Deep Learning Crash Course

Hui Xue

Fall 2021

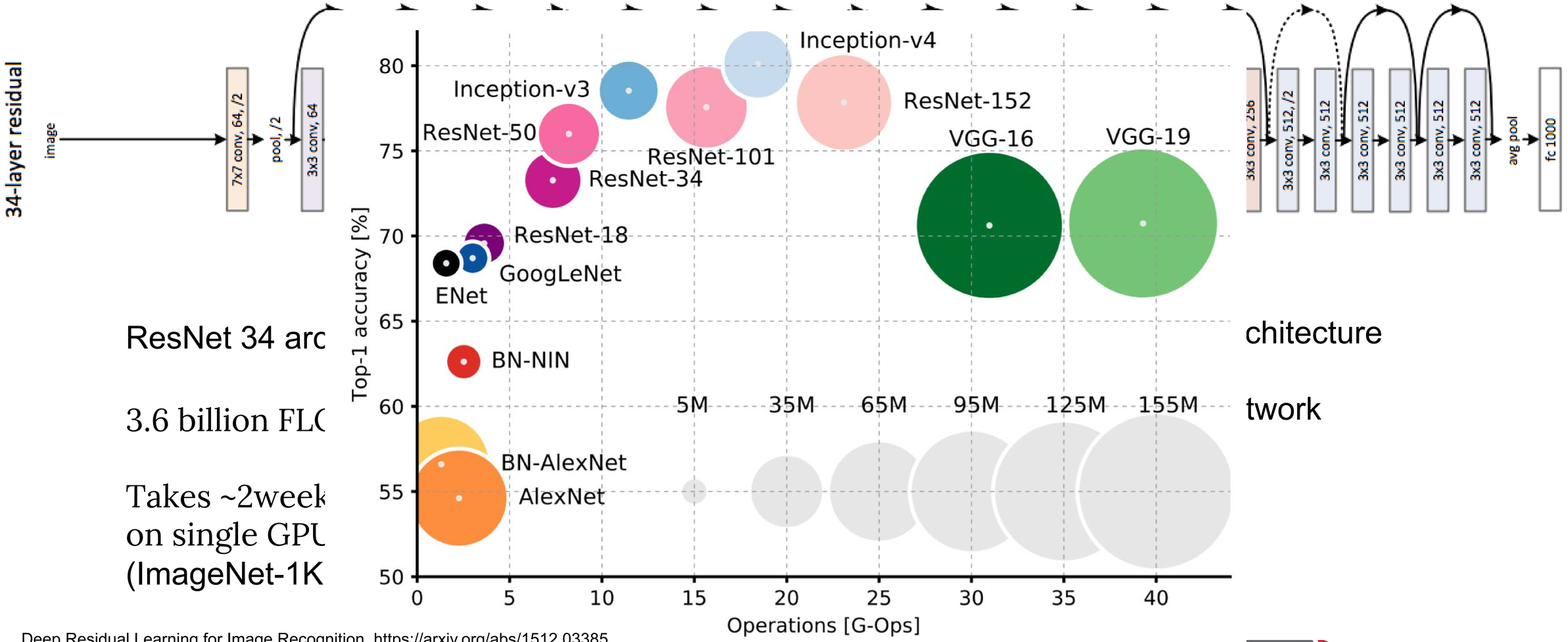
www.deeplearningcrashcourse.org

National Heart, Lung, and Blood Institute

Outline

- Transfer learning
- How Tesla vision works
- Meta Learning
- Contrastive learning

Motivation for “Not” train a whole network



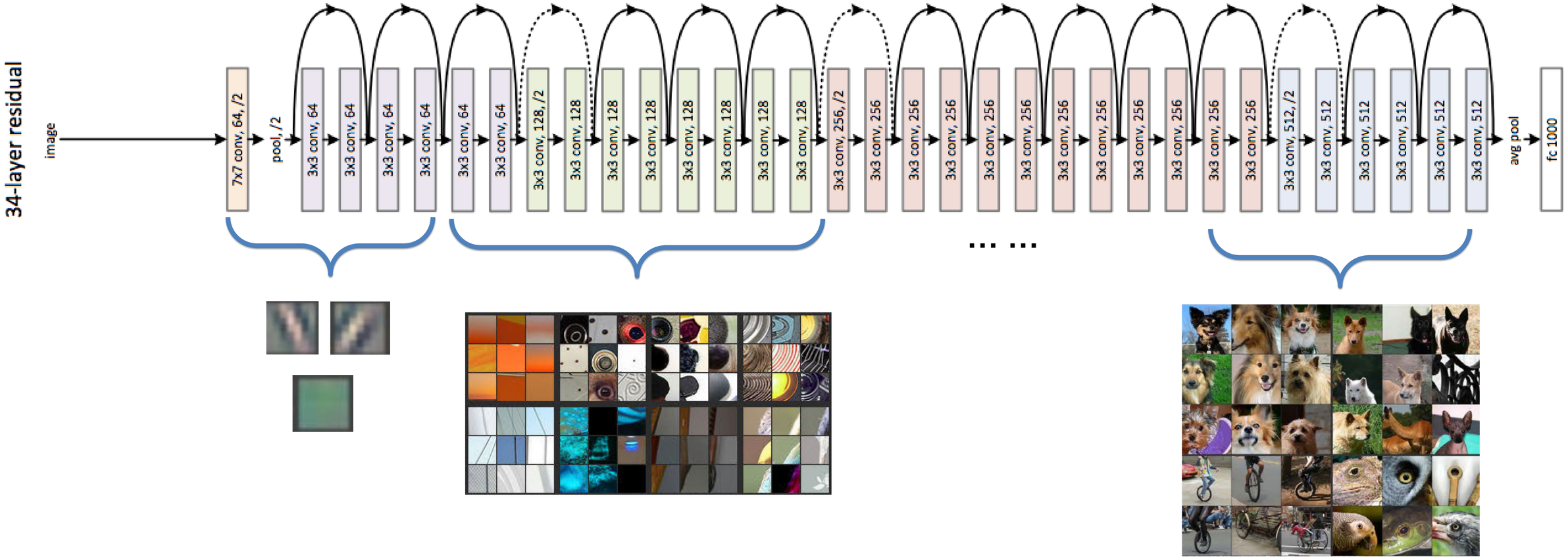
ResNet 34 arc chitecture

3.6 billion FLC twork

Takes ~2week
on single GPU
(ImageNet-1K

Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385
An Analysis of Deep Neural Network Models for Practical Applications. https://arxiv.org/abs/1605.07678

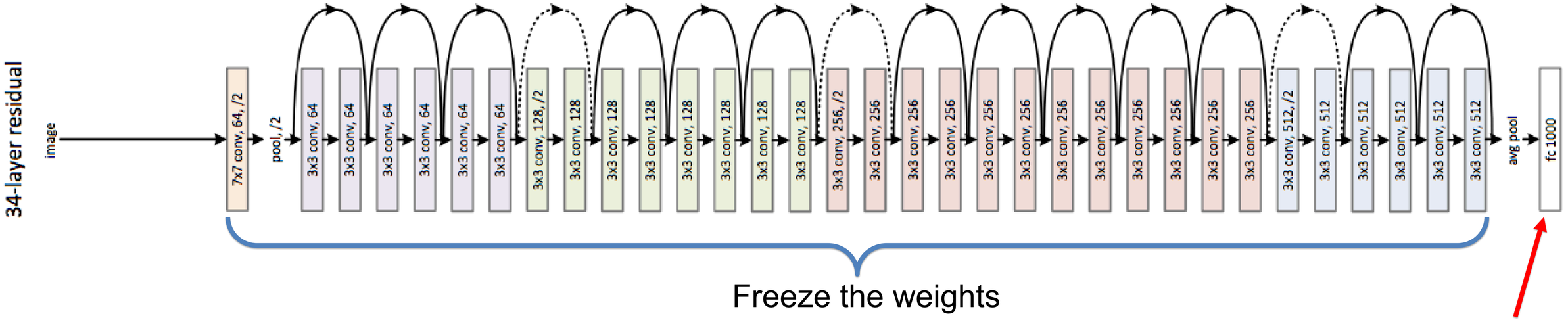
Transfer learning as feature extractor



Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

Visualizing and Understanding Convolutional Networks. https://arxiv.org/abs/1311.2901

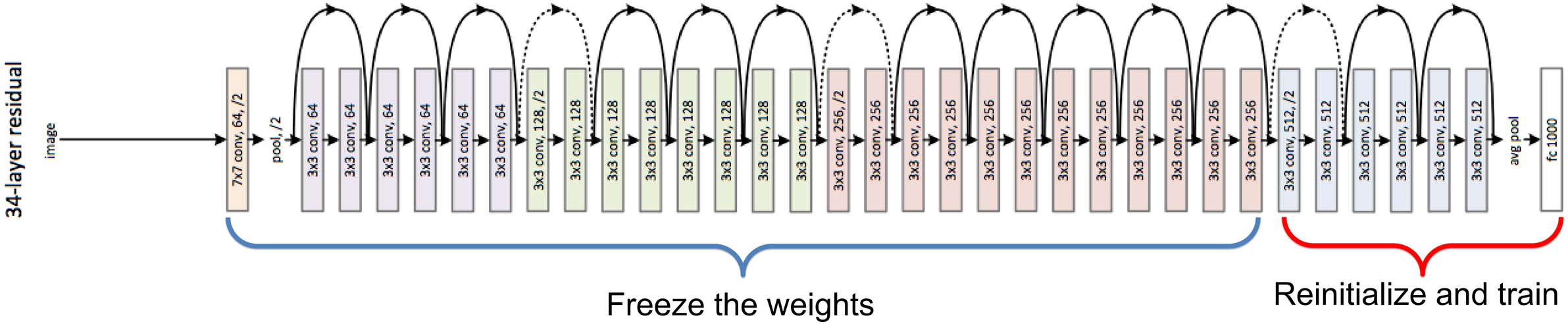
Transfer learning as feature extractor



- Data for new task is not abundant, e.g. 5K images
- Data is similar to common dataset, e.g. a set of different dog breeds
- You are happy with classical architecture – often favoring classification problems

Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385
https://miro.medium.com/max/1200/1*DWUdYY4ZX6AAAIgyK3LCsQ.jpeg

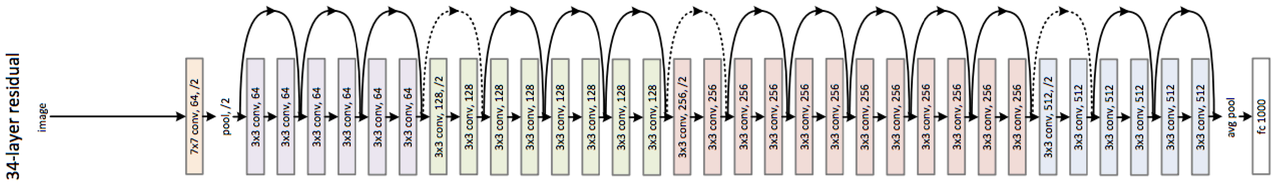
Transfer learning variations



- There are more data for the new task, e.g. 10K images
- Task specific data is like common dataset
- Happy with existing architecture – often for classification problems
- Need to treat the number of re-trained layers as a hyperparameter

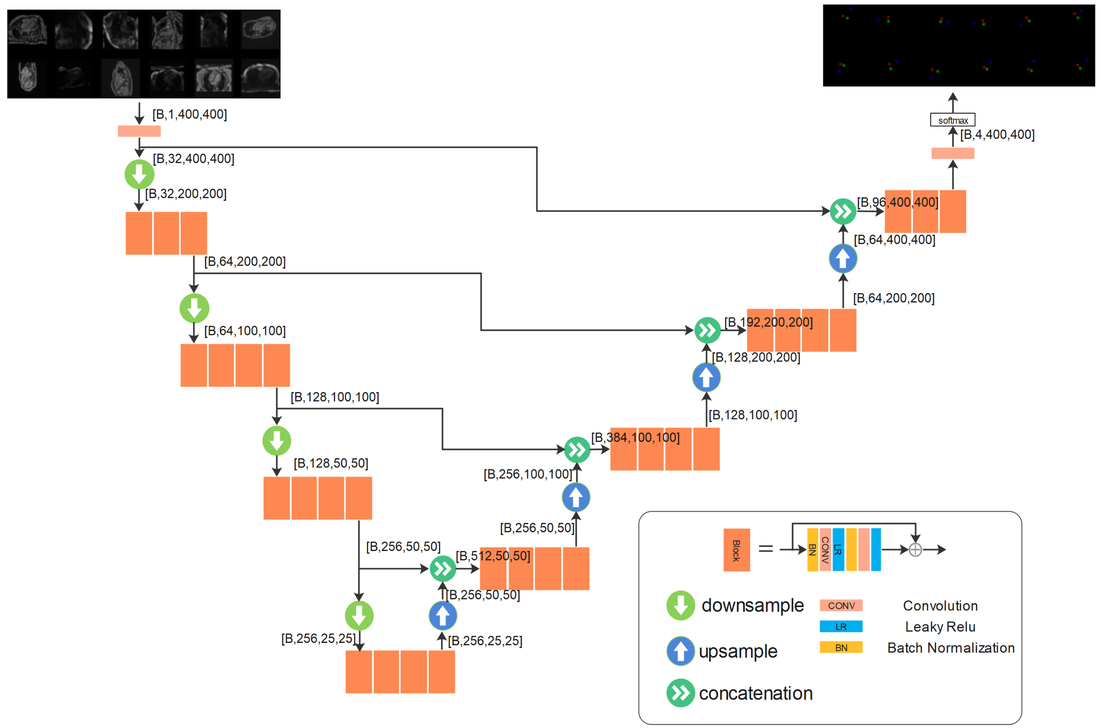
Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

Fine-tuning



Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

- Earlier model has local skip connections
- Still maintain a feed-forward architecture
- Focus more on classification

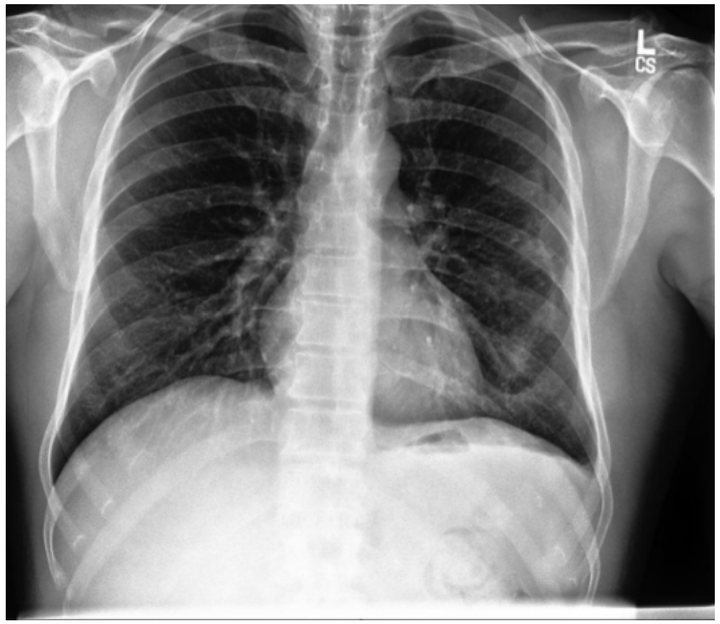


https://arxiv.org/abs/2008.06142

Fine tuning

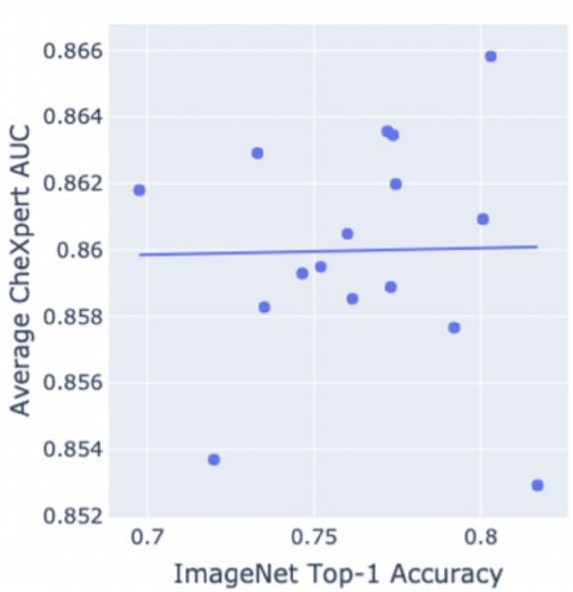
- Optimize all model weights
- Reduce learning rate
- Train less epochs/early stopping if task specific dataset is small
- Test set only includes the task specific data

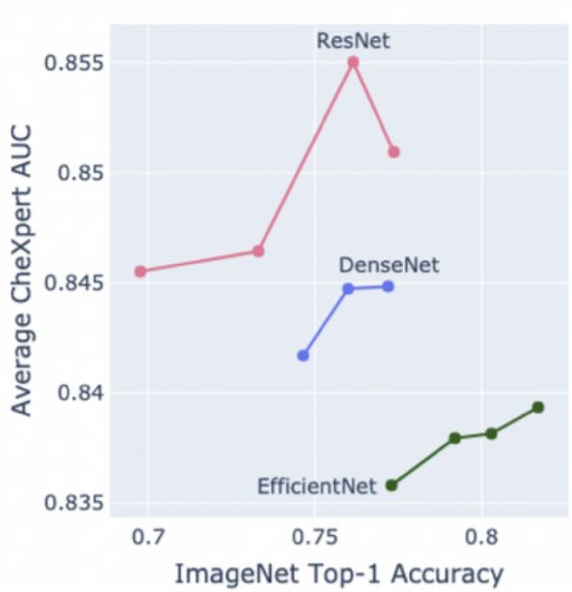
Difference in data distribution

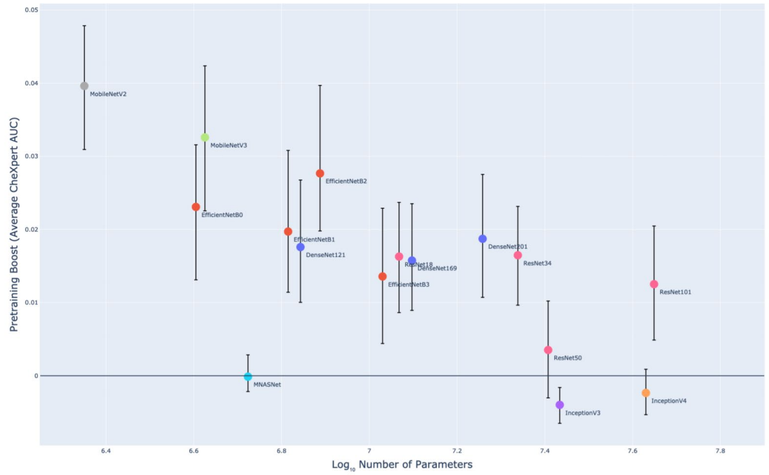


CheXpert dataset consisting of 224,316 chest X-rays of 65,240 patients - https://stanfordmlgroup.github.io/competitions/chexpert/

Pre-trained model may not always give large gain







- Use ImageNet-1K dataset to pretrain 16 different CNN models
- Transfer learning per-trained model on Chest X-ray dataset
- No correlation between chest X-ray performance with Image-Net performance …
- More impact on performance is from model architecture family (e.g. ResNet vs. DenseNet), than model size within one family (e.g. ResNet 34 vs. ResNet 50)
- Pre-training gave more performance boost for smaller model
- Fine-tuning was used

CheXpert dataset consisting of 224,316 chest X-rays of 65,240 patients - https://stanfordmlgroup.github.io/competitions/chexpert/

CheXtransfer: Performance and Parameter Efficiency of ImageNet Models for Chest X-Ray Interpretation. 2021. https://arxiv.org/abs/2101.06871

NIH National Heart, Lung, and Blood Institute

Transfer learning

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

- Transfer learning is to solve a new task $\boldsymbol{T}'$, given the pre-trained task $\boldsymbol{T}$
- When training new task, we do not have access to data $\boldsymbol{D}$ for old task
- f_θ and f'_θ share the same model architecture

$$\theta' = \theta' - \alpha \nabla_\theta \ell'(\boldsymbol{D}', \theta)$$

θ' is initialized as θ for shared layers in feature extraction TL

θ' is initialized as θ for all layers in fine-tuning TL

		$\boldsymbol{D}^{new}$	
Task similarity		Large	Small
	High	Share more layers or fine-tuning	Feature extraction TL
	Low	Fine-tuning	Need bigger gun …

Multi-task learning

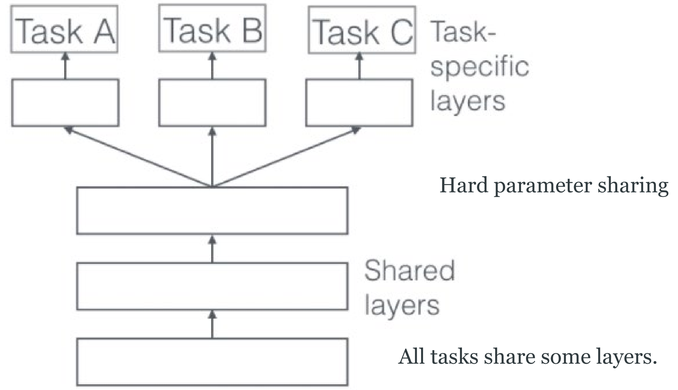
There are a set of tasks : $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$

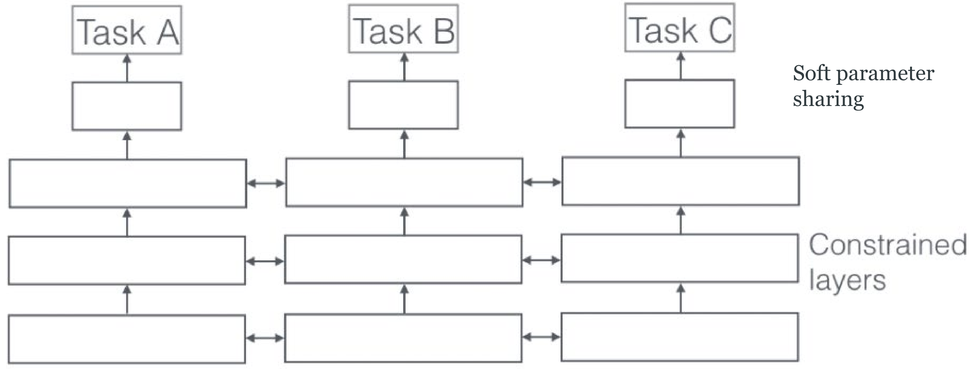
- Target is to train all tasks with some shared configuration
- When training new task, we have access to all N datasets
- f_θ and f'_θ share all or some model parameters

$$\min_{\{\theta^{shared}, \theta^{\mathbf{0}}, \ldots, \theta^{\boldsymbol{N-1}}\}} \sum_{i=0}^{N-1} \ell^i(\boldsymbol{D}^i, \theta^{shared}, \theta^{\boldsymbol{i}}, z^i)$$

z^i is task specifier, e.g. one-hot vector for tasks

- Reduce risk of overfitting
- Learn more general representation from multiple tasks
- Need to balance sampling for tasks for training
- Need to balance loss across tasks for training
- Transfer learning is a sequential way to solve multi-task learning





https://ruder.io/multi-task/

Multi-task learning

There are a set of tasks : $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$

While performance for some tasks does not reach the goal:

sample a batch of tasks $B \sim \boldsymbol{T}_i, \mathrm{i} = 0, \ldots, \mathrm{N}-1$

Need a task sampling strategy

for every task $\boldsymbol{T}_k$ in B, sample a mini-batch of data from $\boldsymbol{D_k}$

Compute loss for tasks in B:

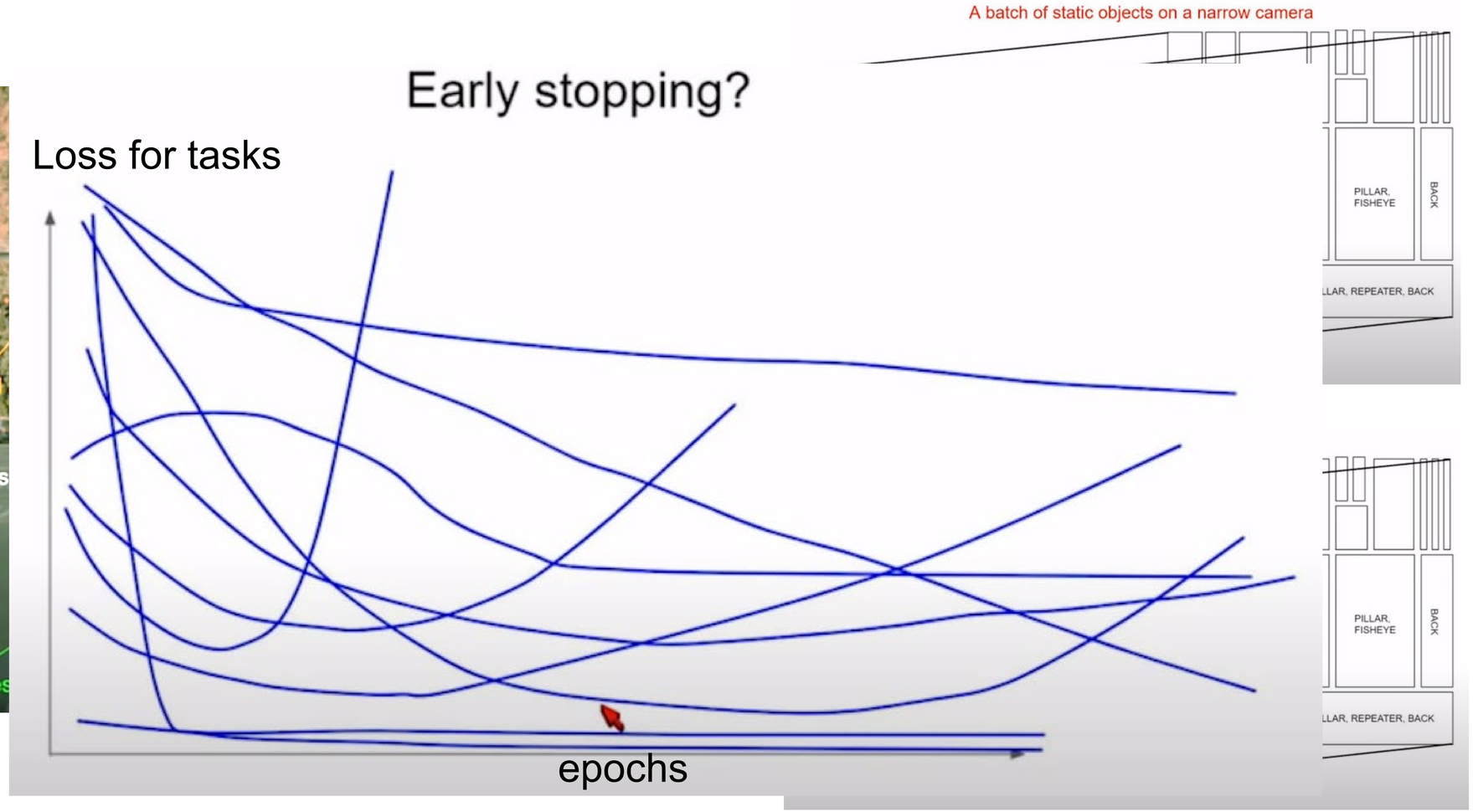
$$\ell\left(B, \theta_B^{shared}, \theta_B^0, \ldots, , \theta_B^{\#B-1}\right) = \frac{1}{\#B} \sum_{k=0}^{\#B-1} \ell^k(\boldsymbol{D}^k, \theta^{shared}, \theta^{\boldsymbol{k}}, z^k)$$

Need to balance loss across tasks

Backprop to related network layers for tasks in B

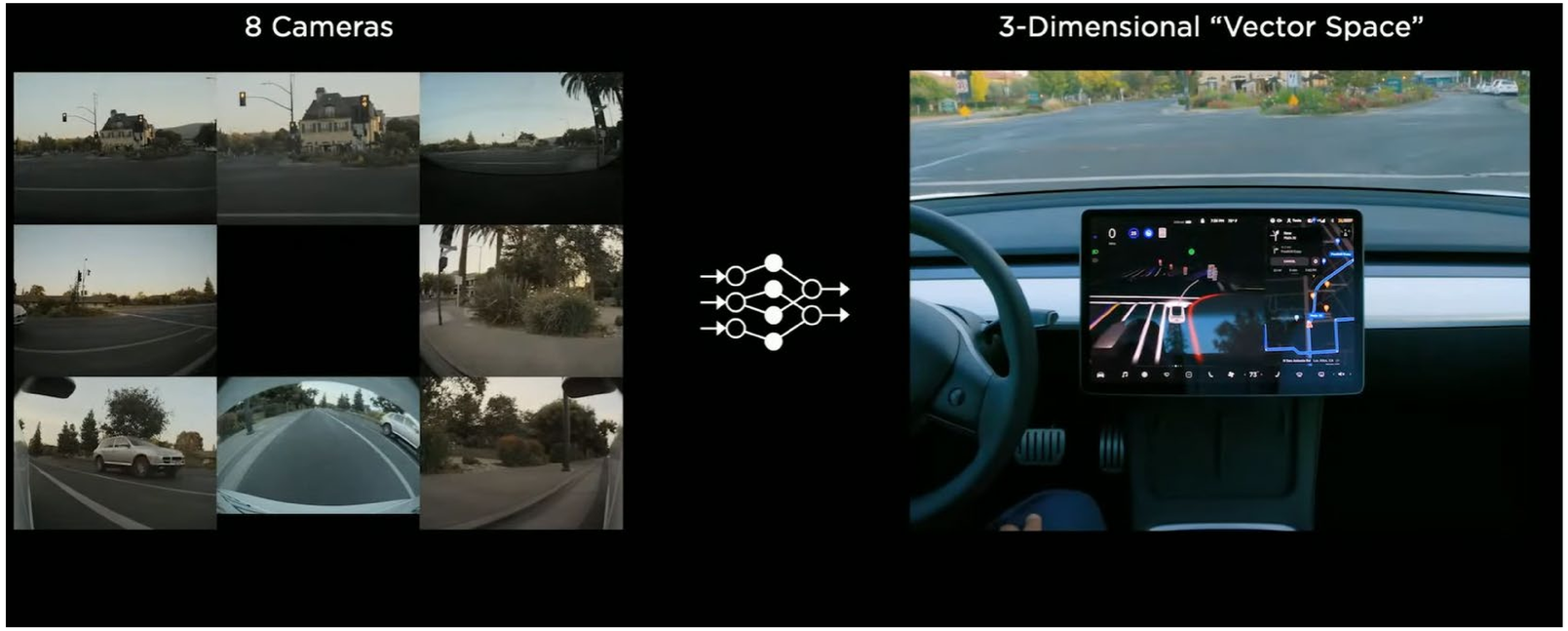
Update relevant parameters using optimizer

Multi-task learning in Tesla



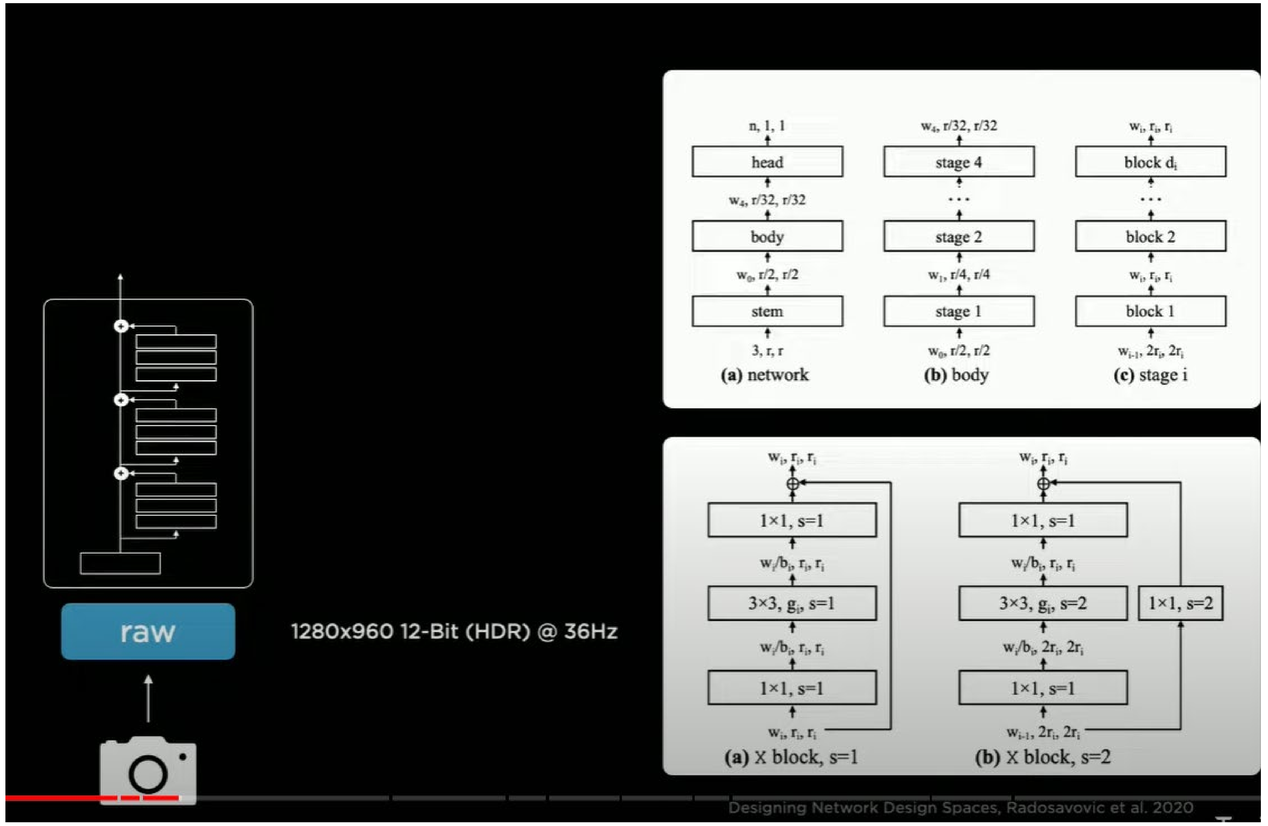
Andrej Karpathy: Tesla Autopilot and Multi-Task Learning for Perception and Prediction, 2020

NN + 3D multi-camera vision



Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 1: CNN encoder

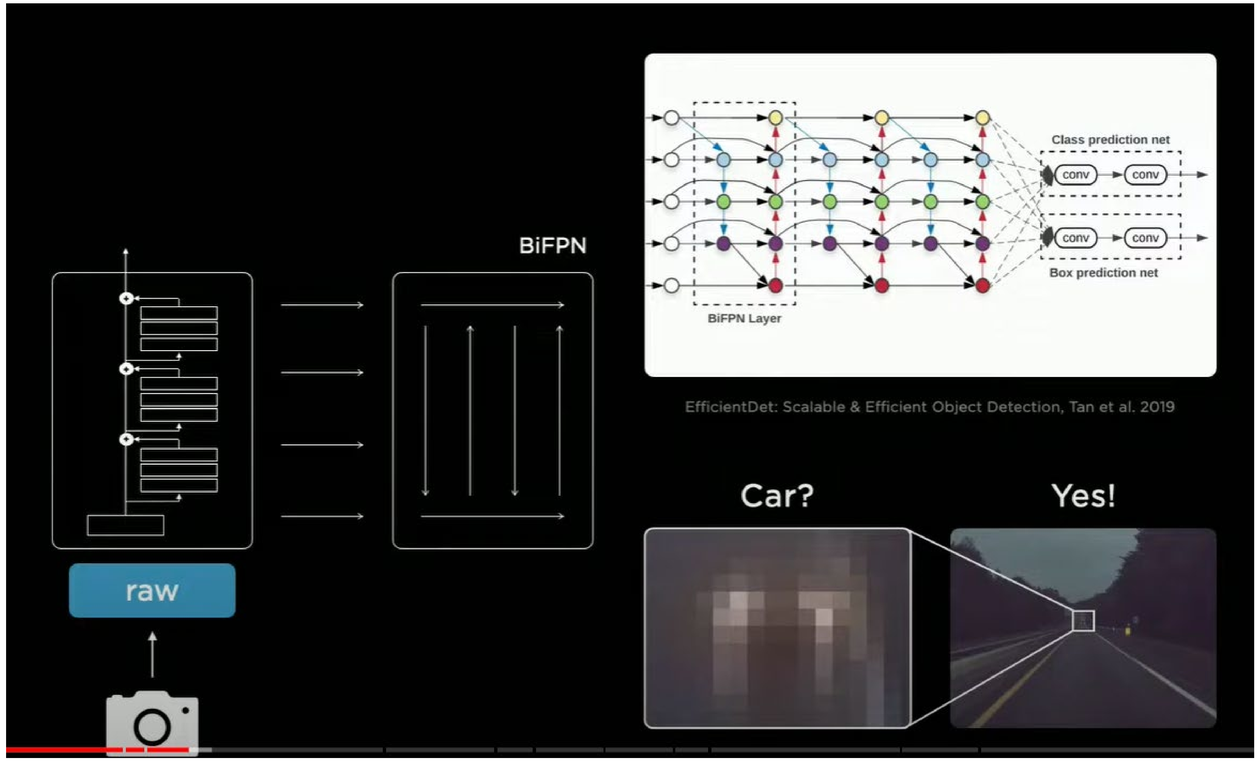


each conv. The block has 3 parameters: the width w_i, bottleneck ratio b_i, and group width g_i. (b) The stride-two ($s = 2$) version.

https://arxiv.org/pdf/2003.13678.pdf

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 2: Feature fusion



BiFPN : **Weighted Bi-directional Feature Pyramid Network**, is a type of feature pyramid network which allows easy and fast multi-scale feature fusion.

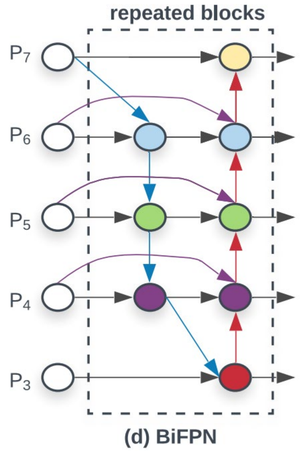


Image size 640x480

P3: 80x60
P4: 40x30
P5: 20x15
P6: 10x7
P7: 5x3

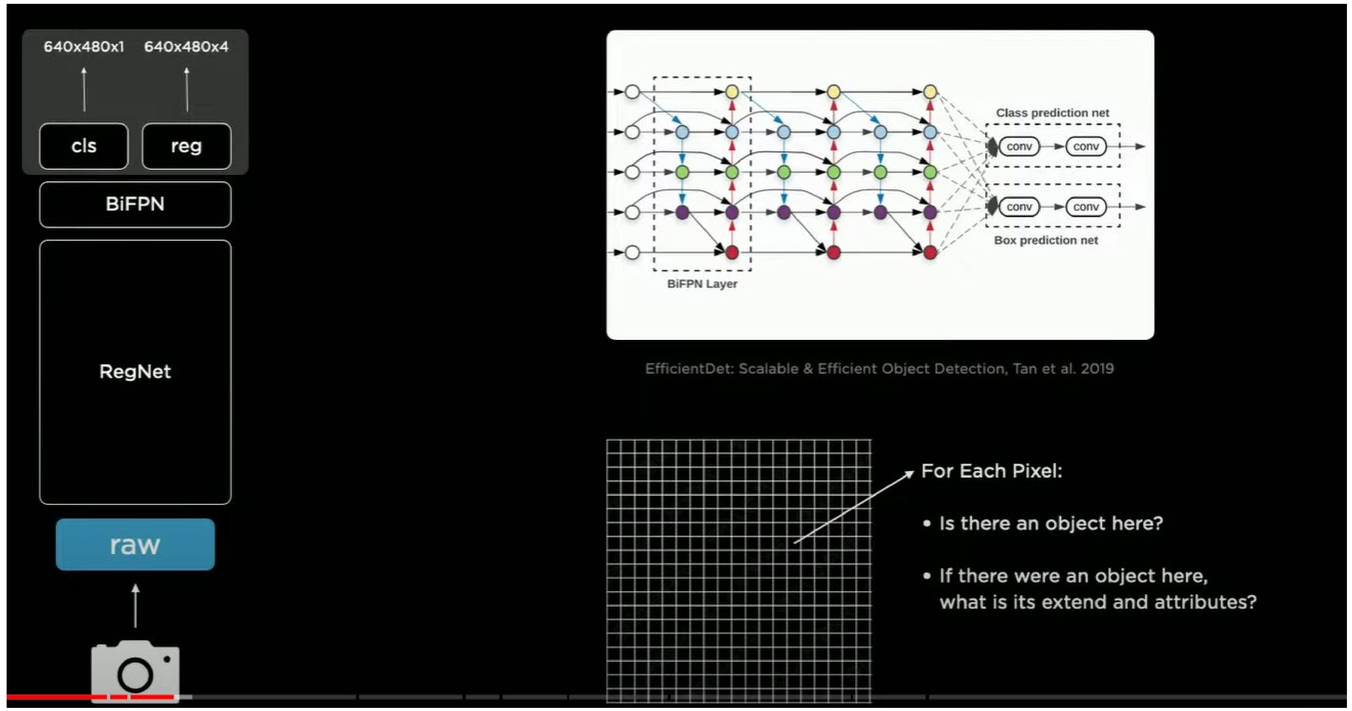
Fusion from up and down, plus residual connection

$$P_6^{out} = conv(downsample(P_5^{out}), P_6^{in}, conv(P_7^{in}, P_6^{in}))$$

EfficientDet: Scalable and Efficient Object Detection. https://arxiv.org/abs/1911.09070v7

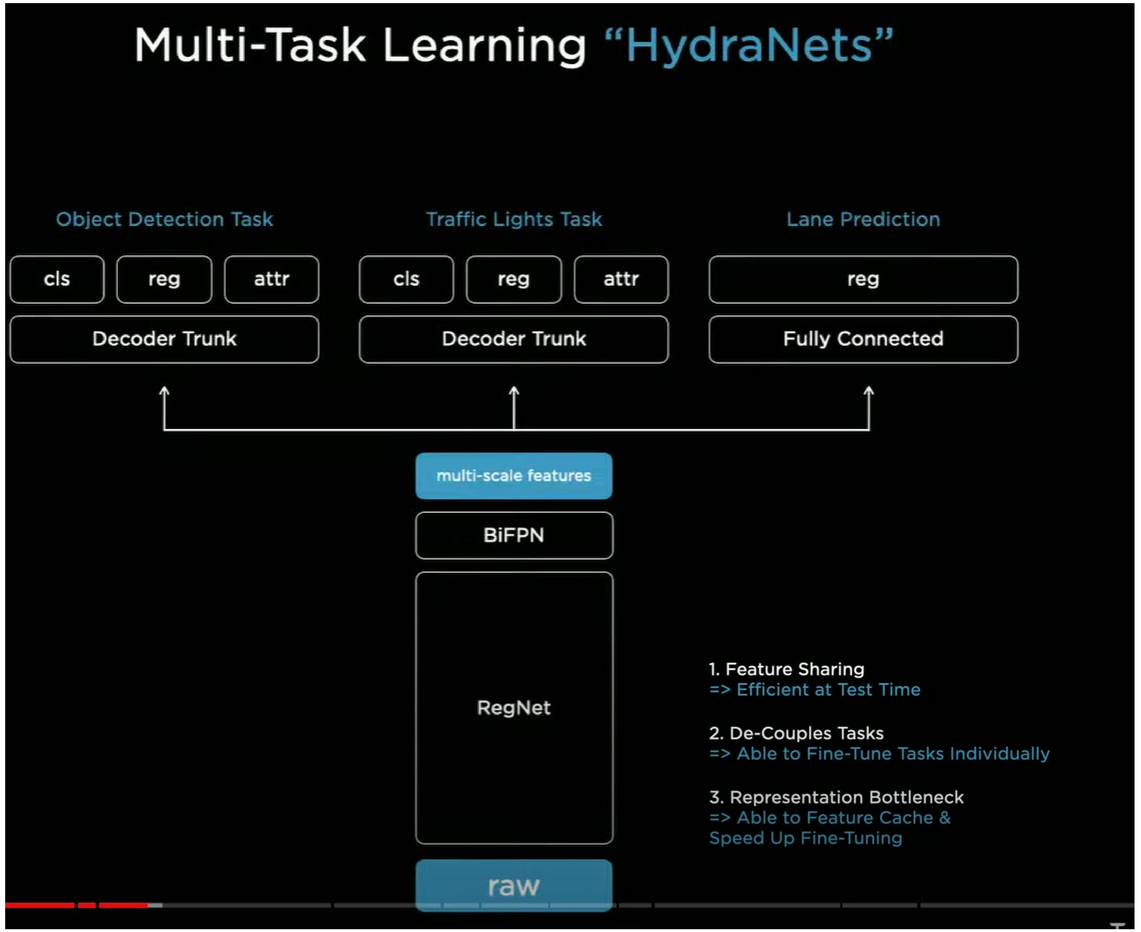
Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 3: One-stage object detector



Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 4: HydraNet

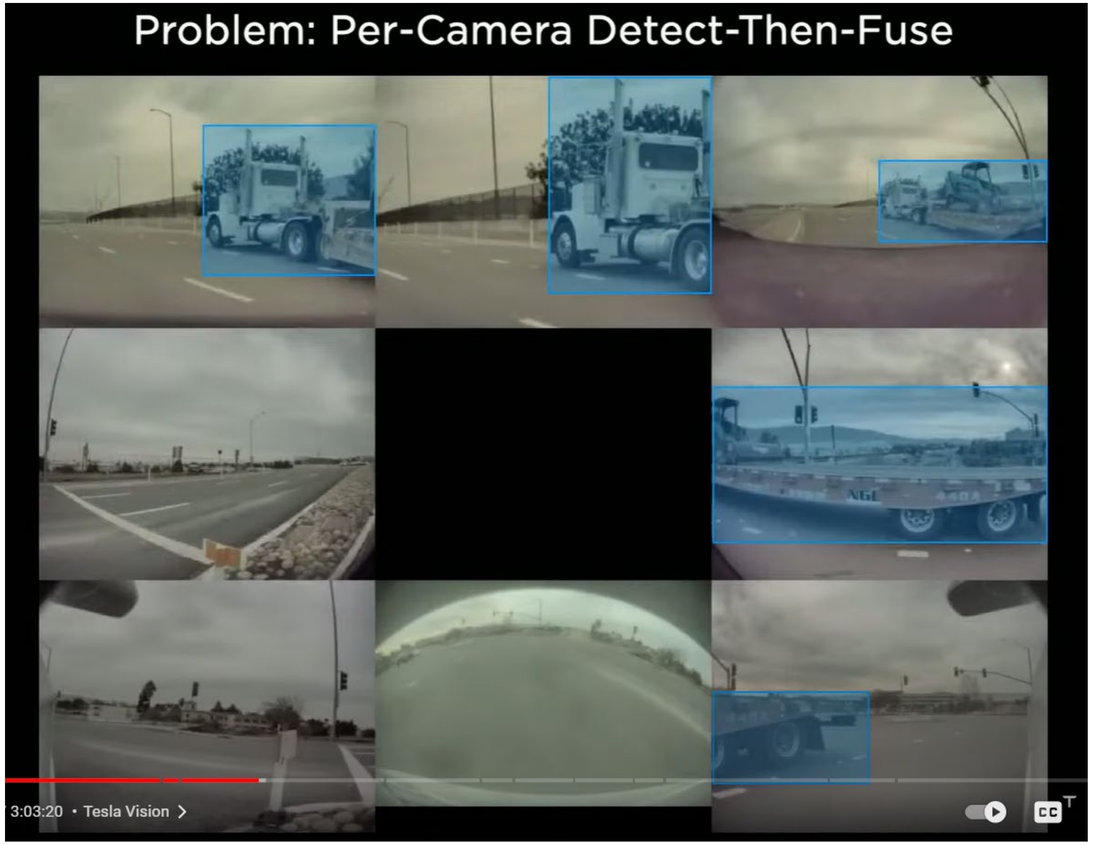


Single 2D picture prediction

in 2016

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

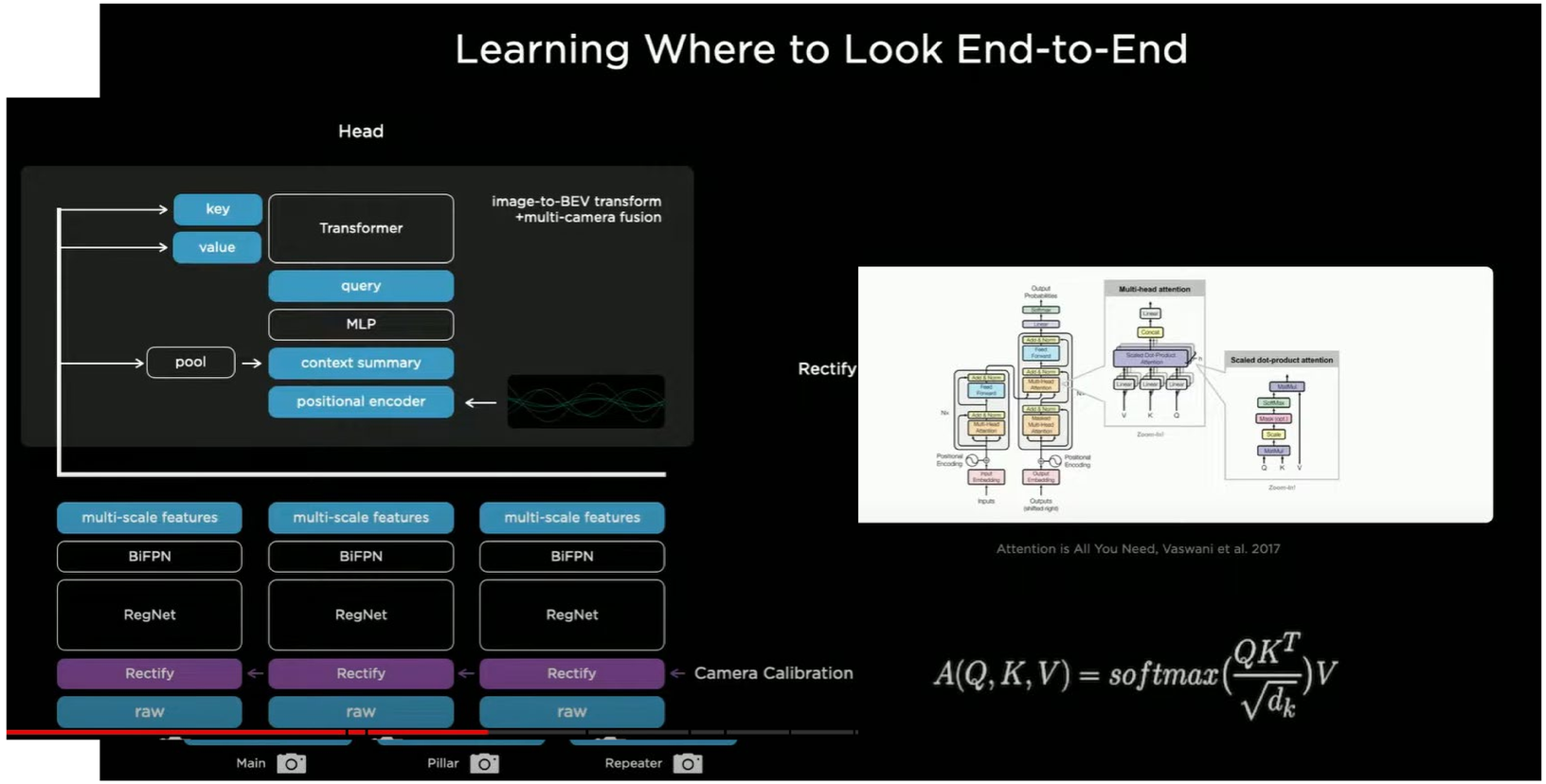
Step 5: Vector space by fusing multiple cameras



Solution: end-to-end NN
Input: 8 cameras
Output: 3D vector space with objects and depths

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 5: Vector space by fusing multiple cameras

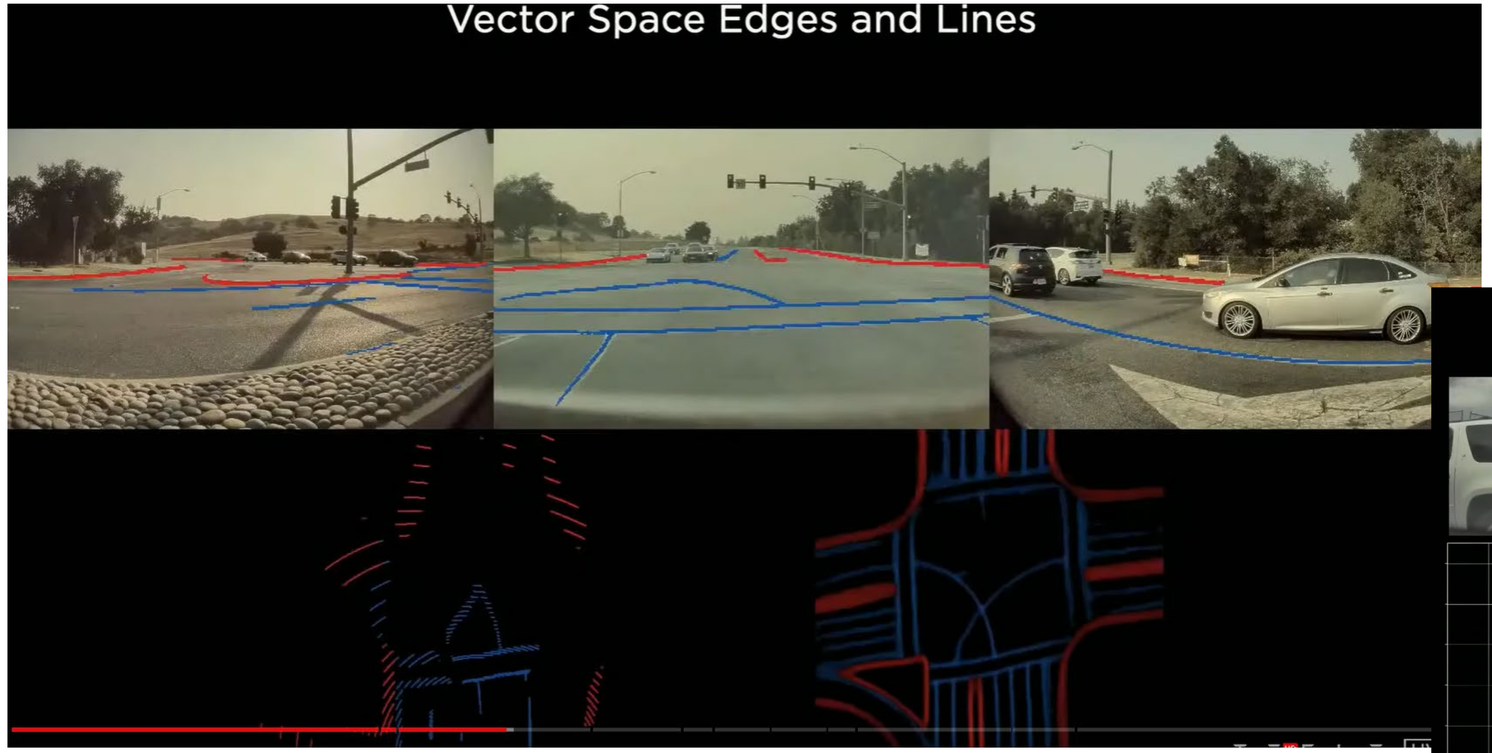


Tesla uses transformer to fuse 8 cameras

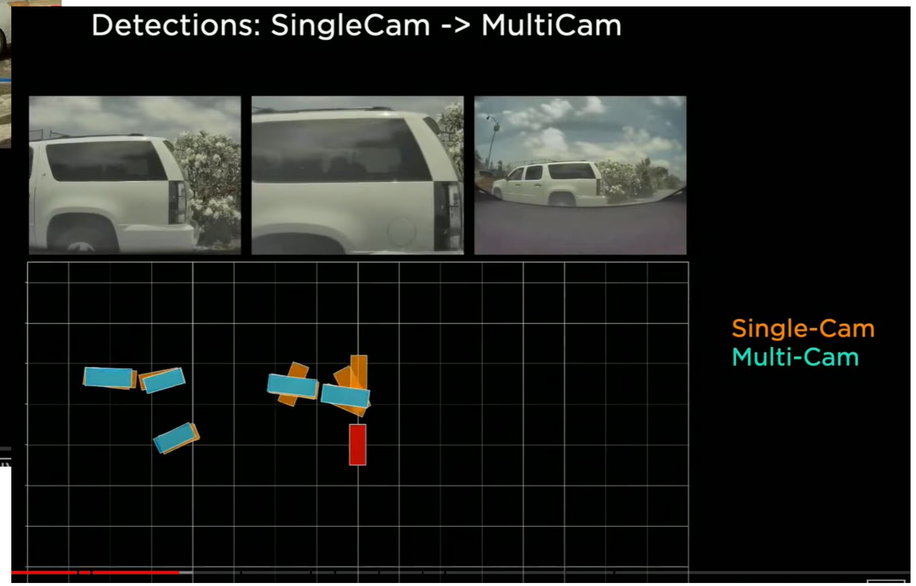
Learn where to look!

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Everything is now in 3D+T space

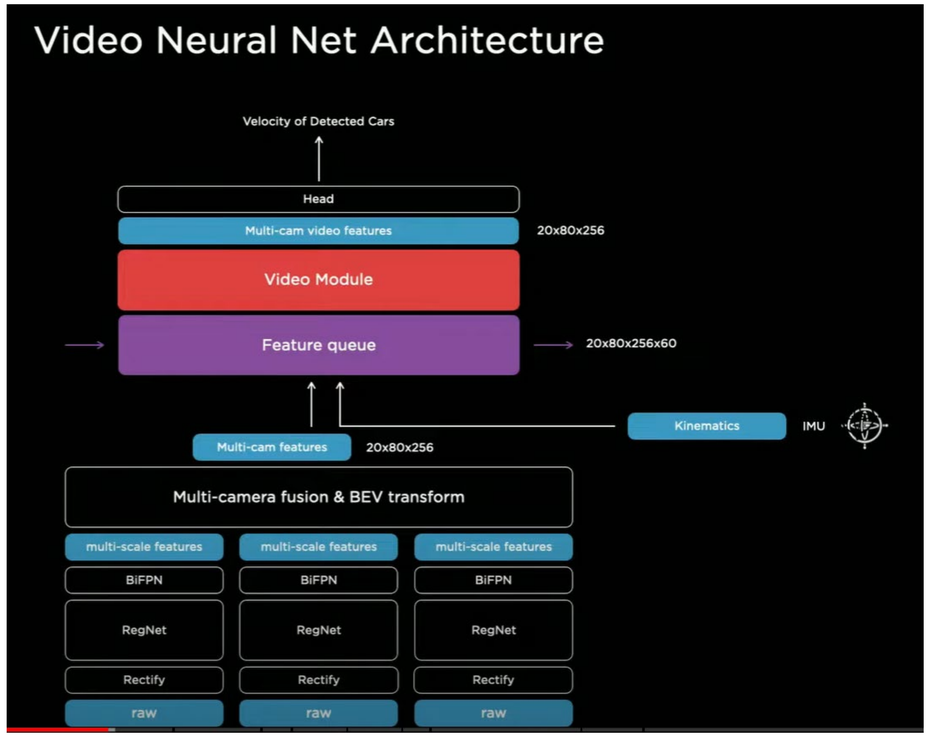


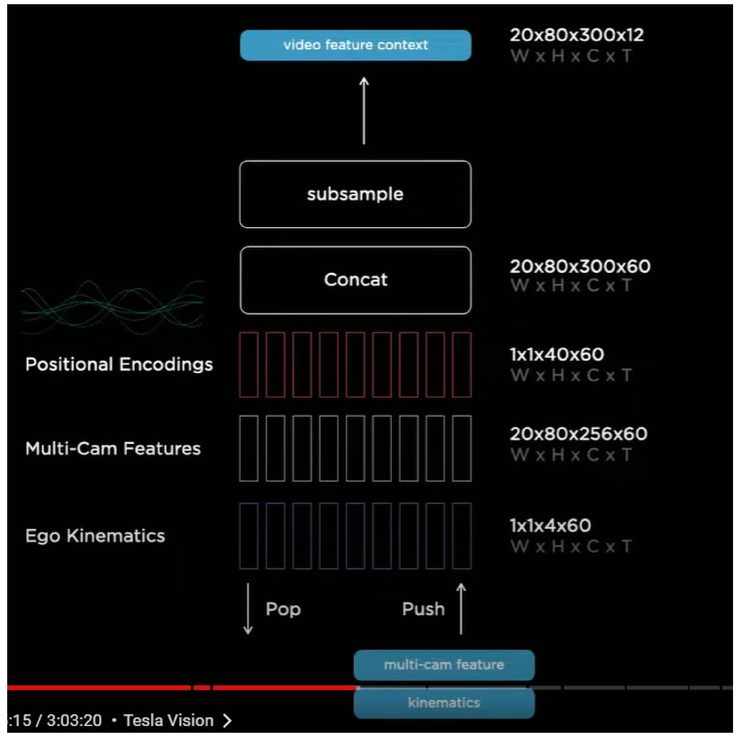
Improve object detection and depth estimation in 3D space

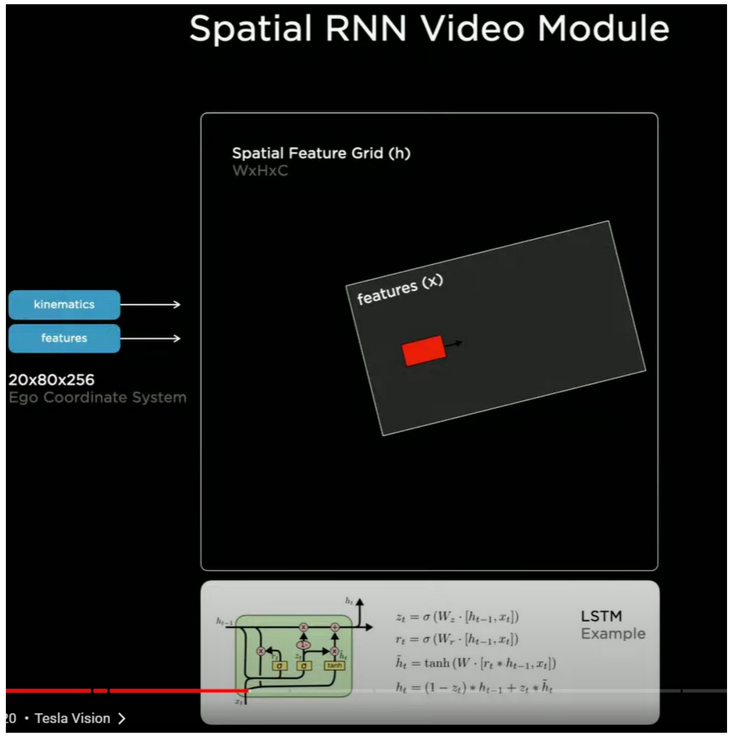


Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 6: 3D+T



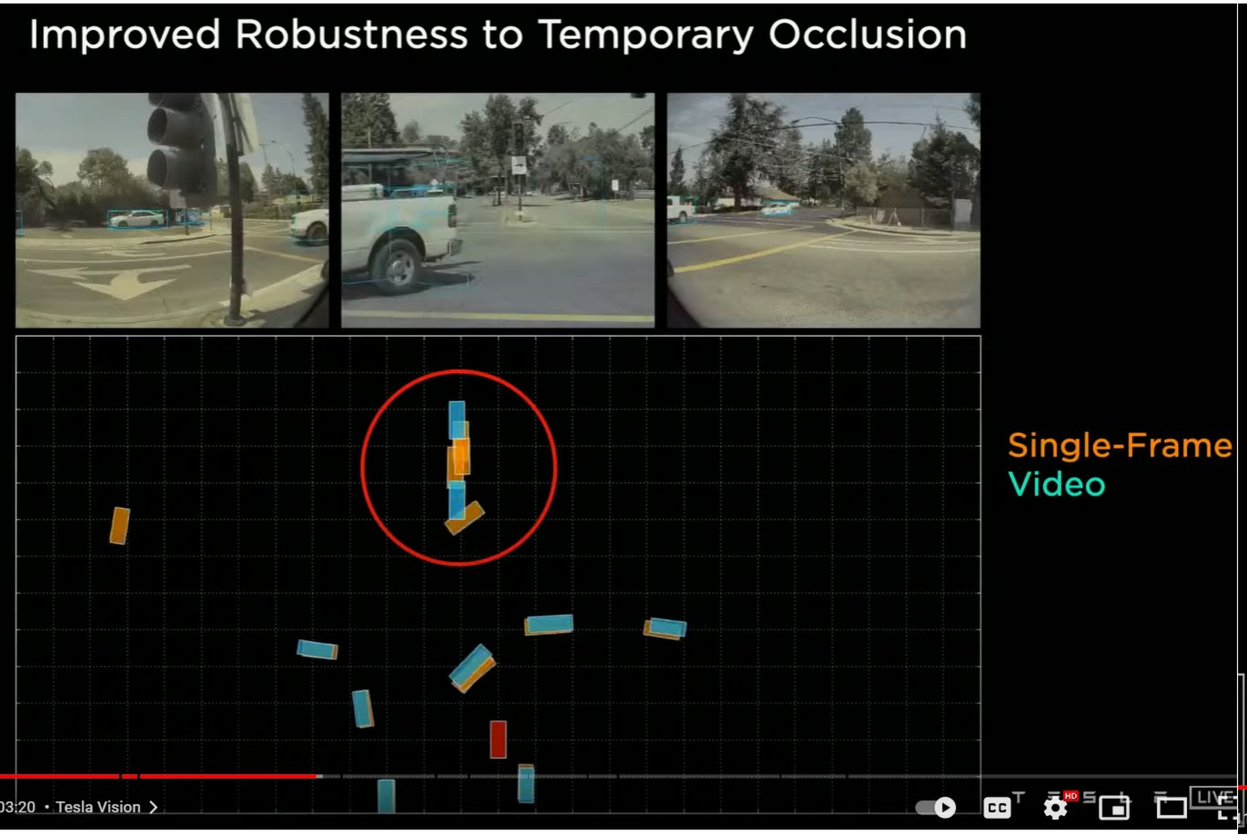


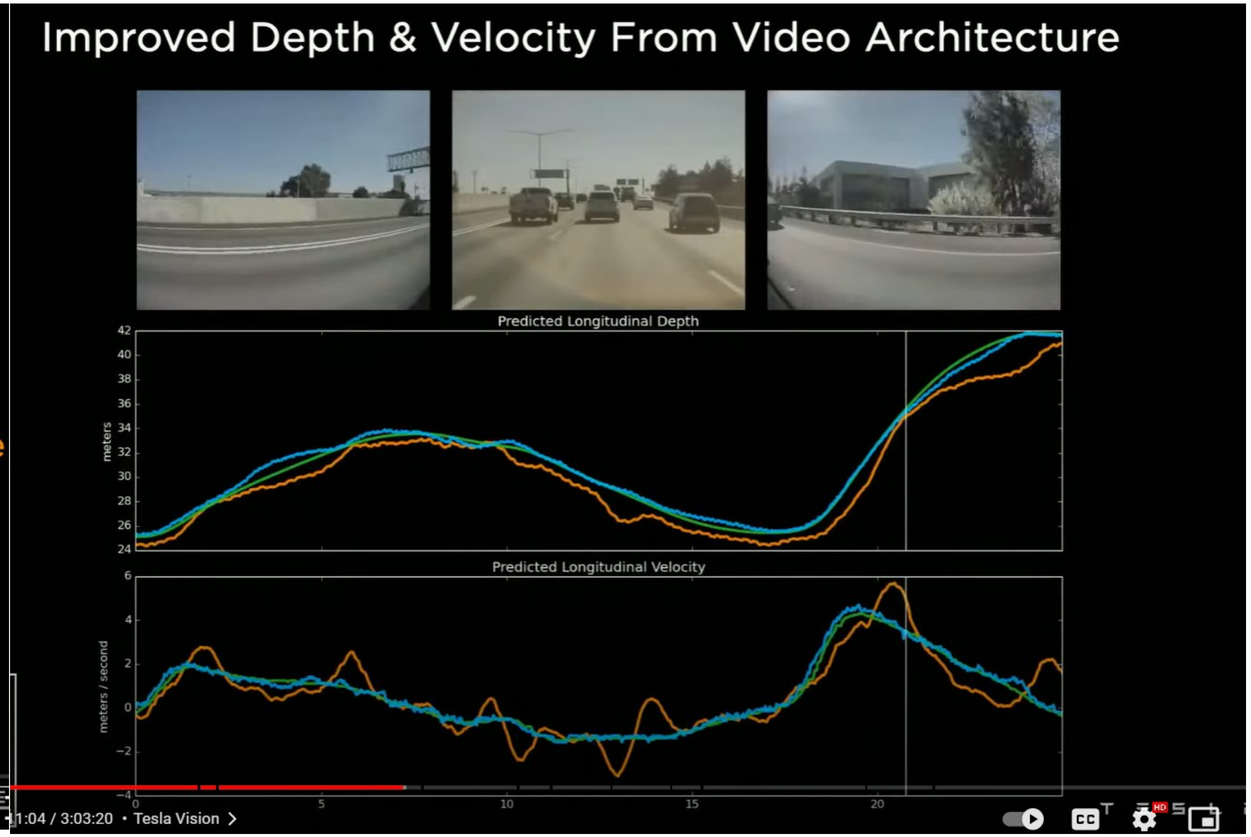


Give car “memory”, improve object detection, improve depth/velocity estimation etc.

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Step 6: 3D+T

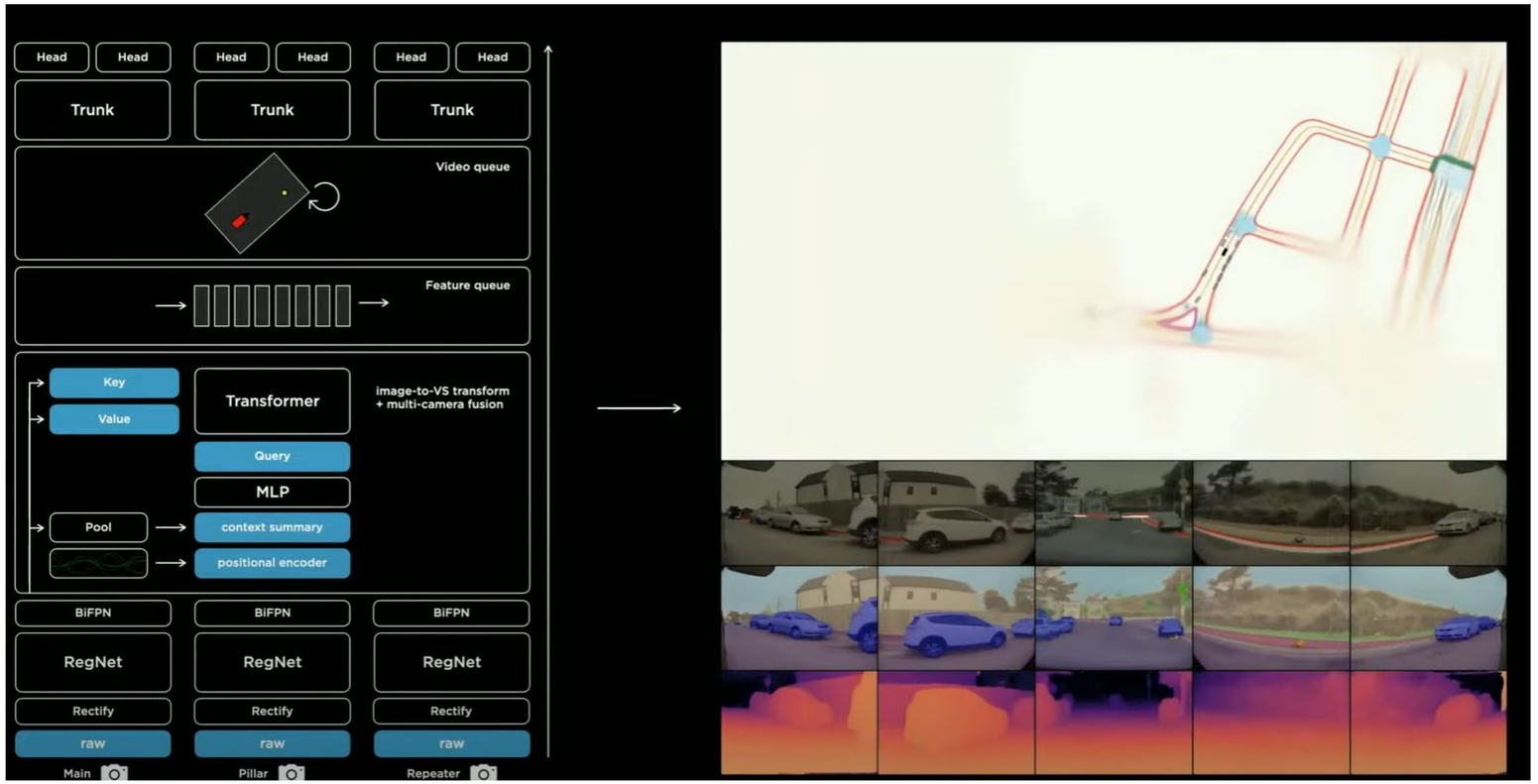




Give car “memory”, improve object detection, improve depth/velocity estimation etc.

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Vision based perception for the autonomous driving



- Single camera embedding (2D)
- Transformer for camera fusion and vector space construction (3D)
- Spatial RNN for temporal memory (3D+T)
- One backbone + multiple heads
- End-to-end training for many networks
- In 2021!

Tesla AI day. https://www.youtube.com/watch?v=j0z4FweCy4M

Outline

- Transfer learning
- **Meta Learning**
- Contrastive learning

Motivation for meta learning

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

	$\boldsymbol{D}^{new}$		
Task similarity		Large	Small
	High	Share more layers or fine-tuning	Feature extraction TL
	Low	Fine-tuning	?

Transfer learning: initialize parameters from pre-trained model, continue training on new dataset

$$\theta = \theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^{new}, \theta)$$

- No guarantee for performance on new dataset/new task
- Works better if $\boldsymbol{D}^{new}$ is large; problematic if $\boldsymbol{D}^{new}$ is small
- Harder to extend to multi-task setting

Model Agnostic Meta Learning (MAML)

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

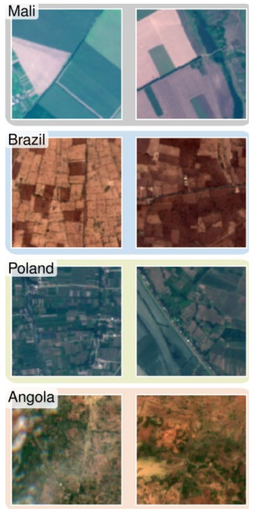
Given a set of tasks $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$, can we solve a new task $\boldsymbol{T}^{new} = \{\boldsymbol{D}^{new}, \ell, f_\theta\}$

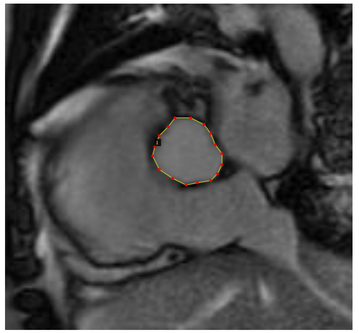
$\boldsymbol{D}^{new}$ can be a small new dataset, e.g.

landscape pictures for a new region

pictures of a new dog breed

Images of a new imaging modality or anatomy





Aorta vessel

What we want:

Given a model and a set of related task, **quickly** learn on new dataset → **quickly** gain new ability

Meta-Learning for Few-Shot Land Cover Classification. 2020. https://arxiv.org/abs/2004.13390

Model Agnostic Meta Learning (MAML)

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

Given a set of tasks $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$, we solve a new task $\boldsymbol{T}^{new} = \{\boldsymbol{D}^{new}, \ell, f_\theta\}$ by utilizing the existing tasks

Compute loss of updated model on new task

$$\min_{\theta} \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell\left(\theta - \alpha \nabla_\theta \ell\left(\boldsymbol{D}^i, \theta\right), \boldsymbol{D}^{new}\right)$$

Optimize model parameter on task i

Search the optimal model parameter as the one which meets two requirements at the same time:

- make progress on the set of tasks
- minimize loss on new task

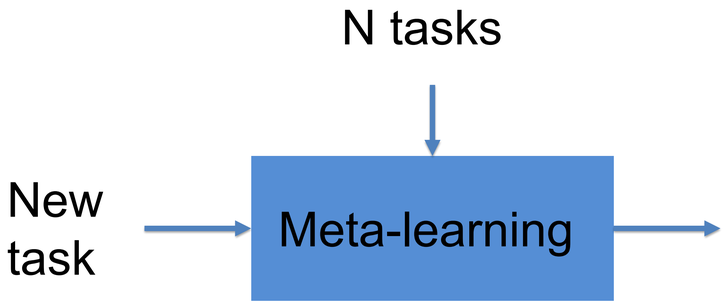
Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

MAML is a type of meta-learning

Meta learning: Learn to learn how to handle new task

An algorithm to find a method F to produce model f for a new task

Input: N old tasks, some data for new task
Output : a good model for the new task



MAML is one of the meta-learning algorithm

Deep learning: Learn a good model for a task

An algorithm to find a model f for a task

Input: training data for this task
Output : a good model for this task

Can learn other components: learning rate, architecture, data augmentation …

Model Agnostic Meta Learning (MAML)

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

Given a set of tasks $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \dots, N-1$, we solve a new task $\boldsymbol{T}^{new} = \{\boldsymbol{D}^{new}, \ell, f_\theta\}$ by utilizing the existing tasks

MAML

$$\min_\theta \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell(\theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^i, \theta), \boldsymbol{D}^{new})$$

- Over a set of related tasks, learn parameters which transfer best to the new task
- Inner loop parameter update step

Fine-tuning

$$\theta = \theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^{new}, \theta)$$

- Harder to utilize multiple tasks
- Start from a fixing point and optimize for where it can reach; suboptimal

Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

Model Agnostic Meta Learning (MAML)

Define the task as : data set $\boldsymbol{D}$, loss function ℓ, model with parameter f_θ, task $\boldsymbol{T} = \{\boldsymbol{D}, \ell, f_\theta\}$

Given a set of tasks $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$, we solve a new task $\boldsymbol{T}^{new} = \{\boldsymbol{D}^{new}, \ell, f_\theta\}$ by utilizing the existing tasks

MAML

$$\min_\theta \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell(\theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^i, \theta), \boldsymbol{D}^{new})$$

- Find model parameter quickly adapted to a new task
- Also suitable for the few-short learning: give a set of tasks, every task has only K samples

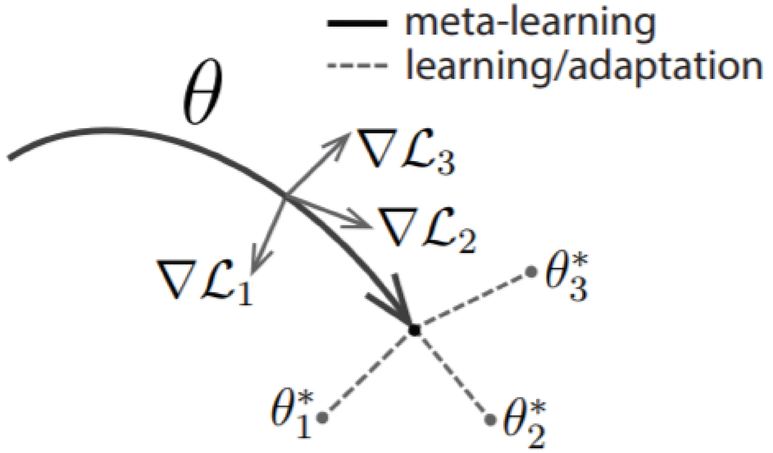


Figure 1. Diagram of our model-agnostic meta-learning algorithm (MAML), which optimizes for a representation θ that can quickly adapt to new tasks.

Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

Model Agnostic Meta Learning (MAML)

MAML algorithm

Given a set of tasks $\boldsymbol{T}_i = \{\boldsymbol{D}, \ell, f_\theta\}_i, i = 0, \ldots, N-1$, and a new task $\boldsymbol{T}^{new} = \{\boldsymbol{D}^{new}, \ell, f_\theta\}$
Given learning rate α, β
Initialize model parameter θ

While not converge :

Sample a mini-batch from $\boldsymbol{D}^{new}$

grad = 0
for all task i:
Sample a mini-batch from $\boldsymbol{D}^i$
Compute updated parameter:
$$\theta' = \theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^i, \theta)$$
Compute gradient $\boldsymbol{\nabla_\theta \ell(\theta', D^{new})}$
grad += $\boldsymbol{\nabla_\theta \ell(\theta', D^{new})}$

Update $\theta = \theta - \beta \cdot \text{grad}$

$$\min_\theta \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell(\theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^i, \theta), \boldsymbol{D}^{new})$$

- Inner loop and outer loop optimization
- Other optimizers are applicable
- Multi-step inner loop is possible
- This algorithm is for “old-new task” setup

Finn, Abbeel, Levine. Model-Agnostic Meta-Learning. ICML 2017

Model Agnostic Meta Learning (MAML)

$$\min_{\theta} \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell(\theta - \alpha \nabla_{\theta} \ell(\boldsymbol{D}^i, \theta), \boldsymbol{D}^{new})$$

$$\theta' = \theta - \alpha \nabla_{\theta} \ell(\boldsymbol{D}^i, \theta)$$

Let θ be a Dx1 vector for all parameters

$$\boldsymbol{\nabla}_{\theta} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}}) = \frac{\boldsymbol{d\theta'}}{\boldsymbol{d}\theta} \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta'}} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}})$$

DxD matrix

Dx1 vector

$$= (\boldsymbol{I} - \alpha \boldsymbol{\mathcal{H}}_{\theta}(\boldsymbol{D}^i, \theta)) \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta'}} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}})$$

$$= \boldsymbol{\nabla}_{\boldsymbol{\theta'}} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}}) - \alpha \boldsymbol{\mathcal{H}}_{\theta}(\boldsymbol{D}^i, \theta) \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta'}} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}})$$

No need to compute and store full Hessian matrix, but Hessian-vector product

Model Agnostic Meta Learning (MAML)

$$\min_{\theta} \sum_{\boldsymbol{task\ i=0}}^{\boldsymbol{N-1}} \ell(\theta - \alpha \nabla_{\theta} \ell(\boldsymbol{D}^{i}, \theta), \boldsymbol{D}^{new})$$

Let θ be a Dx1 vector for all parameters

If we do two inner loop update steps:

$$\phi = \theta - \alpha \nabla_{\theta} \ell(\boldsymbol{D}^{i}, \theta)$$

$$\theta' = \phi - \alpha \nabla_{\phi} \ell(\boldsymbol{D}^{i}, \phi)$$

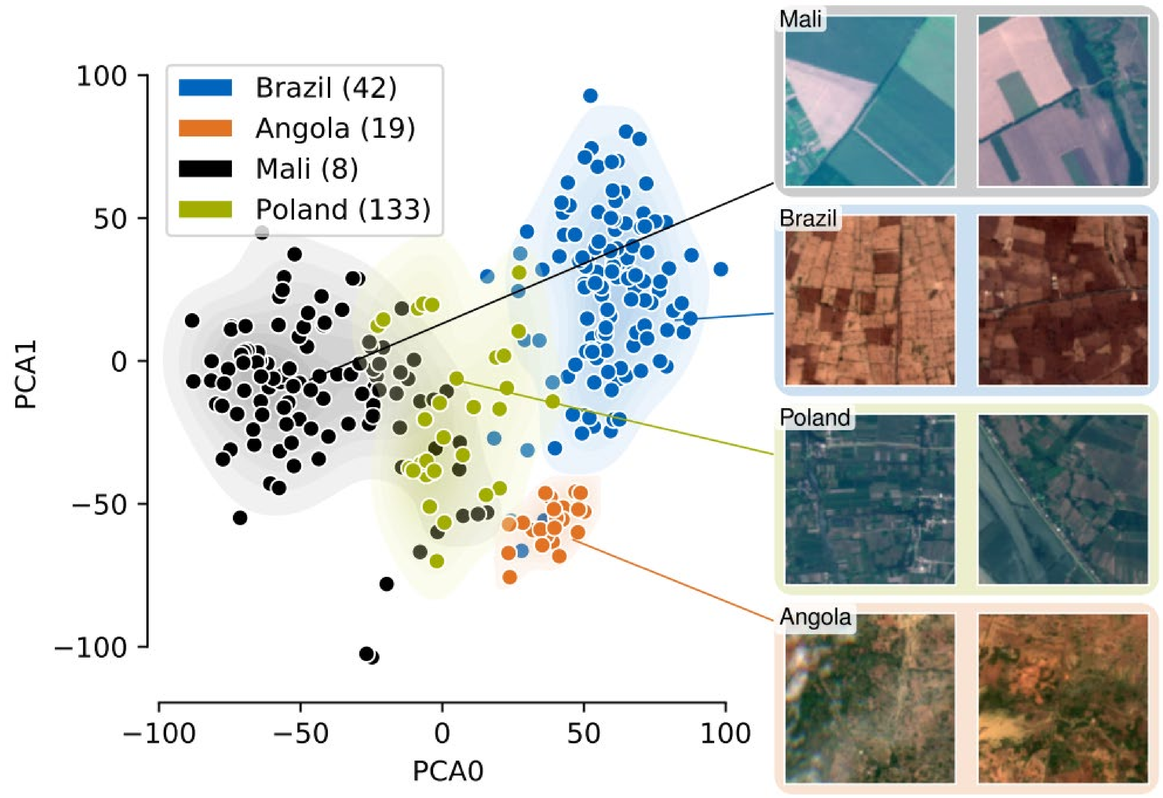
$$\boldsymbol{\nabla}_{\theta} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}}) = \frac{\boldsymbol{d\theta'}}{\boldsymbol{d\theta}} \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta'}} \ell(\boldsymbol{\theta'}, \boldsymbol{D^{new}})$$

Will we get higher order derivatives?

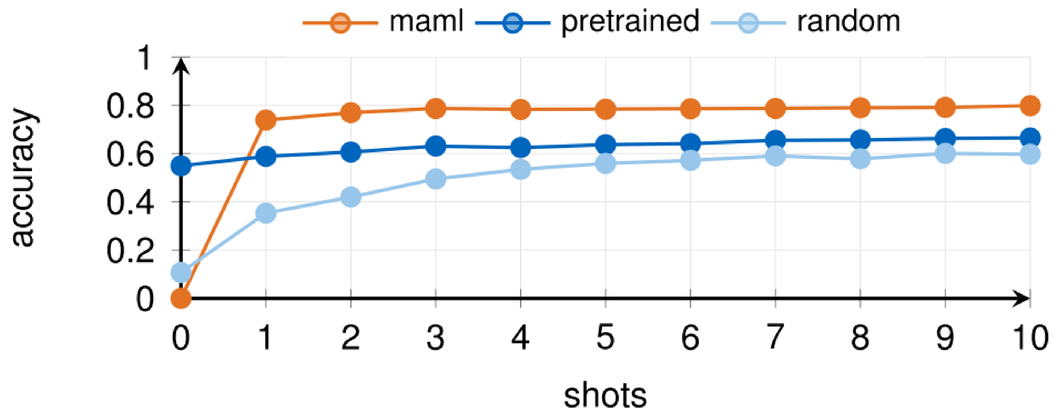
The answer is no …

MAML for few-shot learning

Show improved accuracy for few-shot learning tasks, e.g. cropland image classification from different countries.



125 regions, 4 seasons each region



Few-shot learning setup

Meta-Learning for Few-Shot Land Cover Classification. 2020. https://arxiv.org/abs/2004.13390

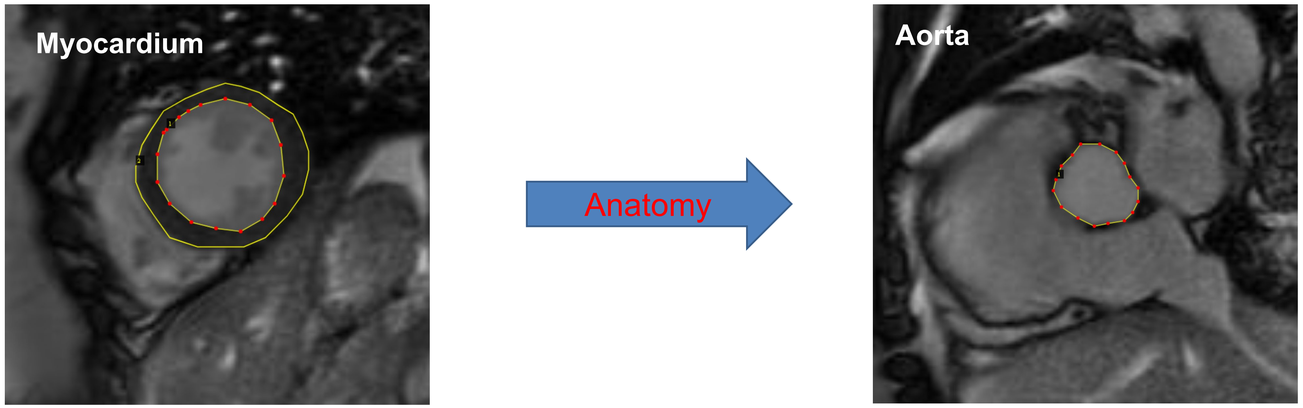
MAML for old-new tasks

Old task with a lot of labelled data

Often called “support”

New task with a few labelled data

Often called “query”



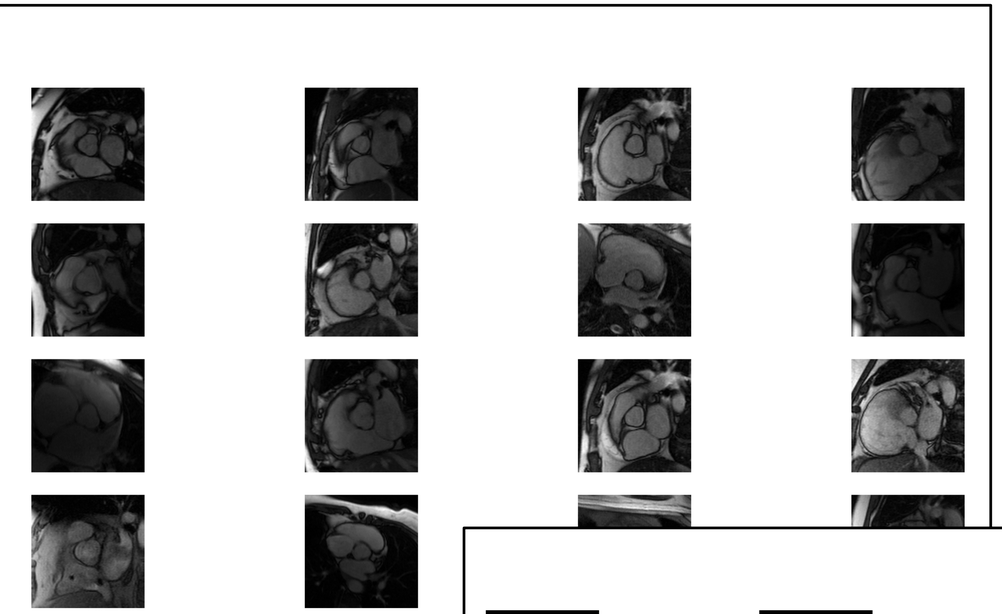
Train set:
- 2,581 cine images
- 120 aorta images

Query set
- 200 aorta images

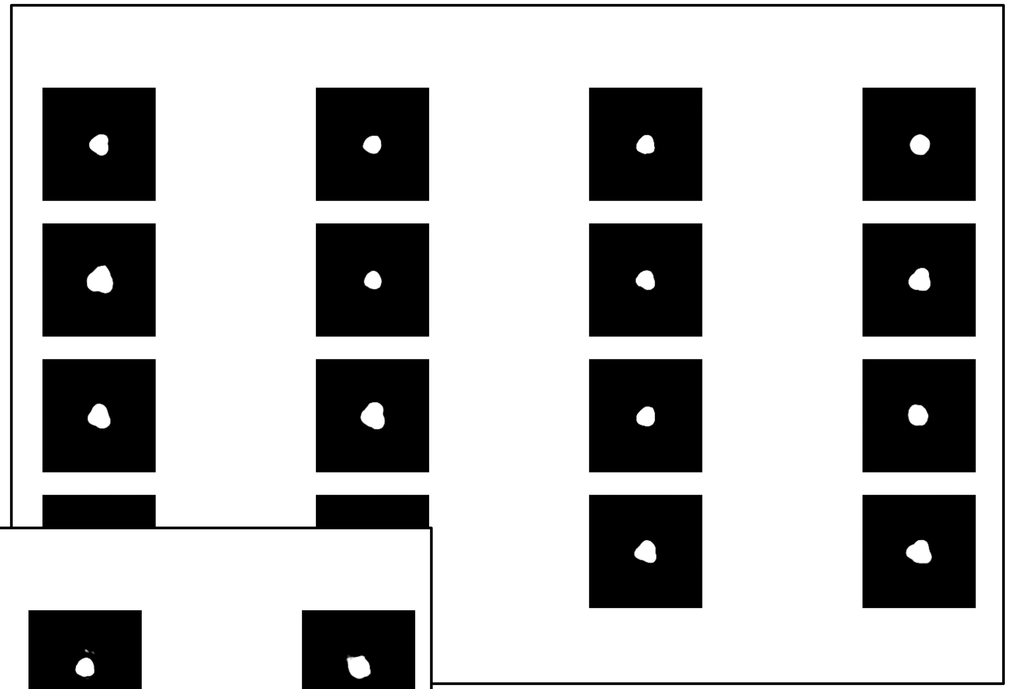
Test set
- 100 new images

- Using existing labelled data to help new tasks
- Can meta-learning help speedup development ? To break barrier between:
  - patient cohort
  - anatomy/imaging
  - disease
  - hospitals/countries
  - … …

MAML for old-new tasks

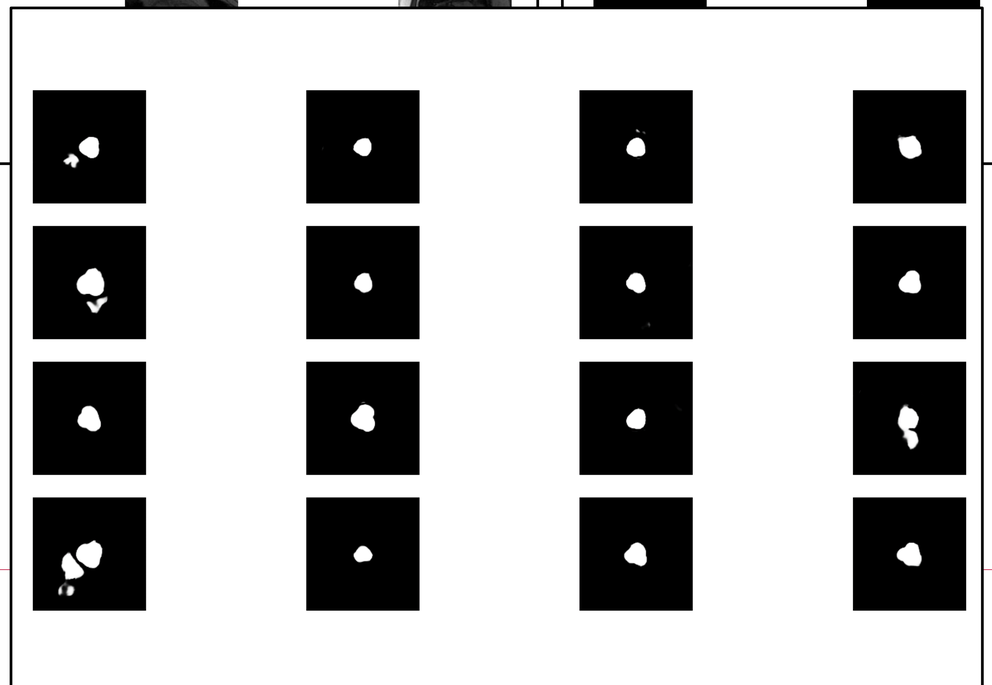


Test cases



MAML

Dice mean: 0.95



Supervised Learning

Dice mean: 0.88

Further developments

Development	What is for
First-order MAML https://arxiv.org/abs/1803.02999	Only keep the first term $\nabla_{\theta}\ell(\boldsymbol{\theta}', \boldsymbol{D}^{new}) \approx \nabla_{\boldsymbol{\theta}'}\ell(\boldsymbol{\theta}', \boldsymbol{D}^{new})$
Implicit MAML https://arxiv.org/abs/1909.04630	Derive meta-gradient using the implicit function theorem; reduce memory footprint
AlphaMAML https://arxiv.org/abs/1905.07435	Automated tune the inner learning rate
MAML++ https://arxiv.org/abs/1810.09502	Improve the generalization performance, convergence speed and computational overhead
Reptile https://arxiv.org/abs/1803.02999	Simplified first-order method with faster updates
ANIL https://arxiv.org/abs/1909.09157	A simplification of MAML where we remove the inner loop for all but the (task-specific) head of a MAML-trained network

Open-source libraries

for all task i:

Sample a mini-batch from $\boldsymbol{D}^i$

Compute updated parameter:

$$\theta' = \theta - \alpha \nabla_\theta \ell(\boldsymbol{D}^i, \theta)$$

Compute gradient $\boldsymbol{\nabla_\theta \ell(\theta', D^{new})}$

grad += $\boldsymbol{\nabla_\theta \ell(\theta', D^{new})}$

- Require to extract and reset model parameters
- Pytorch model often contains its parameter in layers
- Inconvenient to apply MAML on existing models

∇ higher

higher is a library providing support for higher-order optimization, e.g. through unrolled first-order optimization loops, of "meta" aspects of these loops.

https://github.com/facebookresearch/higher

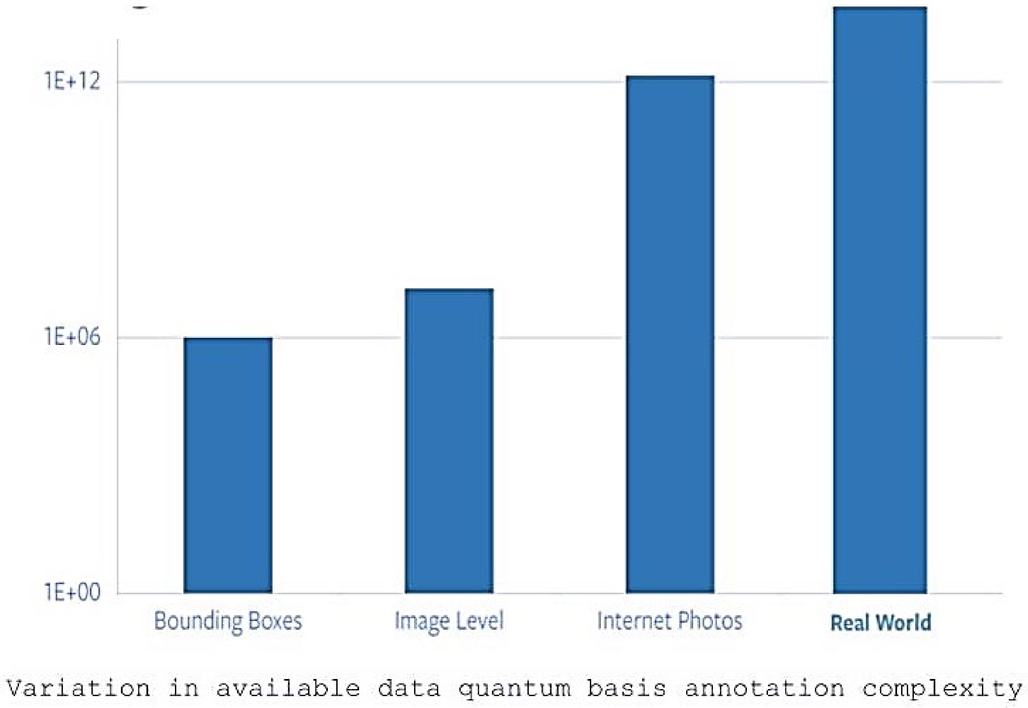
```
model = MyModel()
opt = torch.optim.Adam(model.parameters())

# When you want to branch from the current state of your model and unroll
# optimization, follow this example. This context manager gets a snapshot of the
# current version of the model and optimizer at the point where you want to
# start unrolling and create a functional version `fmodel` which executes the
# forward pass of `model` with implicit fast weights which can be read by doing
# `fmodel.parameters()`, and a differentiable optimizer `diffopt` which ensures
# that at each step, gradient of `fmodel.parameters()` with regard to initial
# fast weights `fmodel.parameters(time=0)` (or any other part of the unrolled
# model history) is defined.

with higher.innerloop_ctx(model, opt) as (fmodel, diffopt):
    for xs, ys in data:
        logits = fmodel(xs)  # modified `params` can also be passed as a kwarg
        loss = loss_function(logits, ys)  # no need to call loss.backwards()
        diffopt.step(loss)  # note that `step` must take `loss` as an argument!
        # The line above gets P[t+1] from P[t] and loss[t]. `step` also returns
        # these new parameters, as an alternative to getting them from
        # `fmodel.fast_params` or `fmodel.parameters()` after calling
        # `diffopt.step`.

        # At this point, or at any point in the iteration, you can take the
        # gradient of `fmodel.parameters()` (or equivalently
        # `fmodel.fast_params`) w.r.t. `fmodel.parameters(time=0)` (equivalently
        # `fmodel.init_fast_params`). i.e. `fast_params` will always have
        # `grad_fn` as an attribute, and be part of the gradient tape.

    # At the end of your inner loop you can obtain these e.g. ...
    grad_of_grads = torch.autograd.grad(
        meta_loss_fn(fmodel.parameters()), fmodel.parameters(time=0))
```

National Heart, Lung, and Blood Institute

Outline

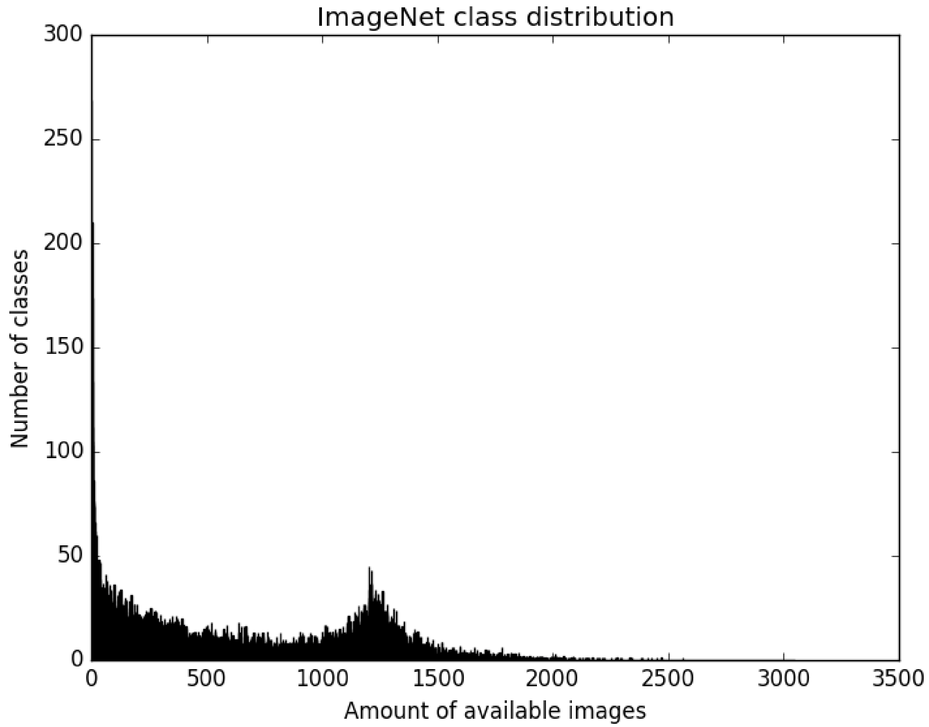
- Transfer learning
- Meta Learning
- **Contrastive learning**

Self-supervised learning



It is impossible to label even a small portion of available data …

https://atcold.github.io/pytorch-Deep-Learning/en/week10/10-1/



Classes: 21841

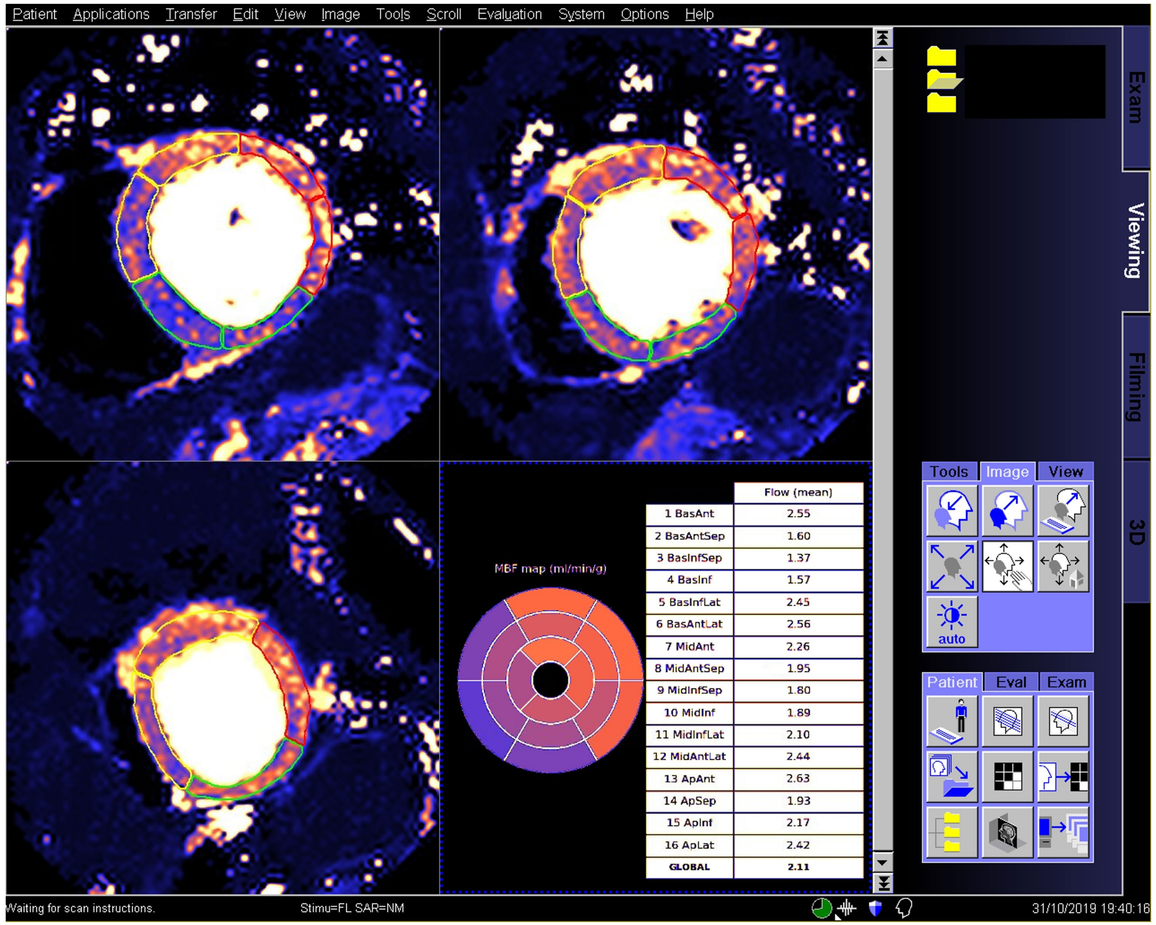
Images: 14,197,122

Data distribution has long tail

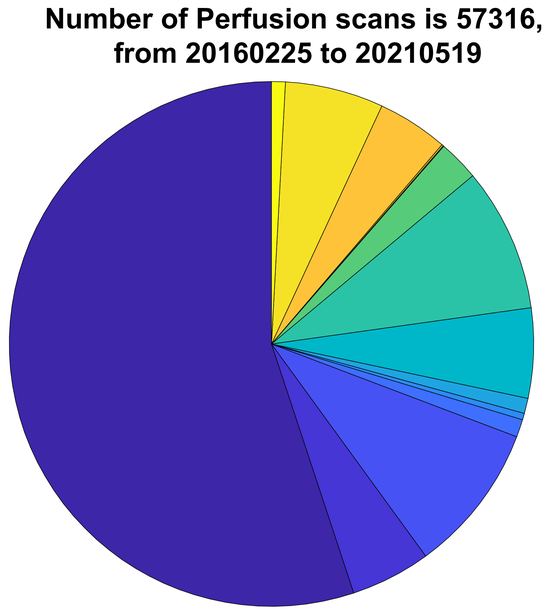
https://datascience.stackexchange.com/questions/11777/what-is-the-distribution-of-categories-in-imagenet-training-set-ilsvrc2012

Self-supervised learning

- Much harder to establish labels requiring domain expertise



Total data available:



Labelled data available:

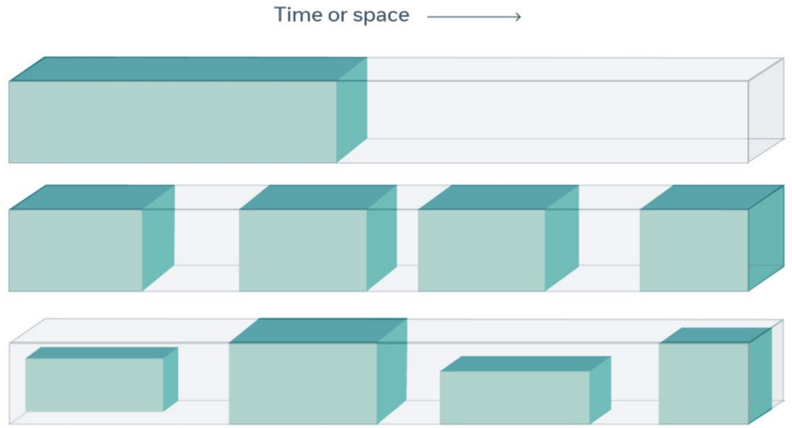
1825 perfusion series from 1034 patients

1.8%

NIH National Heart, Lung, and Blood Institute

Self-supervised learning

- *Self-supervised learning obtains supervisory signals from the data itself, often leveraging the underlying structure in the data.
- Often set up as a training task to predict unobserved or hidden part of the input from any observed or unhidden part of the input.
- Aim to learn the representation/”common sense” from large amount of unlabeled data to boost performance of downstream tasks (e.g. classification, detection etc.)



Predict future sentences given the prompt

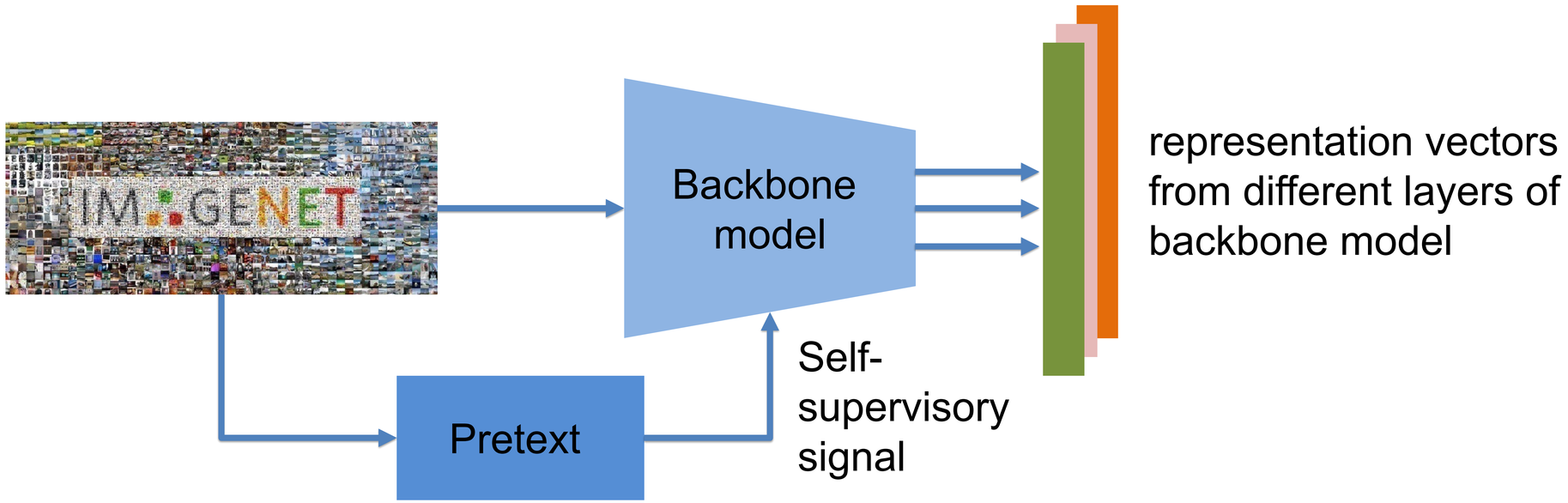
Predict masked works

Predict next video frame

* https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence
https://t.co/BZZgcTW4jO?amp=1

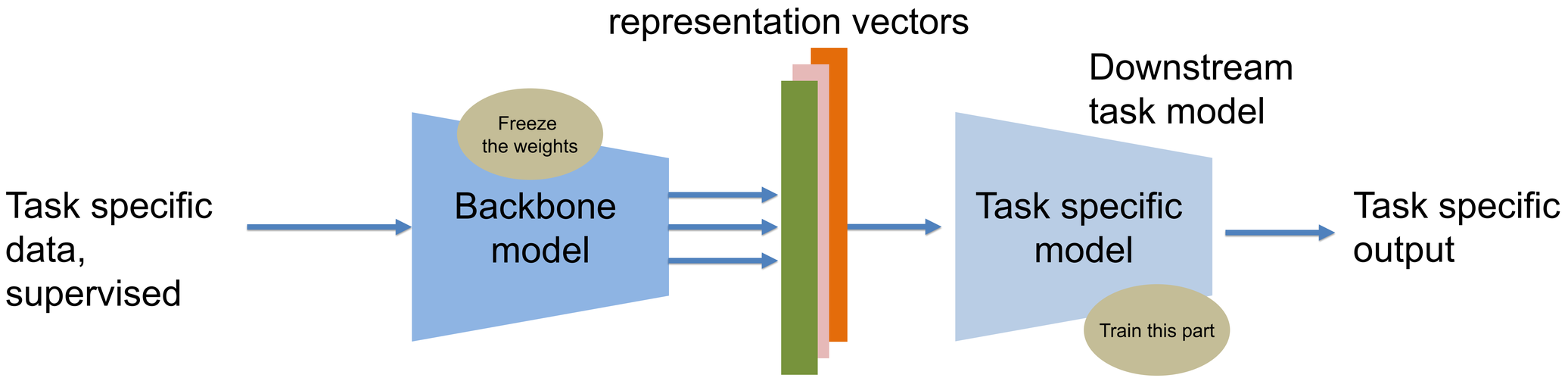
“Pretext” set up for SSL

- Pretext tasks aim to learn data representation with self-established “labels” without human inputs
- Learned data representation is used to help downstream tasks e.g. object segmentation, abnormality detection, …



“Pretext” set up for SSL

- Pretext tasks aim to learn data representation with self-established “labels” without human inputs
- Learned data representation is used to help downstream tasks e.g. object segmentation, abnormality detection, …

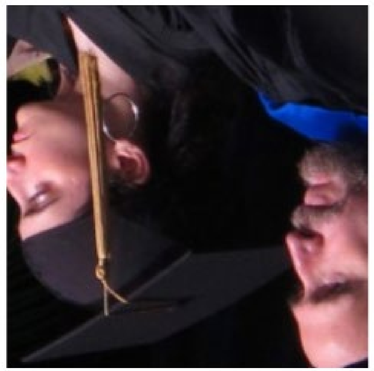


Or, fine tune the entire network; Or, run MAML; …

Pretext as the random rotation

90° rotation

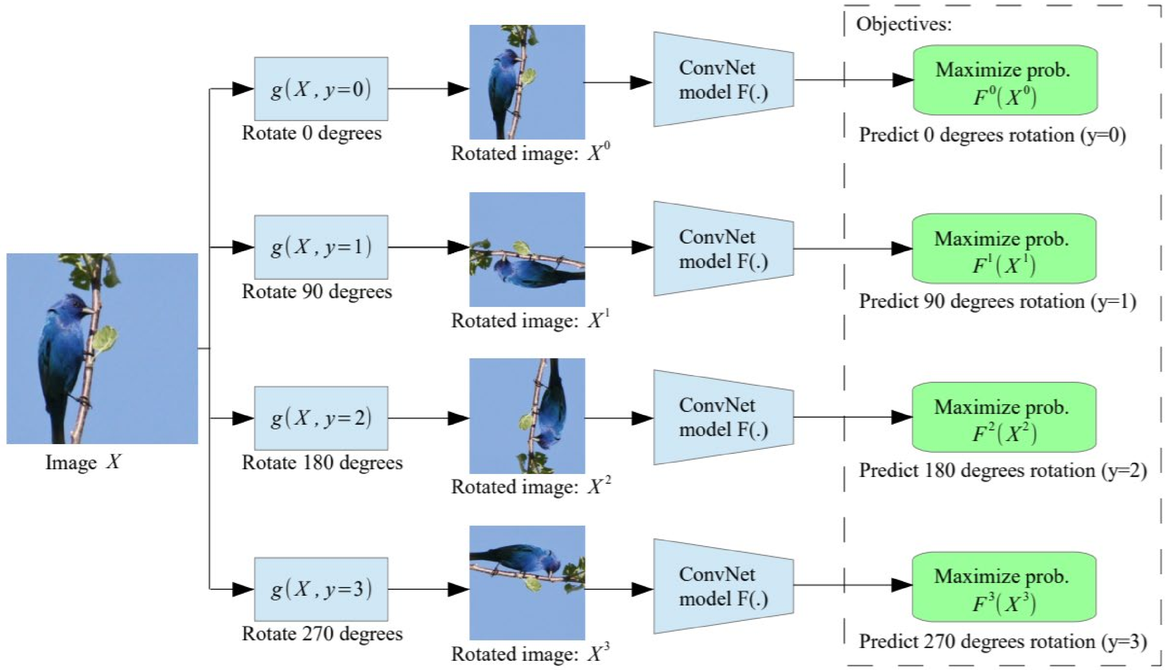
270° rotation



180° rotation

0° rotation

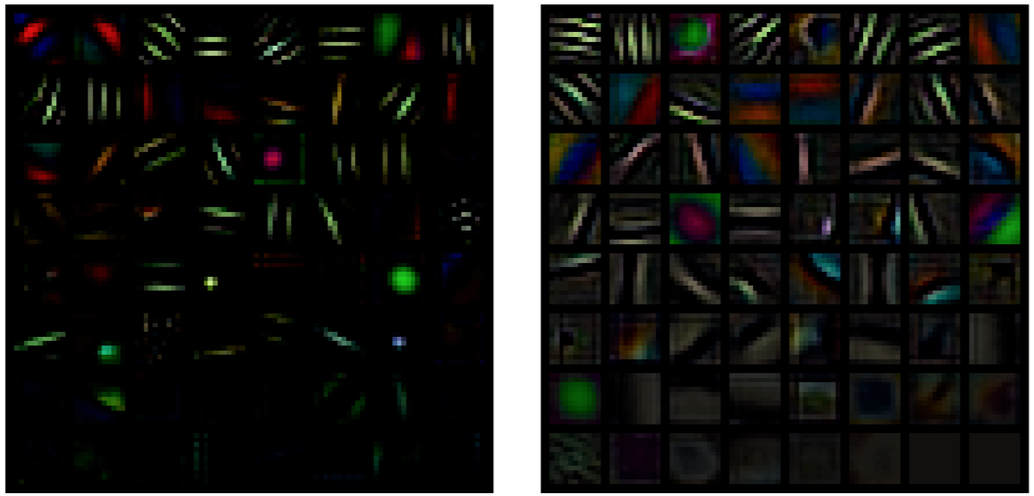
270° rotation



Unsupervised Representation Learning by Predicting Image Rotations. 2018. https://arxiv.org/abs/1803.07728

Pretext as the random rotation

Method	Accuracy
Supervised NIN	92.80
Random Init. + conv	72.50
(Ours) RotNet + non-linear	89.06
(Ours) RotNet + conv	**91.16**
(Ours) RotNet + non-linear (fine-tuned)	91.73
(Ours) RotNet + conv (fine-tuned)	92.17
Roto-Scat + SVM Oyallon & Mallat (2015)	82.3
ExemplarCNN Dosovitskiy et al. (2014)	84.3
DCGAN Radford et al. (2015)	82.8
Scattering Oyallon et al. (2017)	84.7

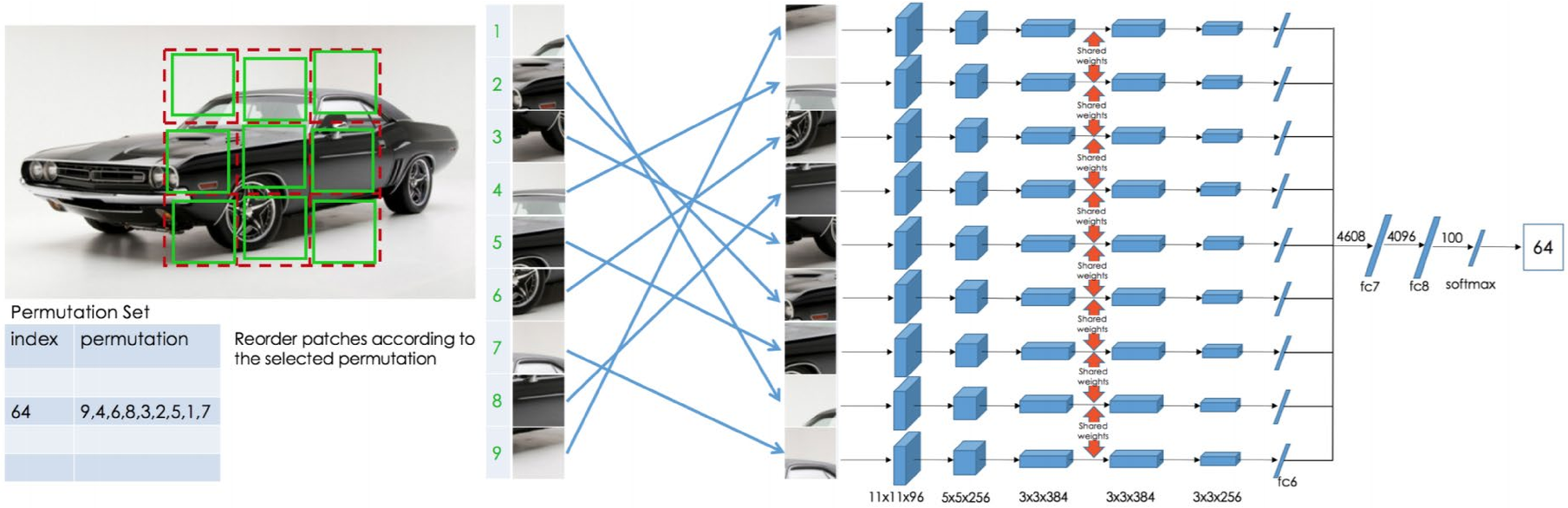


(a) **Supervised** (b) **Self-supervised to recognize rotations**

Figure 4: First layer filters learned by a AlexNet model trained on **(a)** the supervised object recognition task and **(b)** the self-supervised task of recognizing rotated images. We observe that the filters learned by the self-supervised task are mostly oriented edge filters on various frequencies and, remarkably, they seem to have more variety than those learned on the supervised task.

Unsupervised Representation Learning by Predicting Image Rotations. 2018. https://arxiv.org/abs/1803.07728

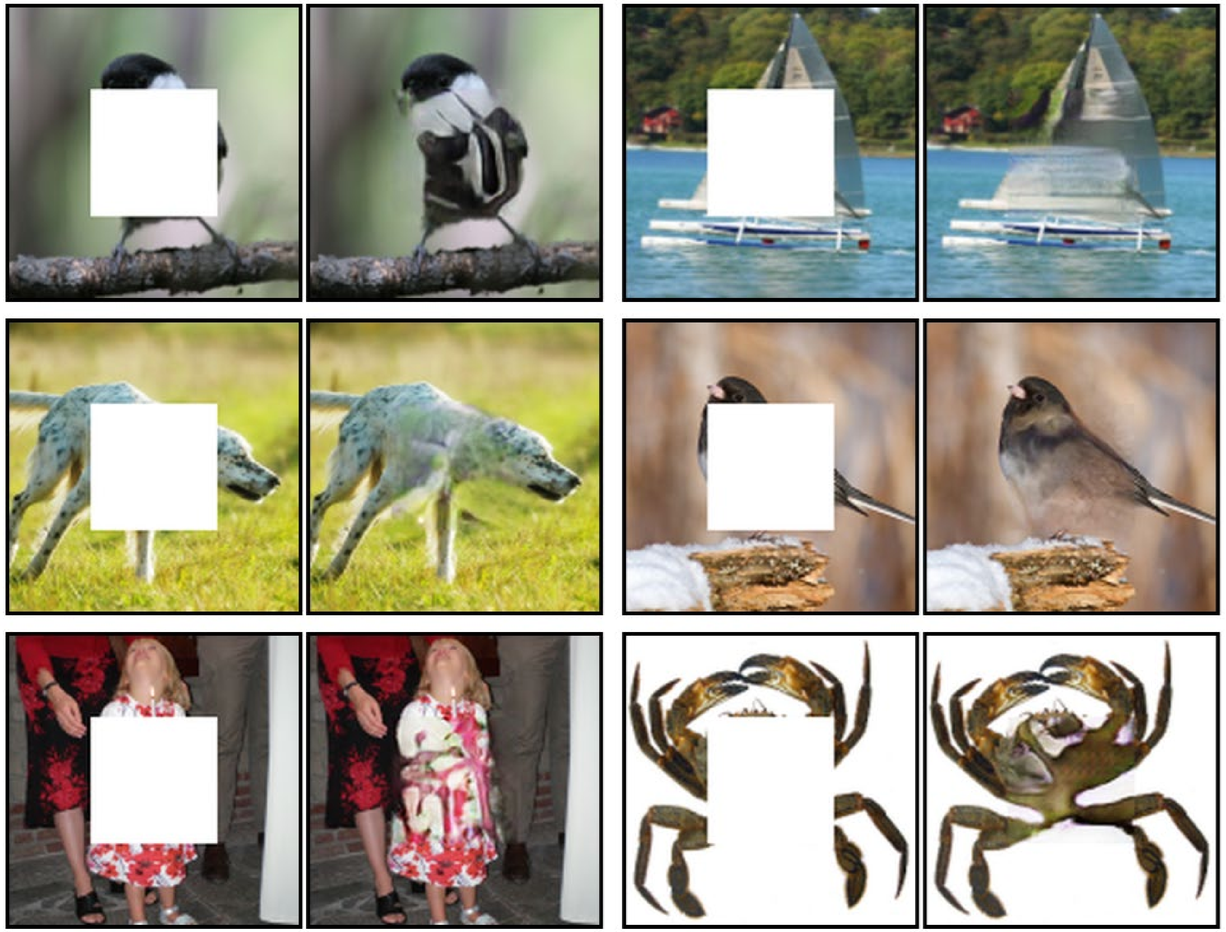
Pretext as the Jigsaw Puzzle game



Randomly pick from one pre-defined permutation

Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. https://arxiv.org/abs/1603.09246

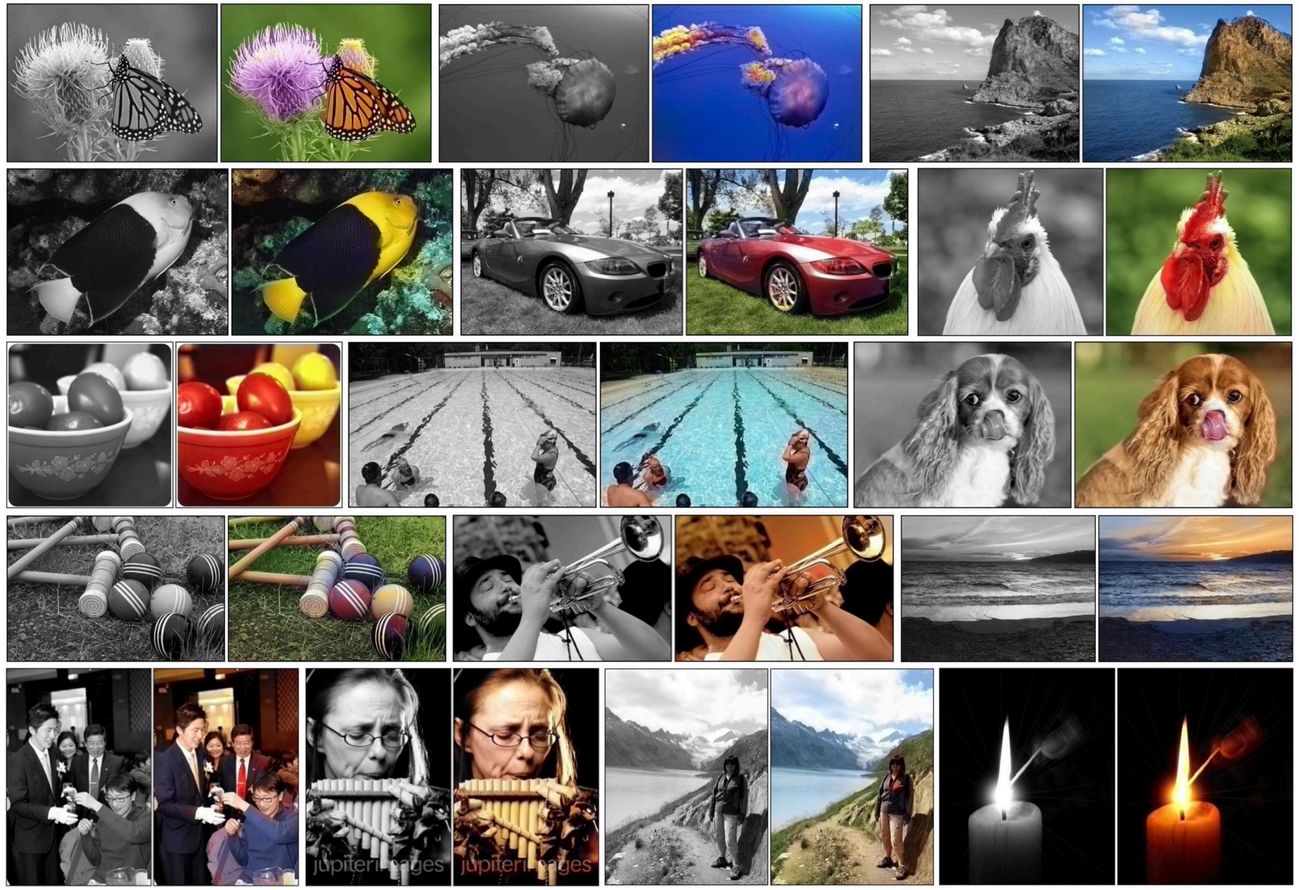
Pretext as the inpainting

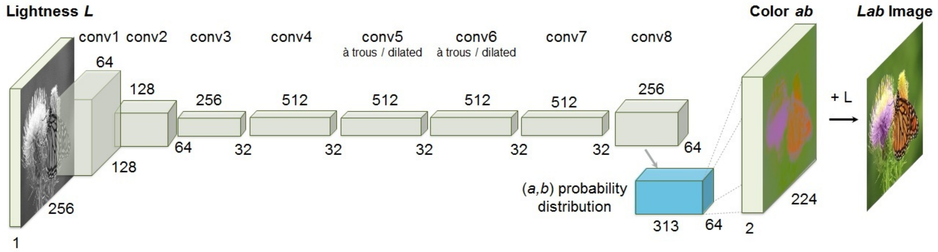


Fill in the masked portion in the image

Context Encoders: Feature Learning by Inpainting. https://arxiv.org/pdf/1604.07379.pdf

Pretext as the coloring task





Trained on 1.3M ImageNet images

Colorful Image Colorization. https://arxiv.org/abs/1603.08511
Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction. https://arxiv.org/abs/1611.09842

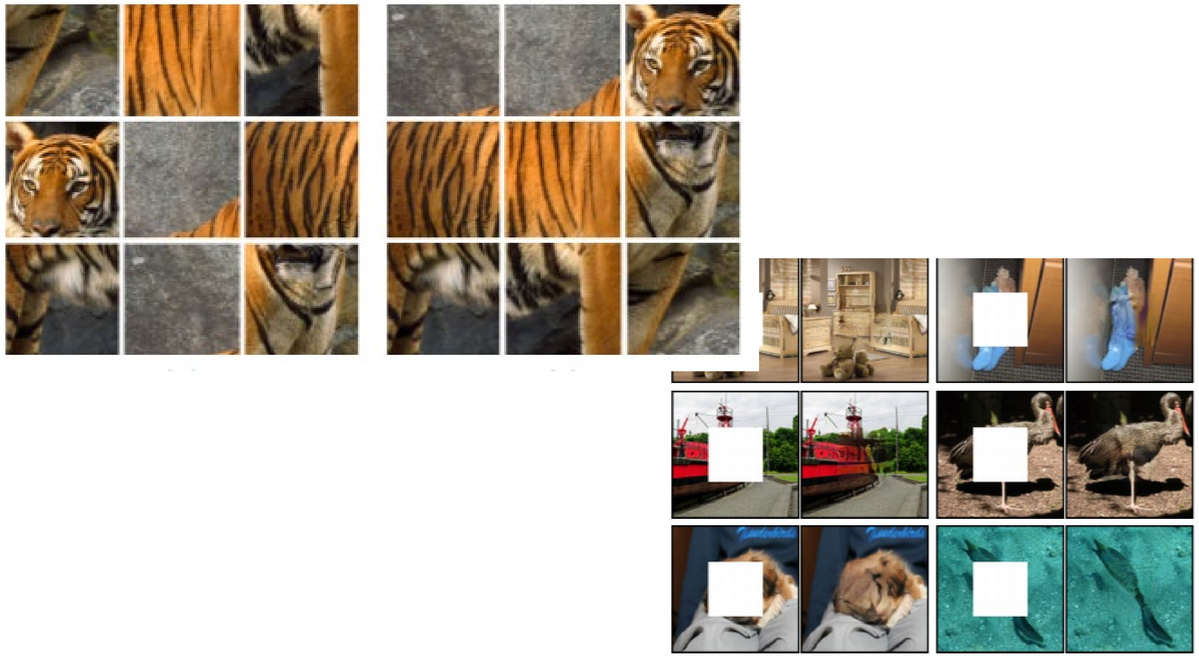
http://richzhang.github.io/colorization/

https://demos.algorithmia.com/colorize-photos

SSL as pretexts

Pros

- Able to generate self-supervisory signal
- Able to use large amount of unlabeled data



Cons

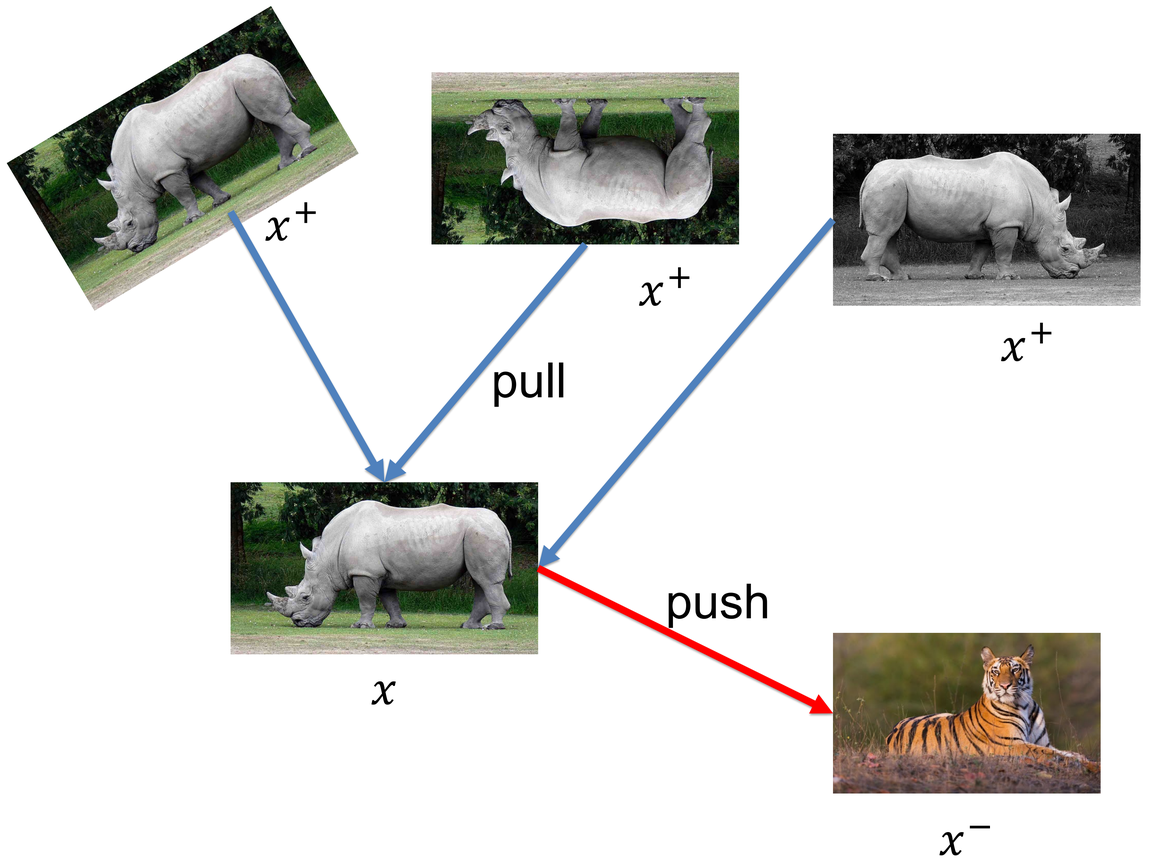
- Still have “feature engineering” to select pretext tasks
- Need careful design the interaction between backbone and task-specific components
- Does not conduct optimization in the representation space, but in “task” space

SSL as contrastive learning

- Define a general framework to optimize in the representation space
- The goal of optimization is :

  Given a reference sample x,

  - for its positive neighbors x^+, reduce its distance to reference in representation space
  - for its negative neighbors x^-, increase its distance to reference in representation space



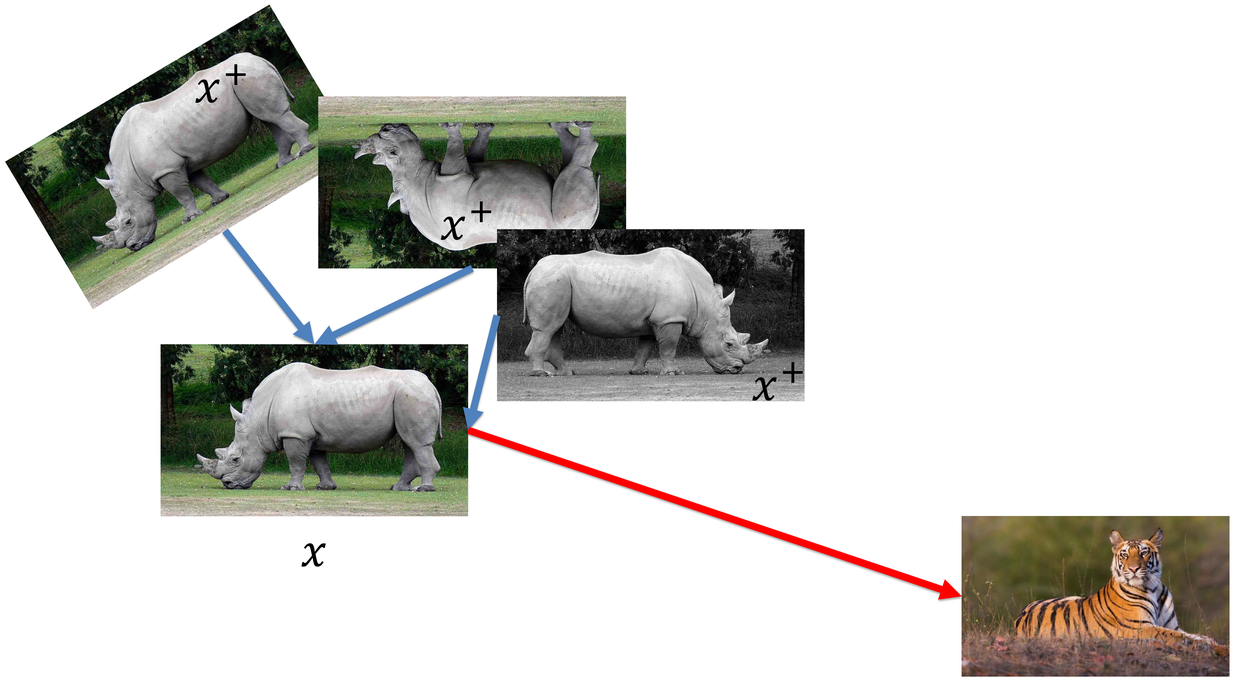
SSL as contrastive learning

- Define a general framework to optimize in the representation space
- The goal of optimization is :

  Given a reference sample x,

  - for its positive neighbors x^+, reduce its distance to reference in representation space
  - for its negative neighbors x^-, increase its distance to reference in representation space

After learning



Contrastive learning

$$score(f(x), f(x^+)) \gg score(f(x), f(x^-))$$

reference sample x

positive sample x^+,

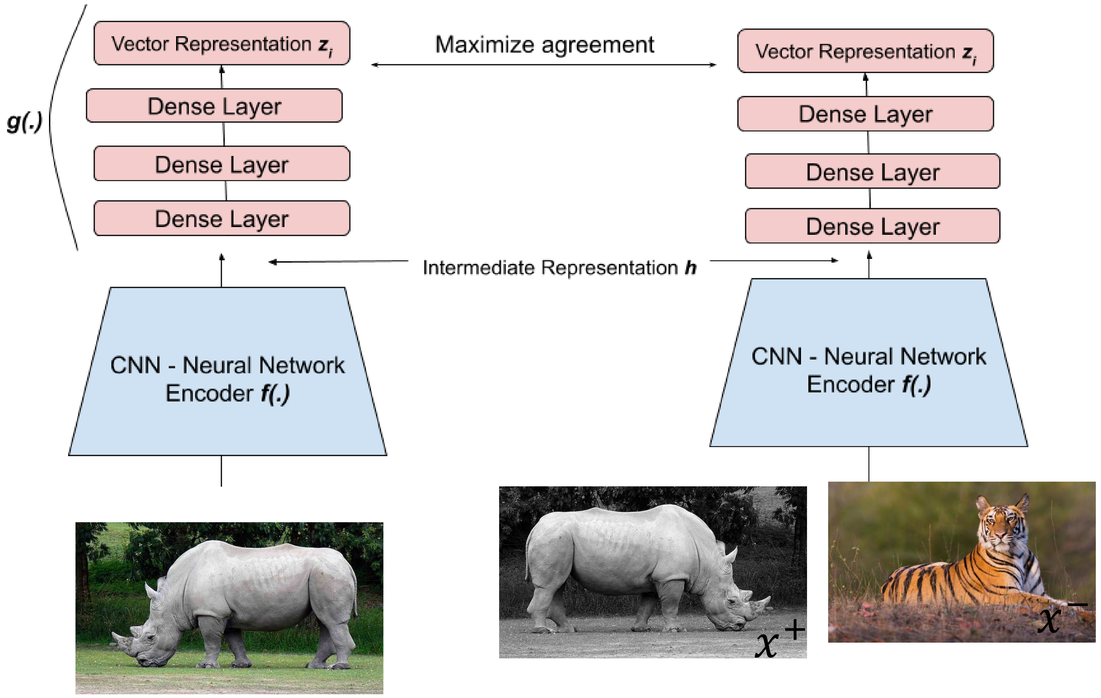
negative sample x^-

$score$ is a function measure the distance between the learned representation between the reference and positive/negative samples

f is the encoding model to be learned from data

g is called the projection header

- converting representation to the space for loss computation
- Empirically improve performance
- Mimic downstream task



SimCLR

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Contrastive learning

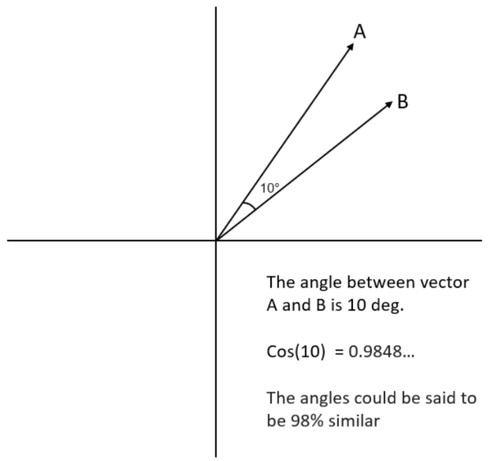
Score function

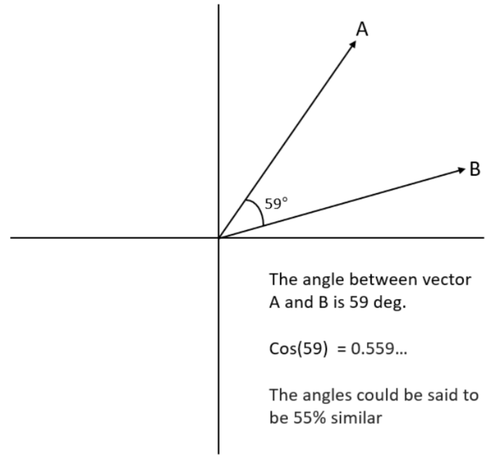
Common choices: L2, L1, cosine …

Given two vectors of attributes, *A* and *B*, the cosine similarity, $\cos(\theta)$, is represented using a dot product and magnitude as

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

where A_i and B_i are components of vector A and B respectively.





https://towardsdatascience.com/understanding-cosine-similarity-and-its-application-fd42f585296a

Contrastive learning

InfoNCE loss

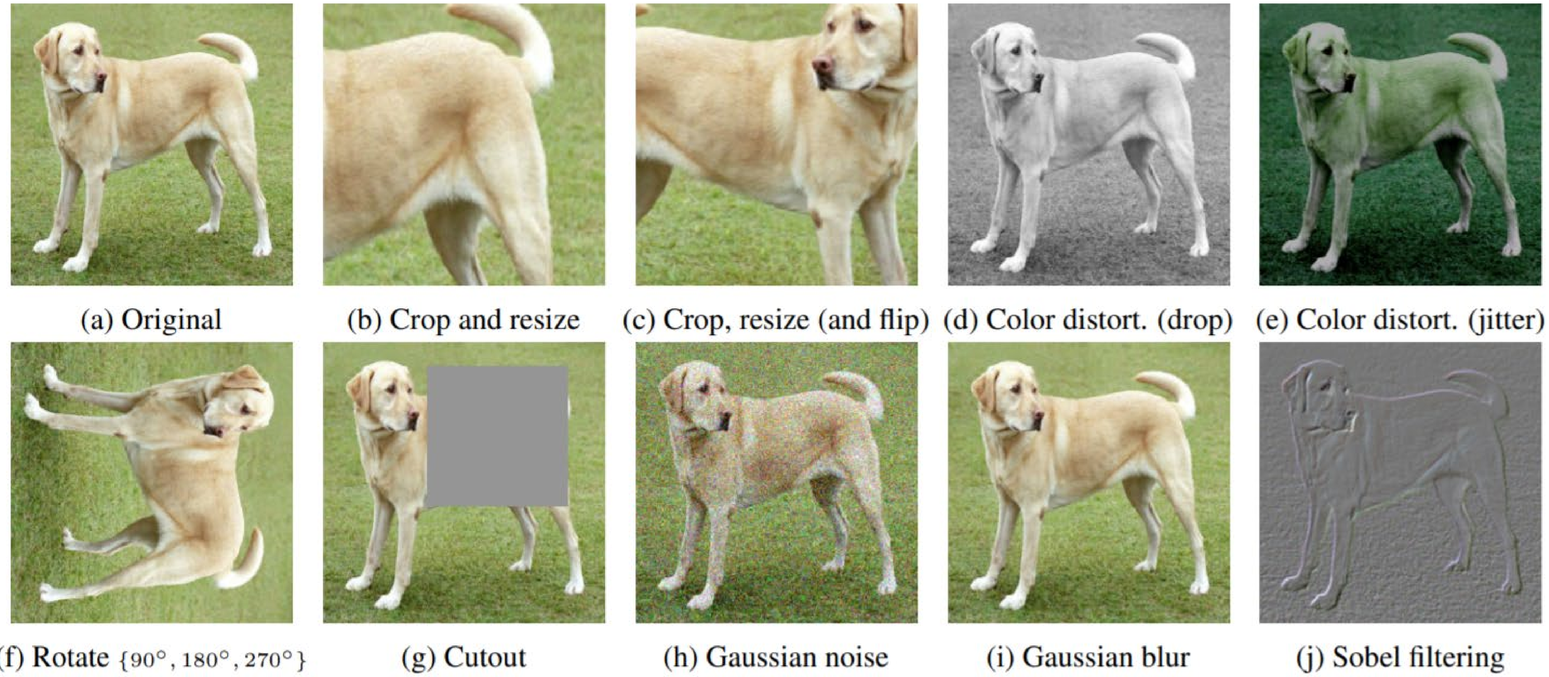
Give the reference sample, a positive sample and a set of negative sample

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

- Nothing more than the cross-entropy loss
- softmax over all scores between ref and positive/negative samples
- Normalized to become probability

Representation Learning with Contrastive Predictive Coding. https://arxiv.org/abs/1807.03748

Contrastive learning



- Define a set of data augmentation transformation
- Randomly sample from data augmentations and create positive samples

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Contrastive learning

SimCLR

- The training enforces the positive pairs to be close and the negative pairs to be far away in representation space
- MLP project head is partly due to the CE type loss used in the framework – often, the classification problem is easier to train than e.g. L2 norm
- Once the model is trained with large amount of data, it can learn a compact and effective representation for downstream tasks (a good encoder)

https://ai.googleblog.com/2020/04/advancing-self-supervised-and-semi.html

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Contrastive learning

SimCLR

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

T is a set of data augmentation

For every sample in the minibatch, apply the data augmentation to create two positive samples

Compute score between all generated samples

Compute InfoNCE loss
For the kth sample in mini-batch, 2k-1 and 2k are positive pair. 2k-1 and others are negative pairs

τ is the temperature parameter, boosting generalization performance

A Simple Framework for Contrastive Learning of Visual Representations. 2020. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

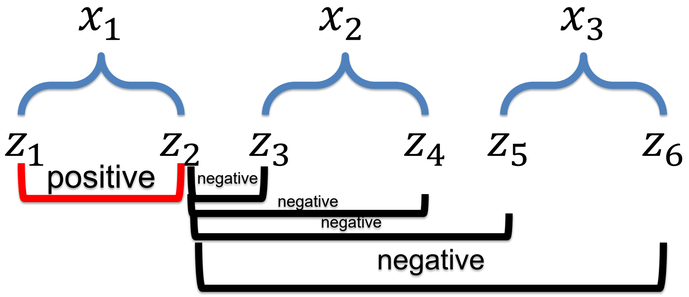
Contrastive learning

SimCLR

$$\textbf{define } \ell(i,j) \textbf{ as } \ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$$

$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$$

A mini-batch with three samples : x_1, x_2, x_3



$$\mathcal{L} = \frac{1}{6}[\ell(1,2) + \ell(2,1) + \ell(3,4) + \ell(4,3) + \ell(5,6) + \ell(6,5)]$$

$$\ell(1,2) = -log \frac{\exp(s(z_1, z_2))}{\exp(s(z_1, z_2)) + \exp(s(z_1, z_3)) + \exp(s(z_1, z_4)) + \exp(s(z_1, z_5)) + \exp(s(z_1, z_6))}$$

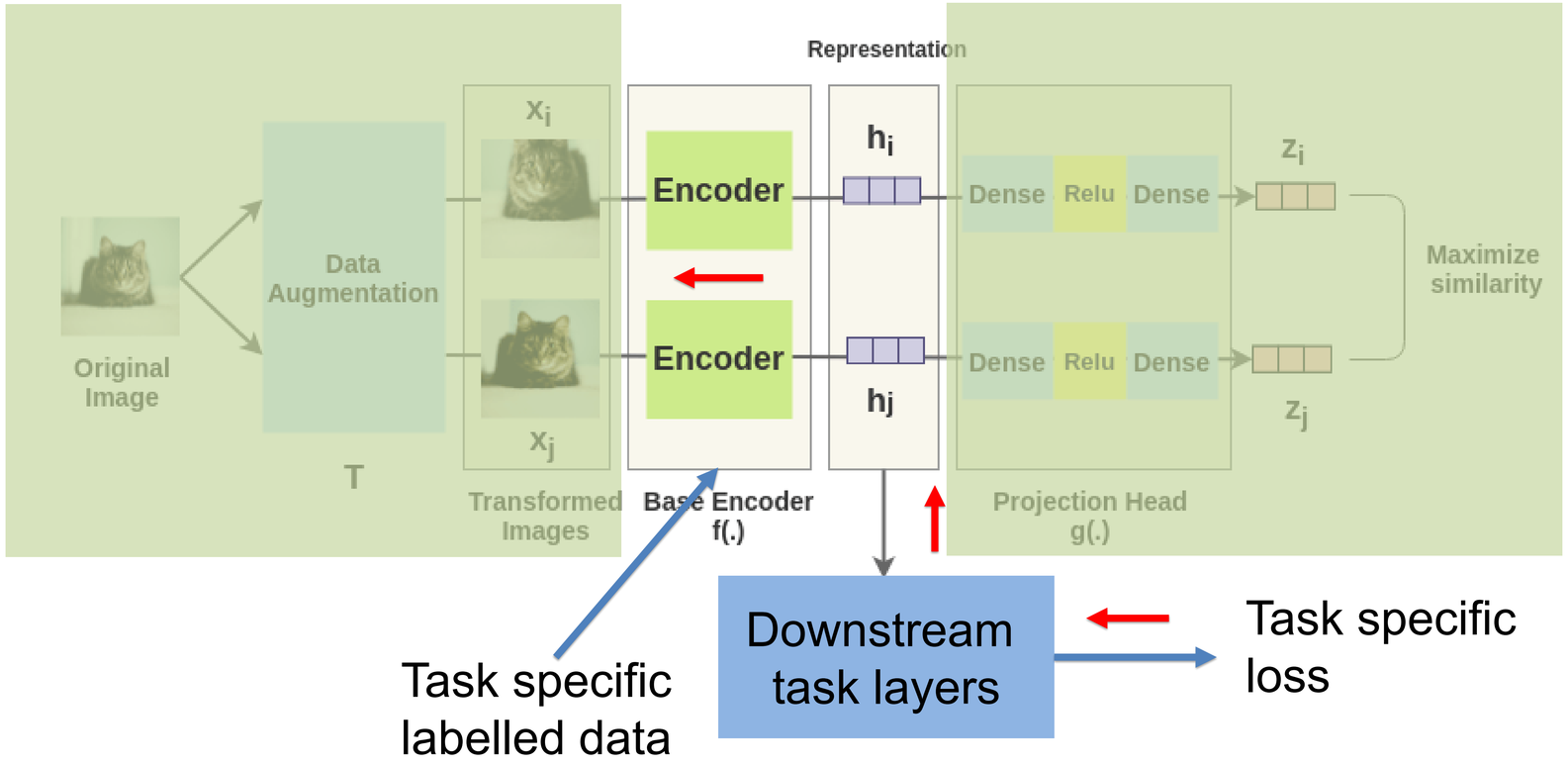
$$\ell(2,1) = -log \frac{\exp(s(z_2, z_1))}{\exp(s(z_2, z_1)) + \exp(s(z_2, z_3)) + \exp(s(z_2, z_4)) + \exp(s(z_2, z_5)) + \exp(s(z_2, z_6))}$$

Count positive and negative pairs correctly

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Contrastive learning

Fine-tune for task specific data



- Train task specific layers
- Or, fine tuning
- Or, Meta-Learning

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

https://amitness.com/2020/03/illustrated-simclr/

Contrastive learning

Very good data efficiency

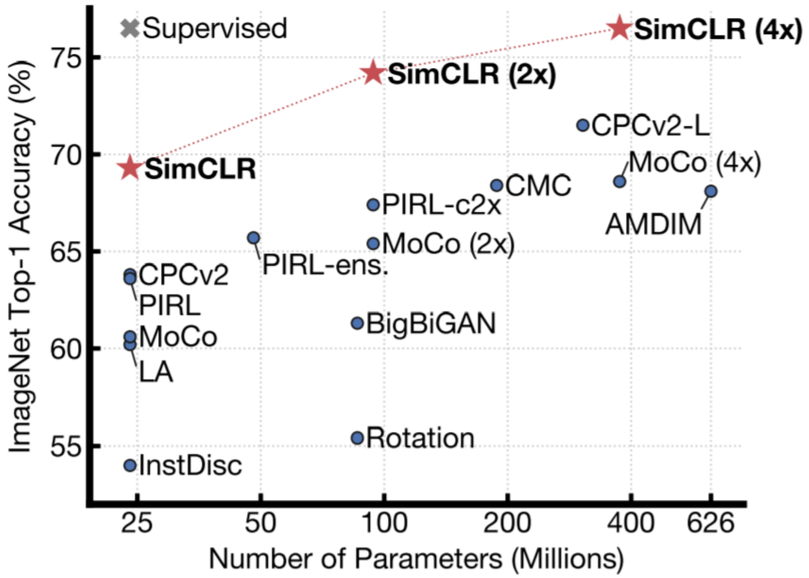


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

- First train SSL model on the ImageNet dataset
- Further fine-tune a linear MLP with a fraction of labelled data
- Having the projection header boosts the performance of a linear classifier trained on the SimCLR-learned representation by more than *10%*
- For 1% of labelled data, 76.5% / 93.2% top-1 / top-5 accuracy on ImageNet

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Contrastive learning

Performance is limited by the batch size for the number of available negative samples

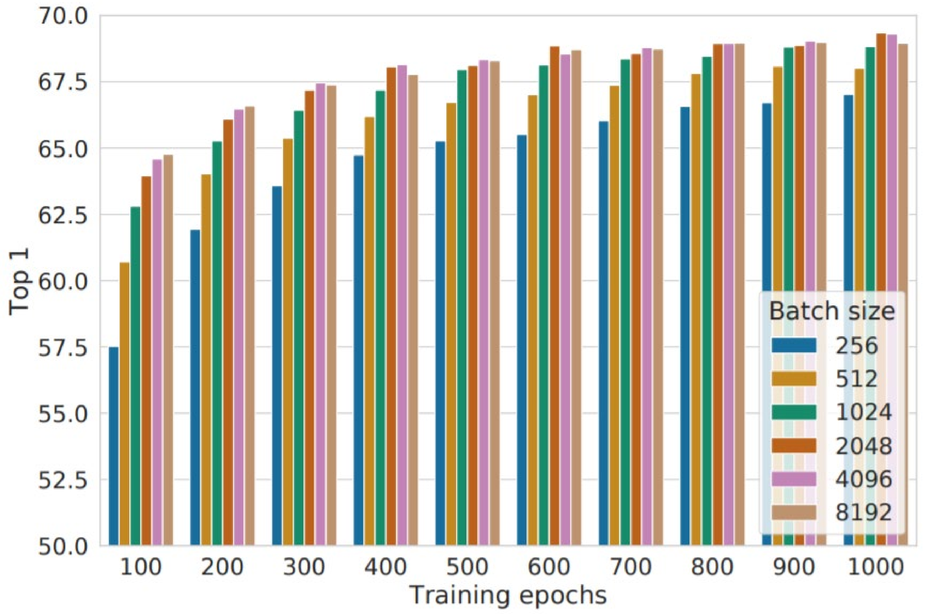


Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.[10]

- In the SimCLR, large batch size is required
- Original paper used Google Large TPU for training – not available for others

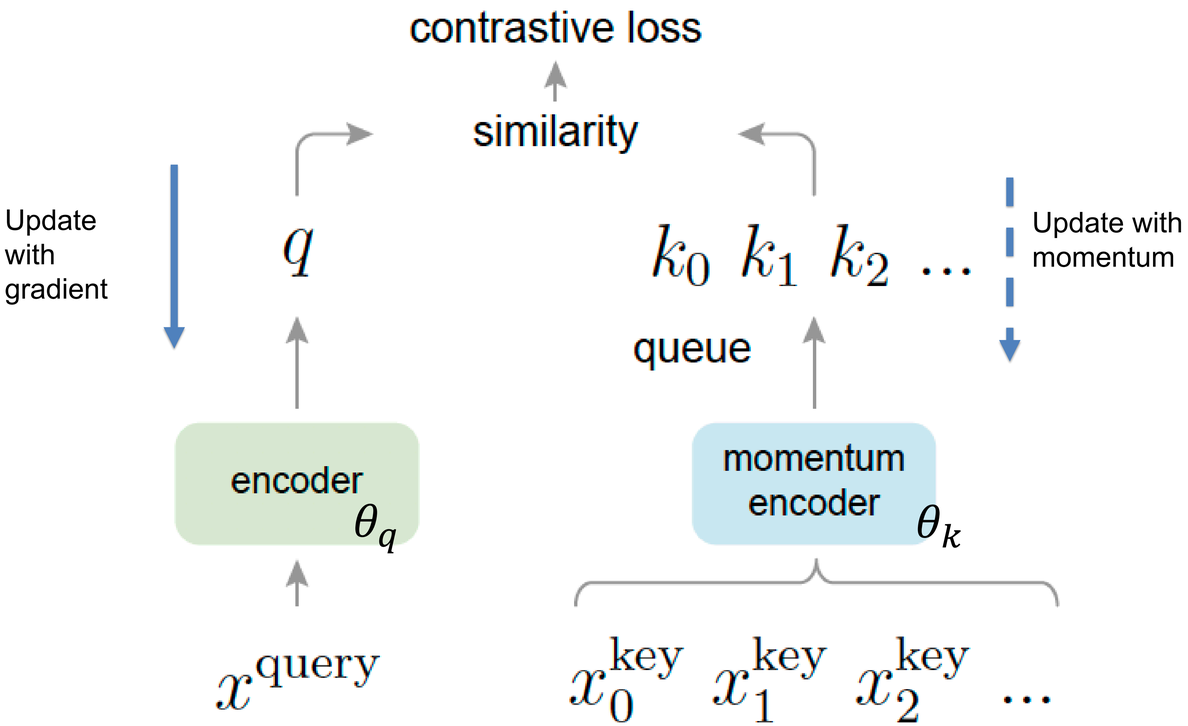
"With 128 TPU v3 cores, it takes ~1.5 hours to train our ResNet-50 with a batch size of 4096 for 100 epochs."

- Further improved by algorithms like Momentum Contrastive Learning (MoCo and MoCo v2, 2020, https://arxiv.org/abs/2003.04297)

A Simple Framework for Contrastive Learning of Visual Representations. https://arxiv.org/abs/2002.05709
Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020. https://arxiv.org/abs/2006.10029

Momentum Contrastive learning (MoCo)

Solve the problem of large batch size in SimCLR



https://github.com/facebookresearch/moco

- Keep a buffer of negative samples (x_i^{key})
- Update encoder for every mini-batch using gradient
- Update momentum encoder using momentum of encoder

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}$$

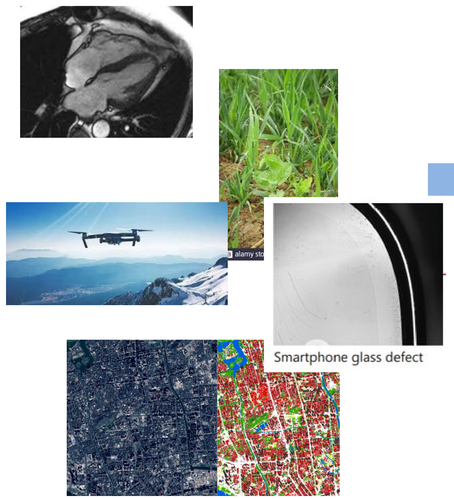
since backprop is not used to update momentum encoder (buffer should be large in size)

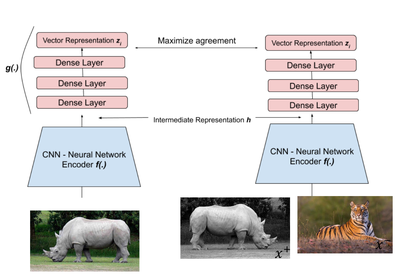
- Roll out negative samples by adding mini-batch into keys

Momentum Contrast for Unsupervised Visual Representation Learning. 2020. https://arxiv.org/abs/1911.05722

Contrastive learning as a new tool

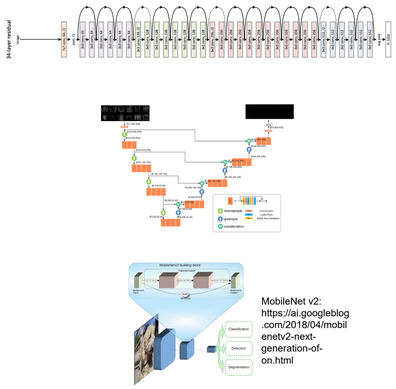
Large data collection for tasks



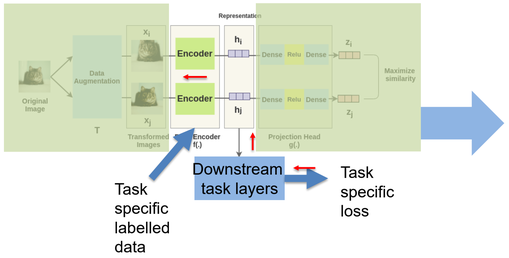


Contrastive learning

with



Baseline models for different tasks



Task specific training

Model deployment with less labels

NIH National Heart, Lung, and Blood Institute