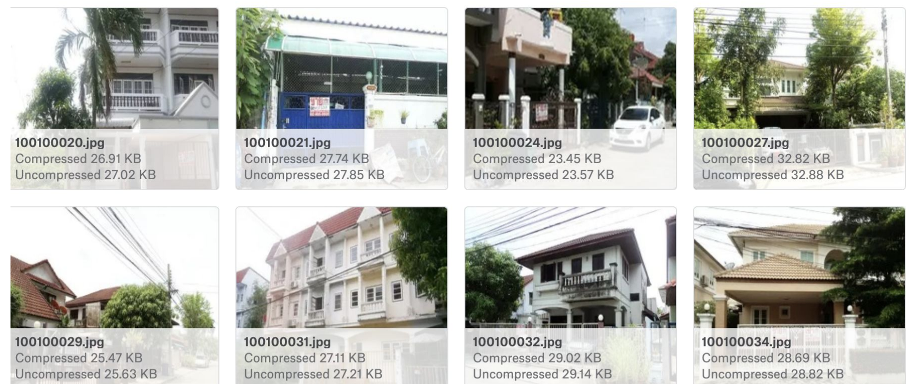
	ListingID	ListingInfoID	ListingTypeID	BuildingNameTH	MaxRentPrice
	ID of houses	ID of houses information	1 = For Sale, 2 = For Rent, 3 = Both, 5 = Ready to move in (new property)		
	100m – 110m	1.00b – 1.10b	1 – 5	[null] 91% ทาวน์เฮ้าส์3ชั้น 0% Other (387) 9%	0 – 299m
1	110053340	1100053070	1		5000000.0
2	100500412	1005004121	1		
3	100100069	1001000691	1		
4	110025306	1100025175	1		
5	110039244	1100039089	1		
6	101400073	1014000731	1		
7	110049848	1100049613	1	ม.นันทวัน วัชรพล	
8	110048432	1100048229	1		
9	110002794	1100002715	1		
10	110013078	1100012954	1		
11	110053026	1100052756	2		800000.0
12	110048599	1100048396	1		5500000.0

A combination of non deep learning method (XGBoost) + Deep learning
XGBoost - good for tabular data
Deep learning - good for unstructured data

Structured vs unstructured data

Trainset_Homedottech_Hackathon.csv (1.18 MB) | 20 of 36 columns | Views

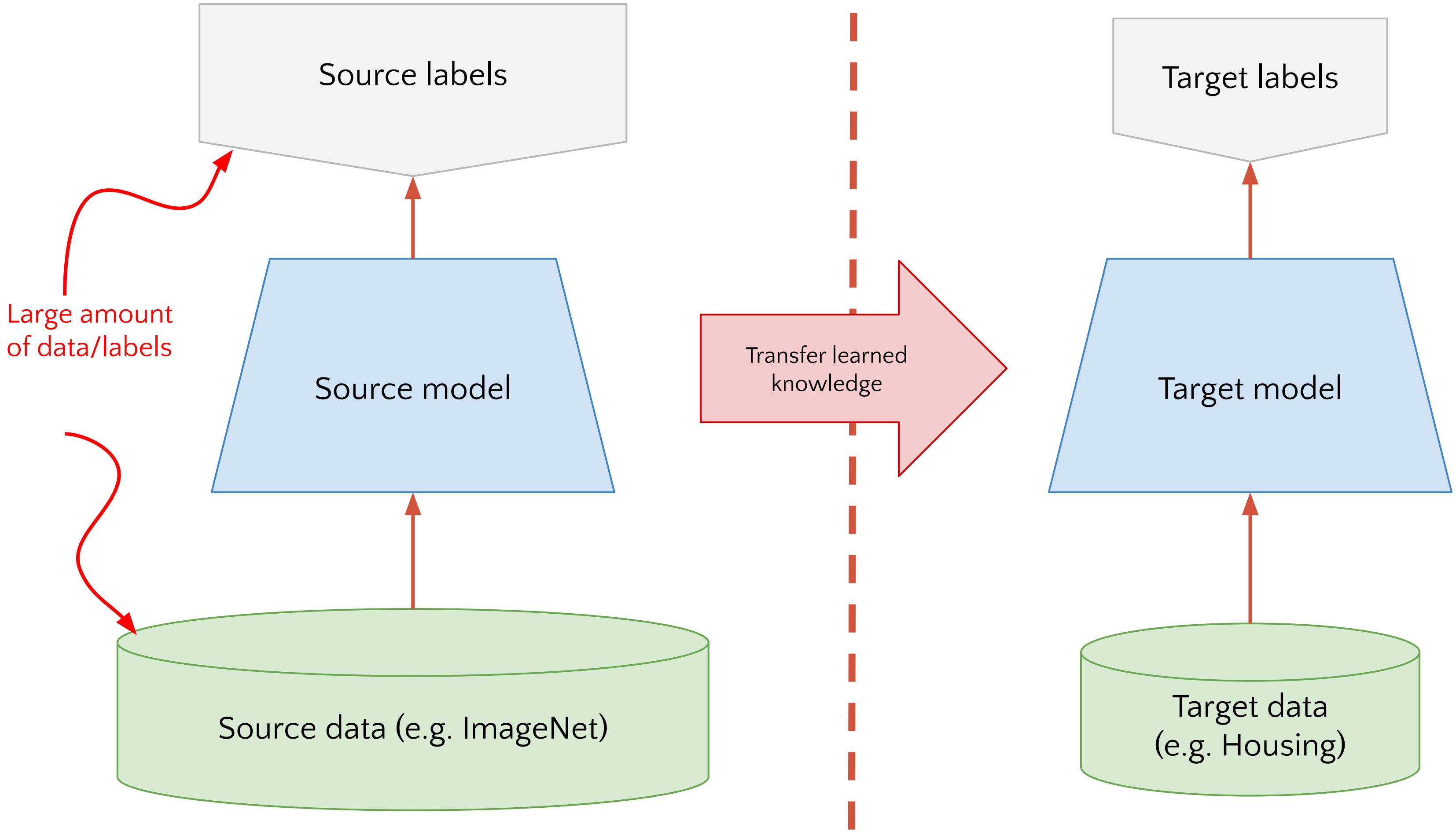
	ListingID	ListingInfoID	ListingTypeID	BuildingNameTH	MaxRentPrice
	ID of houses	ID of houses information	1 = For Sale, 2 = For Rent, 3 = Both, 5 = Ready to move in (new property)		
	100m 110m	1.00b 1.10b	1 5	[null] 91% ทาวน์เฮ้าส์3ชั้น 0% Other (387) 9%	0 299m
1	110053340	1100053070	1		5000000.0
2	100500412	1005004121	1		
3	100100069	1001000691	1		
4	110025306	1100025175	1		
5	110039244	1100039089	1		
6	101400073	1014000731	1		
7	110049848	1100049613	1	ม.นันทวัน วัชรพล	
8	110048432	1100048229	1		
9	110002794	1100002715	1		
10	110013078	1100012954	1		
11	110053026	1100052756	2		800000.0
12	110048599	1100048396	1		5500000.0



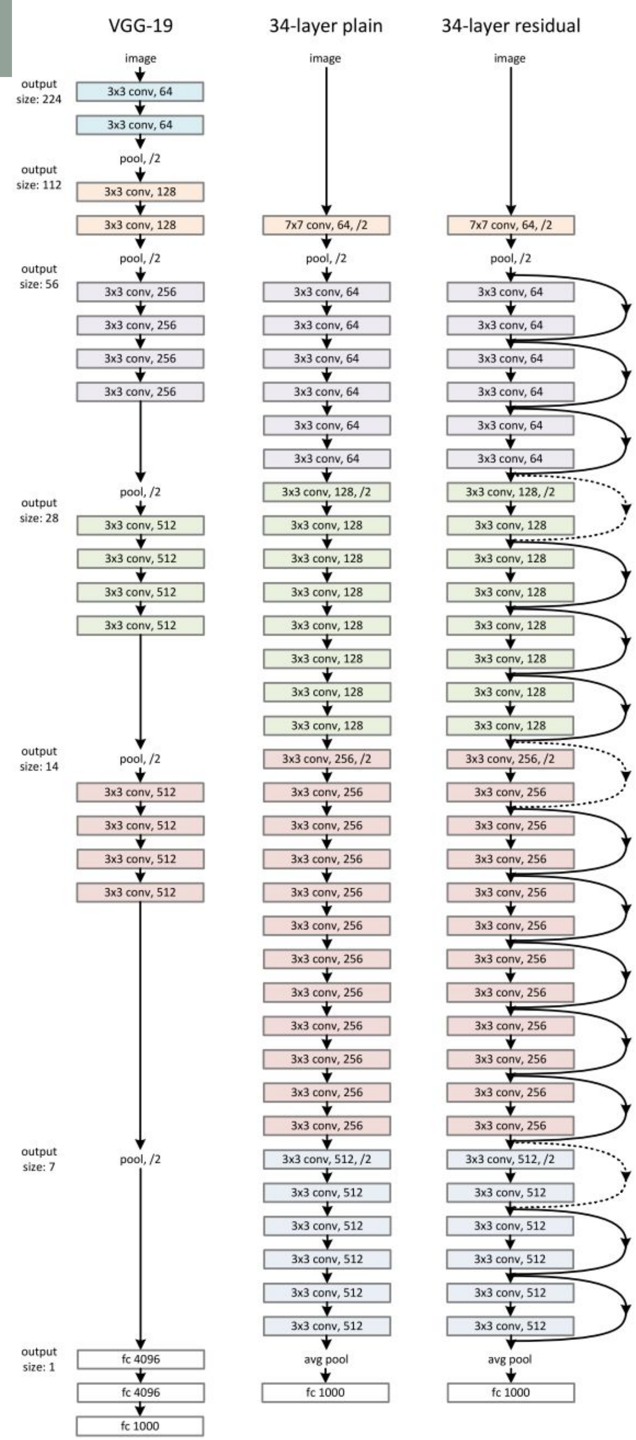
Let's play with XGBoost base model first

https://colab.research.google.com/drive/1N41w5A1mZ5f2bP_VeS7gVO8e6laUmt1_?usp=sharing

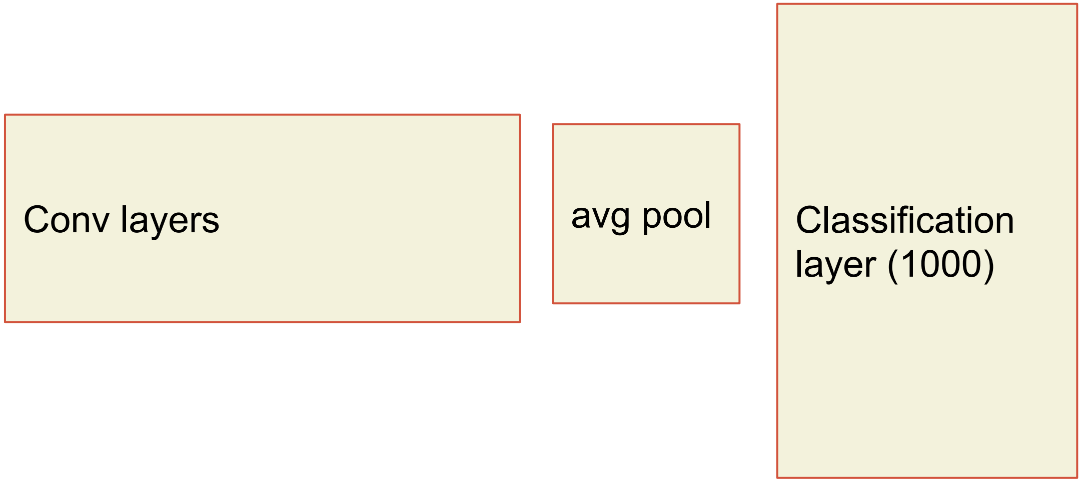
Transfer learning for housing price prediction

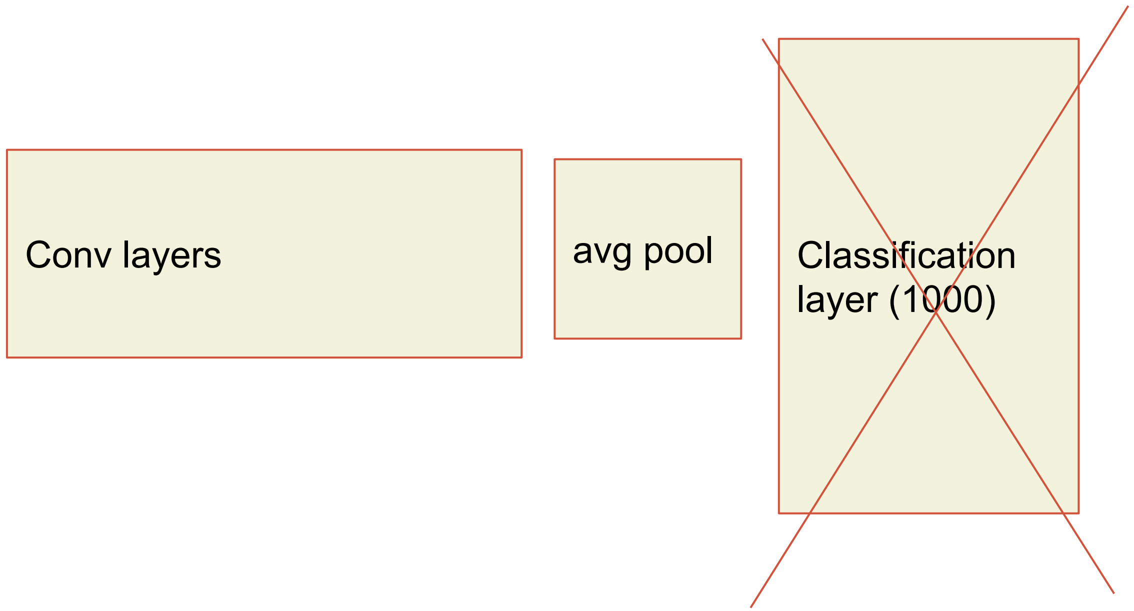


Resnet-34



https://www.kaggle.com/pytorch/resnet34


Conv layers
avg pool
Classification
layer (1000)



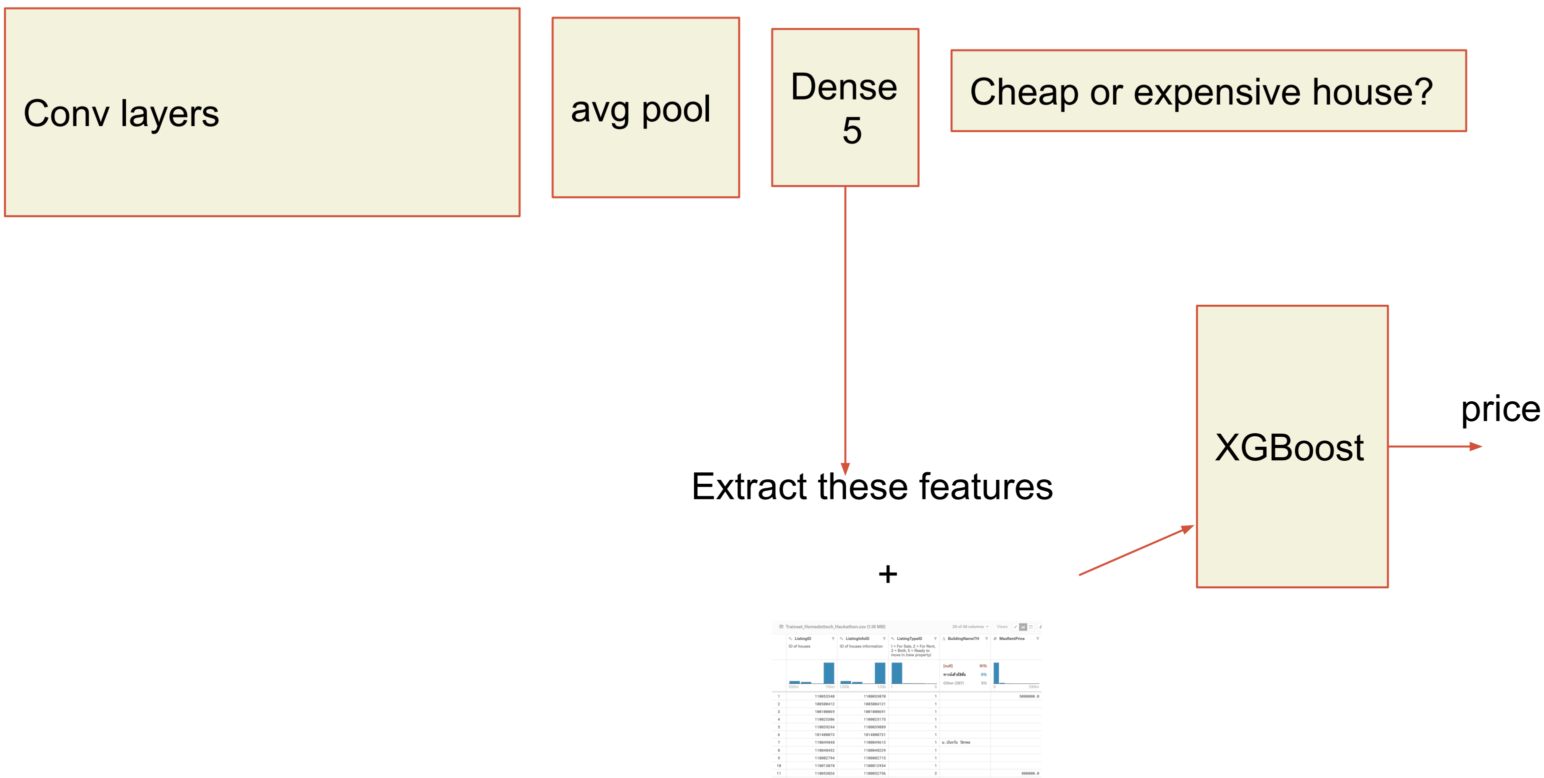
Conv layers
avg pool
Classification
layer (1000)

Conv layers

avg pool

Dense
5

Cheap or expensive house?


Conv layers
avg pool
Dense
5
Cheap or expensive house?
Extract these features
+
XGBoost
price

Summary

You can transfer knowledge from other dataset to help reduce the amount of training data.

More sophisticated fine tuning techniques exist:

Chain-taw, ULMfit, etc. See NVIDIA workshop for more details

https://youtu.be/l8oqxp0up34?t=4410