Three Explorations on **Pre-Training**

an Analysis, an Approach, and an Architecture

Xinlei Chen

Microsoft V+L Summer Talk Series, 09/10/2021

facebook
Artificial Intelligence Research

Pre-Training is Important

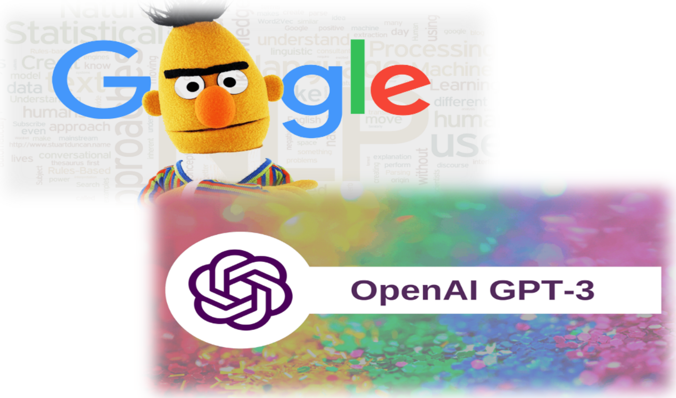
Vision

IMAGENET

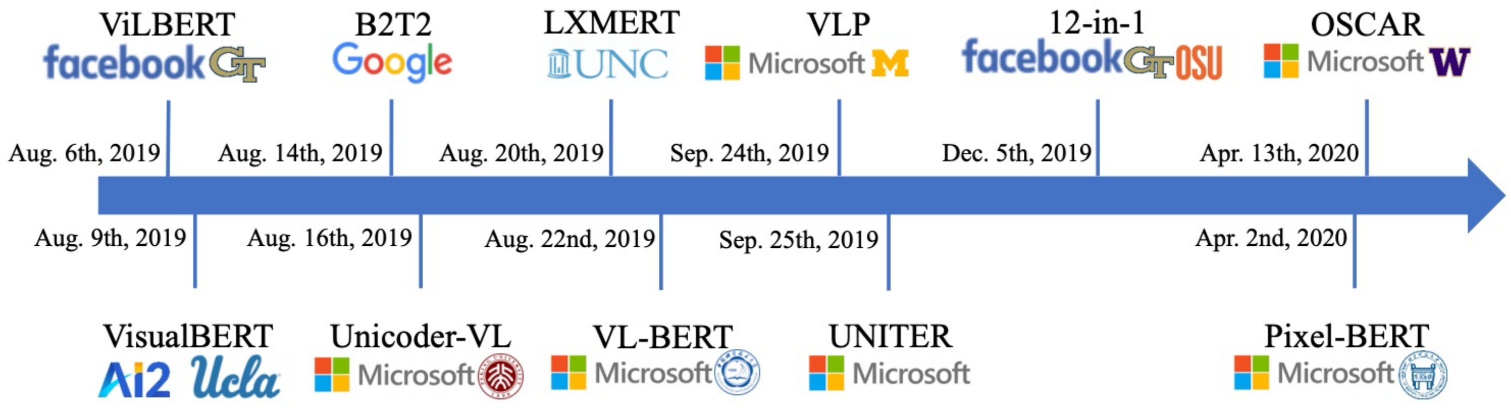
SHAPENET

ACTIVITYNET

Language



Vision + Language



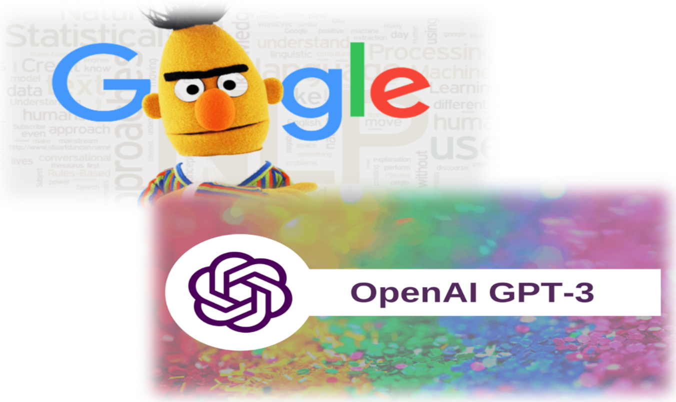
Vision

IMAGENET

SHAPENET

ACTIVITYNET

Language



Slide credit: https://rohit497.github.io/Recent-Advances-in-Vision-and-Language-Research/

Outline of this Talk

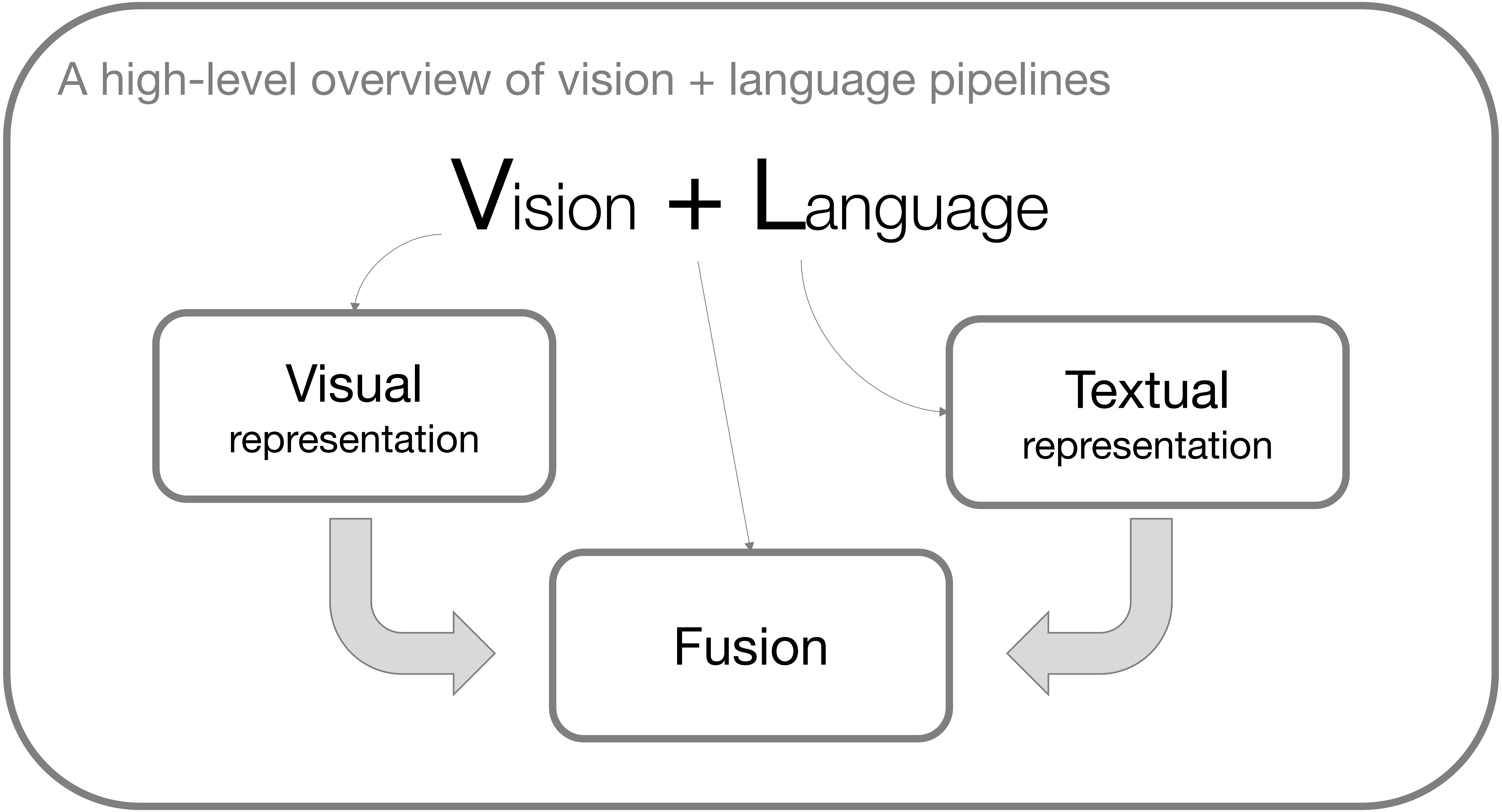
1. (analysis) visual feature pre-training for V + L tasks
2. (approach) self-supervised representation learning with SimSiam
3. (architecture) vision transformers for self-supervised learning

https://arxiv.org/abs/2001.03615
https://github.com/facebookresearch/grid-feats-vqa

Analysis: In Defense of Grid Features for Visual Question Answering

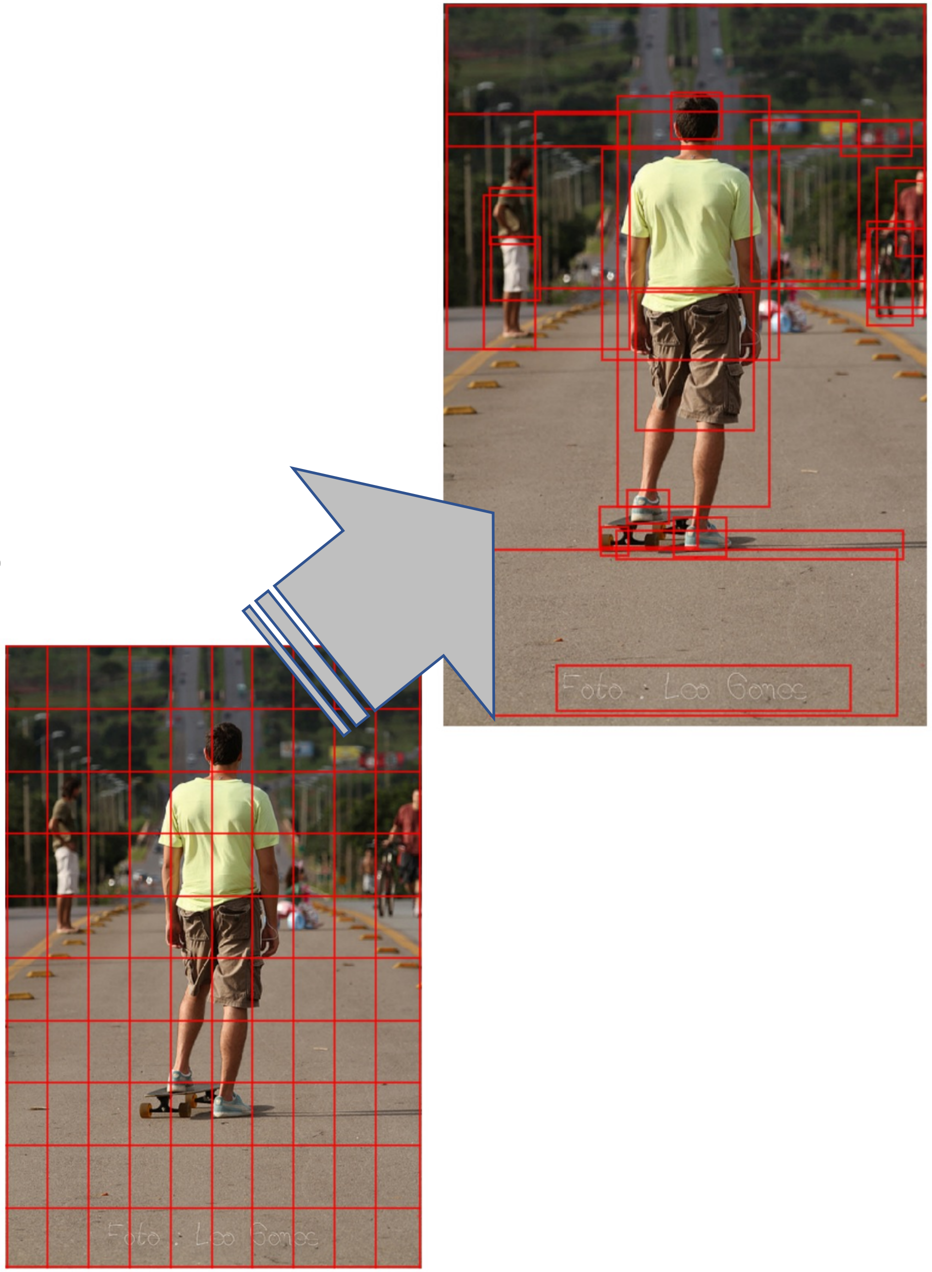
CVPR 2020: Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, Xinlei Chen

facebook
Artificial Intelligence Research


A high-level overview of vision + language pipelines
Vision + Language
Visual
representation
Textual
representation
Fusion

Bottom-Up Attention

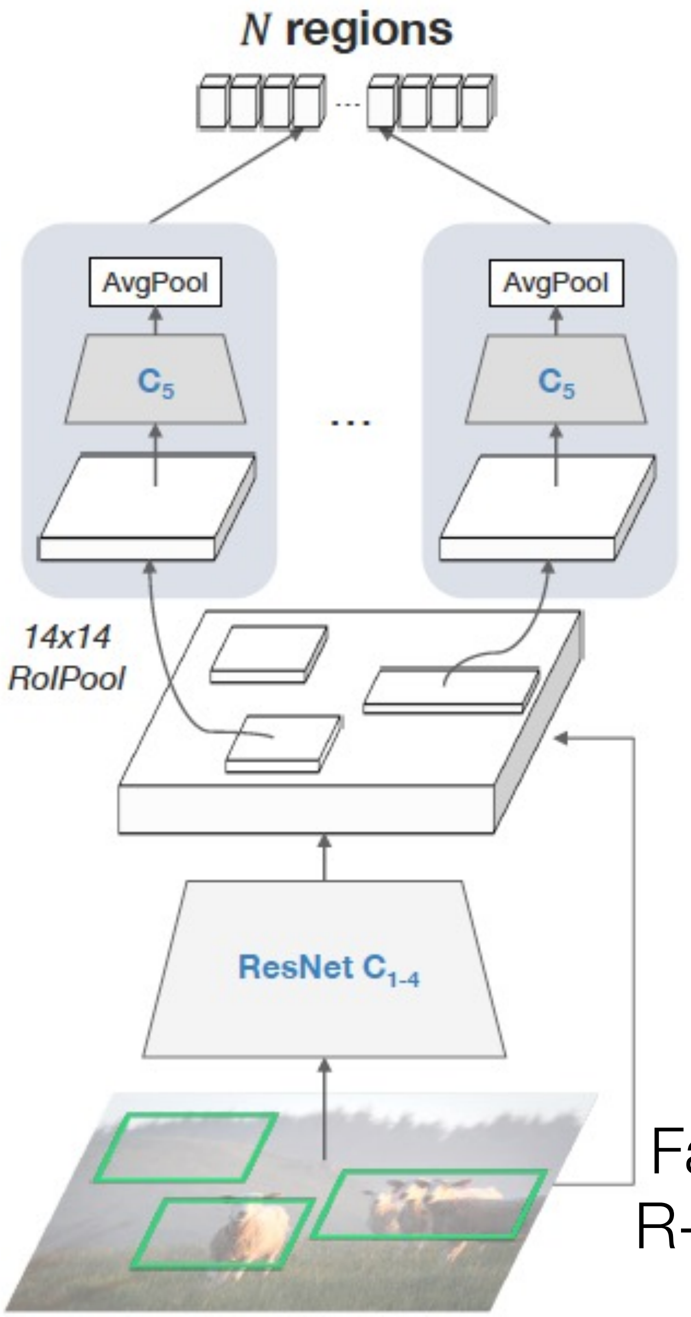
- Idea: representing images with **regions**
  - Use multiple, spatially-localized features to represent an image
  - “Bottom-up” because the regions are selected without top-down input from text and only from image pixels



[Karpathy & Fei-Fei, CVPR 2015] [Anderson et al, CVPR 2018]

Bottom-Up Attention

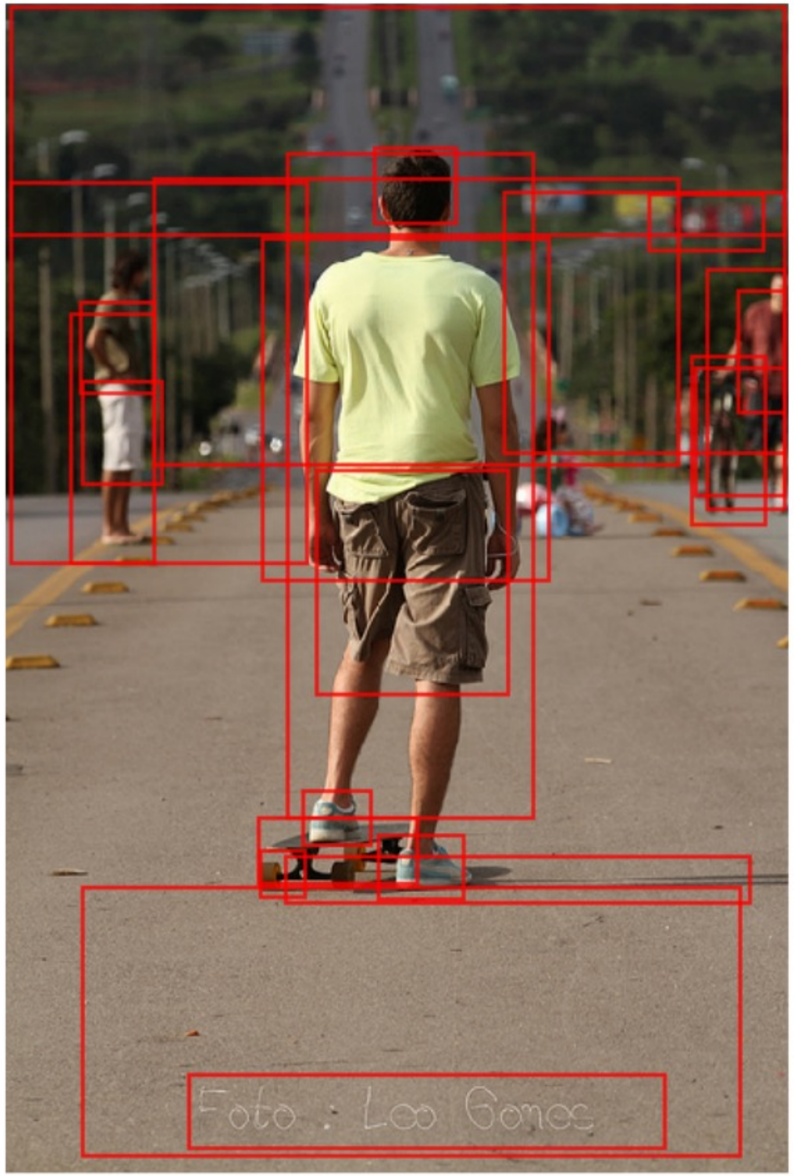
- Implementation
  - Pre-train a Faster R-CNN detector on Visual Genome
    - Tasks: object detection and attribute classification
    - Backbone: ResNet
  - Given an image:
    1. (Region Selection) top-scored regions are selected from Region Proposal Network
    2. (Region Feature Computation) average pooled features are extracted per-region after RoIPool and conv layers
- Has dominated leaderboards since its proposal, still used today



Faster R-CNN

Bottom-Up Attention

- Why is it successful?
  - Intuitive advantages over grid features:
    - Localize individual objects better
    - Capture coarse and fine details
    - Can model object interactions explicitly
- However, multiple factors have changed in comparison to prior work:
  - Pre-training task: classification vs. detection
  - Pre-training dataset: ImageNet vs. VG
  - …
- We conducted a controlled study to understand better



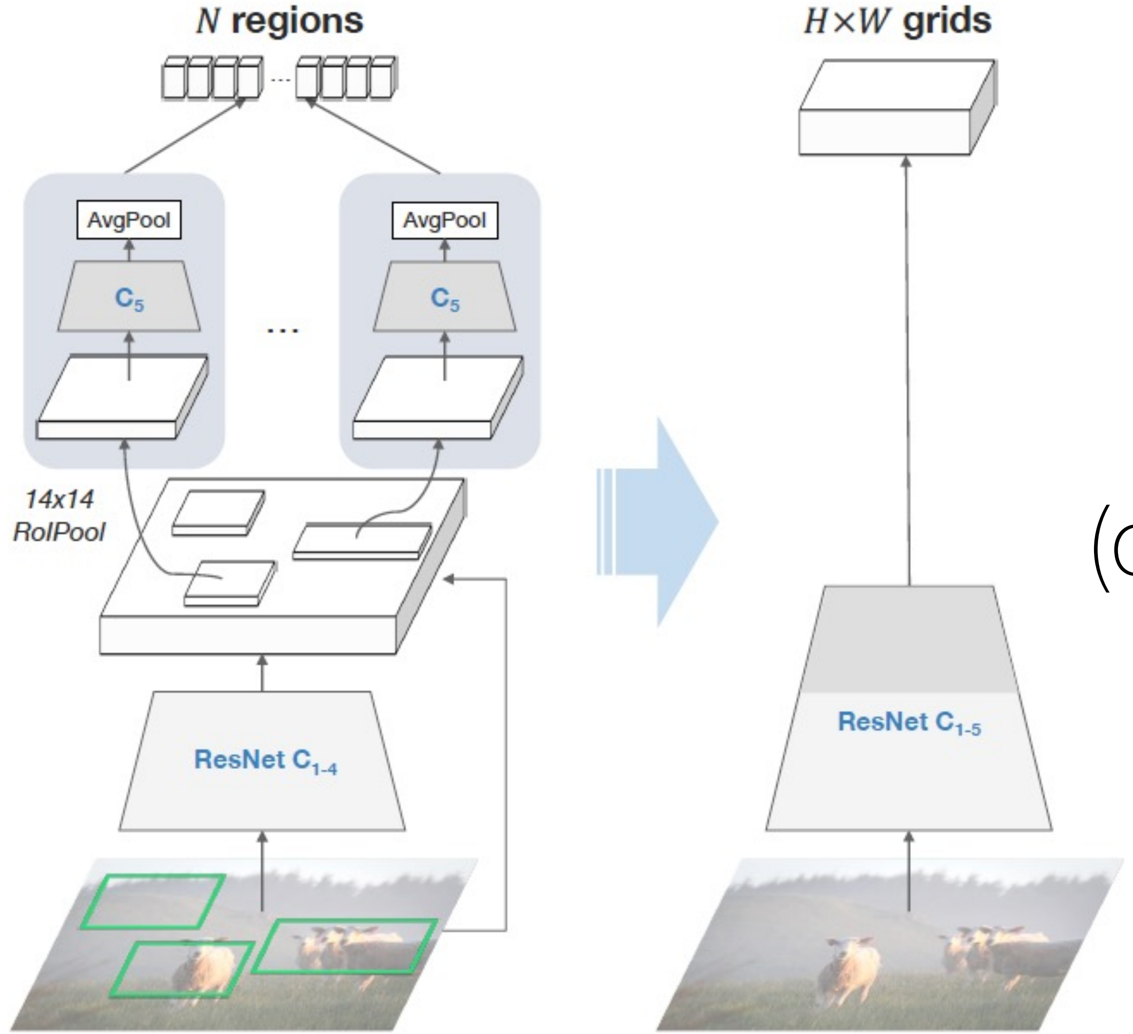
Basic Setups

- Fix pre-training task & dataset
  - Visual Genome Object + Attribute detection
- Fix backbone & input size
  - ResNet-50, 600x1000
- Fix evaluation task & metric
  - VQA, VQA score (accuracy)
- Fix VQA model
  - Pythia (2018 challenge winner)

https://arxiv.org/abs/1807.09956

Study 1: Grid Features from the **Same** Layer

R-CNN
(detector)



CNN
(classifier)

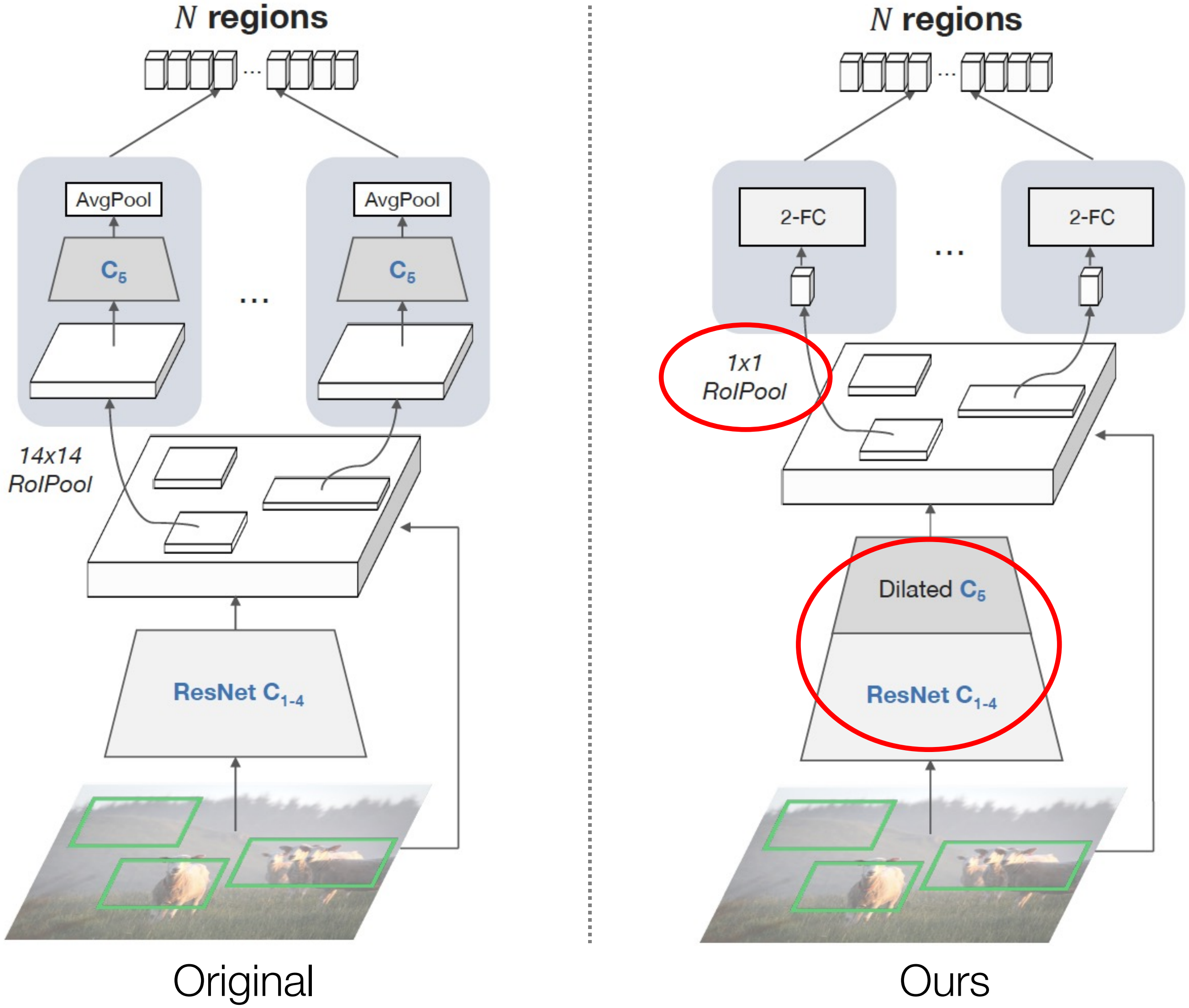
Study 1: Grid Features from the **Same** Layer

Setting	pre-train		VQA score
	task	dataset	
regions	Detection	VG	64.3
grids	Detection	VG	63.6
(prior) grids	Classification	ImageNet	60.8

- Resulting grid features almost work out-of-the-box
- Much closer to the bottom-up region features than previous grid ones pre-trained on ImageNet

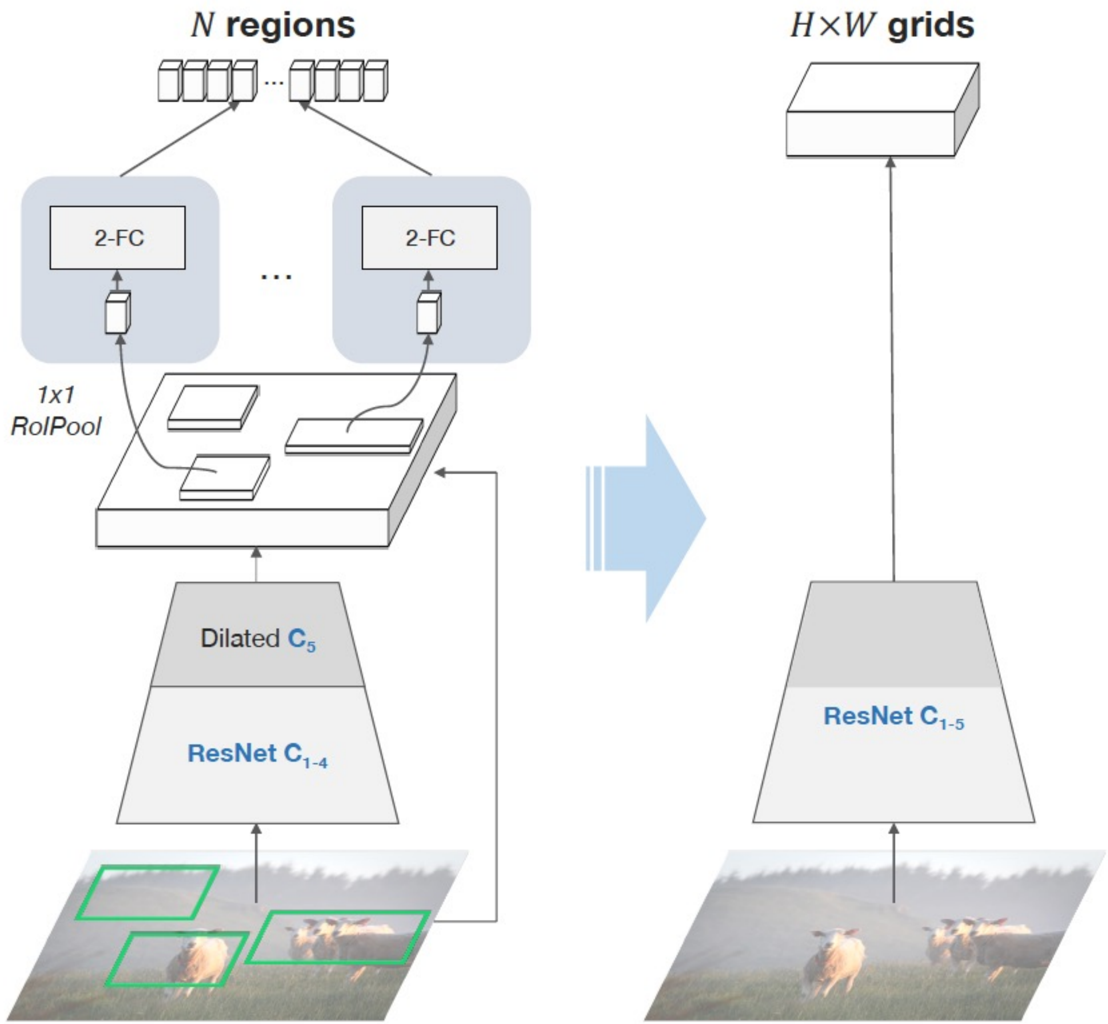
Study 2: Improve Pre-training for Grids

- Pre-trained detector
  - R-CNN, R stands for regions
  - Likely highly optimized for region-level tasks
- Our modification
  - Break the spatial representation of regions in R-CNN
  - 14x14 RoIPool → 1x1
  - Dilated C5 to apply fc layers per-region



Study 2: Improve Pre-training for Grids

R-CNN
(detector)



CNN
(classifier)

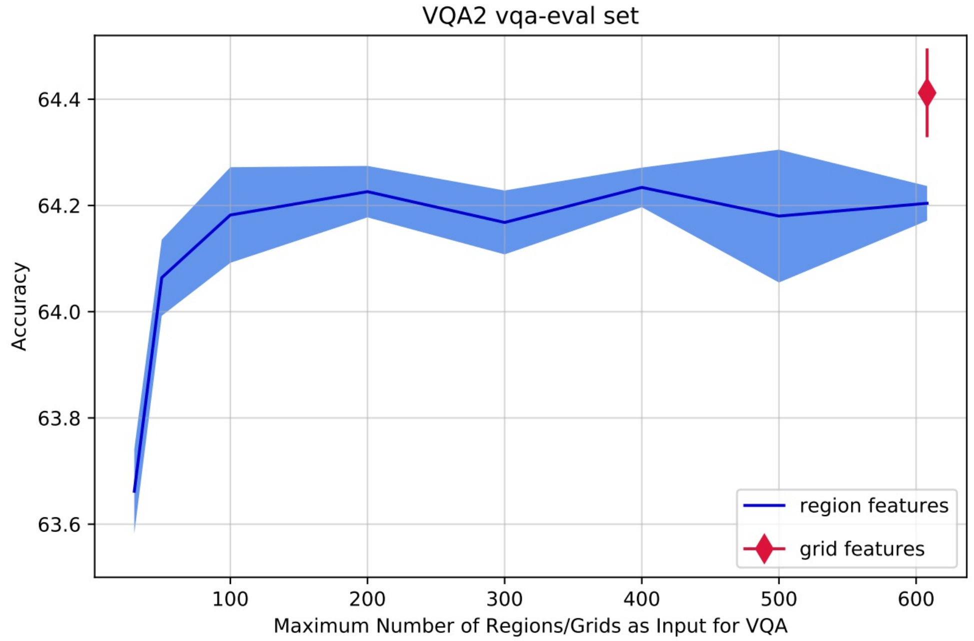
Study 2: Improve Pre-training for Grids

Setting	detector pre-train		VQA score
	RoIPool	AP	
regions	14x14	4.1	<u>64.3</u>
grids			63.6
regions	1x1	2.9	63.9
grids			<u>64.4</u>

- 1x1 RoIPool hurts detection and region features but helps grids
- <u>Grid features can work as well as regions for VQA</u>

Study 3: Number of Visual Features

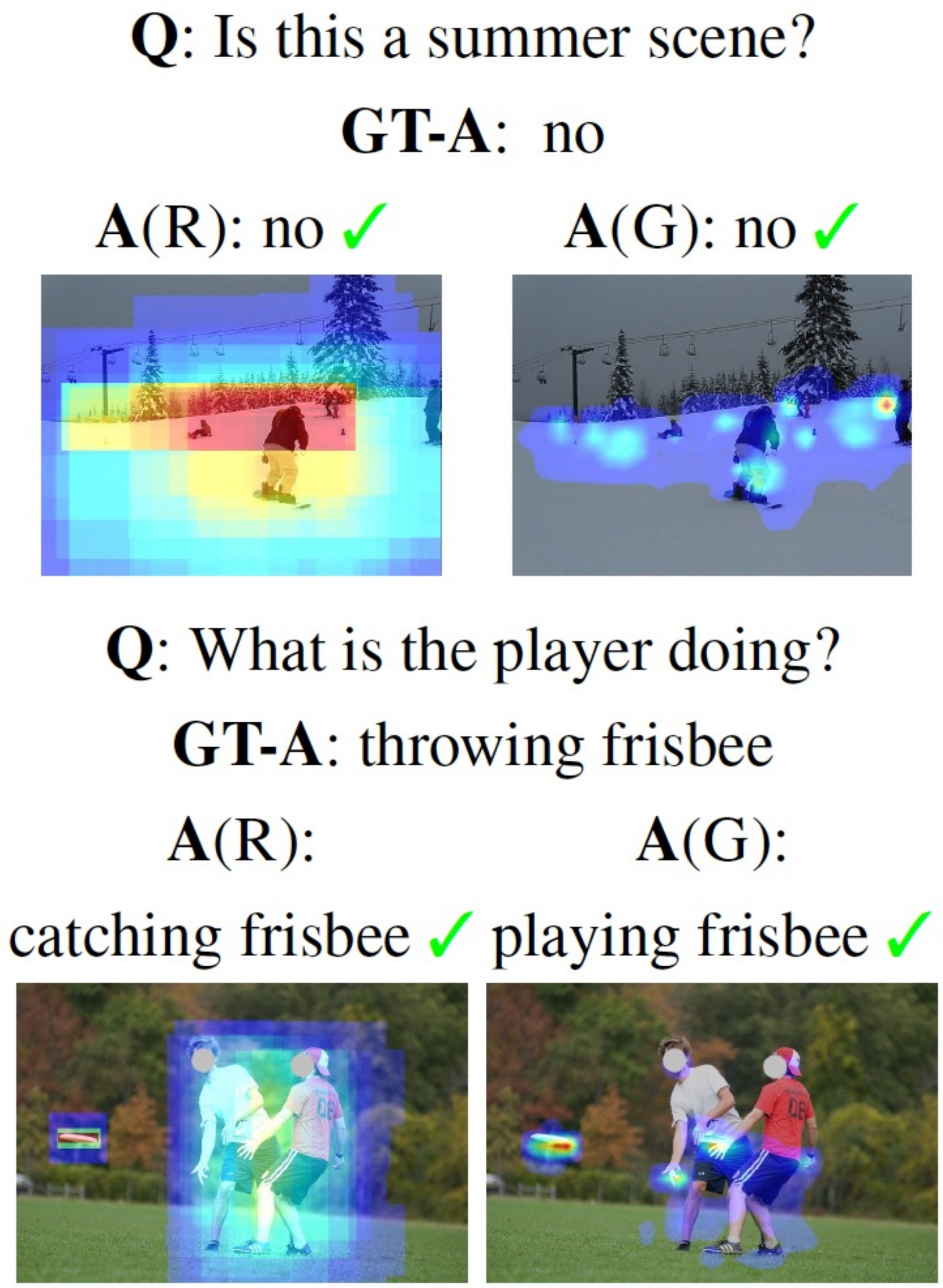
- Motivation
  - N Regions are sparsely sampled; and grids are densely sampled
  - So N is usually smaller than $\mathrm{H} \times \mathrm{W}$, which can benefit grid features
- Observation
  - Region features benefit from a larger N – recall is important
  - Even with bigger N ($\approx \mathrm{H} \times \mathrm{W}$), regions & grids are still at par



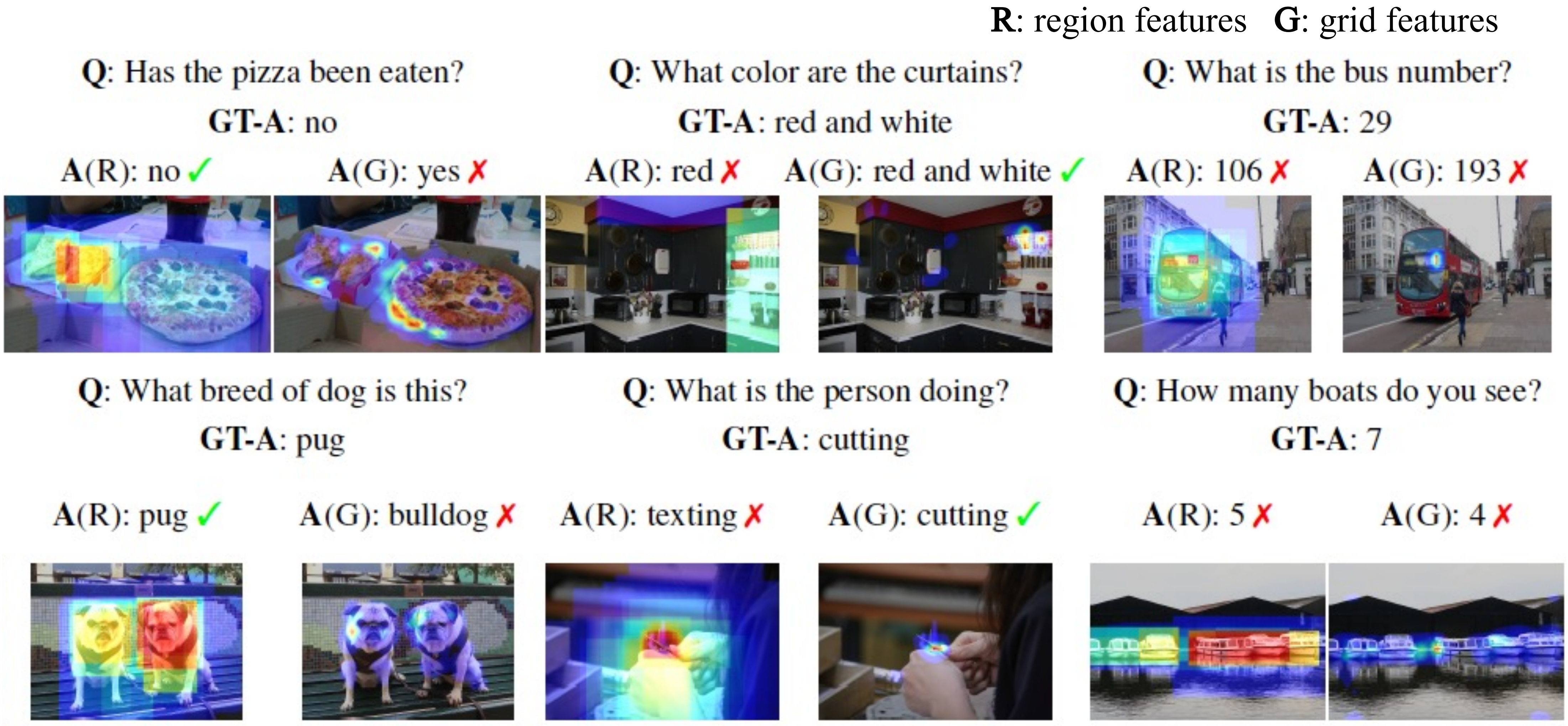
Attention Visualizations

R: region features

G: grid features



Attention Visualizations, *cont.*



“Grids ~ Regions” Holds Across:

- Different backbones
  - ResNet-50, ResNeXt-101
- Different VQA models
  - Pythia (2018 challenge winner), MCAN (2019 winner)
- Different VQA tasks
  - VQA 2.0, VizWiz dataset (focusing on blind users)
- Different other tasks
  - COCO image captioning

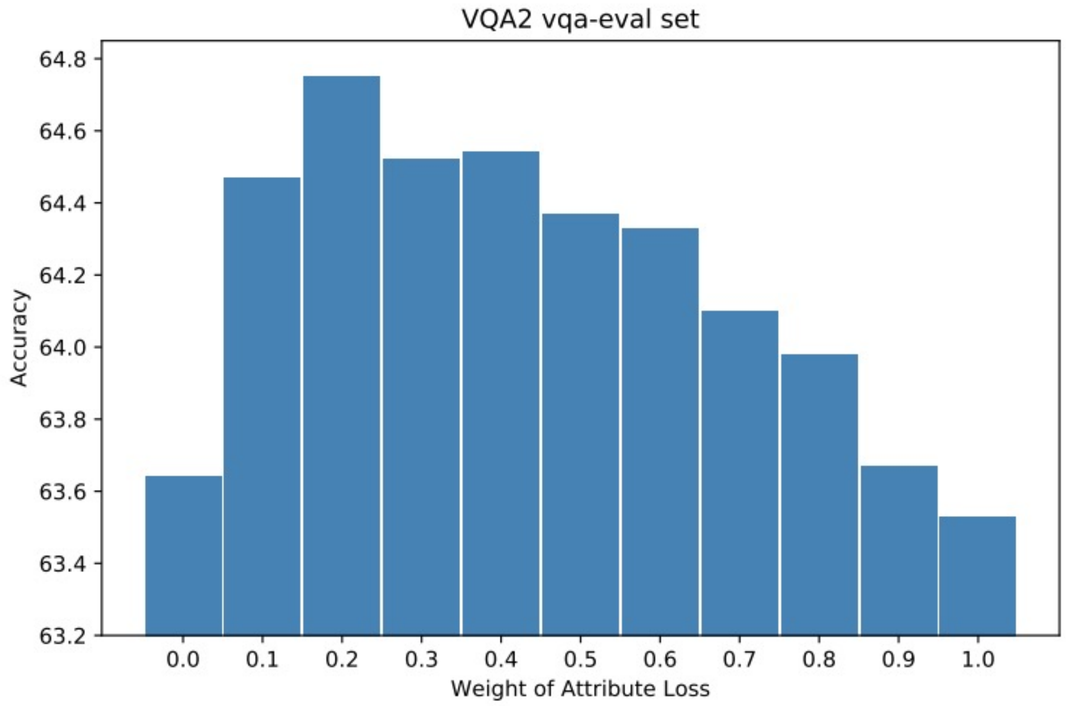
Study 4: Why **Our** Grid Features Work?

1. Pre-training task
   - VG object + attribute detection offers more powerful features

2. Input image size
   - Classification default: 448x448
   - Detection default: 600x1000
   - Grids can get even better with higher resolutions

pre-train	input size	VQA score
ImageNet	448x448	<u>60.8</u>
	600x1000	61.5
	800x1333	61.5
VG	448x448	63.2
	600x1000	<u>64.4</u>
	800x1333	64.6

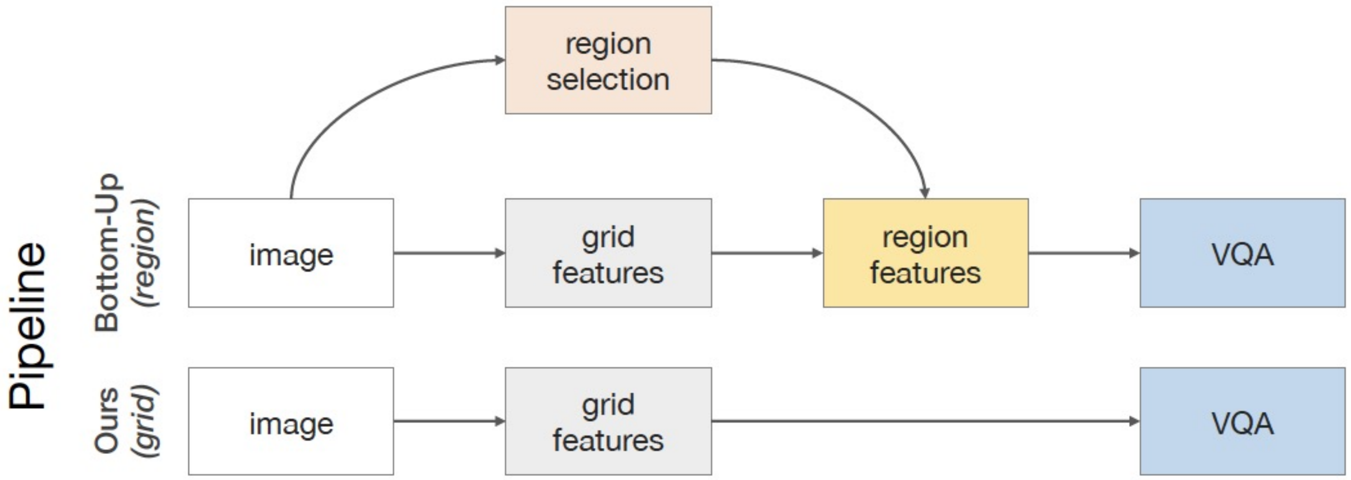
Study 5: How Important is Attributes?



Q: What color is the hydrant? **A**: red

- Intuitively useful for questions concerning attributes

Benefits of Grid Features: **Simplify Pipeline**

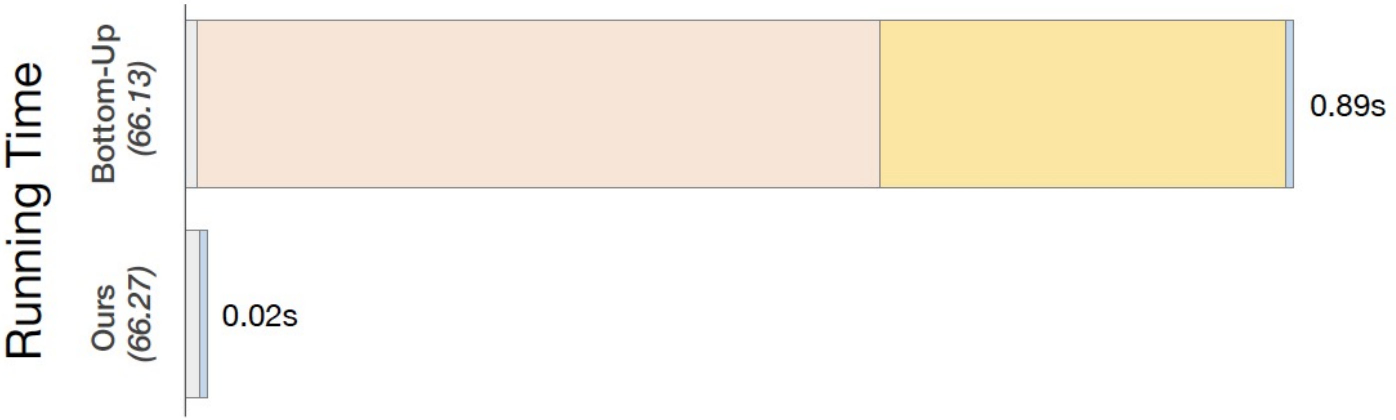


- Without region-related computations, grid features are obtained by single forward-pass of a ConvNet
- This can make end-to-end optimization of visual representations easier for V and L

Benefits of Grid Features: **Speed-up**

Duy-Kien Nguyen

(ResNet-50)



- Without region-related computations, grid features offer significant speed-ups (10 to 40+ times)
- Light-weight: visual features can be extracted online, allowing explorations of early-fusion models between V and L

MoVie: Revisiting Modulated Convolutions for Visual Counting and Beyond, ICLR 2021

Grid Features can **Work** Really Well

method	*features*	VQA Score (Single Model)	
		test-dev	test-std
BUTD (2017 winner)	*Region*	65.32	65.67
Pythia (2018 winner)		70.01	70.24
MCAN (2019 winner)		72.80	-
Ours (2020 winner)	*Grid*	73.98	74.16

VQA 2020 Challenge **Winner**: Our Improved <u>Grid</u> Features

Grid Features can **Work** Really Well, *cont.*

method	*features*	VQA Score (Single Model)	
		test-dev	test-std
BUTD (2017 winner)	*Region*	65.32	65.67
Pythia (2018 winner)		70.01	70.24
MCAN (2019 winner)		72.80	-
Ours (2020 winner)	*Grid*	73.98	74.16
AliceMind (2021 winner)	*Region + Grid*	77.71	-

VQA 2020 Challenge **Winner**: Our Improved <u>Grid</u> Features

VQA 2021 Challenge **Winner**: Region + <u>Grid</u> Features

ArXiv: https://arxiv.org/abs/2011.10566, CVPR 2021
Code: https://github.com/facebookresearch/simsiam

Approach: Exploring **Sim**ple **Siam**ese Representation Learning

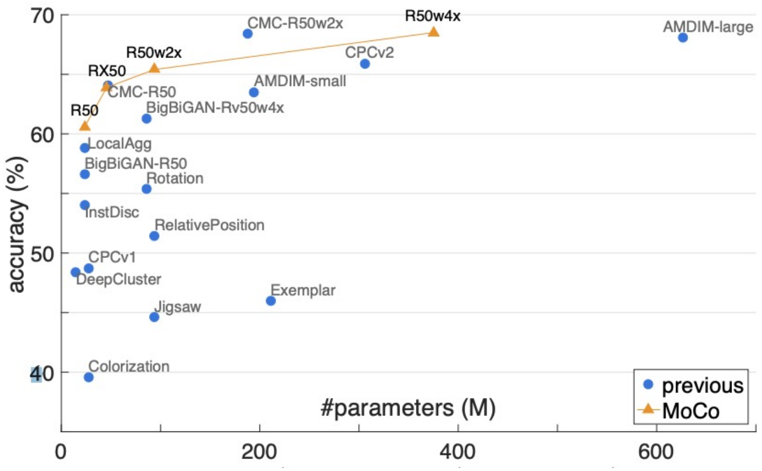

Xinlei Chen


Kaiming He

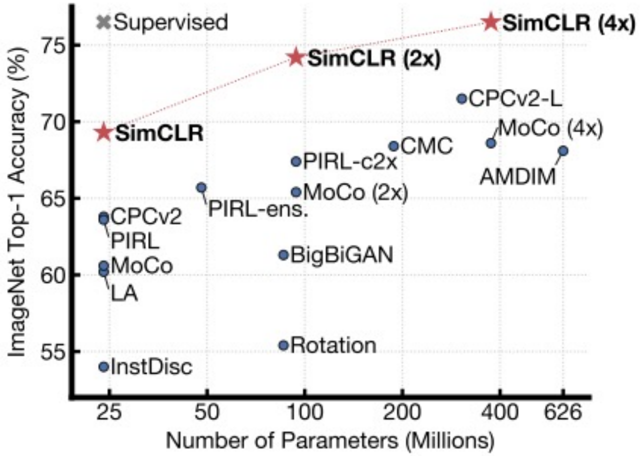
facebook
Artificial Intelligence Research

Many Exciting Frameworks

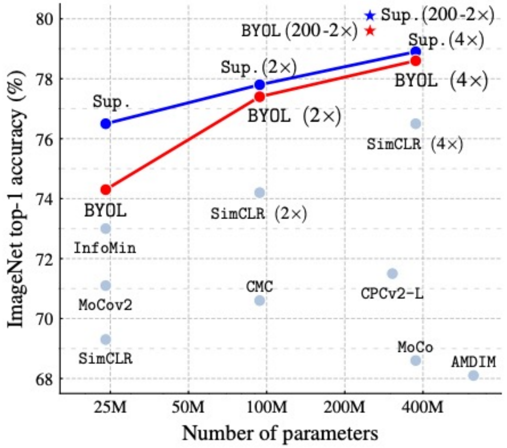
MoCo, 2019.11



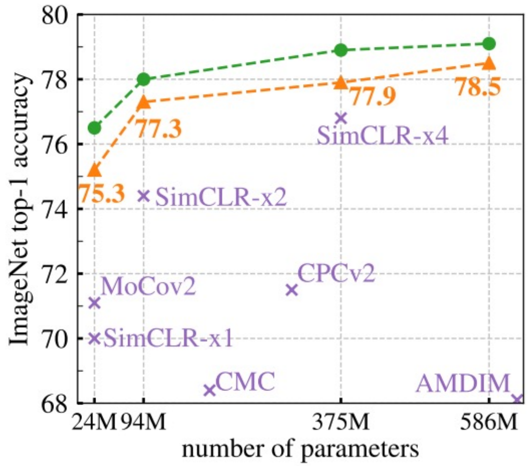
SimCLR, 2020.2



BYOL, 2020.6



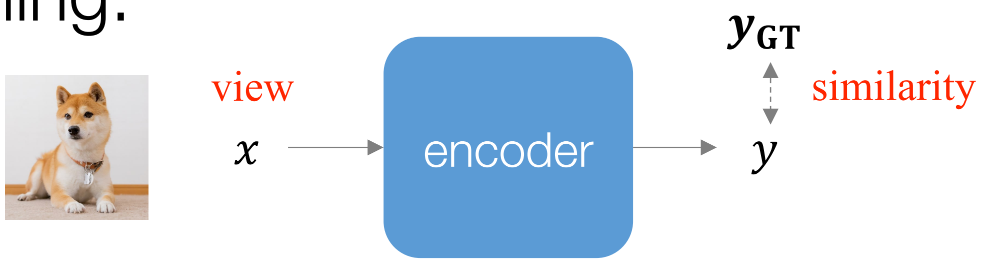
SwAV, 2020.6



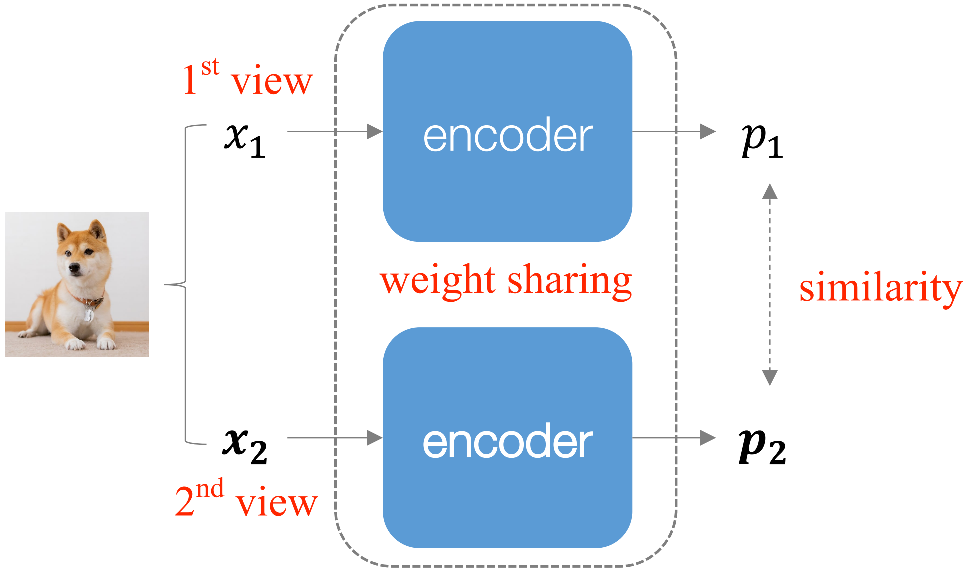
[He et al, CVPR 2020] [Chen et al, ICML 2020] [Grill et al, NeurIPS 2020] [Caron et al, NeurIPS 2020]

Common: Siamese/twin/dual Networks

- Supervised learning:

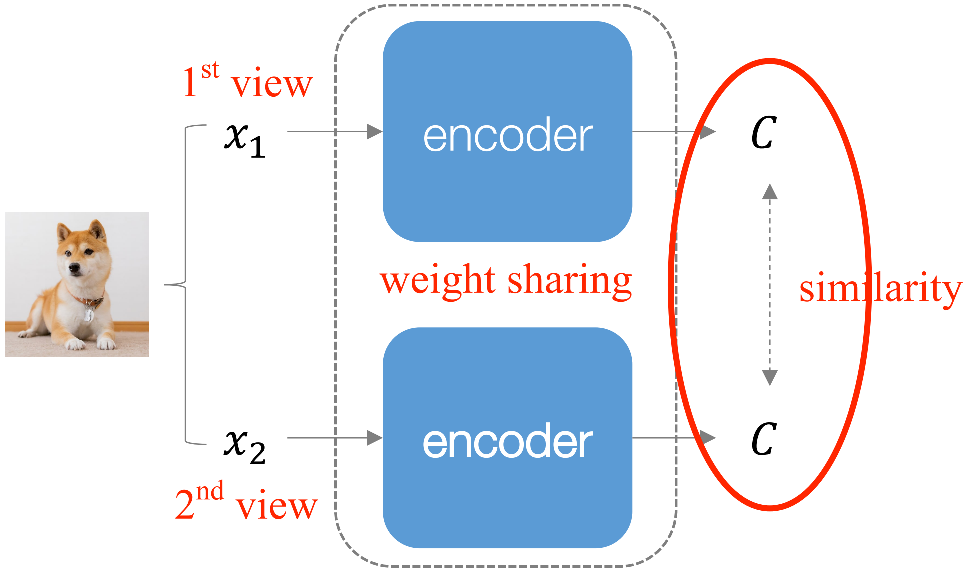


- *(a natural analogy in)* Un-/Self-supervised learning:



Well, Not Quite..

- Undesired *trivial* solution exist:
    - Predicting constant (C) for everything, representation collapses



- Countering strategies?

Contrastive Learning

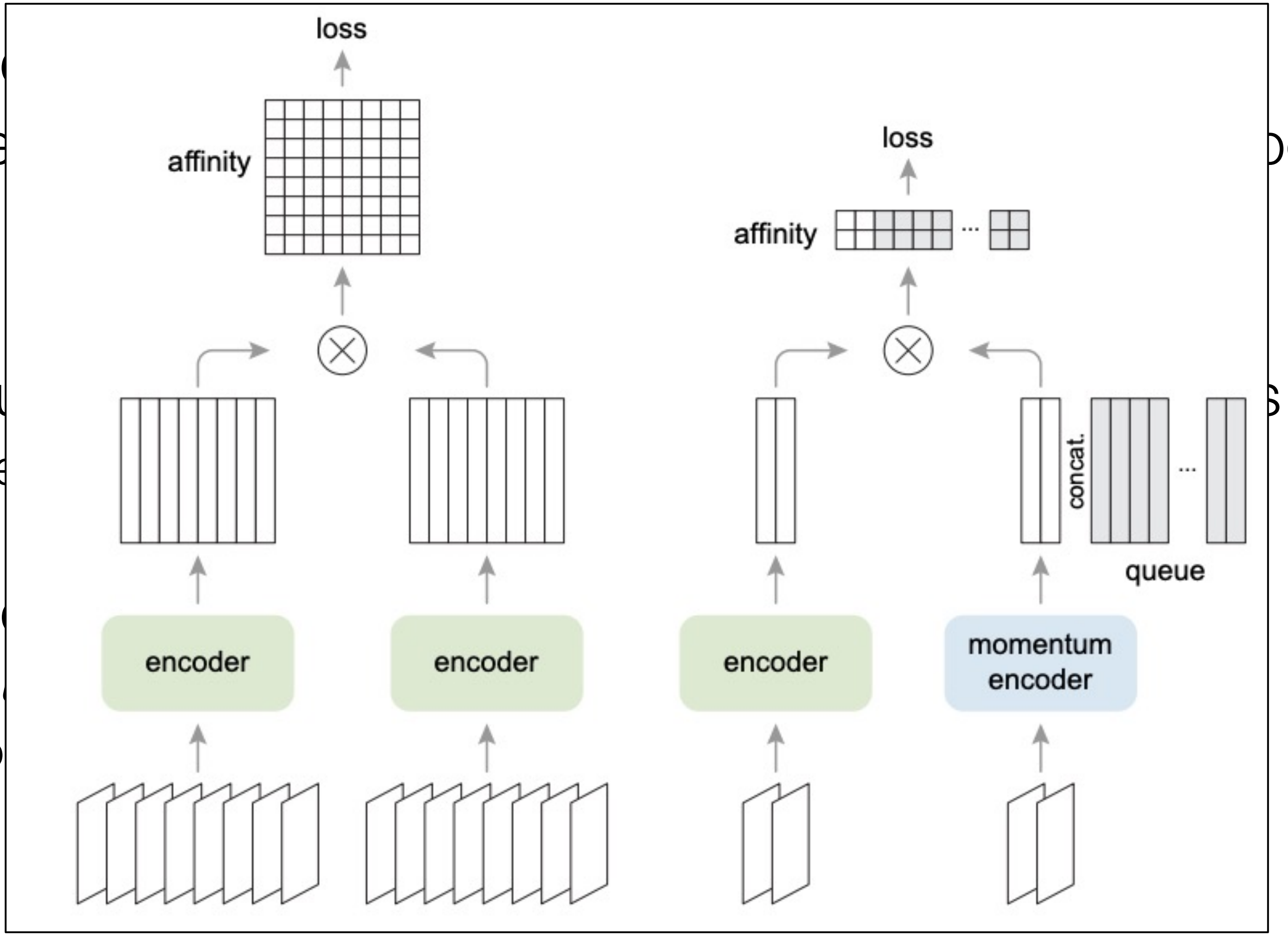
- Explicitly requires **dissimilarity** for views from different images
  - Still requires similarity for views from the same image
  - So, predicting constant is no longer optimal

- Popular loss function:
  - InfoNCE
    - $-\log \frac{\exp(p \cdot p' / \tau)}{\exp(p \cdot p' / \tau) + \sum_{n \in \mathcal{N}} \exp(p \cdot n / \tau)}$
    - $\mathcal{N}$ is the set of views from other images as negatives
    - τ is a temperature parameter

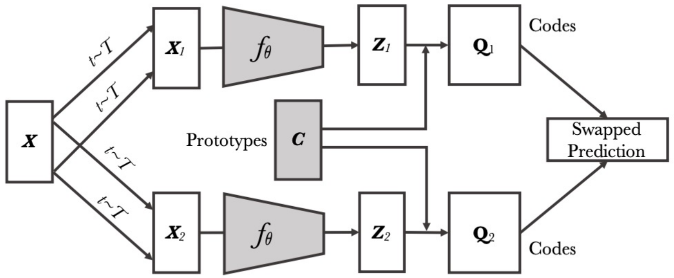
Contrastive Learning

- Drawback
  - Usually re… performance

- Solution in
  - SimCLR … within batch
    - Require
  - MoCo us
    - It *deco*
    - Additio

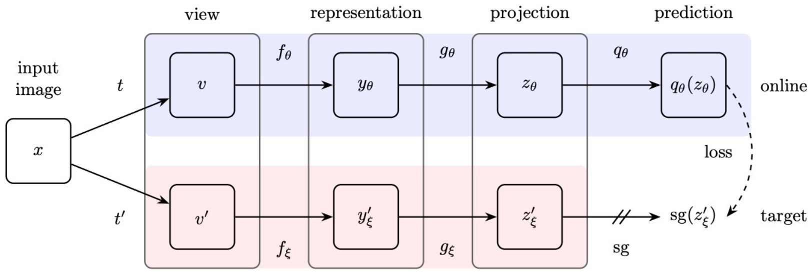


Other strategies



- Balanced online clustering (SwAV)
  - A cluster-center based output representation, p is used to pick center
  - Key: making sure that cluster sizes are balanced (Sinkhorn-Knopp)
    - Constant solution is less likely because otherwise all points are assigned to a singular cluster

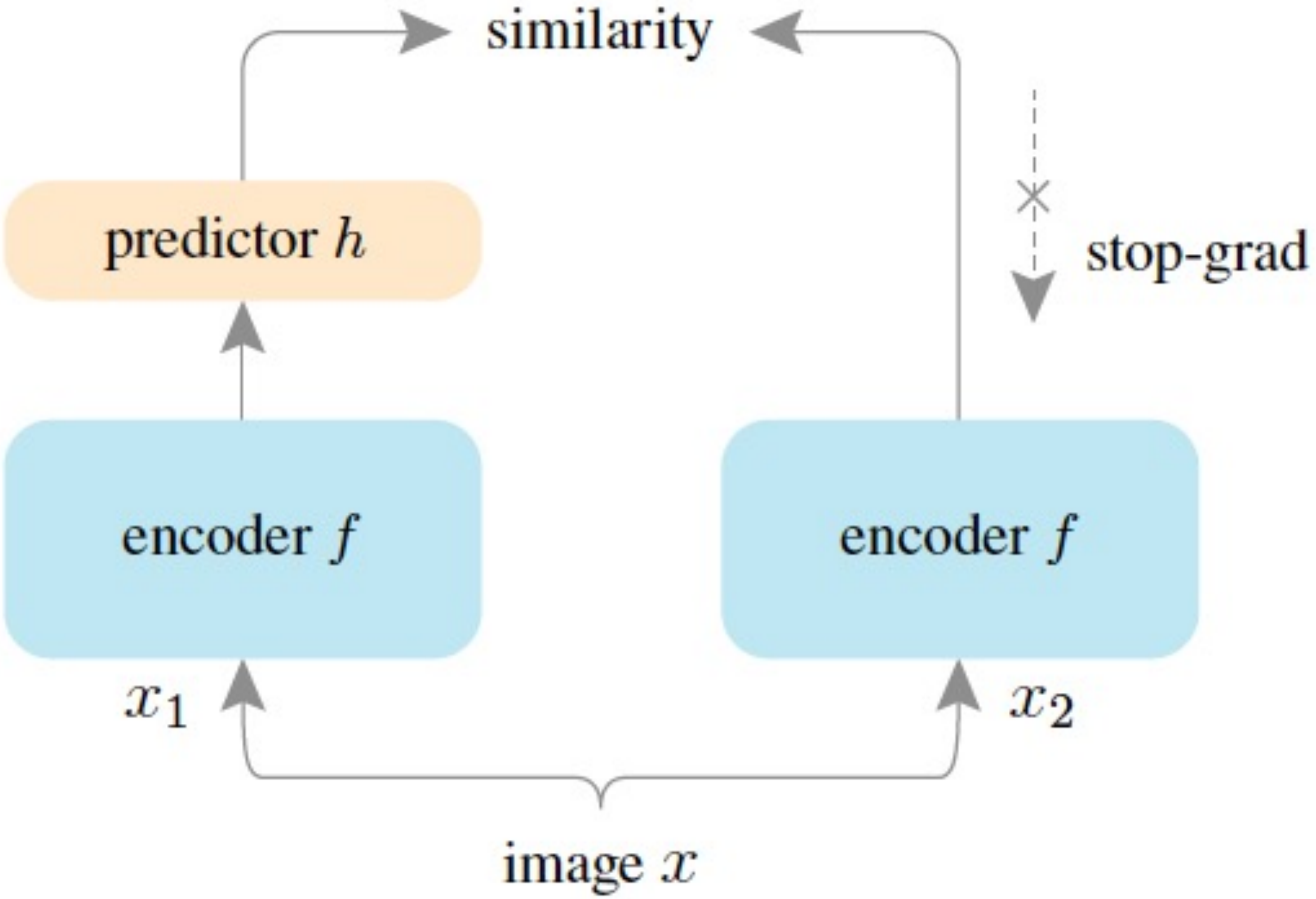
- BYOL
  - Introduces an additional MLP (*predictor*), and uses *momentum encoder*
    - Momentum encoder
      - Exponential moving average (EMA) of base encoder weights
      - So, weights are *not* updated by gradients
      - But need to maintain *two copies* of weights



All These are Rich & Fancy..

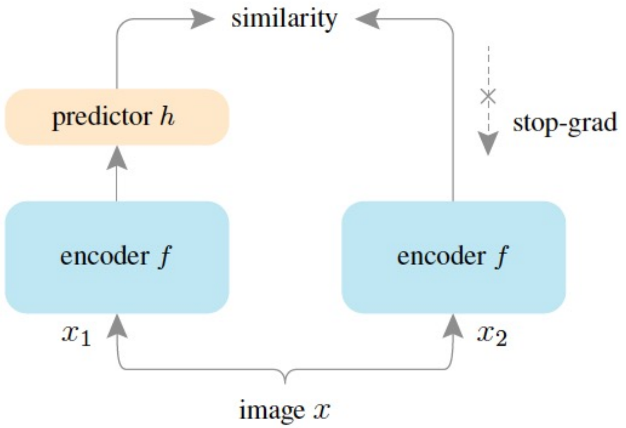
But can a Simple Siamese Network just Work?

Yes, SimSiam!



similarity
predictor h
stop-grad
encoder f
encoder f
x_1
x_2
image x

PyTorch-like Code for SimSiam



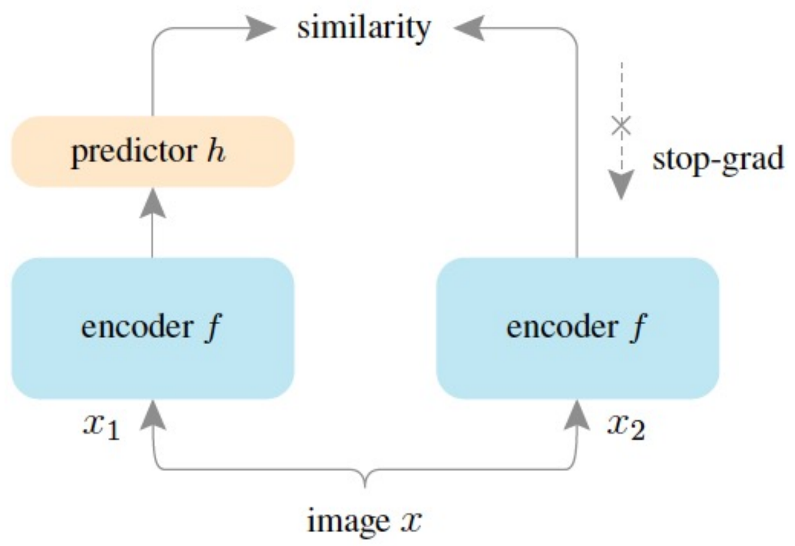
Algorithm 1 SimSiam Pseudocode, PyTorch-like

```
# f: backbone + projection mlp
# h: prediction mlp

for x in loader: # load a minibatch x with n samples
    x1, x2 = aug(x), aug(x) # random augmentation
    z1, z2 = f(x1), f(x2) # projections, n-by-d
    p1, p2 = h(z1), h(z2) # predictions, n-by-d

    L = D(p1, z2)/2 + D(p2, z1)/2 # loss

    L.backward() # back-propagate
    update(f, h) # SGD update

def D(p, z): # negative cosine similarity
    z = z.detach() # stop gradient

    p = normalize(p, dim=1) # l2-normalize
    z = normalize(z, dim=1) # l2-normalize
    return -(p*z).sum(dim=1).mean()
```

- Notes:
  - *Symmetrized* loss
  - l_2 normalized cosine similarity by default
  - Gradient is only back propagated through *predictor*
    - Stop-grad on other

SimSiam Simplifies Those Frameworks

- SimCLR *w/o* negatives
- SwAV *w/o* online clustering
- BYOL *w/o* momentum encoder
- MoCo *w/o* negatives or momentum encoder



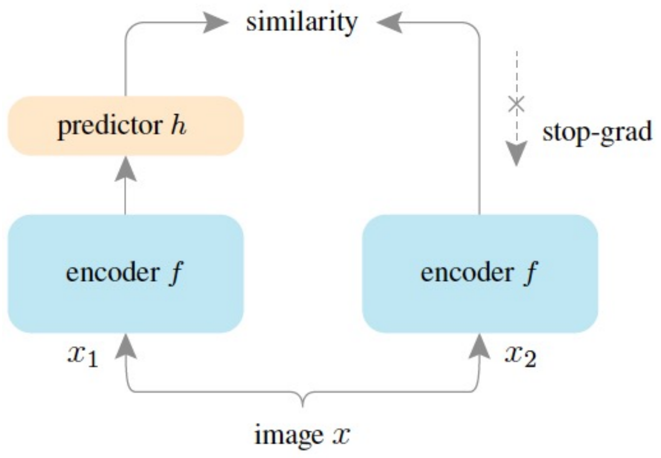
Basic Settings of Experiments

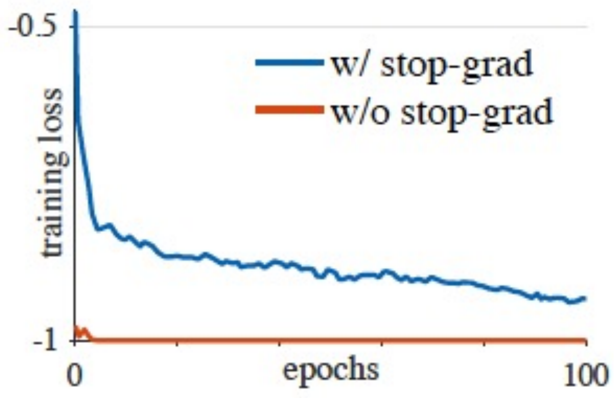
- Encoder: ResNet-50 + 3-layer projector MLP
  - Projector MLP: from SimCLR
  - Sync BatchNorm: from SimCLR/BYOL
- Predictor MLP:
  - From BYOL
  - <u>Bottleneck</u> structure, with smaller hidden dimension than input/output
- Pre-training:
  - SGD + momentum optimizer: following MoCo, no large-batch optimizers (LARS)
  - 100-epoch pre-training
- Evaluation:
  - Linear 1000-way classifier of <u>frozen</u> ResNet pool-5 features on ImageNet train/val

Stop-Grad is Crucial for SimSiam

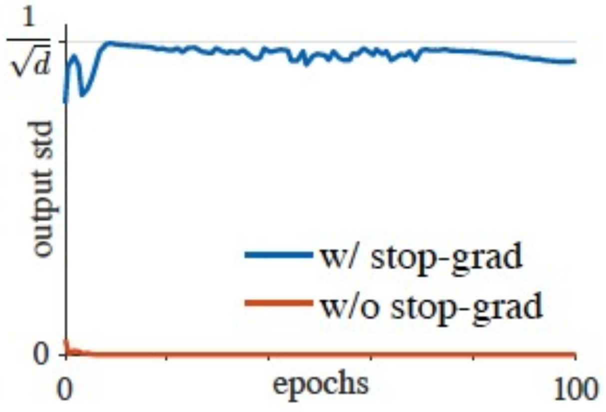
- Without it, representation collapses
  - *Implicit* for momentum encoder

setting	top-1
w/ stop-grad	67.7±0.1
w/o stop-grad	0.1

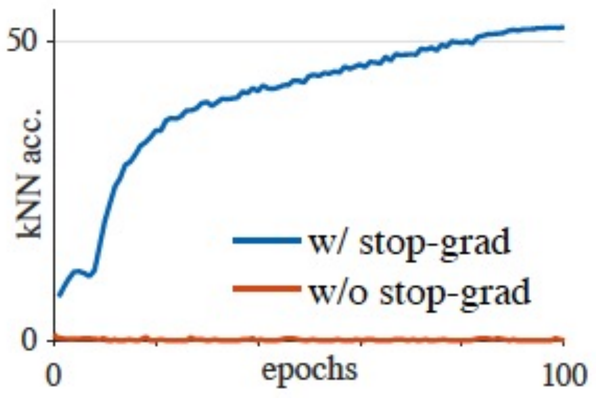




loss curve



monitor 1: std of p



monitor 2: KNN classifier

Predictor is Important

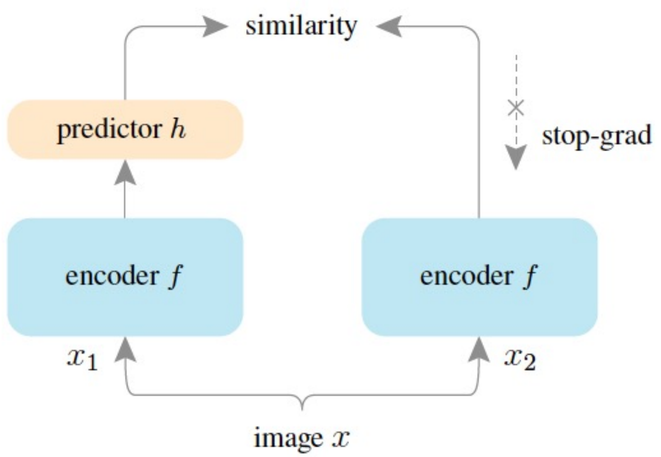
- Tried different settings:

setting	top-1
previous default	67.7
w/o predictor	0.1
random predictor	1.5
not decay predictor lr	68.1

effectively w/o stop-grad: symmetrized loss

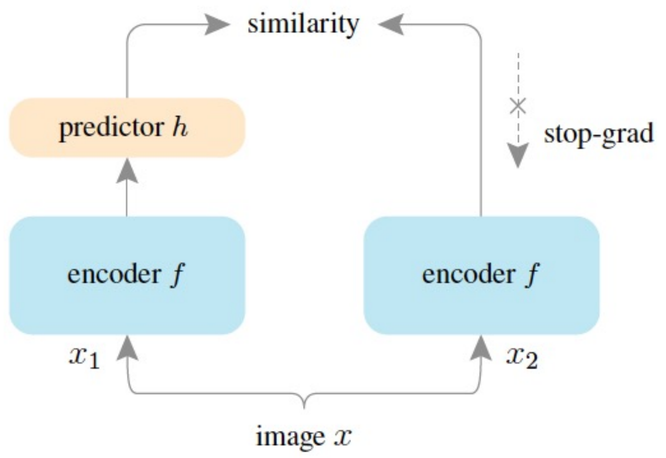
does not converge

default for later explorations



- Not crucial: predictor **can** be removed without collapsing (later)

Robustness: Losses



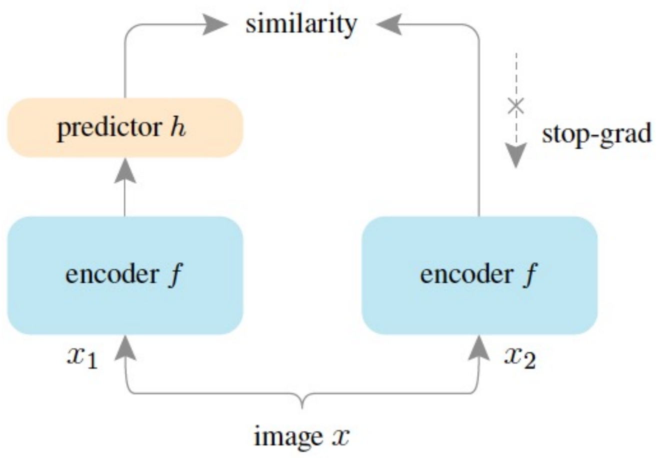
- Cosine vs. soft-max cross-entropy
  - Can work out-of-box
  - Relates to SwAV: a similar loss there

setting	top-1
cosine	68.1
cross-entropy	63.2

- Symmetrized vs. not
  - Symmetrized is better
    - Likely because it trains "longer"
  - SimSiam has advantage over BYOL:
    - Does not need to forward again on the momentum encoder

setting	top-1
symmetrized	68.1
asymmetric	64.8
asymmetric, 2x	67.3

Batch Normalization

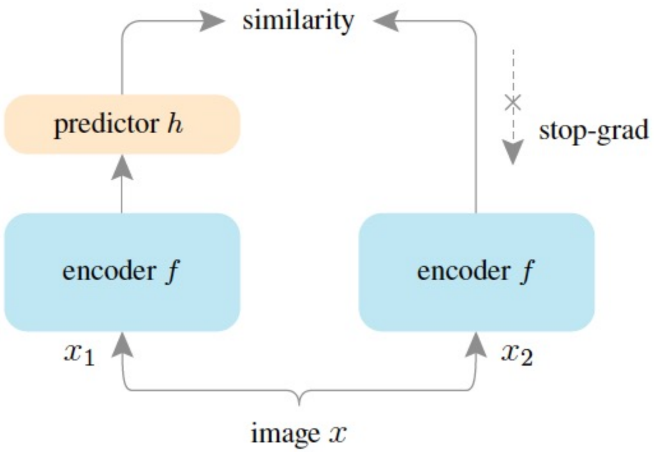


- Batch normalization is required for SimSiam
  - SyncBN on each view *separately*
  - Weight decay applied to BN parameters (different from BYOL, SimCLR)

- Analysis of BN on MLPs

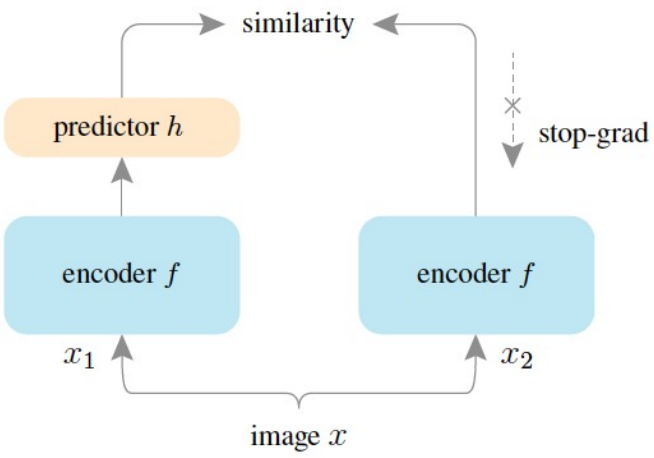
case	proj. hidden	proj. output	pred. hidden	pred. output	top-1
none					34.6
hidden-only	✅		✅		67.4
default	✅	✅	✅		68.1
all	✅	✅	✅	✅	unstable

The Role of Stop-Grad



- Hypothesis
  - Provides a different trajectory that alternates between optimizing *two* sets of variables:
    - θ, network parameters
    - η, hidden representation for an image x, indexed by x
  - Objective function:
    - $L(\theta, \eta) = \mathbb{E}_{x,\mathcal{T}}\left[\left\|\mathcal{F}_\theta(\mathcal{T}(x)) - \eta_x\right\|_2^2\right]$
    - $\mathcal{T}$ stands for transformations, or augmentations to the input image

The Role of Stop-Grad



- Optimization for $L(\theta, \eta) = \mathbb{E}_{x,\mathcal{T}}\left[\left\|\mathcal{F}_\theta(\mathcal{T}(x)) - \eta_x\right\|_2^2\right]$
  - General alternative optimization:
    - Fix η, θ can be optimized with normal gradient decent
    - Fix θ, η can be updated with the expectation $\mathbb{E}_{\mathcal{T}}[\mathcal{F}_\theta(\mathcal{T}(x))]$ over transformations
  - SimSiam: One-step alternation:
    - θ is updated with one-step of gradient decent
    - η is updated with *one sample* of $\mathcal{T}$ only $\mathcal{F}_\theta(\mathcal{T}(x))$ → approximating $\mathbb{E}_{\mathcal{T}}[\mathcal{F}_\theta(\mathcal{T}(x))]$
- Hypothesis of the predictor
  - Fills the **gap** between single-sample and expectation over transformations

Proof-of-Concept 1

- Multi-step alternation:
  - Update θ multiple times (with SGD) before updating η again

	1-step	10-step	100-step	1-epoch
top-1	68.1	68.7	68.9	67.0

  - Has a “momentum encoder” effect that computes predictions with weights from previous iterations
  - Suggest alternating optimization is a valid formulation

Proof-of-Concept 2

- Remove predictor
  - Replace it with a *moving average* of previous $\mathcal{F}_\theta(\mathcal{T}(x))$
  - This is to approximate the expectation $\mathbb{E}_\mathcal{T}[\mathcal{F}_\theta(\mathcal{T}(x))]$

setting	top-1
default, w/ predictor	68.1
w/o predictor	0.1
w/o predictor, w/ moving average	55.0

- Supportive of the hypothesis that predictor is related to expectations

Comparisons to Others, ImageNet

method	batch size	negative pairs	momentum encoder	100-ep	200-ep	400-ep	800-ep
SimCLR	4096	✅		66.5	68.3	69.8	70.4
MoCo	256	✅	✅	67.4	69.9	71.0	72.2
BYOL	4096		✅	66.5	70.6	73.2	74.3
SwAV	4096			66.5	69.1	70.7	71.8
SimSiam	256			68.1	70.0	70.8	71.3

- SimSiam is batch size friendly, momentum encoder free, and competitive

Comparisons to Others, VOC Detection

Pre-train	AP50	AP75	AP
Supervised	74.4	42.4	42.7
SimCLR	75.9	46.8	50.1
MoCo	77.1	48.5	52.5
BYOL	77.1	47.0	49.9
SwAV	75.5	46.5	49.6
SimSiam (Optimal)	77.3	48.5	52.5

- All methods generally perform well, and *outperform* ImageNet supervised pre-training

Are Siamese Networks the Bare Minimum?

- A natural and effective tool to learn invariance
  - Invariance: Two views of the same concept should produce the same output
  - While invariance like “*translation*” can be baked into “*convolutions*” as **inductive biases**, more complex transformations (e.g., color, scale, rotation) are harder to design the counterparts
  - In such cases, Siamese network at least serves as a strong **data-driven** baseline
  - Further removal of inductive biases?
    - MoCo v3, ViT can also work (next!)

ArXiv: https://arxiv.org/abs/2104.02057, ICCV 2021
Code: https://github.com/facebookresearch/moco-v3

Architecture: An Empirical Study of Training Self-Supervised Vision Transformers



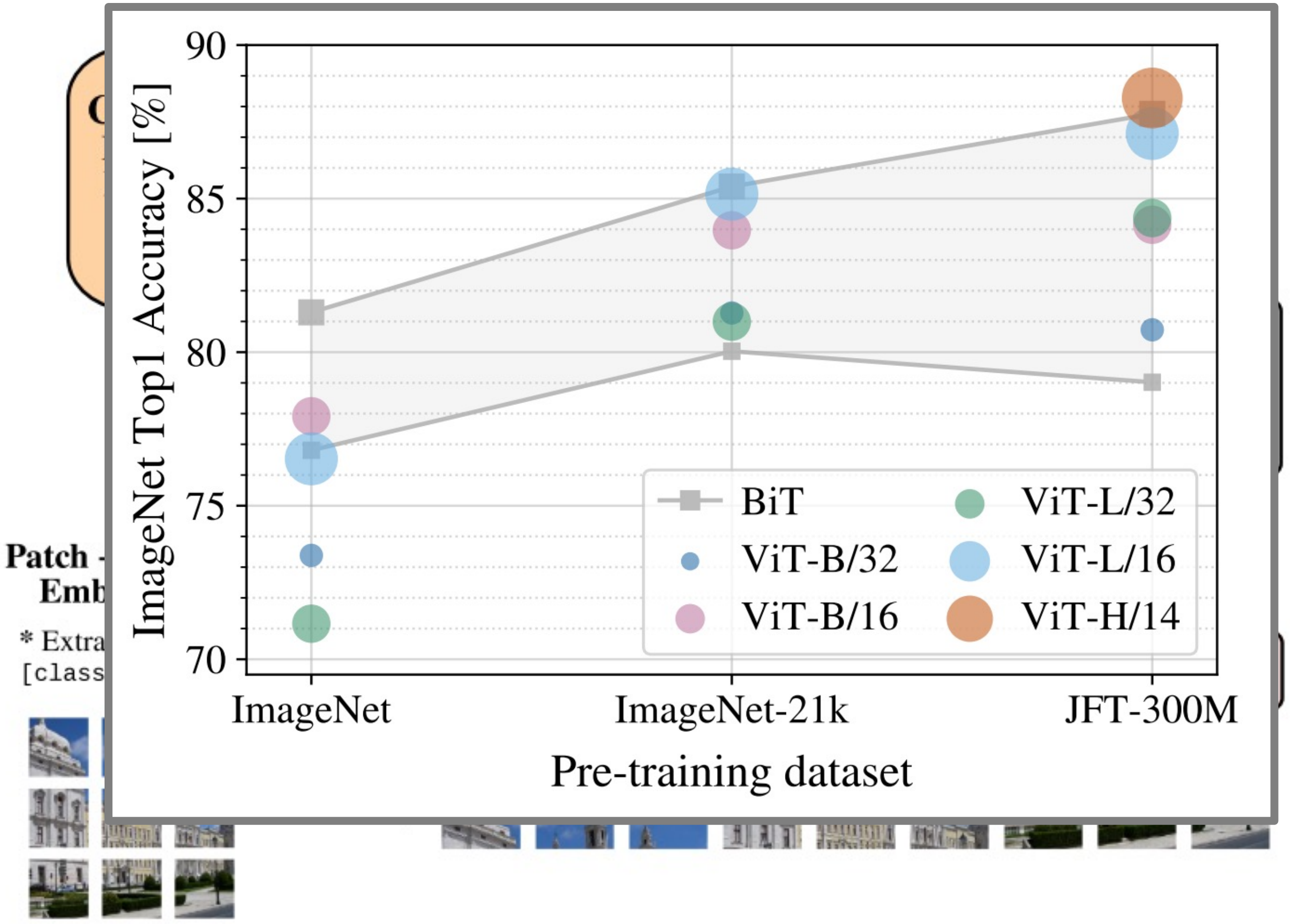
Xinlei Chen* Saining Xie* Kaiming He

facebook
Artificial Intelligence Research

Vision Transformer (ViT)

- Less inductive bias
  - Translation invariance
  - Flat architecture
    - Not pyramidal

- Scalable
  - w/ bigger model
  - w/ larger data



[Dosovitskiy et al, ICLR 2021]

Baseline: MoCo v3

Algorithm 1 MoCo v3: PyTorch-like Pseudocode

```
# f_q: encoder: backbone + proj mlp + pred mlp
# f_k: momentum encoder: backbone + proj mlp
# m: momentum coefficient
# tau: temperature

for x in loader: # load a minibatch x with N samples
    x1, x2 = aug(x), aug(x) # augmentation
    q1, q2 = f_q(x1), f_q(x2) # queries: [N, C] each
    k1, k2 = f_k(x1), f_k(x2) # keys: [N, C] each

    loss = ctr(q1, k2) + ctr(q2, k1) # symmetrized
    loss.backward()

    update(f_q) # optimizer update: f_q
    f_k = m*f_k + (1-m)*f_q # momentum update: f_k

# contrastive loss
def ctr(q, k):
    logits = mm(q, k.t()) # [N, N] pairs
    labels = range(N) # positives are in diagonal
    loss = CrossEntropyLoss(logits/tau, labels)
    return 2 * tau * loss
```

ResNet-50	top-1
MoCo v2	72.2
MoCo v3 (TPU)	73.8
MoCo v3 (GPU)	74.6

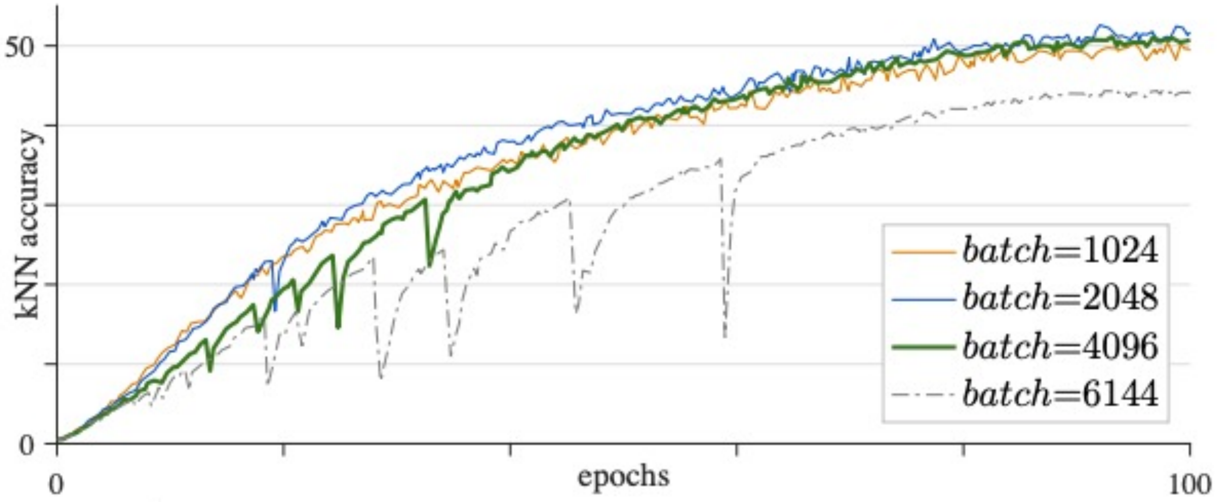
- **Mo**mentum encoder + **Co**ntrastive learning
- Removed:
    - Momentum queue
- Added:
    - Predictor
    - Other BYOL recipes
        - “BYOL w/ negatives”
        - BYOL top-1: 74.3

Study Setups

- Encoder: ViT-B/16
    - For 224x224 input, it leads to 196 patches, each with size 16x16
- Pre-training:
    - AdamW optimizer, typical for transformer architectures
    - 4096 batch size, 100-epoch
- Linear-eval:
    - 1000-way classifier on ImageNet 1K, on frozen ViT [class] features

Instability Issues

- Large batch size, large lr training is more challenging for ViT
  - "Dips": instability influences training
  - Indicating training is only "partially" successful, and "partially" failed
  - LAMB does not fix the issue

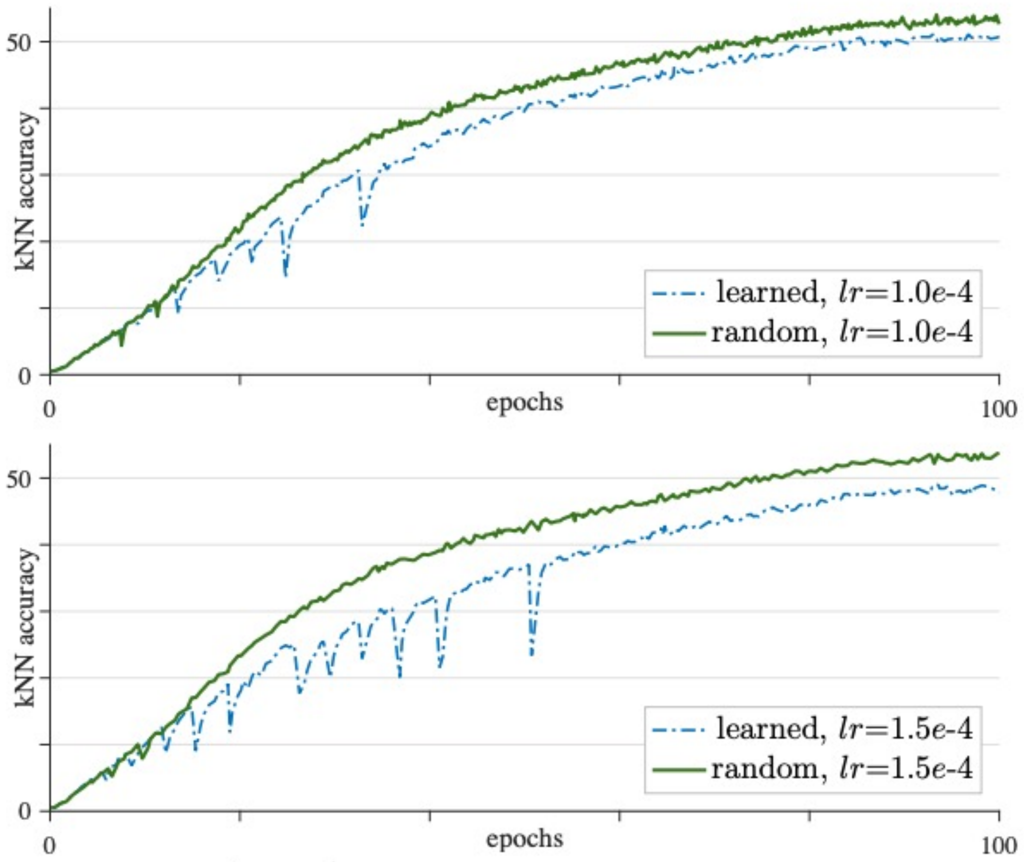


batch	1024	2048	4096	6144
linear acc.	71.5	72.6	72.2	69.7

[You et al, ICLR 2020]

Trick to Improve Instability

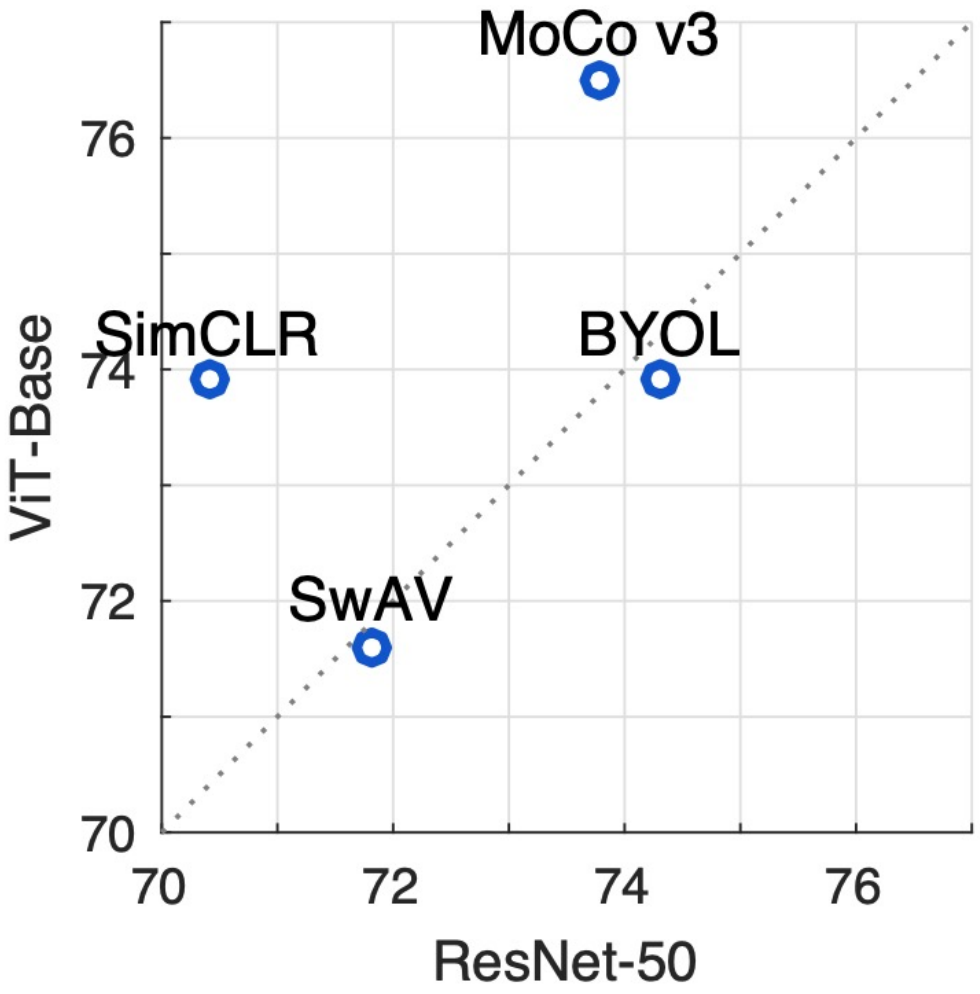
- Random patch projection
  - I.e., Stop-Grad right after patch projection
  - Narrows down solution space
- Generally helpful
  - Works with SimCLR, BYOL, etc.
- Not a fundamental solution
  - Sensitive to initialization



lr, $\times 10^{-4}$	0.5	1.0	1.5
learned patch proj.	70.4	72.2	71.7
random patch proj.	**70.8**	**72.8**	**73.4**

Siamese-based Frameworks

- Such frameworks generally transfer **well**
  - Yield reasonable results
- Behave differently
  - Contrastive learning-based methods have an edge on ViT



Quantitative Comparison of Frameworks

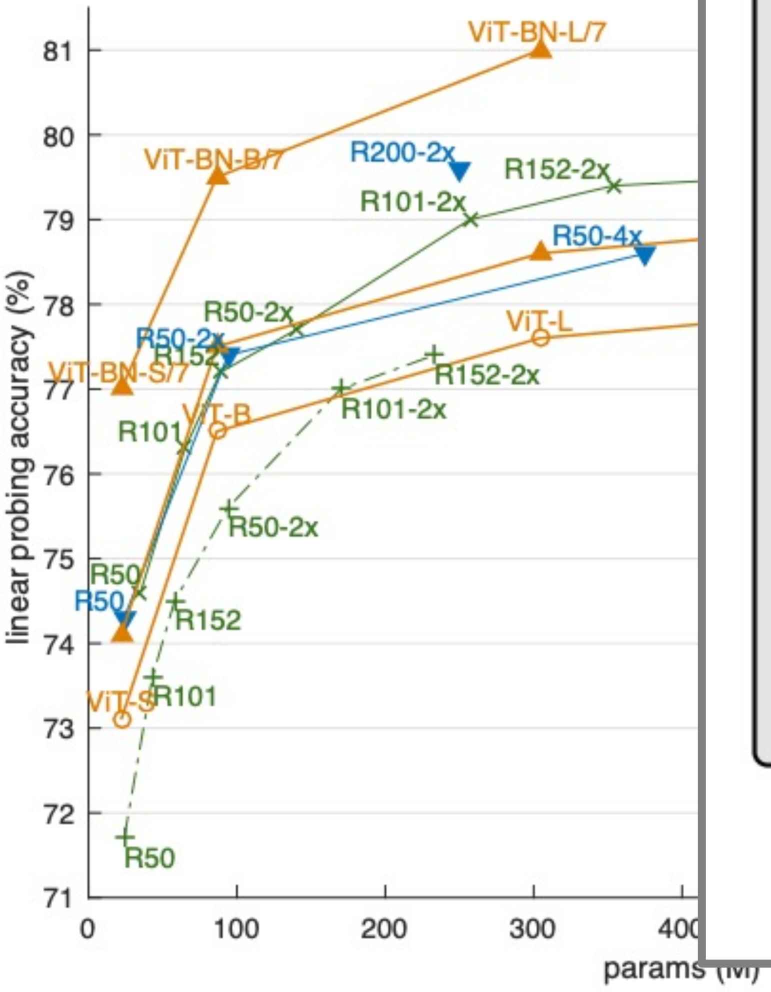
method	contrastive	momentum encoder	R-50	ViT-S	ViT-B
MoCo v3	✅	✅	73.8	72.5	76.5
SimCLR	✅		70.4	69.0	73.9
BYOL		✅	74.3	71.0	73.9
SwAV			71.8	67.1	71.6

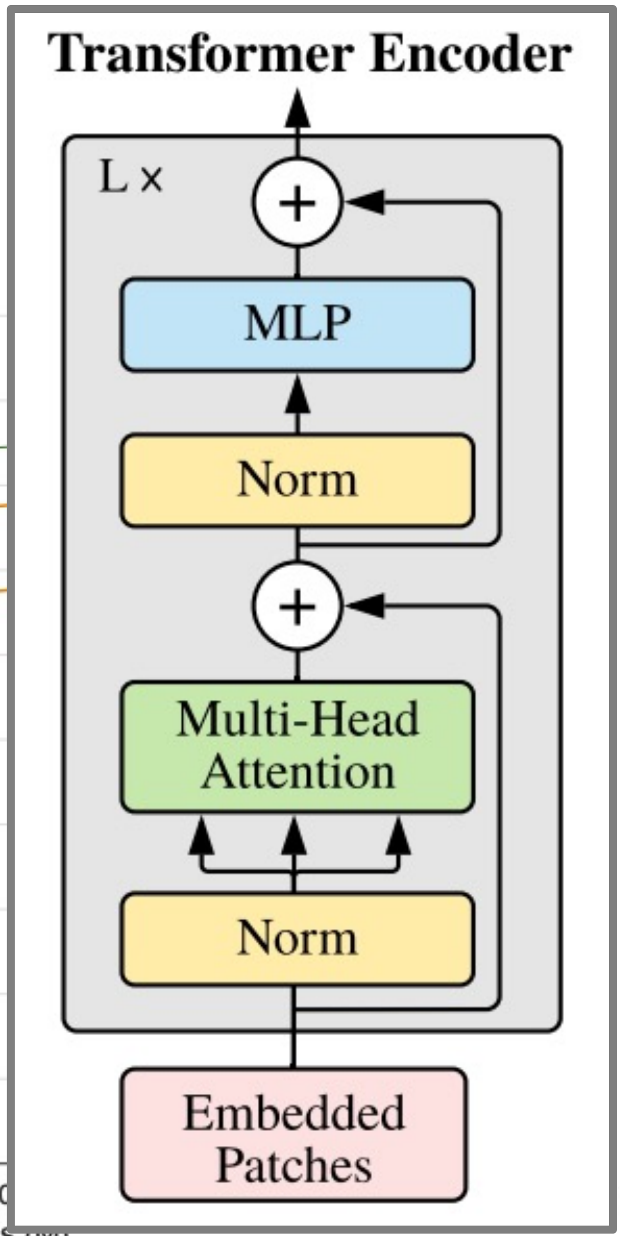
- All tend to work out-of-the-box, w/ MoCo v3 an overall winner in ViT

BatchNorm Helps ViT

- Yields 1% improvement by replacing LayerNorm
    - Best: 81.0 w/ ViT-L/7

- However, incurs instability if applied to attention block





End-to-End Fine-Tuning

- MoCo v3 pre-training helps *beyond* linear-eval
  - Good initialization for end-to-end fine-tuning

method	pre-train data	ViT-S	ViT-B	ViT-L
Masked patch pred.	JFT-300M	-	79.9	-
DEiT	-	79.9	81.8	n/a
DINO	ImageNet-1K	81.5	82.8	n/a
MoCo v3	ImageNet-1K	81.4	83.2	84.1

Take-Aways

1. Grid features work just as well as region features for V + L
   - https://github.com/facebookresearch/grid-feats-vqa

1. Simple Siamese network can learn without collapsing
   - https://github.com/facebookresearch/simsiam

2. ViT works with Siamese based frameworks, subject to instability
   - https://github.com/facebookresearch/moco-v3