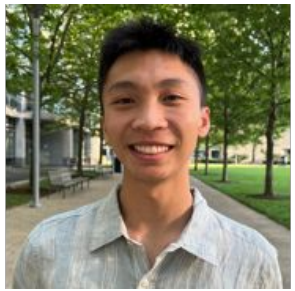
CS294-158 Deep Unsupervised Learning

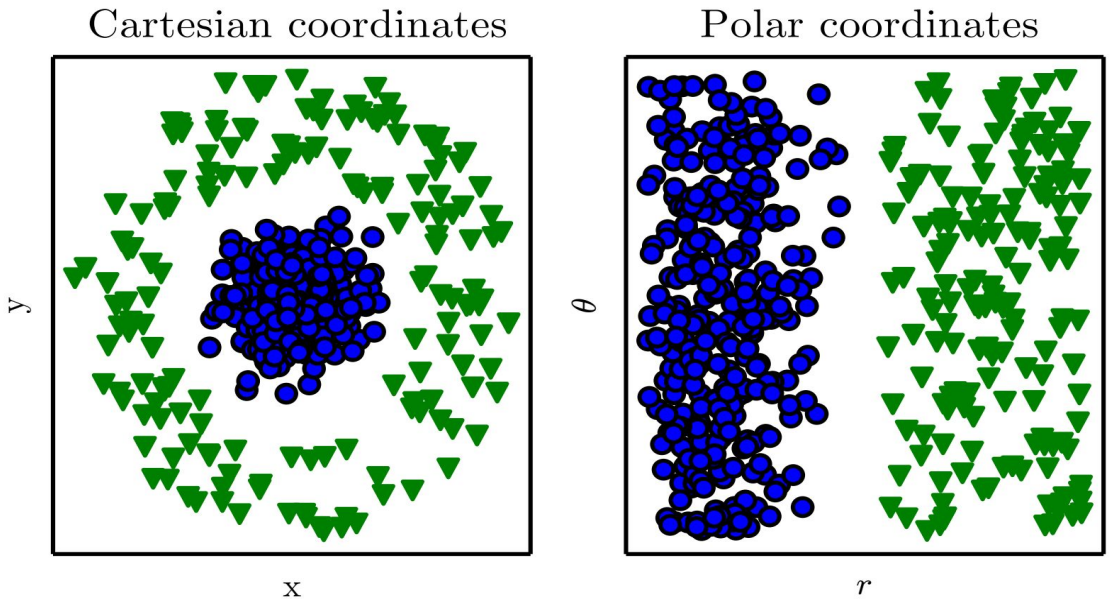
Lecture 7 Self-Supervised Learning





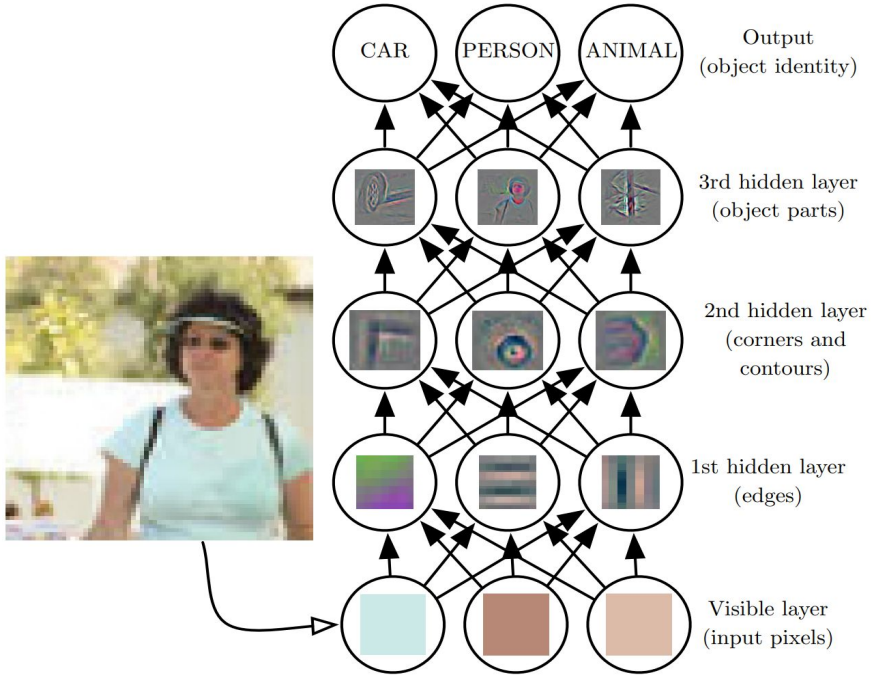

Pieter Abbeel, Wilson Yan, Kevin Frans, Philipp Wu

Reminder: Representations Matter



Goodfellow

Depth often refines representations



Goodfellow

Today

- Goal: representation learning
  - i.e. pre-train a NN so it can be finetuned to good performance on a downstream task with limited downstream data
- How about generative models we covered so far?
  - e.g. AR, Flow, VAE, GAN, Diffusion
  - Yes, they can also achieve this
- Today: alternative approaches to representation learning, which do not involve a generative model

What is Self-Supervised Learning?

- A version of unsupervised learning where data provides the supervision
- In general, withhold some part of the data and the task a neural network to predict it from the remaining parts
- Details decide what proxy loss or pretext task the network tries to solve, and depending on the quality of the task, good semantic features can be obtained without actual labels

Motivation: LeCake

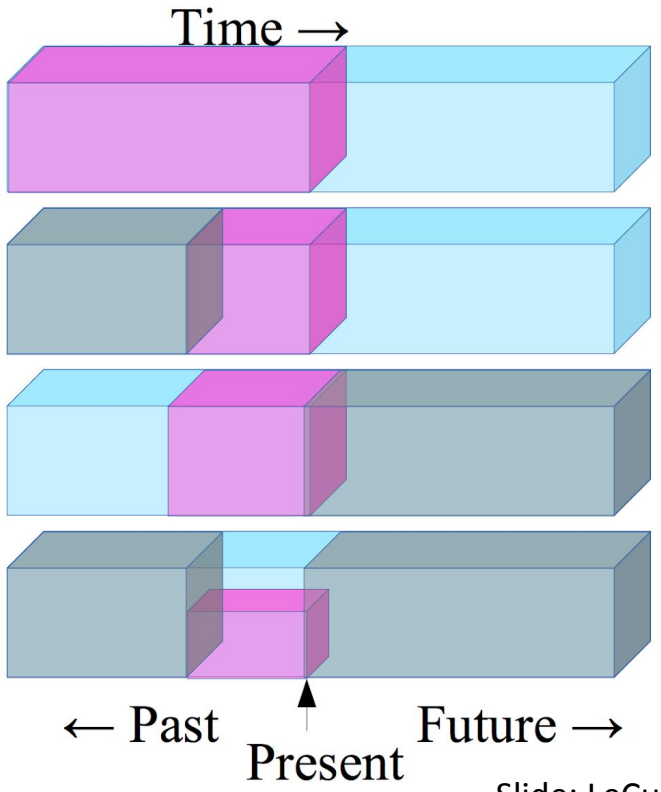
- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**



Yann LeCun's cake

Motivation

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- Pretend there is a part of the input you don't know and predict that.

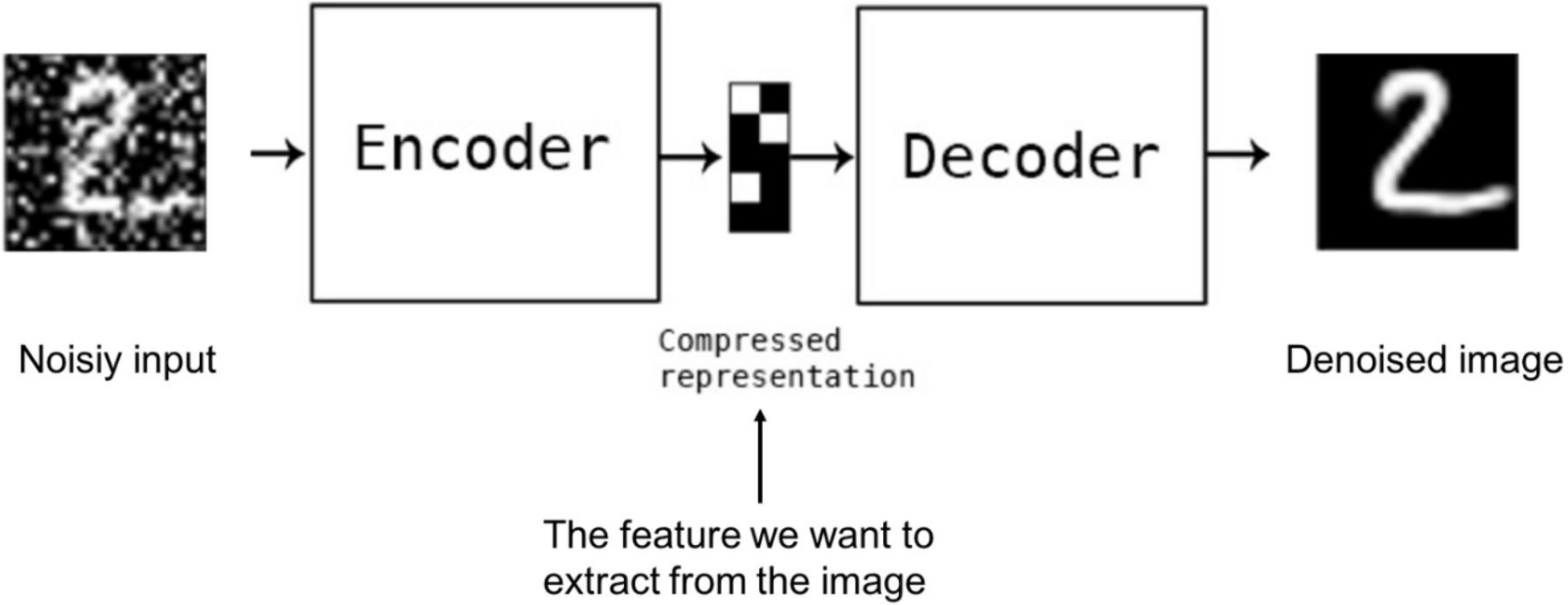


Slide: LeCun

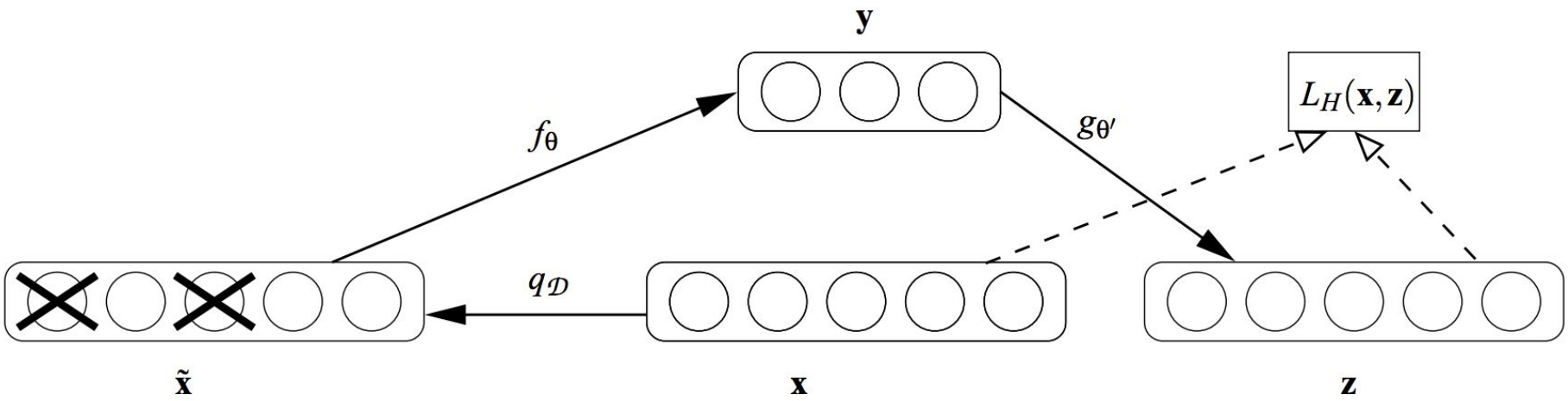
Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

Denoising Autoencoder



Denoising Autoencoder



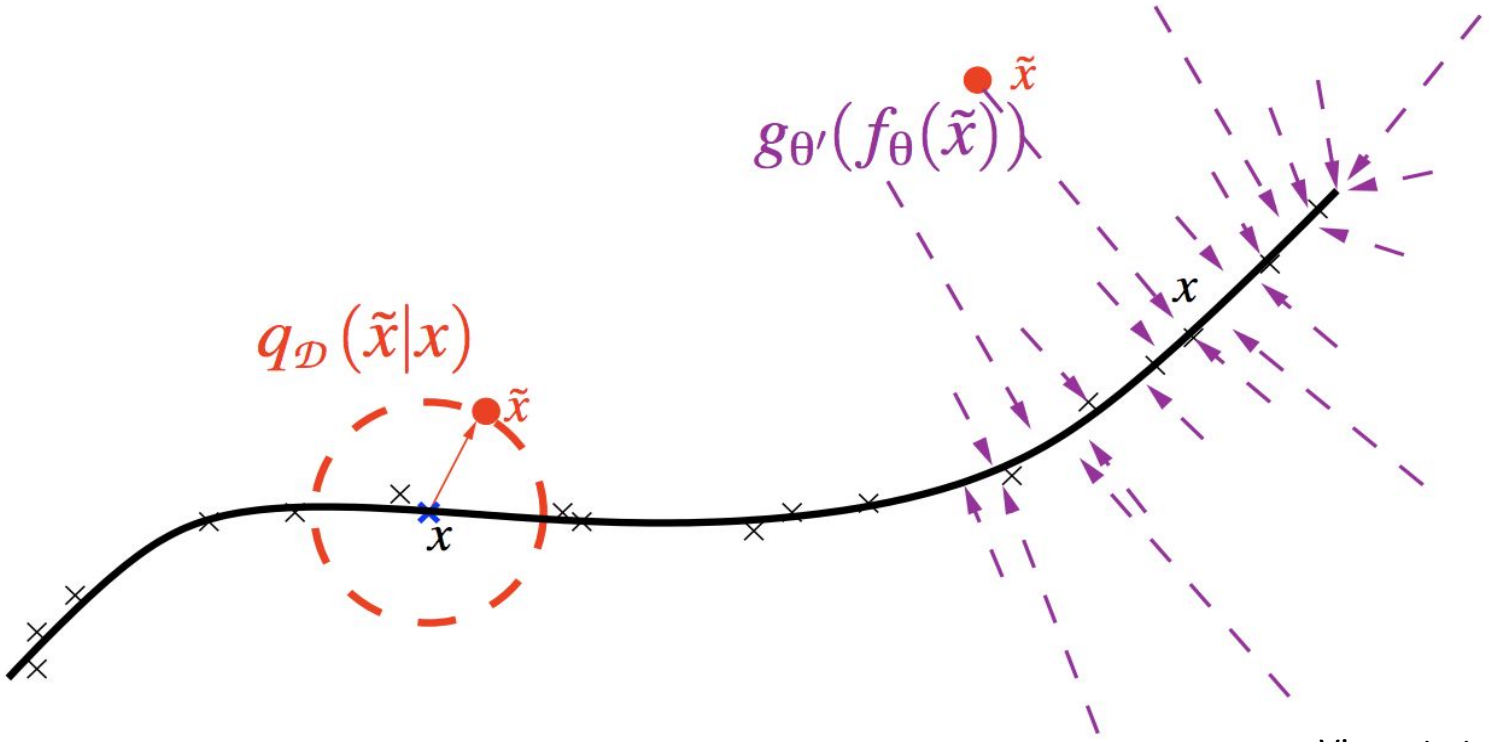
Vincent et al 2010

Denoising Autoencoder

- Additive isotropic *Gaussian noise* (GS): $\tilde{\mathbf{x}}|\mathbf{x} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)$;
- *Masking noise* (MN): a fraction ν of the elements of $\mathbf{x}$ (chosen at random for each example) is forced to 0;
- *Salt-and-pepper noise* (SP): a fraction ν of the elements of $\mathbf{x}$ (chosen at random for each example) is set to their minimum or maximum possible value (typically 0 or 1) according to a fair coin flip.

Vincent et al 2010

Denoising Autoencoder



Vincent et al 2010

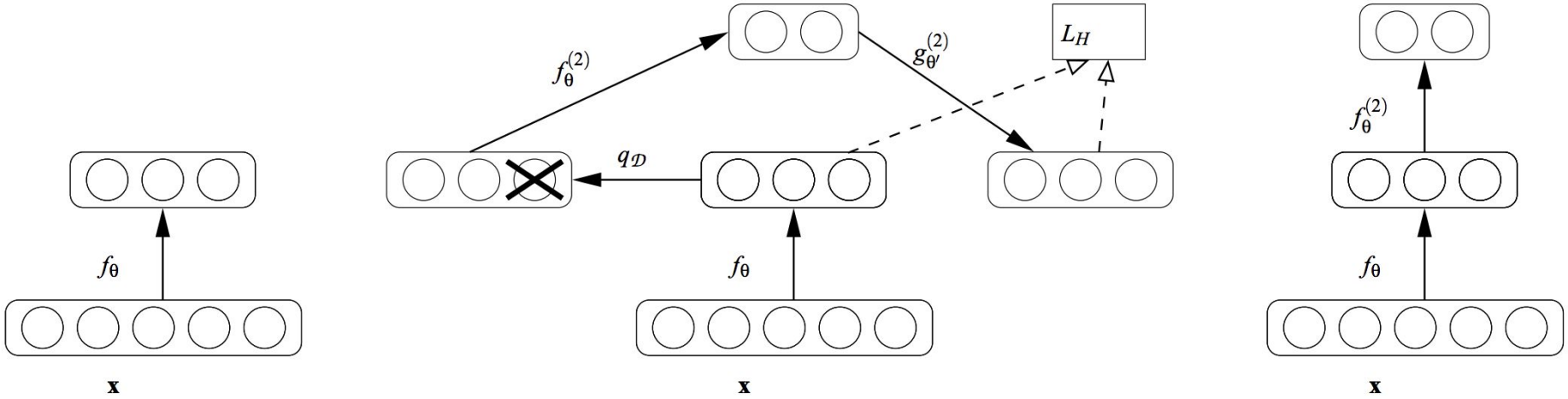
Emphasizing corrupted dimensions

$$L_{2,\alpha}(\mathbf{x},\mathbf{z}) = \alpha\left(\sum_{j\in\mathcal{J}(\tilde{\mathbf{x}})}(\mathbf{x}_j-\mathbf{z}_j)^2\right)+\beta\left(\sum_{j\notin\mathcal{J}(\tilde{\mathbf{x}})}(\mathbf{x}_j-\mathbf{z}_j)^2\right)$$

$$\begin{aligned} L_{\mathrm{IH},\alpha}(\mathbf{x},\mathbf{z}) \quad = \quad & \alpha\left(-\sum_{j\in\mathcal{J}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right) \\ & +\beta\left(-\sum_{j\notin\mathcal{J}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right) \end{aligned}$$

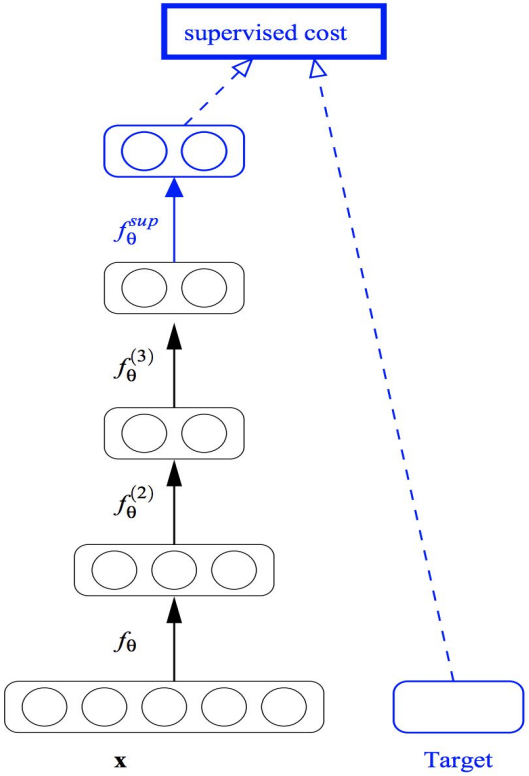
Vincent et al 2010

Stacked Denoising Autoencoder



Vincent et al 2010

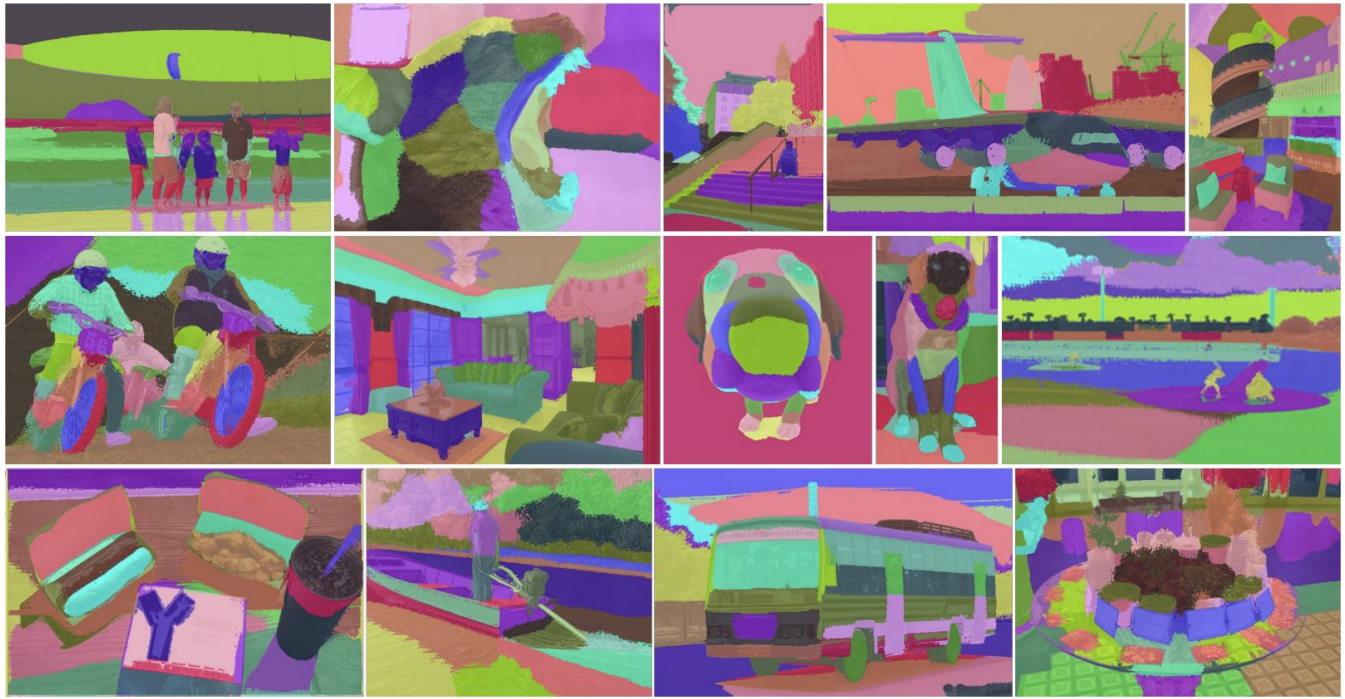
Denoising Autoencoder



Vincent et al 2010

Diffusion Models

EmerDiff: pixel



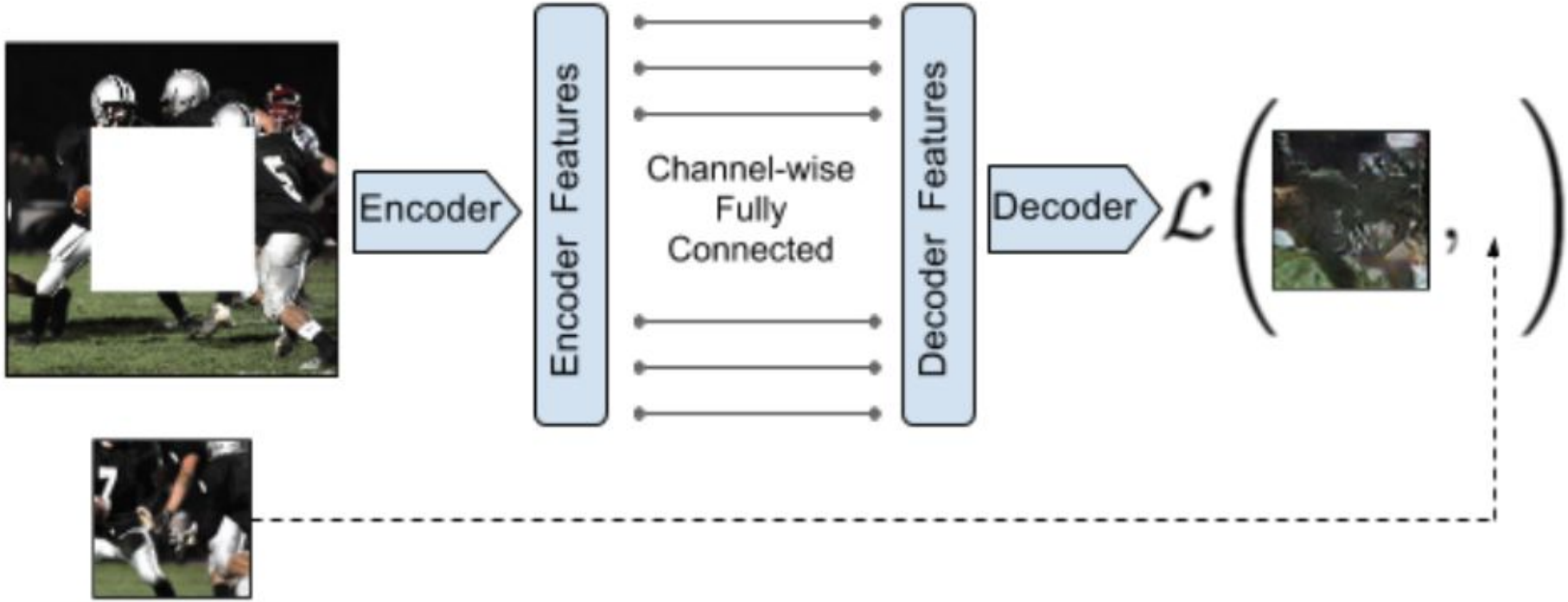
Predict missing pieces





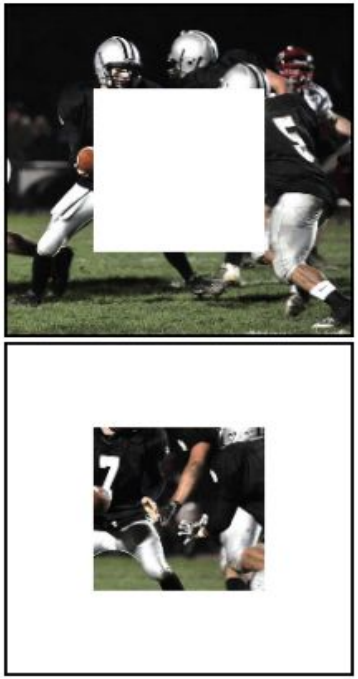
Pathak et al 2016

Context Encoders

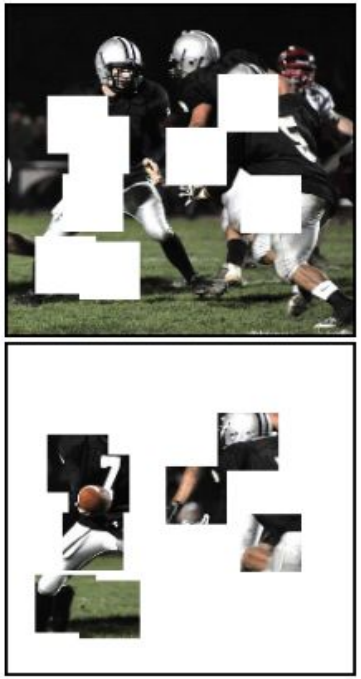


Pathak et al 2016

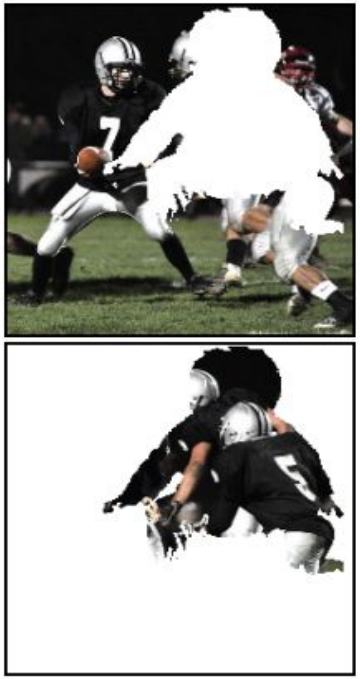
Context Encoders



(a) Central region



(b) Random block



(c) Random region

Pathak et al 2016

Context Encoders

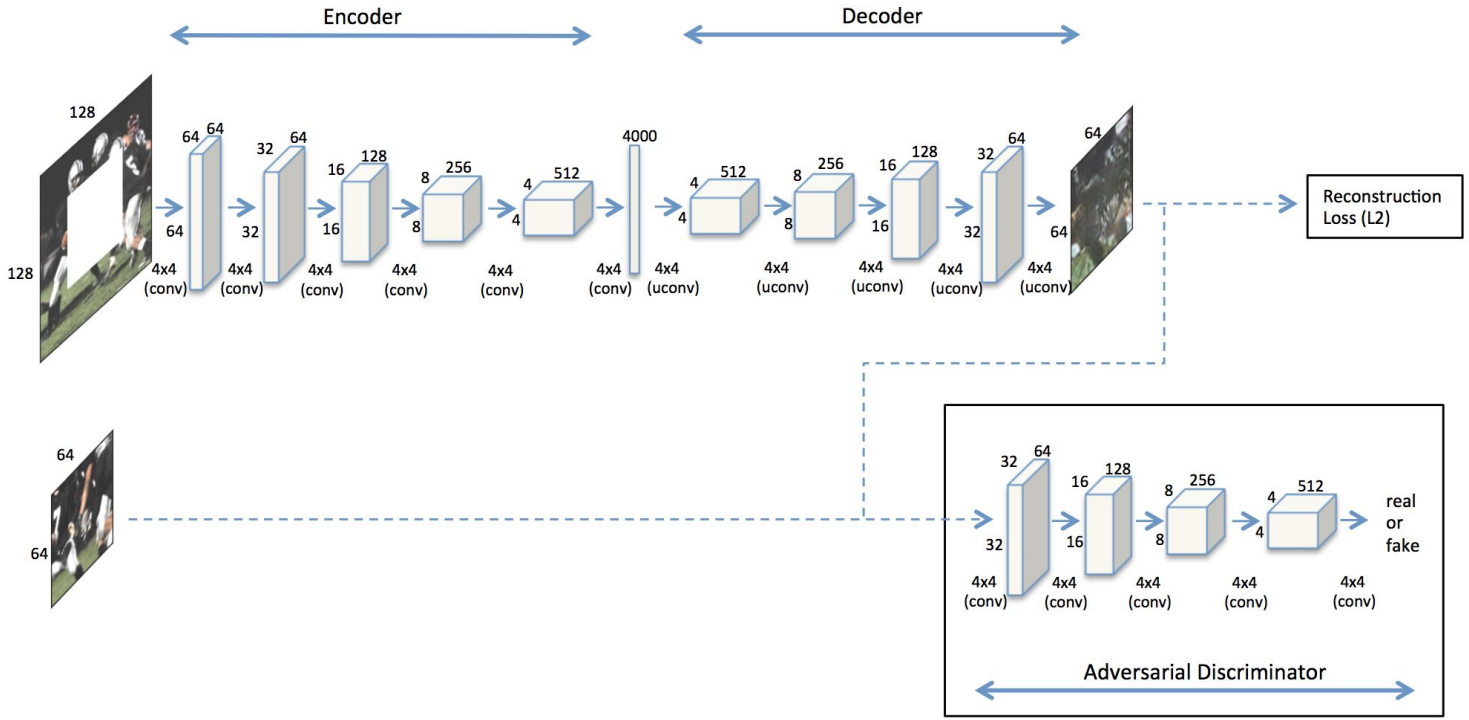
$$\mathcal{L}_{rec}(x) = \|\hat{M} \odot (x - F((1 - \hat{M}) \odot x))\|_2^2$$

$$\mathcal{L}_{adv} = \max_D \quad \mathbb{E}_{x \in \mathcal{X}}[\log(D(x)) + \log(1 - D(F((1 - \hat{M}) \odot x)))]$$

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}$$

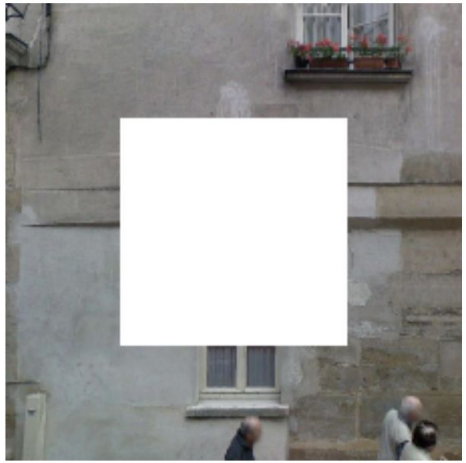
Pathak et al 2016

Context Encoders



Pathak et al 2016

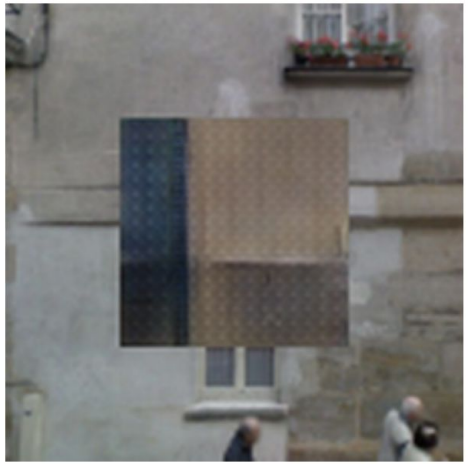
Context Encoders



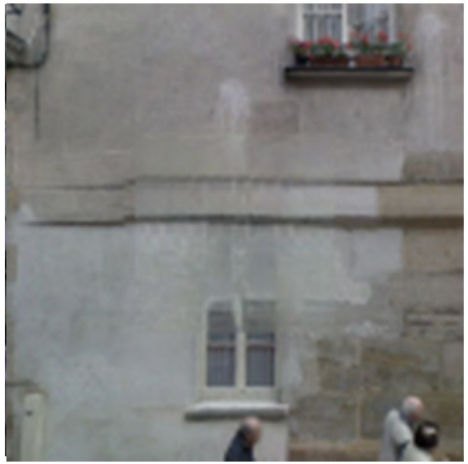
Input Image



L2 Loss



Adversarial Loss



Joint Loss

Pathak et al 2016

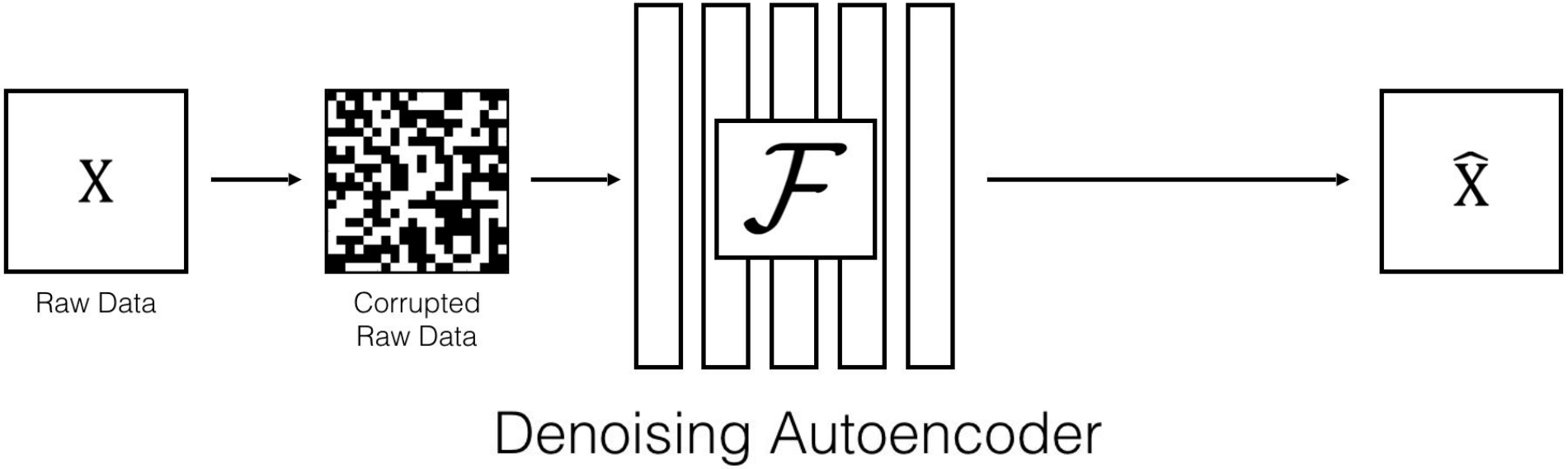
Context Encoders

Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	**78.2%**	**56.8%**	**48.0%**
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Doersch *et al.* [7]	context	4 weeks	55.3%	**46.6%**	-
Wang *et al.* [39]	motion	1 week	**58.4%**	44.0%	-
Ours	context	14 hours	56.5%	44.5%	**29.7%**

Table 2: Quantitative comparison for classification, detection and semantic segmentation. Classification and Fast-RCNN Detection results are on the PASCAL VOC 2007 test set. Semantic segmentation results are on the PASCAL VOC 2012 validation set from the FCN evaluation described in Section 5.2.3, using the additional training data from [18], and removing overlapping images from the validation set [28].

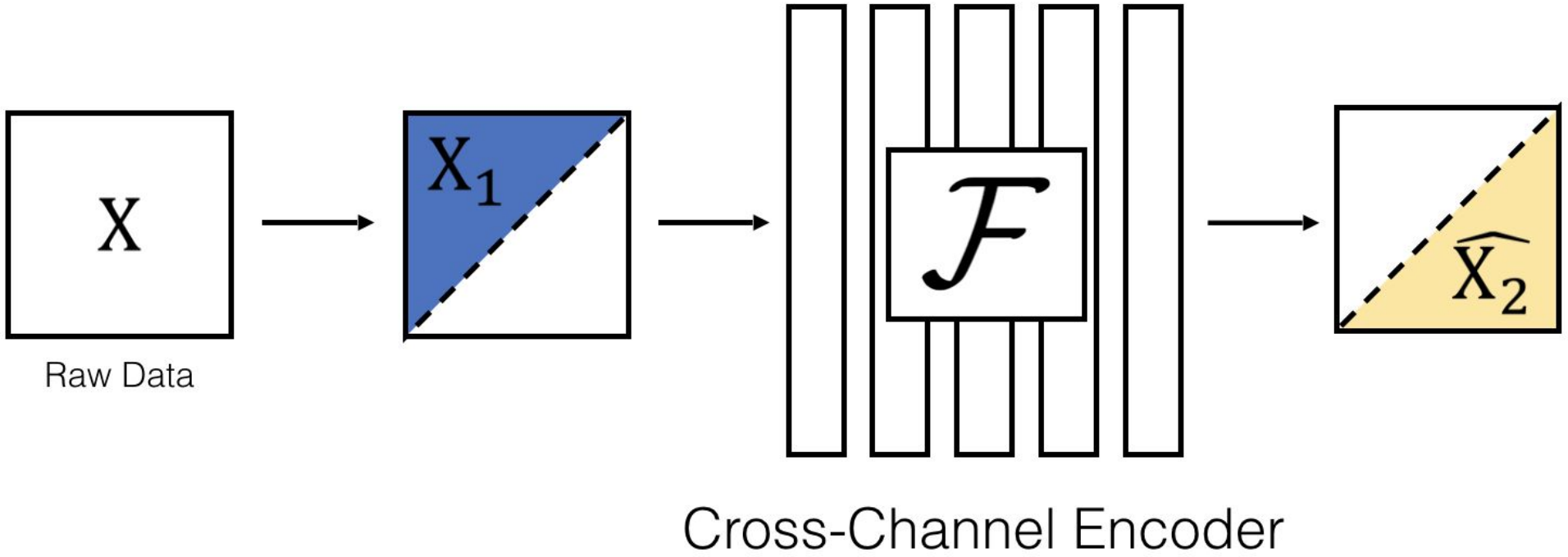
Pathak et al 2016

Predicting one view from another



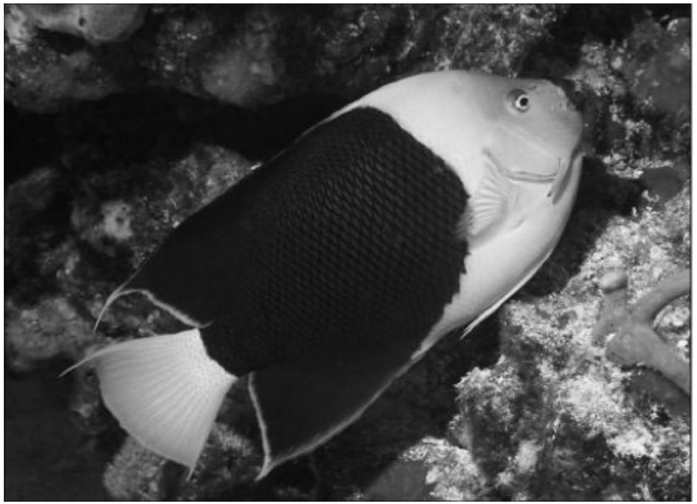
Slide: Richard Zhang

Predicting one view from another



Slide: Richard Zhang

Predicting one view from another



$\mathcal{F}$

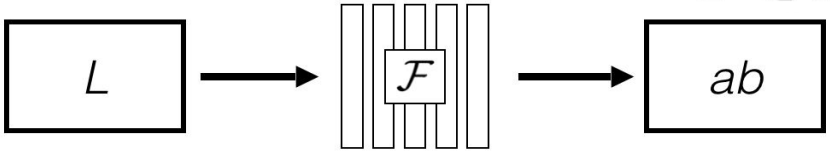


Grayscale image: *L* channel

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

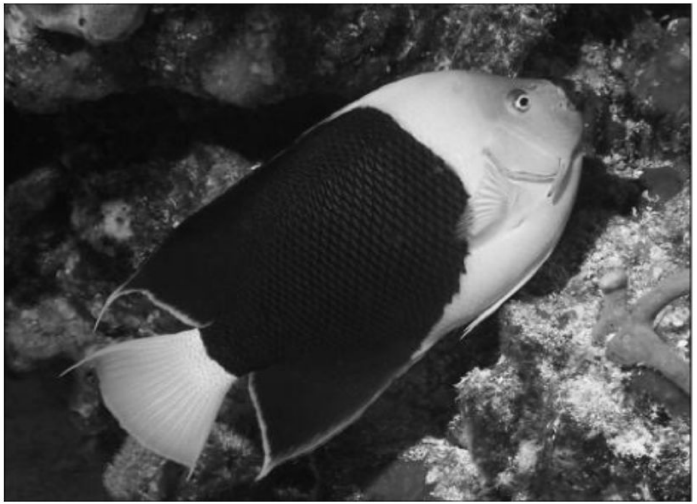
Color information: *ab* channels

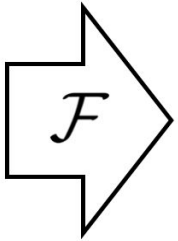
$\hat{\mathbf{Y}} \in \mathbb{R}^{H \times W \times 2}$



Slide: Richard Zhang

Predicting one view from another





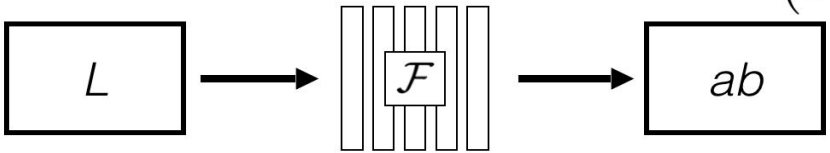


Grayscale image: *L* channel

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

Concatenate (*L*,*ab*) channels

$(\mathbf{X}, \widehat{\mathbf{Y}})$



Slide: Richard Zhang

Predicting one view from another



Ground Truth



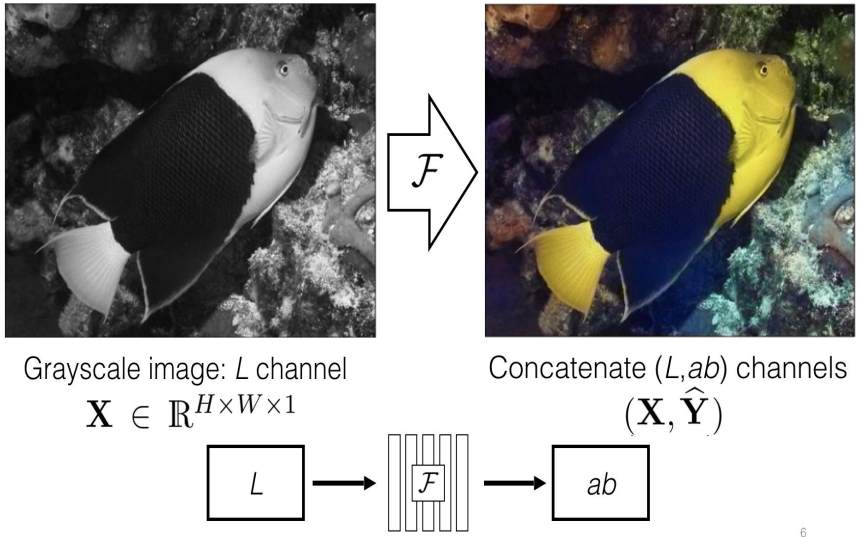
L2 regression



Pixelwise classification

Slide: Richard Zhang

Predicting one view from another

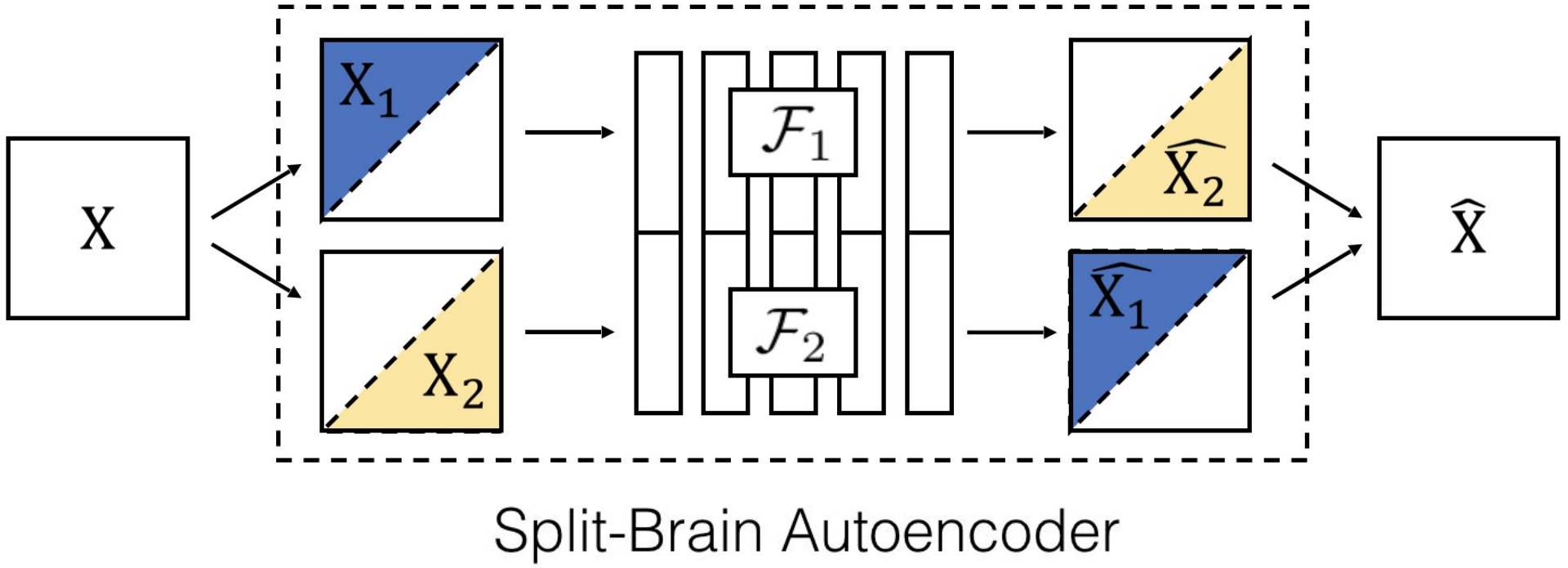


$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

$$\mathrm{L}(\widehat{\mathbf{Z}}, \mathbf{Z}) = -\frac{1}{HW} \sum_{h,w} \sum_{q} \mathbf{Z}_{h,w,q} \log(\widehat{\mathbf{Z}}_{h,w,q})$$

Slide: Richard Zhang

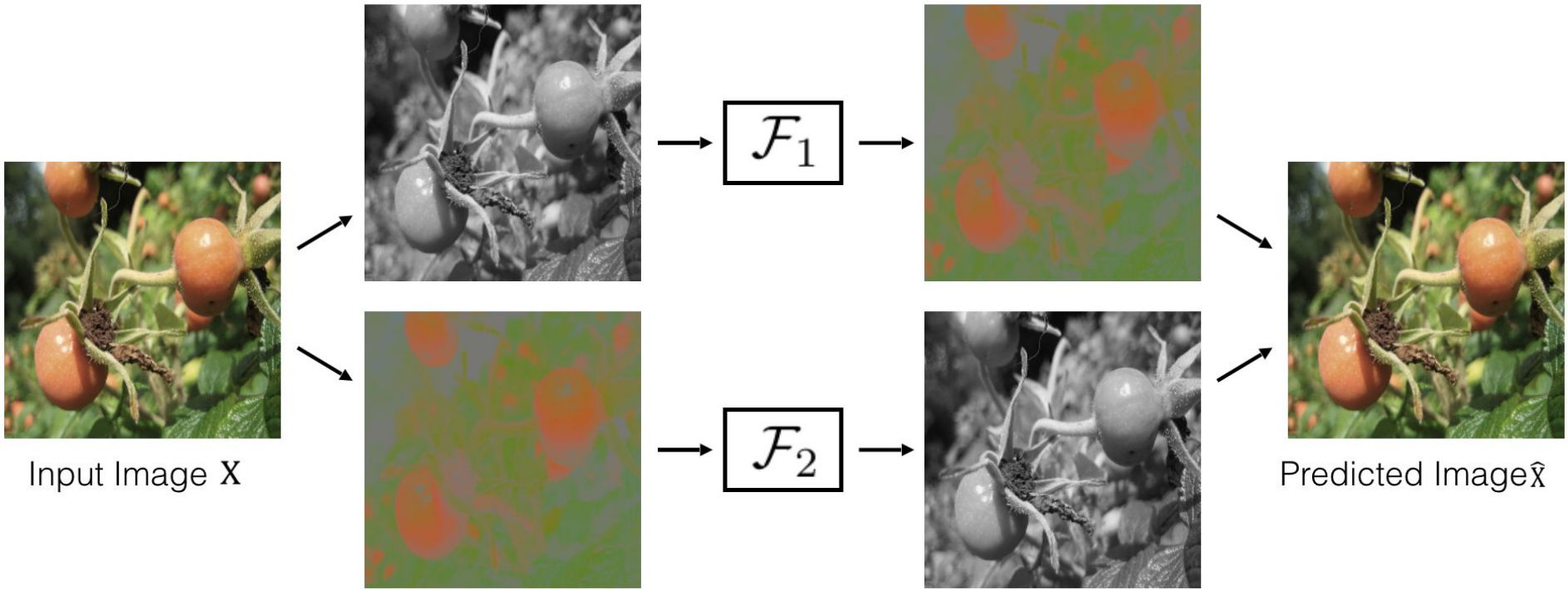
Predicting one view from another



Split-Brain Autoencoder

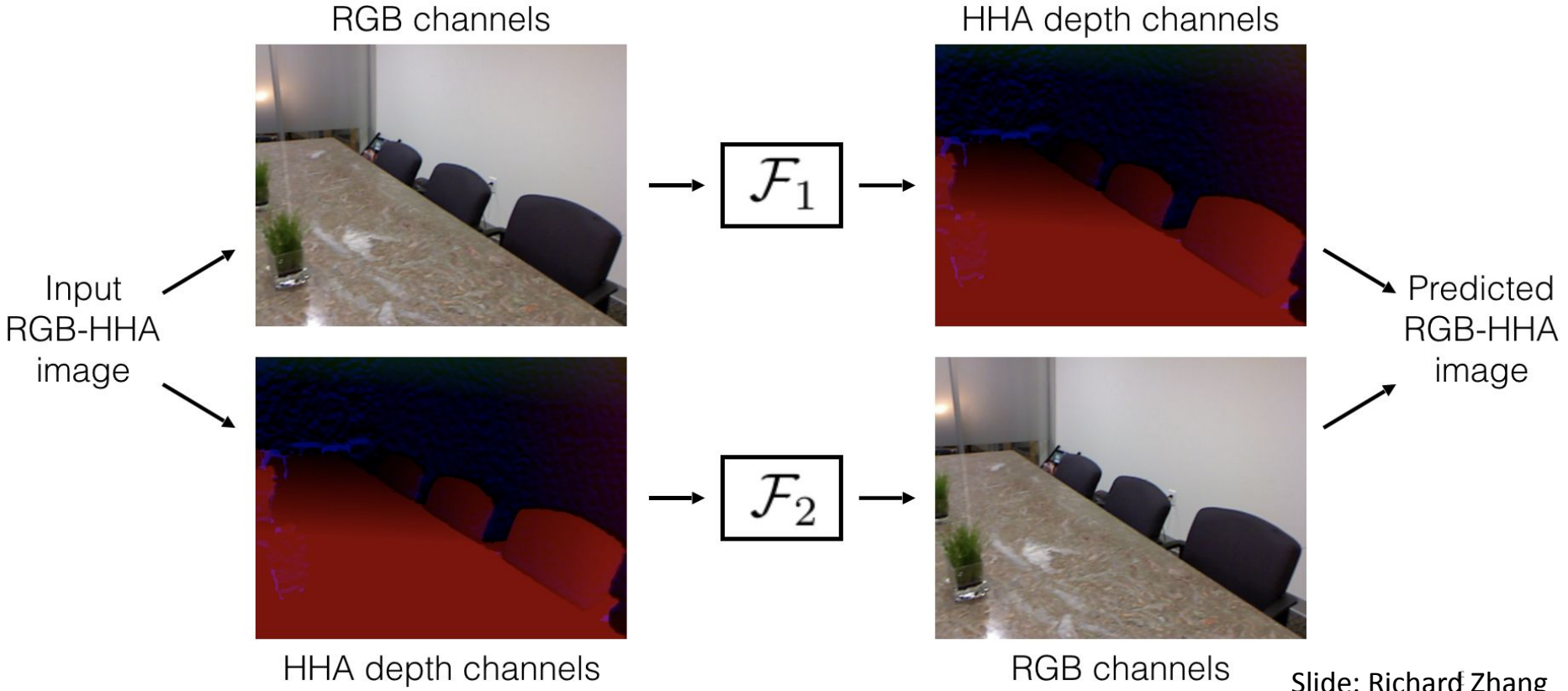
Slide: Richard Zhang

Predicting one view from another



Slide: Richard Zhang

Predicting one view from another



Slide: Richard Zhang

Temporal coherence of color

Task: given a color video …
Colorize all frames of a gray scale version using a reference frame



Reference Frame



Gray-scale Video

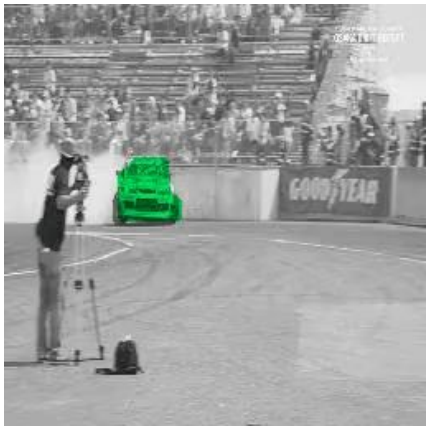
Slide: Zisserman

Tracking emerges from colorization









GIFs from Google AI Blog post

MAE

Masked Autoencoders Are Scalable Vision Learners

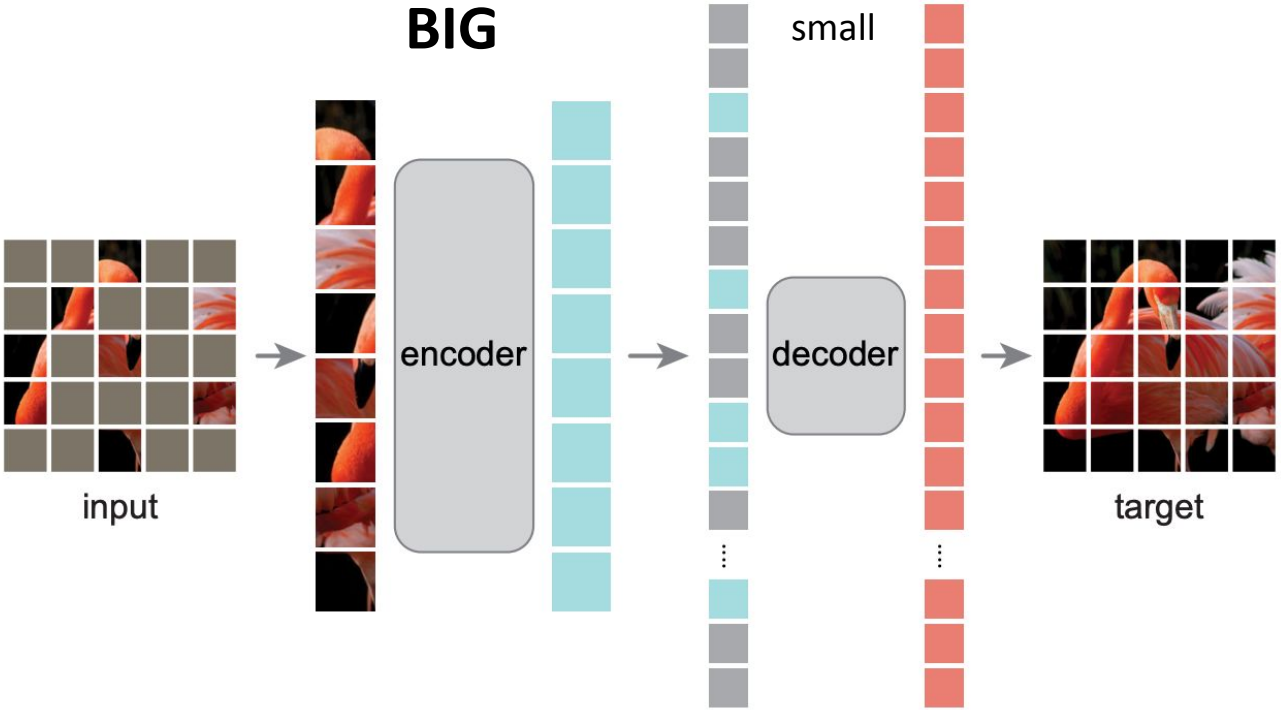
Kaiming He*,† Xinlei Chen* Saining Xie Yanghao Li Piotr Dollár Ross Girshick

*equal technical contribution †project lead

Facebook AI Research (FAIR)

Nov, 2021

MAE



Architecture: Vision Transformer (ViT)

MAE on ImageNet validation images

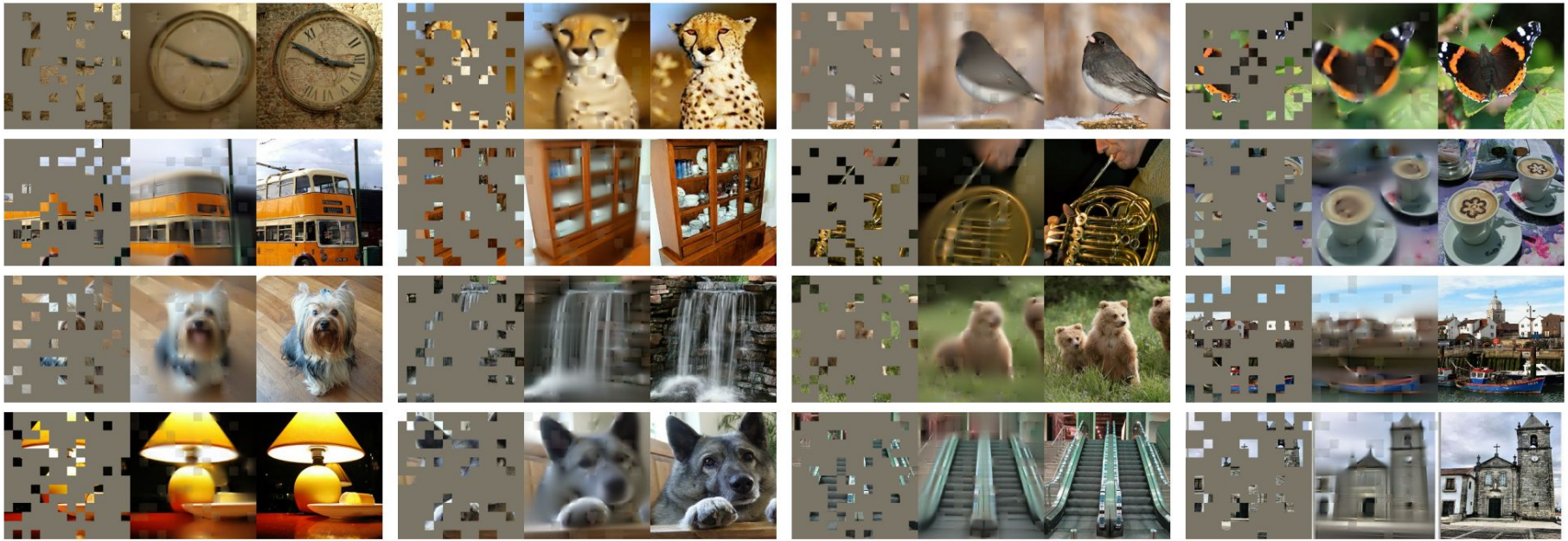


Figure 2. Example results on ImageNet *validation* images. For each triplet, we show the masked image (left), our MAE reconstruction† (middle), and the ground-truth (right). The masking ratio is 80%, leaving only 39 out of 196 patches. More examples are in the appendix.

†*As no loss is computed on visible patches, the model output on visible patches is qualitatively worse. One can simply overlay the output with the visible patches to improve visual quality. We intentionally opt not to do this, so we can more comprehensively demonstrate the method's behavior.*

MAE on CoCo validation images



Figure 3. Example results on COCO validation images, using an MAE trained on ImageNet (the same model weights as in Figure 2). Observe the reconstructions on the two right-most examples, which, although different from the ground truth, are semantically plausible.

Masking Ratio

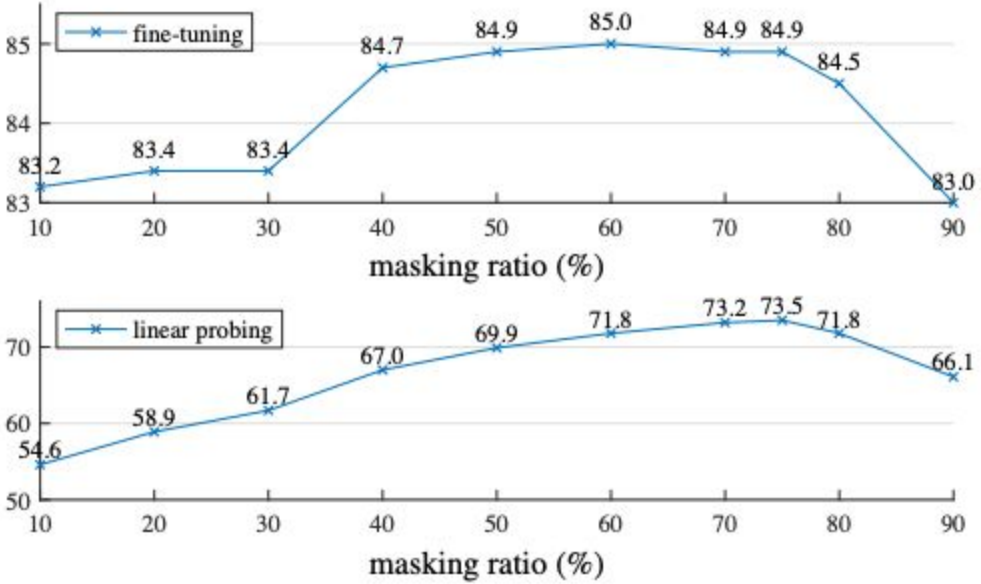


Figure 5. **Masking ratio**. A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

Comparison with Prior SOTA

method	pre-train data	ViT-B	ViT-L	ViT-H	ViT-H_{448}
scratch, our impl.	-	82.3	82.6	83.1	-
DINO [5]	IN1K	82.8	-	-	-
MoCo v3 [9]	IN1K	83.2	84.1	-	-
BEiT [2]	IN1K+DALLE	83.2	85.2	-	-
MAE	IN1K	83.6	85.9	86.9	**87.8**

Table 3. **Comparisons with previous results on ImageNet-1K**. The pre-training data is the ImageNet-1K training set (except the tokenizer in BEiT was pre-trained on 250M DALLE data [50]). All self-supervised methods are evaluated by end-to-end fine-tuning. The ViT models are B/16, L/16, H/14 [16]. The best for each column is underlined. All results are on an image size of 224, except for ViT-H with an extra result on 448. Here our MAE reconstructs normalized pixels and is pre-trained for 1600 epochs.

MAE Cousins / Derivatives

- BeIT
- VideoMAE
- SiamMAE
- Audio-MAE
- M3AE
- MultiMAE
- Multi-View / Masked World Models for Visual Control (covered in 2nd half of lecture)

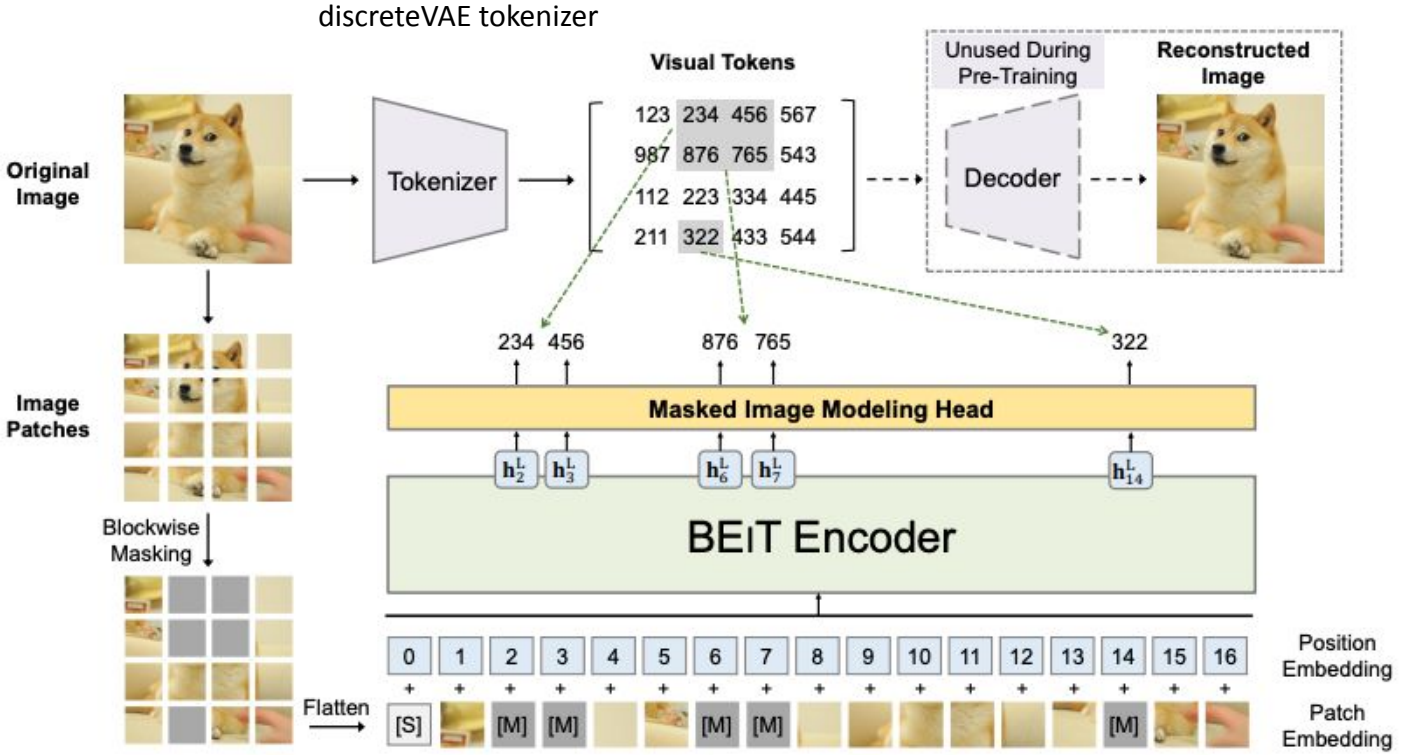
BEIT

BEIT: BERT Pre-Training of Image Transformers

Hangbo Bao[†,*] **Li Dong**[‡], **Songhao Piao**[†], **Furu Wei**[‡]
† Harbin Institute of Technology
‡ Microsoft Research
https://aka.ms/beit

[June 2021 / Sep 2022]

BEIT Architecture



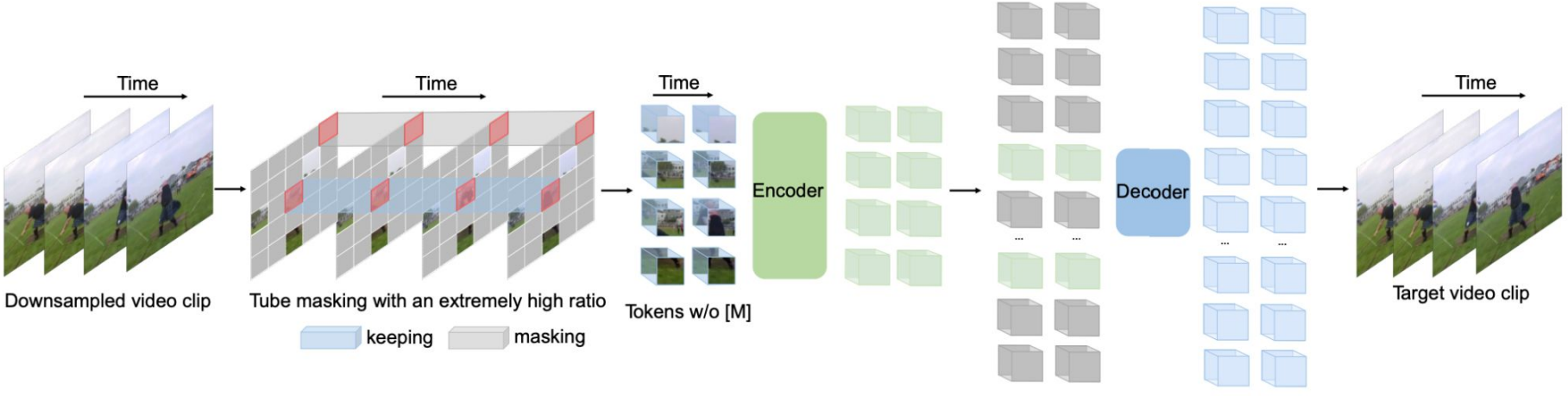
VideoMAE

VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training

Zhan Tong [1,2*] **Yibing Song** [2] **Jue Wang** [2] **Limin Wang** [1,3†]
[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]Tencent AI Lab [3]Shanghai AI Lab
tongzhan@smail.nju.edu.cn {yibingsong.cv, arphid}@gmail.com lmwang@nju.edu.cn

[Oct, 2022]

VideoMAE Architecture



Observations

– High masking ratio: 90% to 95%

– Impressive results even on very small datasets, e.g. 3k videos

– Data quality is more important than data quantity for Self Supervised Video Pretraining. Domain shift between pre-training and target datasets is an important factor.

– VideoMAE with the vanilla ViT backbone can achieve 87.4% on Kinects-400, 75.4% on SomethingSomething V2, 91.3% on UCF101, and 62.6% on HMDB51, without using any extra data.

Experiments on Something-Something V2

Method	Backbone	Extra data	Ex. labels	Frames	GFLOPs	Param	Top-1	Top-5
TEINet$_{En}$ [40]	ResNet50$_{\times 2}$	ImageNet-1K	✓	8+16	99×10×3	50	66.5	N/A
TANet$_{En}$ [41]	ResNet50$_{\times 2}$		✓	8+16	99×2×3	51	66.0	90.1
TDN$_{En}$ [75]	ResNet101$_{\times 2}$		✓	8+16	198×1×3	88	69.6	92.2
SlowFast [23]	ResNet101	Kinetics-400	✓	8+32	106×1×3	53	63.1	87.6
MViTv1 [22]	MViTv1-B		✓	64	455×1×3	37	67.7	90.9
TimeSformer [6]	ViT-B	ImageNet-21K	✓	8	196×1×3	121	59.5	N/A
TimeSformer [6]	ViT-L		✓	64	5549×1×3	430	62.4	N/A
ViViT FE [3]	ViT-L	IN-21K+K400	✓	32	995×4×3	N/A	65.9	89.9
Motionformer [51]	ViT-B		✓	16	370×1×3	109	66.5	90.1
Motionformer [51]	ViT-L		✓	32	1185×1×3	382	68.1	91.2
Video Swin [39]	Swin-B		✓	32	321×1×3	88	69.6	92.7
VIMPAC [65]	ViT-L	HowTo100M+DALLE	✗	10	N/A×10×3	307	68.1	N/A
BEVT [77]	Swin-B	IN-1K+K400+DALLE	✗	32	321×1×3	88	70.6	N/A
MaskFeat↑312 [80]	MViT-L	Kinetics-600	✓	40	2828×1×3	218	75.0	95.0
VideoMAE	ViT-B	Kinetics-400	✗	16	180×2×3	87	69.7	92.3
VideoMAE	ViT-L	Kinetics-400	✗	16	597×2×3	305	74.0	94.6
VideoMAE	ViT-S	*no external data*	✗	16	57×2×3	22	66.8	90.3
VideoMAE	ViT-B		✗	16	180×2×3	87	70.8	92.4
VideoMAE	ViT-L		✗	16	597×2×3	305	74.3	94.6
VideoMAE	ViT-L		✗	32	1436×1×3	305	**75.4**	**95.2**

Table 6: **Comparison with the state-of-the-art methods on Something-Something V2**. Our VideoMAE reconstructs normalized cube pixels and is pre-trained with a masking ratio of 90% for 2400 epochs. "Ex. labels ✗" means only *unlabelled* data is used during the pre-training phase. "N/A" indicates the numbers are not available for us.

Experiments on Kinetics 400

Method	Backbone	Extra data	Ex. labels	Frames	GFLOPs	Param	Top-1	Top-5
NL I3D [78]	ResNet101	ImageNet-1K	✓	128	359×10×3	62	77.3	93.3
TANet [41]	ResNet152		✓	16	242×4×3	59	79.3	94.1
TDN$_{En}$ [75]	ResNet101		✓	8+16	198×10×3	88	79.4	94.4
TimeSformer [6]	ViT-L	ImageNet-21K	✓	96	8353×1×3	430	80.7	94.7
ViViT FE [3]	ViT-L		✓	128	3980×1×3	N/A	81.7	93.8
Motionformer [51]	ViT-L		✓	32	1185×10×3	382	80.2	94.8
Video Swin [39]	Swin-L		✓	32	604×4×3	197	83.1	95.9
ViViT FE [3]	ViT-L	JFT-300M	✓	128	3980×1×3	N/A	83.5	94.3
ViViT [3]	ViT-H	JFT-300M	✓	32	3981×4×3	N/A	84.9	95.8
VIMPAC [65]	ViT-L	HowTo100M+DALLE	✗	10	N/A×10×3	307	77.4	N/A
BEVT [77]	Swin-B	IN-1K+DALLE	✗	32	282×4×3	88	80.6	N/A
MaskFeat↑352 [80]	MViT-L	Kinetics-600	✗	40	3790×4×3	218	87.0	97.4
ip-CSN [69]	ResNet152	*no external data*	✗	32	109×10×3	33	77.8	92.8
SlowFast [23]	R101+NL		✗	16+64	234×10×3	60	79.8	93.9
MViTv1 [22]	MViTv1-B		✗	32	170×5×1	37	80.2	94.4
MaskFeat [80]	MViT-L		✗	16	377×10×1	218	84.3	96.3
VideoMAE	ViT-S	*no external data*	✗	16	57×5×3	22	79.0	93.8
VideoMAE	ViT-B		✗	16	180×5×3	87	81.5	95.1
VideoMAE	ViT-L		✗	16	597×5×3	305	85.2	96.8
VideoMAE	ViT-H		✗	16	1192×5×3	633	**86.6**	**97.1**
VideoMAE↑320	ViT-L	*no external data*	✗	32	3958×4×3	305	86.1	97.3
VideoMAE↑320	ViT-H		✗	32	7397×4×3	633	**87.4**	**97.6**

Table 7: **Comparison with the state-of-the-art methods on Kinetics-400**. Our VideoMAE reconstructs normalized cube pixels. Here models are self-supervised pre-trained with a masking ratio of 90% for 1600 epochs on Kinetics-400. VideoMAE↑320 is initialized from its 224^2 resolution counterpart and then fine-tuned for evaluation. "Ex. labels ✗" means only *unlabelled* data is used during the pre-training phase. "N/A" indicates the numbers are not available for us.

Siam MAE

Siamese Masked Autoencoders

Agrim Gupta[1]* **Jiajun Wu**[1] **Jia Deng**[2] **Li Fei-Fei**[1]
[1]Stanford University, [2]Princeton University

[May 2023]

SiamMAE: Architecture

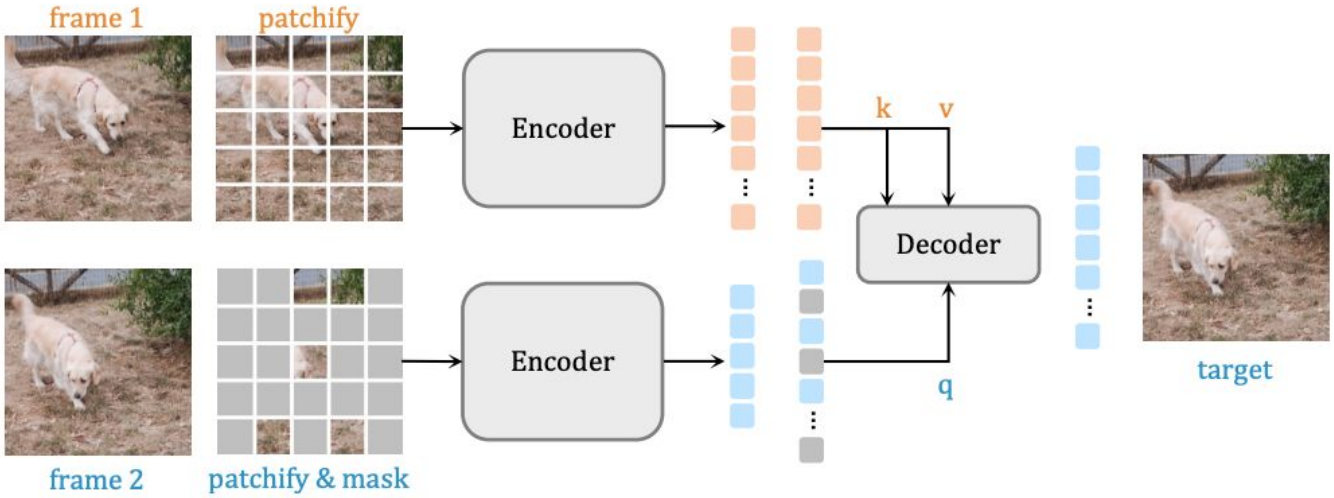


Figure 1: **Siamese Masked Autoencoders.** During pre-training we randomly sample a pair of video frames and randomly mask a huge fraction (95%) of patches of the future frame while leaving the past frame unchanged. The two frames are processed *independently* by a siamese encoder parametrized by a ViT [31]. The decoder consists of a sequence of cross-attention layers and predicts missing patches in the future frame. Videos available at this project page.

SiamMAE: Key idea

– By masking a large fraction (95%) of patches in the future frame while leaving the past frame unchanged, SiamMAE encourages the network to focus on object motion and learn object-centric representations.

– SiamMAE outperform state-of-the-art self-supervised methods on video object segmentation, pose keypoint propagation, and semantic part propagation tasks

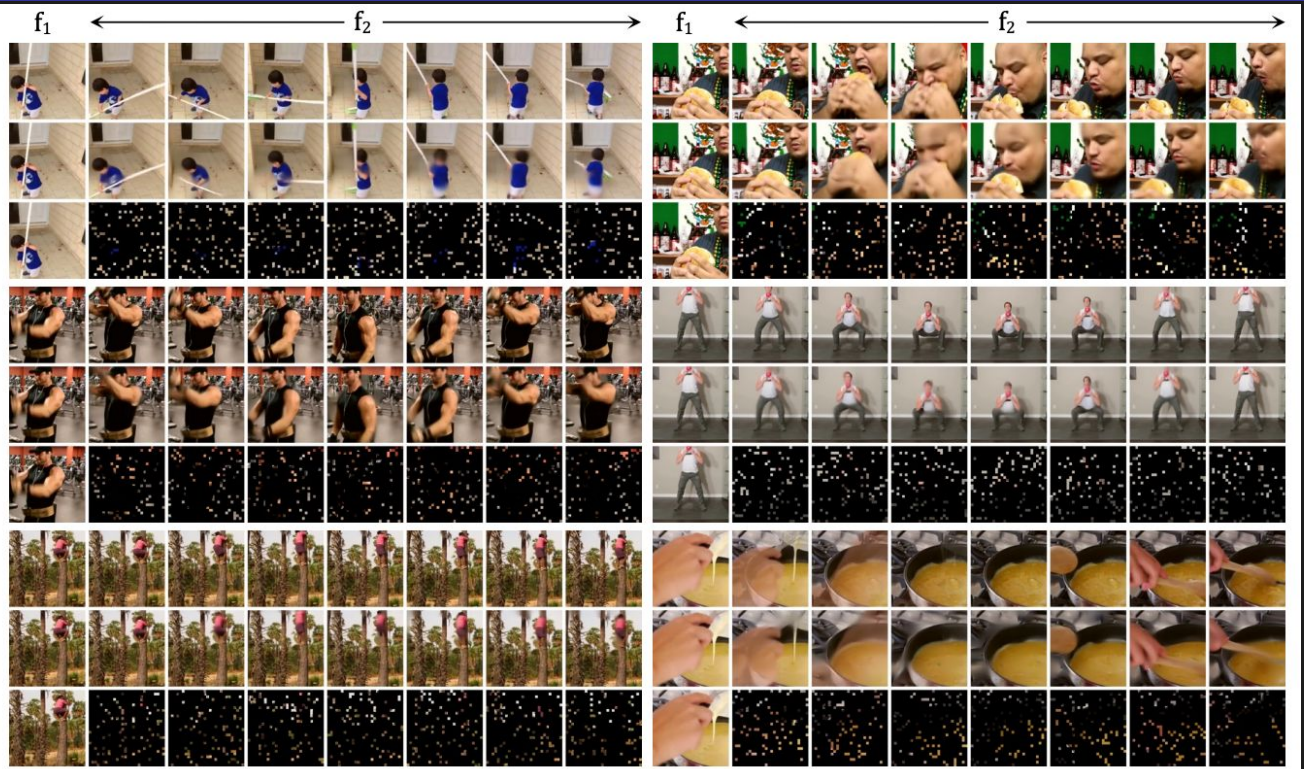


Figure 2: **Visualizations** on the Kinetics-400 [93] validation set (masking ratio 90%). For each video sequence, we sample a clip of 8 frames with a frame gap of 4 and show the original video (top), SiamMAE output (middle), and masked future frames (bottom). Reconstructions are shown with f_1 as the first frame of the video clip and f_2 as the remaining frames, using a SiamMAE pre-trained ViT-S/8 encoder with a masking ratio of 95%.



Figure 3: **Qualitative** results on three downstream tasks: video object segmentation (DAVIS-2017 [95]), human pose propagation (JHMDB [96]) and semantic part propagation (VIP [97]).

Method	Backbone	Dataset	DAVIS $\mathcal{J}\&\mathcal{F}_m$	DAVIS $\mathcal{J}_m$	DAVIS $\mathcal{F}_m$	VIP mIoU	JHMDB PCK@0.1	JHMDB PCK@0.2
Supervised [98]	ResNet-50	ImageNet	66.0	63.7	68.4	39.5	59.2	78.3
SimSiam [20]	ResNet-50	ImageNet	66.3	64.5	68.2	35.0	58.4	77.5
MoCo [19]	ResNet-50	ImageNet	65.4	63.2	67.6	36.1	60.4	79.3
TimeCycle [14]	ResNet-50	VLOG	40.7	41.9	39.4	28.9	57.7	78.5
UVC [12]	ResNet-50	Kinetics	56.3	54.5	58.1	34.2	56.0	76.6
VFS [16]	ResNet-50	Kinetics	68.9	66.5	71.3	43.2	60.9	80.7
MAE-ST [27]	ViT-L/16	Kinetics	54.6	55.5	53.6	33.2	44.4	72.5
MAE [24]	VIT-B/16	ImageNet	53.5	52.1	55.0	28.1	44.6	73.4
VideoMAE [28]	ViT-S/16	Kinetics	39.3	39.7	38.9	23.3	41.0	67.9
Dino [17]	ViT-S/16	ImageNet	61.8	60.2	63.4	36.2	45.6	75.0
SiamMAE (ours)	ViT-S/16	Kinetics	**62.0**	**60.3**	**63.7**	**37.3**	**47.0**	**76.1**
Dino [17]	ViT-S/8	ImageNet	69.9	66.6	73.1	39.5	56.5	80.3
SiamMAE (ours)	ViT-S/8	Kinetics	**71.4**	**68.4**	**74.5**	**45.9**	**61.9**	**83.8**

Table 1: **Comparison with prior work** on three downstream tasks: video object segmentation (DAVIS-2017 [95]), human pose propagation (JHMDB [96]) and semantic part propagation (VIP [97])

Audio-MAE

Masked Autoencoders that Listen

Po-Yao Huang[1] Hu Xu[1] Juncheng Li[2] Alexei Baevski[1]
Michael Auli[1] Wojciech Galuba[1] Florian Metze[1] Christoph Feichtenhofer[1]

[1]Meta AI [2]Carnegie Mellon University

Audio-MAE

– Encodes audio spectrogram patches with a high masking ratio, feeding only the non-masked tokens through encoder layers.

– The decoder then re-orders and decodes the encoded context padded with mask tokens, in order to reconstruct the input spectrogram.

– Local window attention in the decoder, as audio spectrograms are highly correlated in local time and frequency bands.

– Fine-tune the encoder with a lower masking ratio on target datasets.

– Empirically, Audio-MAE sets new state-of-the-art performance on six audio and speech classification tasks, outperforming other recent models that use external supervised pre-training.

Audio-MAE: Architecture

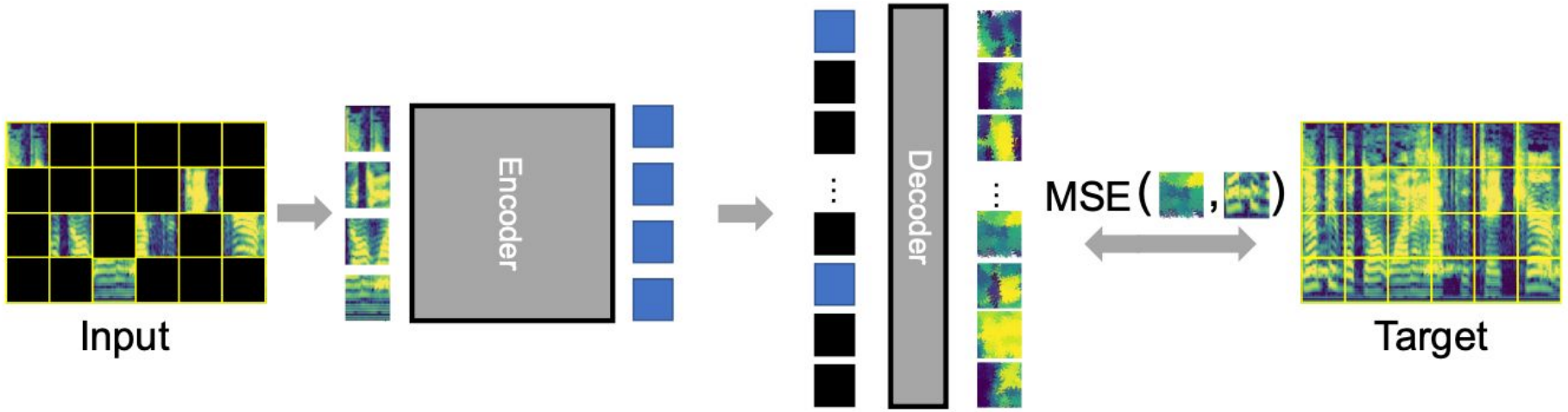


Figure 1: **Audio-MAE for audio self-supervised learning**. An audio recording is first transformed into a spectrogram and split into patches. We embed patches and mask out a large subset (80%). An encoder then operates on the visible (20%) patch embeddings. Finally, a decoder processes the order-restored embeddings and mask tokens to reconstruct the input. Audio-MAE is minimizing the mean square error (MSE) on the masked portion of the reconstruction and the input spectrogram.

Model	Backbone	PT-Data	AS-20K	AS-2M	ESC-50	SPC-2	SPC-1	SID
No pre-training								
ERANN [58]	CNN	-	-	45.0	89.2	-	-	-
PANN [59]	CNN	-	27.8	43.1	83.3	61.8	-	-
In-domain self-supervised pre-training								
wav2vec 2.0 [33]	Transformer	LS	-	-	-	-	96.2*	75.2*
HuBERT [35]	Transformer	LS	-	-	-	-	96.3*	81.4*
Conformer [37]	Conformer	AS	-	41.1	88.0	-	-	-
SS-AST [18]	ViT-B	AS+LS	31.0	-	88.8	98.0	96.0	64.3
Concurrent MAE-based works								
MaskSpec [43]	ViT-B	AS	32.3	47.1	89.6	97.7	-	-
MAE-AST [38]	ViT-B	AS+LS	30.6	-	90.0	97.9	95.8	63.3
Audio-MAE (global)	ViT-B	AS	36.6±.11	46.8±.06	93.6±.11	**98.3**±.06	**97.6**±.06	94.1±.06
Audio-MAE (local)	ViT-B	AS	**37.0**±.11	**47.3**±.11	**94.1**±.10	**98.3**±.06	96.9±.00	**94.8**±.11
Out-of-domain supervised pre-training								
PSLA [30]	EffNet [60]	IN	31.9	44.4	-	96.3	-	-
AST [10]	DeiT-B	IN	34.7	45.9	88.7	98.1	95.5	41.1
MBT [11]	ViT-B	IN-21K	31.3	44.3	-	-	-	-
HTS-AT [29]	Swin-B	IN	-	47.1	97.0†	98.0	-	-
PaSST [28]	DeiT-B	IN	-	47.1	96.8†	-	-	-

Table 2: **Comparison with other state-of-the-art models** on audio and speech classification tasks. Metrics are mAP for AS and accuracy (%) for ESC/SPC/SID. For pre-training (PT) dataset, AS:AudioSet, LS:LibriSpeech, and IN:ImageNet. †: Fine-tuning results with additional supervised training on AS-2M. We gray-out models pre-trained with external non-audio datasets (*e.g.*, ImageNet). Best single models in AS-2M are compared (no ensembles). *: linear evaluation results from [53].

MultiMAE

MultiMAE: Multi-modal Multi-task Masked Autoencoders

Roman Bachmann* David Mizrahi* Andrei Atanov Amir Zamir
Swiss Federal Institute of Technology Lausanne (EPFL)

https://multimae.epfl.ch

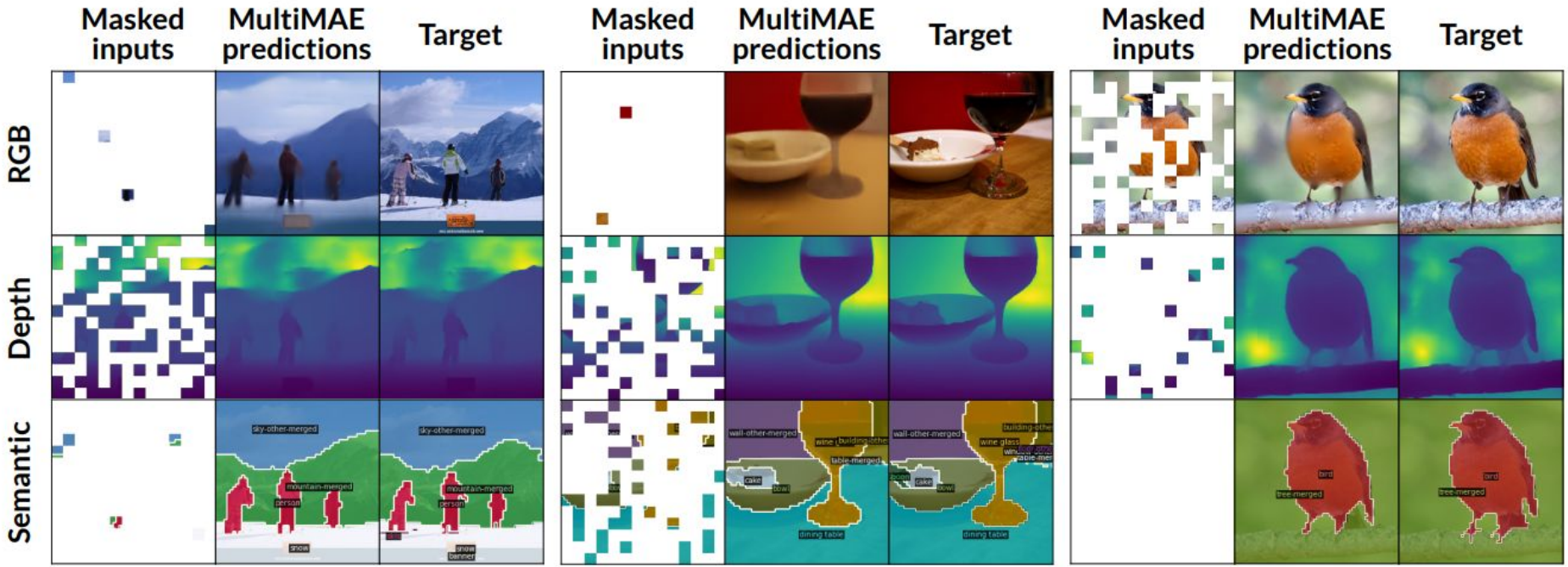


Figure 1. **MultiMAE pre-training objective.** We randomly select 1/6 of all 16×16 image patches from multiple modalities and learn to reconstruct the remaining 5/6 masked patches from them. The figure shows validation examples from ImageNet, where masked inputs (left), predictions (middle), and non-masked images (right) for RGB (top), depth (middle), and semantic segmentation (bottom) are provided. Since we do not compute a loss on non-masked patches, we overlay the input patches on the predictions.

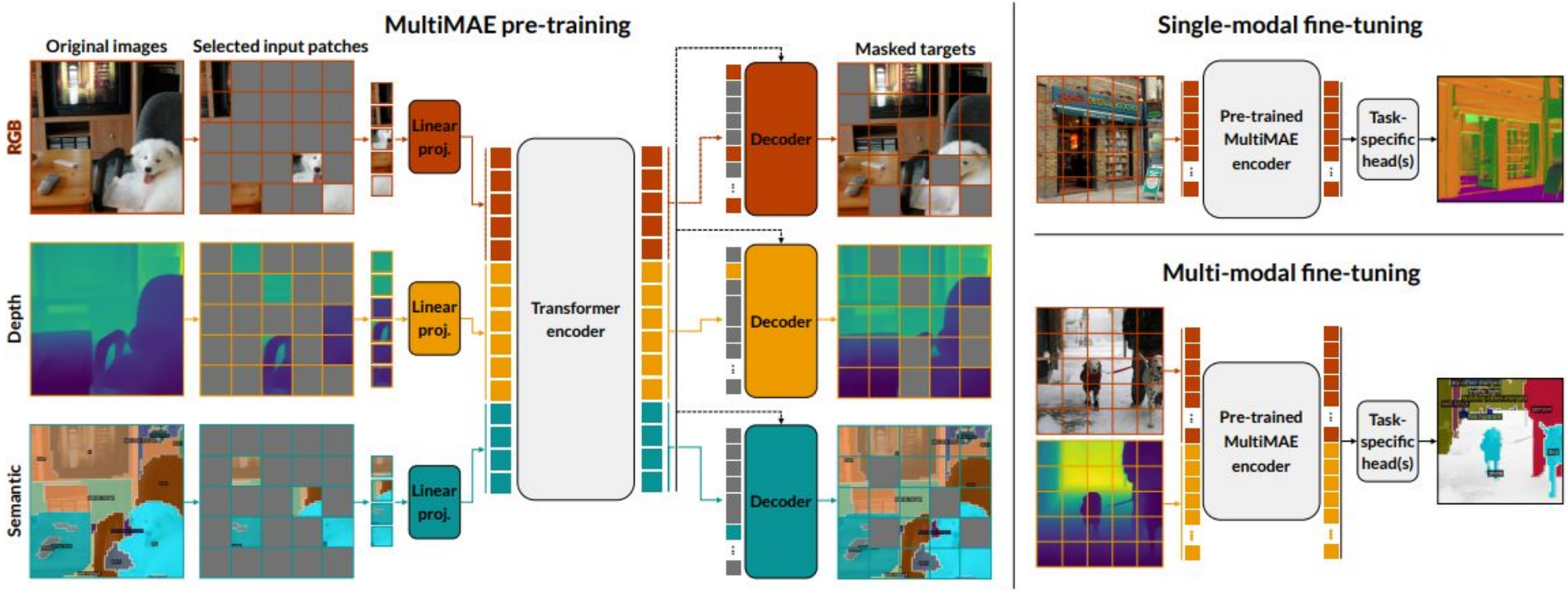


Figure 2. **(Left) MultiMAE pre-training**: A small subset of randomly sampled patches from multiple modalities (e.g., RGB, depth, and semantic segmentation) is linearly projected to tokens with a fixed dimension and encoded using a Transformer. Task-specific decoders reconstruct the masked-out patches by first performing a cross-attention step from queries to the encoded tokens, followed by a shallow Transformer. The queries consist of mask tokens (in gray), with the task-specific encoded tokens added at their respective positions. **(Right) Fine-tuning**: By pre-training on multiple modalities, MultiMAE lends itself to fine-tuning on single-modal and multi-modal downstream tasks. No masking is performed at transfer time.

MultiMAE observations

– like MAE, encoder only processes non-masked tokens

– like MAE, shallow decoders

– pseudolabels for non-RGB modalities

MultiMAE Experiments

Method	IN-1K (C)	ADE20K (S)	Hypersim (S)	NYUv2 (S)	NYUv2 (D)
Supervised [81]	81.8	45.8	33.9	50.1	80.7
DINO [12]	83.1	44.6	32.5	47.9	81.3
MoCo-v3 [17]	82.8	43.7	31.7	46.6	80.9
MAE [35]	**83.3**	**46.2**	<u>36.5</u>	<u>50.8</u>	<u>85.1</u>
MultiMAE	**83.3**	**46.2**	**37.0**	**52.0**	**86.4**

Table 1. **Fine-tuning with RGB-only.** We report the top-1 accuracy (↑) on ImageNet-1K (IN-1K) [23] classification (C), mIoU (↑) on ADE20K [102] , Hypersim [68] , and NYUv2 [73] semantic segmentation (S), as well as δ_1 accuracy (↑) on NYUv2 depth (D). Text in **bold** and <u>underline</u> indicates the first and second-best results, respectively. All methods are pre-trained on ImageNet-1K (with pseudo labels for MultiMAE).

MultiMAE Experiments

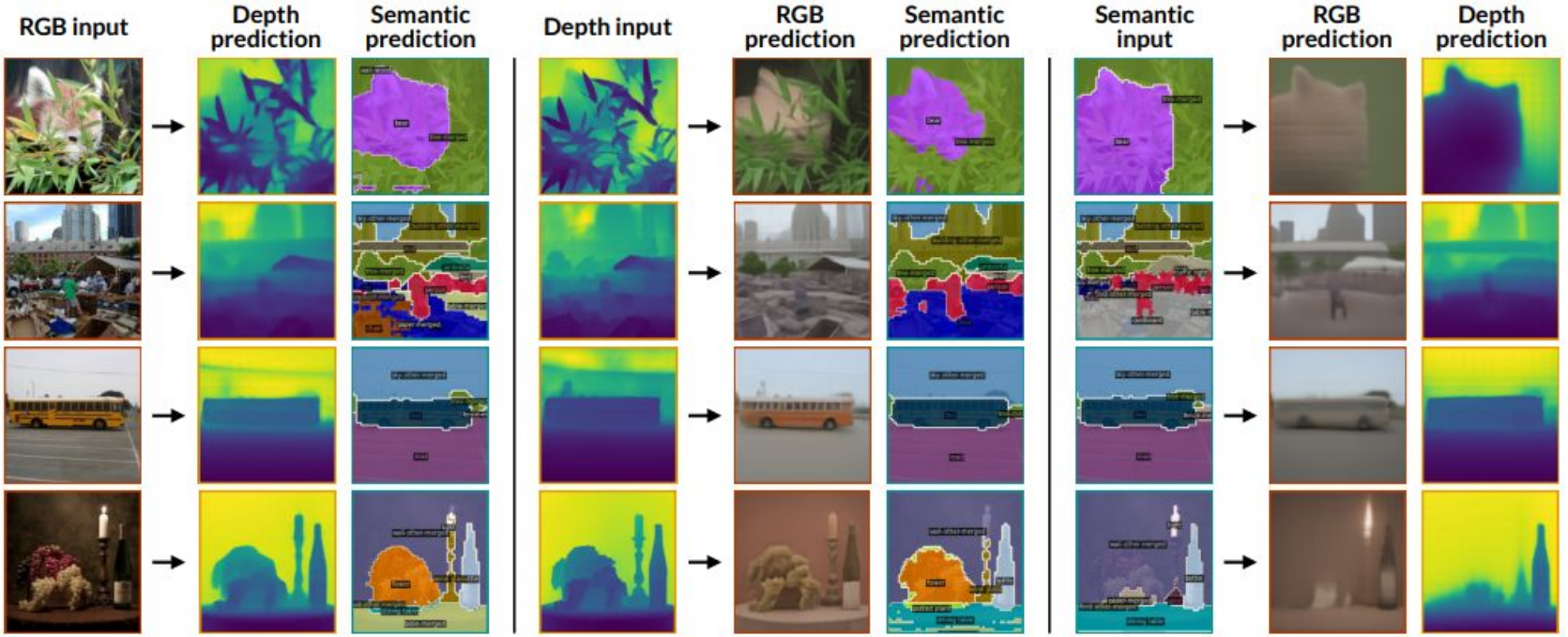


Figure 4. **Single-modal predictions.** We visualize MultiMAE cross-modal predictions on ImageNet-1K validation images. Only a single, full modality is used as input. The predictions remain plausible despite the absence of input patches from other modalities.

M3AE: MultiModal MAE

Multimodal Masked Autoencoders Learn Transferable Representations

Xinyang Geng[1 *] **Hao Liu**[1 2 * †] **Lisa Lee**[2]
Dale Schuurmans[2] **Sergey Levine**[1] **Pieter Abbeel**[1]
[1]UC Berkeley [2]Google Research, Brain Team
[*] Equal contribution. [†] Project lead.
{young.geng, hao.liu}@berkeley.edu

[Oct 2022]

M3AE Contributions

– Until M3AE: dominant multi-modal representation learning paradigm was contrastive learning (CLIP, ALIGN)

– Downside of cross-modal contrastive: only works with paired data

– We find that multimodal pretraining of M3AE on CC12M achieves significantly higher performance on the ImageNet-1k linear classification benchmark [33] compared to pre-training on images only (MAE).

– M3AE performs best when we apply a high mask ratio (75%) on language, while in contrast, language models like BERT conventionally use a low mask ratio (15%)

– Encoder: image patches and language tokens, ViT

– Decoder: light weight, following MAE

M3AE: Architecture

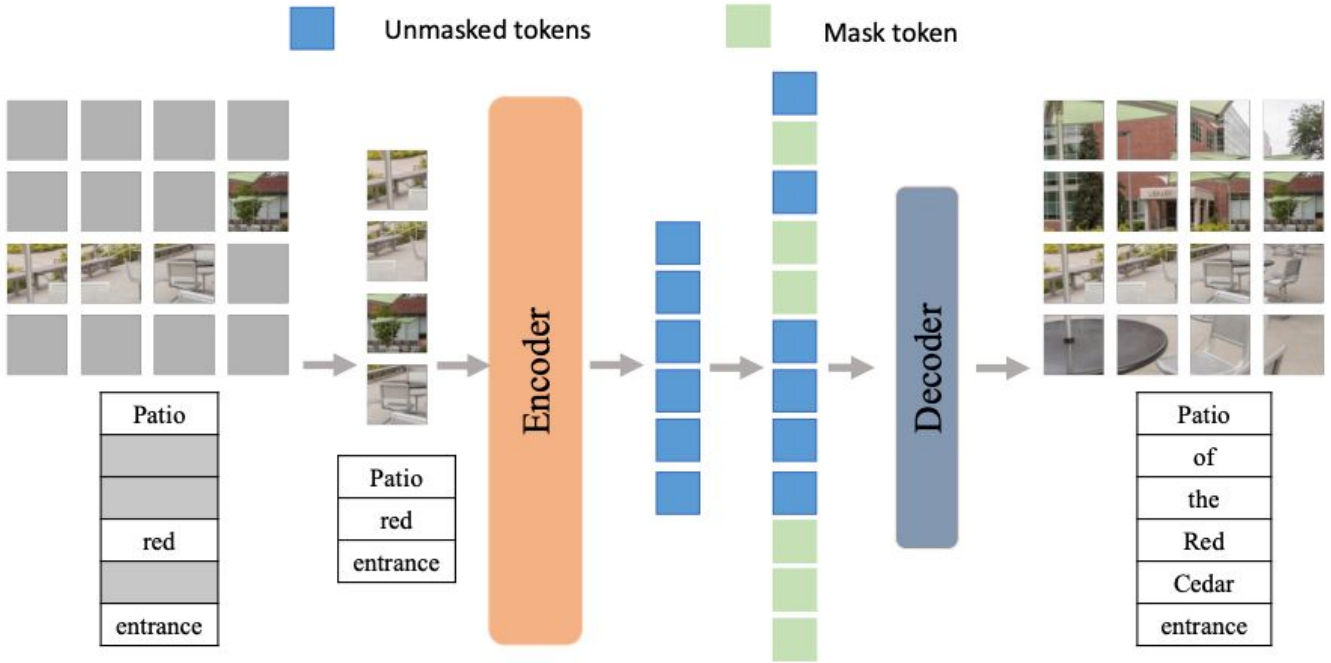


Figure 1: Multimodal masked autoencoder (M3AE) consists of an encoder that maps language tokens and image patches to a shared representation space, and a decoder that reconstructs the original image and language from the representation.

Comparison with MAE

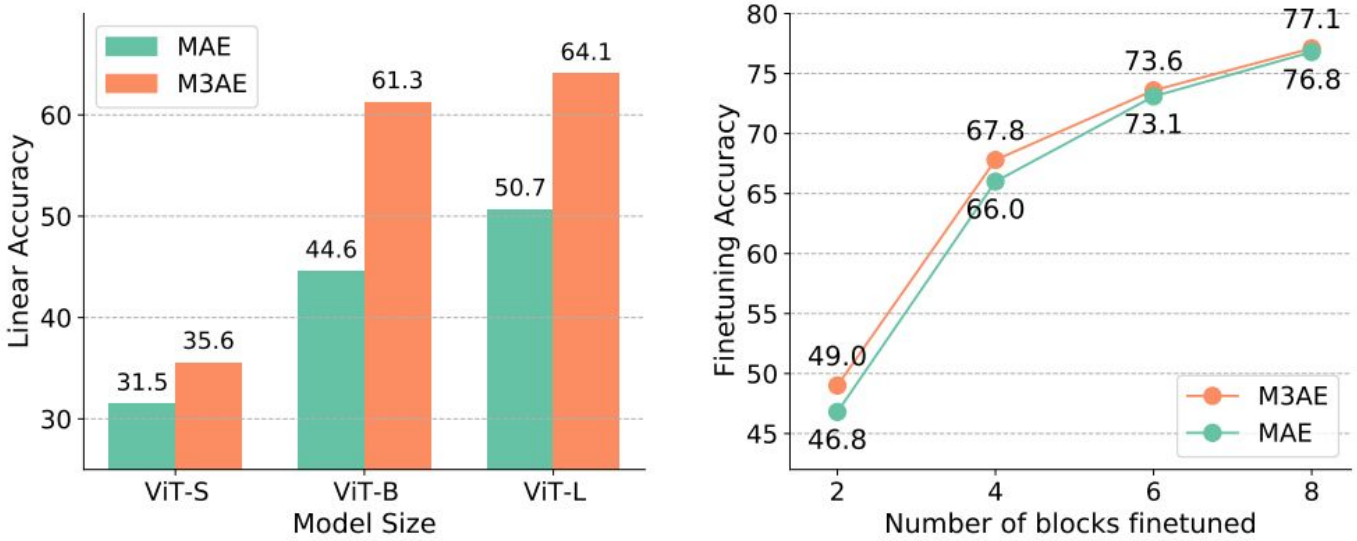


Figure 4: Left: Comparing the linear classification accuracy ViT model variants of different capacities (ViT-S/B/L). All models are pre-trained for 50 epochs. M3AE scales well with model size, outperforming MAE in every setting. **Right**: Comparing finetuning different number of blocks for ViT-L. All models are pre-trained for 50 epochs.

Multiview MWM

Multi-View Masked World Models for Visual Robotic Manipulation

Younggyo Seo [*1] Junsu Kim [*1] Stephen James [2] Kimin Lee [3] Jinwoo Shin [1] Pieter Abbeel [4]

[covered in a later section of lecture]

Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL, DINO/DINOv2, JEPA, I-JEPA, V-JEPA
  - Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control
  - R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

Relative Position of Image Patches

Unsupervised Visual Representation Learning by Context Prediction

Carl Doersch[1,2] Abhinav Gupta[1] Alexei A. Efros[2]

[1] School of Computer Science
Carnegie Mellon University

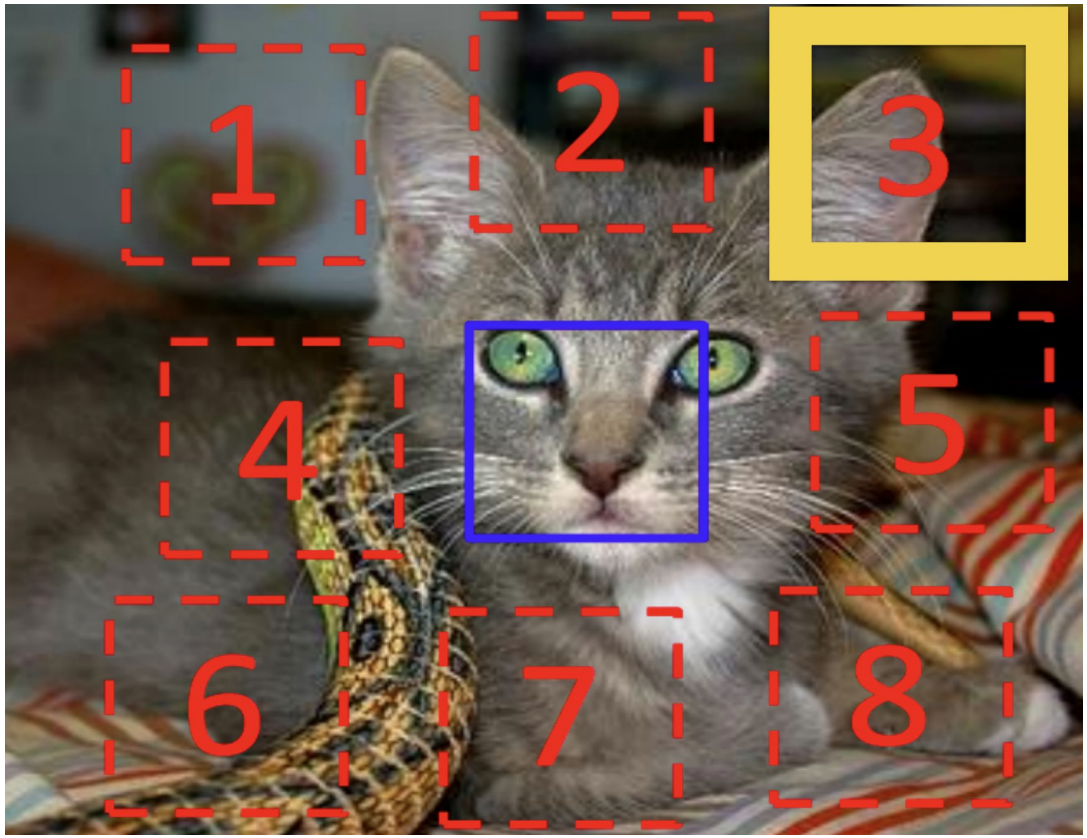
[2] Dept. of Electrical Engineering and Computer Science
University of California, Berkeley



Task: Predict the relative position of the second patch with respect to the first

Slide: Zisserman

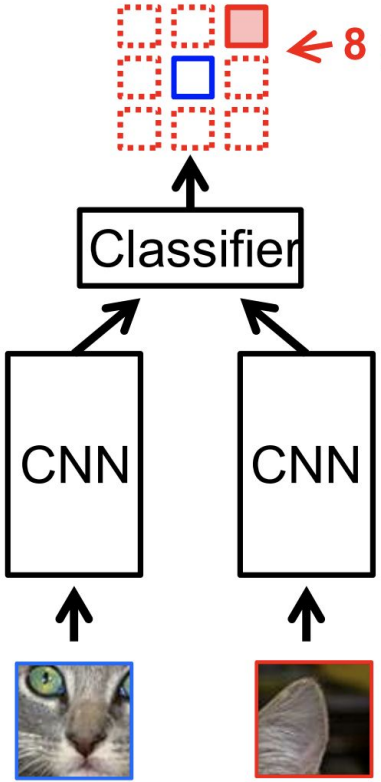
Relative Position of Image Patches

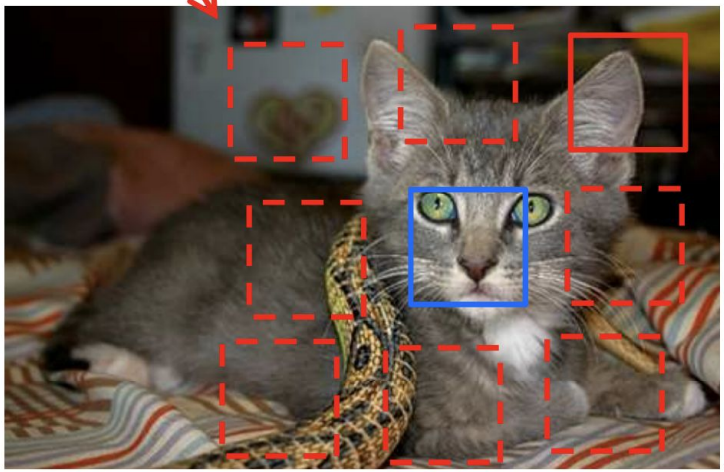


Doersch, Gupta, Efros

Slide: Zisserman

Relative Position of Image Patches

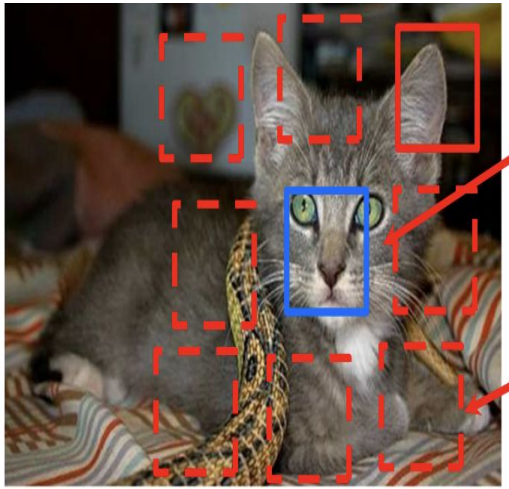




Randomly Sample Patch
Sample Second Patch

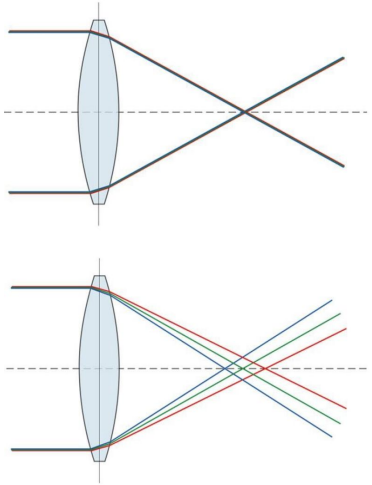
Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

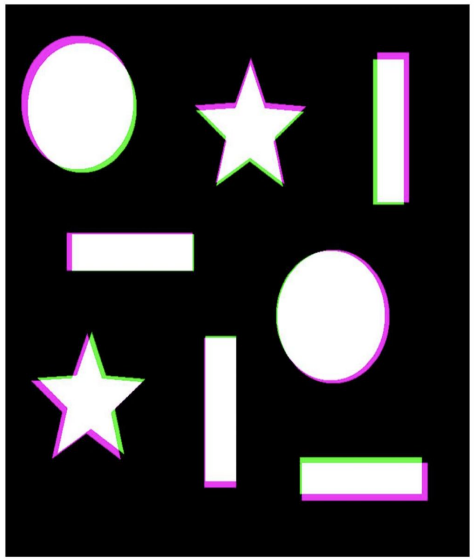
Relative Position of Image Patches



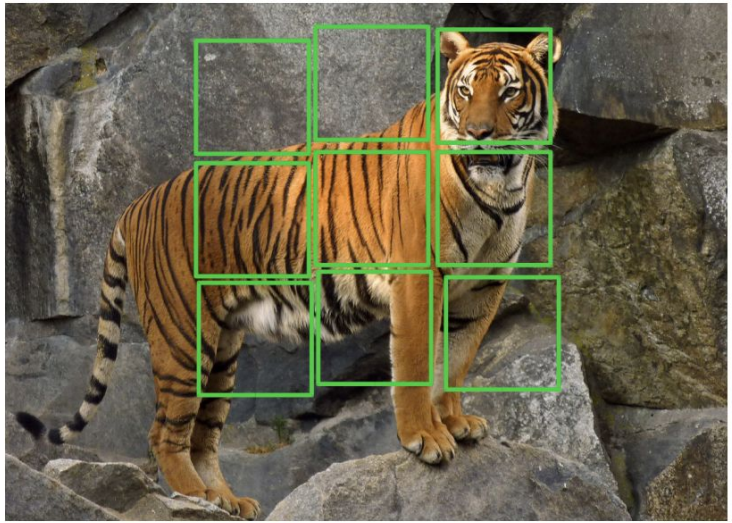
Include a gap

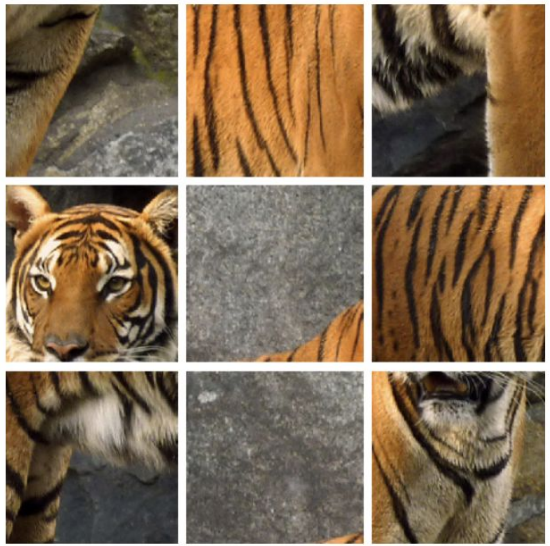
Jitter the patch locations

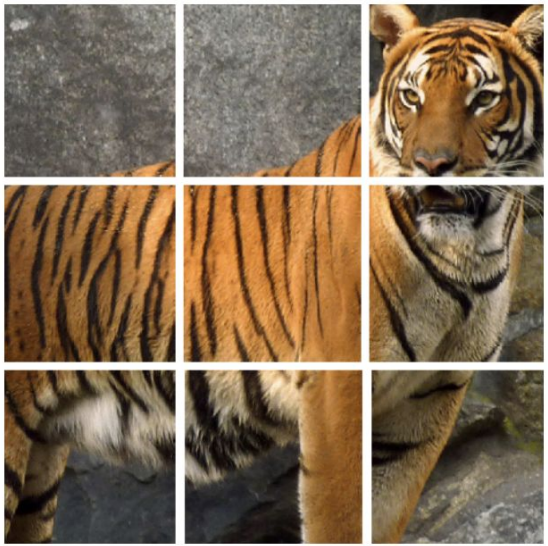




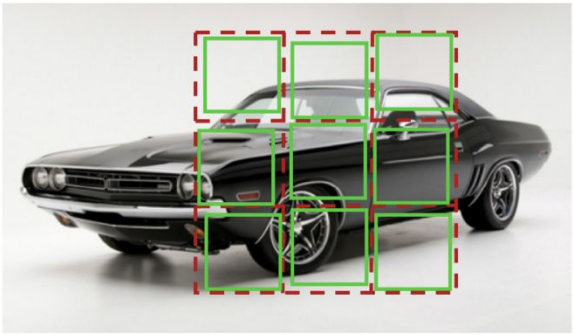
Solving Jigsaw Puzzles







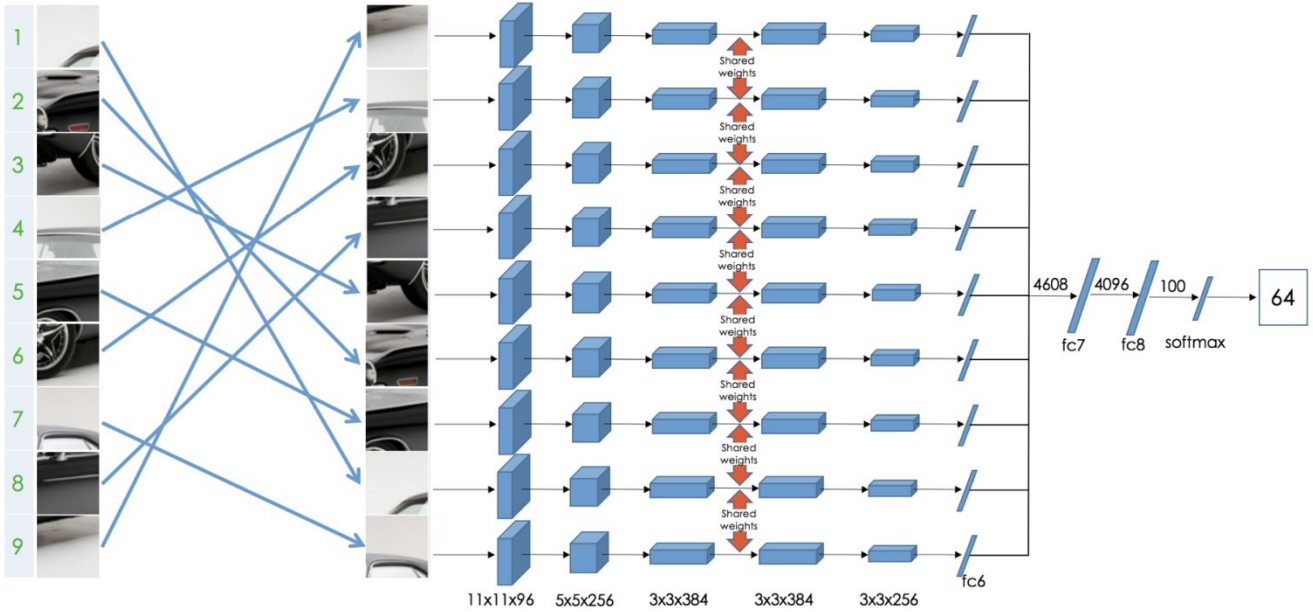
Solving Jigsaw Puzzles



Permutation Set

index	permutation
64	9,4,6,8,3,2,5,1,7

Reorder patches according to the selected permutation



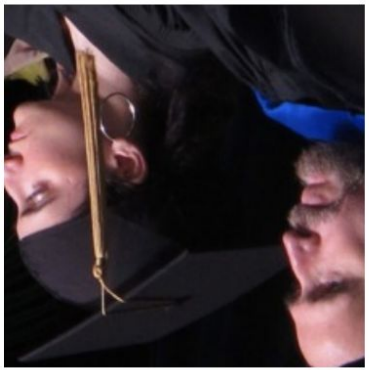
Rotation



90° rotation



270° rotation



180° rotation

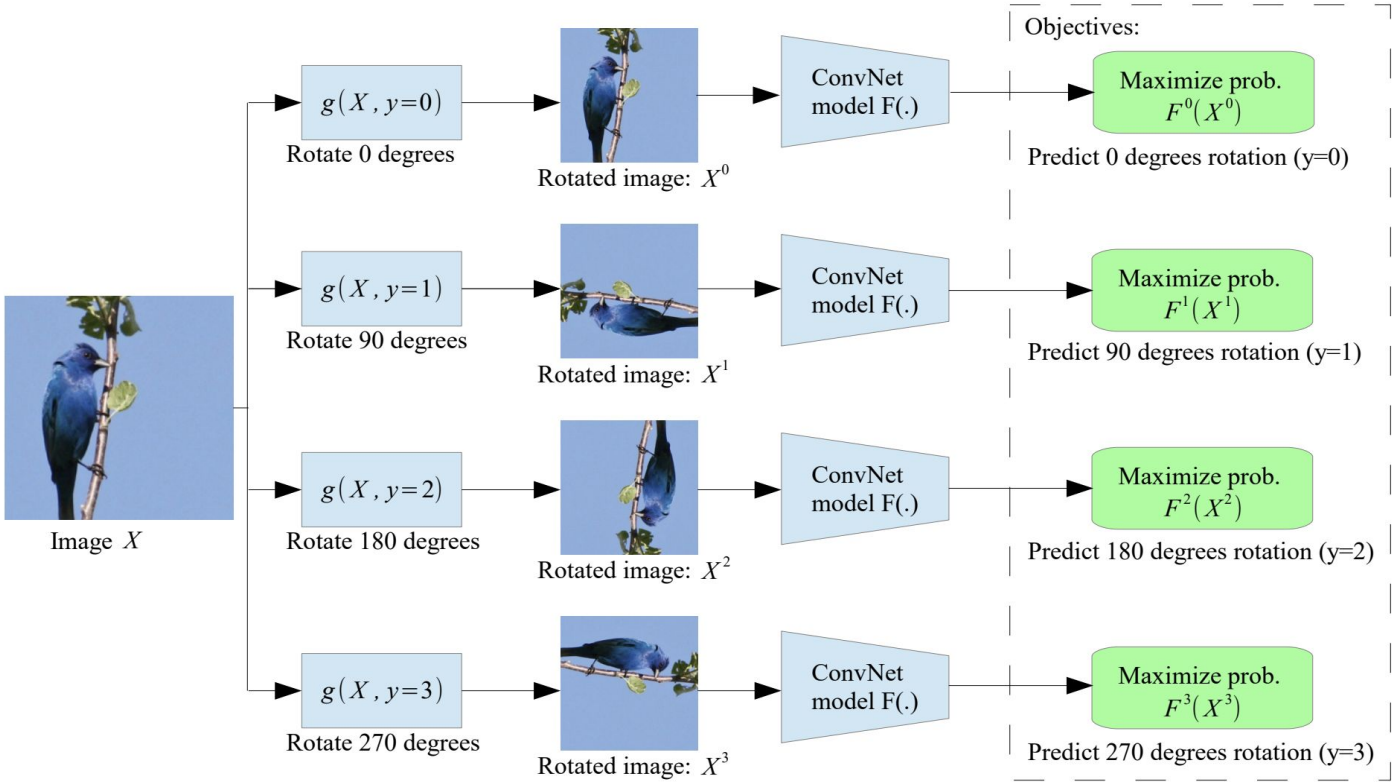


0° rotation



270° rotation

Rotation


Image X
$g(X, y=0)$
Rotate 0 degrees
Rotated image: X^0
$g(X, y=1)$
Rotate 90 degrees
Rotated image: X^1
$g(X, y=2)$
Rotate 180 degrees
Rotated image: X^2
$g(X, y=3)$
Rotate 270 degrees
Rotated image: X^3
ConvNet model F(.)
ConvNet model F(.)
ConvNet model F(.)
ConvNet model F(.)
Objectives:
Maximize prob. $F^0(X^0)$
Predict 0 degrees rotation (y=0)
Maximize prob. $F^1(X^1)$
Predict 90 degrees rotation (y=1)
Maximize prob. $F^2(X^2)$
Predict 180 degrees rotation (y=2)
Maximize prob. $F^3(X^3)$
Predict 270 degrees rotation (y=3)

Rotation

# Rotations	Rotations	CIFAR-10 Classification Accuracy
4	0°, 90°, 180°, 270°	**89.06**
8	0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°	88.51
2	0°, 180°	87.46
2	90°, 270°	85.52

Rotation

Method	Conv4	Conv5
ImageNet labels from (Bojanowski & Joulin, 2017)	59.7	59.7
Random from (Noroozi & Favaro, 2016)	27.1	12.0
Tracking Wang & Gupta (2015)	38.8	29.8
Context (Doersch et al., 2015)	45.6	30.4
Colorization (Zhang et al., 2016a)	40.7	35.2
Jigsaw Puzzles (Noroozi & Favaro, 2016)	45.3	34.6
BIGAN (Donahue et al., 2016)	41.9	32.2
NAT (Bojanowski & Joulin, 2017)	-	36.0
(Ours) RotNet	50.0	43.8

Rotation

Method	Conv1	Conv2	Conv3	Conv4	Conv5
ImageNet labels	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Random rescaled Krähenbühl et al. (2015)	17.5	23.0	24.5	23.2	20.6
Context (Doersch et al., 2015)	16.2	23.3	30.2	31.7	29.6
Context Encoders (Pathak et al., 2016b)	14.1	20.7	21.0	19.8	15.5
Colorization (Zhang et al., 2016a)	12.5	24.5	30.4	31.5	30.3
Jigsaw Puzzles (Noroozi & Favaro, 2016)	18.2	28.8	34.0	33.9	27.1
BIGAN (Donahue et al., 2016)	17.7	24.5	31.0	29.9	28.0
Split-Brain (Zhang et al., 2016b)	17.7	29.3	35.4	35.2	32.8
Counting (Noroozi et al., 2017)	18.0	30.6	34.3	32.5	25.7
(Ours) RotNet	**18.8**	**31.7**	**38.7**	**38.2**	**36.5**

Rotation

	Classification (%mAP)		Detection (%mAP)	Segmentation (%mIoU)
Trained layers	fc6-8	all	all	all
ImageNet labels	78.9	79.9	56.8	48.0
Random		53.3	43.4	19.8
Random rescaled Krähenbühl et al. (2015)	39.2	56.6	45.6	32.6
Egomotion (Agrawal et al., 2015)	31.0	54.2	43.9	
Context Encoders (Pathak et al., 2016b)	34.6	56.5	44.5	29.7
Tracking (Wang & Gupta, 2015)	55.6	63.1	47.4	
Context (Doersch et al., 2015)	55.1	65.3	51.1	
Colorization (Zhang et al., 2016a)	61.5	65.6	46.9	35.6
BIGAN (Donahue et al., 2016)	52.3	60.1	46.9	34.9
Jigsaw Puzzles (Noroozi & Favaro, 2016)	-	67.6	53.2	37.6
NAT (Bojanowski & Joulin, 2017)	56.7	65.3	49.4	
Split-Brain (Zhang et al., 2016b)	63.0	67.1	46.7	36.0
ColorProxy (Larsson et al., 2017)		65.9		38.4
Counting (Noroozi et al., 2017)	-	67.7	51.4	36.6
(Ours) RotNet	**70.87**	**72.97**	**54.4**	**39.1**

Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

Contrastive Predictive Coding

Representation Learning with Contrastive Predictive Coding

Aaron van den Oord
DeepMind
avdnoord@google.com

Yazhe Li
DeepMind
yazhe@google.com

Oriol Vinyals
DeepMind
vinyals@google.com

Abstract

While supervised learning has enabled great progress in many applications, unsupervised learning has not seen such widespread adoption, and remains an important and challenging endeavor for artificial intelligence. In this work, we propose a universal unsupervised learning approach to extract useful representations from high-dimensional data, which we call Contrastive Predictive Coding. The key insight of our model is to learn such representations by predicting the future in *latent* space by using powerful autoregressive models. We use a probabilistic contrastive loss which induces the latent space to capture information that is maximally useful to predict future samples. It also makes the model tractable by using negative sampling. While most prior work has focused on evaluating representations for a particular modality, we demonstrate that our approach is able to learn useful representations achieving strong performance on four distinct domains: speech, images, text and reinforcement learning in 3D environments.

July 2018

Contrastive Predictive Coding

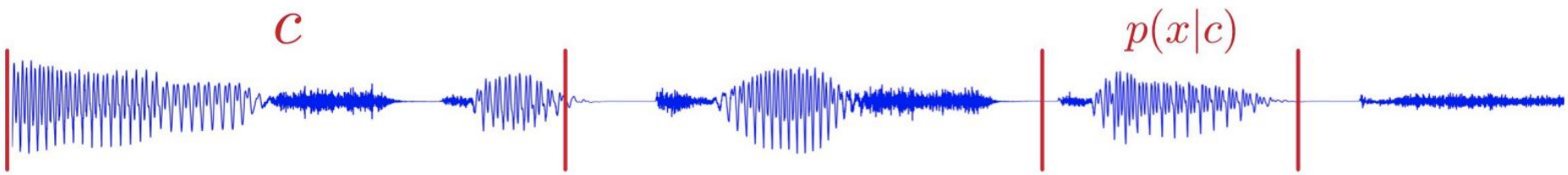


Figure from Alex Graves

Contrastive Predictive Coding

Don't directly predict x



Figure from Alex Graves

Contrastive Predictive Coding

$$\max_{\theta} MI(c; z)$$

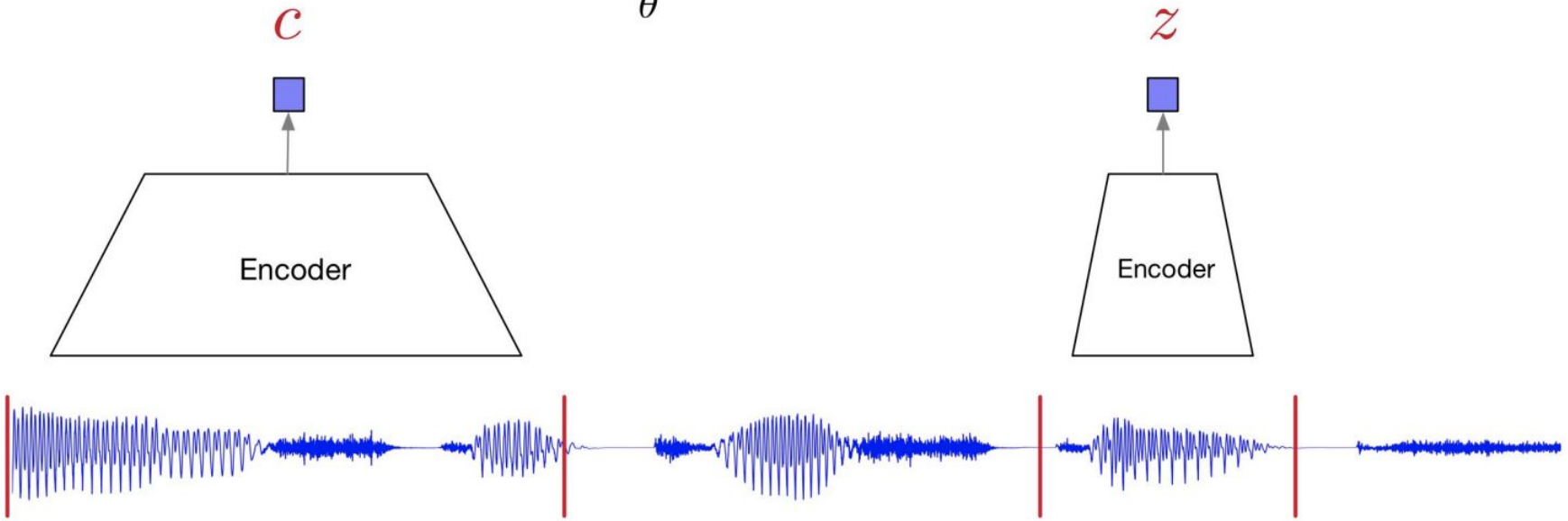


Figure from Alex Graves

Contrastive Predictive Coding

$$\frac{\exp f(c, z_i)}{\sum_j \exp f(c, z_j)}$$

Bilinear dot product

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

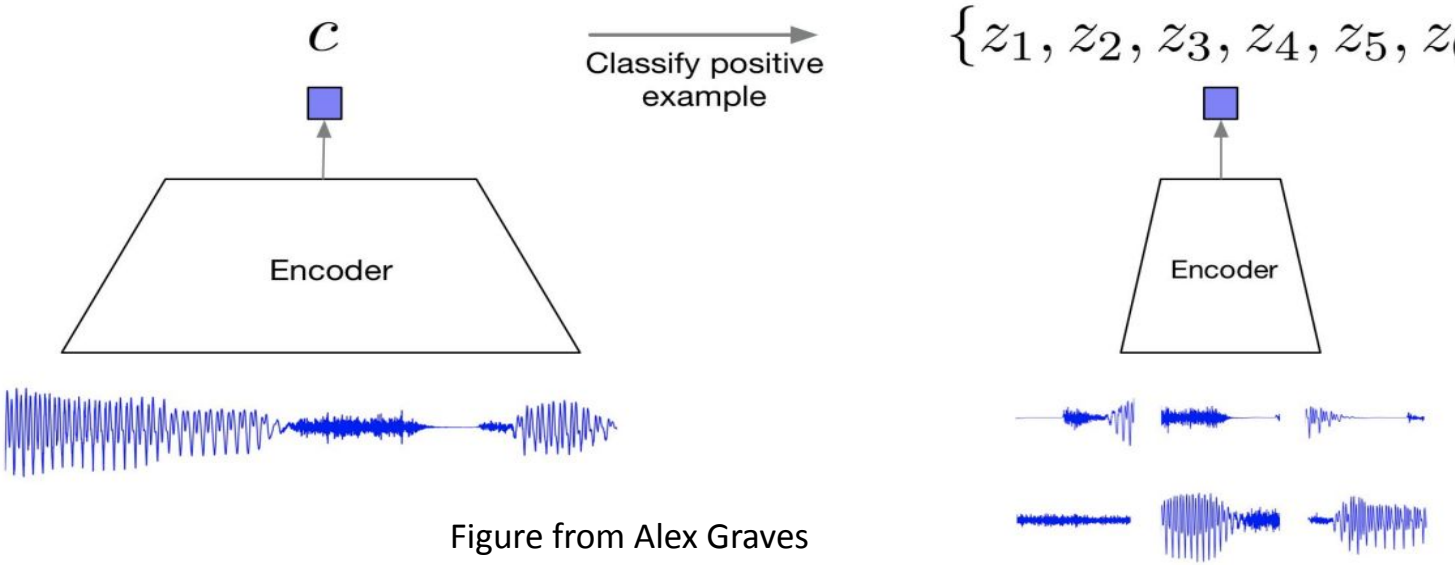


Figure from Alex Graves

Contrastive Predictive Coding

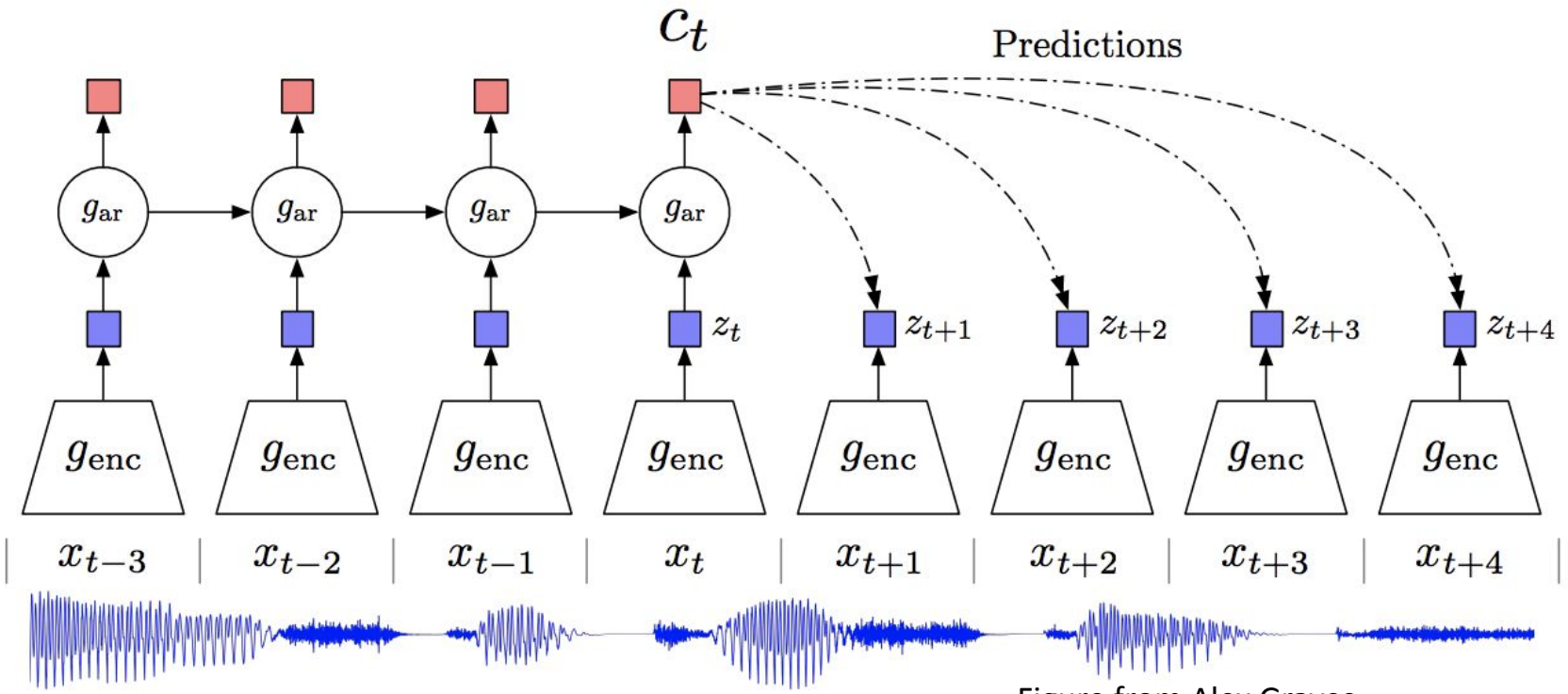
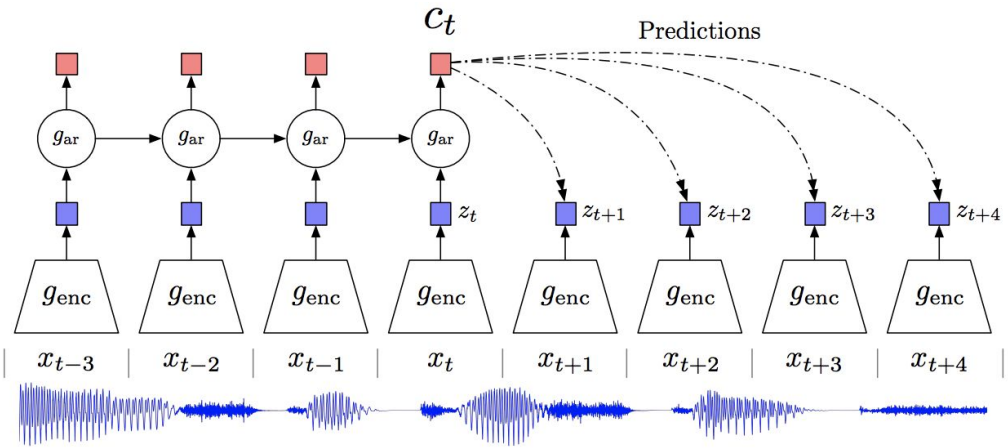


Figure from Alex Graves

Contrastive Predictive Coding



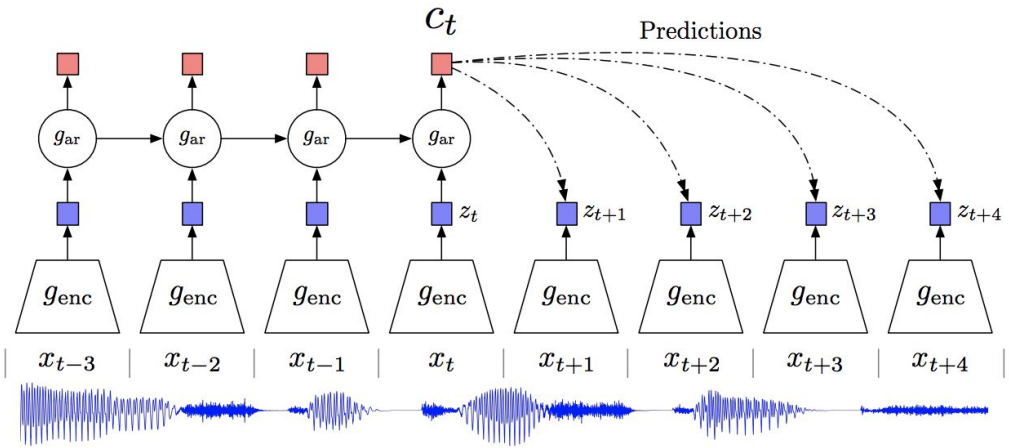
InfoNCE

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

Figure from Alex Graves

Contrastive Predictive Coding



InfoNCE

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

Can be viewed as categorical cross-entropy of classifying the positive sample correctly

Figure from Alex Graves

Contrastive Predictive Coding

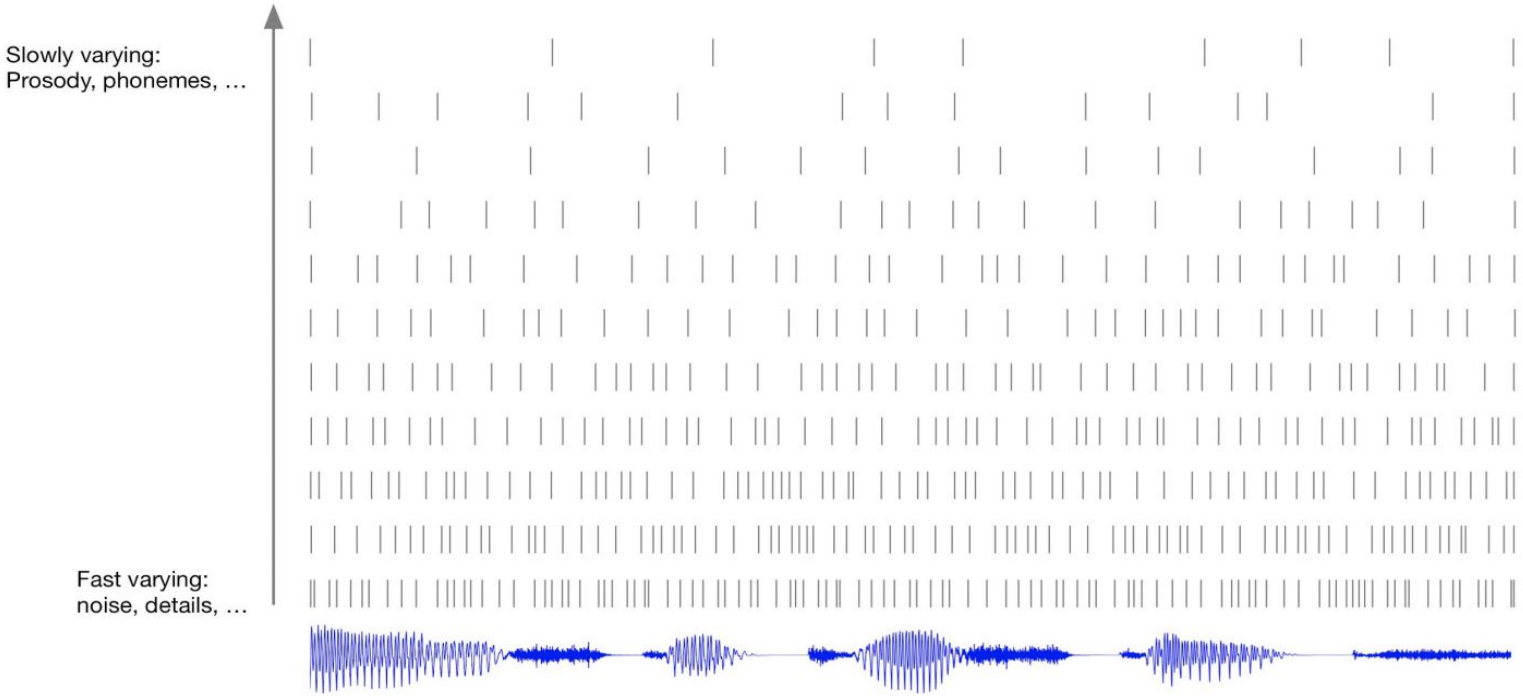


Figure from Alex Graves

Contrastive Predictive Coding

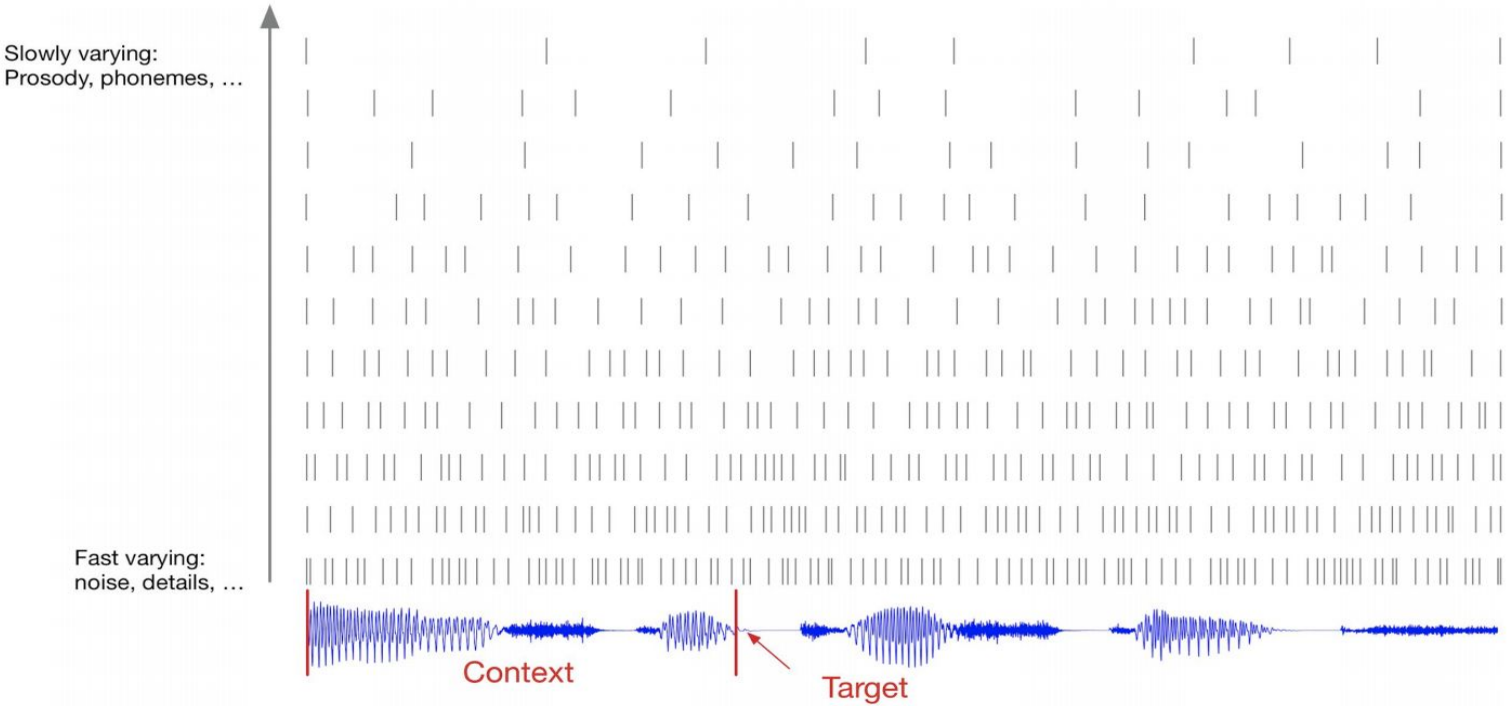


Figure from Alex Graves

Contrastive Predictive Coding

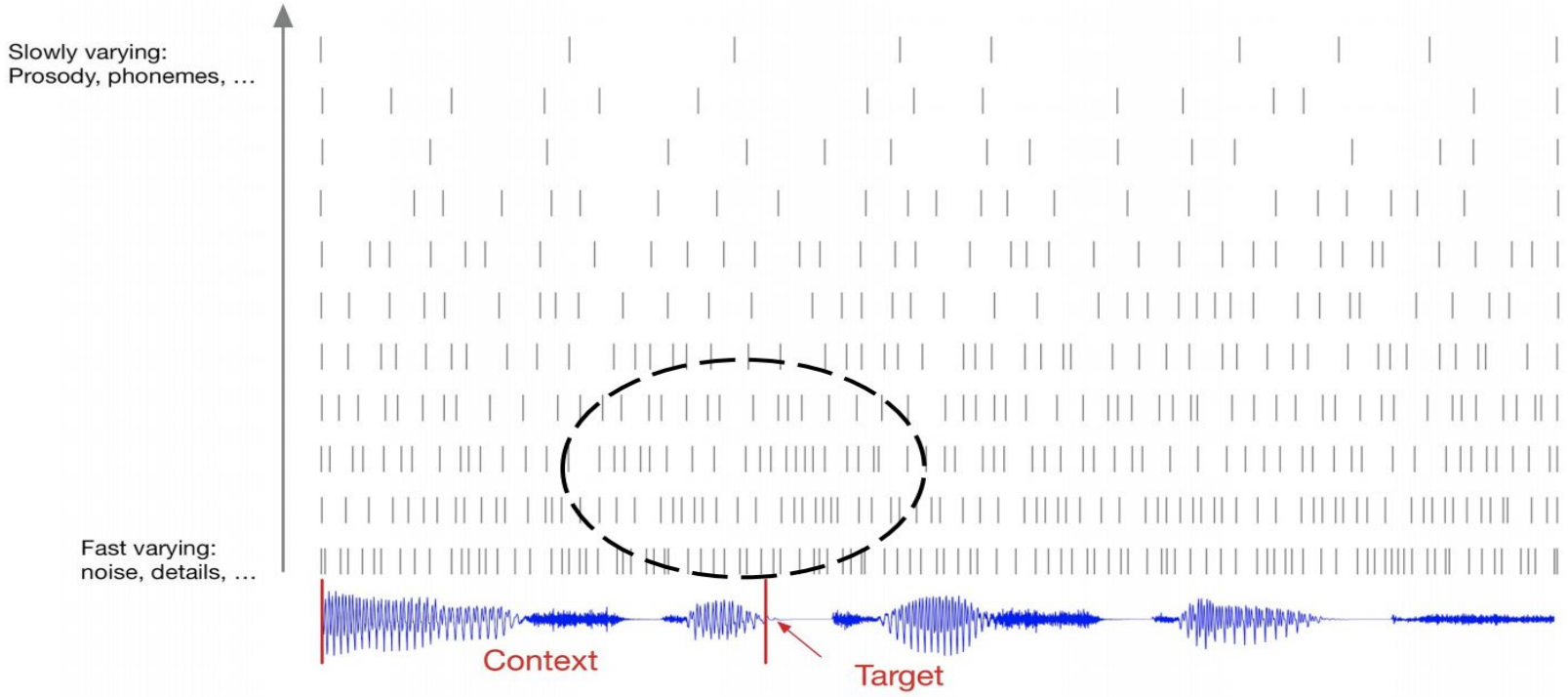


Figure from Alex Graves

Contrastive Predictive Coding

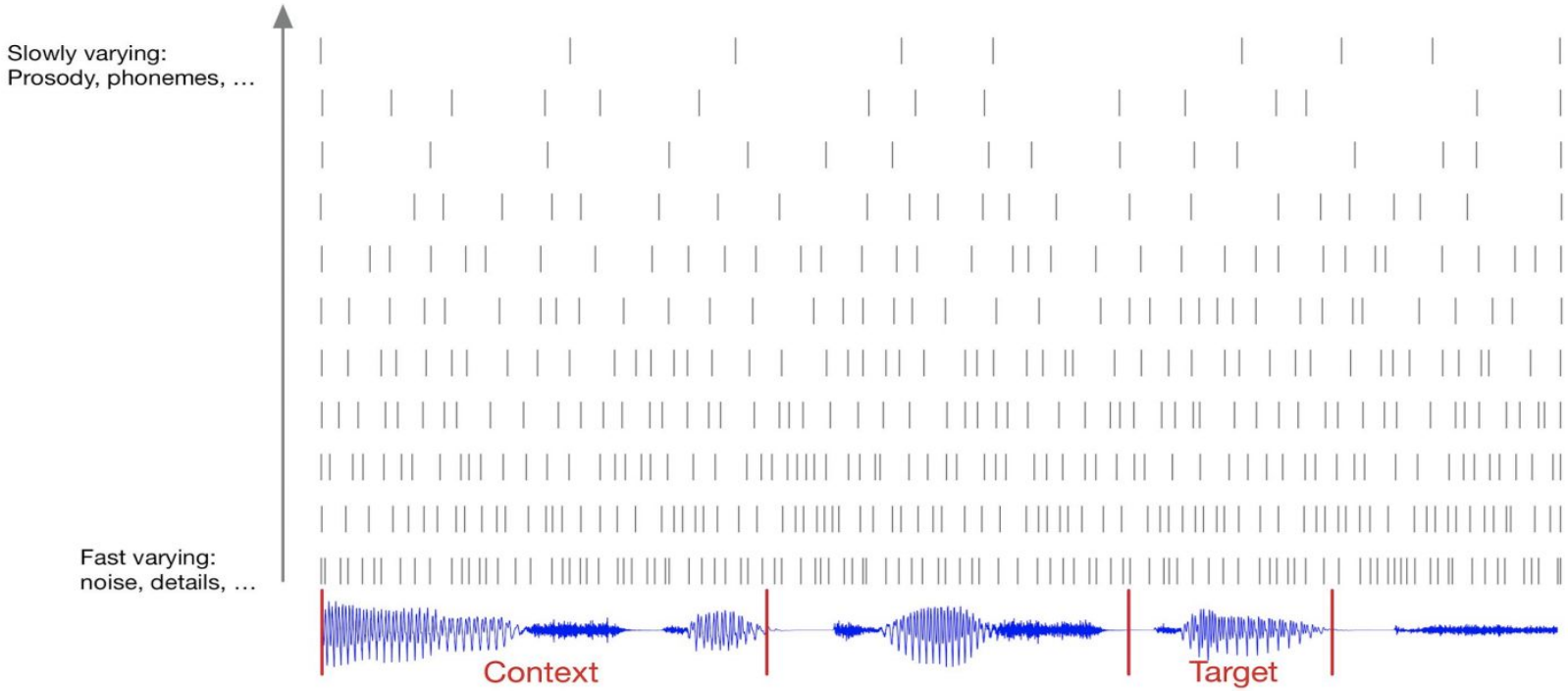


Figure from Alex Graves

Contrastive Predictive Coding

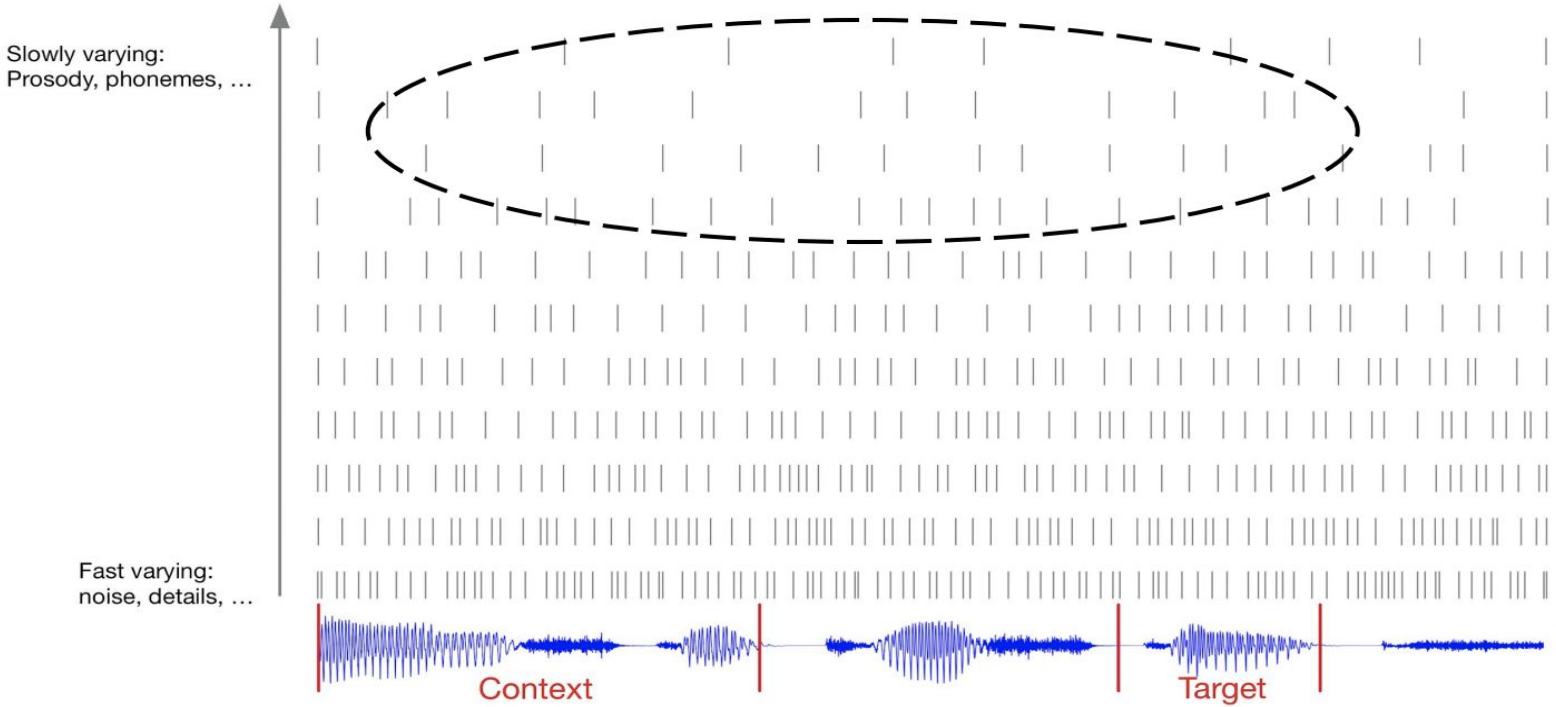


Figure from Alex Graves

CPC - Speech

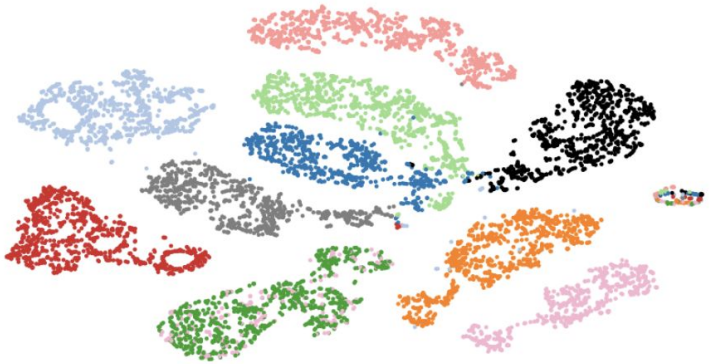


Figure 2: t-SNE visualization of audio (speech) representations for a subset of 10 speakers (out of 251). Every color represents a different speaker.

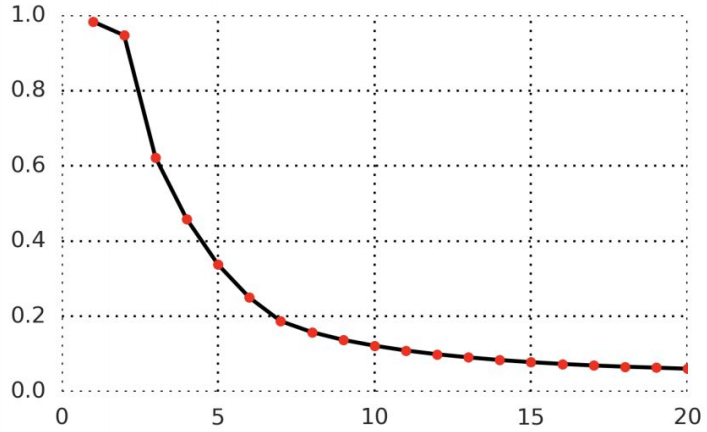


Figure 3: Average accuracy of predicting the positive sample in the contrastive loss for 1 to 20 latent steps in the future of a speech waveform. The model predicts up to 200ms in the future as every step consists of 10ms of audio.

CPC - Speech

Method	ACC
Phone classification	
Random initialization	27.6
MFCC features	39.7
CPC	64.6
Supervised	74.6
Speaker classification	
Random initialization	1.87
MFCC features	17.6
CPC	97.4
Supervised	98.5

Table 1: LibriSpeech phone and speaker classification results. For phone classification there are 41 possible classes and for speaker classification 251. All models used the same architecture and the same audio input sizes.

Method	ACC
#steps predicted	
2 steps	28.5
4 steps	57.6
8 steps	63.6
12 steps	64.6
16 steps	63.8
Negative samples from	
Mixed speaker	64.6
Same speaker	65.5
Mixed speaker (excl.)	57.3
Same speaker (excl.)	64.6
Current sequence only	65.2

Table 2: LibriSpeech phone classification ablation experiments. More details can be found in Section 3.1.

CPC - ImageNet

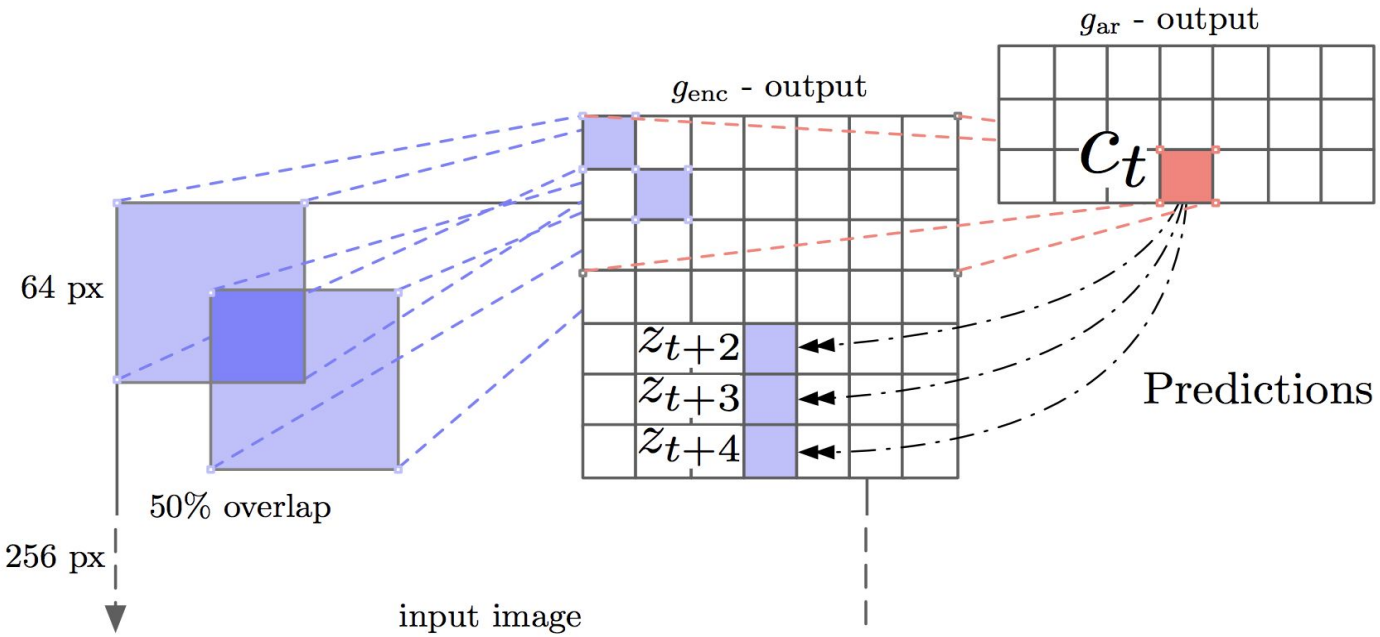


Figure 4: Visualization of Contrastive Predictive Coding for images (2D adaptation of Figure 1).

CPC - ImageNet





CPC - ImageNet

Method	Top-1 ACC
Using AlexNet conv5	
Video [28]	29.8
Relative Position [11]	30.4
BiGan [35]	34.8
Colorization [10]	35.2
Jigsaw [29] *	38.1
Using ResNet-V2	
Motion Segmentation [36]	27.6
Exemplar [36]	31.5
Relative Position [36]	36.2
Colorization [36]	39.6
CPC	**48.7**

Table 3: ImageNet top-1 unsupervised classification results. *Jigsaw is not directly comparable to the other AlexNet results because of architectural differences.

Method	Top-5 ACC
Motion Segmentation (MS)	48.3
Exemplar (Ex)	53.1
Relative Position (RP)	59.2
Colorization (Col)	62.5
Combination of MS + Ex + RP + Col	69.3
CPC	**73.6**

Table 4: ImageNet top-5 unsupervised classification results. Previous results with MS, Ex, RP and Col were taken from [36] and are the best reported results on this task.

CPC - ImageNet

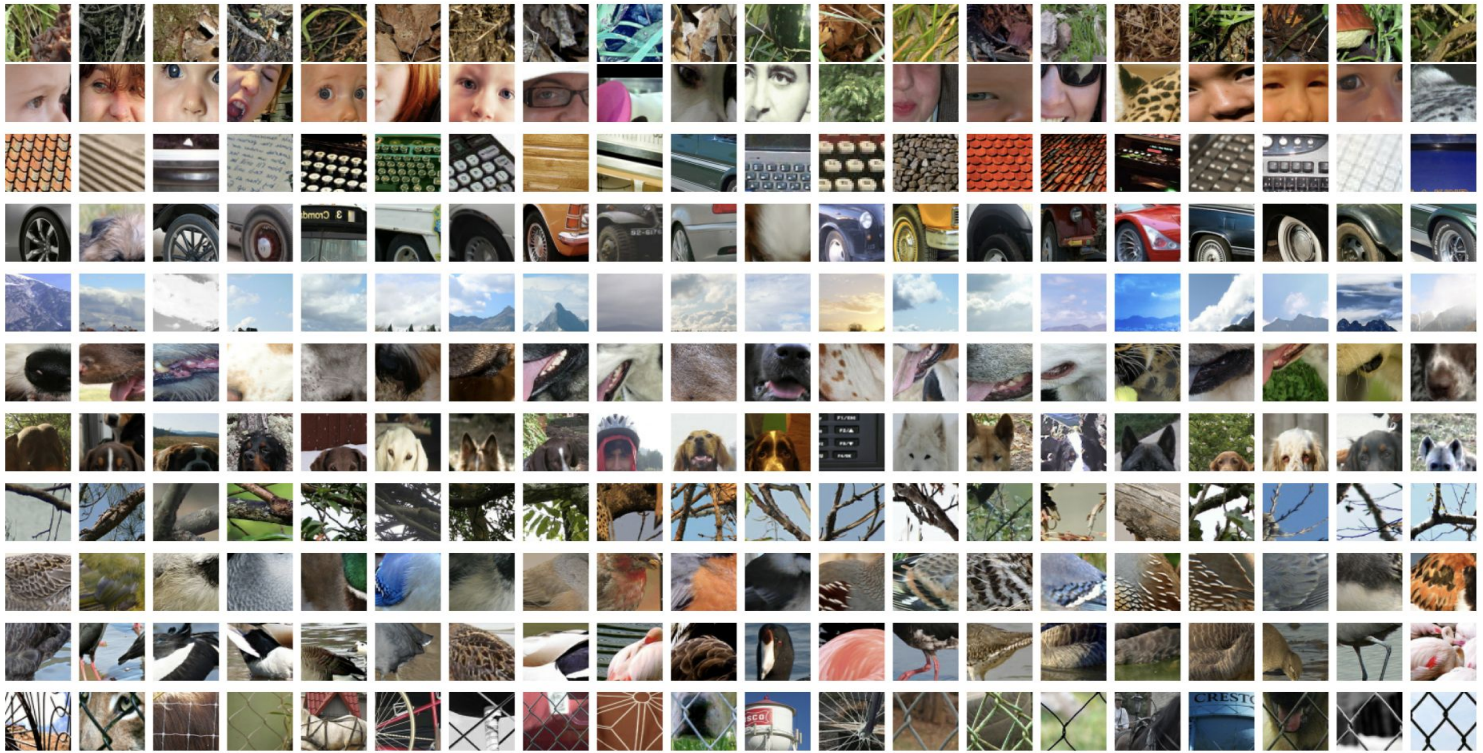


Figure 5: Every row shows image patches that activate a certain neuron in the CPC architecture.

CPC - Natural Language Processing

Method	MR	CR	Subj	MPQA	TREC
Paragraph-vector [40]	74.8	78.1	90.5	74.2	91.8
Skip-thought vector [26]	75.5	79.3	92.1	86.9	91.4
Skip-thought + LN [41]	79.5	82.6	93.4	89.0	-
CPC	76.9	80.1	91.2	87.7	96.8

Table 5: Classification accuracy on five common NLP benchmarks. We follow the same transfer learning setup from Skip-thought vectors [26] and use the BookCorpus dataset as source. [40] is an unsupervised approach to learning sentence-level representations. [26] is an alternative unsupervised learning approach. [41] is the same skip-thought model with layer normalization trained for 1M iterations.

Oord, Li, Vinyals 2018

CPC - Reinforcement Learning

Auxiliary loss is on policy
Predict 30 steps in the future

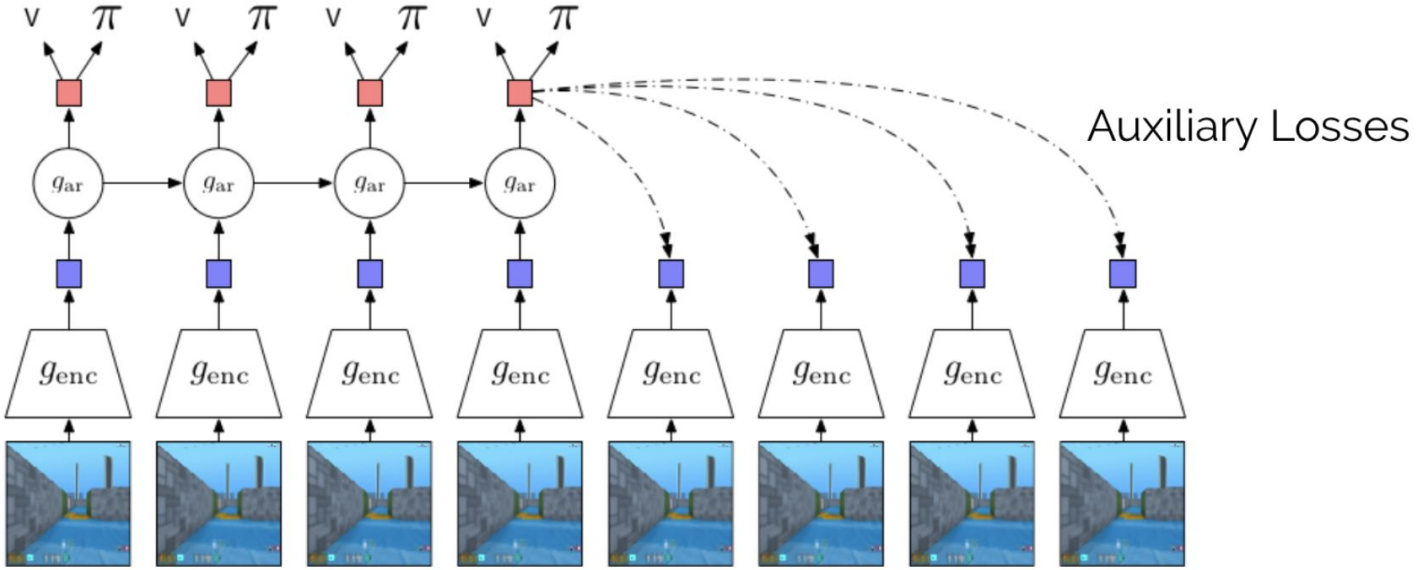


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

Data-Efficient Image Recognition with Contrastive Predictive Coding

Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S. M. Ali Eslami, Aaron van den Oord
DeepMind
London, UK

Abstract

Human observers can learn to recognize new categories of images from a handful of examples, yet doing so with machine perception remains an open challenge. We hypothesize that data-efficient recognition is enabled by representations which make the variability in natural signals more predictable. We therefore revisit and improve Contrastive Predictive Coding, an unsupervised objective for learning such representations. This new implementation produces features which support state-of-the-art linear classification accuracy on the ImageNet dataset. When used as input for non-linear classification with deep neural networks, this representation allows us to use 2–5× less labels than classifiers trained directly on image pixels. Finally, this unsupervised representation substantially improves transfer learning to object detection on PASCAL VOC-2007, surpassing fully supervised pre-trained ImageNet classifiers.

May 2019

CPCv2 - Large Scale CPC on ImageNet



Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

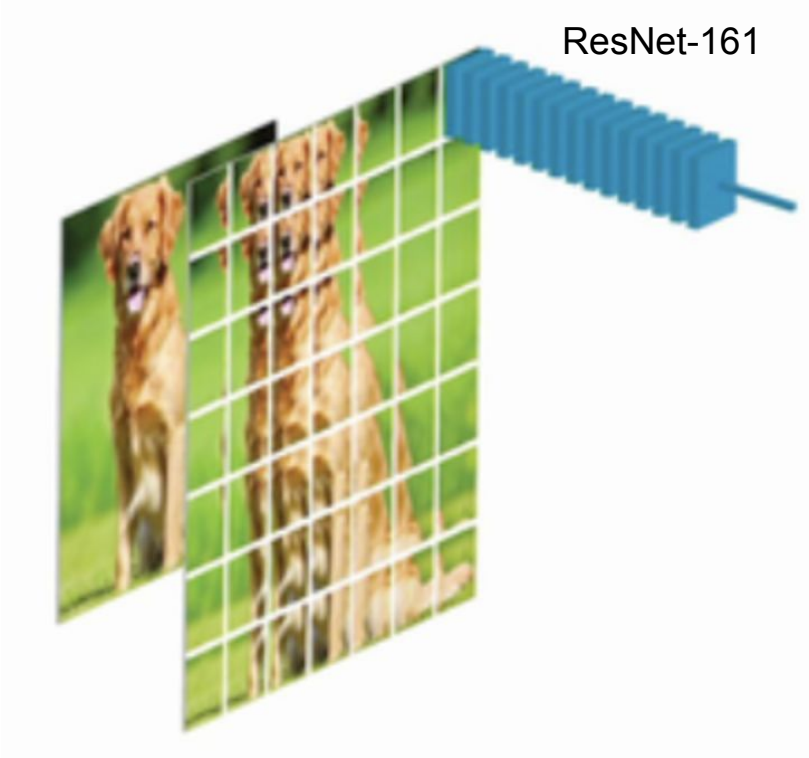


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

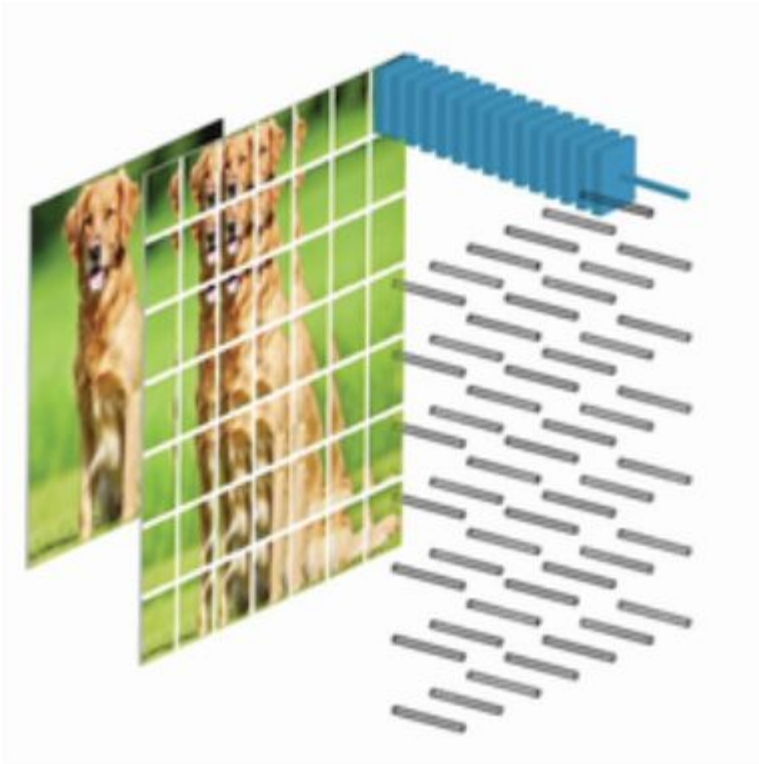


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

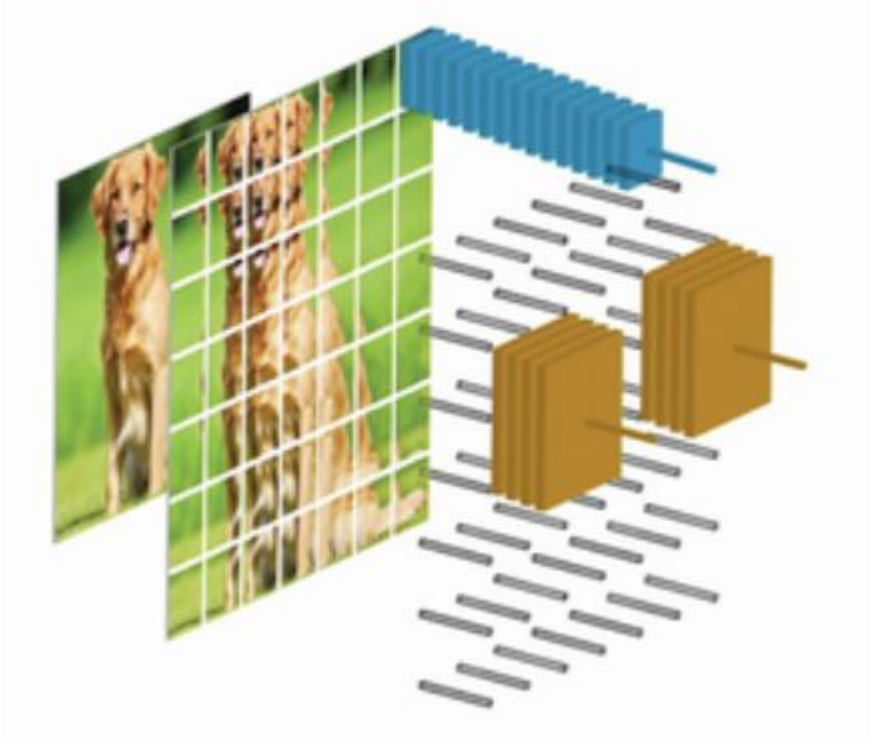


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

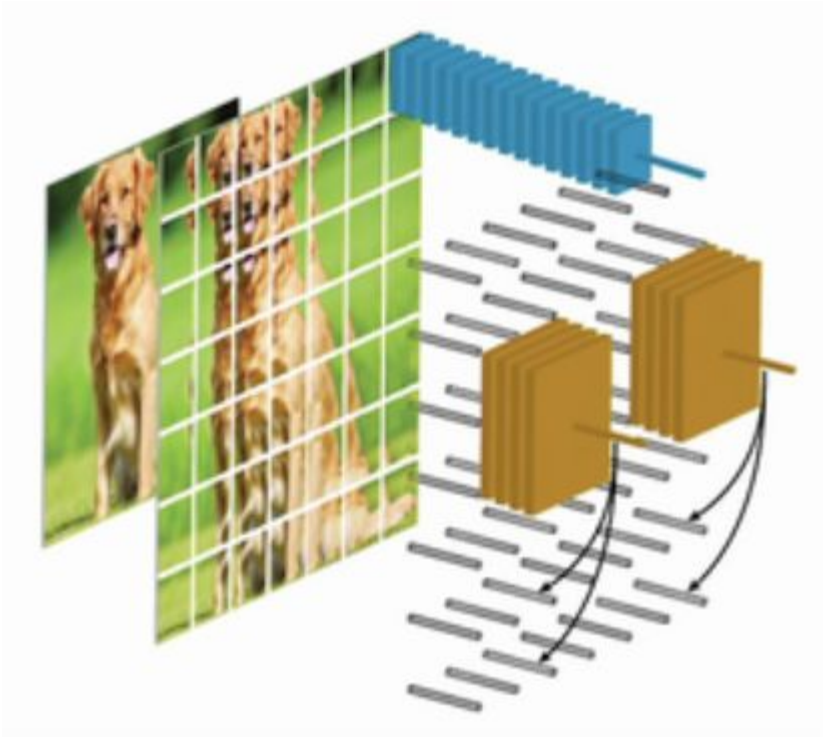


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

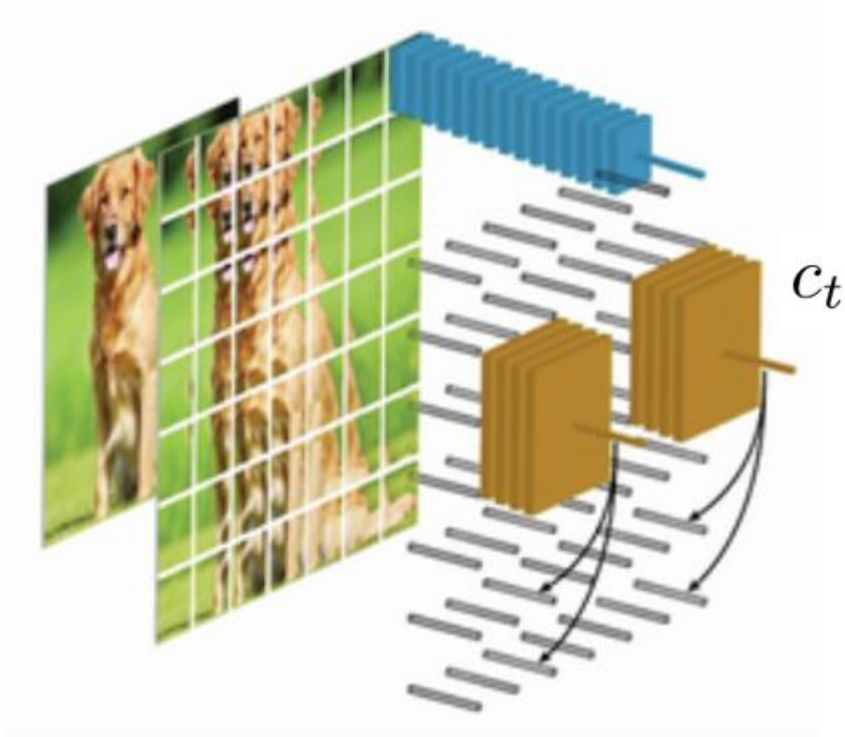


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

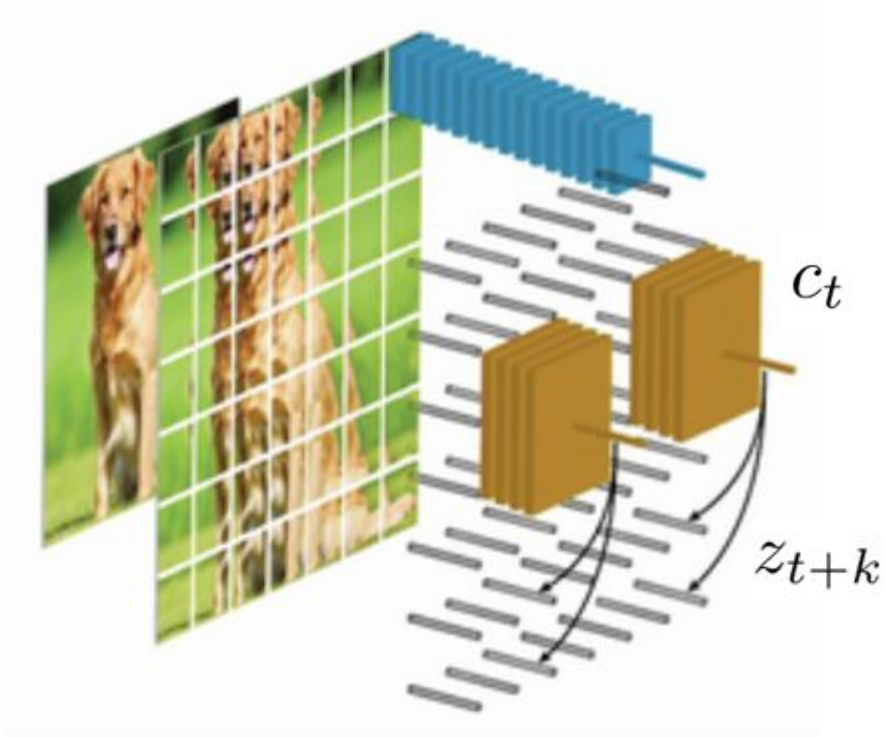
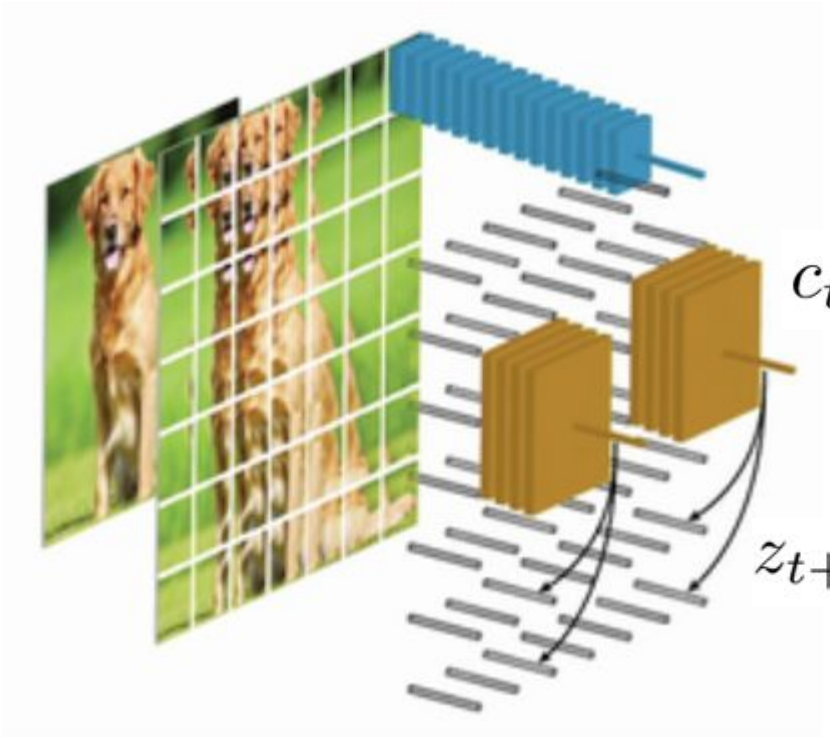


Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet



$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

Figure from Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

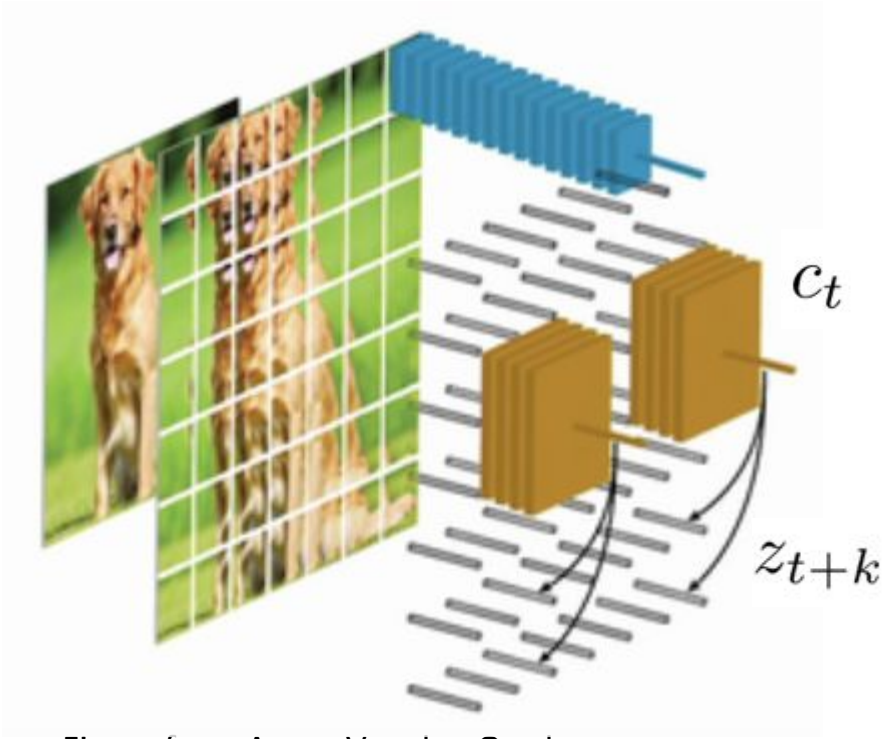


Figure from Aaron Van den Oord

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_X\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

CPCv2 - Large Scale CPC on ImageNet

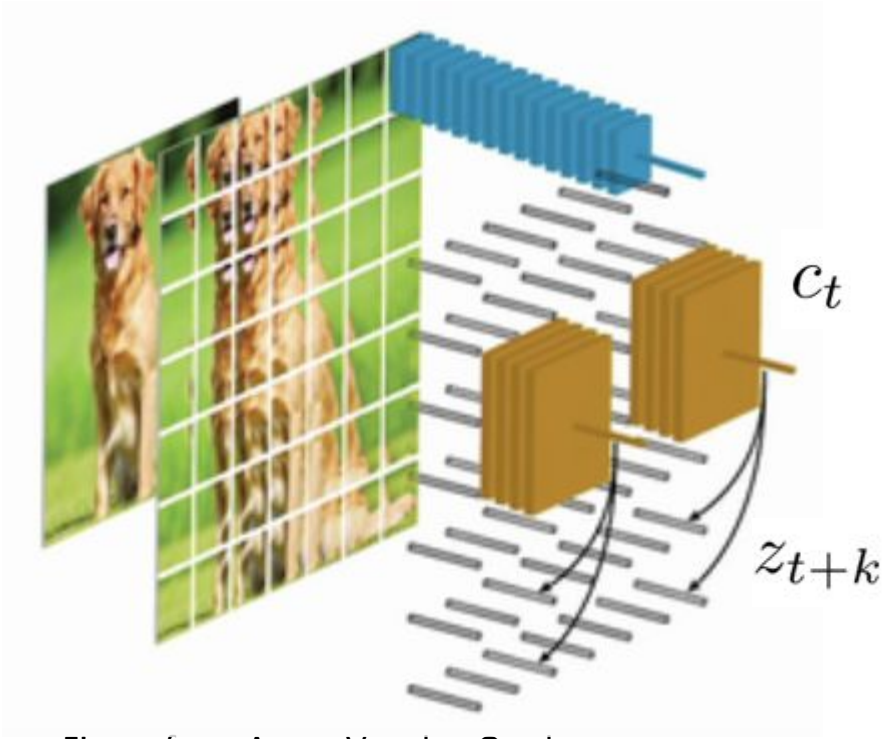


Figure from Aaron Van den Oord

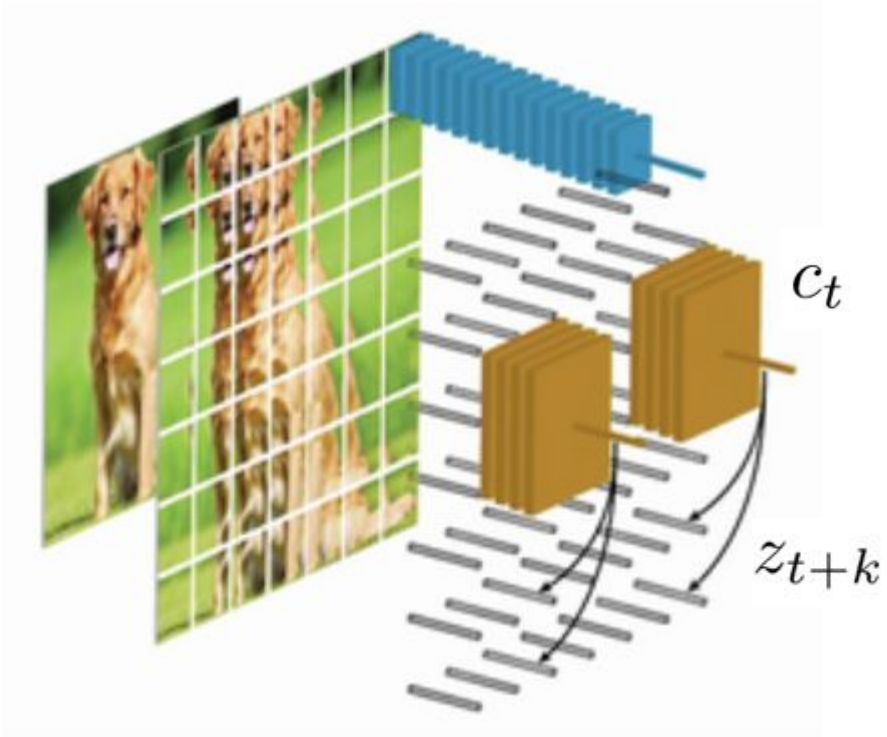
$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_X\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Negatives

1. Other patches within image
2. Patches from other images

CPCv2 - Large Scale CPC on ImageNet



InfoNCE Loss

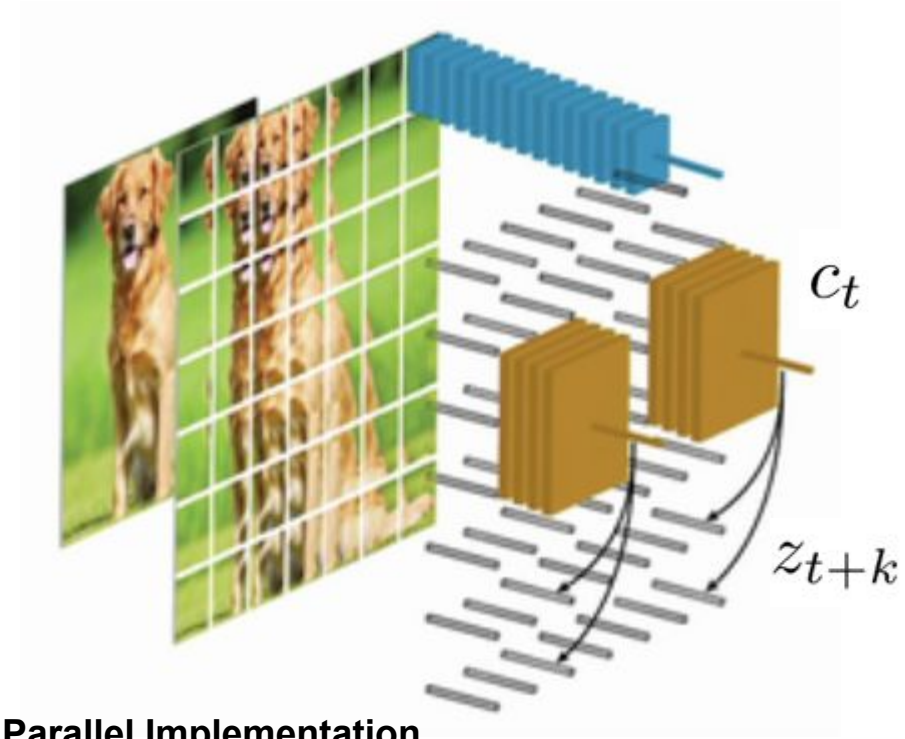
$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_X\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Negatives

1. Other patches within image
2. Patches from other images

CPCv2 - Large Scale CPC on ImageNet



Parallel Implementation with PixelCNN (masked conv) and 1x1 conv

InfoNCE Loss

$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_X\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Negatives

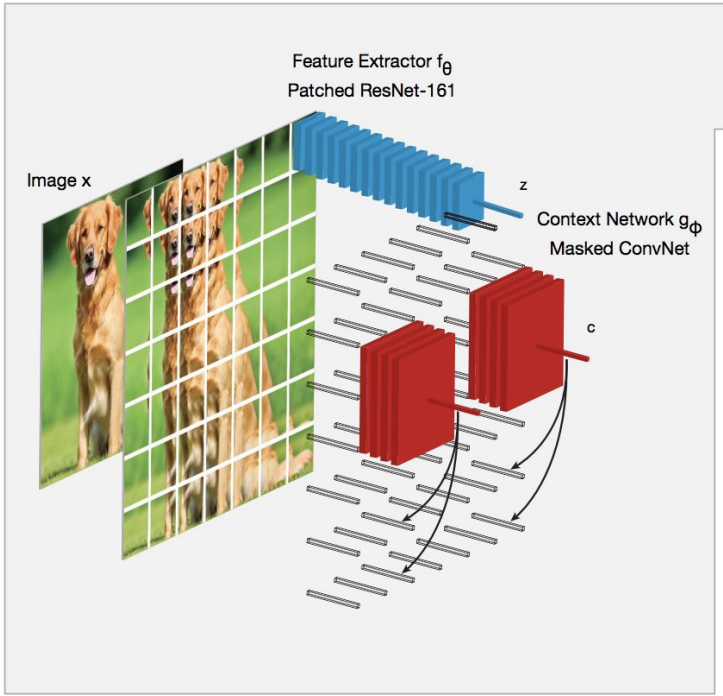
1. Other patches within image
2. Patches from other images

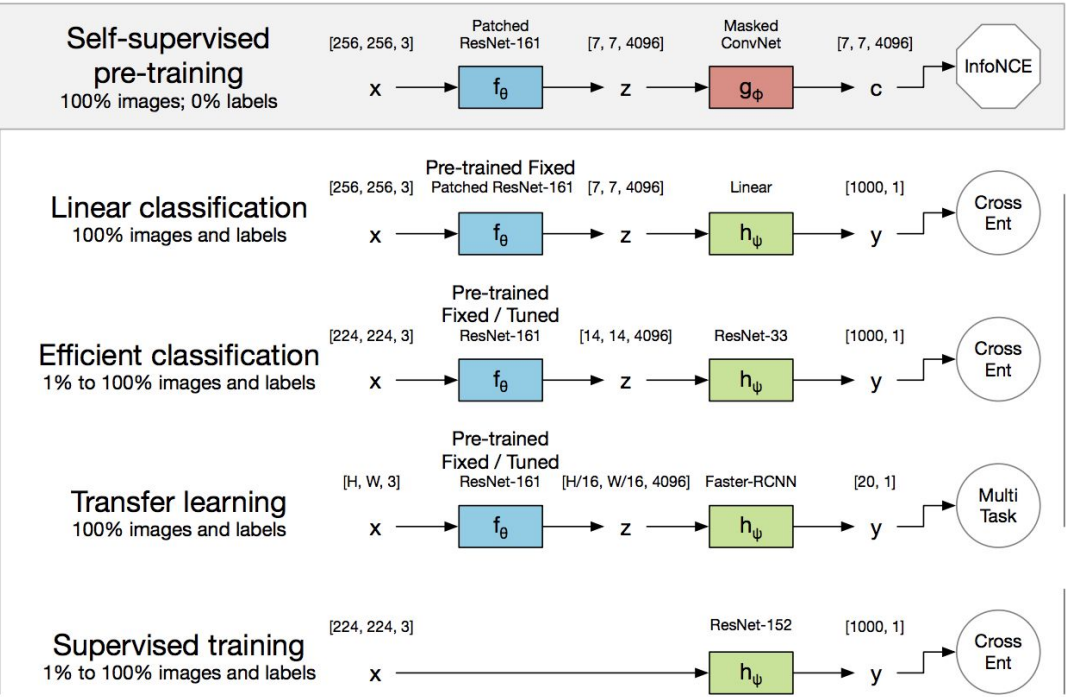
CPCv2 - Large Scale CPC on ImageNet

Contrastive Predictive Coding 2.0 (CPCv2)

- Train CPC on unlabeled ImageNet
- Train as long as possible (500 epochs) - 1 week
- Augment every patch with a lot of spatial and color augmentation [**extremely crucial**]
- Effective number of negatives = number of instances * number of patches per instance = 16 * 36 = 576

CPCv2 - Large Scale CPC on ImageNet





CPCv2 - Linear Classification

Linear Classifier Score (Imagenet)

Method	Architecture	Params. (M)	Top-1 Acc.
Motion Segmentation	ResNet-101	28	27.6
Exemplar	ResNet-101	28	31.5
Relative Position	ResNet-101	28	36.2
Colorization	ResNet-101	28	39.6
CPC v1	ResNet-101	28	48.7
Rotation	RevNet-50 ×4	86	55.4
BigBiGAN	RevNet-50 ×4	86	61.3
AMDIM	Custom-103	626	68.1
CMC	ResNet-50 ×2	188	68.4
Momentum Contrast	ResNet-50 ×4	375	68.6
CPC v2	ResNet-161	305	**71.5**
Local Aggregation	ResNet-50	24	60.2
Momentum Contrast	ResNet-50	24	60.6
CPC v2	ResNet-50	24	**63.8**

CPCv1 ---> CPCv2

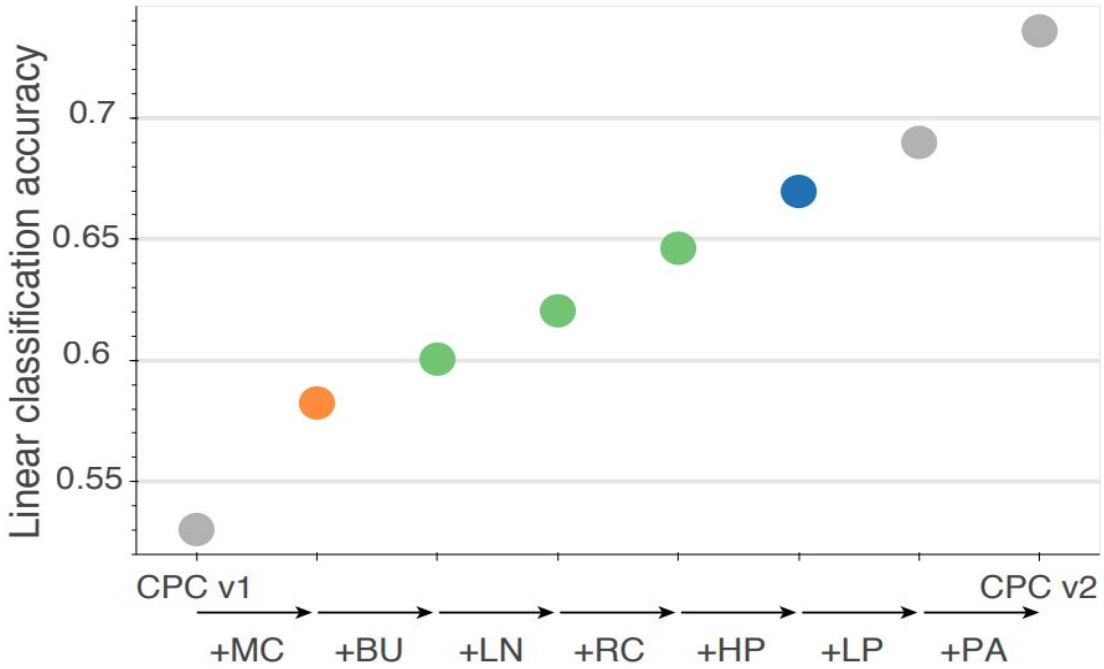
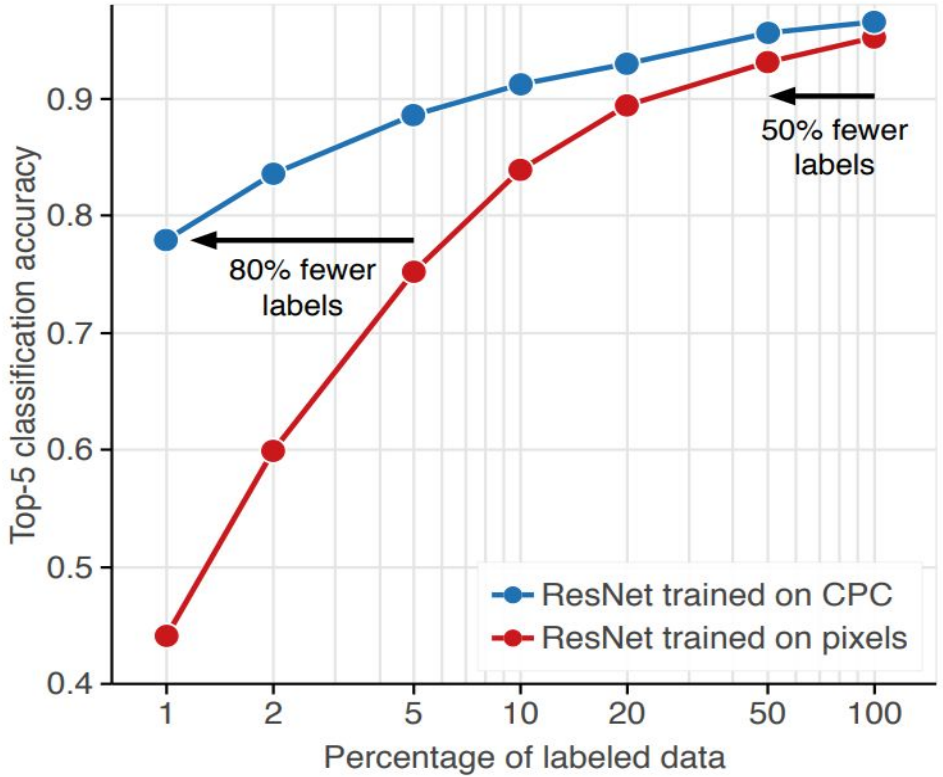


Figure 3. Linear classification performance of new variants of CPC, which incrementally add a series of modifications. MC: model capacity. BU: bottom-up spatial predictions. LN: layer normalization. RC: random color-dropping. HP: horizontal spatial predictions. LP: larger patches. PA: further patch-based augmentation. Note that these accuracies are evaluated on a custom validation set and are therefore not directly comparable to the results we report on the official validation set.

CPCv2 - Data-Efficient Image Recognition



Instance Discrimination







attract

repel

Instance Discrimination







attract

repel

1. MoCo
2. SimCLR

Momentum Contrast (MoCo)

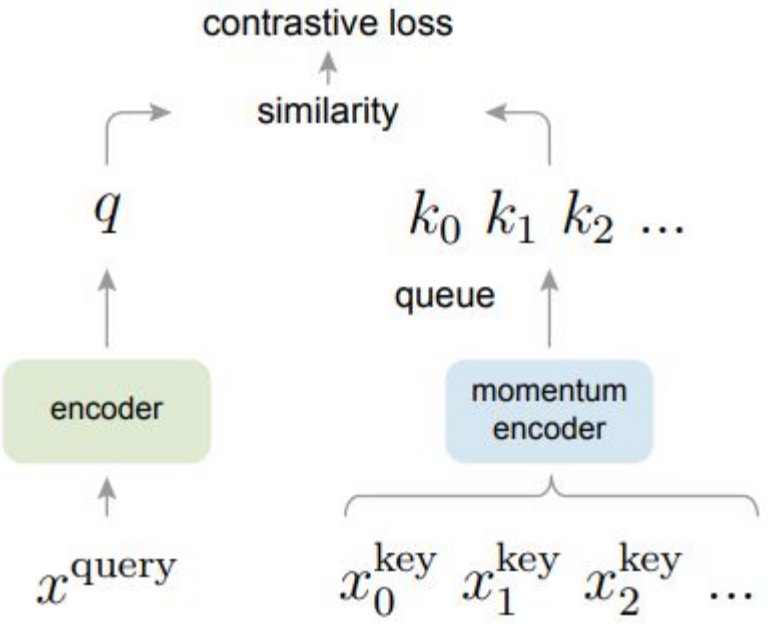
Momentum Contrast for Unsupervised Visual Representation Learning

Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

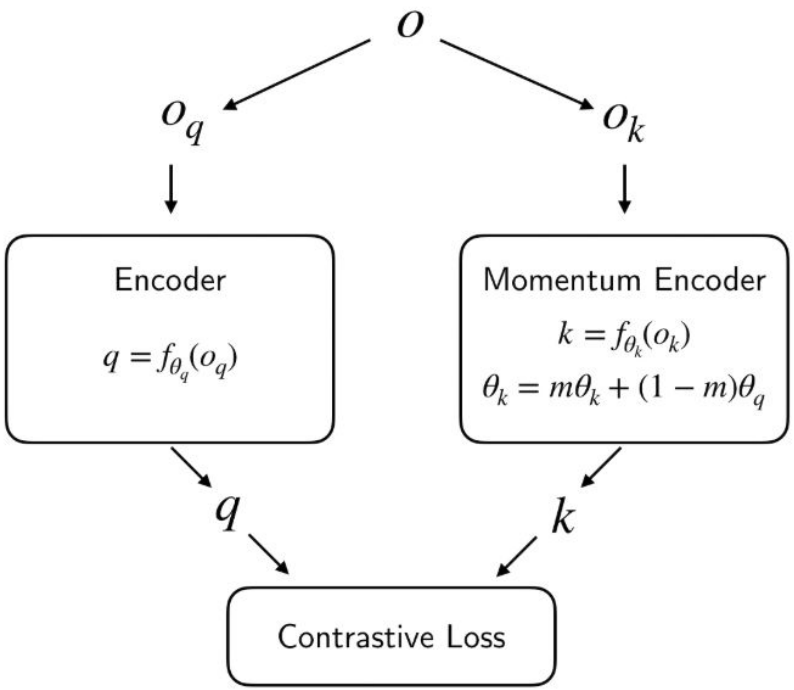
Facebook AI Research (FAIR)

Nov 2019

Momentum Contrast (MoCo)



Momentum Contrast (MoCo)



$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

Momentum Contrast (MoCo)

Algorithm 1 Pseudocode of MoCo in a PyTorch-like style.

```
# f_q, f_k: encoder networks for query and key
# queue: dictionary as a queue of K keys (CxK)
# m: momentum
# t: temperature

f_k.params = f_q.params # initialize
for x in loader: # load a minibatch x with N samples
   x_q = aug(x) # a randomly augmented version
   x_k = aug(x) # another randomly augmented version

   q = f_q.forward(x_q) # queries: NxC
   k = f_k.forward(x_k) # keys: NxC
   k = k.detach() # no gradient to keys

   # positive logits: Nx1
   l_pos = bmm(q.view(N,1,C), k.view(N,C,1))

   # negative logits: NxK
   l_neg = mm(q.view(N,C), queue.view(C,K))

   # logits: Nx(1+K)
   logits = cat([l_pos, l_neg], dim=1)

   # contrastive loss, Eqn.(1)
   labels = zeros(N) # positives are the 0-th
   loss = CrossEntropyLoss(logits/t, labels)

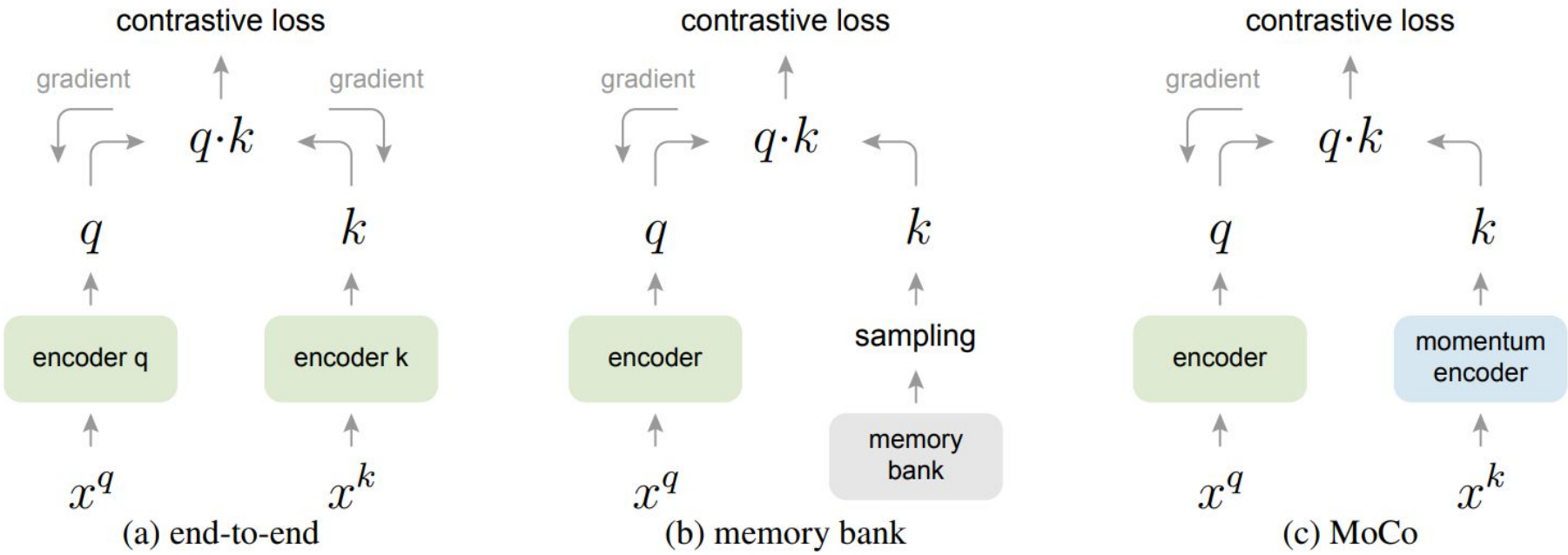
   # SGD update: query network
   loss.backward()
   update(f_q.params)

   # momentum update: key network
   f_k.params = m*f_k.params+(1-m)*f_q.params

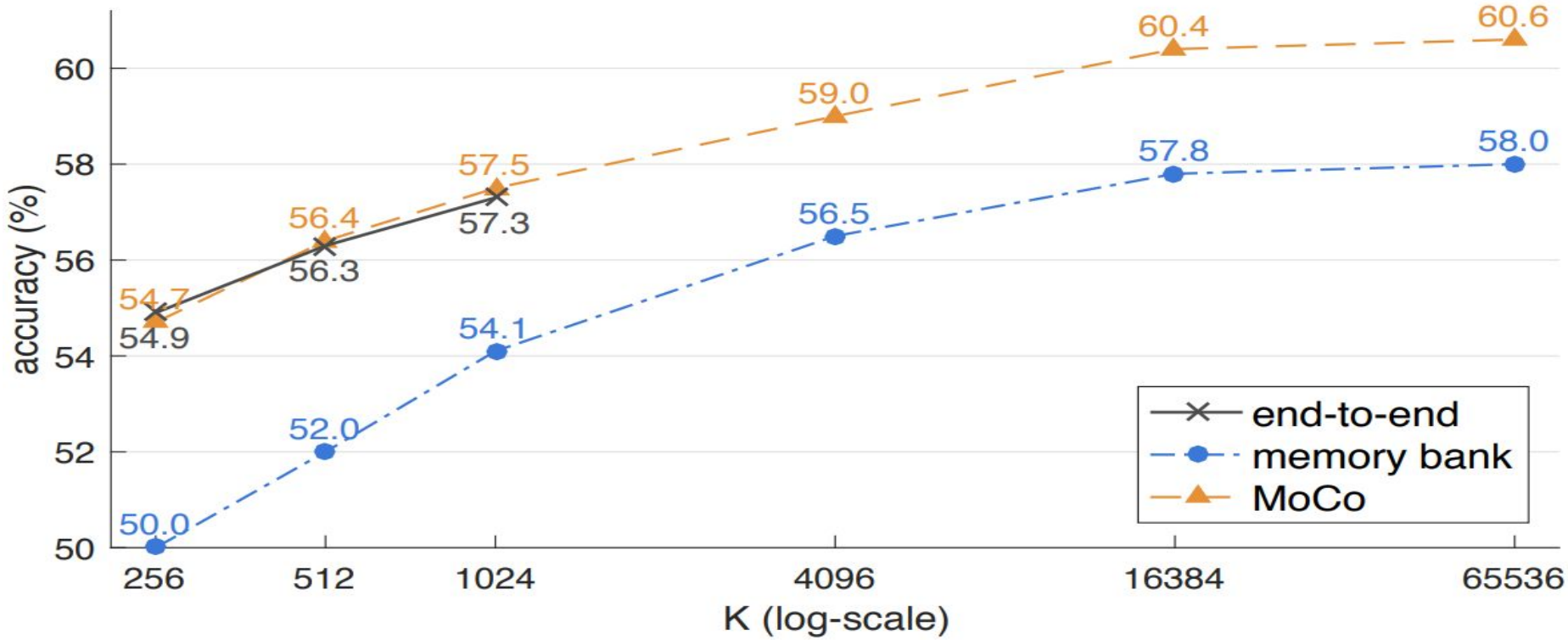
   # update dictionary
   enqueue(queue, k) # enqueue the current minibatch
   dequeue(queue) # dequeue the earliest minibatch
```

bmm: **batch matrix multiplication**; mm: **matrix multiplication**; cat: **concatenation**.

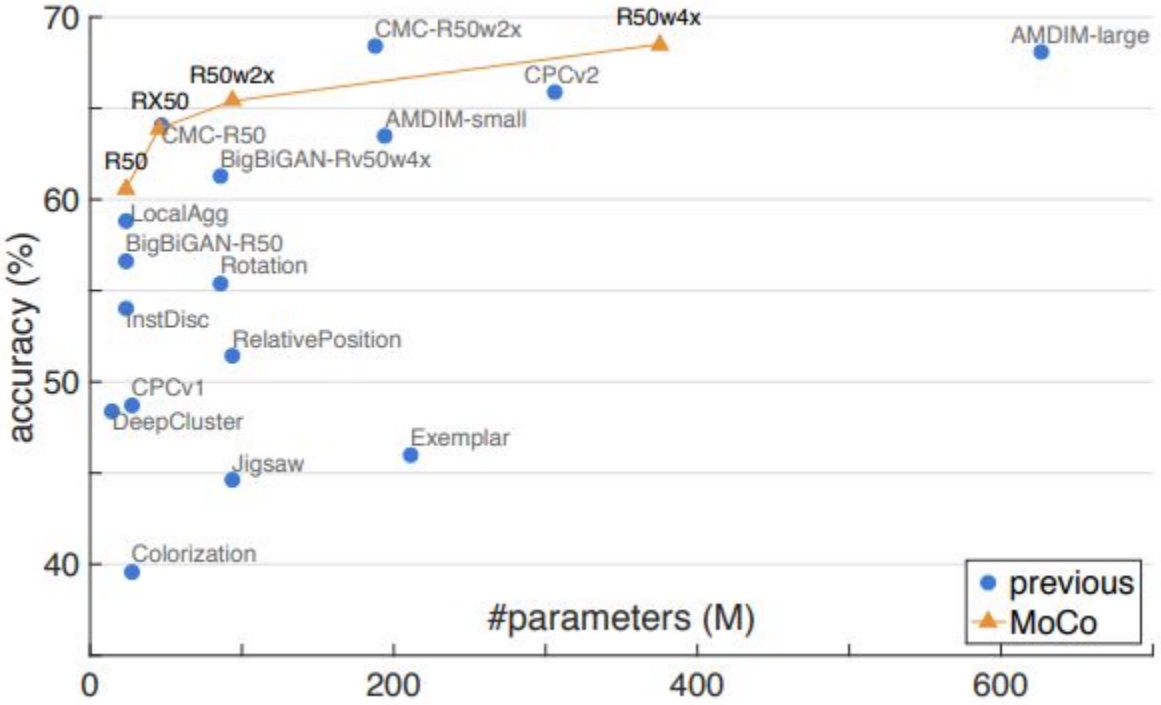
Momentum Contrast (MoCo)



Momentum Contrast (MoCo)



Momentum Contrast (MoCo)

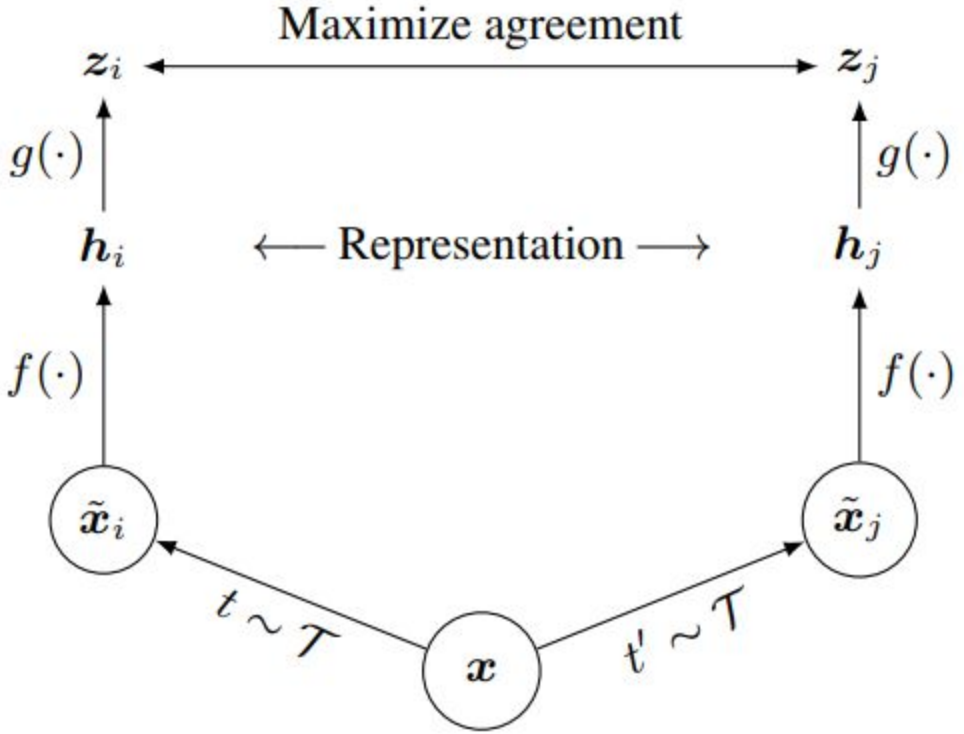


SimCLR

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]

SimCLR

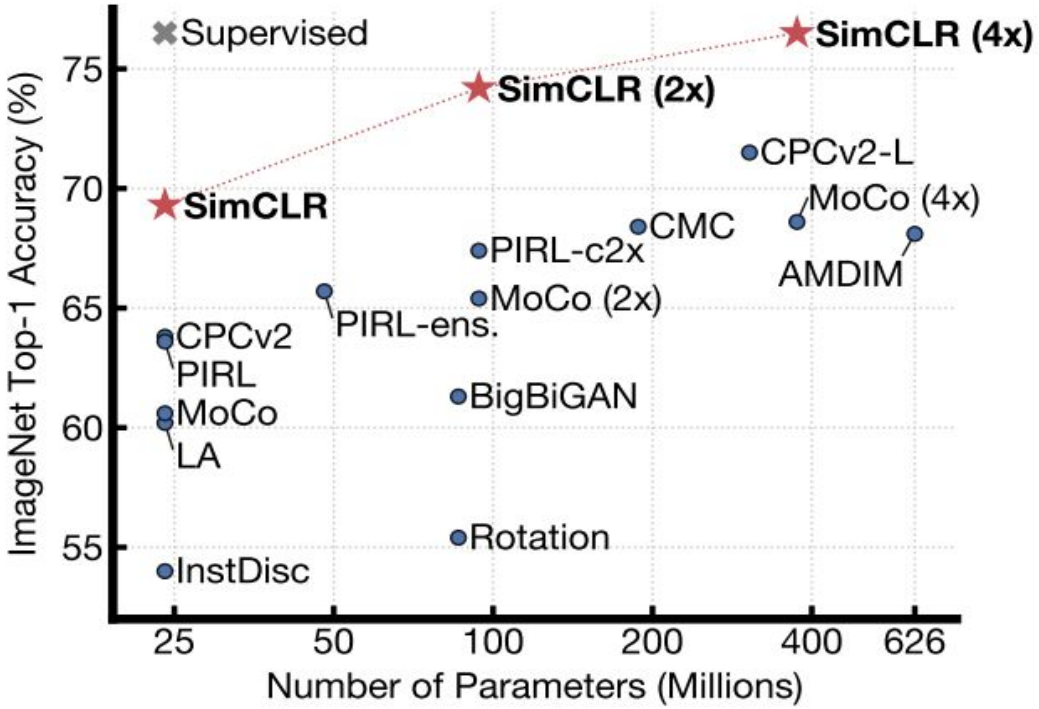

Maximize agreement
z_i
z_j
$g(\cdot)$
$g(\cdot)$
h_i
Representation
h_j
$f(\cdot)$
$f(\cdot)$
$\tilde{x}_i$
$\tilde{x}_j$
$t \sim \mathcal{T}$
$t' \sim \mathcal{T}$
x

SimCLR

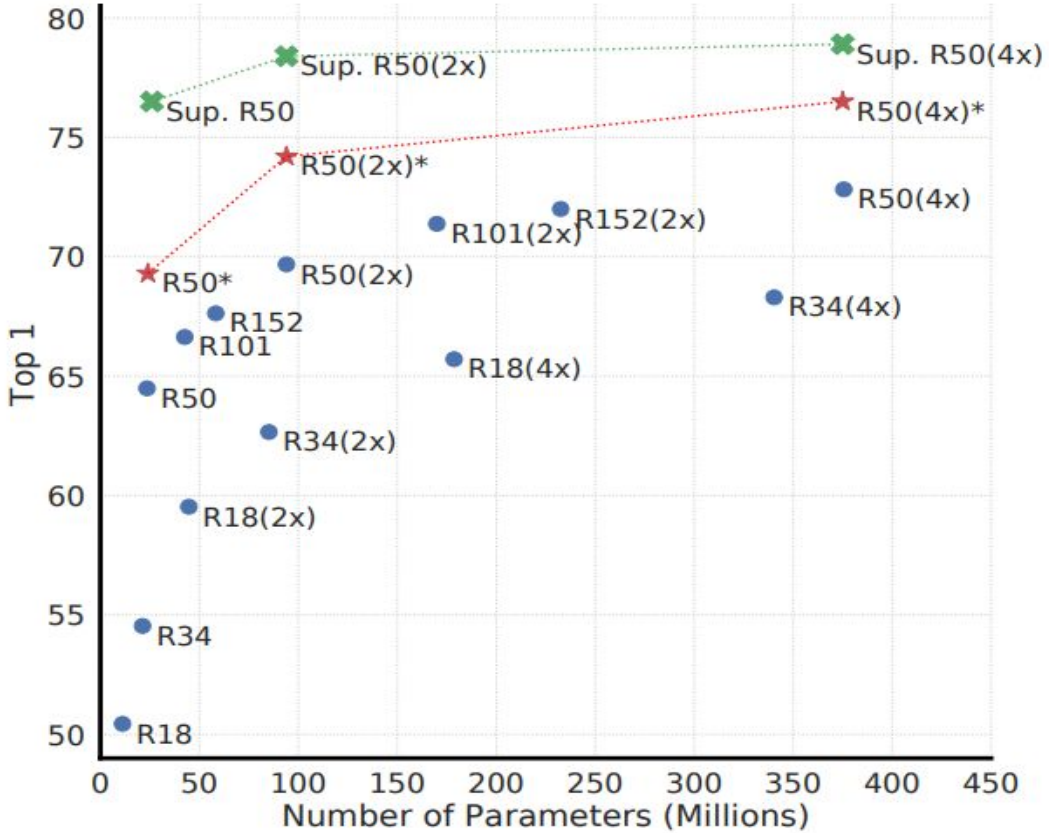
Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, temperature τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \dots, N\}$ **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 $i \in \{1, \dots, 2N\}$ and $j \in \{1, \dots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\tau \|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i,j) = -\log \frac{\exp(s_{i,j})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network f

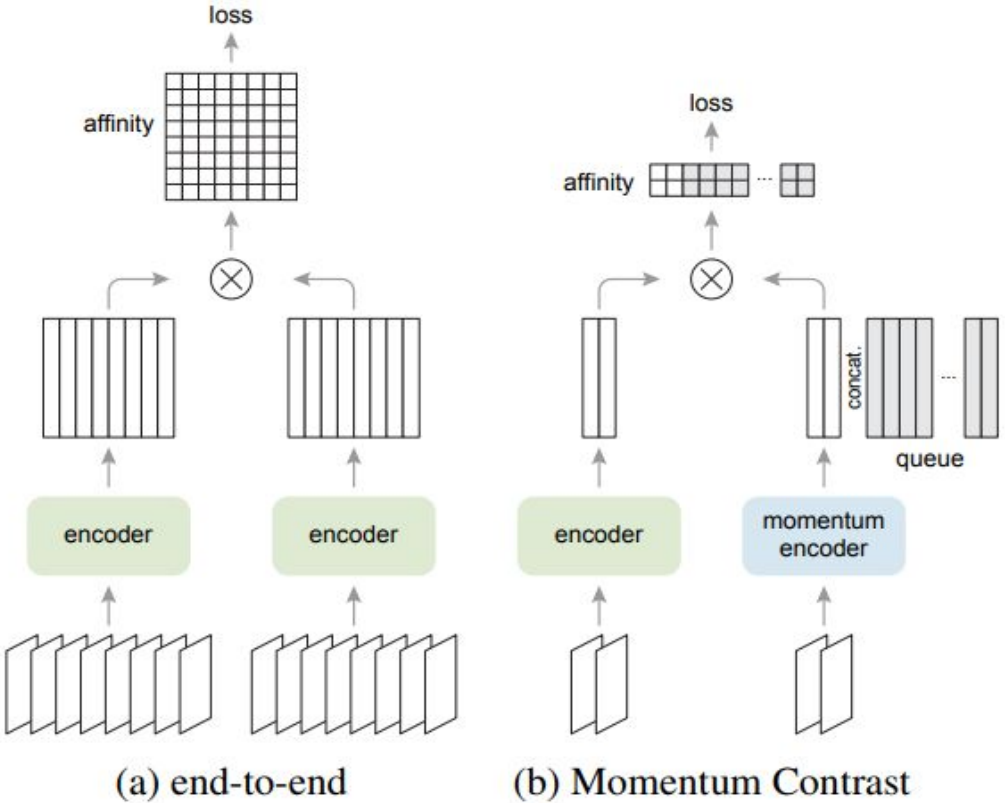
SimCLR


Supervised
SimCLR (4x)
SimCLR (2x)
CPCv2-L
MoCo (4x)
SimCLR
CMC
PIRL-c2x
AMDIM
MoCo (2x)
PIRL-ens.
CPCv2
PIRL
BigBiGAN
MoCo
LA
Rotation
InstDisc
ImageNet Top-1 Accuracy (%)
75
70
65
60
55
25
50
100
200
400
626
Number of Parameters (Millions)

SimCLR


Sup. R50(4x)
Sup. R50(2x)
Sup. R50
R50(4x)*
R50(2x)*
R50(4x)
R152(2x)
R101(2x)
R50*
R50(2x)
R34(4x)
R152
R101
R18(4x)
R50
R34(2x)
R18(2x)
R34
R18
Top 1
80
75
70
65
60
55
50
0
50
100
150
200
250
300
350
400
450
Number of Parameters (Millions)

MoCov2 vs SimCLR



MoCov2 vs SimCLR

case	unsup. pre-train MLP	aug+	cos	epochs	ImageNet acc.	VOC detection AP_{50}	AP	AP_{75}
supervised					76.5	81.3	53.5	58.8
MoCo v1				200	60.6	81.5	55.9	62.6
(a)	✓			200	66.2	82.0	56.4	62.6
(b)		✓		200	63.4	82.2	56.8	63.2
(c)	✓	✓		200	67.3	**82.5**	57.2	63.9
(d)	✓	✓	✓	200	67.5	82.4	57.0	63.6
(e)	✓	✓	✓	**800**	**71.1**	**82.5**	**57.4**	**64.0**

MoCov2 vs SimCLR

case	unsup. pre-train MLP	aug+	cos	epochs	batch	ImageNet acc.
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	**67.5**
*results of **longer** unsupervised training follow:*						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	**71.1**

MoCo v3

An Empirical Study of Training Self-Supervised Vision Transformers

Xinlei Chen* Saining Xie* Kaiming He
Facebook AI Research (FAIR)

Abstract

This paper does not describe a novel method. Instead, it studies a straightforward, incremental, yet must-know baseline given the recent progress in computer vision: self-supervised learning for Vision Transformers (ViT). While

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
```

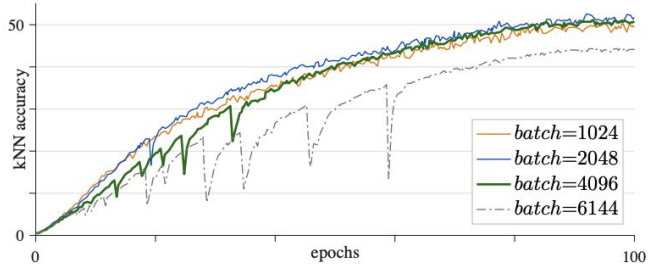
Notes: `mm` is matrix multiplication. `k.t()` is `k`'s transpose. The prediction head is excluded from `f_k` (and thus the momentum update).

R50, 800-ep	MoCo v2 [12]	MoCo v2+ [13]	MoCo v3
linear acc.	71.1	72.2	**73.8**

The improvement here is mainly due to the extra prediction head and large-batch (4096) training.

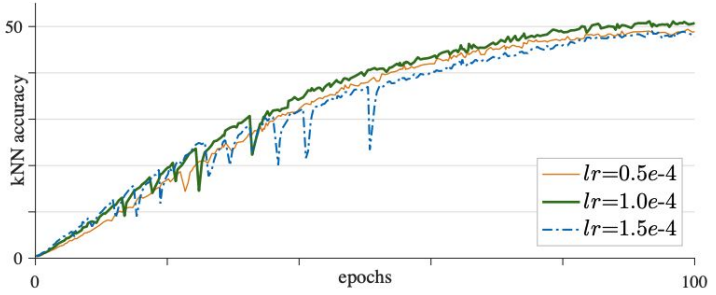
MoCo v3

In principle, it is straightforward to replace a ResNet backbone with a ViT backbone in the contrastive/Siamese self-supervised frameworks. But in practice, a main challenge we have met is the *instability* of training.



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

Figure 1. **Training curves of different batch sizes** (MoCo v3,



lr, $\times 1e$-4	0.5	1.0	1.5
linear acc.	70.4	72.2	71.7

Figure 2. **Training curves of different learning rates** (MoCo v3,

MoCo v3

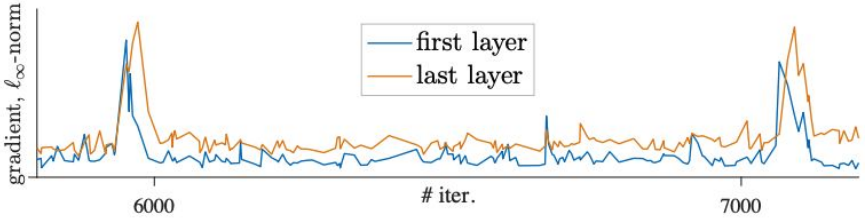
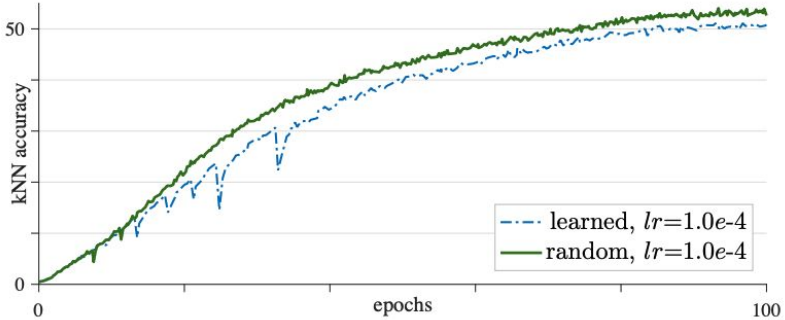
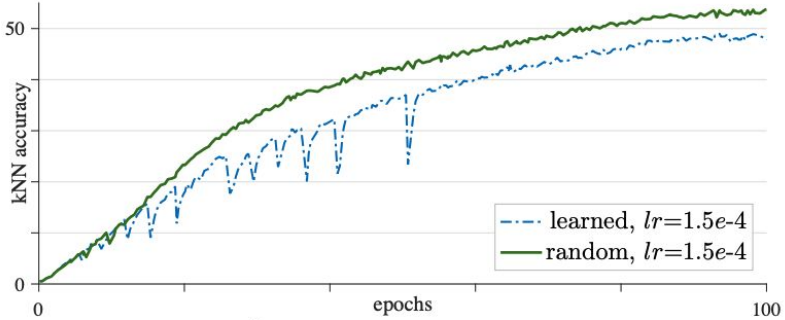


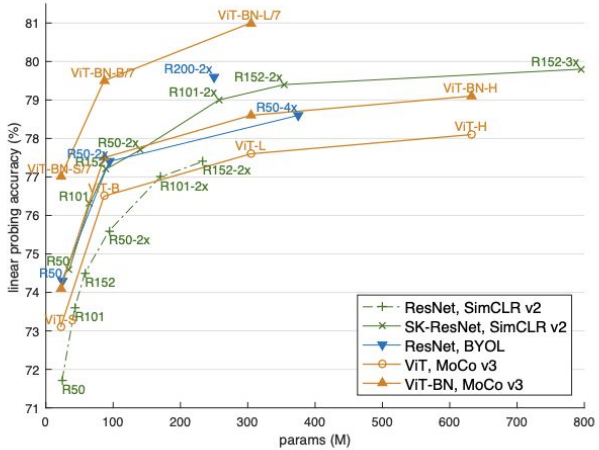
Figure 4. We monitor the gradient magnitude, shown as relative values for the layer. A "spike" in the gradient causes a "dip" in the training curve. We observe that a spike happens earlier in the first layer, and are delayed by tens of iterations in the last layers.





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**

MoCo v3



MoCo v3 w/	ViT-S	ViT-B	ViT-L	ViT-H
ViT baseline	73.4	76.7	77.6	78.1
ViT-BN	74.1	77.5	78.6	79.1
ViT-BN/7	77.0	79.5	**81.0**	-

Figure 8. **Comparisons with state-of-the-art big ResNets**, presented as parameters-*vs*.-accuracy trade-off. All entries are pre-trained with two 224×224 crops, and are evaluated by linear probing. SimCLR v2 results are from Table 1 in [11], and BYOL results are from Table 1 in [18].

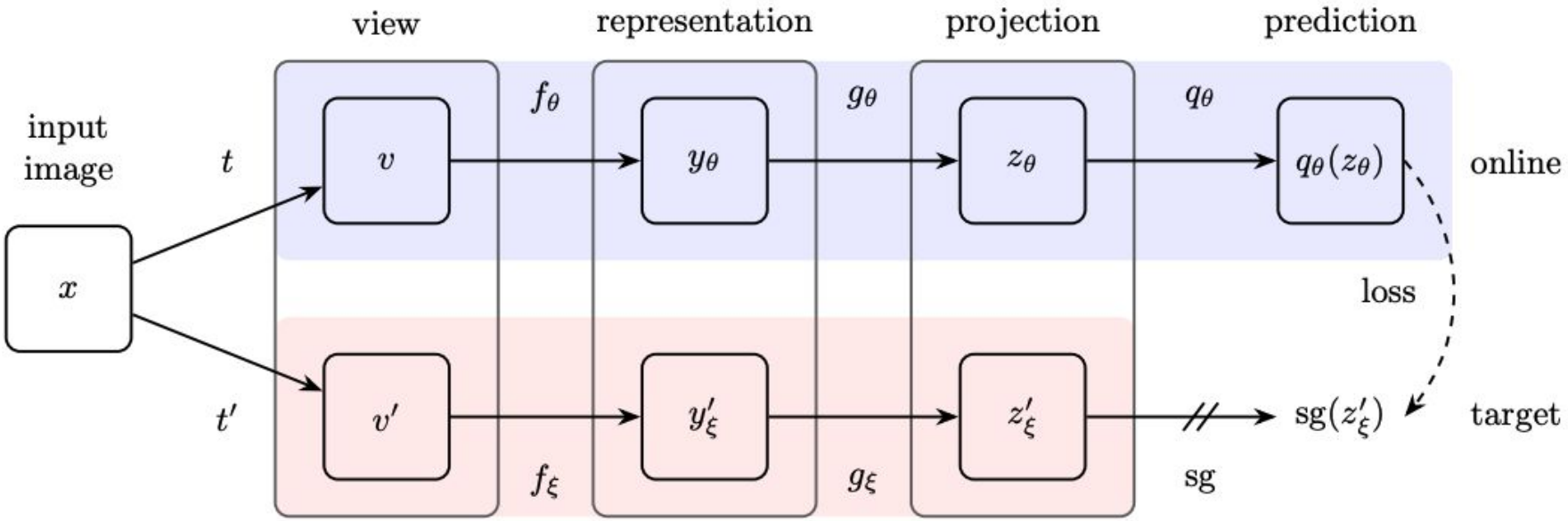
BYOL

Bootstrap Your Own Latent
A New Approach to Self-Supervised Learning

Jean-Bastien Grill[*,1] **Florian Strub**[*,1] **Florent Altché**[*,1] **Corentin Tallec**[*,1] **Pierre H. Richemond**[*,1,2]
Elena Buchatskaya[1] **Carl Doersch**[1] **Bernardo Avila Pires**[1] **Zhaohan Daniel Guo**[1]
Mohammad Gheshlaghi Azar[1] **Bilal Piot**[1] **Koray Kavukcuoglu**[1] **Rémi Munos**[1] **Michal Valko**[1]

[1]DeepMind [2]Imperial College

BYOL

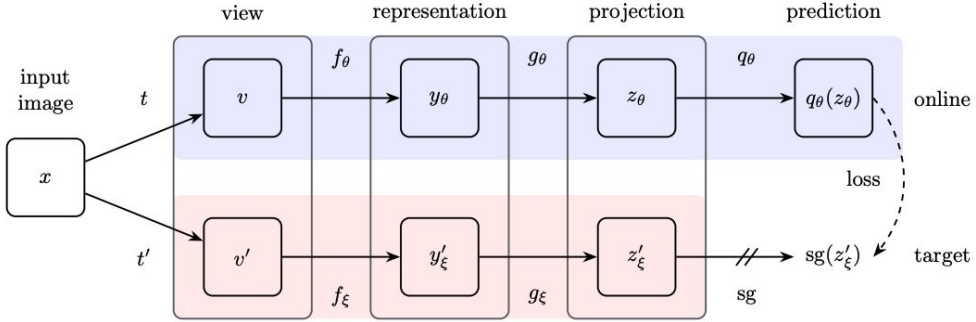

view
representation
projection
prediction
input image
t
t'
x
v
v'
f_θ
f_ξ
y_θ
y'_ξ
g_θ
g_ξ
z_θ
z'_ξ
q_θ
$q_\theta(z_\theta)$
online
loss
$\mathrm{sg}(z'_\xi)$
target
sg

BYOL

Normalize features

$$\overline{z}'_\xi \triangleq z'_\xi / \|z'_\xi\|_2$$

$$\overline{q_\theta}(z_\theta) \triangleq q_\theta(z_\theta) / \|q_\theta(z_\theta)\|_2$$



$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}(z_\theta) - \overline{z}'_\xi\right\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\left\|q_\theta(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}$$

$$\theta \leftarrow \mathrm{optimizer}\left(\theta, \nabla_\theta \mathcal{L}^{\mathtt{BYOL}}_{\theta,\xi}, \eta\right)$$

$$\xi \leftarrow \tau\xi + (1-\tau)\theta,$$

BYOL

Method	Top-1	Top-5
Local Agg.	60.2	-
PIRL [35]	63.6	-
CPC v2 [32]	63.8	85.3
CMC [11]	66.2	87.0
SimCLR [8]	69.3	89.0
MoCo v2 [37]	71.1	-
InfoMin Aug. [12]	73.0	91.1
BYOL (ours)	**74.3**	**91.6**

(a) ResNet-50 encoder.

Method	Architecture	Param.	Top-1	Top-5
SimCLR [8]	ResNet-50 (2×)	94M	74.2	92.0
CMC [11]	ResNet-50 (2×)	94M	70.6	89.7
BYOL (ours)	ResNet-50 (2×)	94M	77.4	93.6
CPC v2 [32]	ResNet-161	305M	71.5	90.1
MoCo [9]	ResNet-50 (4×)	375M	68.6	-
SimCLR [8]	ResNet-50 (4×)	375M	76.5	93.2
BYOL (ours)	ResNet-50 (4×)	375M	78.6	94.2
BYOL (ours)	ResNet-200 (2×)	250M	**79.6**	**94.8**

(b) Other ResNet encoder architectures.

Table 1: Top-1 and top-5 accuracies (in %) under linear evaluation on ImageNet.

BYOL

Method	Top-1		Top-5	
	1%	10%	1%	10%
Supervised [77]	25.4	56.4	48.4	80.4
InstDisc	-	-	39.2	77.4
PIRL [35]	-	-	57.2	83.8
SimCLR [8]	48.3	65.6	75.5	87.8
BYOL (ours)	**53.2**	**68.8**	**78.4**	**89.0**

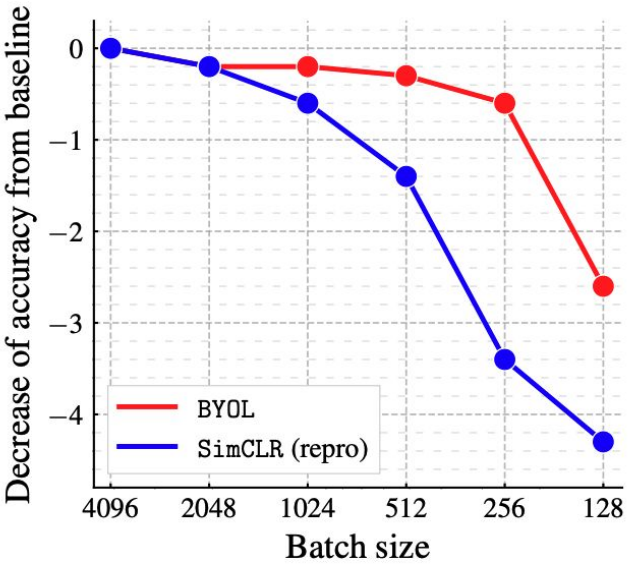
(a) ResNet-50 encoder.

Method	Architecture	Param.	Top-1		Top-5	
			1%	10%	1%	10%
CPC v2 [32]	ResNet-161	305M	-	-	77.9	91.2
SimCLR [8]	ResNet-50 (2×)	94M	58.5	71.7	83.0	91.2
BYOL (ours)	ResNet-50 (2×)	94M	62.2	73.5	84.1	91.7
SimCLR [8]	ResNet-50 (4×)	375M	63.0	74.4	85.8	92.6
BYOL (ours)	ResNet-50 (4×)	375M	69.1	75.7	87.9	92.5
BYOL (ours)	ResNet-200 (2×)	250M	**71.2**	**77.7**	**89.5**	**93.7**

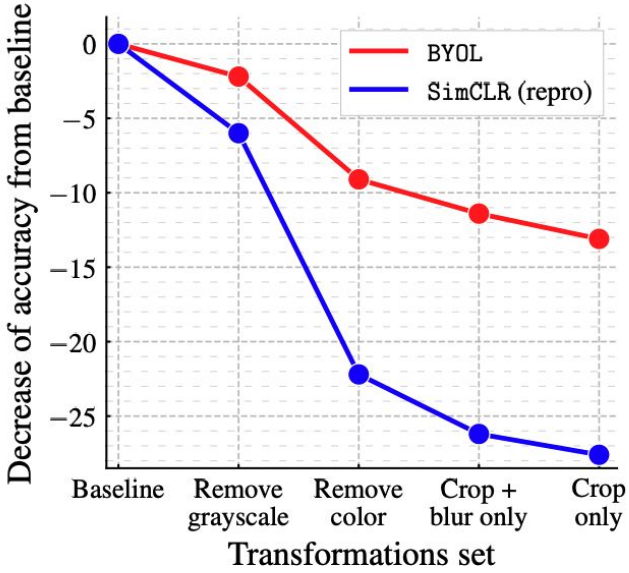
(b) Other ResNet encoder architectures.

Table 2: Semi-supervised training with a fraction of ImageNet labels.

BYOL



(a) Impact of batch size



(b) Impact of progressively removing transformations

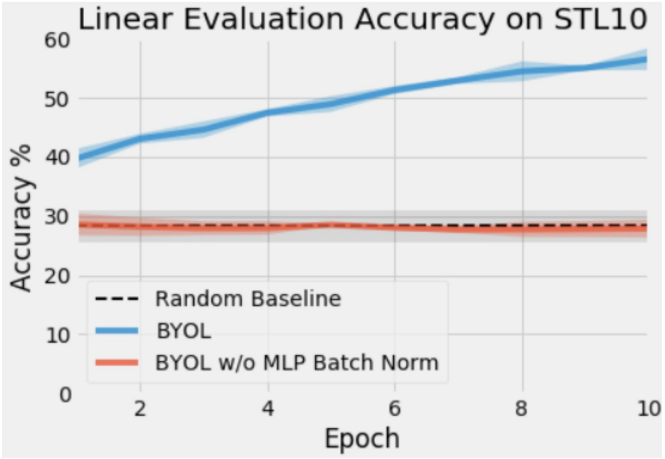
Figure 3: Decrease in top-1 accuracy (in % points) of BYOL and our own reproduction of SimCLR at 300 epochs, under linear evaluation on ImageNet.

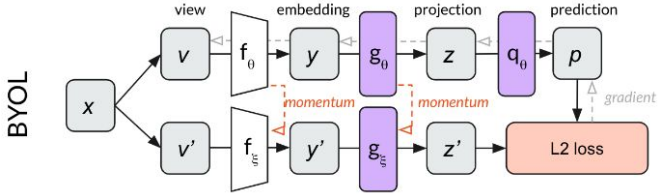
BYOL

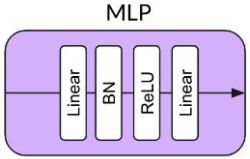
Another perspective

(1) BYOL often performs no better than random when batch normalization is removed, and

(2) the presence of batch normalization implicitly causes a form of contrastive learning.



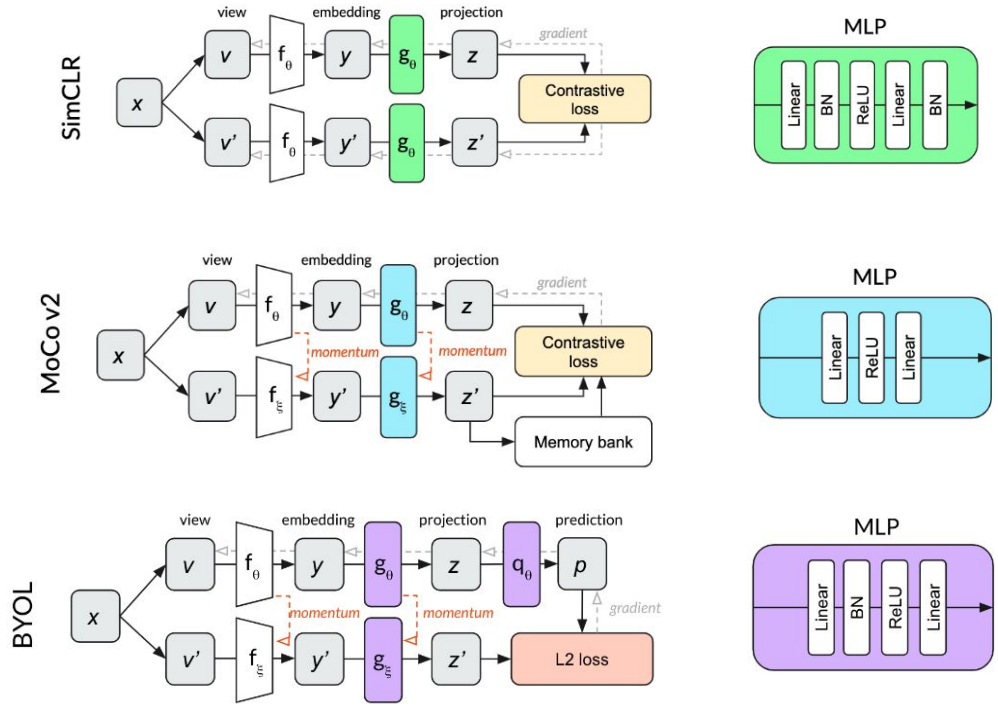




- Batch norm needed to prevent mode collapse
- Implicit contrastive learning - common mode between examples in the minibatch removed

https://imbue.com/research/2020-08-24-understanding-self-supervised-contrastive-learning/

Summary



https://imbue.com/research/2020-08-24-understanding-self-supervised-contrastive-learning/

Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

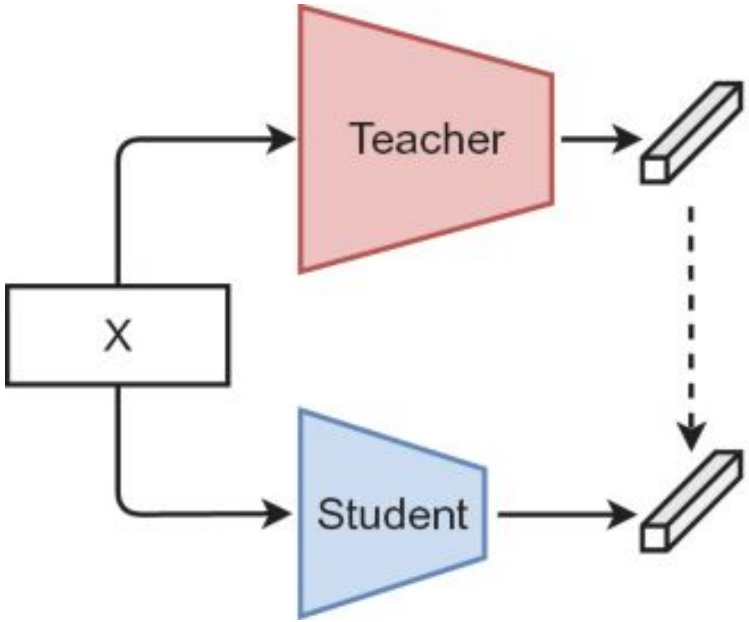
DINO

Emerging Properties in Self-Supervised Vision Transformers

Mathilde Caron[1,2] Hugo Touvron[1,3] Ishan Misra[1] Hervé Jegou[1]
Julien Mairal[2] Piotr Bojanowski[1] Armand Joulin[1]

[1] Facebook AI Research [2] Inria* [3] Sorbonne University

DINO



Consider **knowledge distillation**

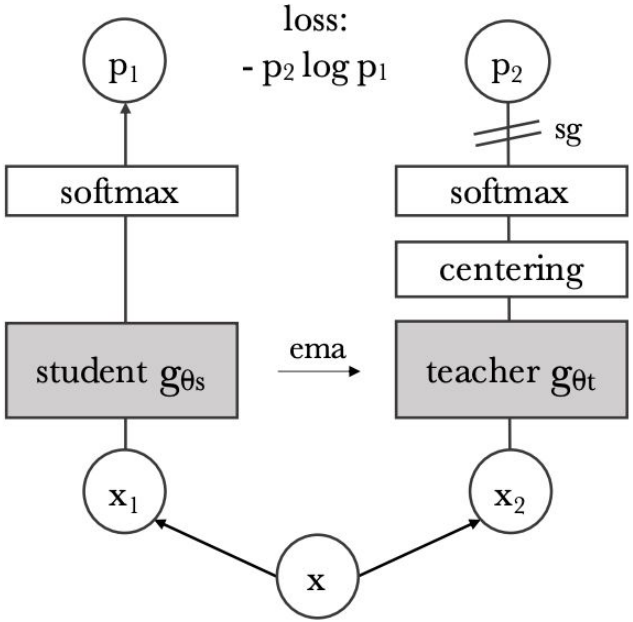
- Student network g_{θ_t} tries to match a teacher network g_{θ_s}
- Minimize the cross entropy of the distributions

$$\min_{\theta_s} H(P_t(x), P_s(x)) \quad P_s(x)^{(i)} = \frac{\exp(g_{\theta_s}(x)^{(i)}/\tau_s)}{\sum_{k=1}^{K} \exp(g_{\theta_s}(x)^{(k)}/\tau_s)}$$

$$\text{where } H(a, b) = -a \log b$$

DINO

Self supervised learning as knowledge distillation



Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

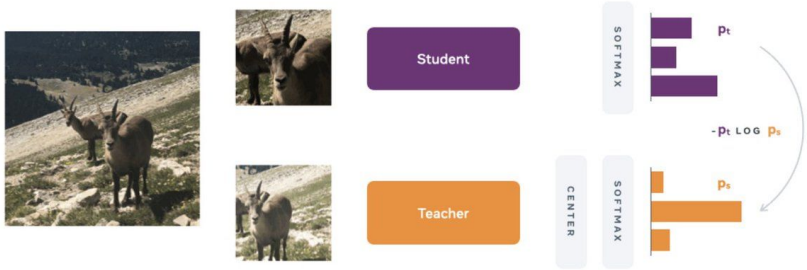
    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

DINO

Apply centering to avoid collapse - use EMA so things work across different batch sizes

$$c \leftarrow mc + (1-m)\frac{1}{B}\sum_{i=1}^{B} g_{\theta_t}(x_i)$$



Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

DINO

Supervised



DINO



	Random	Supervised	DINO
ViT-S/16	22.0	27.3	45.9
ViT-S/8	21.8	23.7	44.7

Threshold attention map get mask

Compare similarity to ground truth mask

Linear and k-NN classification on ImageNet.

Method	Arch.	Param.	im/s	Linear	k-NN
Supervised	RN50	23	1237	79.3	79.3
SCLR [12]	RN50	23	1237	69.1	60.7
MoCov2 [15]	RN50	23	1237	71.1	61.9
InfoMin [67]	RN50	23	1237	73.0	65.3
BarlowT [81]	RN50	23	1237	73.2	66.0
OBoW [27]	RN50	23	1237	73.8	61.9
BYOL [30]	RN50	23	1237	74.4	64.8
DCv2 [10]	RN50	23	1237	75.2	67.1
SwAV [10]	RN50	23	1237	**75.3**	65.7
DINO	RN50	23	1237	**75.3**	**67.5**
Supervised	ViT-S	21	1007	79.8	79.8
BYOL* [30]	ViT-S	21	1007	71.4	66.6
MoCov2* [15]	ViT-S	21	1007	72.7	64.4
SwAV* [10]	ViT-S	21	1007	73.5	66.3
DINO	ViT-S	21	1007	**77.0**	**74.5**
Comparison across architectures					
SCLR [12]	RN50w4	375	117	76.8	69.3
SwAV [10]	RN50w2	93	384	77.3	67.3
BYOL [30]	RN50w2	93	384	77.4	–
DINO	ViT-B/16	85	312	78.2	76.1
SwAV [10]	RN50w5	586	76	78.5	67.1
BYOL [30]	RN50w4	375	117	78.6	–
BYOL [30]	RN200w2	250	123	79.6	73.9
DINO	ViT-S/8	21	180	79.7	**78.3**
SCLRv2 [13]	RN152w3+SK	794	46	79.8	73.1
DINO	ViT-B/8	85	63	**80.1**	77.4

DINO

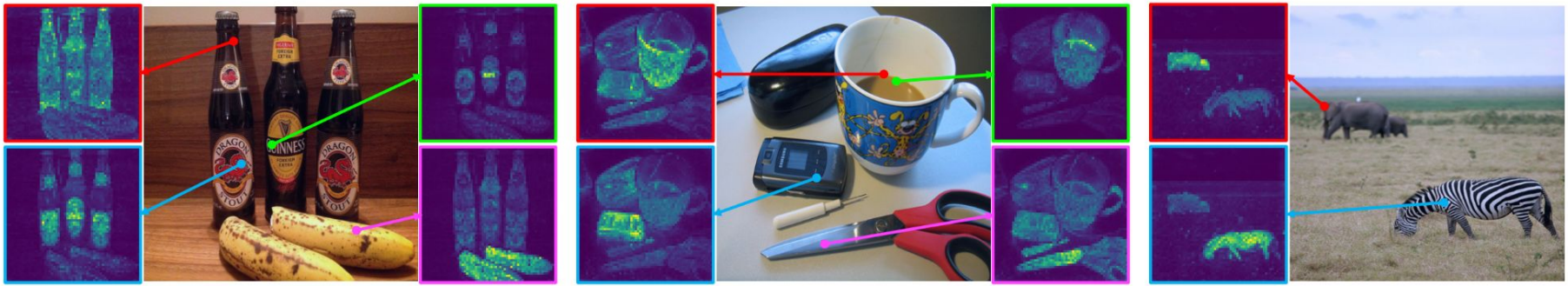


Figure 8: **Self-attention for a set of reference points.** We visualize the self-attention module from the last block of a ViT-S/8 trained with DINO. The network is able to separate objects, though it has been trained with no supervision at all.

DINO

Table 7: **Important component for self-supervised ViT pre-training.** Models are trained for 300 epochs with ViT-S/16. We study the different components that matter for the k-NN and linear ("Lin.") evaluations. For the different variants, we highlight the differences from the default DINO setting. The best combination is the momentum encoder with the multicrop augmentation and the cross-entropy loss. We also report results with BYOL [30], MoCo-v2 [15] and SwAV [10].

	Method	Mom.	SK	MC	Loss	Pred.	k-NN	Lin.
1	DINO	✓	✗	✓	CE	✗	72.8	76.1
2		✗	✗	✓	CE	✗	0.1	0.1
3		✓	✓	✓	CE	✗	72.2	76.0
4		✓	✗	✗	CE	✗	67.9	72.5
5		✓	✗	✓	MSE	✗	52.6	62.4
6		✓	✗	✓	CE	✓	71.8	75.6
7	BYOL	✓	✗	✗	MSE	✓	66.6	71.4
8	MoCov2	✓	✗	✗	INCE	✗	62.0	71.6
9	SwAV	✗	✓	✓	CE	✗	64.7	71.8

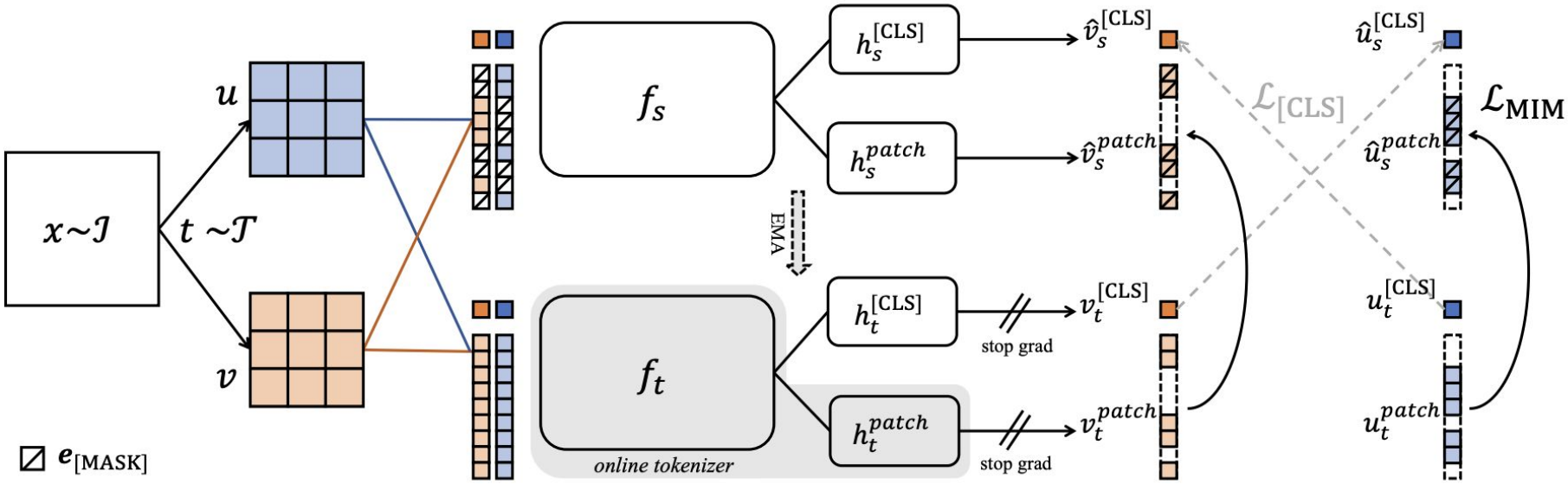
SK: Sinkhorn-Knopp, MC: Multi-Crop, Pred.: Predictor
CE: Cross-Entropy, MSE: Mean Square Error, INCE: InfoNCE

iBOT

iBOT 🤖: Image BERT Pre-Training with Online Tokenizer

Jinghao Zhou[1] **Chen Wei**[2] **Huiyu Wang**[2] **Wei Shen**[3] **Cihang Xie**[4] **Alan Yuille**[2] **Tao Kong**[1]
[1]ByteDance [2]Johns Hopkins University [3]Shanghai Jiao Tong University [4]UC Santa Cruz

iBOT


$x \sim \mathcal{I}$
$t \sim \mathcal{T}$
u
v
$e_{[MASK]}$
f_s
f_t
online tokenizer
EMA
$h_s^{[CLS]}$
h_s^{patch}
$h_t^{[CLS]}$
h_t^{patch}
stop grad
stop grad
$\hat{v}_s^{[CLS]}$
$\hat{v}_s^{patch}$
$v_t^{[CLS]}$
v_t^{patch}
$\mathcal{L}_{[CLS]}$
$\hat{u}_s^{[CLS]}$
$\hat{u}_s^{patch}$
$u_t^{[CLS]}$
u_t^{patch}
$\mathcal{L}_{MIM}$

iBOT

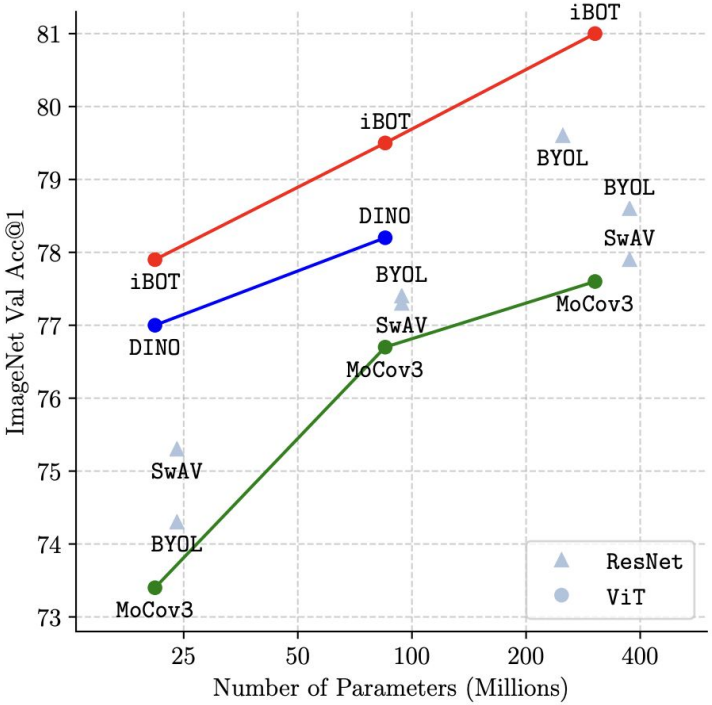


Table 6: **Object detection (Det.) & instance segmentation (ISeg.) on COCO and Semantic segmentation (Seg.) on ADE20K.** We report the results of ViT-S/16 (left) and ViT-B/16 (right). Seg.† denotes using a linear head for semantic segmentation.

Method	Arch.	Param.	Det. AP^b	ISeg. AP^m	Seg. mIoU
Sup.	Swin-T	29	48.1	41.7	44.5
MoBY	Swin-T	29	48.1	41.5	44.1
Sup.	ViT-S/16	21	46.2	40.1	44.5
iBOT	ViT-S/16	21	**49.4**	**42.6**	**45.4**

Method	Det. AP^b	ISeg. AP^m	Seg.† mIoU	Seg. mIoU
Sup.	49.8	43.2	35.4	46.6
BEiT	50.1	43.5	27.4	45.8
DINO	50.1	43.4	34.5	46.8
iBOT	**51.2**	**44.2**	**38.3**	**50.0**

DINO - V2

DINOv2: Learning Robust Visual Features without Supervision

Maxime Oquab**, Timothée Darcet**, Théo Moutakanni**,
Huy V. Vo*, Marc Szafraniec*, Vasil Khalidov*, Pierre Fernandez, Daniel Haziza,
Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba,
Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat,
Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal[1],
Patrick Labatut*, Armand Joulin*, Piotr Bojanowski*

Meta AI Research [1]*Inria*

DINO - V2

ViT-L

	INet-1k k-NN	INet-1k linear
iBOT	72.9	82.3
+(our reproduction)	74.5 ↑ 1.6	83.2 ↑ 0.9
+LayerScale, Stochastic Depth	75.4 ↑ 0.9	82.0 ↓ 1.2
+128k prototypes	76.6 ↑ 1.2	81.9 ↓ 0.1
+KoLeo	78.9 ↑ 2.3	82.5 ↑ 0.6
+SwiGLU FFN	78.7 ↓ 0.2	83.1 ↑ 0.6
+Patch size 14	78.9 ↑ 0.2	83.5 ↑ 0.4
+Teacher momentum 0.994	79.4 ↑ 0.5	83.6 ↑ 0.1
+Tweak warmup schedules	80.5 ↑ 1.1	83.8 ↑ 0.2
+Batch size 3k	81.7 ↑ 1.2	84.7 ↑ 0.9
+Sinkhorn-Knopp	81.7 =	84.7 =
+Untying heads = DINOv2	82.0 ↑ 0.3	84.5 ↓ 0.2

DINO-V2

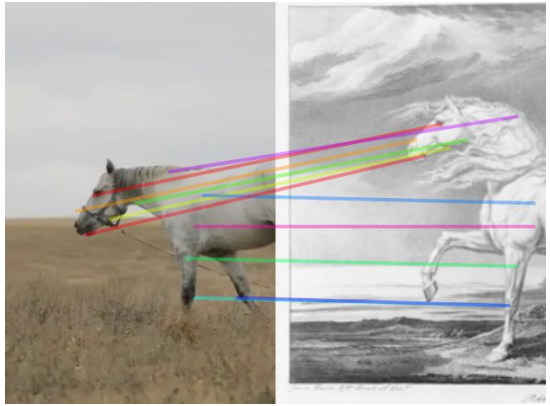
Method	Arch.	Data	Text sup.	kNN val	linear val	linear ReaL	linear V2
Weakly supervised							
CLIP	ViT-L/14	WIT-400M	✓	79.8	84.3	88.1	75.3
CLIP	ViT-L/14_{336}	WIT-400M	✓	80.5	85.3	88.8	75.8
SWAG	ViT-H/14	IG3.6B	✓	82.6	85.7	88.7	77.6
OpenCLIP	ViT-H/14	LAION-2B	✓	81.7	84.4	88.4	75.5
OpenCLIP	ViT-G/14	LAION-2B	✓	83.2	86.2	89.4	77.2
EVA-CLIP	ViT-g/14	custom*	✓	**83.5**	86.4	89.3	77.4
Self-supervised							
MAE	ViT-H/14	INet-1k	×	49.4	76.6	83.3	64.8
DINO	ViT-S/8	INet-1k	×	78.6	79.2	85.5	68.2
SEERv2	RG10B	IG2B	×	–	79.8	–	–
MSN	ViT-L/7	INet-1k	×	79.2	80.7	86.0	69.7
EsViT	Swin-B/W=14	INet-1k	×	79.4	81.3	87.0	70.4
Mugs	ViT-L/16	INet-1k	×	80.2	82.1	86.9	70.8
iBOT	ViT-L/16	INet-22k	×	72.9	82.3	87.5	72.4
DINOv2	ViT-S/14	LVD-142M	×	79.0	81.1	86.6	70.9
	ViT-B/14	LVD-142M	×	82.1	84.5	88.3	75.1
	ViT-L/14	LVD-142M	×	**83.5**	86.3	89.5	78.0
	ViT-g/14	LVD-142M	×	**83.5**	**86.5**	**89.6**	**78.4**

DINO - V2



Figure 1: **Visualization of the first PCA components.** We compute a PCA between the patches of the images from the same column (a, b, c and d) and show their first 3 components. Each component is matched to a different color channel. Same parts are matched between related images despite changes of pose, style or even objects. Background is removed by thresholding the first PCA component.

DINO-V2



Feature matching



Image retrieval



Segmentation

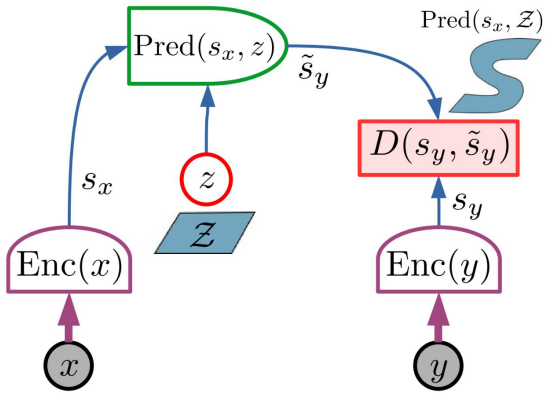


Depth prediction

JEPA

A Path Towards Autonomous Machine Intelligence
Version 0.9.2, 2022-06-27

Yann LeCun



The main advantage of JEPA is that it performs predictions in representation space, eschewing the need to predict every detail of $\mathbf{y}$*, and enabling the elimination of irrelevant details by the encoders. More precisely, the main advantage of this architecture for representing multi-modal dependencies is twofold: (1) the encoder function* $s_y = \text{Enc}(y)$ *may possess invariance properties that will make it produce the same* s_y *for a set of different* y*. This makes the energy constant over this set and allows the model to capture complex multi-modal dependencies; (2) The latent variable* z*, when varied over a set* $\mathcal{Z}$*, can produce a set of plausible predictions* $\text{Pred}(s_x, \mathcal{Z}) = \{\tilde{s}_y = \text{Pred}(s_x, z)\ \forall z \in \mathcal{Z}\}$
If x *is a video clip of a car approaching a fork in the road,* s_x *and* s_y *may represent the position, orientation, velocity and other characteristics of the car before and after the fork, respectively, ignoring irrelevant details such as the trees bordering the road or the texture of the sidewalk.* z *may represent whether the car takes the left branch or the right branch of the road.*

Figure 12: *The Joint-Embedding Predictive Architecture (JEPA) consists of two encoding branches. The first branch computes* s_x*, a representation of* x *and the second branch* s_y *a representation of* y*.*

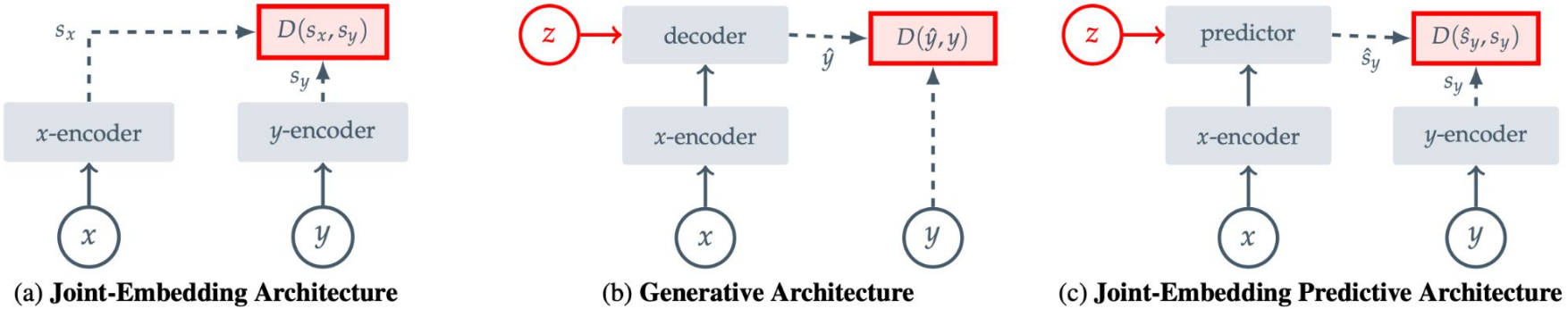
I-JEPA

Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture

Mahmoud Assran[1,2,3*] **Quentin Duval**[1] **Ishan Misra**[1] **Piotr Bojanowski**[1]
Pascal Vincent[1] **Michael Rabbat**[1,3] **Yann LeCun**[1,4] **Nicolas Ballas**[1]

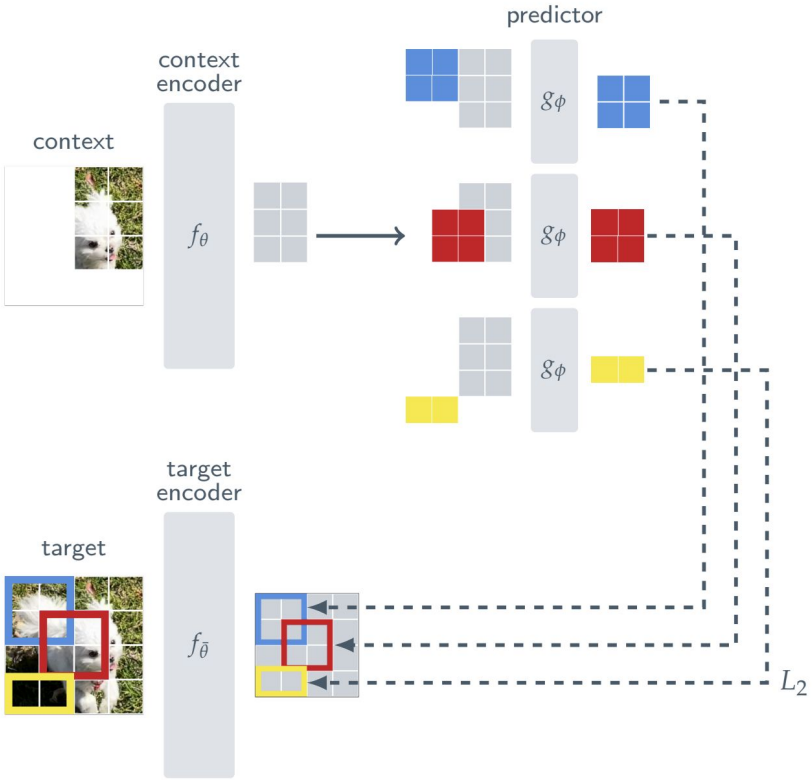
[1]Meta AI (FAIR) [2]McGill University [3] Mila, Quebec AI Institute [4]New York University

I-JEPA



(a) **Joint-Embedding Architecture** (b) **Generative Architecture** (c) **Joint-Embedding Predictive Architecture**

I-JEPA



Context Encoder, Target Encoder and Predictor are ViTs

Predictor

- Transformer encoder
- Concat context tokens
- Have masked tokens for prediction patches

$$\hat{\boldsymbol{s}}_y(i) = \{\hat{\boldsymbol{s}}_{y_j}\}_{j\in B_i} = g_\phi(\boldsymbol{s}_x, \{\boldsymbol{m}_j\}_{j\in B_i})$$

$$\frac{1}{M}\sum_{i=1}^{M} D\left(\hat{\boldsymbol{s}}_y(i), \boldsymbol{s}_y(i)\right) = \frac{1}{M}\sum_{i=1}^{M}\sum_{j\in B_i} \|\hat{\boldsymbol{s}}_{y_j} - \boldsymbol{s}_{y_j}\|_2^2.$$

I-JEPA

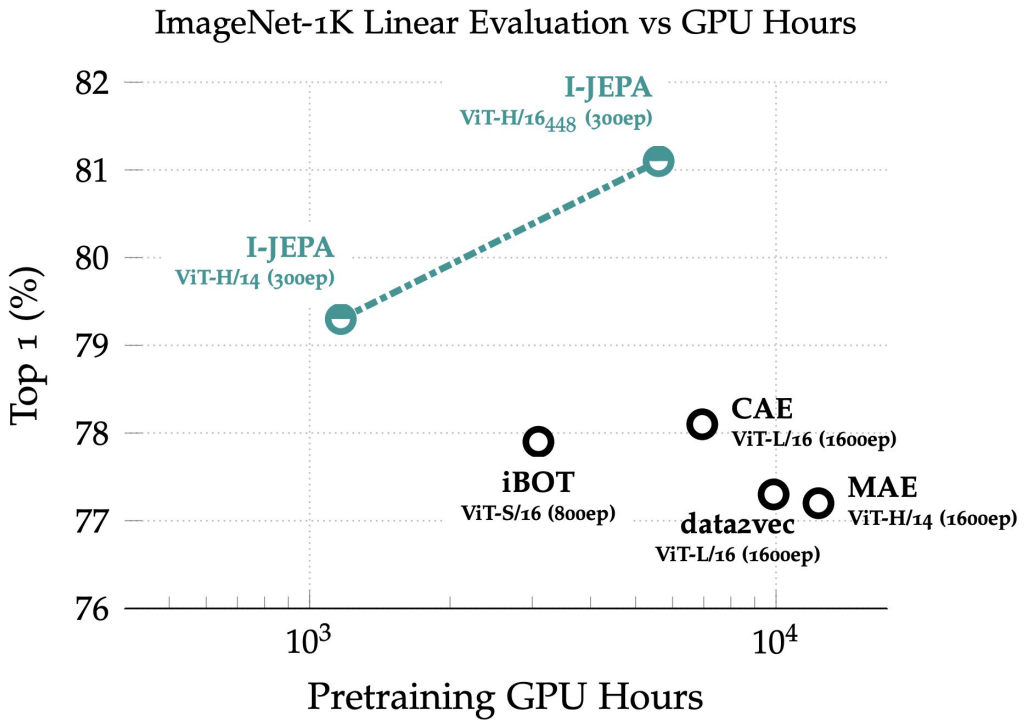


Figure 4. Examples of our context and target-masking strategy. Given an image, we randomly sample 4 target blocks with scale in the range $(0.15, 0.2)$ and aspect ratio in the range $(0.75, 1.5)$. Next, we randomly sample a context block with scale in the range $(0.85, 1.0)$ and remove any overlapping target blocks. Under this strategy, the target-blocks are relatively semantic, and the context-block is informative, yet sparse (efficient to process).

Context and Target Selection

I-JEPA

Method	Arch.	Epochs	Top-1
Methods without view data augmentations			
data2vec [8]	ViT-L/16	1600	77.3
MAE [36]	ViT-B/16	1600	68.0
	ViT-L/16	1600	76.0
	ViT-H/14	1600	77.2
CAE [22]	ViT-B/16	1600	70.4
	ViT-L/16	1600	78.1
I-JEPA	ViT-B/16	600	72.9
	ViT-L/16	600	77.5
	ViT-H/14	300	79.3
	ViT-H/16_{448}	300	**81.1**
Methods using extra view data augmentations			
SimCLR v2 [21]	RN152 (2×)	800	79.1
DINO [18]	ViT-B/8	300	80.1
iBOT [79]	ViT-L/16	250	**81.0**



Freeze context encoder and predictor

Train a RCDM (representation conditioned diffusion model to visualize predictions

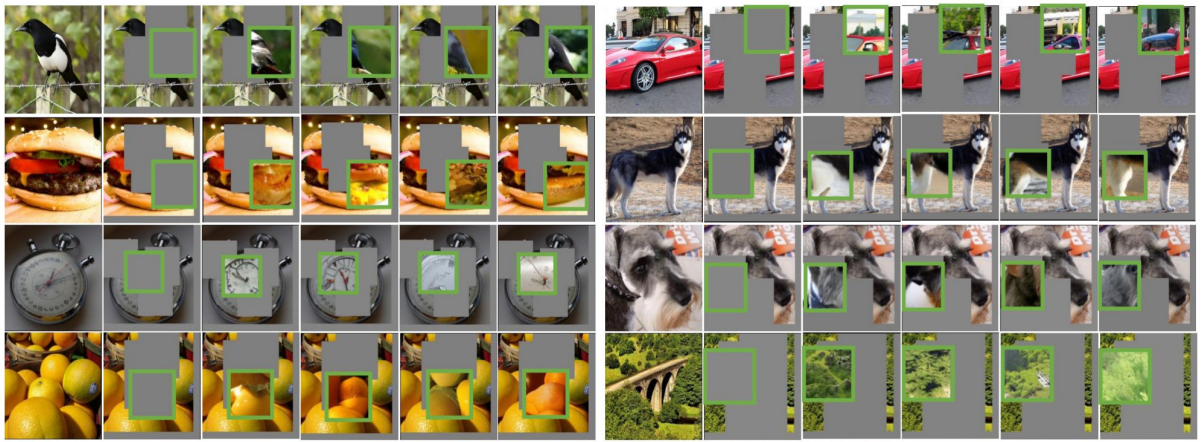


Figure 6. **Visualization of I-JEPA predictor representations**. For each image: first column contains the original image; second column contains the context image, which is processed by a pretrained I-JEPA ViT-H/14 encoder. Green bounding boxes in subsequent columns contain samples from a generative model decoding the output of the pretrained I-JEPA predictor, which is conditioned on positional mask tokens corresponding to the location of the green bounding box. Qualities that are common across samples represent information that

V-JEPA

Revisiting Feature Prediction for Learning Visual Representations from Video

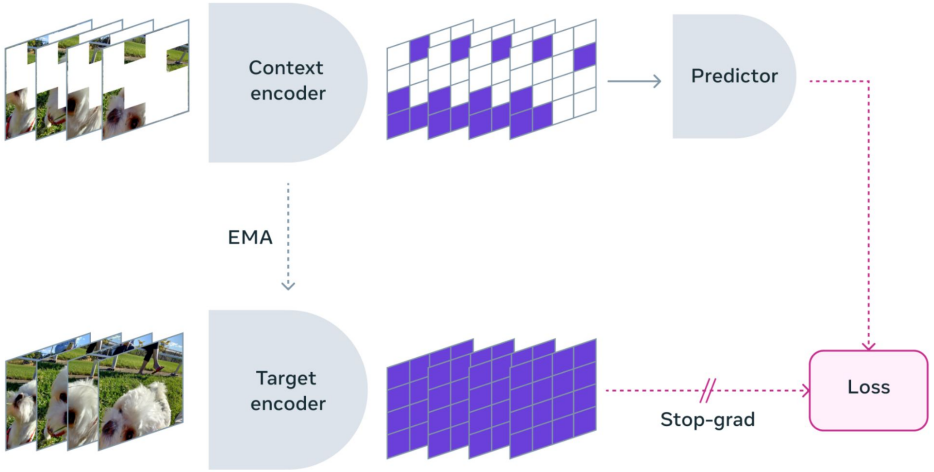
Adrien Bardes[1,2,3], **Quentin Garrido**[1,4], **Jean Ponce**[3,5,6], **Xinlei Chen**[1], **Michael Rabbat**[1], **Yann LeCun**[1,5,6], **Mahmoud Assran**[1,†], **Nicolas Ballas**[1,†]

[1]FAIR at Meta, [2]Inria, [3]École normale supérieure, CNRS, PSL Research University, [4]Univ. Gustave Eiffel, CNRS, LIGM, [5]Courant Institute, New York University, [6]Center for Data Science, New York University
[†]Joint last author

V-JEPA

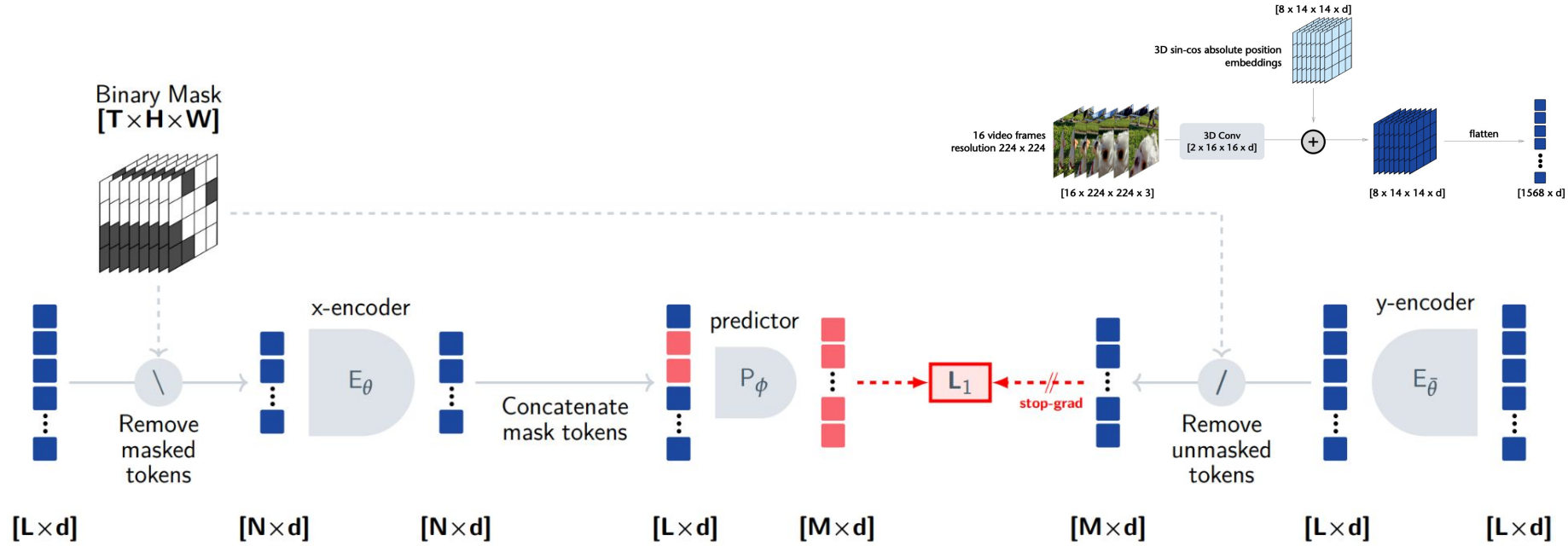
We seek to answer the simple question:

How effective is feature prediction as a stand-alone objective for unsupervised learning from video with modern tools?



- Performs well on downstream video/image tasks
- Better than pixel prediction approaches if freezing weights
- Competitive with full fine tuning
- Shorter training schedules

V-JEPA



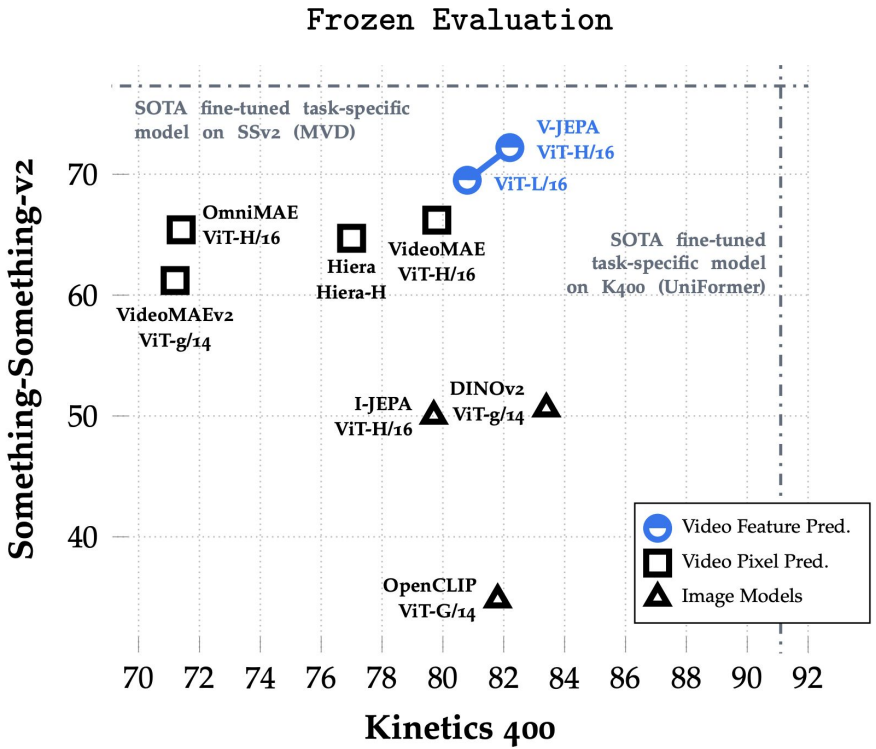
Short range masks: union of 8 randomly sampled target blocks converting 15 % of each frame

Long range masks:union of 2 randomly rampled target blocks covering 70% of each frame

~90% mask ratio

Train on large dataset of 2 million videos from publicly available dataset

V-JEPA


Frozen Evaluation
SOTA fine-tuned task-specific model on SSv2 (MVD)
V-JEPA ViT-H/16
ViT-L/16
OmniMAE ViT-H/16
Hiera Hiera-H
VideoMAE ViT-H/16
SOTA fine-tuned task-specific model on K400 (UniFormer)
VideoMAEv2 ViT-g/14
I-JEPA ViT-H/16
DINOv2 ViT-g/14
OpenCLIP ViT-G/14
Video Feature Pred.
Video Pixel Pred.
Image Models
Something-Something-v2
70
60
50
40
70 72 74 76 78 80 82 84 86 88 90 92
Kinetics 400

V-JEPA

Comparison with State-of-the-Art Models.

Method	Arch.	Params.	Data	*Video Tasks* K400 (16×8×3)	SSv2 (16×2×3)	AVA	*Image Tasks* IN1K	Places205	iNat21
Methods pretrained on Images									
I-JEPA	ViT-H/16_{512}	630M	IN22K	79.7	50.0	19.8	84.4	66.5	85.7
OpenCLIP	ViT-G/14	1800M	LAION	81.8	34.8	23.2	85.3	**70.2**	83.6
DINOv2	ViT-g/14	1100M	LVD-142M	**83.4**	50.6	24.3	**86.2**	68.4	**88.8**
Methods pretrained on Videos									
MVD	ViT-L/16	200M	IN1K+K400	79.4	66.5	19.7	73.3	59.4	65.7
OmniMAE	ViT-H/16	630M	IN1K+SSv2	71.4	65.4	16.0	76.3	60.6	72.4
VideoMAE	ViT-H/16	630M	K400	79.8	66.2	20.7	72.3	59.1	65.5
VideoMAEv2	ViT-g/14	1100M	Un.Hybrid	71.2	61.2	12.9	71.4	60.6	68.3
Hiera	Hiera-H	670M	K400	77.0	64.7	17.5	71.4	59.5	61.7
V-JEPA	ViT-L/16	200M	VideoMix2M	80.8	69.5	25.6	74.8	60.3	67.8
	ViT-H/16	630M		**82.0**	71.4	**25.8**	75.9	61.7	67.9
	ViT-H/16_{384}	630M		81.9	**72.2**	25.0	**77.4**	**62.8**	**72.6**

V-JEPA

Pixels vs. Featurized Targets.

Target	Arch.	Frozen Evaluation K400 (16×1×1)	Frozen Evaluation SSv2 (16×1×1)	Frozen Evaluation IN1K	Fine-Tuning K400-ft (16×5×3)
Pixels	ViT-L/16	68.6	66.0	73.3	85.4
Features	ViT-L/16	**73.7**	**66.2**	**74.8**	**85.6**

Average Pooling vs. Adaptive Pooling.

Method	Arch.	Frozen Evaluation K400 (16×1×1) Avg.	K400 (16×1×1) Att.	SSv2 (16×1×1) Avg.	SSv2 (16×1×1) Att.
V-JEPA	ViT-L/16	56.7	**73.7**	50.1	**66.2**

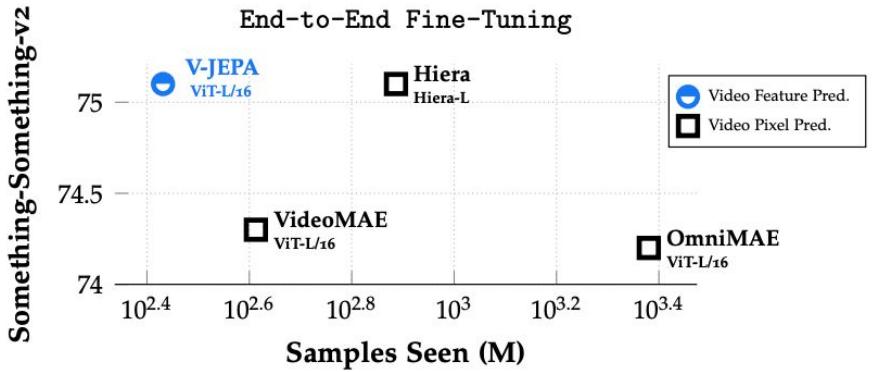
Pretraining Data Distribution.

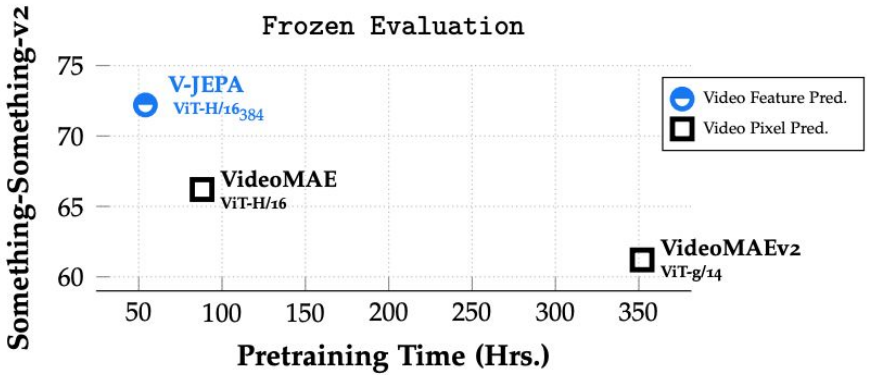
Arch.	Data	#Samples	Frozen Evaluation K400 (16×1×1)	SSv2 (16×1×1)	IN1K	Avg.
ViT-L/16	K710	700K	**75.8**	63.2	73.7	70.9
	K710+SSv2	900K	72.9	**67.4**	72.8	71.0
	K710+HT	1900K	74.5	64.2	**74.8**	71.1
	VideoMix2M	2000K	73.7	66.2	**74.8**	**71.5**
ViT-H/16	K710+SSv2	900K	**75.7**	66.8	73.7	72.0
	VideoMix2M	2000K	74.0	**68.5**	**75.9**	**72.8**

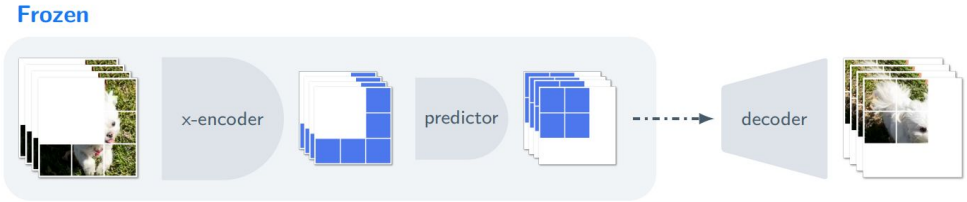
Ablating Prediction Task.

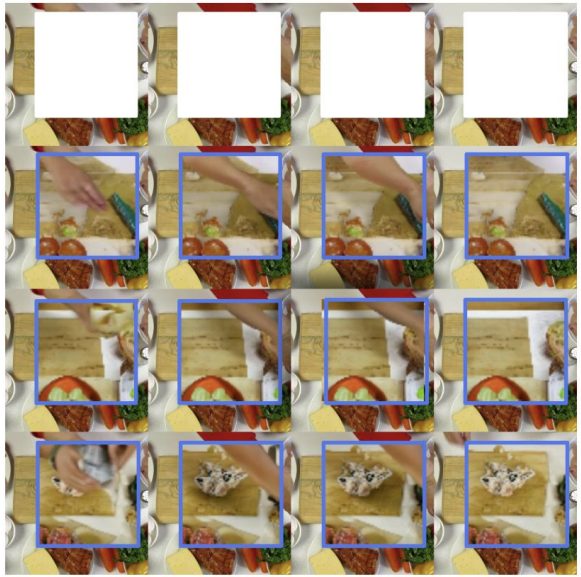
Masking	Frozen Evaluation K400 (16×1×1)	SSv2 (16×1×1)	IN1K
random-tube[0.9]	51.5	46.4	55.6
causal multi-block[6]	61.3	49.8	66.9
causal multi-block[12]	71.9	63.6	72.2
multi-block	**72.9**	**67.4**	**72.8**

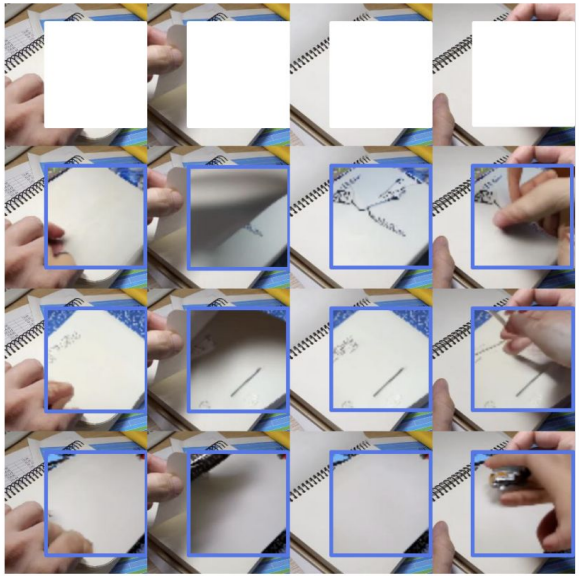
V-JEPA


End-to-End Fine-Tuning
Something-Something-v2
75
74.5
74
V-JEPA
ViT-L/16
Hiera
Hiera-L
VideoMAE
ViT-L/16
OmniMAE
ViT-L/16
Video Feature Pred.
Video Pixel Pred.
$10^{2.4}$
$10^{2.6}$
$10^{2.8}$
10^{3}
$10^{3.2}$
$10^{3.4}$
Samples Seen (M)


Frozen Evaluation
Something-Something-v2
75
70
65
60
V-JEPA
ViT-H/16_{384}
VideoMAE
ViT-H/16
VideoMAEv2
ViT-g/14
Video Feature Pred.
Video Pixel Pred.
50
100
150
200
250
300
350
Pretraining Time (Hrs.)


Frozen
x-encoder
predictor
decoder





Outline

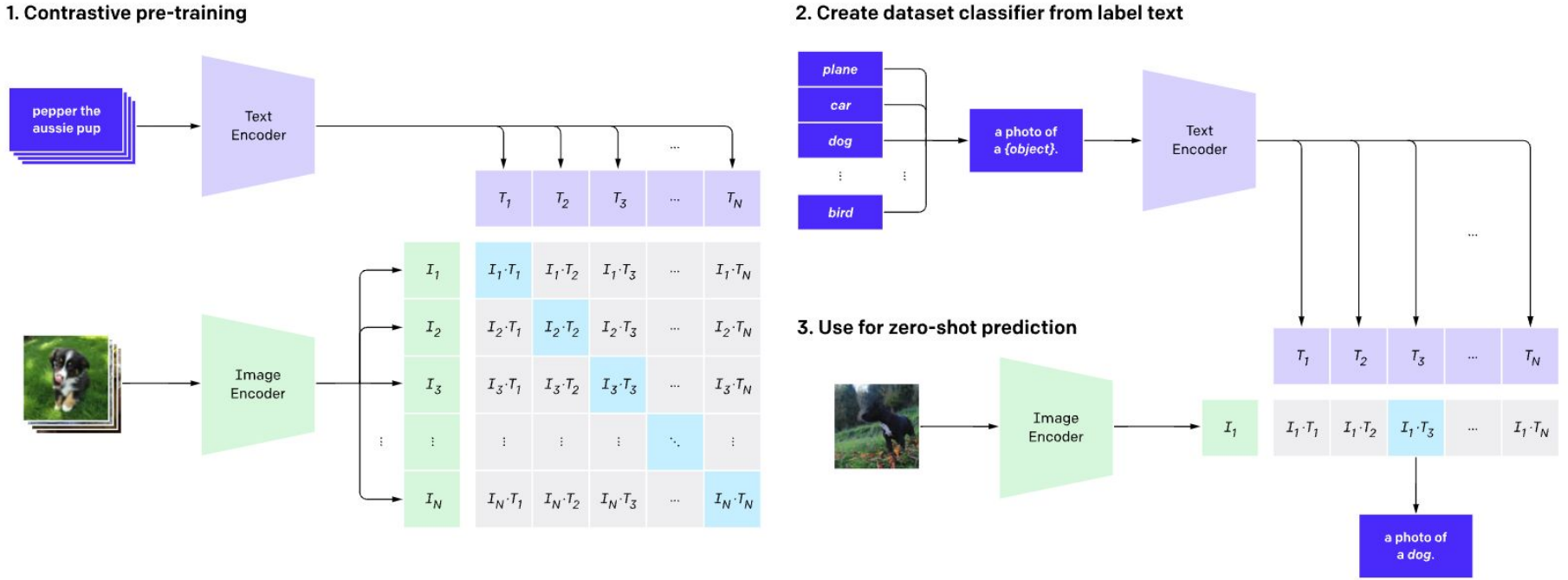
- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

CLIP

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

CLIP


1. Contrastive pre-training
pepper the aussie pup
Text Encoder
T_1 T_2 T_3 ... T_N
Image Encoder
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
2. Create dataset classifier from label text
plane
car
dog
bird
a photo of a {object}.
Text Encoder
3. Use for zero-shot prediction
Image Encoder
I_1
T_1 T_2 T_3 ... T_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
a photo of a dog.

CLIP

Dataset

- Existing text annotated image dataset at the time were relatively small
- YFCC100M
  - Text metadata quality is low, some captions are automatically generated file names like “20160716 113957.JPG”
- Constructed dataset of 400M image-text pairs
- Images searched with one of 500K generated queries

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

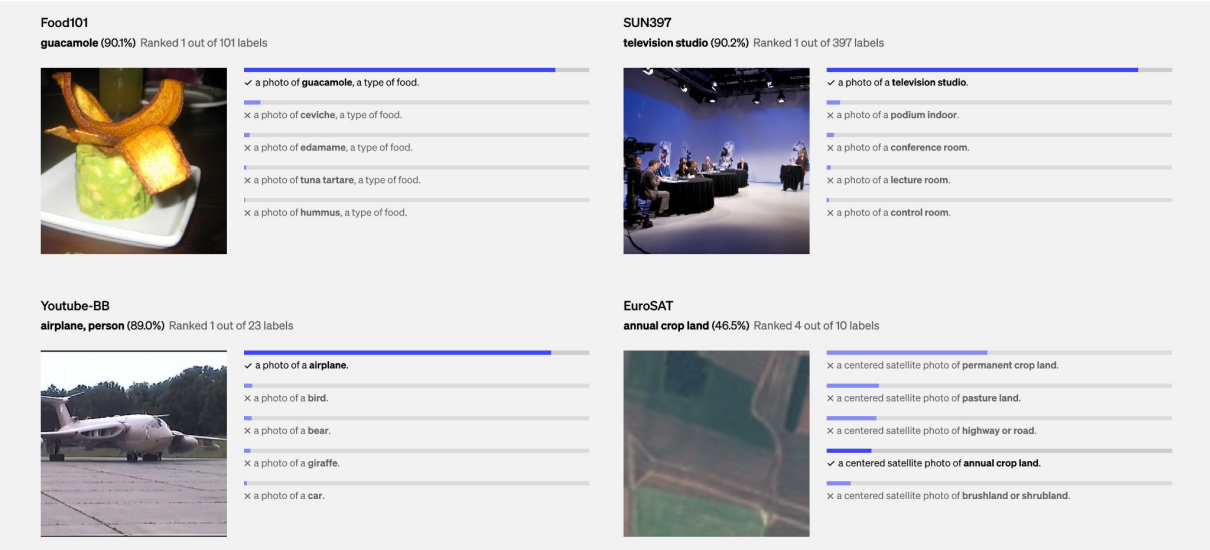
# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

CLIP



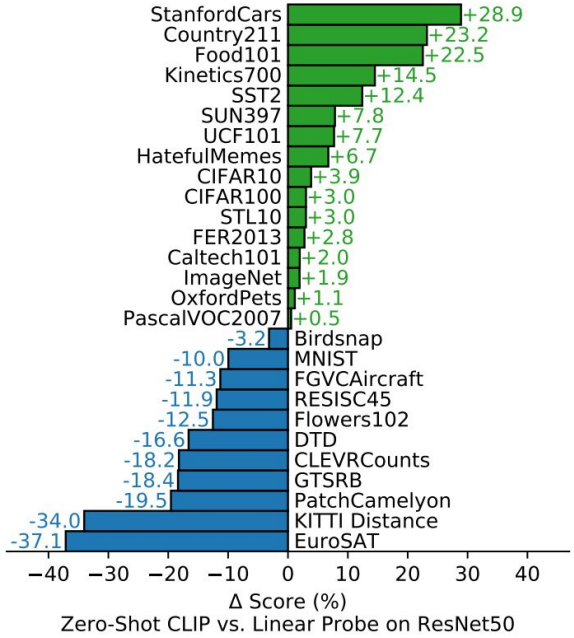
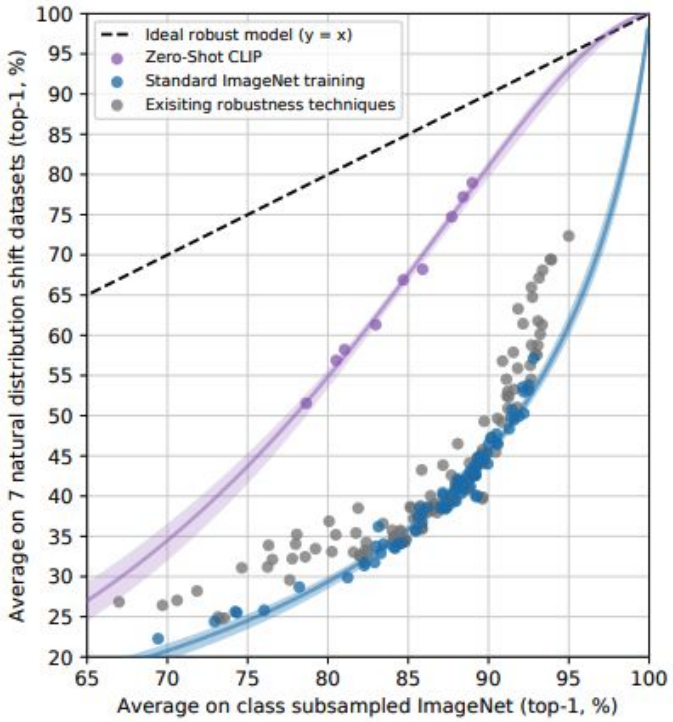
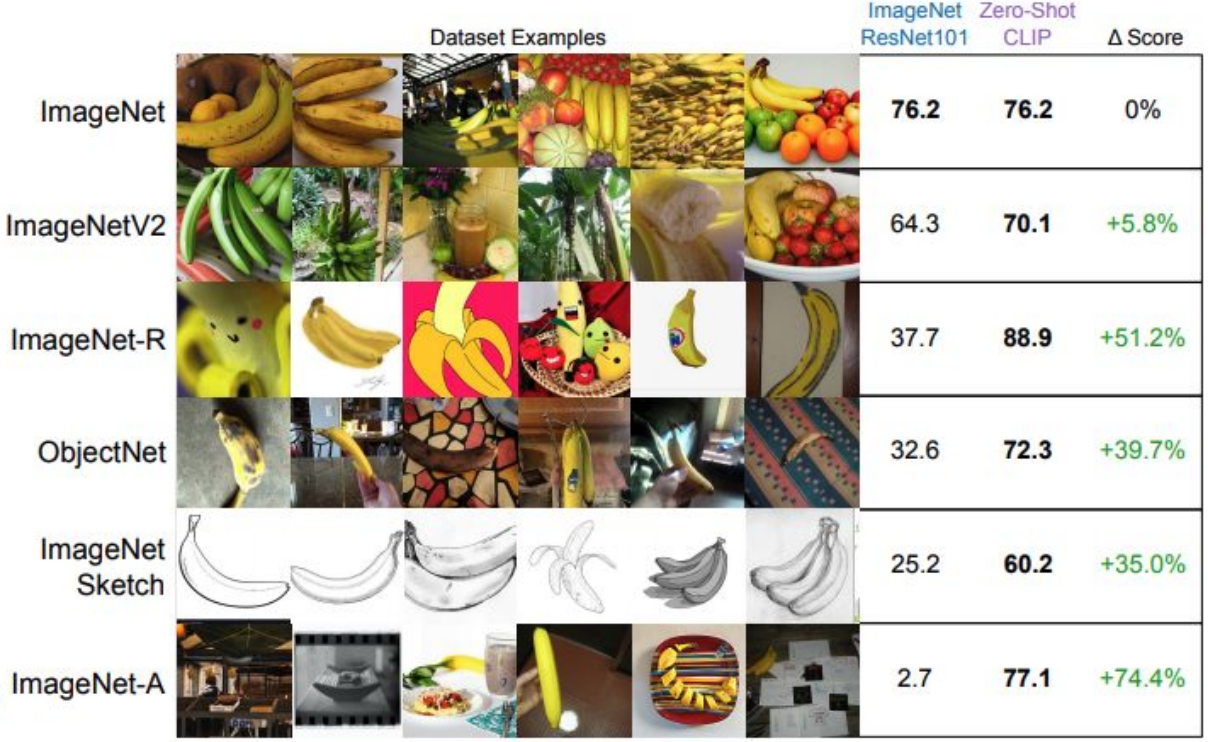


Figure 5. **Zero-shot CLIP is competitive with a fully supervised baseline.** Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet-50 features on 16 datasets, including ImageNet.

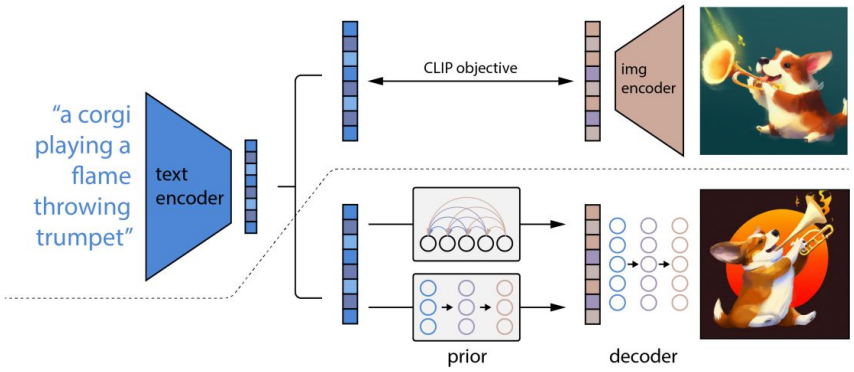
CLIP


Ideal robust model (y = x)
Zero-Shot CLIP
Standard ImageNet training
Exisiting robustness techniques
Average on 7 natural distribution shift datasets (top-1, %)
Average on class subsampled ImageNet (top-1, %)
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
65
70
75
80
85
90
95
100

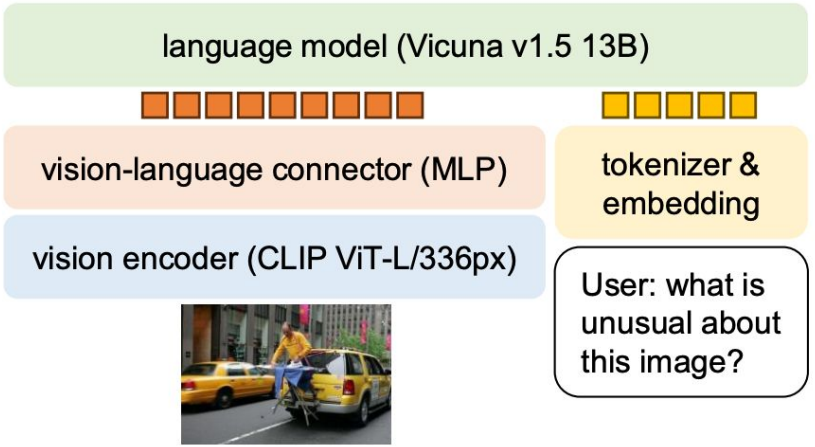

Dataset Examples
ImageNet ResNet101
Zero-Shot CLIP
Δ Score
ImageNet
76.2
76.2
0%
ImageNetV2
64.3
70.1
+5.8%
ImageNet-R
37.7
88.9
+51.2%
ObjectNet
32.6
72.3
+39.7%
ImageNet Sketch
25.2
60.2
+35.0%
ImageNet-A
2.7
77.1
+74.4%

CLIP

CLIP learns features useful for other model



unCLIP

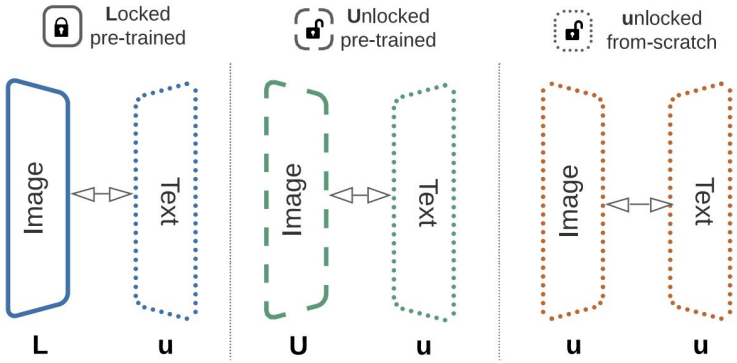


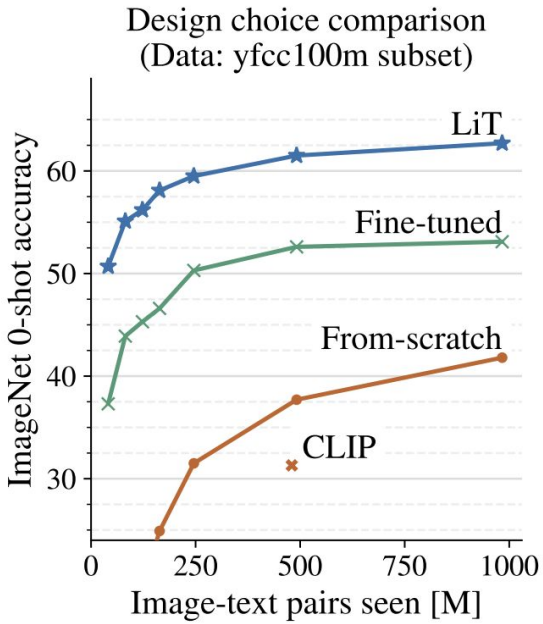
LLaVA

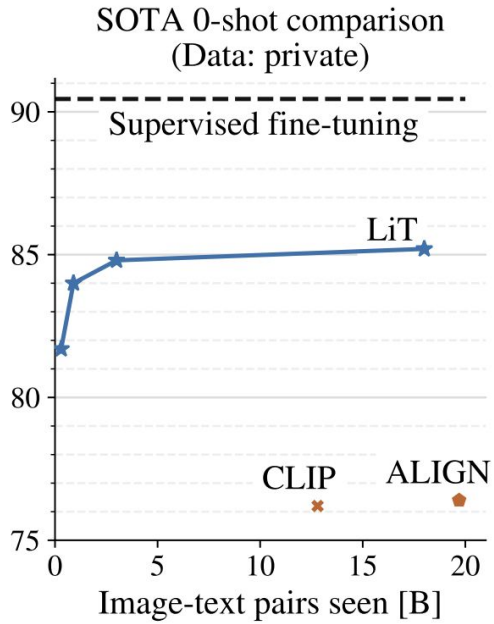
LiT🔥: Zero-Shot Transfer with Locked-image text Tuning

Xiaohua Zhai*† Xiao Wang* Basil Mustafa* Andreas Steiner* Daniel Keysers Alexander Kolesnikov Lucas Beyer*†
Google Research, Brain Team, Zürich

LiT


Locked
pre-trained
Unlocked
pre-trained
unlocked
from-scratch
Image
Text
L
u
U
u
u
u


Design choice comparison
(Data: yfcc100m subset)
ImageNet 0-shot accuracy
LiT
Fine-tuned
From-scratch
CLIP
30
40
50
60
0
250
500
750
1000
Image-text pairs seen [M]


SOTA 0-shot comparison
(Data: private)
Supervised fine-tuning
LiT
CLIP
ALIGN
75
80
85
90
0
5
10
15
20
Image-text pairs seen [B]

LiT

Model: ViT-B/16	Pre-training				LiT		
	Dataset	Labels?	Full IN	10-shot	0-shot	I→T	T→I
MoCo-v3 [11]	IN	n	76.7	60.6	55.4	33.5	17.6
DINO [5]	IN	n	78.2	61.2	55.5	33.4	18.2
AugReg [55]	IN21k	y	77.4	63.9	55.9	30.3	17.2
AugReg [55]	IN	y	77.7	77.1	64.3	25.4	13.8
AugReg [55]	Places	y	-	22.5	28.5	25.1	12.9

Table 3. The role of pre-training method for the image model: as long as it is general, it does not matter. The background coloring denotes whether a value is similar or far away from the others in that column.

We generally perform evaluation on 0-shot ImageNet classification ("0-shot") and MSCOCO image ("T→I") and text ("I→T") retrieval.

LiT

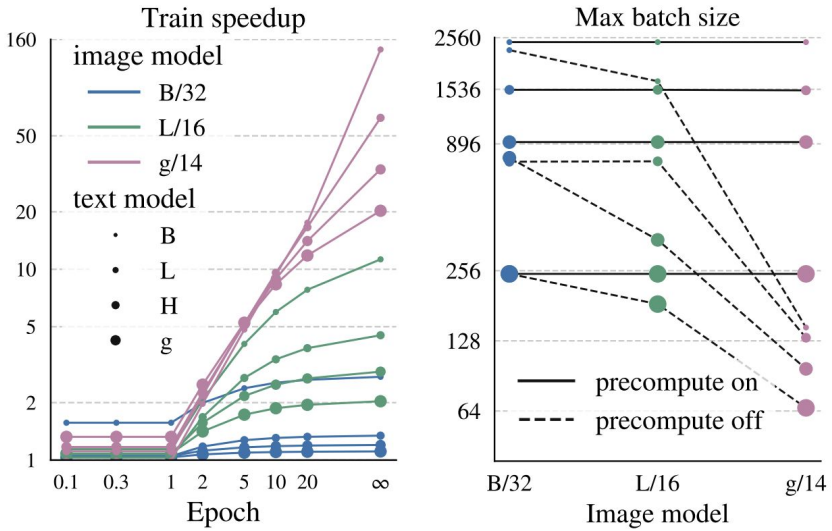


Figure 10. **Left**: Pre-computing image embeddings accelerates LiT, when tuning for more than a single epoch. **Right**: Pre-computing image embeddings in LiT allows larger batch size in memory.

SigLIP

Sigmoid Loss for Language Image Pre-Training

Xiaohua Zhai* Basil Mustafa Alexander Kolesnikov Lucas Beyer*
Google DeepMind, Zürich, Switzerland
{xzhai, basilm, akolesnikov, lbeyer}@google.com

SigLIP

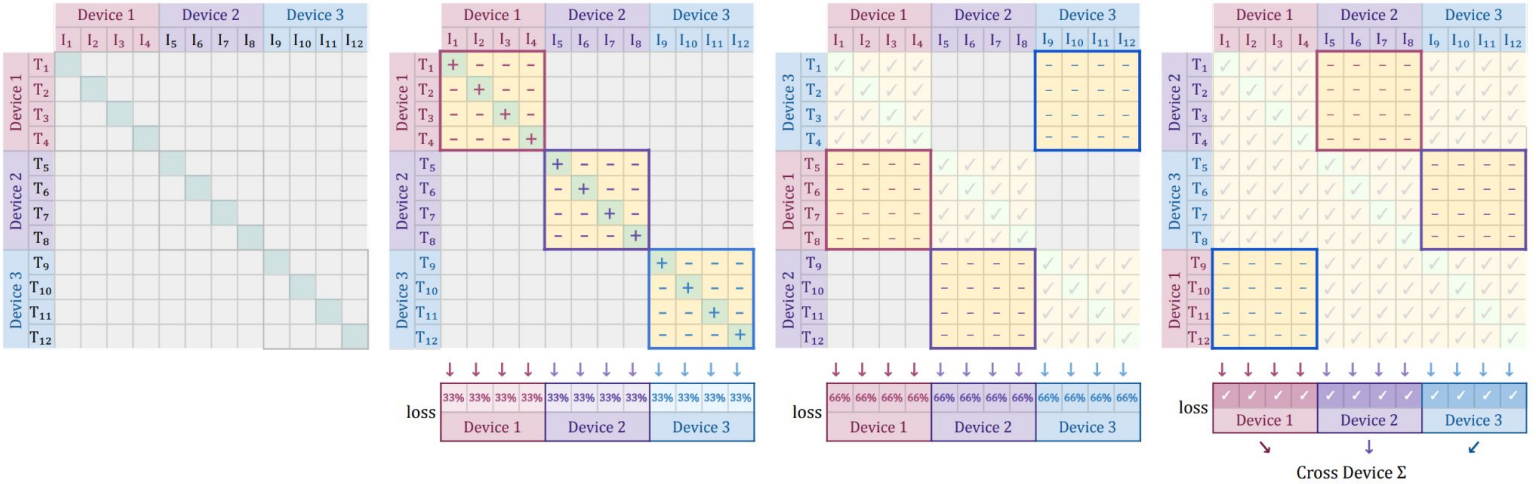
$$-\frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\overbrace{\log\frac{e^{t\mathbf{x}_i\cdot\mathbf{y}_i}}{\sum_{j=1}^{|\mathcal{B}|}e^{t\mathbf{x}_i\cdot\mathbf{y}_j}}}^{\text{image}\to\text{text softmax}}+\overbrace{\log\frac{e^{t\mathbf{x}_i\cdot\mathbf{y}_i}}{\sum_{j=1}^{|\mathcal{B}|}e^{t\mathbf{x}_j\cdot\mathbf{y}_i}}}^{\text{text}\to\text{image softmax}}\right)$$

$$-\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\sum_{j=1}^{|\mathcal{B}|}\underbrace{\log\frac{1}{1+e^{z_{ij}(-t\mathbf{x}_i\cdot\mathbf{y}_j+b)}}}_{\mathcal{L}_{ij}}$$

SigLIP

$$-\frac{1}{|\mathcal{B}|}\underbrace{\sum_{d_i=1}^{D}}_{\text{A: } \forall \text{ device } d_i}\ \underbrace{\sum_{d_j=1}^{D}}_{\text{B: swap negs across devices}}\ \underbrace{\sum_{i=bd_i}^{b(d_i+1)}}_{\text{all local positives}}\ \underbrace{\sum_{j=bd_j}^{b(d_j+1)}}_{\text{negs from next device}}\mathcal{L}_{ij}$$

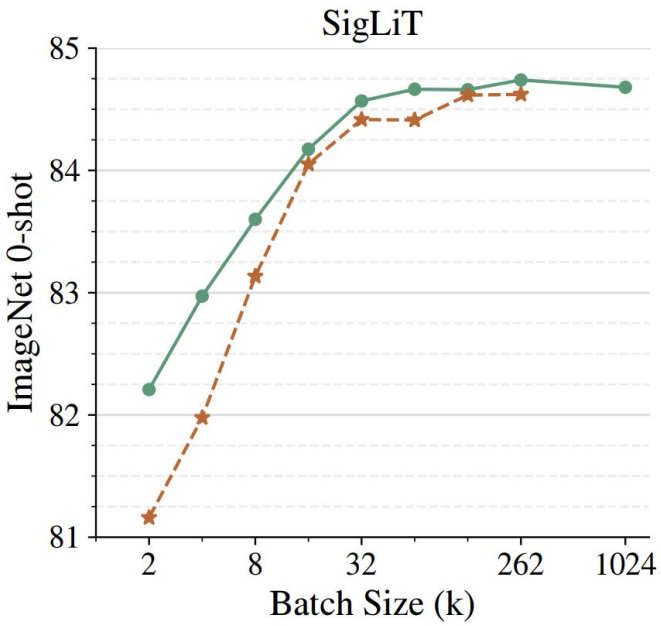
C: per device loss

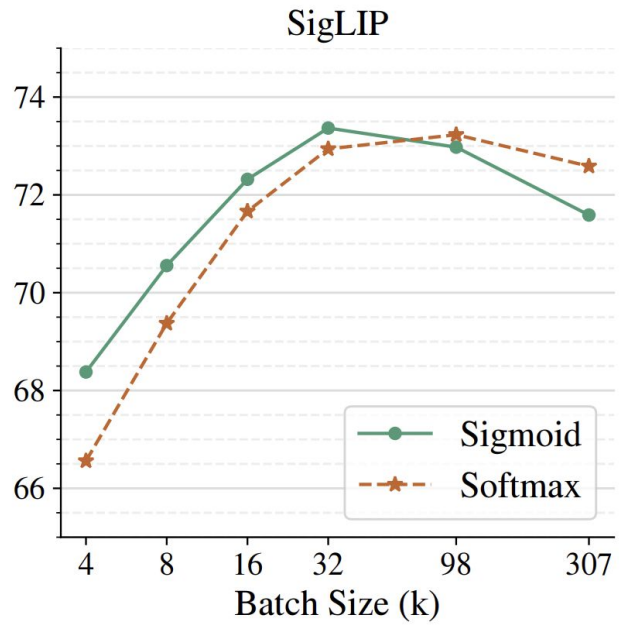


SigLIP

	Image	Text	BS	#TPUv4	Days	INet-0
SigLiT	❄ B/8	L*	32 k	4	1	79.8
SigLiT	❄ g/14	L	20 k	4	2	84.5
SigLIP	🔓 B/16	B	16 k	16	3	71.0
SigLIP	B/16	B	32 k	32	2	72.1
SigLIP	B/16	B	32 k	32	5	73.4

* *We use a variant of the L model with 12 layers.*





FLIP

Scaling Language-Image Pre-training via Masking

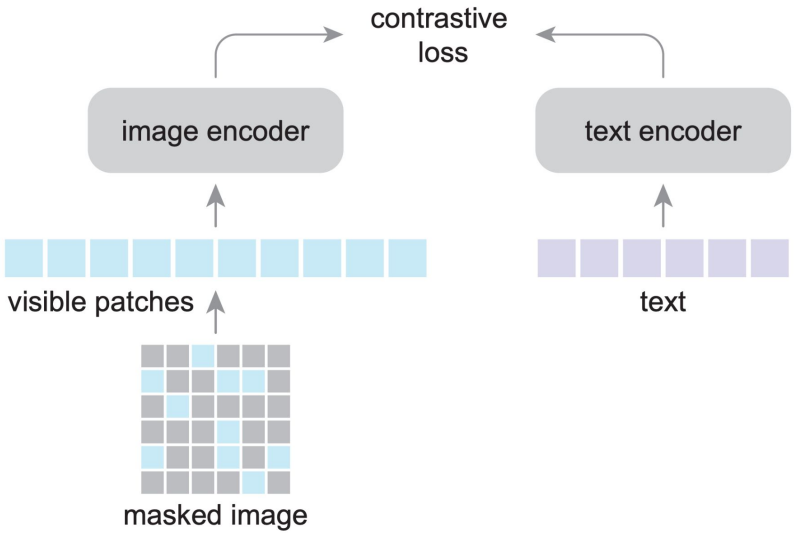
Yanghao Li* Haoqi Fan* Ronghang Hu* Christoph Feichtenhofer† Kaiming He†

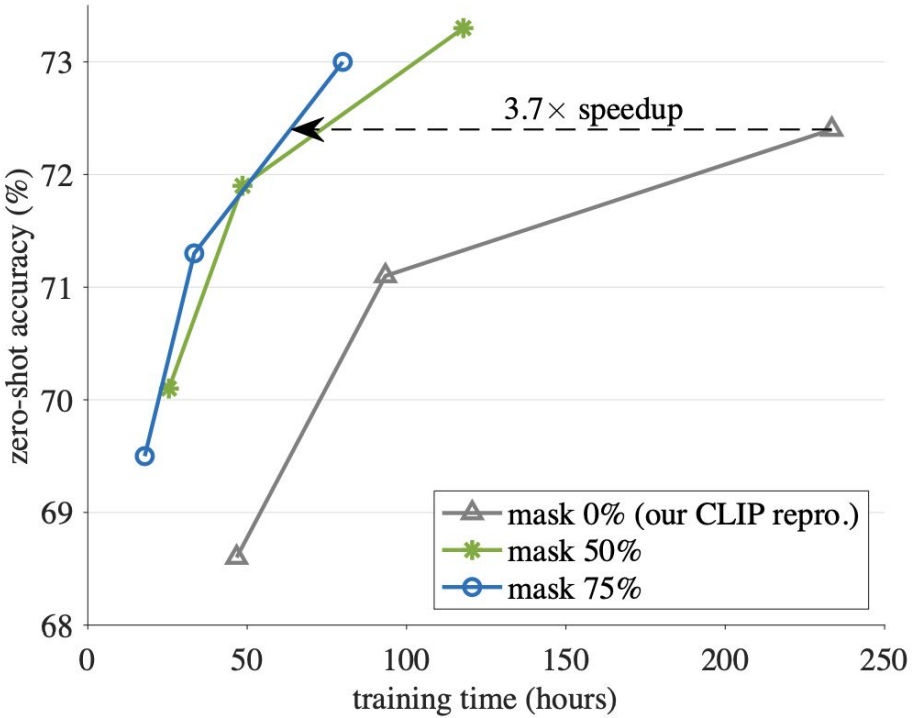
*equal technical contribution, †equal advising

Meta AI, FAIR

https://github.com/facebookresearch/flip

FLIP

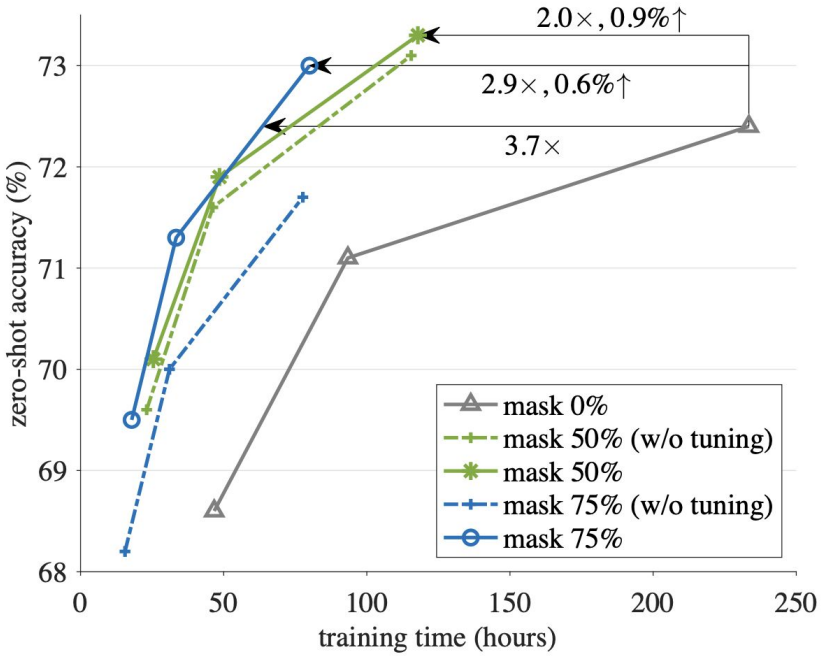

contrastive
loss
image encoder
text encoder
visible patches
text
masked image


3.7× speedup
zero-shot accuracy (%)
training time (hours)
mask 0% (our CLIP repro.)
mask 50%
mask 75%
73
72
71
70
69
68
0
50
100
150
200
250

FLIP

	mask 50%	mask 75%
baseline	69.6	68.2
+ tuning	**70.1**	**69.5**

(e) **Unmasked tuning**. The distribution shift by masking is reduced by a short tuning.



FLIP

case	data	epochs	B/16	L/16	L/14	H/14
CLIP [52]	WIT-400M	32	68.6	-	75.3	-
OpenCLIP [36]	LAION-400M	32	67.1	-	72.8	-
CLIP, our repro.	LAION-400M	32	68.2	72.4	73.1	-
FLIP	LAION-400M	32	68.0	74.3	74.6	75.5

Table 2. **Zero-shot accuracy on ImageNet-1K classification**, compared with various CLIP baselines. The image size is 224. The entries noted by grey are pre-trained on a different dataset. Our models use a 64k batch, 50% masking ratio, and unmasked tuning.

case	data	epochs	model	zero-shot	linear probe	fine-tune
CLIP [52]	WIT-400M	32	L/14	75.3	83.9†	-
CLIP [52], our transfer	WIT-400M	32	L/14	75.3	83.0	87.4
OpenCLIP [36]	LAION-400M	32	L/14	72.8	82.1	86.2
CLIP, our repro.	LAION-400M	32	L/16	72.4	82.6	86.3
FLIP	LAION-400M	32	L/16	74.3	83.6	86.9

Table 3. **Linear probing and fine-tuning accuracy on ImageNet-1K classification**, compared with various CLIP baselines. The entries noted by grey are pre-trained on a different dataset. The image size is 224. †: CLIP in [52] optimizes with L-BFGS; we use SGD instead.

SLIP

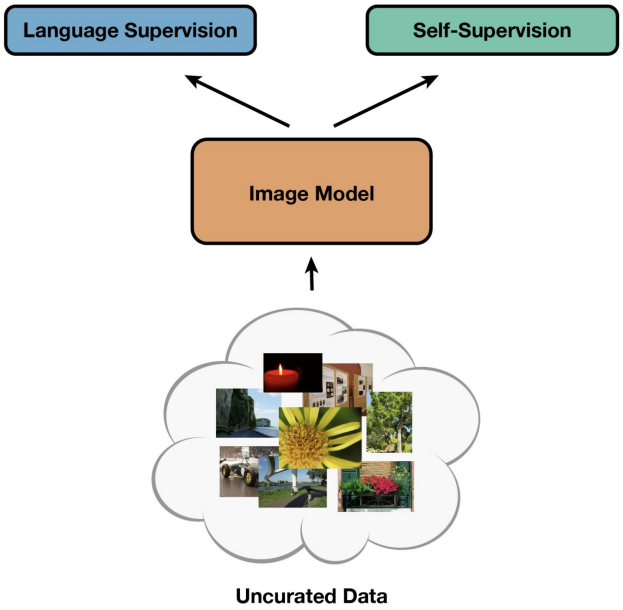
SLIP: Self-supervision meets Language-Image Pre-training

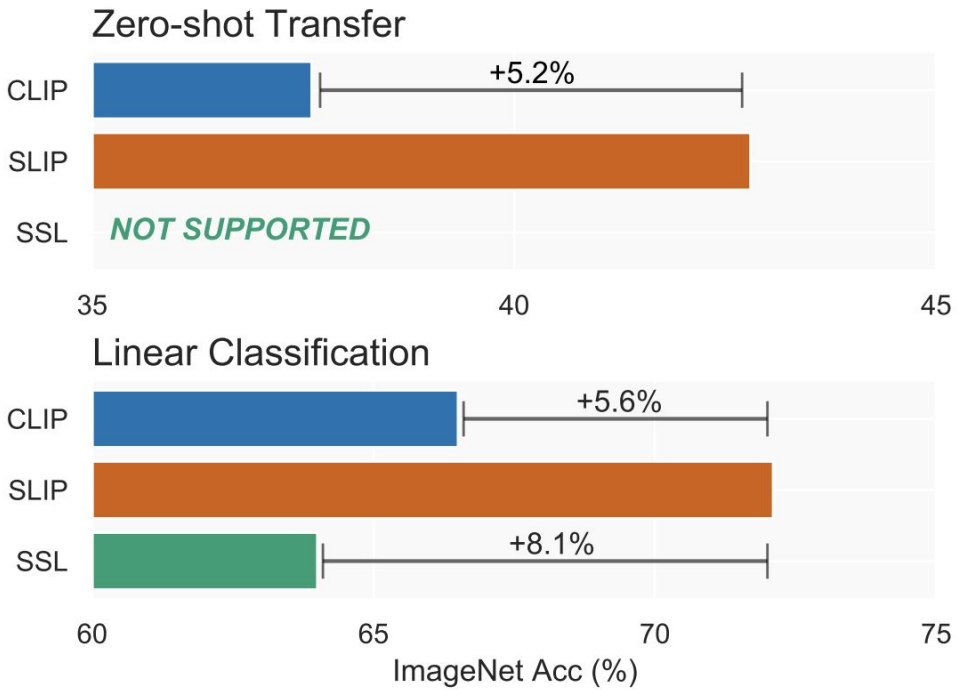
Norman Mu[1] Alexander Kirillov[2] David Wagner[1] Saining Xie[2]

[1]UC Berkeley, [2]Facebook AI Research (FAIR)

Code: https://github.com/facebookresearch/SLIP

SLIP


Language Supervision
Self-Supervision
Image Model
Uncurated Data


Zero-shot Transfer
CLIP
SLIP
SSL
+5.2%
NOT SUPPORTED
35
40
45
Linear Classification
CLIP
SLIP
SSL
+5.6%
+8.1%
60
65
70
75
ImageNet Acc (%)

SLIP

Algorithm 1 SLIP-SimCLR: PyTorch-like Pseudocode

```
# fi, ft: image, text encoders
# hi, ht: CLIP image, text projectors
# hs: SimCLR projector
# c: SimCLR loss scale
def forward(img, text):
   xi, x1, x2 = crop(img), aug(img), aug(img)
   yt = tokenize(text)

   wi, w1, w2 = fi(xi, x1, x2)
   wt = ft(yt)

   z1, z2 = hs(w1), hs(w2) # SSL embed: N x C2
   zi, zt = hi(wi), ht(wt) # CLIP embed: N x C1

   loss = c * simclr(z1, z2) + clip(zi, zt)
   return loss
```

```
# s: learnable log logit scale
def clip(zi, zt):
   zi, zt = normalize(zi, zt)
   label = range(N)
   logit = exp(s) * zi @ zt.T

   li = CrossEntropy(logit, label)
   lt = CrossEntropy(logit.T, label)

   loss = (li + lt) / 2
   return loss

# tau: softmax temperature
def simclr(z1, z2):
   z1, z2 = normalize(z1, z2)
   label = range(N)
   mask = eye(N) * 1e9

   logit = z1 @ z2.T
   logit1 = z1 @ z1.T - mask
   logit2 = z2 @ z2.T - mask

   logit1 = cat(logit, logit1)
   logit2 = cat(logit.T, logit2)

   l1 = CrossEntropy(logit1 / tau)
   l2 = CrossEntropy(logit2 / tau)

   loss = (l1 + l2) / 2
   return loss
```

SLIP

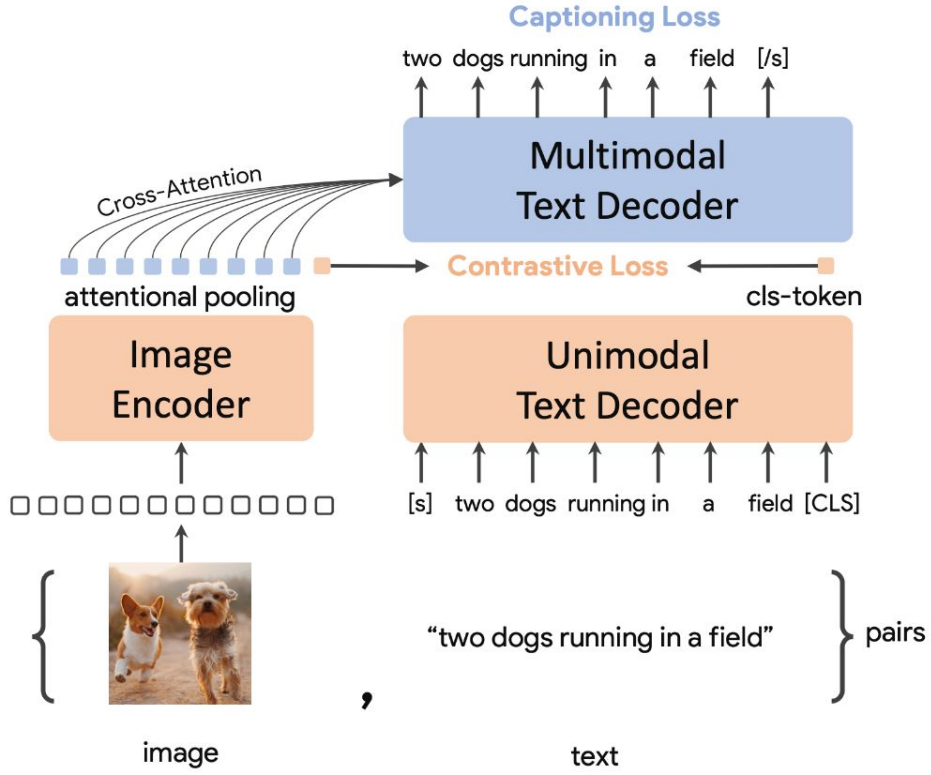
Model	Method	0-shot	Linear	Finetuned
ViT-S/16	CLIP	32.7	59.3	78.2
	SimCLR	-	58.1	79.9
	SLIP	**38.3** (+5.6)	**66.4** (+7.1)	**80.3** (+0.4)
ViT-B/16	CLIP	37.6	66.5	80.5
	SimCLR	-	64.0	82.5
	SLIP	**42.8** (+5.2)	**72.1** (+5.6)	**82.6** (+0.1)
ViT-L/16	CLIP	40.4	70.5	81.0
	SimCLR	-	66.7	84.0
	SLIP	**46.2** (+4.8)	**76.0** (+5.5)	**84.2** (+0.2)

CoCa

CoCa: Contrastive Captioners are Image-Text Foundation Models

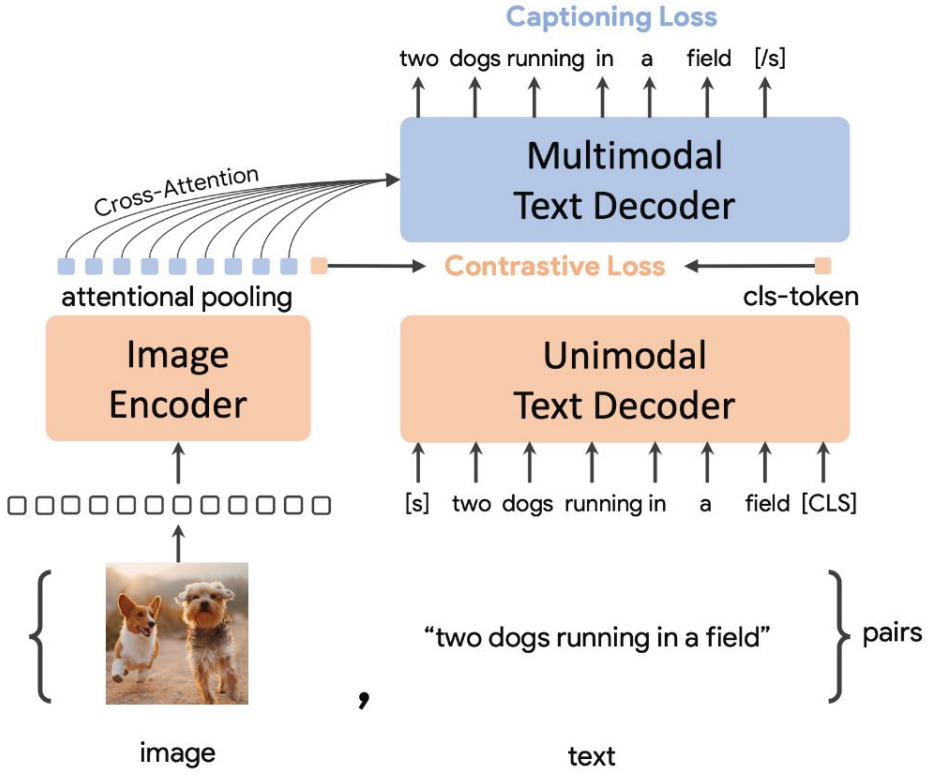
Jiahui Yu[†] **Zirui Wang**[†]
{jiahuiyu, ziruiw}@google.com
Vijay Vasudevan **Legg Yeung** **Mojtaba Seyedhosseini** **Yonghui Wu**
Google Research

CoCa



- Contrastive learning with conventional encoder decoder transformer
- Achieving 91% top 1 ImageNet accuracy post fine tuning

CoCa



Algorithm 1 Pseudocode of Contrastive Captioners architecture.

```
# image, text.ids, text.labels, text.mask: paired {image, text} data
# con_query: 1 query token for contrastive embedding
# cap_query: N query tokens for captioning embedding
# cls_token_id: a special cls_token_id in vocabulary

def attentional_pooling(features, query):
  out = multihead_attention(features, query)
  return layer_norm(out)

img_feature = vit_encoder(image) # [batch, seq_len, dim]
con_feature = attentional_pooling(img_feature, con_query) # [batch, 1, dim]
cap_feature = attentional_pooling(img_feature, cap_query) # [batch, N, dim]

ids = concat(text.ids, cls_token_id)
mask = concat(text.mask, zeros_like(cls_token_id)) # unpad cls_token_id
txt_embs = embedding_lookup(ids)
unimodal_out = lm_transformers(txt_embs, mask, cross_attn=None)
multimodal_out = lm_transformers(
    unimodal_out[:, :-1, :], mask, cross_attn=cap_feature)
cls_token_feature = layer_norm(unimodal_out)[:, -1:, :] # [batch, 1, dim]

con_loss = contrastive_loss(con_feature, cls_token_feature)
cap_loss = softmax_cross_entropy_loss(
    multimodal_out, labels=text.labels, mask=text.mask)
```

`vit_encoder`: vision transformer based encoder; `lm_transformer`: language-model transformers.

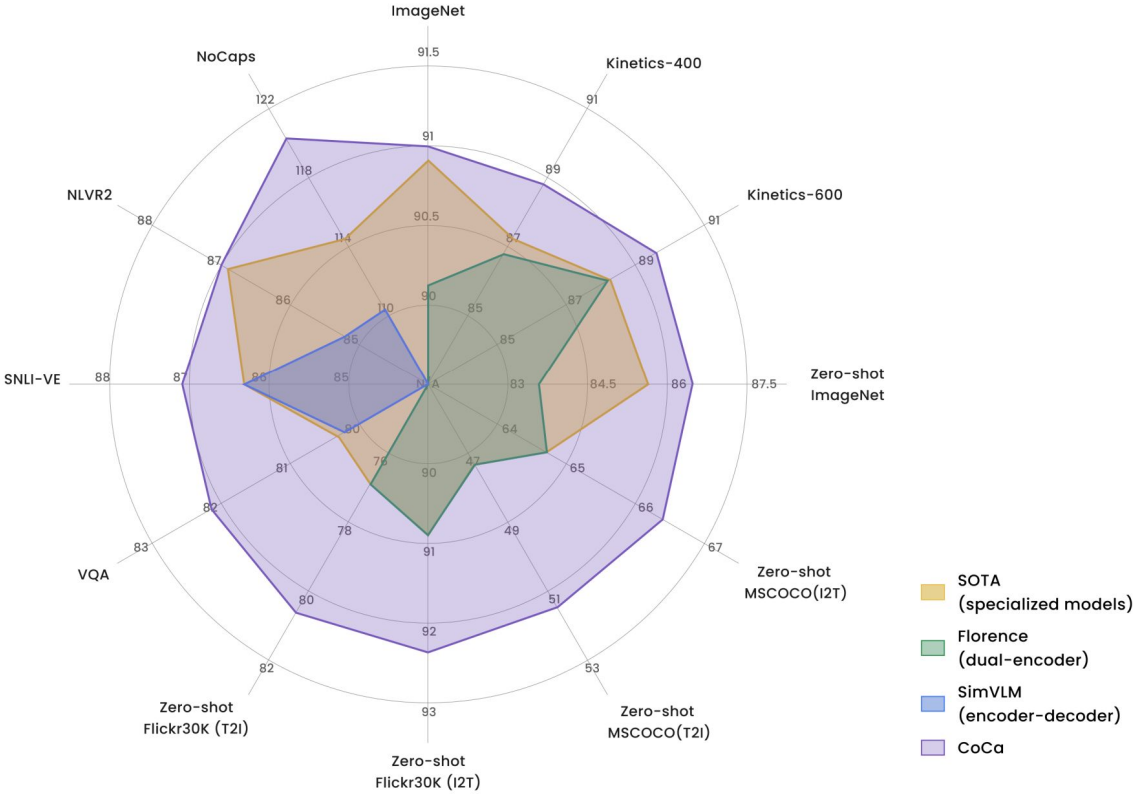
CoCa

$$\mathcal{L}_{\text{Con}} = -\frac{1}{N}\Big(\underbrace{\sum_{i}^{N} \log \frac{\exp(x_i^\top y_i/\sigma)}{\sum_{j=1}^{N}\exp(x_i^\top y_j/\sigma)}}_{\text{image-to-text}} + \underbrace{\sum_{i}^{N} \log \frac{\exp(y_i^\top x_i/\sigma)}{\sum_{j=1}^{N}\exp(y_i^\top x_j/\sigma)}}_{\text{text-to-image}}\Big), \tag{2}$$

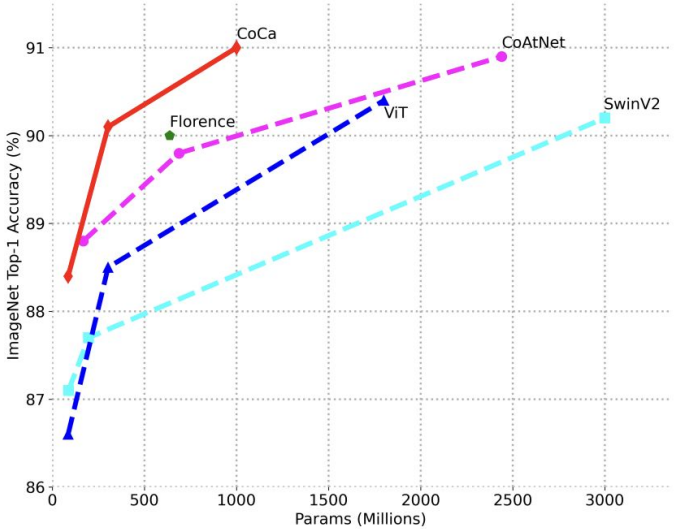
$$\mathcal{L}_{\text{Cap}} = -\sum_{t=1}^{T} \log P_\theta(y_t | y_{<t}, x).$$

CoCa

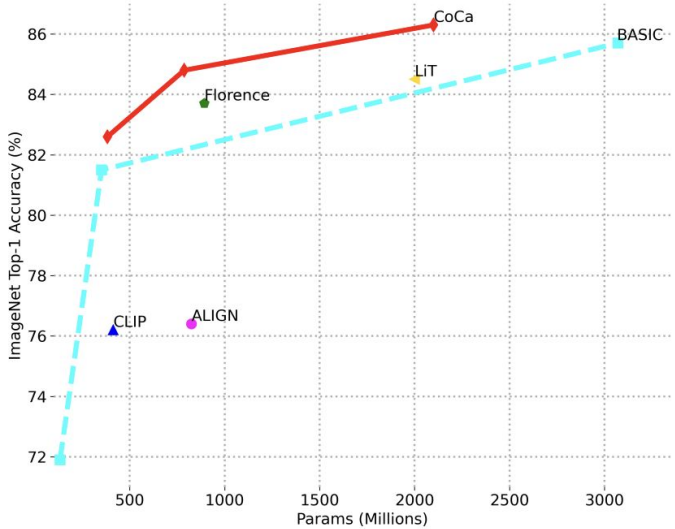
- Trained from scratch on JFT-3B dataset and ALIGN dataset
- JFT is a (internal) Google classification benchmark
  - Randomly sample a caption from a templated prompt
  - Ie “a photo of the cat, animal”



CoCa



(a) Finetuned ImageNet Top-1 Accuracy.



(b) Zero-Shot ImageNet Top-1 Accuracy.

Figure 5: Image classification scaling performance of model sizes.

CoCa

	Flickr30K (1K test set)						MSCOCO (5K test set)					
	Image → Text			Text → Image			Image → Text			Text → Image		
Model	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
CLIP [12]	88.0	98.7	99.4	68.7	90.6	95.2	58.4	81.5	88.1	37.8	62.4	72.2
ALIGN [13]	88.6	98.7	99.7	75.7	93.8	96.8	58.6	83.0	89.7	45.6	69.8	78.6
FLAVA [35]	67.7	94.0	-	65.2	89.4	-	42.7	76.8	-	38.4	67.5	-
FILIP [61]	89.8	99.2	99.8	75.0	93.4	96.3	61.3	84.3	90.4	45.9	70.6	79.3
Florence [14]	90.9	99.1	-	76.7	93.6	-	64.7	85.9	-	47.2	71.4	-
CoCa-Base	89.8	98.8	99.8	76.8	93.7	96.8	63.8	84.7	90.7	47.5	72.4	80.9
CoCa-Large	91.4	99.2	99.9	79.0	95.1	97.4	65.4	85.6	91.4	50.1	73.8	81.8
CoCa	**92.5**	**99.5**	**99.9**	**80.4**	**95.7**	**97.7**	**66.3**	**86.2**	**91.8**	**51.2**	**74.2**	**82.0**

Table 3: Zero-shot image-text retrieval results on Flickr30K [62] and MSCOCO [63] datasets.

CoCa

Model	ImageNet
ALIGN [13]	88.6
Florence [14]	90.1
MetaPseudoLabels [51]	90.2
CoAtNet [10]	90.9
ViT-G [21]	90.5
+ Model Soups [52]	90.9
CoCa (frozen)	90.6
CoCa (finetuned)	**91.0**

Model	K-400	K-600	K-700	Moments-in-Time
ViViT [53]	84.8	84.3	-	38.0
MoViNet [54]	81.5	84.8	79.4	40.2
VATT [55]	82.1	83.6	-	41.1
Florence [14]	86.8	88.0	-	-
MaskFeat [56]	87.0	88.3	80.4	
CoVeR [11]	87.2	87.9	78.5	46.1
CoCa (frozen)	88.0	88.5	81.1	47.4
CoCa (finetuned)	**88.9**	**89.4**	**82.7**	**49.0**

Table 2: Image classification and video action recognition with frozen encoder or finetuned encoder.

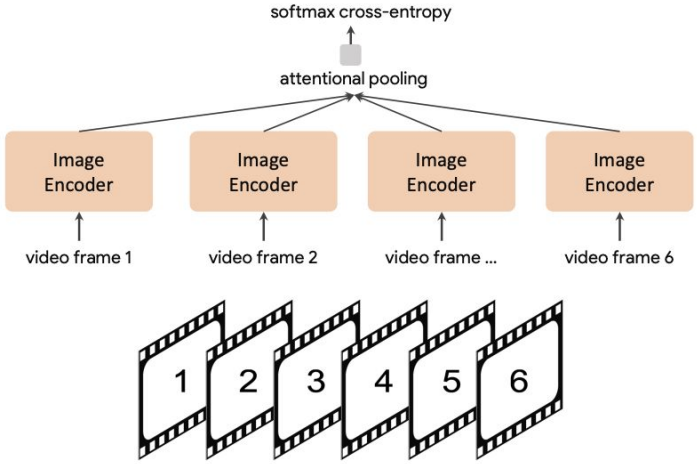


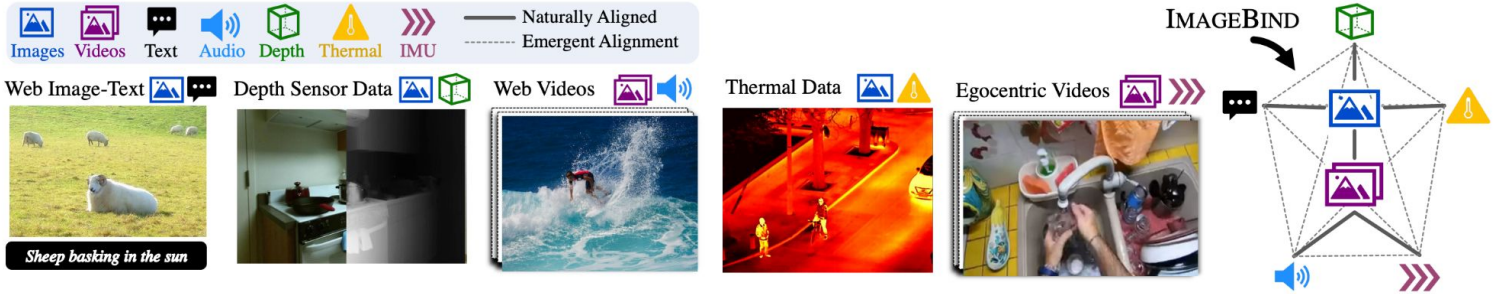
Figure 3: CoCa for video recognition.

ImageBind

IMAGEBIND: One Embedding Space To Bind Them All

Rohit Girdhar* Alaaeldin El-Nouby* Zhuang Liu Mannat Singh
Kalyan Vasudev Alwala Armand Joulin Ishan Misra*
FAIR, Meta AI

https://facebookresearch.github.io/ImageBind

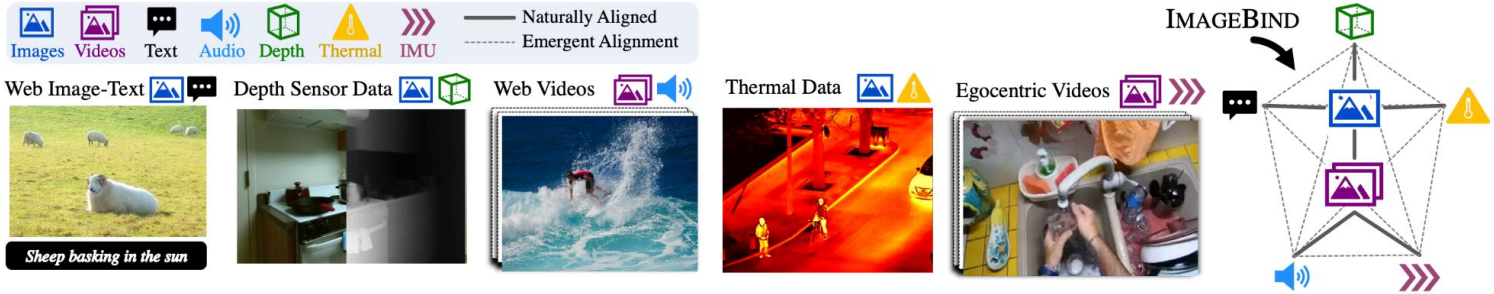


ImageBind

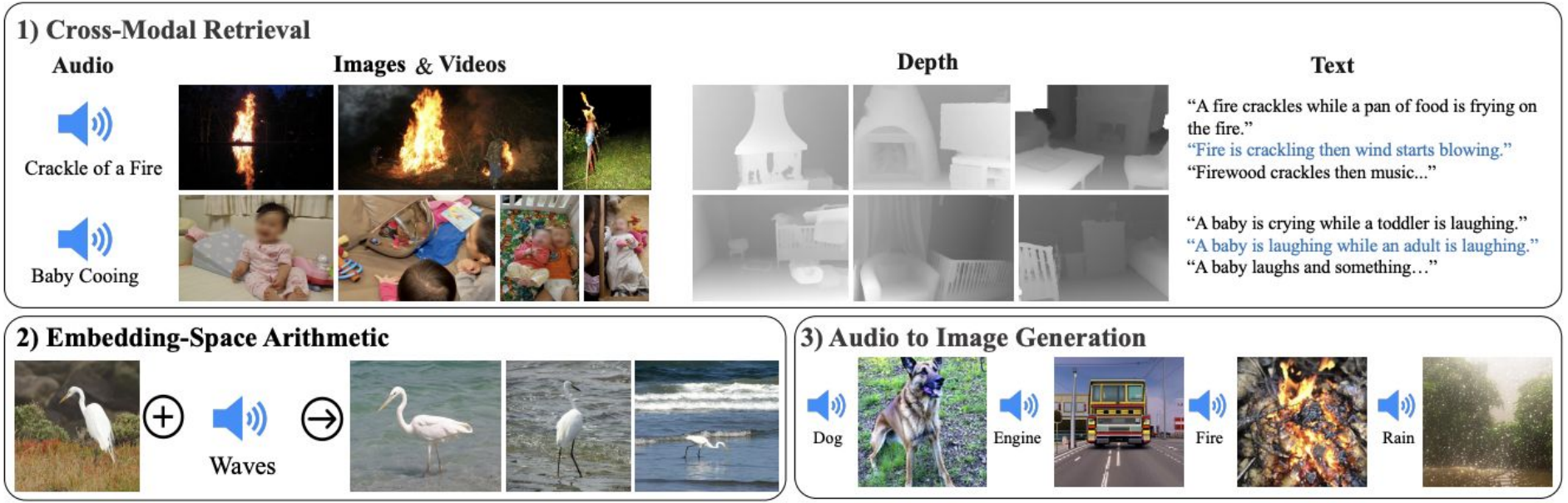
Train with (Image, Modality) pairs

Transformer for all modality encoders

Train with InfoNCE loss



ImageBind


1) Cross-Modal Retrieval
Audio
Images & Videos
Depth
Text
Crackle of a Fire
Baby Cooing
"A fire crackles while a pan of food is frying on the fire."
"Fire is crackling then wind starts blowing."
"Firewood crackles then music..."
"A baby is crying while a toddler is laughing."
"A baby is laughing while an adult is laughing."
"A baby laughs and something…"
2) Embedding-Space Arithmetic
Waves
3) Audio to Image Generation
Dog
Engine
Fire
Rain

BLIP-2

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models

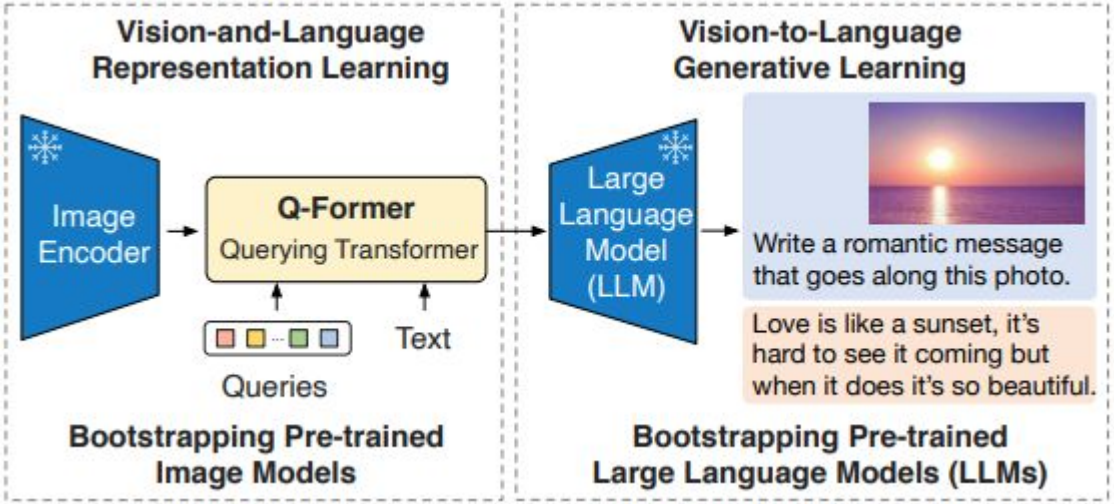
Junnan Li Dongxu Li Silvio Savarese Steven Hoi
Salesforce Research

https://github.com/salesforce/LAVIS/tree/main/projects/blip2

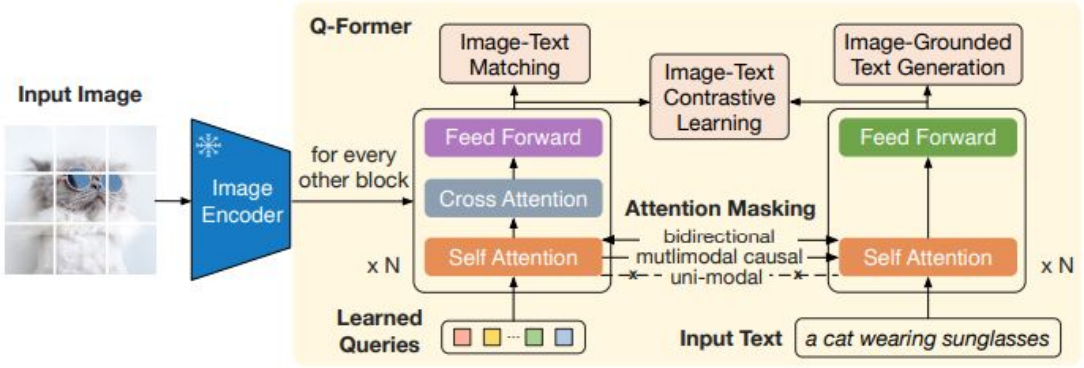
- Goal: Efficient, zero-shot capabilities in vision-language tasks
- Pitfalls:
  - end-to-end training on vision-language models is expensive
  - Finetuning from pretrained LLMs or ViTs can result in catastrophic forgetting
  - Aligning image and language modalities is hard

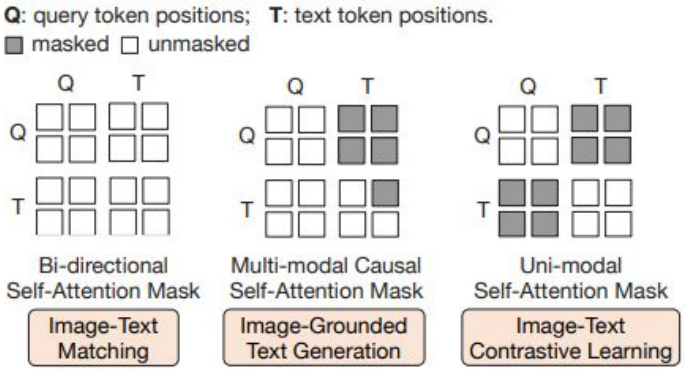
BLIP-2

- Solution: Q-Former
  - Learn the image-language modality alignment, **then** finetune for language generation
- **2-stage pretraining:** vision-language representation learning, vision-to-language generative learning



BLIP-2



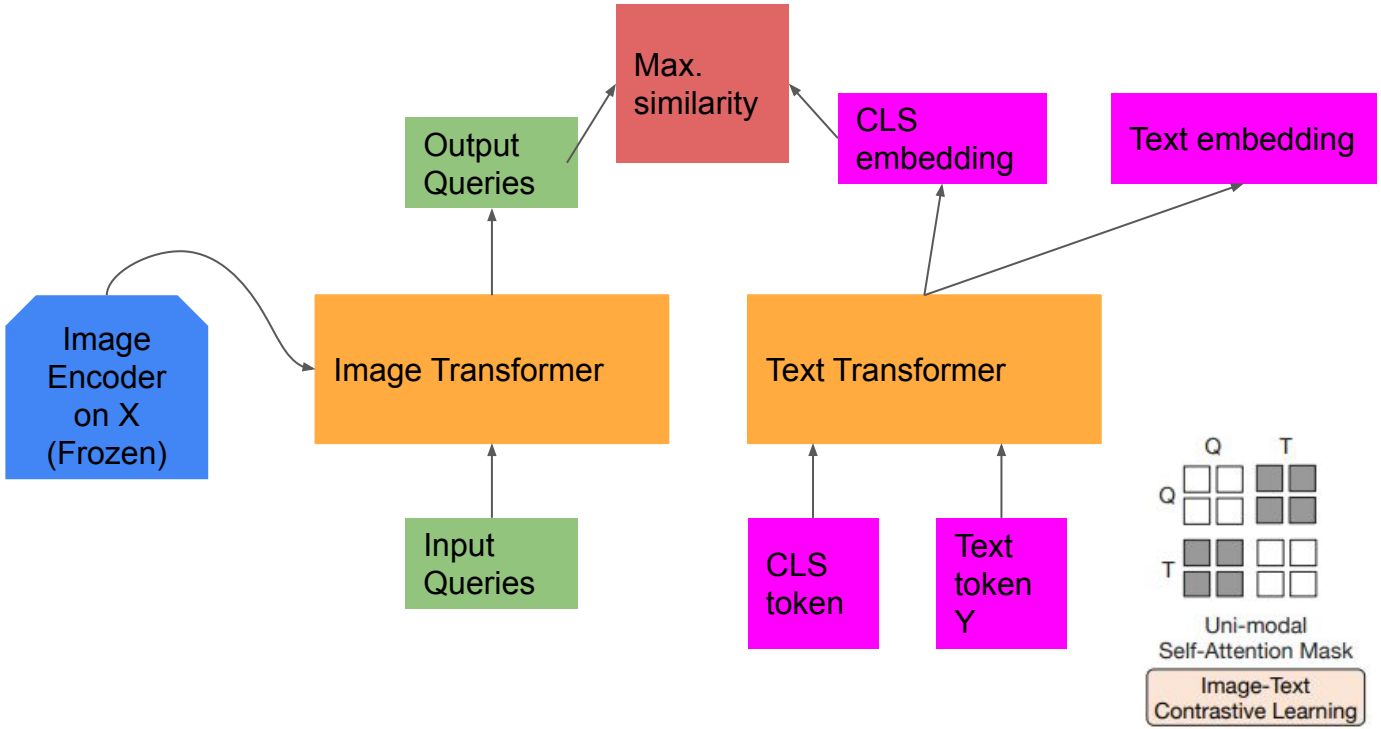


- Representation learning just focuses on this block
- Three objectives: Image-Grounded Text Generation (**ITG**), Image-Text Matching **(ITM)**, Image-Text Contrastive Learning (**ITC**)
- Image encoder alternatively cross attends, queries and text self-attend
- Delineate image X, text Y as an image-text pair (here, image + caption)

BLIP-2

ITC (Image-Text Contrastive Learning)

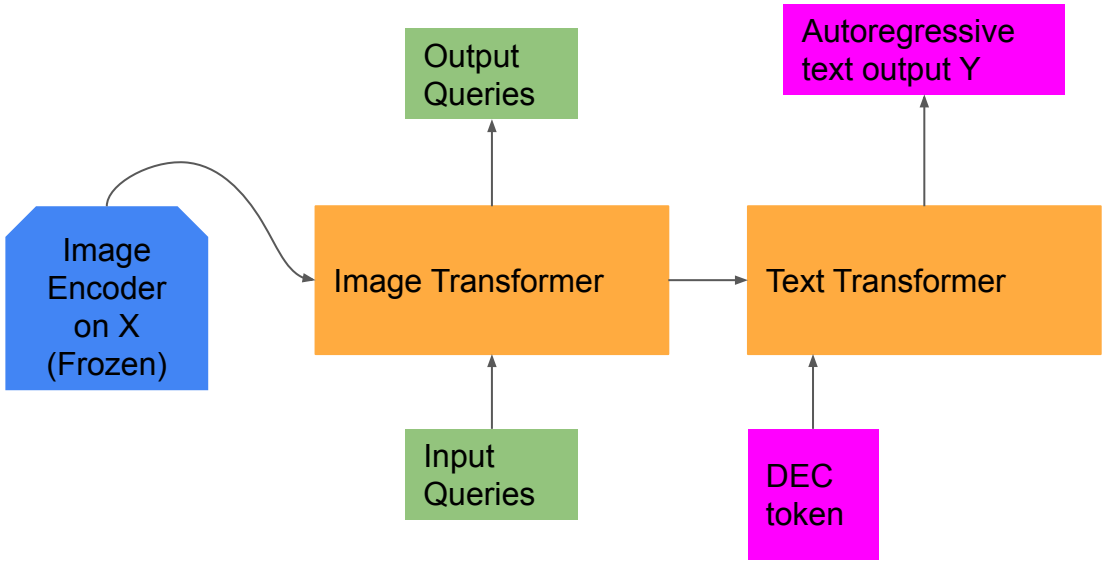
- Maximum similarity computed among queries **and taken as text-image similarity**
- Similarities **contrasted** between positive image-text pairs and negatives in-batch
- **Result:** aligning relevant image-text modalities

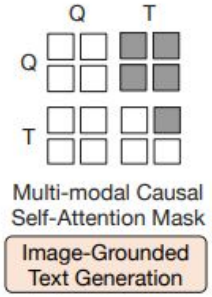


BLIP-2

ITG (Image-Grounded Text Generation)

- Queries attend on one another, separate from text transformer
- Text transformer takes in decoder token and conditions on queries to do text generation (using as a target the same text label as in ITC)

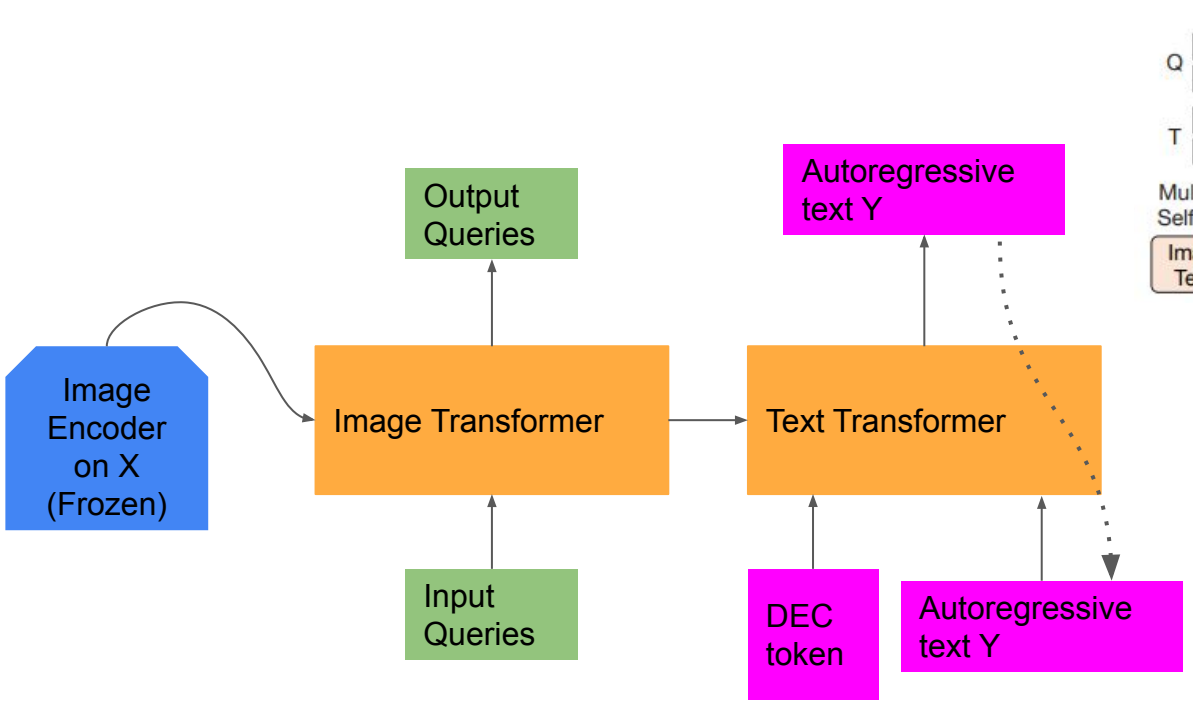




BLIP-2

ITG (Image-Grounded Text Generation)

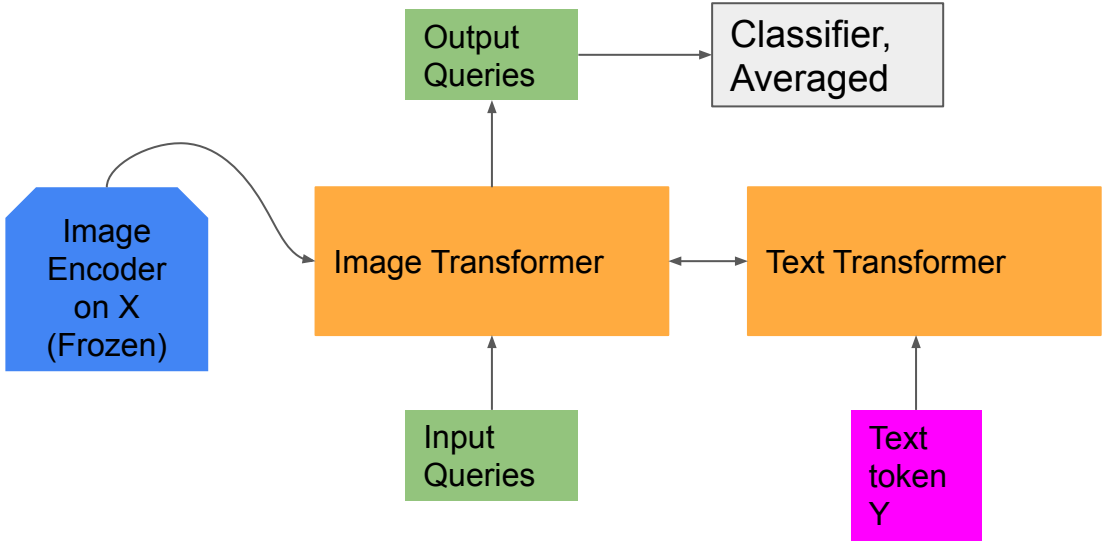
- Text labels are generated autoregressively - resulting attention mask shown above
- Train on text generation loss
- **Result:** creating informative vision-language tokens

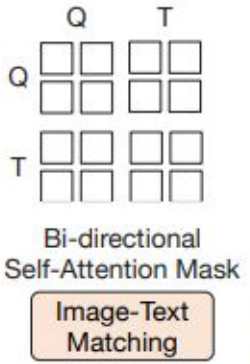


BLIP-2

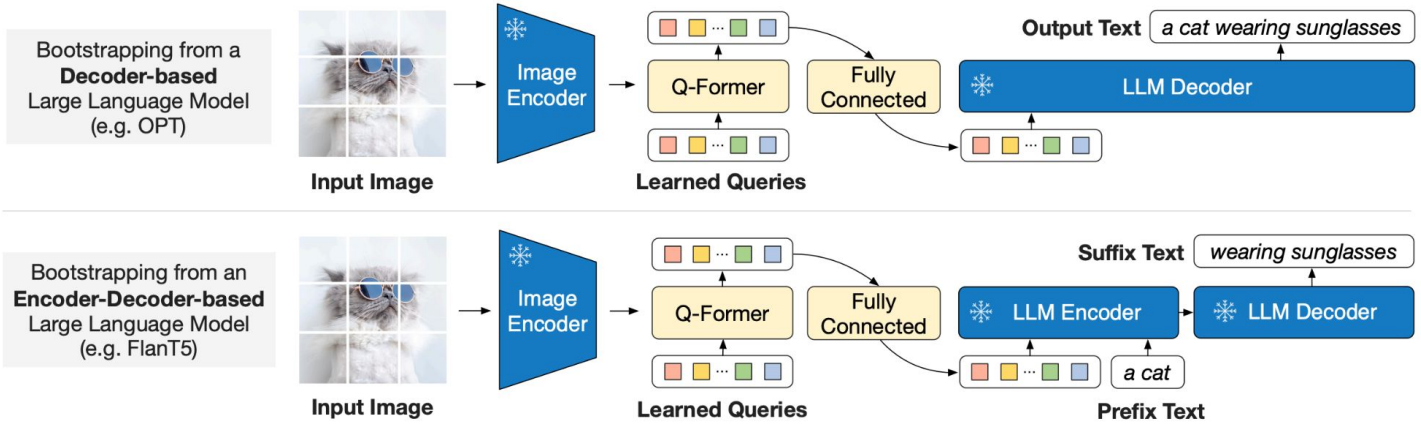
ITM (Image-Text Matching)

- Queries and text attend on one another unrestricted
- Output queries classified and averaged, trained on classifying (X,Y) as a pair or not
- **Hard negative mining** utilized
- **Result:** fine-grained representation learning





BLIP-2



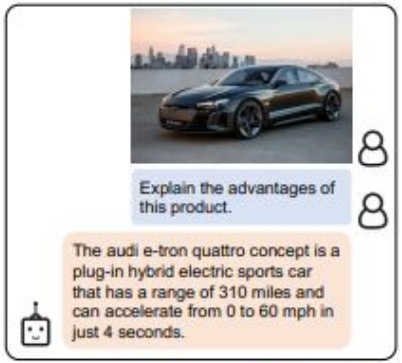
- Second pretraining stage: use X as input to encoder, Y as output from LLM Decoder
- Q-Former is finetuned on the same data, along with a projection layer between Q-Former and the LLM Decoder (language modeling loss)

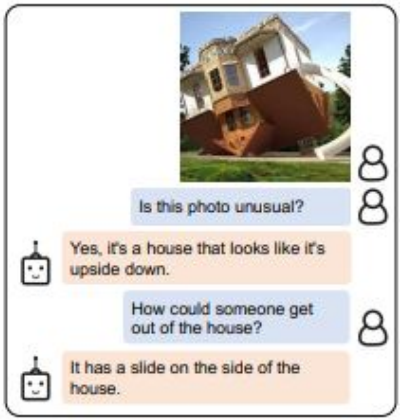
BLIP-2

Training

- Trained on 129 million image-caption pairs
  - COCO, Visual Genome, LAION400M, etc.
- Used CapFilt to generate synthetic captions from BLIP-1 and filter on caption similarity with CLIP embeddings
- Random augmentations: cropping, horizontal flipping

BLIP-2





Models	#Trainable Params	Open-sourced?	**Visual Question Answering** VQAv2 (test-dev) VQA acc.	**Image Captioning** NoCaps (val) CIDEr	SPICE	**Image-Text Retrieval** Flickr (test) TR@1	IR@1
BLIP (Li et al., 2022)	583M	✓	-	113.2	14.8	96.7	86.7
SimVLM (Wang et al., 2021b)	1.4B	✗	-	112.2	-	-	-
BEIT-3 (Wang et al., 2022b)	1.9B	✗	-	-	-	94.9	81.5
Flamingo (Alayrac et al., 2022)	10.2B	✗	56.3	-	-	-	-
BLIP-2	188M	✓	**65.0**	**121.6**	**15.8**	**97.6**	**89.7**

- BLIP shows notable improvements especially in zero-shot visual question answering
- Image-text retrieval doesn't utilize language generation stage (so only the BLIP visual encoder gets used)

BLIP-2

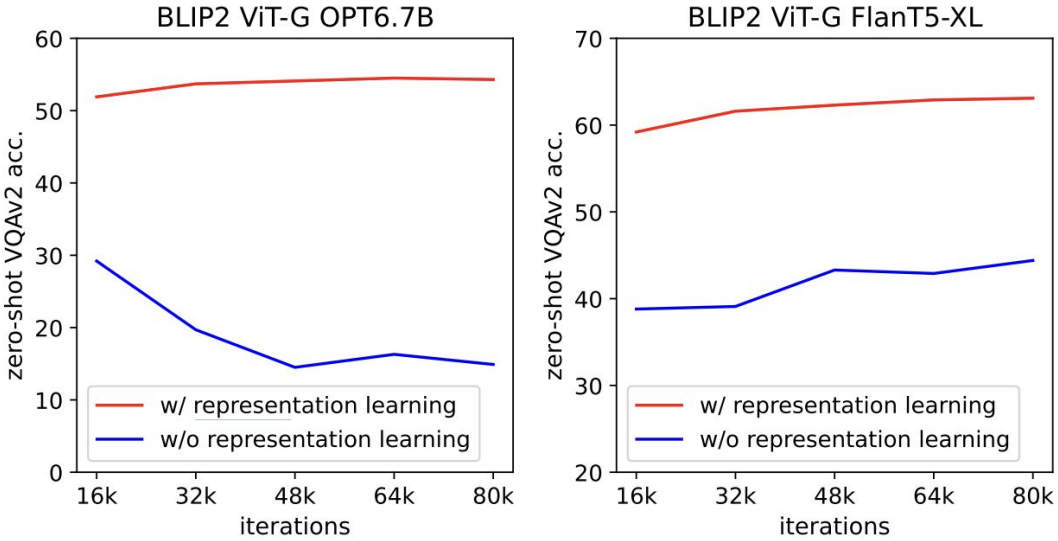


Figure 5. Effect of vision-language representation learning on vision-to-language generative learning. Without representation learning, the Q-Former fails the bridge the modality gap, leading to significantly lower performance on zero-shot VQA.

Outline

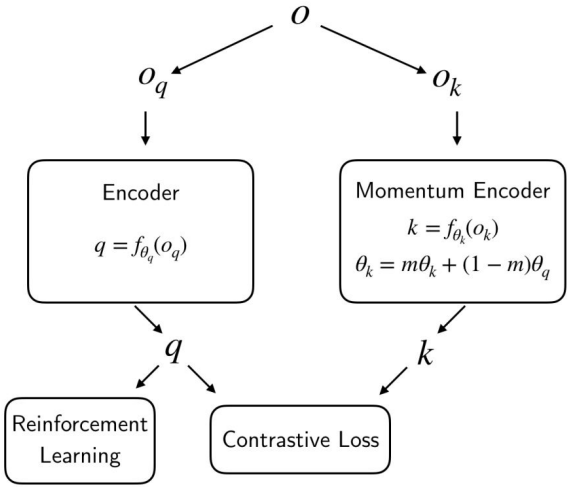
- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

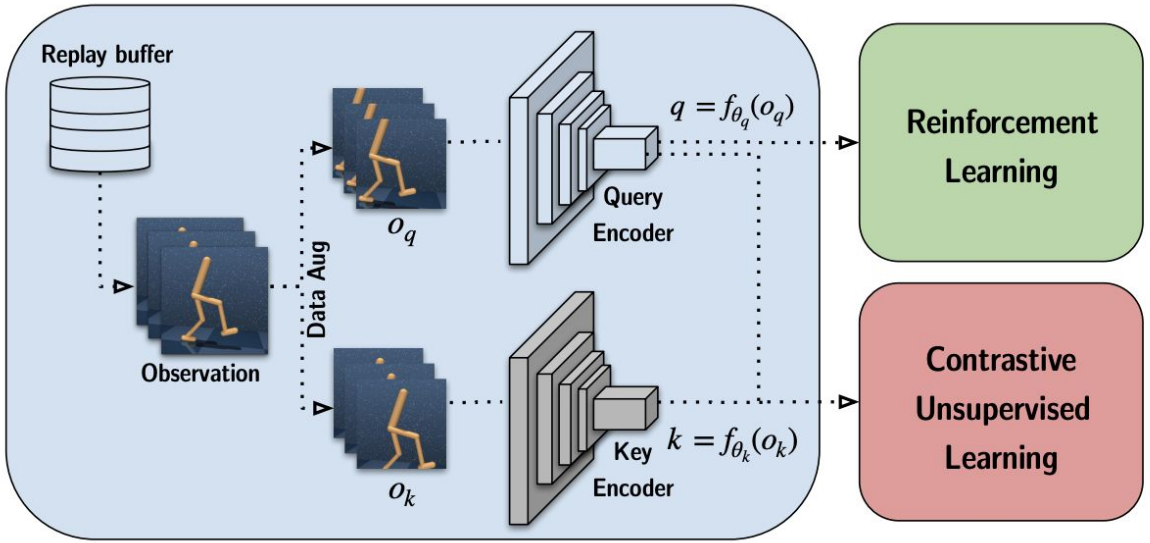
CURL

CURL: Contrastive Unsupervised Representations for Reinforcement Learning

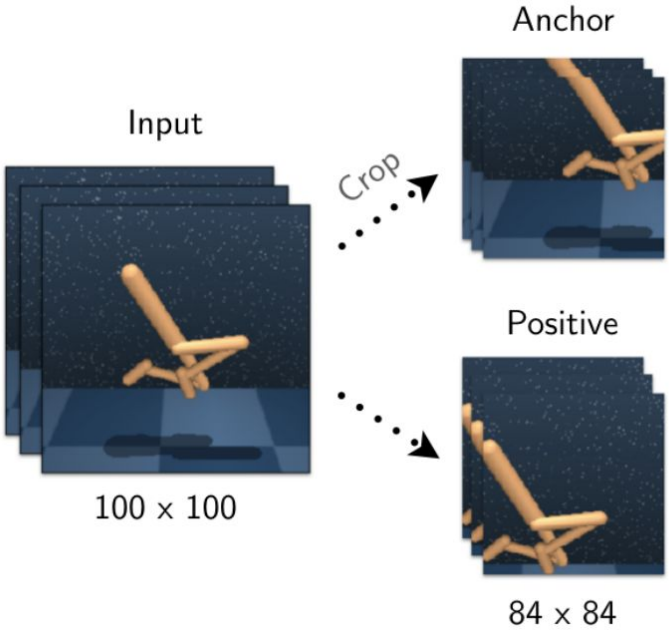
Aravind Srinivas* [1] **Michael Laskin*** [1] **Pieter Abbeel** [1]

CURL

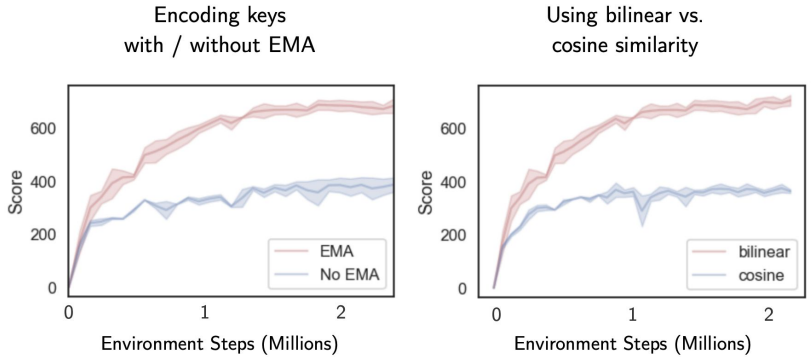

o
o_q
o_k
Encoder
$q = f_{\theta_q}(o_q)$
Momentum Encoder
$k = f_{\theta_k}(o_k)$
$\theta_k = m\theta_k + (1-m)\theta_q$
q
k
Reinforcement Learning
Contrastive Loss


Replay buffer
Observation
Data Aug
o_q
o_k
Query Encoder
Key Encoder
$q = f_{\theta_q}(o_q)$
$k = f_{\theta_k}(o_k)$
Reinforcement Learning
Contrastive Unsupervised Learning

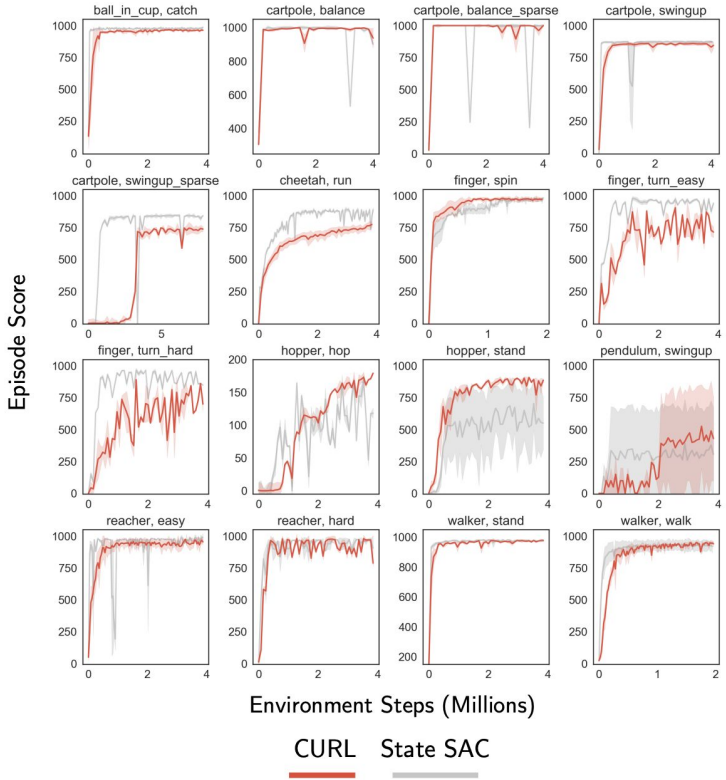
CURL

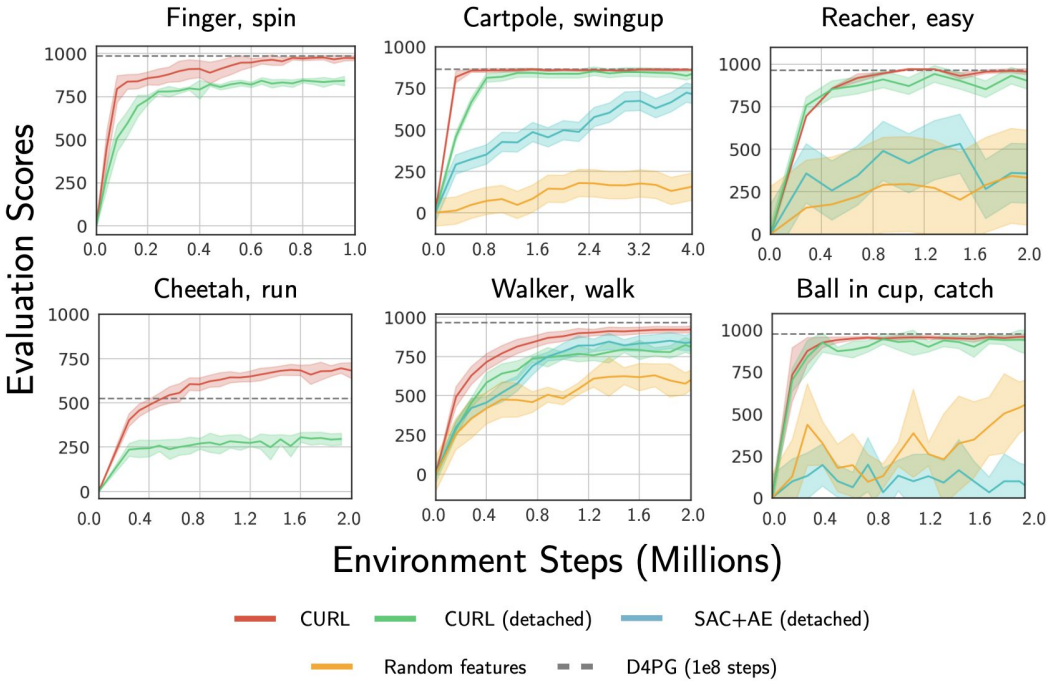


$$\mathcal{L}_q = \log \frac{\exp(q^T W k_+)}{\exp(q^T W k_+) + \sum_{i=0}^{K-1} \exp(q^T W k_i)}$$



CURL


ball_in_cup, catch
cartpole, balance
cartpole, balance_sparse
cartpole, swingup
cartpole, swingup_sparse
cheetah, run
finger, spin
finger, turn_easy
finger, turn_hard
hopper, hop
hopper, stand
pendulum, swingup
reacher, easy
reacher, hard
walker, stand
walker, walk
Episode Score
Environment Steps (Millions)
CURL
State SAC


Finger, spin
Cartpole, swingup
Reacher, easy
Cheetah, run
Walker, walk
Ball in cup, catch
Evaluation Scores
Environment Steps (Millions)
CURL
CURL (detached)
SAC+AE (detached)
Random features
D4PG (1e8 steps)

R3M

R3M: A Universal Visual Representation for Robot Manipulation

Suraj Nair[1,*], **Aravind Rajeswaran**[2], **Vikash Kumar**[2], **Chelsea Finn**[1], **Abhinav Gupta**[2]

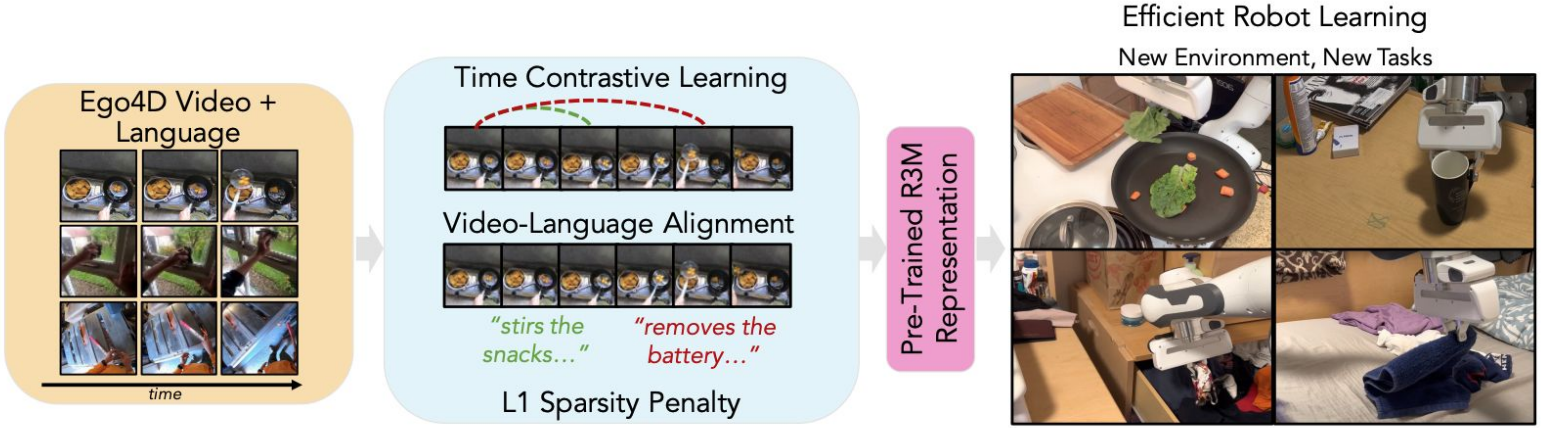
[1]Stanford University, [2]Meta AI

R3M

Motivation: Plug-and-play, general Visual Representations for Robotics must contain Three main ingredients…

1. Temporal dynamics of the scene i.e. how states might transition to other states
2. A prior over semantic relevance: should focus on task relevant features like objects and their relationships
3. Be compact, excluding features irrelevant to the previous two criteria such as backgrounds

R3M

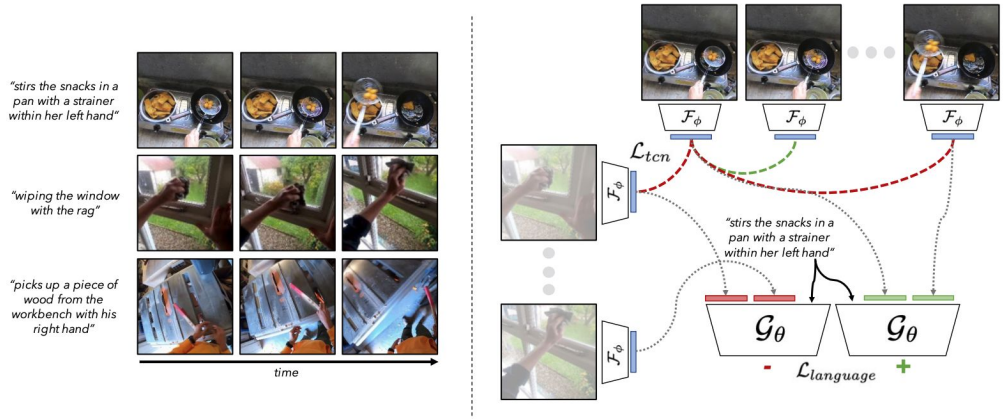


Ego4d is diverse, in-the-wild, and language annotated

Contains 3,500 hours of data from 70 locations across the globe

R3M

Time Contrastive Learning encodes temporal dynamics into the representation



$$\mathcal{L}_{tcn} = -\sum_{b \in B} \log \frac{e^{\mathcal{S}(z_i^b, z_j^b)}}{e^{\mathcal{S}(z_i^b, z_j^b)} + e^{\mathcal{S}(z_i^b, z_k^b)} + e^{\mathcal{S}(z_i^b, z_i^{\neq b})}}$$

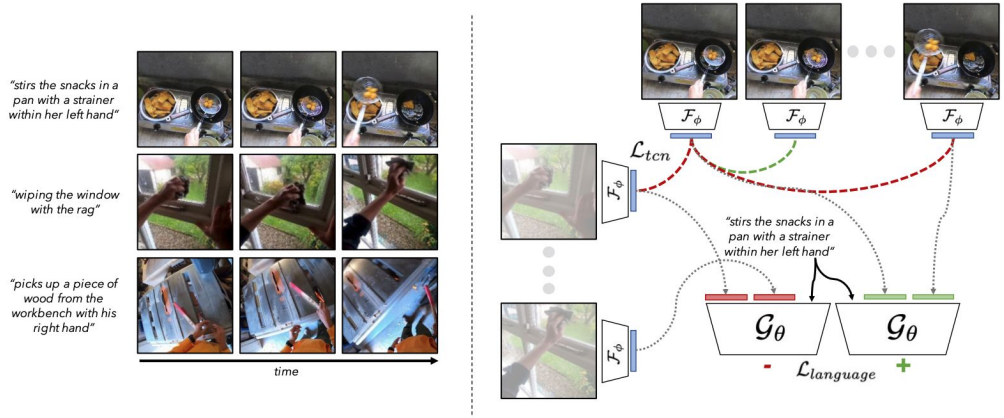
1. Frames closer in time are more perceptually similar than frames farther apart in time
2. Frames from the same videos are more perceptually similar than those from other videos

$$\mathcal{S}(x_1, x_2) = -||x_1 - x_2||_2^2$$

$$[I_i, I_{j>i}, I_{k>j}]^{1:B}$$

R3M

Video-Language alignment encourages F to capture semantically relevant features



1. Video language alignment should increase over the course of the video
2. Frames from captioned video should be more aligned language than frames from another video

$$\mathcal{L}_{language} = -\sum_{b \in B} \log \frac{e^{\mathcal{G}_\theta(z_0^b, z_{j>i}^b, l^b)}}{e^{\mathcal{G}_\theta(z_0^b, z_{j>i}^b, l^b)} + e^{\mathcal{G}_\theta(z_0^b, z_i^b, l^b)} + e^{\mathcal{G}_\theta(z_0^{\neq b}, z_{j>i}^{\neq b}, l^b)}}$$

$$\mathcal{G}_\theta(\mathcal{F}_\phi(I_0), \mathcal{F}_\phi(I_i), l)$$

$$[I_i, I_{j>i}, l]^{1:B}$$

R3M

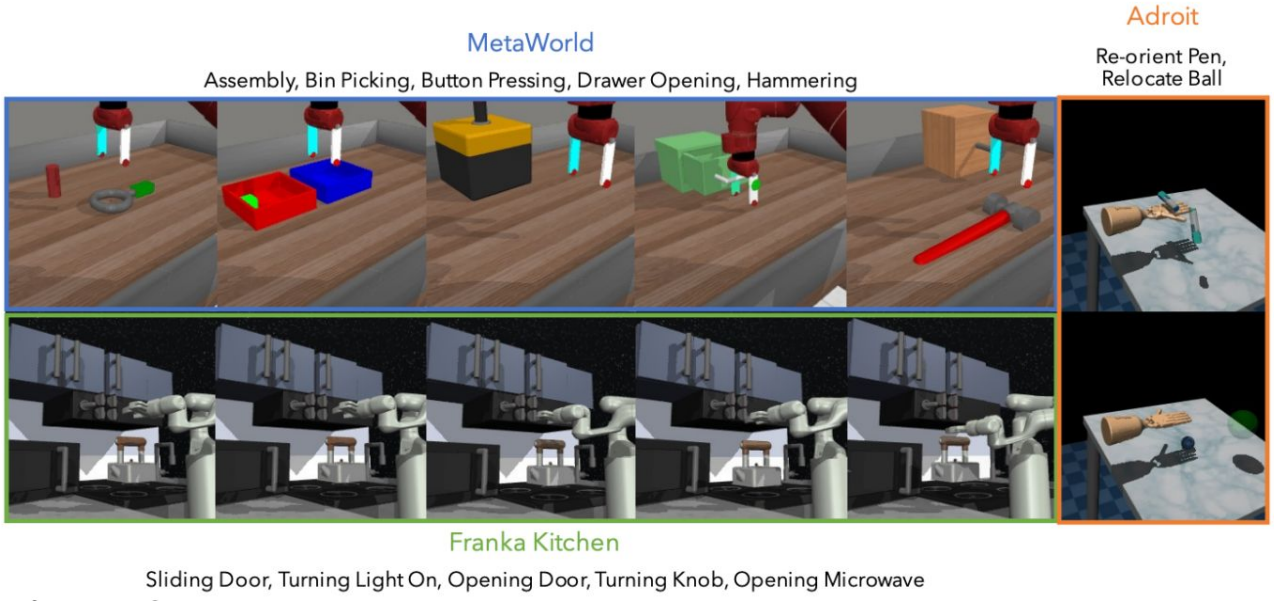
Joint Optimization - Simple regularizations encourage sparsity of representations

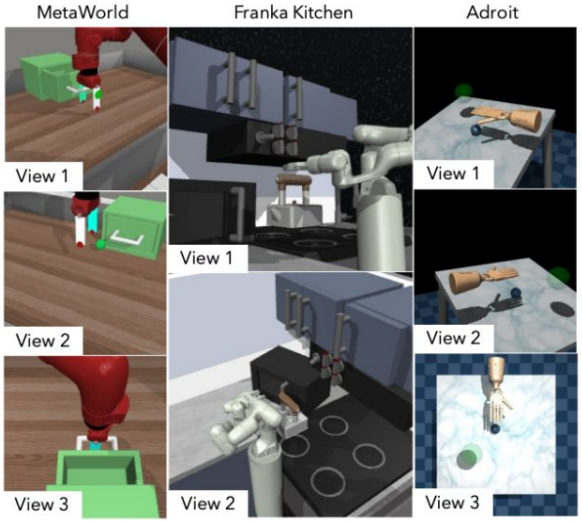
- ResNet18, ResNet34, and ResNet50 architectures optimized with Adam are all released
- Random cropping at the video level

L1 reduces representations to only **critical features**
L2 probably has more of a **regularizing effect** than anything

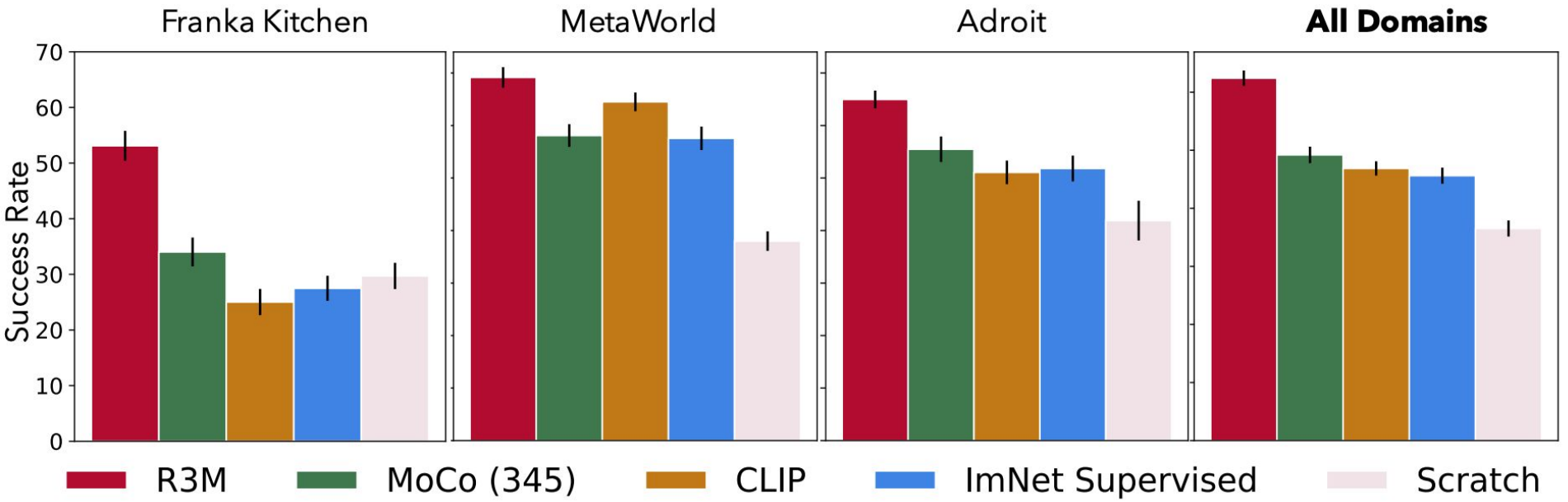
$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{I^{1:B}_{0,i,j,k} \sim \mathcal{D}}[\lambda_1 \mathcal{L}_{tcn} + \lambda_2 \mathcal{L}_{language} + \lambda_3 ||\mathcal{F}_\phi(I_i)||_1 + \lambda_4 ||\mathcal{F}_\phi(I_i)||_2]$$

R3M


MetaWorld
Assembly, Bin Picking, Button Pressing, Drawer Opening, Hammering
Adroit
Re-orient Pen, Relocate Ball
Franka Kitchen
Sliding Door, Turning Light On, Opening Door, Turning Knob, Opening Microwave


MetaWorld
Franka Kitchen
Adroit
View 1
View 2
View 3
View 1
View 2
View 1
View 2
View 3

R3M


Franka Kitchen
MetaWorld
Adroit
All Domains
Success Rate
70
60
50
40
30
20
10
0
R3M
MoCo (345)
CLIP
ImNet Supervised
Scratch

R3M

Environment	*Supervised* R3M	R3M(-Aug)	R3M(-L1)	*Self-Supervised* R3M(-Lang)
Franka Kitchen	**53.1** ±2.7%	**51.1** ±2.7%	46.7 ±2.7%	47.2±2.9%
MetaWorld	**69.2** ±2.0%	**68.9** ±2.1%	65.0 ±2.4%	67.0±2.0%
Adroit	65.0 ±1.7%	61.3 ±2.1%	**66.5** ±1.6%	45.6 ±3.3%
All Domains	**62.4** ±1.3%	60.4 ±1.4%	59.4 ±1.5%	53.2 ±1.5%

Table 1: **Ablating Components of R3M**. We see report success rate of downstream imitation learning on variants of R3M. We observe that on average, removing the L1 penalty have a negative impact, particularly on the Franka Kitchen and MetaWorld environments. Lastly, removing language grounding has the most significant drop in performance, particularly on the Adroit tasks.

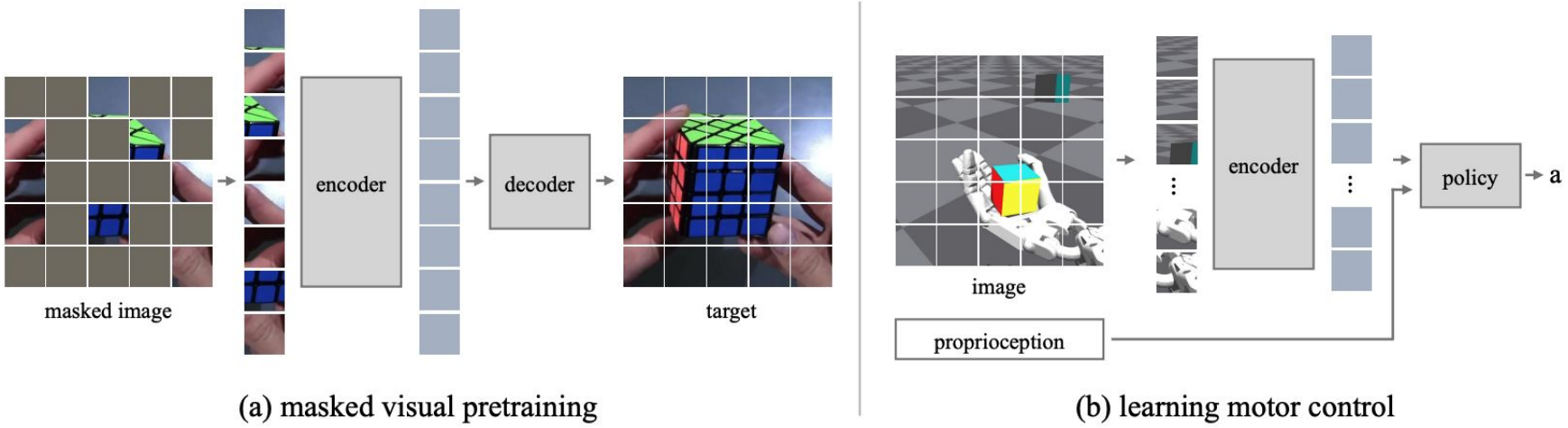
MVP

Masked Visual Pre-training for Motor Control

Tete Xiao [*1] Ilija Radosavovic [*1] Trevor Darrell [†1] Jitendra Malik [†1]

MVP

Use MAE learned features for robotic control



MVP

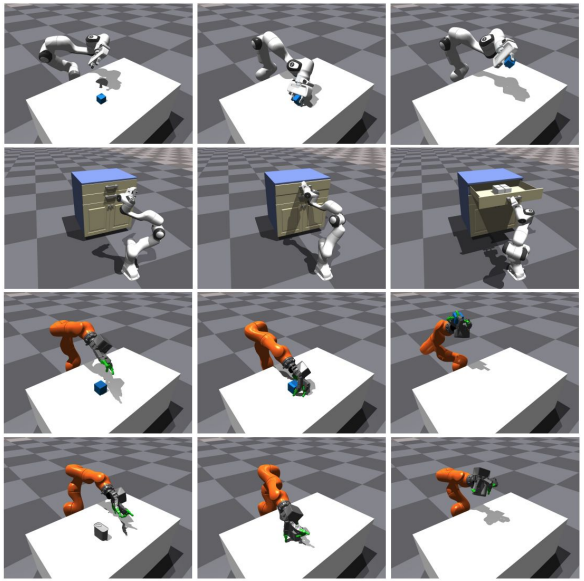
Establishes a benchmark dataset and evaluation suite

Train on Human-Object-Interaction dataset of 700k images:

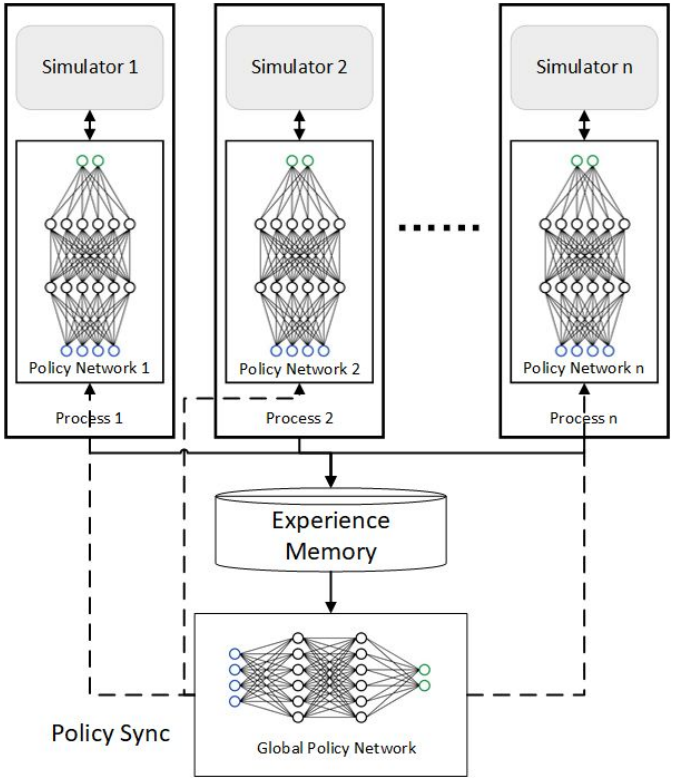
1. Epic Kitchens
2. Youtube 10 Days of Hands
3. Something 2 Something

Evaluate on new PixMC Benchmark - Train with PPO

	RLBench	RoboSuite	MetaWorld	Ours
Simulator	CoppeliaSim	MuJoCo	MuJoCo	IsaacGym
Fast				✓
#Arms	1	8	1	2
#Hands				✓
#Tasks	100	9	50	8
Rewards		✓	✓	✓



MVP

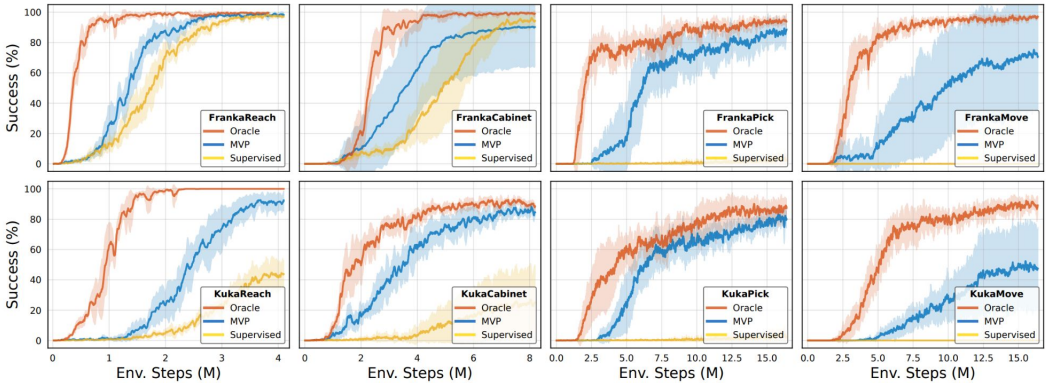


Evaluate with highly parallelized PPO

- MLP policy
- Critic has same architecture and learns from same representations
- State is MAE representation + proprioception
- Action space is position control in joint angle space
- Learn to handcrafted dense rewards

MVP

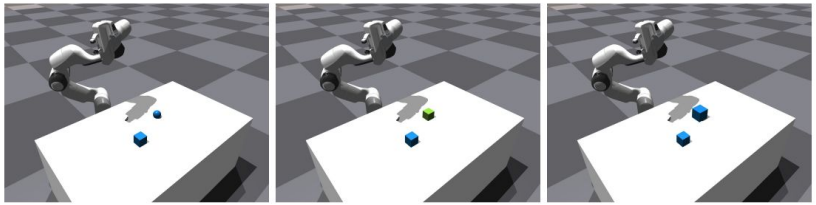
Self-supervised on large data is better than supervised on smaller data (ImageNet)

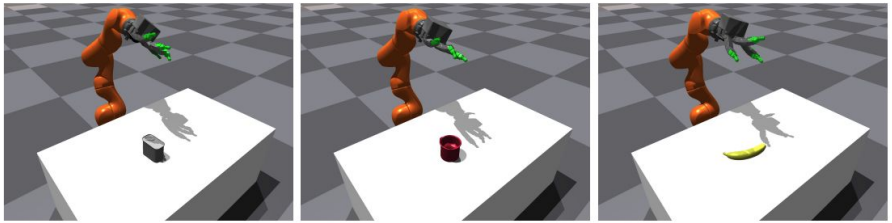


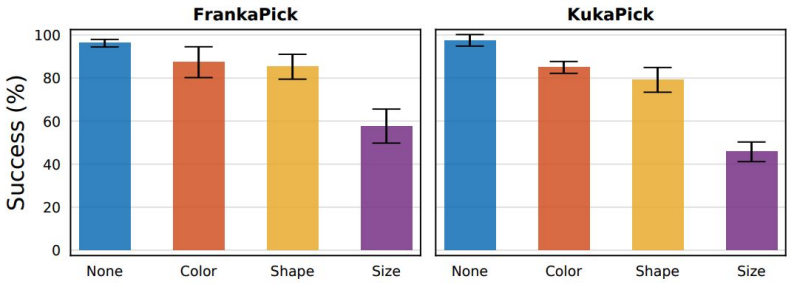
Oracle has access to hand-engineered state - location of objects, 3d poses, direction to goal vectors

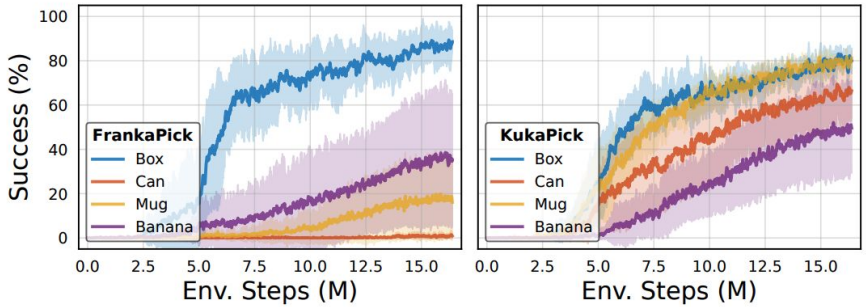
MVP

Representations robust to distractors and generalize different object types











Egocentric human videos
(e.g. Ego4D, Epic-Kitchens etc)

+ Large volume and diversity

- Lacks modalities important for EAI
  (e.g. proprioception, actions etc)



Robot execution trajectories
(e.g. BAIR Robot Dataset, BC-Z etc)

+ Matching modality and embodiments

- Lacks volume and diversity
  (physical system, lab setup etc.)



Simulators
(e.g. Habitat, MuJoCo etc)

+ Side information available
  (e.g. joint sensors, object poses)

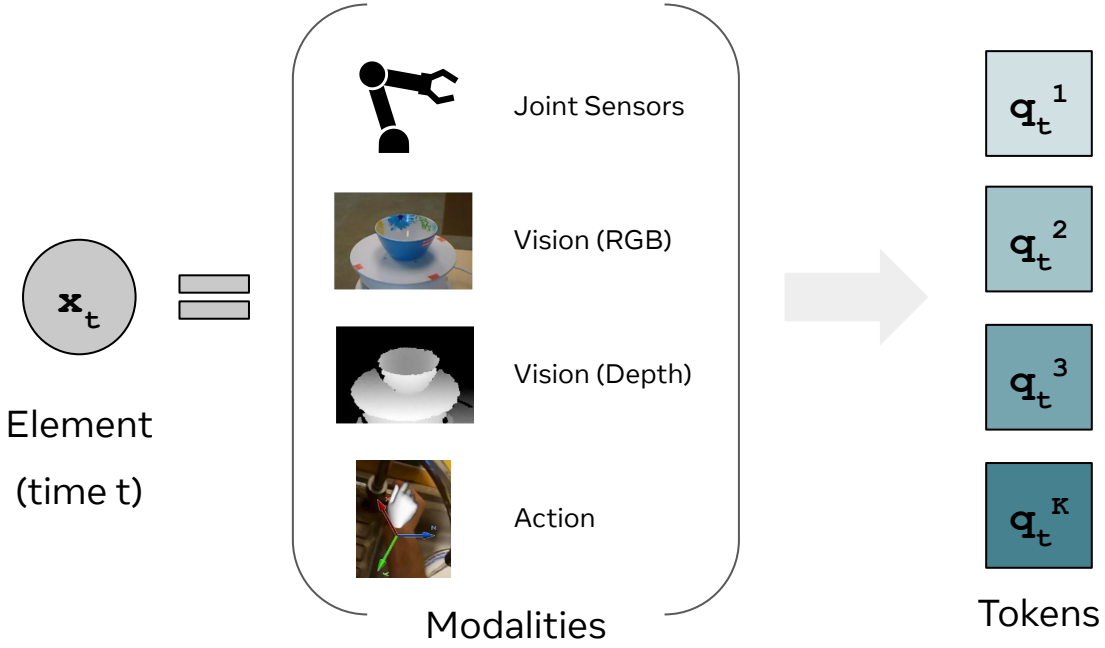
- Inaccurate physics for transfer

Goal: Develop a unified learning paradigm for trajectories that are multi-modal and heterogeneous

MTM

$$\tau = (\mathbf{x}_0, \mathbf{x}_1, \ldots \mathbf{x}_t, \ldots \mathbf{x}_H)$$

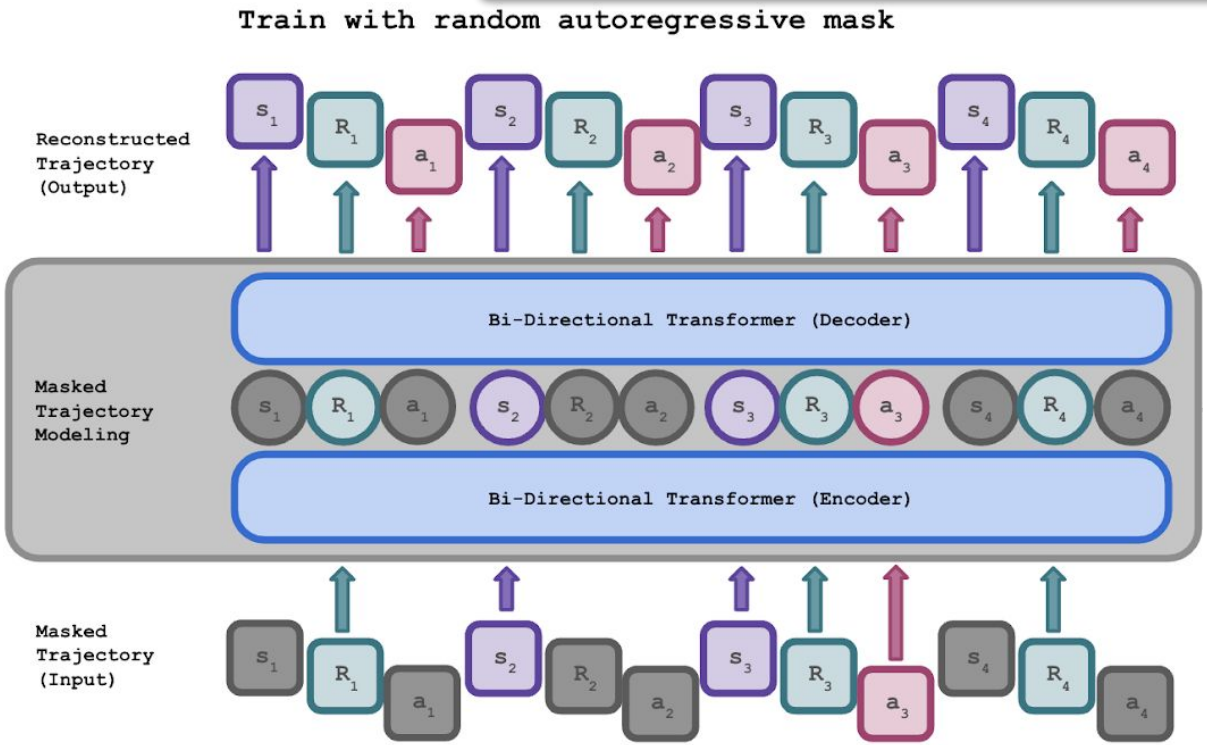
Trajectory is a generic sequence of *elements*.



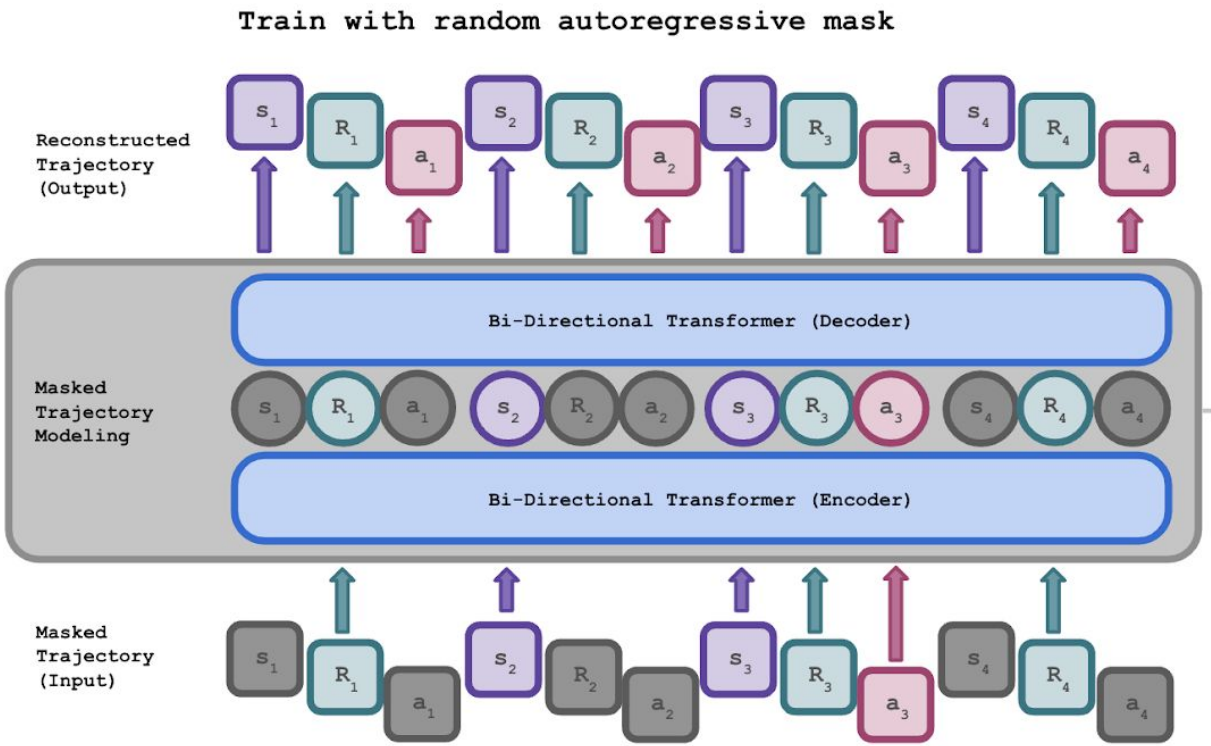
Lift to common embedding space with **modality-specific encoders.**

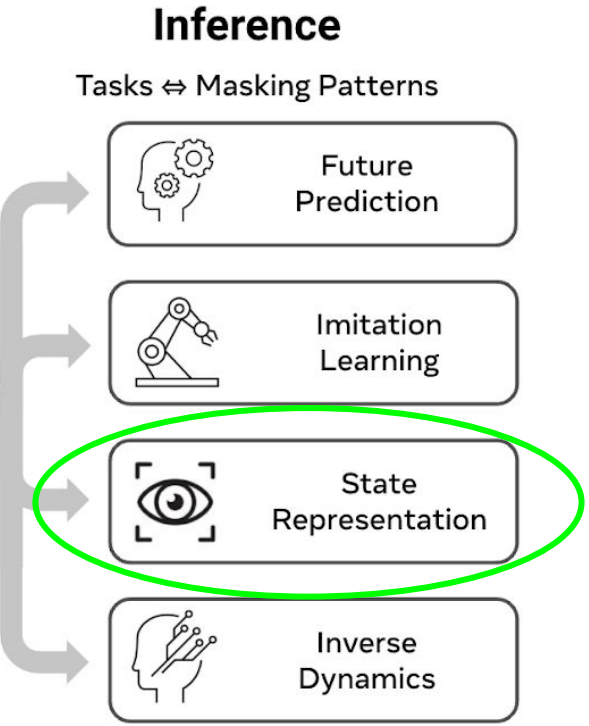
MTM

Missing modalities ⇔ Masked as a constraint

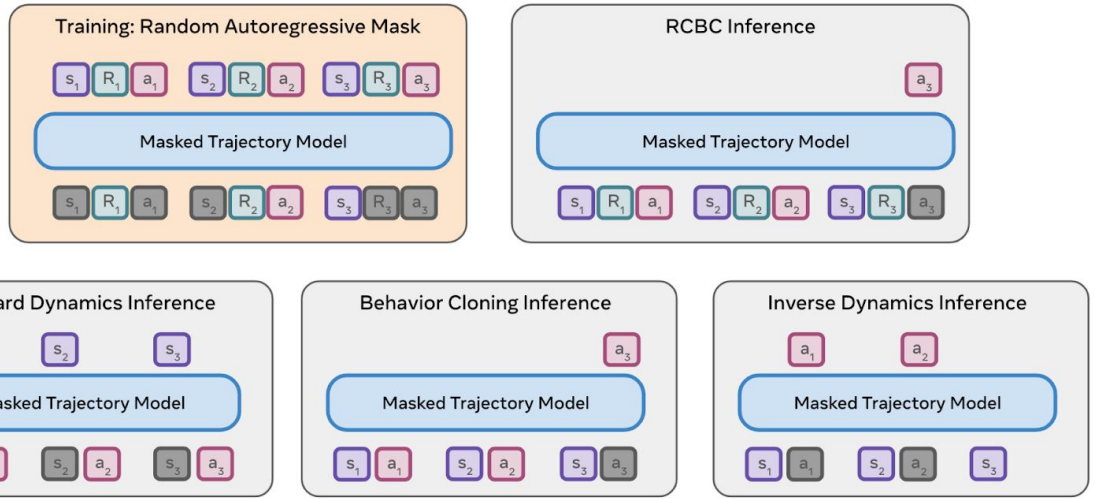


MTM


Train with random autoregressive mask
Reconstructed Trajectory (Output)
Masked Trajectory Modeling
Bi-Directional Transformer (Decoder)
Bi-Directional Transformer (Encoder)
Masked Trajectory (Input)


Inference
Tasks ⇔ Masking Patterns
Future Prediction
Imitation Learning
State Representation
Inverse Dynamics

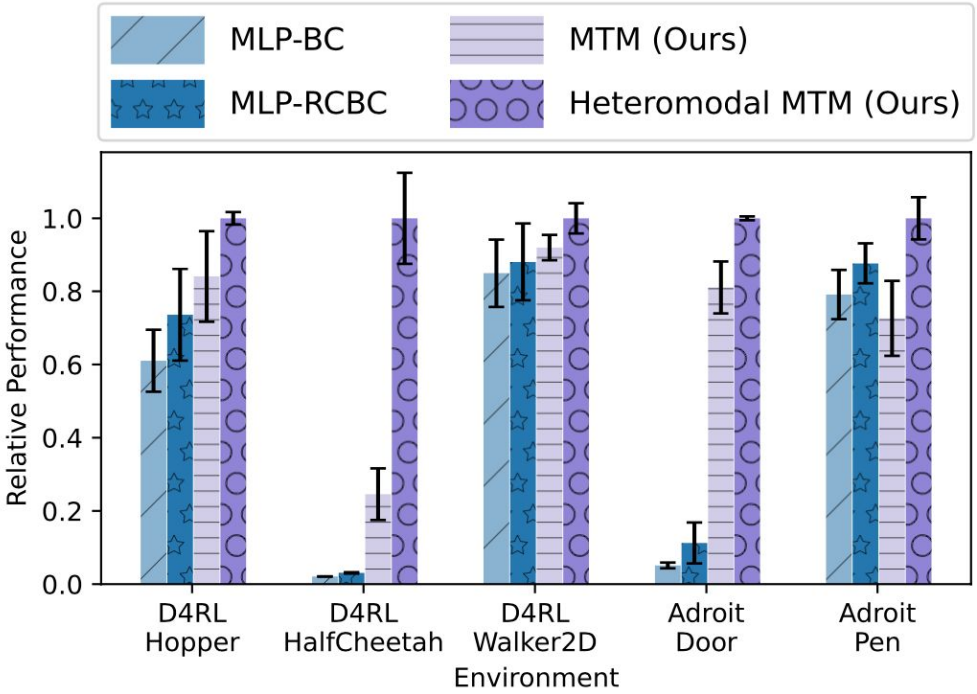
MTM



Summary

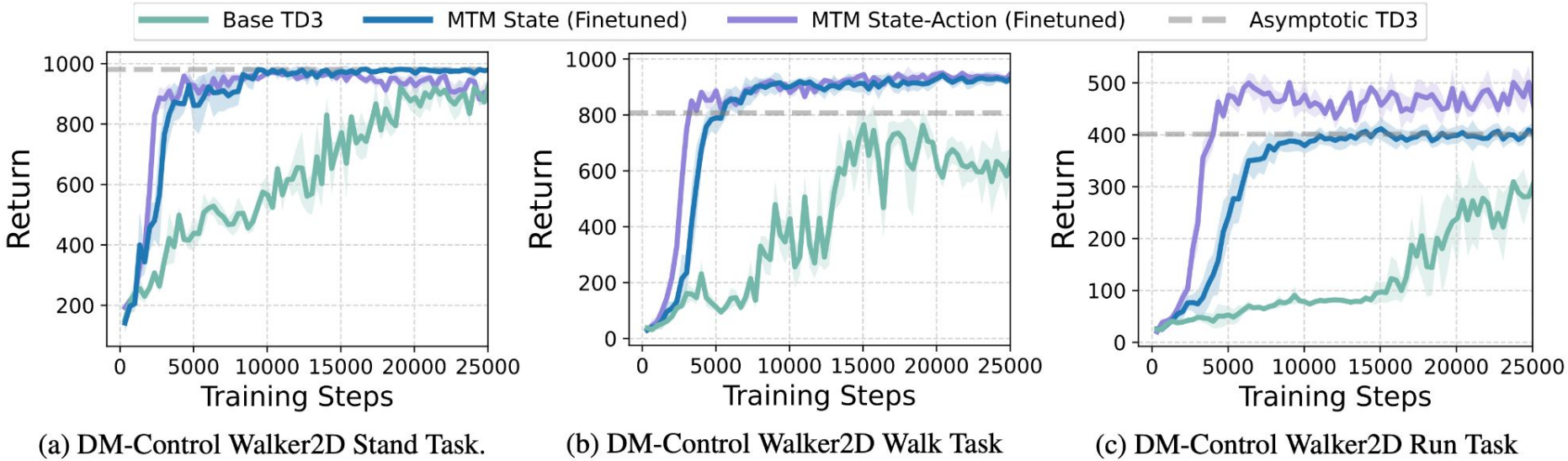
- **Random autoregressive masking helps downstream prediction tasks**
- Competitive RCBC on continuous control tasks
- **Heteromodal dataset training capabilities**
- **Representation learning**

MTM



- **Setup:** Only a small fraction (~1%) of the dataset has full (s, R, a) trajectories. Remainder of dataset is missing actions.
- **Baselines:** Can train only on the labelled subset of data.
- **Heteromodal MTM:** Train on mixture dataset, with missing modalities treated as if they were masked out.

MTM



(a) DM-Control Walker2D Stand Task.

(b) DM-Control Walker2D Walk Task

(c) DM-Control Walker2D Run Task

Setup: (1) Pretrain MTM model on offline dataset.

(2) Use state encoder of MTM and feed it to a standard RL algorithms (TD3)

Masked World Models for Visual Control

Masked World Models for Visual Control

Younggyo Seo[1,2,*] **Danijar Hafner**[2,3,4] **Hao Liu**[2] **Fangchen Liu**[2]

Stephen James[2,†] **Kimin Lee**[3] **Pieter Abbeel**[2]

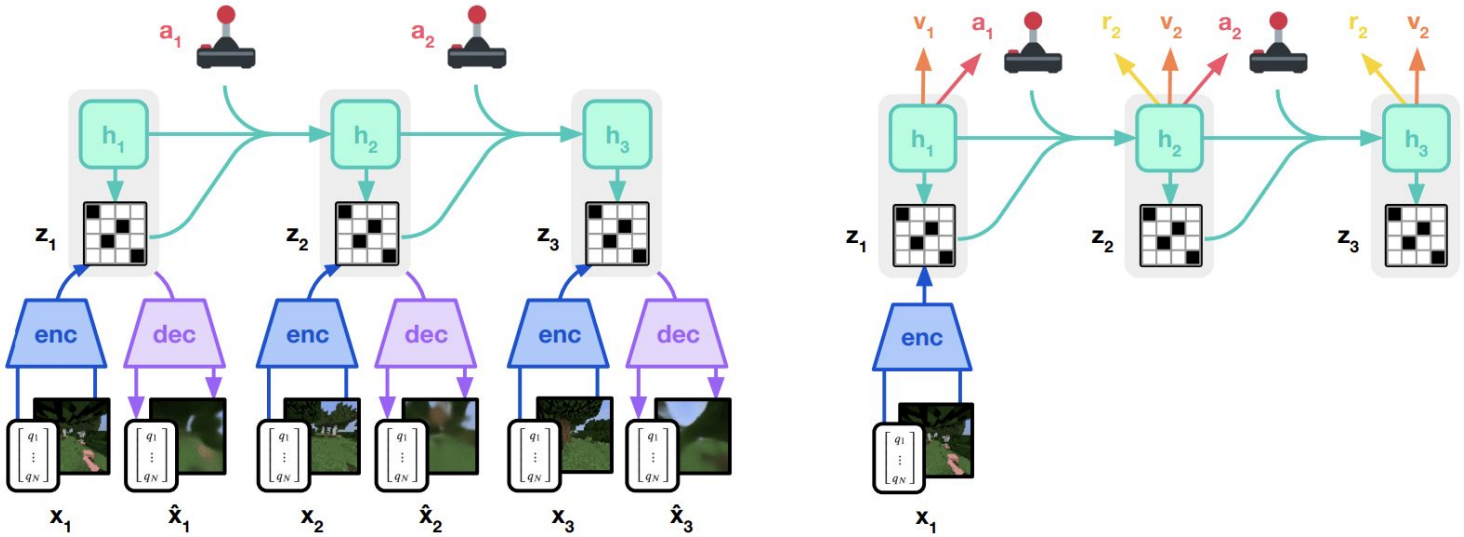
[1] KAIST [2] UC Berkeley [3] Google Research [4] University of Toronto

Masked World Models for Visual Control

Mastering Diverse Domains through World Models

Danijar Hafner;[1,2] Jurgis Pasukonis;[1] Jimmy Ba;[2] Timothy Lillicrap[1]

[1]DeepMind [2]University of Toronto



(a) World Model Learning

(b) Actor Critic Learning

Masked World Models for Visual Control

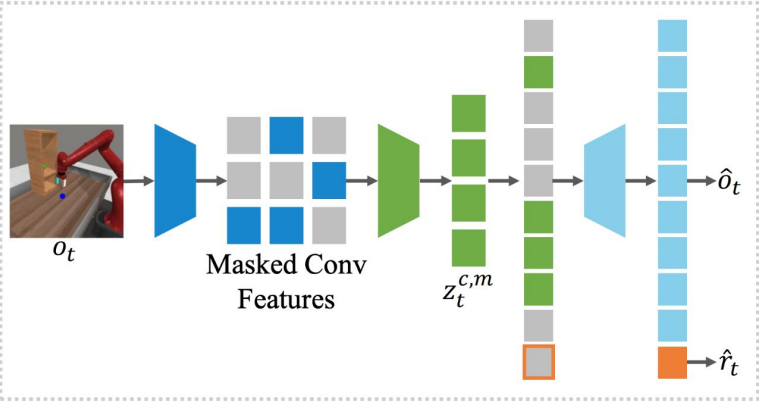
Main idea: Decouple visual representation learning and **dynamics learning**

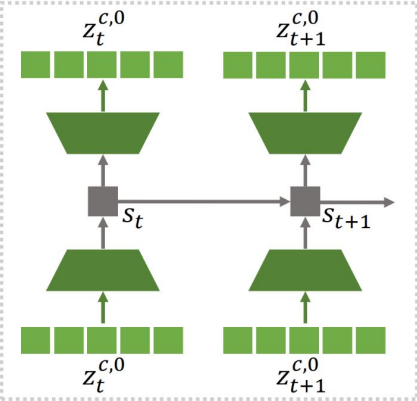
Visual Representation Learning

- Training an autoencoder with convolutional feature masking
- Reward prediction to encode task-relevant information

Dynamics learning

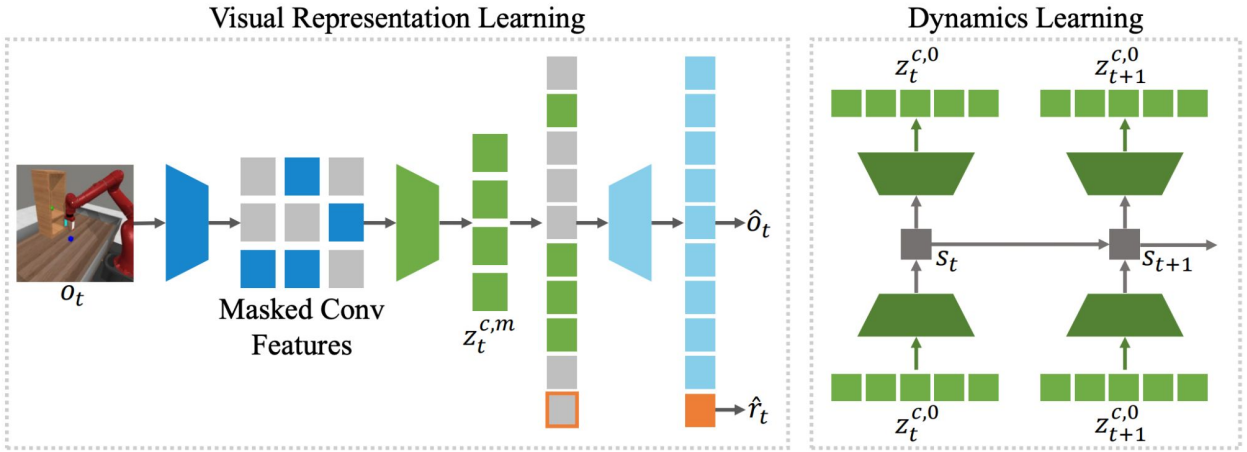
- Training a recurrent state-space model (RSSM) that reconstructs frozen autoencoder representations





(From Younggyo)

Masked World Models for Visual Control



$$\mathcal{L}^{\text{mwm}}(\phi,\theta) = \frac{1}{B}\sum_{j=1}^{B}\Big(\underbrace{-\ln p_\phi(o_j|z_j^{c,m}) - \ln p_\phi(r_j|z_j^{c,m})}_{\text{visual representation learning}}$$

$$\underbrace{-\ln p_\theta(z_j^{c,0}|s_j) - \ln p_\theta(r_j|s_j) + \beta\,\mathrm{KL}\Big[q_\theta(s_j|s_{j-1},a_{j-1},z_j^{c,0}) \,\|\, p_\theta(\hat{s}_j|s_{j-1},a_{j-1})\Big]}_{\text{dynamics learning}}\Big)$$

Masked World Models for Visual Control

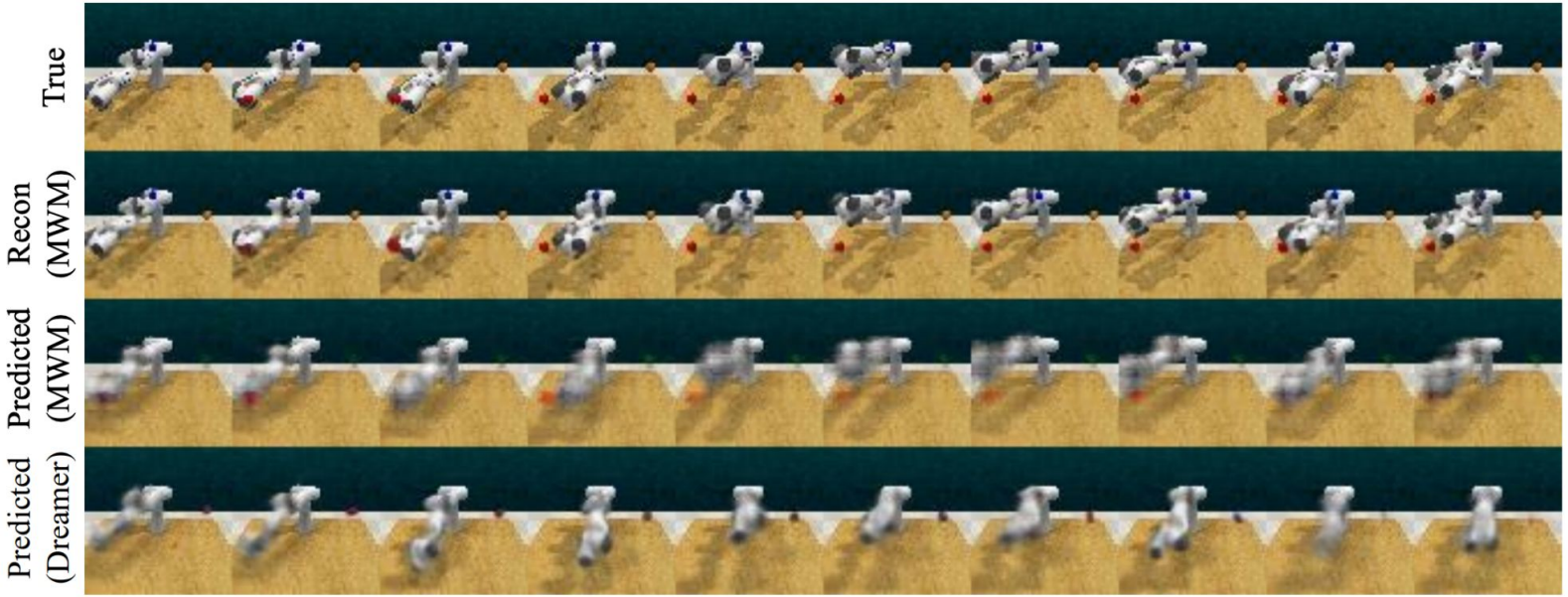
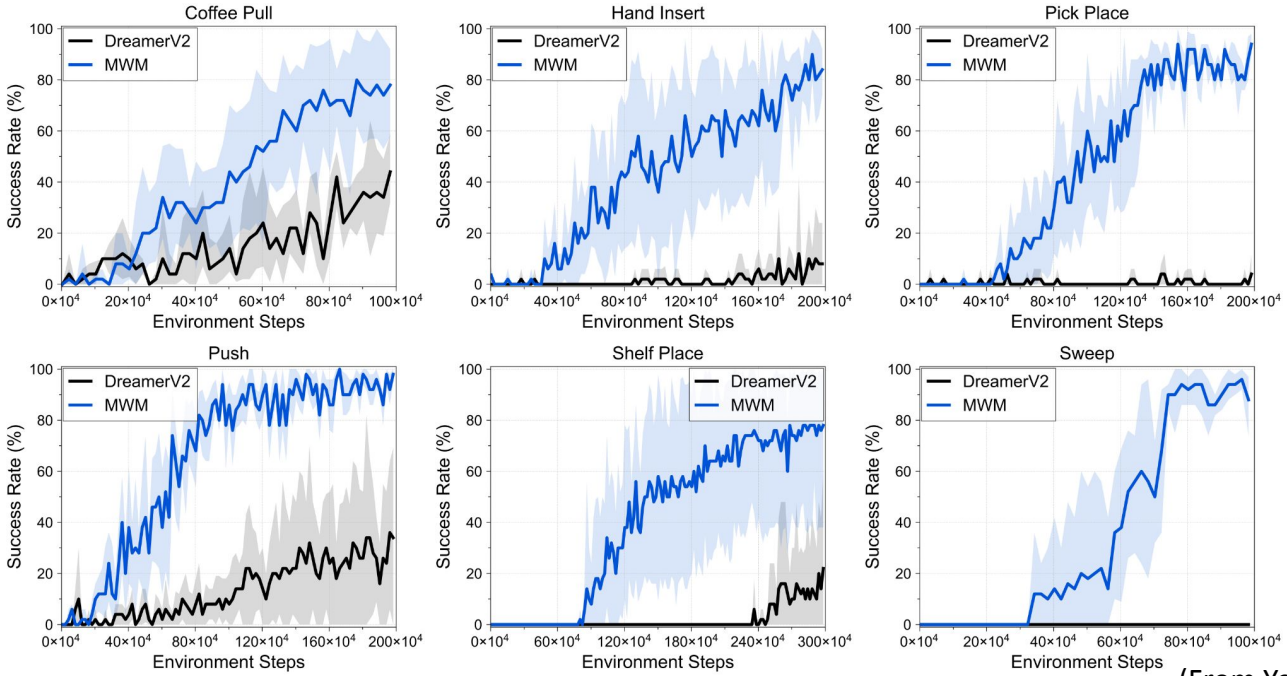
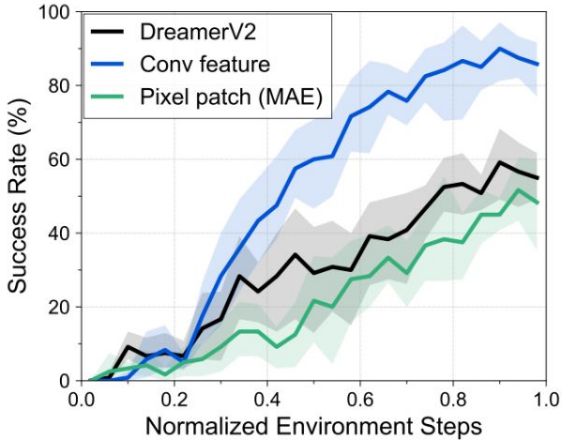


Figure 7: Future frames reconstructed with the autoencoder (*i.e.*, Recon) and predicted by latent dynamics models (*i.e.*, Predicted). Best viewed as video provided in Appendix B.

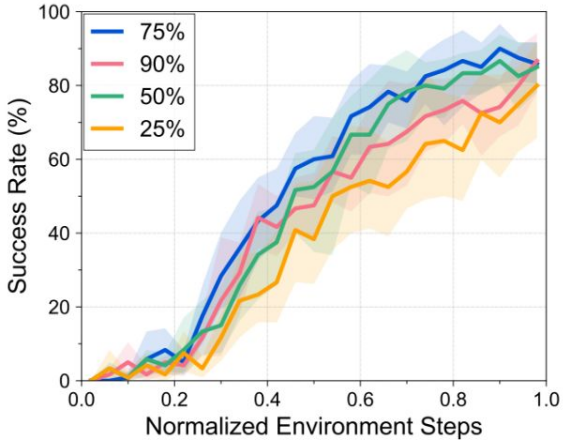
- **MWM** outperforms **DreamerV2** on challenging Meta-world tasks



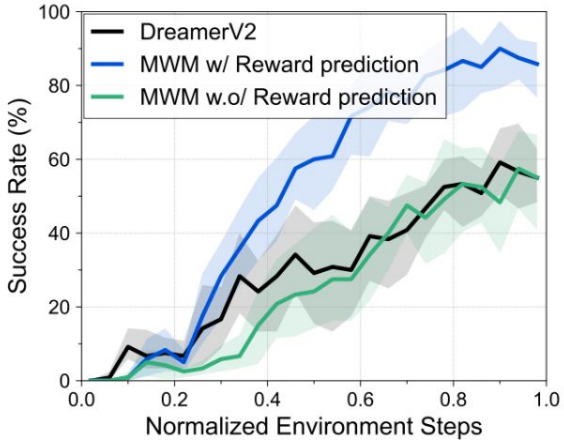
(From Younggyo)



(a) Feature masking



(b) Masking ratio



(c) Reward prediction

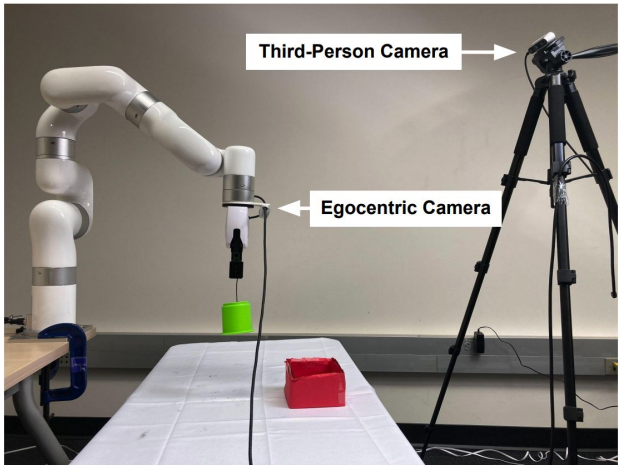
Figure 6: Learning curves on three manipulation tasks from Meta-world that investigate the effect of (a) convolutional feature masking, (b) masking ratio, and (c) reward prediction. The solid line and shaded regions represent the mean and stratified bootstrap confidence interval across 12 runs.

Multi-View MAE

- Multiple cameras have often been used for visual robotic manipulation



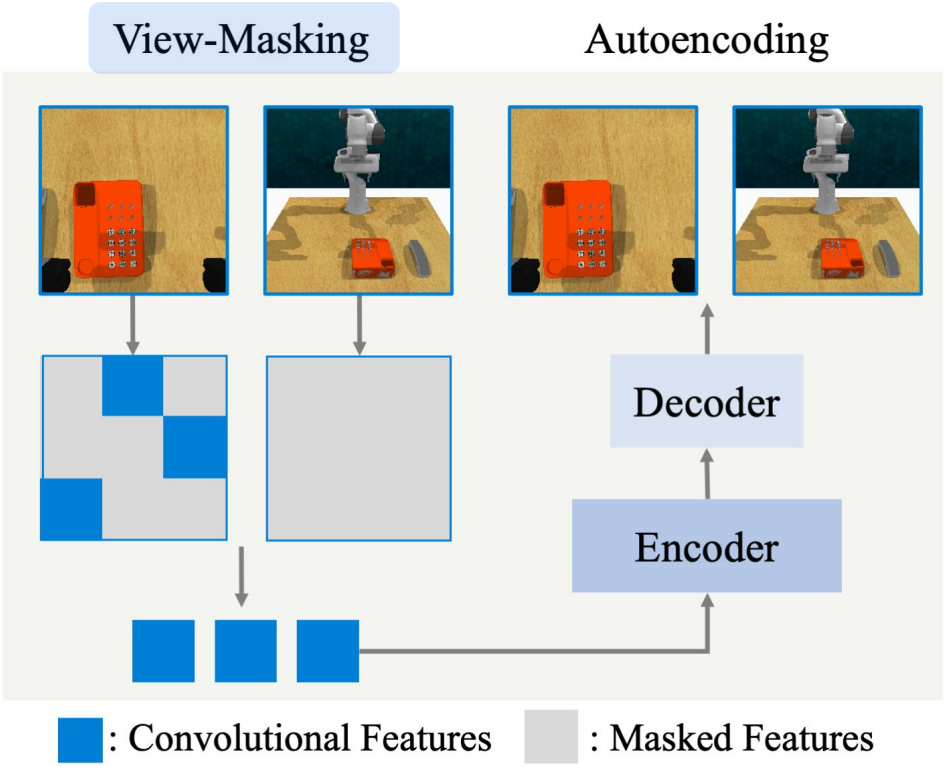
[Akkaya et al., 2019]



[Jangir et al., 2022]

(From Younggyo)

Multi-View MAE

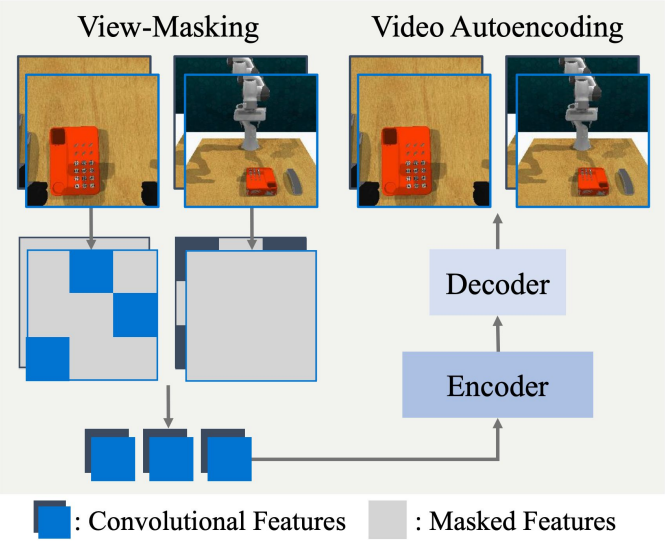


Main Idea:
Reconstruct masked viewpoints to learn cross-view information

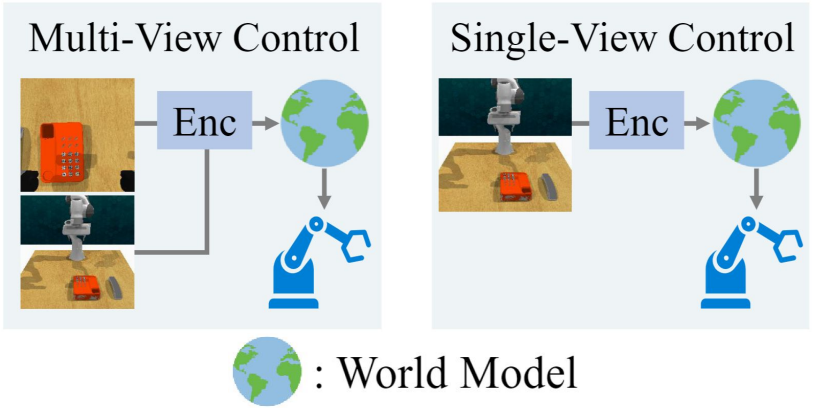
(From Younggyo)

Multi-View MAE

MV-MAE can extract both ***multi-view*** and ***single-view*** representations



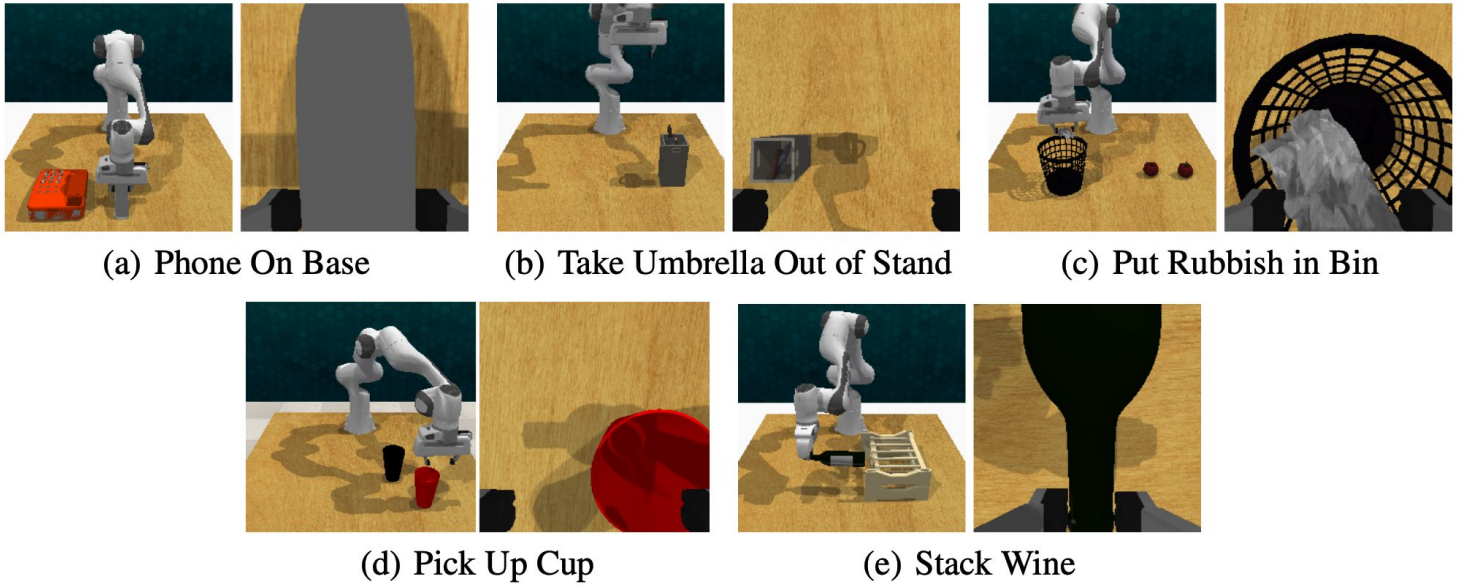
Visual robotic manipulation with **multi-view** or **single-view** data



(From Younggyo)

Multi-View MAE

- RLBench [James et al., 2020] with **front** and **wrist** cameras
  - Widely-used camera configuration



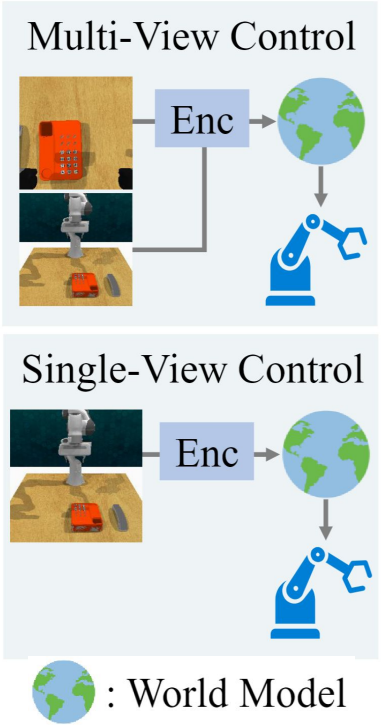
(a) Phone On Base (b) Take Umbrella Out of Stand (c) Put Rubbish in Bin

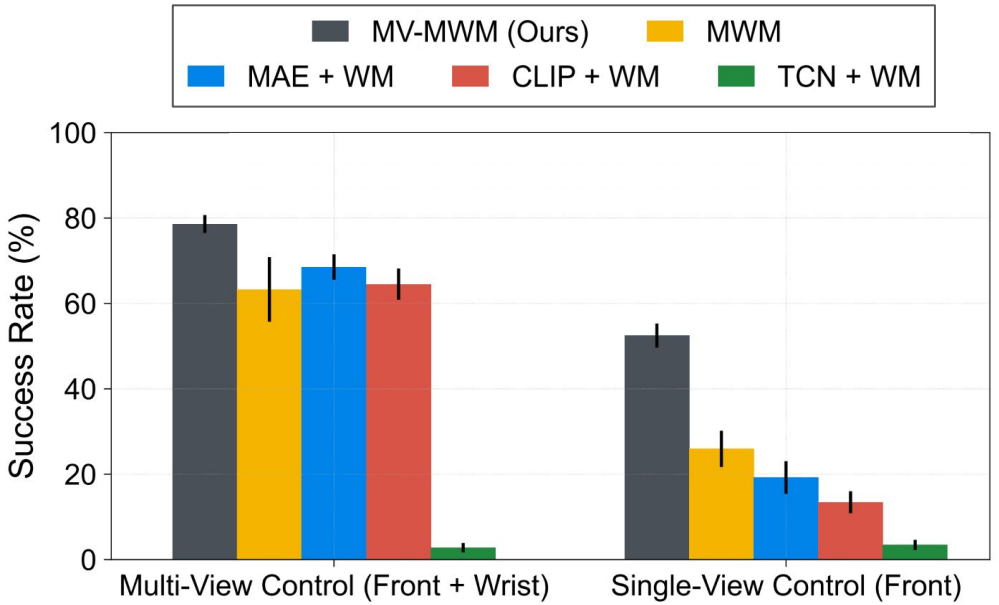
(d) Pick Up Cup (e) Stack Wine

(From Younggyo)

Multi-View MAE

- MV-MWM outperforms both single-view and multi-view baselines

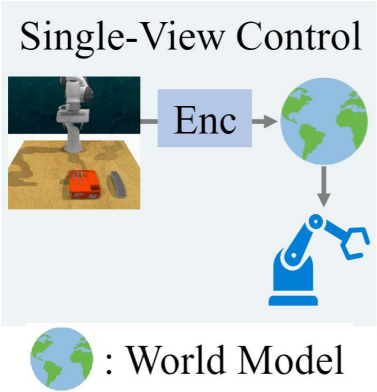




(From Younggyo)

Multi-View MAE

- MV-MWM is also outperforming baselines in imitation learning setup



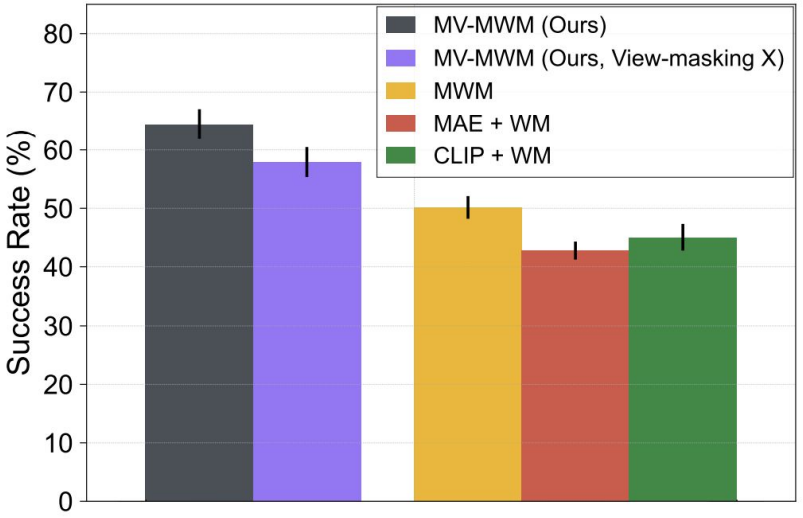
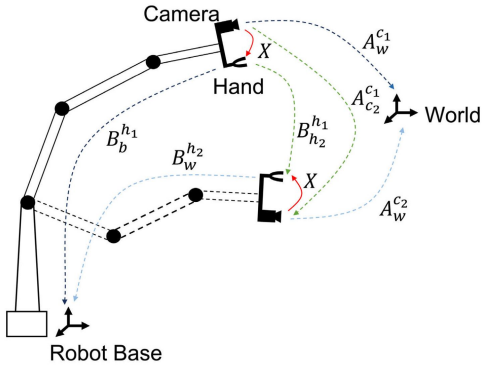


Figure 8. Aggregate success rate of imitation learning agents on five single-view control tasks. The result shows the mean and stratified bootstrap confidence interval across 20 runs.

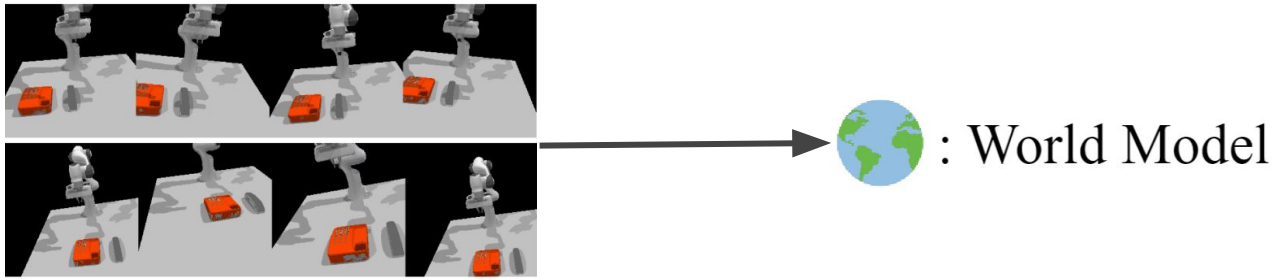
(From Younggyo)

Multi-View MAE



Motivation:
Camera calibration is a tedious procedure

- **Solution:** Training a **viewpoint-robust** policy with viewpoint randomization



Viewpoint randomization

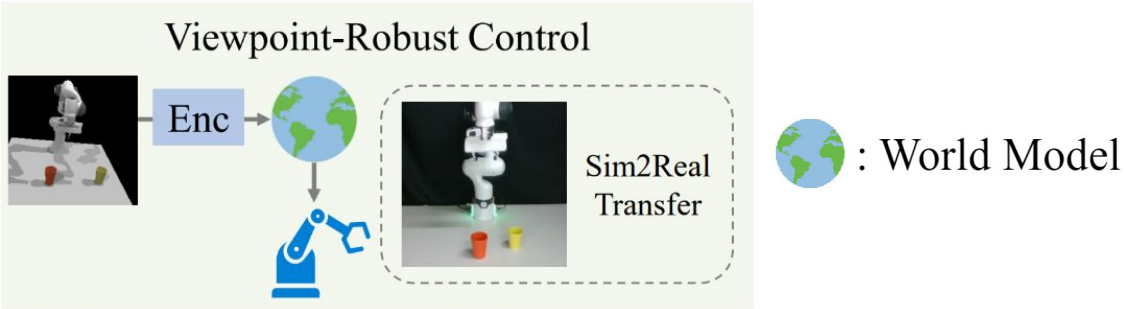
(From Younggyo)

Multi-View MAE

- **Step 1:** Multi-view representation learning with viewpoint randomization



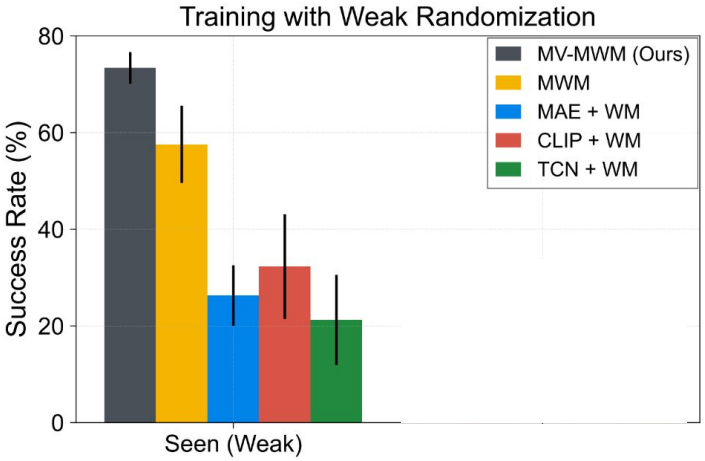
- **Step 2:** Learn a world model for viewpoint-robust control

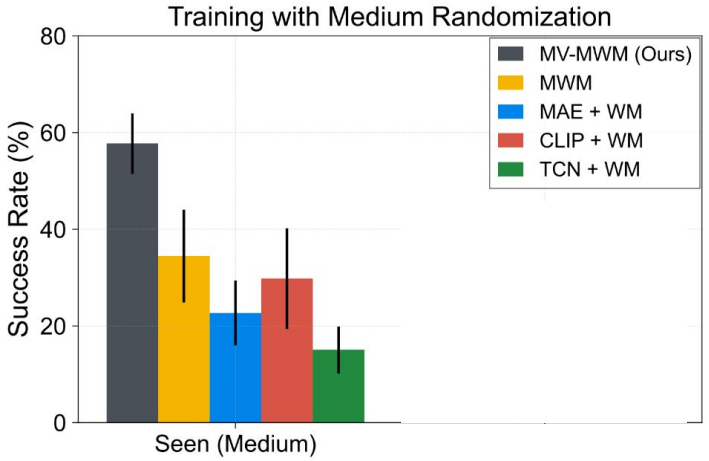


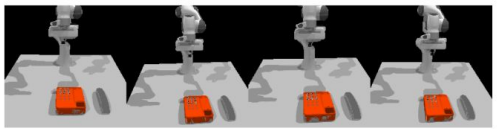
(From Younggyo)

Multi-View MAE

- **MV-MWM** learns a policy with aggressive viewpoint randomization







(a) Weak randomization

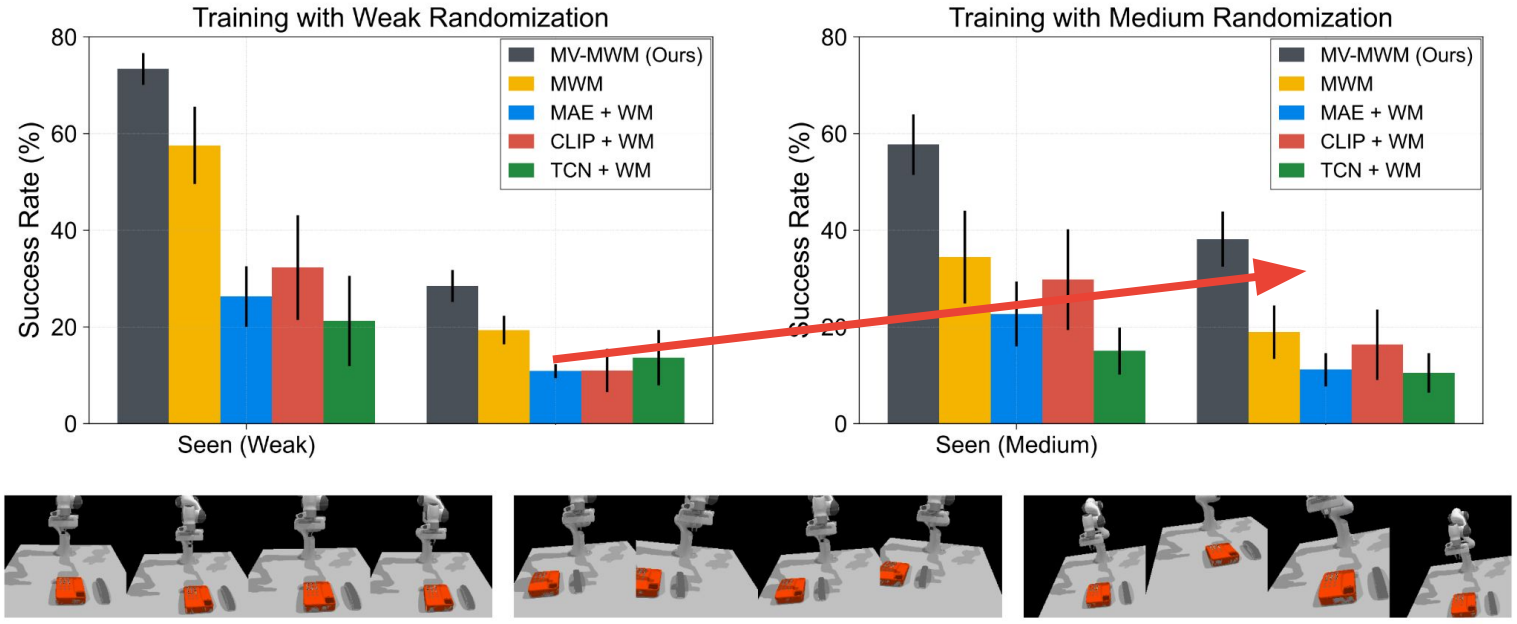


(b) Medium randomization

(From Younggyo)

Multi-View MAE

- **MV-MWM** learns a policy with aggressive viewpoint randomization

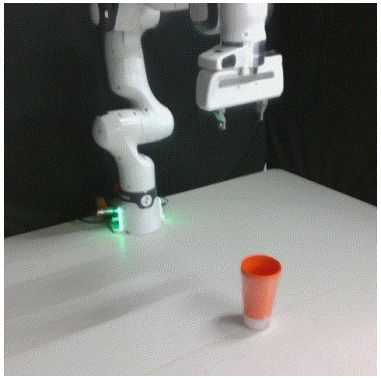


(a) Weak randomization (b) Medium randomization (c) Strong randomization

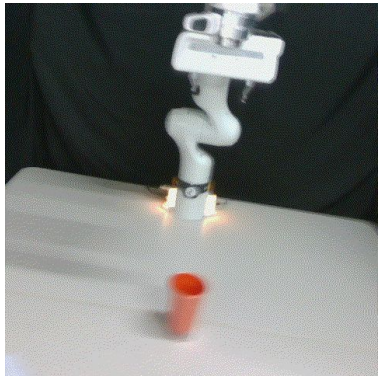
(From Younggyo)

Multi-View MAE

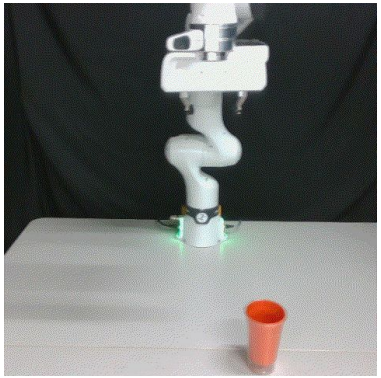
- **Zero-Shot Sim2Real Transfer with Hand-held Cameras**
  - Without proprioceptive states, depth, and adaptation



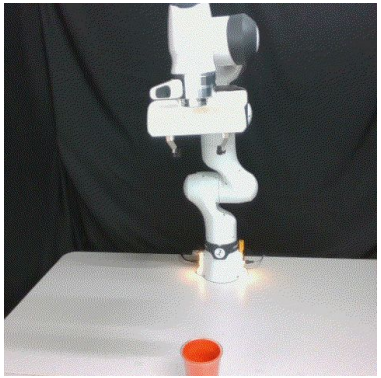
Rotation



Shake



Translation



Zoom

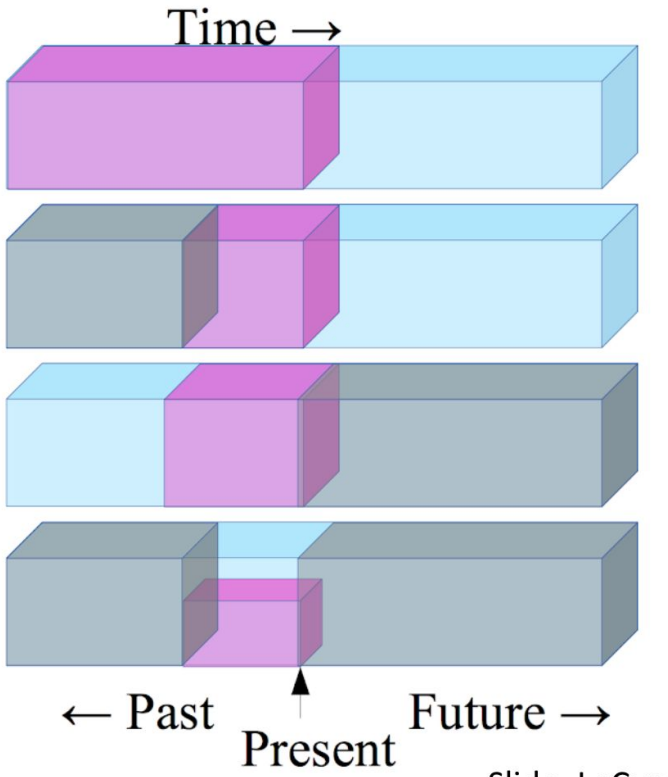
(From Younggyo)

Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2

Predicting neighbouring context

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- Pretend there is a part of the input you don't know and predict that.



Slide: LeCun

Word Embeddings

$$w^{aardvark} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{at} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots w^{zebra} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

(From 224n Stanford)

Word Embeddings

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

(From 224n Stanford)

Word Embeddings

Applying SVD to X:

$$\underset{|V|}{\begin{bmatrix} \\ X \\ \\ \end{bmatrix}} = \underset{|V|}{\begin{bmatrix} | & | & \\ u_1 & u_2 & \cdots \\ | & | & \end{bmatrix}} \underset{|V|}{\begin{bmatrix} \sigma_1 & 0 & \cdots \\ 0 & \sigma_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}} \underset{|V|}{\begin{bmatrix} - & v_1 & - \\ - & v_2 & - \\ & \vdots & \end{bmatrix}}$$

(From 224n Stanford)

Word Embeddings

SVD approach suffers from:

- Sparsity
- SVD computation costs
- Infrequent words
- Noise from frequent words
- There are hacks to fix some of these (ex TF-IDF) but still not very reliable

(From 224n Stanford)

n-gram Language Models

Unigram

$$P(w_1, w_2, \cdots, w_n) = \prod_{i=1}^{n} P(w_i)$$

Bigram

$$P(w_1, w_2, \cdots, w_n) = \prod_{i=2}^{n} P(w_i | w_{i-1})$$

(From 224n Stanford)

word2vec

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

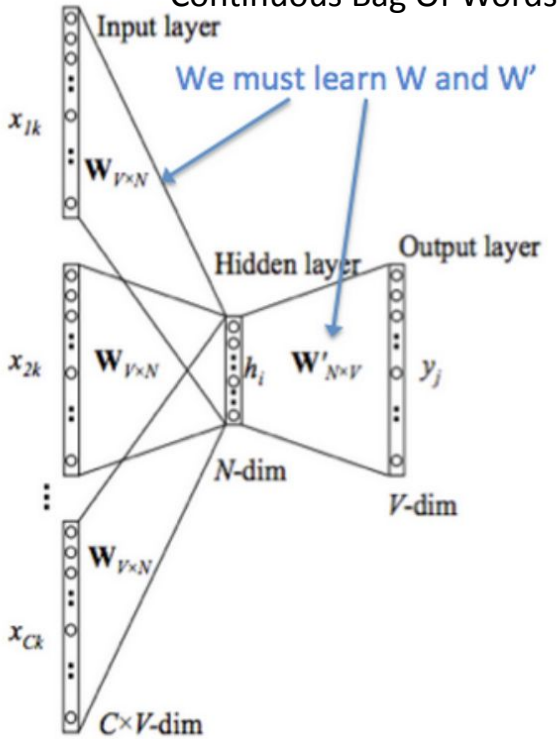
Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Abstract

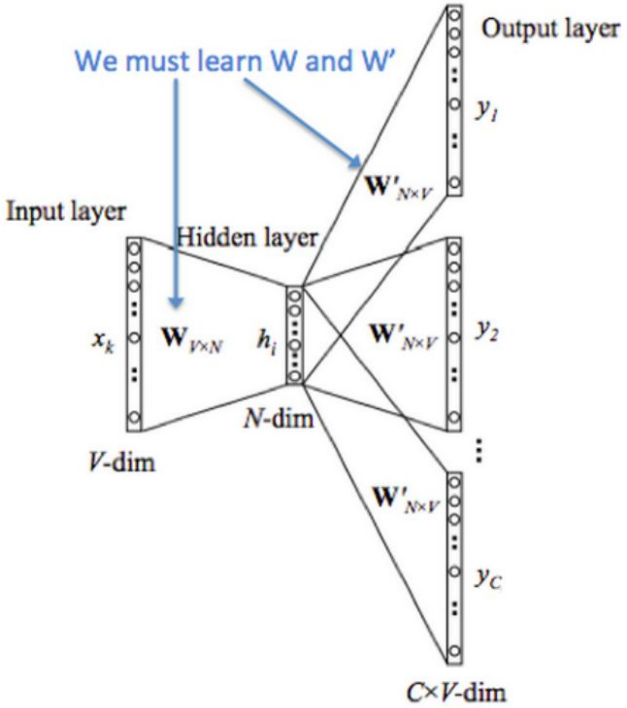
We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

word2vec

Continuous Bag Of Words (CBOW)



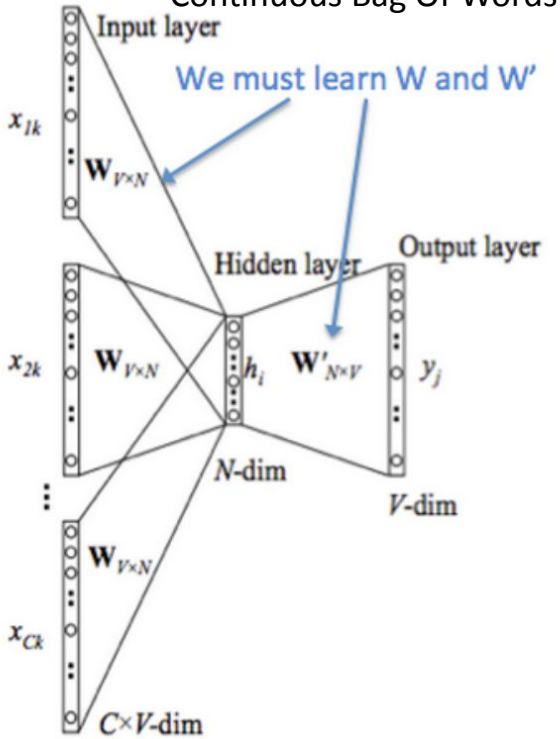
Skip Gram



(From 224n Stanford)

word2vec - CBOW

Continuous Bag Of Words (CBOW)

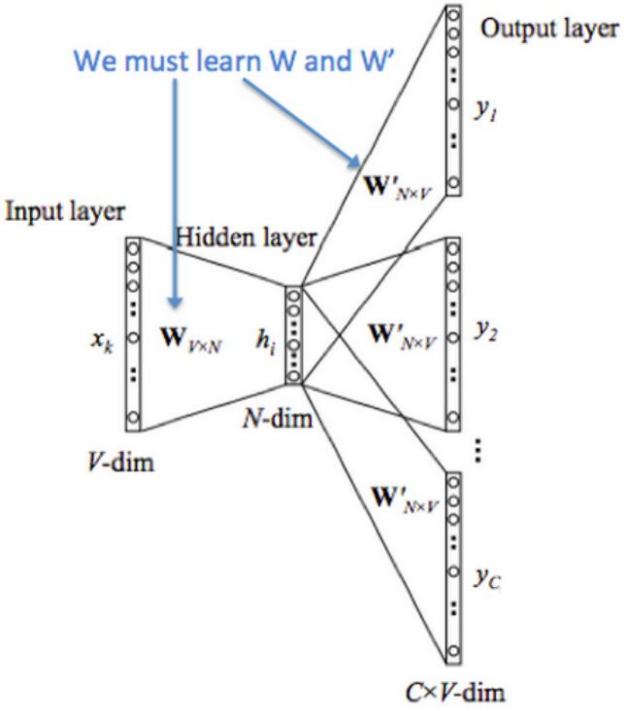


$$\begin{aligned}\text{minimize } J &= -\log P(w_c|w_{c-m},\ldots,w_{c-1},w_{c+1},\ldots,w_{c+m}) \\ &= -\log P(u_c|\hat{v}) \\ &= -\log \frac{\exp(u_c^T\hat{v})}{\sum_{j=1}^{|V|}\exp(u_j^T\hat{v})} \\ &= -u_c^T\hat{v} + \log\sum_{j=1}^{|V|}\exp(u_j^T\hat{v})\end{aligned}$$

(From 224n Stanford)

word2vec - Skip Gram

Skip Gram



$$\begin{aligned}
\text{minimize } J &= -\log P(w_{c-m},\ldots,w_{c-1},w_{c+1},\ldots,w_{c+m}|w_c) \\
&= -\log \prod_{j=0,j\neq m}^{2m} P(w_{c-m+j}|w_c) \\
&= -\log \prod_{j=0,j\neq m}^{2m} P(u_{c-m+j}|v_c) \\
&= -\log \prod_{j=0,j\neq m}^{2m} \frac{\exp(u_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|}\exp(u_k^T v_c)} \\
&= -\sum_{j=0,j\neq m}^{2m} u_{c-m+j}^T v_c + 2m\log\sum_{k=1}^{|V|}\exp(u_k^T v_c)
\end{aligned}$$

$$\begin{aligned}
J &= -\sum_{j=0,j\neq m}^{2m} \log P(u_{c-m+j}|v_c) \\
&= \sum_{j=0,j\neq m}^{2m} H(\hat{y}, y_{c-m+j})
\end{aligned}$$

(From 224n Stanford)

word2vec - Skip Gram

Skip-gram model

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_O|w_I)=\frac{\exp\left({v'_{w_O}}^{\top}v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^{\top}v_{w_I}\right)}$$

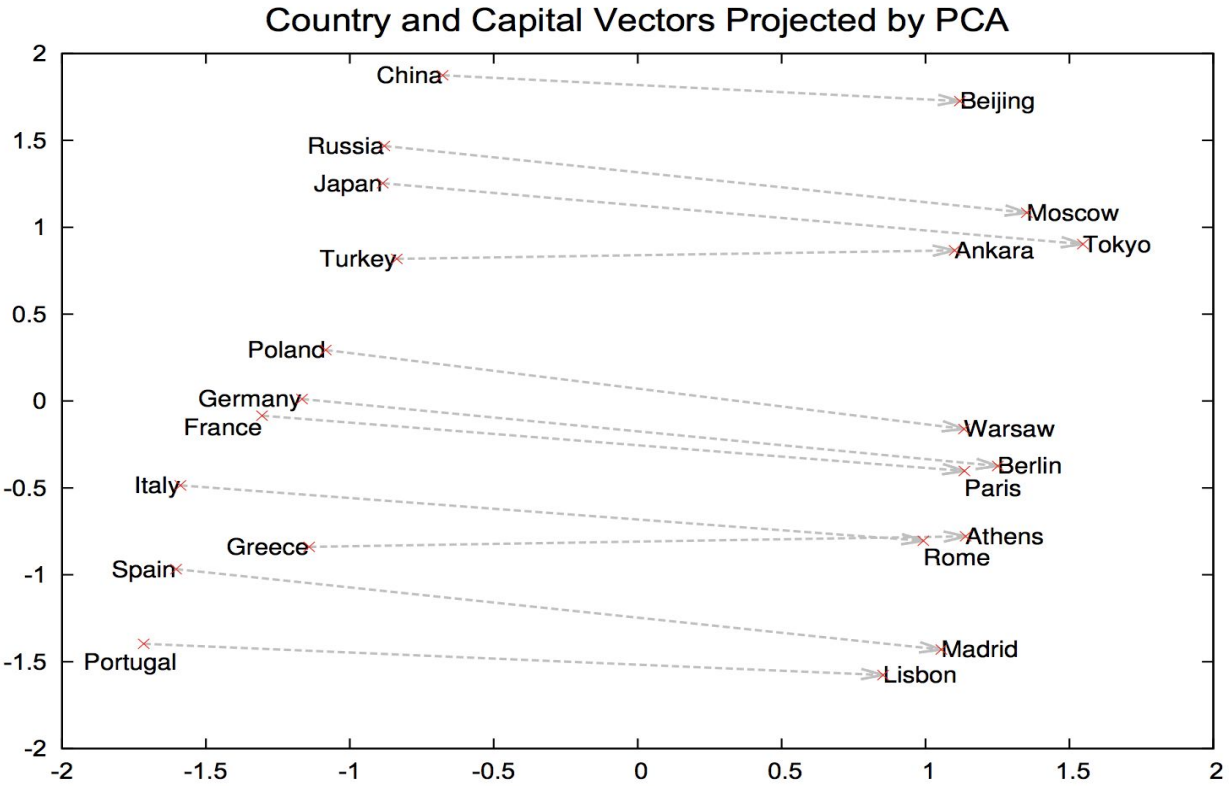
Don't have to have the denominator over all words in the vocabulary

- Can use negative sampling

$$\log\sigma({v'_{w_O}}^{\top}v_{w_I})+\sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\sigma(-{v'_{w_i}}^{\top}v_{w_I})\right]$$

$$P(w_i)=1-\sqrt{\frac{t}{f(w_i)}}$$

word2vec


Country and Capital Vectors Projected by PCA
China
Beijing
Russia
Japan
Moscow
Tokyo
Turkey
Ankara
Poland
Warsaw
Germany
France
Berlin
Paris
Italy
Athens
Greece
Rome
Spain
Madrid
Portugal
Lisbon
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

word2vec

	NEG-15 with 10^{-5} subsampling	HS with 10^{-5} subsampling
Vasco de Gama	Lingsugur	Italian explorer
Lake Baikal	Great Rift Valley	Aral Sea
Alan Bean	Rebbeca Naomi	moonwalker
Ionian Sea	Ruegen	Ionian Islands
chess master	chess grandmaster	Garry Kasparov

Table 4: Examples of the closest entities to the given short phrases, using two different models.

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

GloVe

Consider counting based statistical approaches

Word co occurrences X, where X_{ij} is the number of times j occurs in the context of i

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Ratios of co-occurrence probabilities can encode meaning

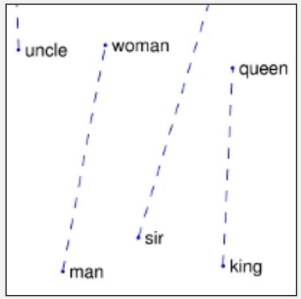
GloVe

Vector dot product to be similar to likelihood of of their co occurrence

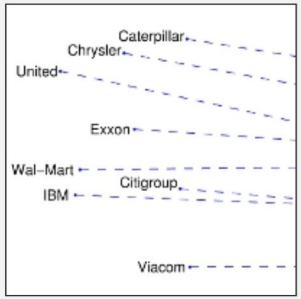
$$w_i \cdot w_j = \log P(i|j)$$

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

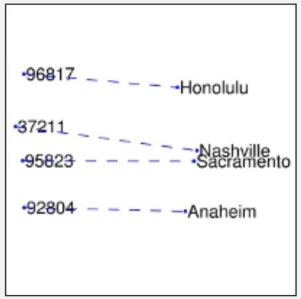
GloVe



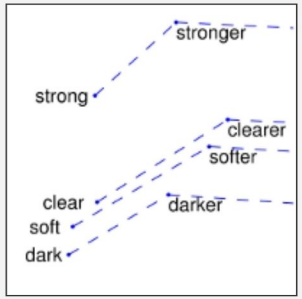
man - woman



company - ceo



city - zip code



comparative - superlative

0. *frog*
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus


3. litoria


4. leptodactylidae


5. rana


7. eleutherodactylus

Table 2: Results on the word analogy task, given as percent accuracy. Underlined scores are best within groups of similarly-sized models; bold scores are best overall. HPCA vectors are publicly available[2]; (i)vLBL results are from (Mnih et al., 2013); skip-gram (SG) and CBOW results are from (Mikolov et al., 2013a,b); we trained SG† and CBOW† using the `word2vec` tool[3]. See text for details and a description of the SVD models.

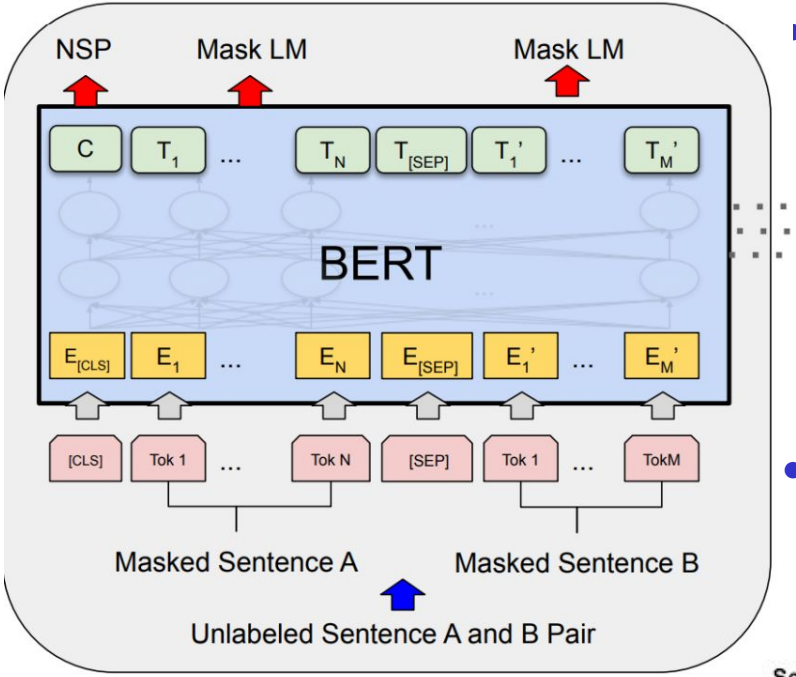
Model	Dim.	Size	Sem.	Syn.	Tot.
ivLBL	100	1.5B	55.9	50.1	53.2
HPCA	100	1.6B	4.2	16.4	10.8
GloVe	100	1.6B	67.5	54.3	60.3
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	64.8	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	80.8	61.5	70.3
SVD	300	6B	6.3	8.1	7.3
SVD-S	300	6B	36.7	46.6	42.1
SVD-L	300	6B	56.6	63.0	60.1
CBOW†	300	6B	63.6	67.4	65.7
SG†	300	6B	73.0	66.0	69.1
GloVe	300	6B	77.4	67.0	71.7
CBOW	1000	6B	57.3	68.9	63.7
SG	1000	6B	66.1	65.1	65.6
SVD-L	300	42B	38.4	58.2	49.2
GloVe	300	42B	**81.9**	**69.3**	**75.0**

BERT

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Oct 2018



Pre-training

- Task 1 - Masked Language Model:
  - 15% mask ratio
  - 10% of the time use a random token
  - 10% of the time leave unchanged
  - Loss only on masked tokens

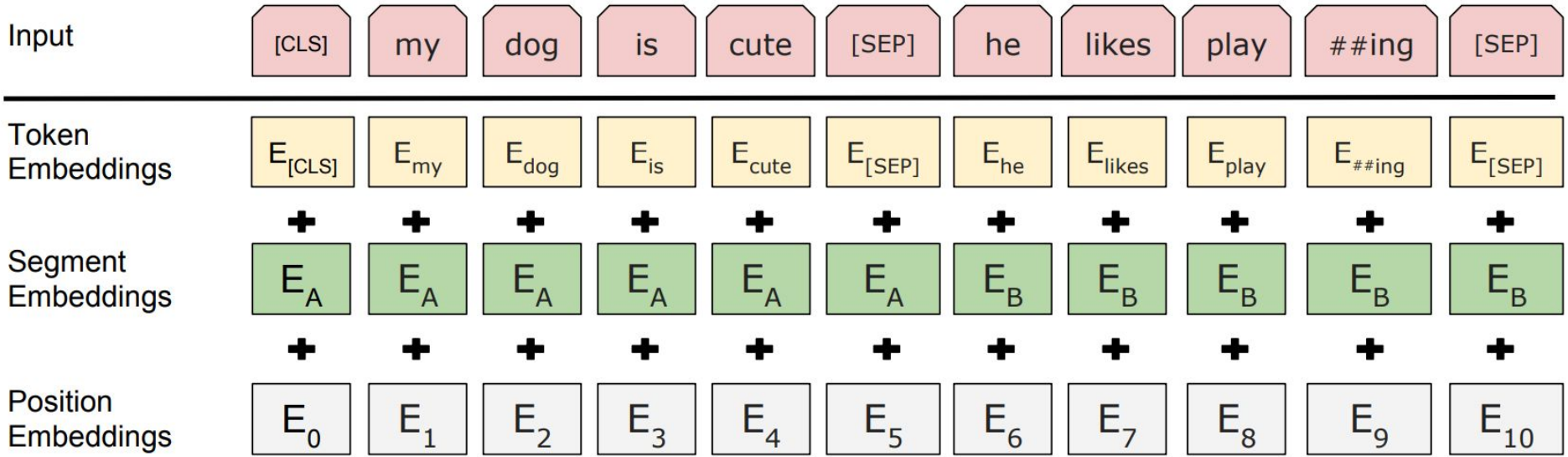
Input: `The man went to the [MASK]1 . He bought a [MASK]2 of milk .`
Labels: `[MASK]1 = store; [MASK]2 = gallon`

- Task 2 - Next Sentence Prediction
  - 50/50 next sentence directly follows vs a random one from the dataset

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

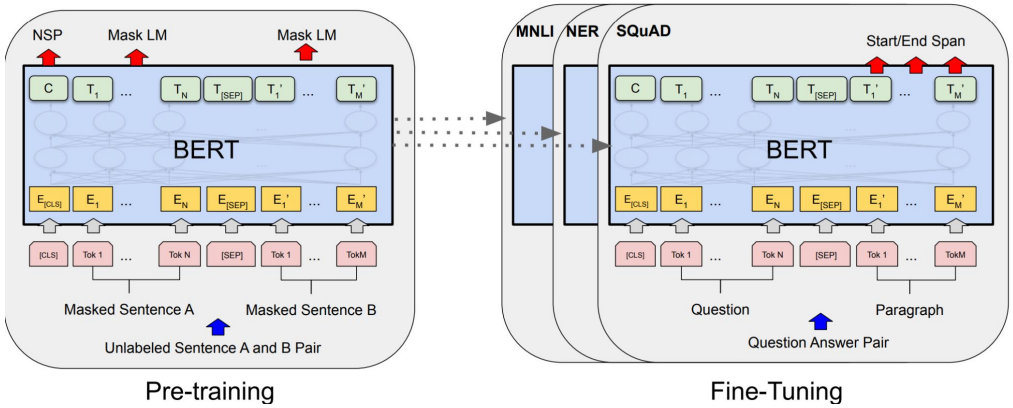
BERT


Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
E[CLS] Emy Edog Eis Ecute E[SEP] Ehe Elikes Eplay E##ing E[SEP]
Segment Embeddings
EA EA EA EA EA EA EB EB EB EB EB
Position Embeddings
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10

BERT

Pre-training data:

- BookCorpus (800M words)
- English Wikipedia (2500M words)



Fine Tuning

- For each task, inputs and outputs given to BERT
- Pretraining enables
  - Sentence A/B type tasks, ie sentence pairs in paraphrasing, hypothesis-premise pairs, question passage pairs
  - Token level tasks by looking at features per token
  - CLS for whole sentence level tasks.
- Evaluate on GLUE - 11 NLP tasks

BERT

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard).

BERT

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $\text{BERT}_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

BERT

Feature based

- Extract out frozen features
- Learn classifier for Named Entity Recognition task

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

RoBERTa

RoBERTa: A Robustly Optimized BERT Pretraining Approach

Yinhan Liu[*§] Myle Ott[*§] Naman Goyal[*§] Jingfei Du[*§] Mandar Joshi[†]
Danqi Chen[§] Omer Levy[§] Mike Lewis[§] Luke Zettlemoyer[†§] Veselin Stoyanov[§]

Jul 2019

RoBERTa

Greatly simplify the process to train BERT

- Dynamic Masking
  - Original BERT performed masking at the data processing step
- Next sentence prediction loss not needed

Masking	SQuAD 2.0	MNLI-m	SST-2
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
$\text{BERT}_{\text{BASE}}$	88.5/76.3	84.3	92.8	64.3
$\text{XLNet}_{\text{BASE}}$ (K = 7)	–/81.3	85.8	92.7	66.1
$\text{XLNet}_{\text{BASE}}$ (K = 6)	–/81.0	85.6	93.4	66.7

RoBERTa

- Training with large batches
- Text encoding with BPE (50K vocab size)

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

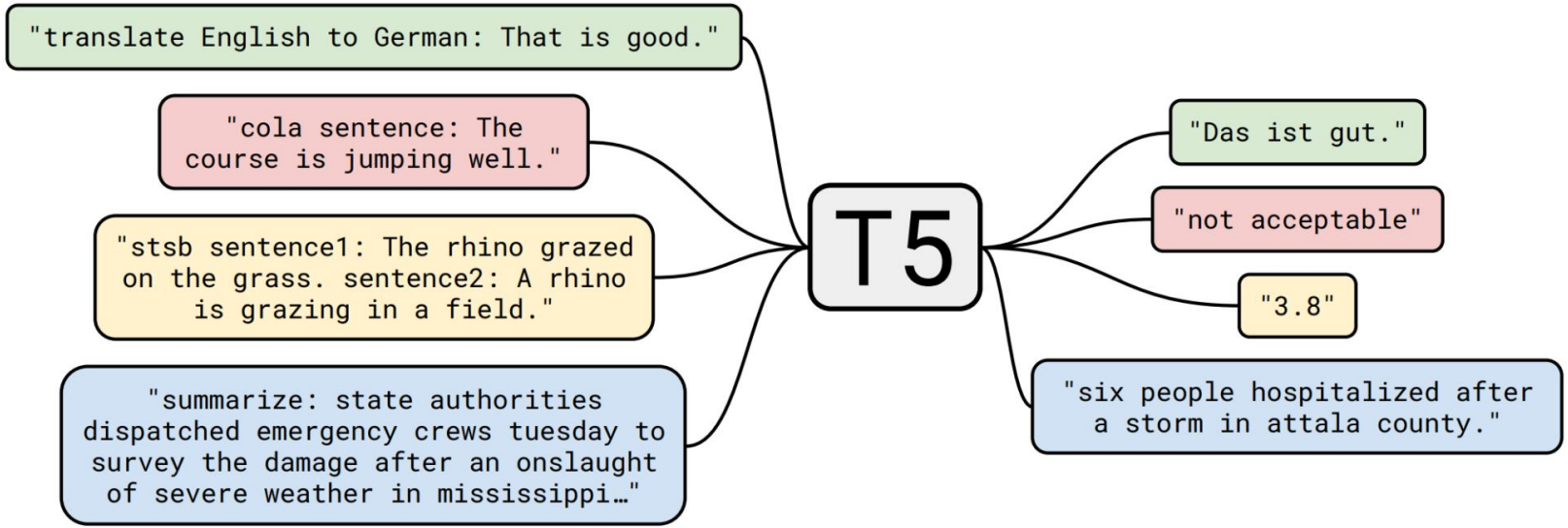
Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

T5

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel*	CRAFFEL@GMAIL.COM
Noam Shazeer*	NOAM@GOOGLE.COM
Adam Roberts*	ADAROB@GOOGLE.COM
Katherine Lee*	KATHERINELEE@GOOGLE.COM
Sharan Narang	SHARANNARANG@GOOGLE.COM
Michael Matena	MMATENA@GOOGLE.COM
Yanqi Zhou	YANQIZ@GOOGLE.COM
Wei Li	MWEILI@GOOGLE.COM
Peter J. Liu	PETERJLIU@GOOGLE.COM

T5

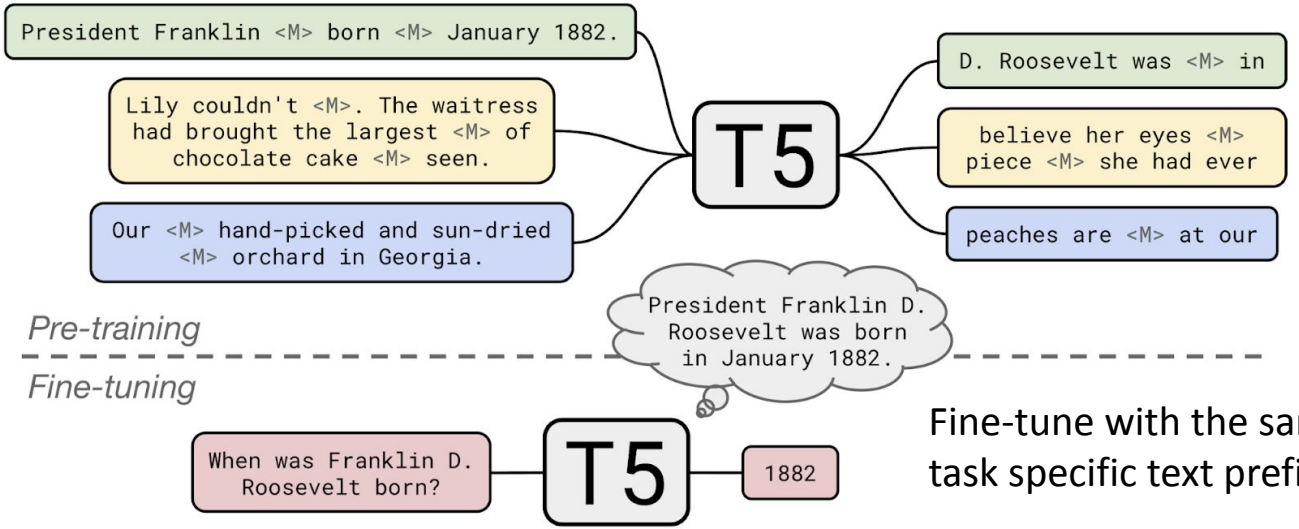


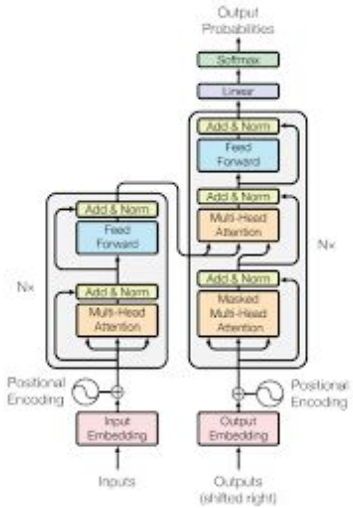
Cast all tasks as language input, language output

Explore different architectures and pre training tasks

T5

Finetuning



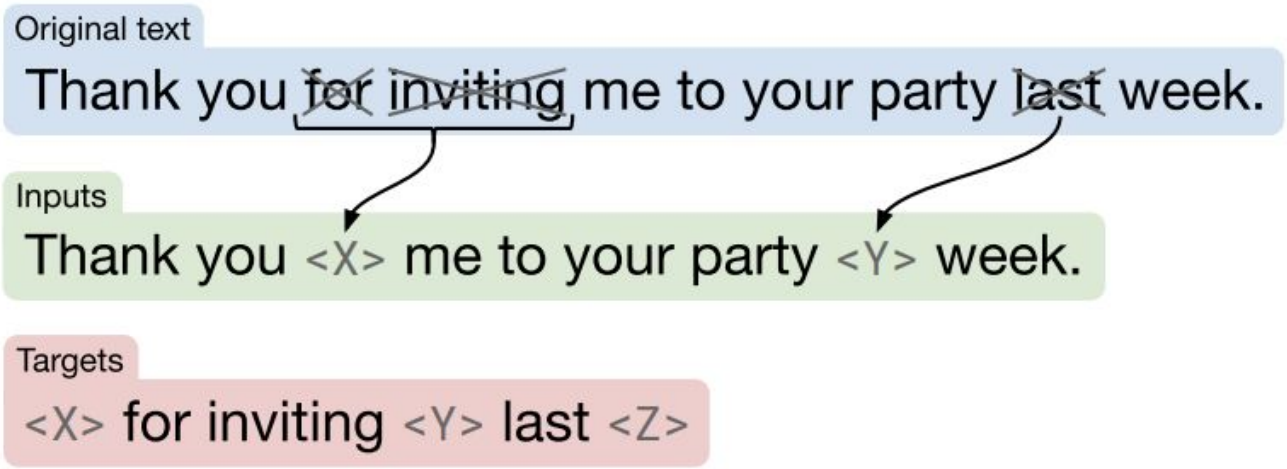


Fine-tune with the same input output format. Add a task specific text prefix to the model, ex.

Input: translate English to German: That is good.
Output: Das ist gut.

T5

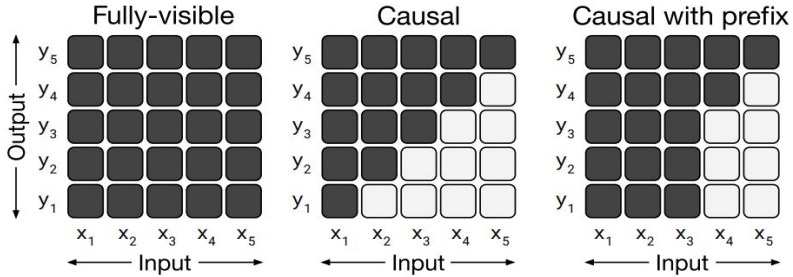
Denoising Objective



Sentinel token to delineate removed spans (unique ids that are added to the token vocab)

T5

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>



Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39
Encoder-decoder	LM	$2P$	M	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec, shared	LM	P	M	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec, 6 layers	LM	P	$M/2$	78.67	18.26	75.32	64.06	26.13	38.42	26.89
Language model	LM	P	M	73.78	17.54	53.81	56.51	25.23	34.31	25.38
Prefix LM	LM	P	M	79.68	17.84	76.87	64.86	26.28	37.51	26.76

T5

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

Table 4: Performance of the three disparate pre-training objectives described in Section 3.3.1.

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

Table 5: Comparison of variants of the BERT-style pre-training objective. In the first two variants, the model is trained to reconstruct the original uncorrupted text segment. In the latter two, the model only predicts the sequence of corrupted tokens.

T5

Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4[a]	69.2[b]	97.1[a]	**93.6**[b]	**91.5**[b]	92.7[b]	92.3[b]
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

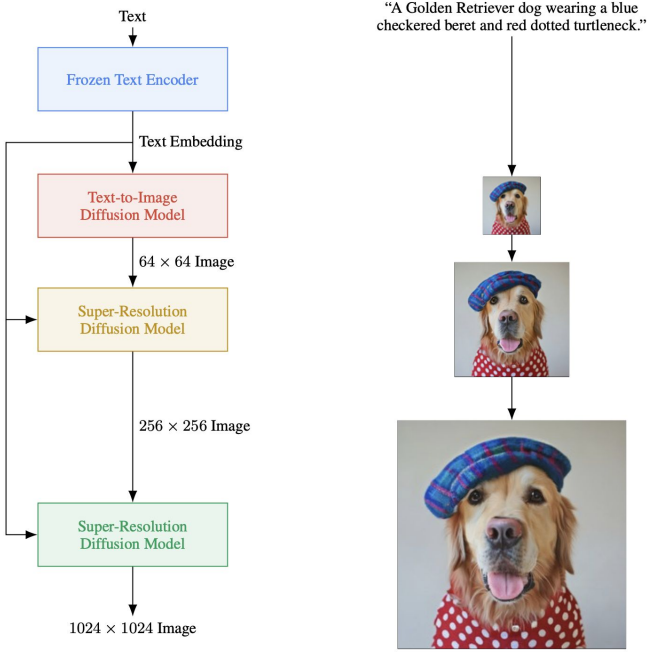
Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8[c]	**90.7**[b]	91.3[a]	91.0[a]	**99.2**[a]	89.2[a]	91.8[a]
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

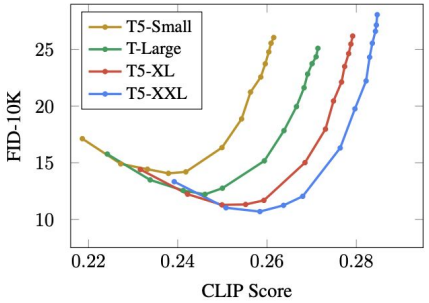
Model	SQuAD EM	SQuAD F1	SuperGLUE Average	BoolQ Accuracy	CB F1	CB Accuracy	COPA Accuracy
Previous best	90.1[a]	95.5[a]	84.6[d]	87.1[d]	90.5[d]	95.2[d]	90.6[d]
T5-Small	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-Base	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-Large	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	**91.26**	**96.22**	**88.9**	**91.2**	**93.9**	**96.8**	**94.8**

Model	MultiRC F1a	MultiRC EM	ReCoRD F1	ReCoRD Accuracy	RTE Accuracy	WiC Accuracy	WSC Accuracy
Previous best	84.4[d]	52.5[d]	90.6[d]	90.0[d]	88.2[d]	69.9[d]	89.0[d]
T5-Small	69.3	26.3	56.3	55.4	73.3	66.9	70.5
T5-Base	79.7	43.1	75.0	74.2	81.5	68.3	80.8
T5-Large	83.3	50.7	86.8	85.9	87.8	69.3	86.3
T5-3B	86.8	58.3	91.2	90.4	90.7	72.1	90.4
T5-11B	**88.1**	**63.3**	**94.1**	**93.4**	**92.5**	**76.9**	**93.8**

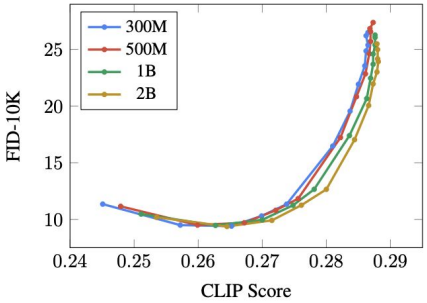
T5

T5's effect in Imagen - using T5's text encoder



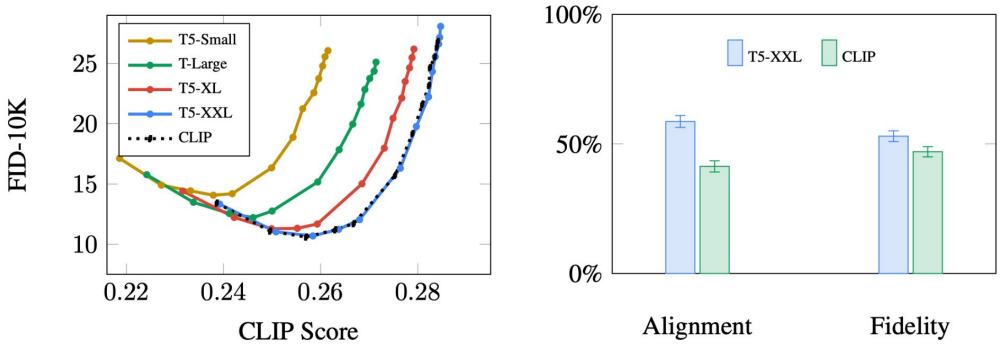


(a) Impact of encoder size.



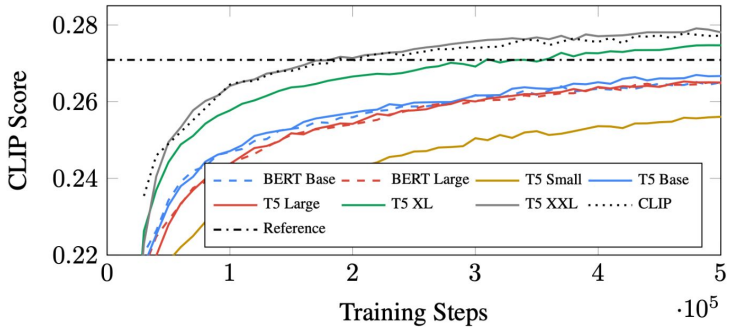
(b) Impact of U-Net size.

T5



(a) Pareto curves comparing various text encoders. (b) Comparing T5-XXL and CLIP on DrawBench.

Figure A.5: Comparison between text encoders for text-to-image generation. For Fig. A.5a, we sweep over guidance values of $[1, 1.25, 1.5, 1.75, 2, 3, 4, 5, 6, 7, 8, 9, 10]$



UL2

UL2: Unifying Language Learning Paradigms

Yi Tay* Mostafa Dehghani*

Vinh Q. Tran♯ Xavier Garcia♯ Jason Wei♯ Xuezhi Wang♯ Hyung Won Chung♯

Siamak Shakeri♯ Dara Bahri♭ Tal Schuster♭ Huaixiu Steven Zheng△

Denny Zhou△ Neil Houlsby△ Donald Metzler△

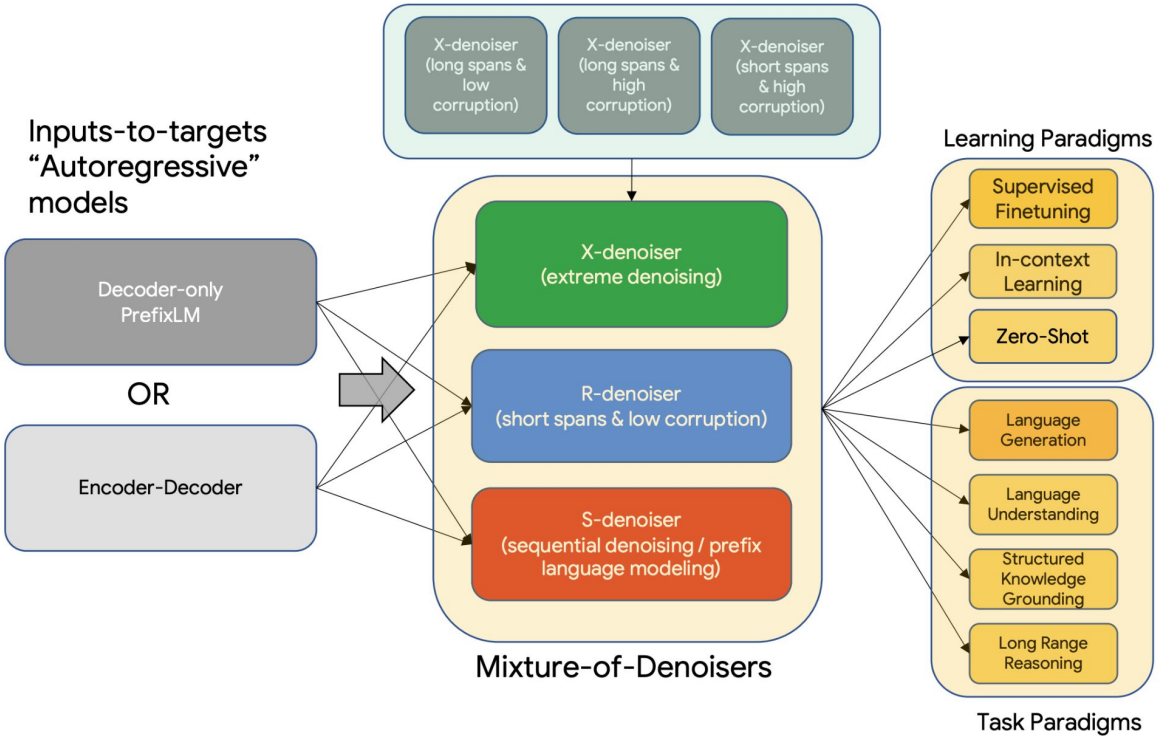
Google Brain

UL2

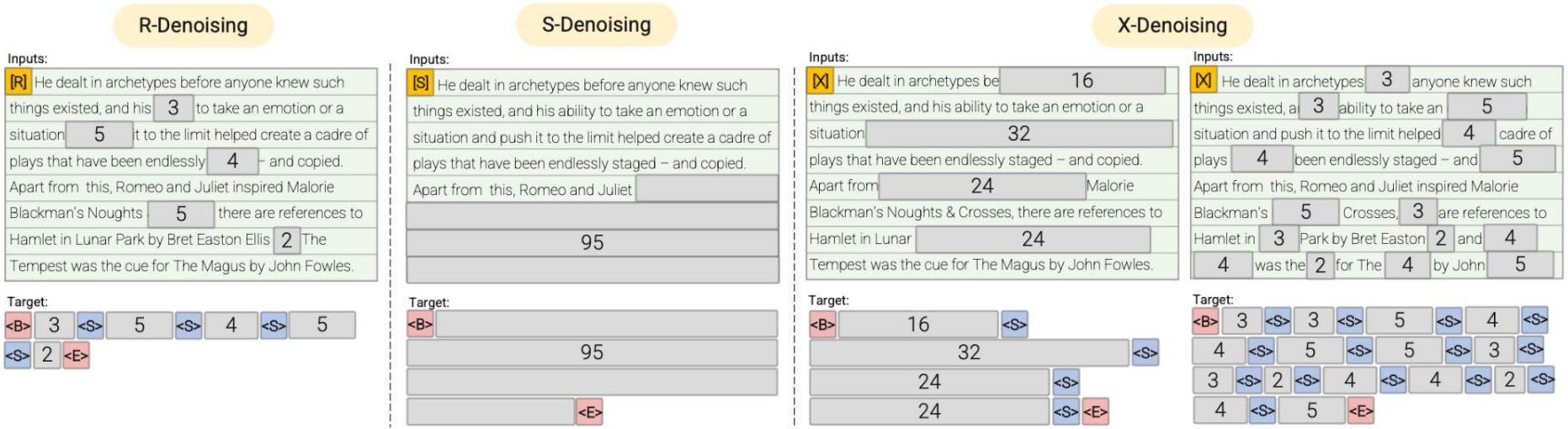
Does training on different pre training tasks help?

Formulate 3 types of pretraining

- Extreme denoisin
- Low corruption
- Sequential denoising



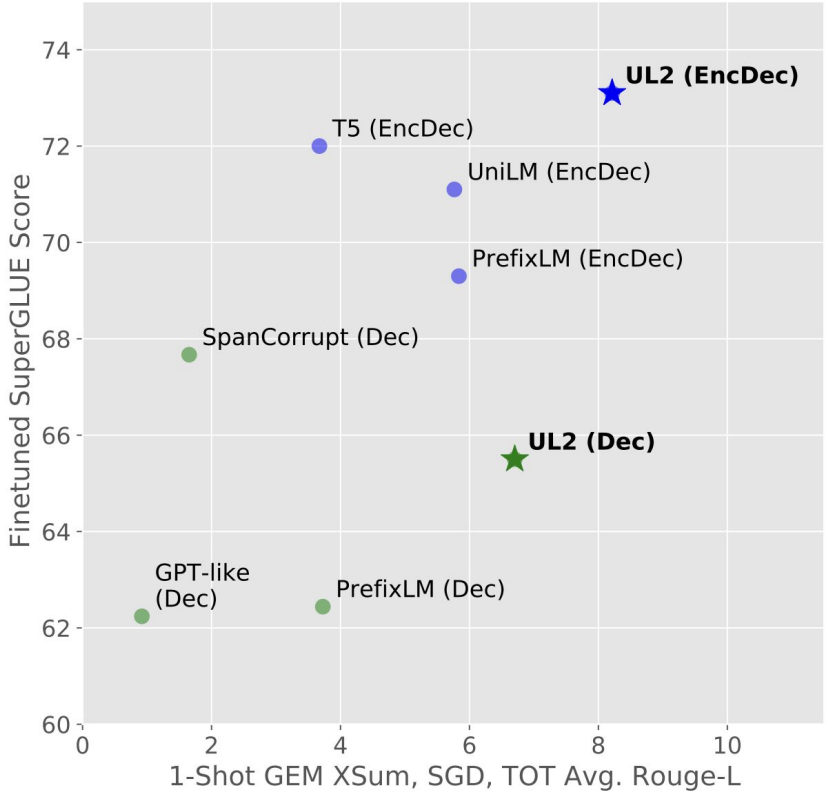
UL2


R-Denoising
Inputs:
[R] He dealt in archetypes before anyone knew such
things existed, and his 3 to take an emotion or a
situation 5 it to the limit helped create a cadre of
plays that have been endlessly 4 – and copied.
Apart from this, Romeo and Juliet inspired Malorie
Blackman's Noughts 5 there are references to
Hamlet in Lunar Park by Bret Easton Ellis 2 The
Tempest was the cue for The Magus by John Fowles.
Target:
<B> 3 <S> 5 <S> 4 <S> 5
<S> 2 <E>
S-Denoising
Inputs:
[S] He dealt in archetypes before anyone knew such
things existed, and his ability to take an emotion or a
situation and push it to the limit helped create a cadre of
plays that have been endlessly staged – and copied.
Apart from this, Romeo and Juliet
95
Target:
<B>
95
<E>
X-Denoising
Inputs:
[X] He dealt in archetypes be 16
things existed, and his ability to take an emotion or a
situation 32
plays that have been endlessly staged – and copied.
Apart from 24 Malorie
Blackman's Noughts & Crosses, there are references to
Hamlet in Lunar 24
Tempest was the cue for The Magus by John Fowles.
Target:
<B> 16 <S>
32 <S>
24 <S>
24 <S> <E>
Inputs:
[X] He dealt in archetypes 3 anyone knew such
things existed, a 3 ability to take an 5
situation and push it to the limit helped 4 cadre of
plays 4 been endlessly staged – and 5
Apart from this, Romeo and Juliet inspired Malorie
Blackman's 5 Crosses, 3 are references to
Hamlet in 3 Park by Bret Easton 2 and 4
4 was the 2 for The 4 by John 5
Target:
<B> 3 <S> 3 <S> 5 <S> 4 <S>
4 <S> 5 <S> 5 <S> 3 <S>
3 <S> 2 <S> 4 <S> 4 <S> 2 <S>
4 <S> 5 <E>

UL2

Table 9: Results on SuperGLUE dev set. We compare with T5-11B (Raffel et al., 2019), ST-MoE-32B (Zoph et al., 2022) and PaLM-8B, PaLM-62B and PaLM-540B (Chowdhery et al., 2022). Scores reported are the peak validation scores per task.

Model	BoolQ	CB	CoPA	MultiRC	Record	RTE	WiC	WSC	Avg
PaLM 62B	90.6	96.4/95.7	98.0	87.7/61.9	93.0/92.4	89.5	75.9	96.2	89.2
PaLM 540B	92.2	100/100	100	90.1/69.2	94.0/94.6	95.7	78.8	100	92.6
ST-MoE 32B$_{269B}$	**93.1**	**100/100**	**100**	**90.4/69.9**	**95.0/95.6**	**95.7**	**81.0**	**100**	**93.2**
PaLM 8B	87.6	96.4/92.1	86.0	81.6/64.0	89.7/89.3	84.5	73.4	88.5	83.4
T5 11B	90.8	94.9/96.4	98.0	87.4/66.1	93.8/93.2	93.9	77.3	96.2	89.9
UL2 20B	90.8	98.7/98.2	99.0	88.4/64.8	93.7/93.2	92.1	77.3	98.1	90.7



Outline

- Reconstruct from a corrupted (or partial) version
  - Denoising AutoEncoder / Diffusion
  - In-painting / Masked AutoEncoder: MAE, VideoMAE, Audio-MAE, BeIT, M3AE, MultiMAE, SiamMAE
  - Colorization, Split-Brain AutoEncoder
- Visual common sense tasks
  - Relative patch prediction
  - Jigsaw puzzles
  - Rotation
- Contrastive Learning
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination: SimCLR, MoCo-v1,2,3, BYOL
- Feature Prediction: DINO/DINOv2/iBOT, JEPA, I-JEPA, V-JEPA
- Text-Image: CLIP, LiT, SigLIP, FLIP, SLIP, CoCa, BLIP/BLIP-2, ImageBind
- RL and Control: R3M, CURL, MVP, MTM, Multi-View MAE and Masked World Models for Visual Control
- Language
  - Word2vec and Glove
  - BERT, RoBERTa, T5, UL2