DINOv2: Learning Robust Visual Features without Supervision

OQUAB, M., DARCET, T., MOUTAKANNI, T., ET AL. (2024).

Presented by
Dahye Kim

1-1.Motivation & Background

SSL revolutionized NLP

Proves that models can learn representations from raw data without explicit labels

Success of foundation models in NLP

Suggests similar approaches could work in computer vision

Goal: Vision Fountation Model

Develop general-purpose visual features that work across image distributions and tasks without finetuning.

1-2. Challenges in Vision Foundation Models

1. **Text-supervised models (e.g.CLIP)** rely on image-text pairs, limiting their ability to capture pixel-level details

2. Prior SSL methods (e.g., DINO, iBOT) struggled with **scalability** and **dataset quality**

Mostly trained on small curated datasets (e.g., ImageNet-1k)

Poor generalization beyond ImageNet

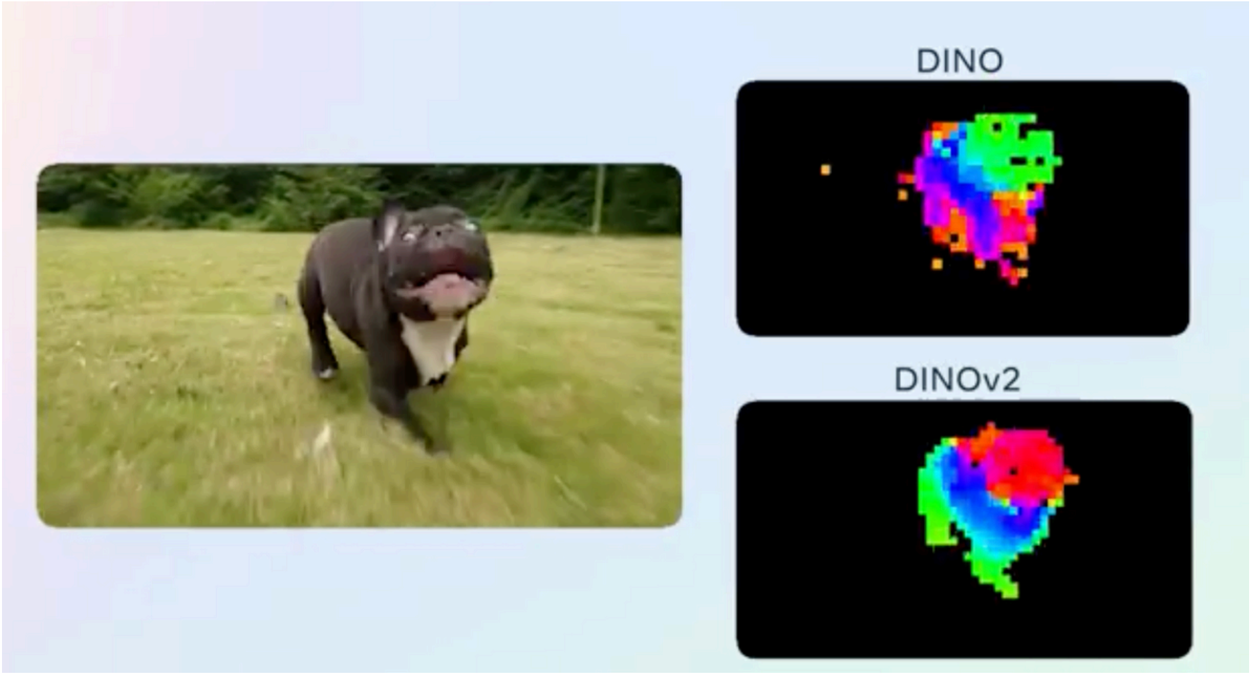
1-3. Evolution From DINO to DINOv2

DINO (2021)

introduced SSL via student-teacher distillation.

Limitations of DINO

Small dataset, no patch-level learning, inefficient training.



1-4. Key Contributions of *DINOv2*

LVD-142M dataset for better pretraining

Advanced SSL techniques (DINO + iBOT + SWAV losses)

Scalable training efficiency with lower memory usage

State-of-the-art performance across multiple vision benchmarks

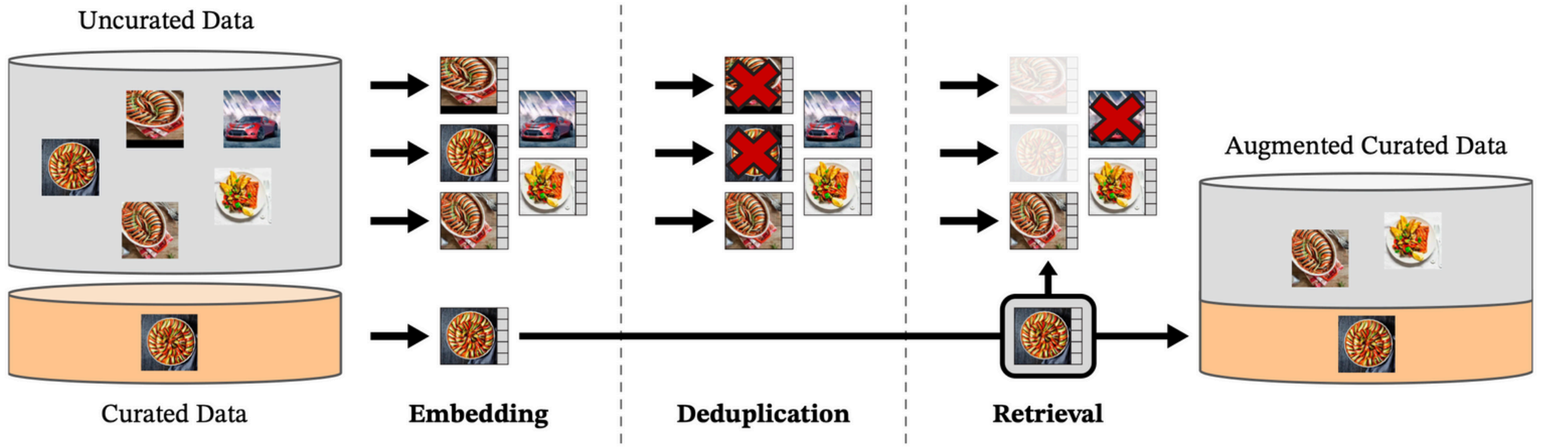
2. DINOv2: Data Pipeline

Building LVD-142M: A Curated Pretraining Dataset

- Assembled from a large pool of uncurated images by retrieving those similar to curated datasets like ImageNet-22k
- Filtering process:
  - Deduplication (removes near-duplicates to ensure diversity)
  - NSFW filtering (removes inappropriate content)
  - Face blurring (enhances privacy)

Self-Supervised Image Retrieval for Curation

- Used a self-supervised ViT-H/16 model to extract image embeddings.
- Cosine similarity + k-means clustering to retrieve diverse, high-quality images.
- Trade-off: Using 4 nearest neighbors (N=4) balances retrieval quality and dataset diversity.



3-1. DINOv2's Architecture : Vision Transformer (ViT)

Why Vision Transformers?

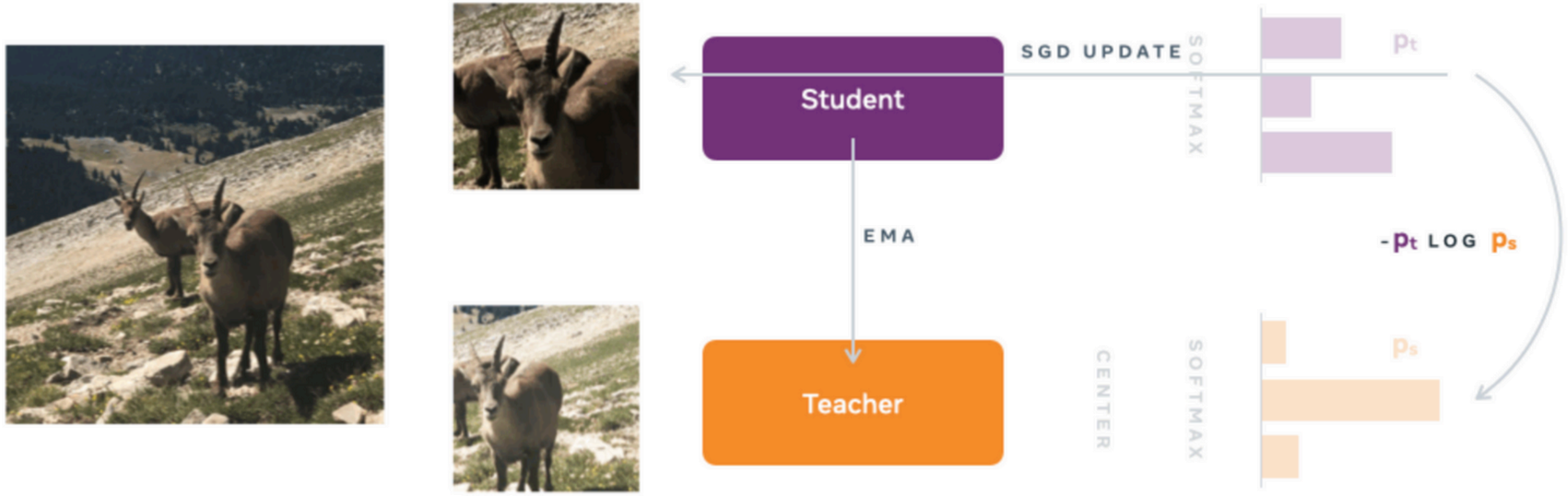
- Unlike CNNs, ViTs process images like **sequences**, capturing **long-range** dependencies effectively.
- **Self-attention** in ViTs allows better feature learning across an entire image.

How DINOv2 Uses ViTs:

- Uses a **ViT-based architecture** as the backbone.
- Processes both **global (image-level) and local (patch-level) features**.
- **Enables self-supervised training** via the student-teacher framework.

3-2. Self-Supervised Learning Framework

Student-Teacher Framework (DINO/iBOT Inspired)



Student model

Learns by matching the teacher's softmax outputs

Teacher model

Exponential Moving Average (EMA) of the student

Training encourages feature consistency across different image augmentations

Self-Distillation Mechanism

3-3. Loss Functions Used

DINO Loss (Image-Level Representation)

iBOT Loss (Patch-Level Representation)

Sinkhorn-Knopp Centering (Inspired by SWAV)

KoLeo Regularizer

DINO Loss: Image-level objective

- Cross-entropy between student & teacher feature distributions.
- Uses "prototype vectors" for contrastive learning.

$$\mathcal{L}_{DINO} = -\sum p_t \log p_s$$

3-3. Loss Functions Used

DINO Loss (Image-Level Representation)

iBOT Loss (Patch-Level Representation)

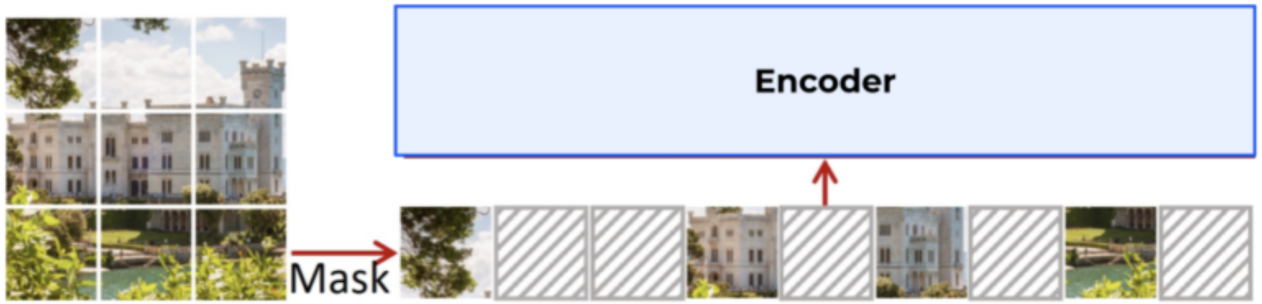
Sinkhorn-Knopp Centering (Inspired by SWAV)

KoLeo Regularizer

iBOT Loss: Patch-Level Objective

- Masked Image Modeling (MIM):
- Random patches are masked in the student but visible in the teacher.
- The student must predict missing patches, learning local context.

$$\mathcal{L}_{iBOT} = -\sum_{i} p_{ti} \log p_{si}$$



3-3. Loss Functions Used

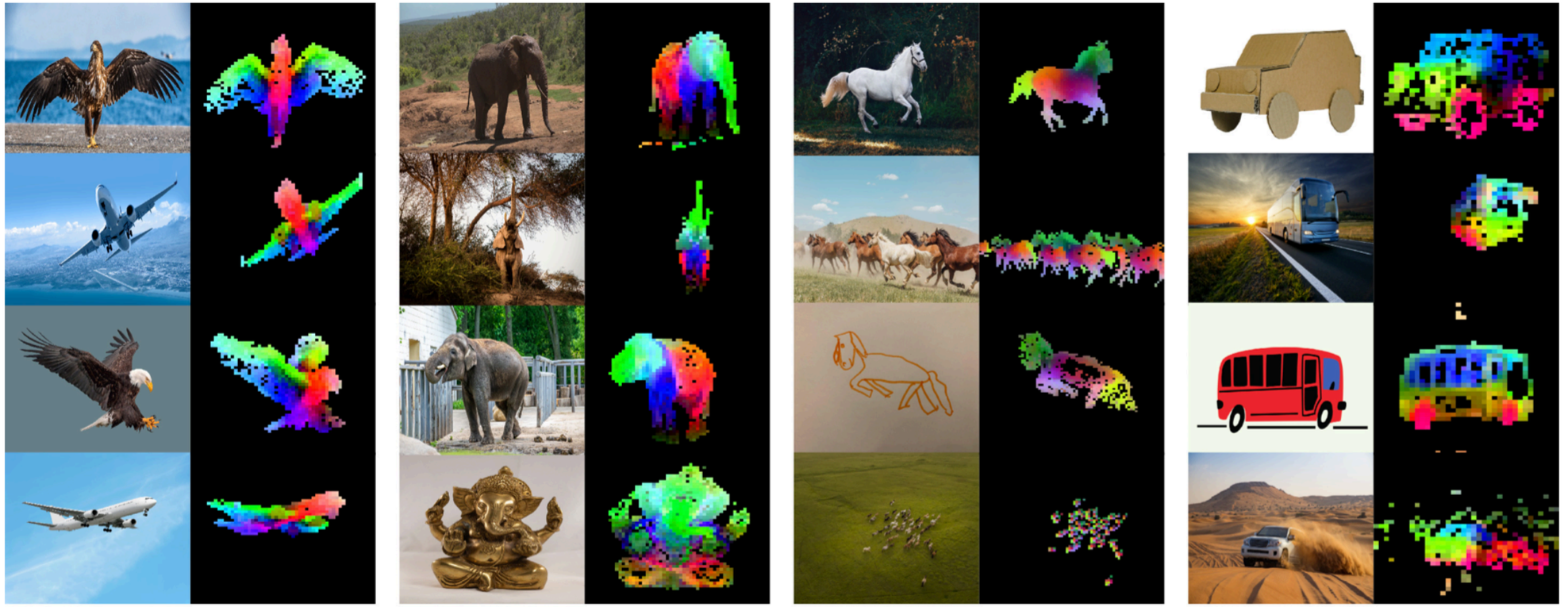


Figure 1: **Visualization of the first PCA components.** We compute a PCA between the patches of the images from the same column (a, b, c and d) and show their first 3 components. Each component is matched to a different color channel. Same parts are matched between related images despite changes of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

3-3. Loss Functions Used

DINO Loss (Image-Level Representation)

iBOT Loss (Patch-Level Representation)

Untying head weights between both objectives to solve
1) underfitting issue in the **patch-level** classification task and
2) overfitting issue in the **image-level** classification task

Sinkhorn-Knopp Centering (Inspired by SWAV)

KoLeo Regularizer

3-3. Loss Functions Used

DINO Loss (Image-Level Representation)

iBOT Loss (Patch-Level Representation)

Sinkhorn-Knopp Centering (Inspired by SWAV)

KoLeo Regularizer

Sinkhorn-Knopp Centering (Inspired by SwAV)

- Normalizes teacher outputs for better feature distribution.
- Prevents representation collapse.

KoLeo Regularizer

- Spreads features uniformly to improve retrieval & instance recognition.

4-1. Efficient Implementation and Scaling: Memory & Compute Optimizations

Compared to the iBOT implementation, the DINOv2 code runs around 2× faster using only 1/3 of the memory.

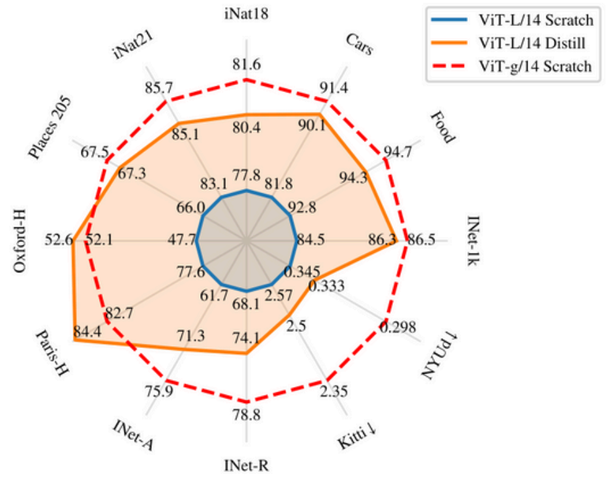
- Custom FlashAttention: Reduces memory usage & speeds up self-attention calculations.
- Sequence Packing:
  - Concatenates small & large image crops into a single long sequence.
  - Uses block-diagonal attention masking for efficiency.
- Efficient Stochastic Depth: Skips unnecessary residual computations → faster training.

4-2. Efficient Implementation and Scaling: Fully-Sharded Data Parallelism (FSDP)

- Instead of storing full model replicas per GPU, DINOv2 shards parameters across multiple GPUs.
- 50% lower communication costs compared to traditional Distributed Data Parallel (DDP).

4-3. Efficient Implementation and Scaling: Knowledge Distillation for Smaller Models

- ViT-L models learn from a frozen ViT-g teacher.
- Removes need for training small models from scratch.
- Maintains high performance with lower compute cost.



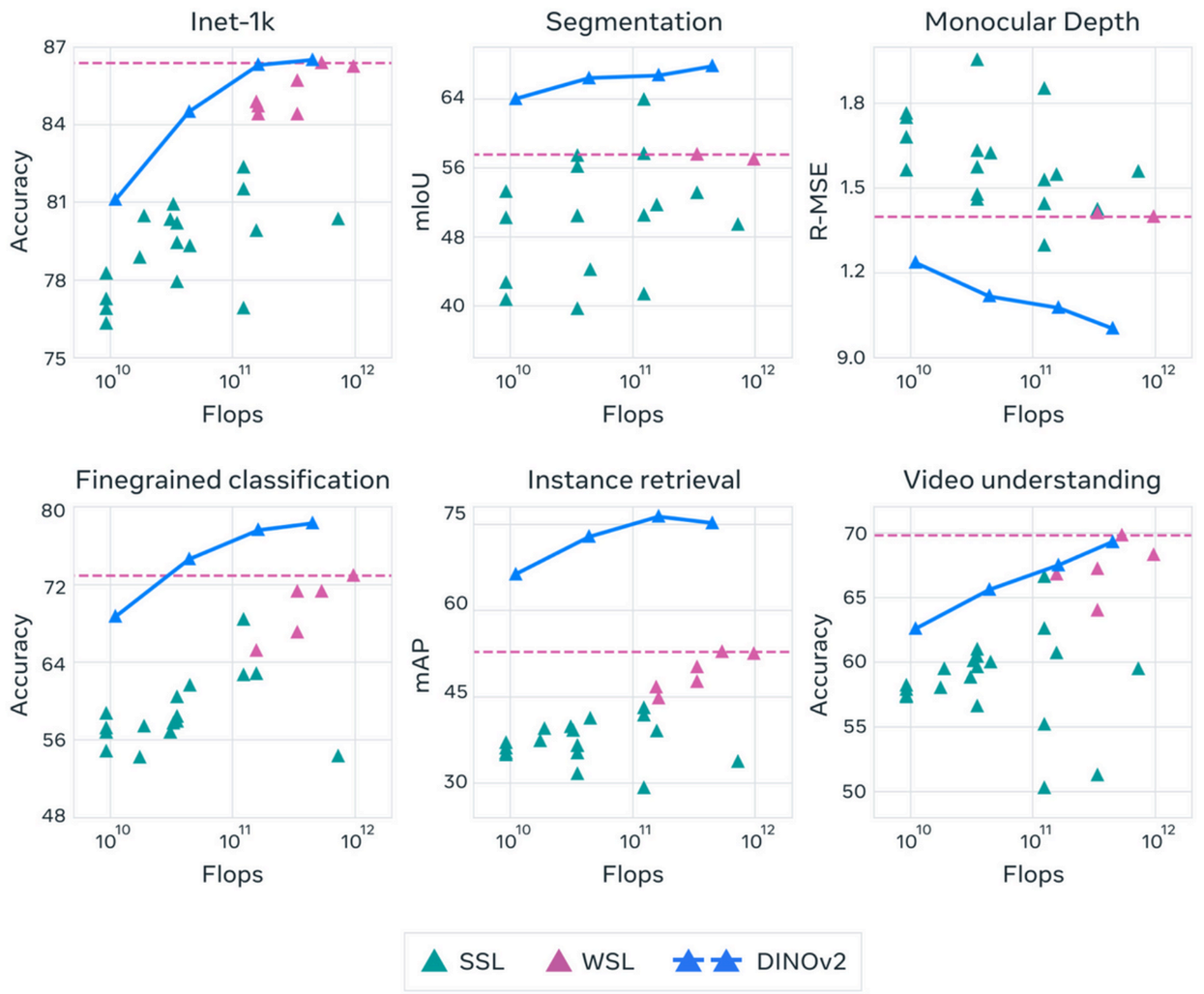
(a) Comparison on individual metrics

Arch	Method	INet-1k	Segm.	Depth↓	Classif.
ViT-g/14	Scratch	86.5	73.4	1.00	92.1
ViT-L/14	Scratch	84.5	72.2	1.10	90.2
ViT-L/14	Distill	**86.3**	**73.3**	**1.08**	**91.2**

Arch	Method	Finegr.	Retriev.	ARSketch	Video
ViT-g/14	Scratch	78.3	75.2	77.0	69.3
ViT-L/14	Scratch	75.8	71.3	69.5	67.3
ViT-L/14	Distill	**77.6**	**76.3**	**74.5**	**67.5**

(b) Averaged metrics on 8 vision tasks

5. Results


Inet-1k
Segmentation
Monocular Depth
Finegrained classification
Instance retrieval
Video understanding
Accuracy
mIoU
R-MSE
mAP
Flops
SSL
WSL
DINOv2

5-1. Results: ImageNet-1k Classification

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**

Linear evaluation on ImageNet-1k of frozen pretrained features

- +4.2% Top-1 Accuracy on ImageNet over iBOT.
- Matches or surpasses OpenCLIP and EVA-CLIP

5-2. Results: Video and other image Classification

		Image classification			Video classification		
Feature	Arch	iNat2018	iNat2021	Places205	K400	UCF-101	SSv2
OpenCLIP	ViT-G/14	73.0	76.0	**69.8**	78.3	90.7	35.8
MAE	ViT-H/14	31.0	32.3	52.4	54.2	70.6	29.2
DINO	ViT-B/8	59.6	68.3	60.4	64.5	85.0	32.6
iBOT	ViT-L/16	66.3	74.6	64.4	72.6	88.6	**38.7**
DINOv2	ViT-S/14	69.0	74.2	62.9	67.8	87.0	33.1
	ViT-B/14	76.4	81.1	66.2	73.2	89.1	34.4
	ViT-L/14	80.4	85.1	67.3	76.3	90.5	35.6
	ViT-g/14	**81.6**	**85.7**	67.5	**78.4**	**91.2**	38.3

Table 7: **Linear evaluation on other image and video classification.** The image benchmarks contain a large quantity of fine-grained examples about objects or scenes. The video benchmarks cover action classification and human-object interaction. All the features are frozen with a linear probe on top.

5-3. Results: Instance Recognition

		Oxford		Paris		Met			AmsterTime
Feature	Arch	M	H	M	H	GAP	GAP-	ACC	mAP
OpenCLIP	ViT-G/14	50.7	19.7	79.2	60.2	6.5	23.9	34.4	24.6
MAE	ViT-H/14	11.7	2.2	19.9	4.7	7.5	23.5	30.5	4.2
DINO	ViT-B/8	40.1	13.7	65.3	35.3	17.1	37.7	43.9	24.6
iBOT	ViT-L/16	39.0	12.7	70.7	47.0	25.1	54.8	58.2	26.7
DINOv2	ViT-S/14	68.8	43.2	84.6	68.5	29.4	54.3	57.7	43.5
	ViT-B/14	72.9	49.5	90.3	78.6	36.7	63.5	66.1	45.6
	ViT-L/14	**75.1**	**54.0**	**92.7**	**83.5**	**40.0**	68.9	71.6	**50.0**
	ViT-g/14	73.6	52.3	92.1	82.6	36.8	**73.6**	**76.5**	46.7

Table 9: **Evaluation of frozen features on instance-level recognition.** We consider 4 different benchmarks and report their main metrics.

- Ranks images based on cosine similarity between features
- Significant mAP improvements (+41% over SSL on Oxford-Hard), +34% over weakly-supervised models on Oxford-Hard).

5-4. Results: Semantic Segmentation & Depth Estimation

Method	Arch.	ADE20k (62.9) lin.	ADE20k (62.9) +ms	CityScapes (86.9) lin.	CityScapes (86.9) +ms	Pascal VOC (89.0) lin.	Pascal VOC (89.0) +ms
OpenCLIP	ViT-G/14	39.3	46.0	60.3	70.3	71.4	79.2
MAE	ViT-H/14	33.3	30.7	58.4	61.0	67.6	63.3
DINO	ViT-B/8	31.8	35.2	56.9	66.2	66.4	75.6
iBOT	ViT-L/16	44.6	47.5	64.8	74.5	82.3	84.3
DINOv2	ViT-S/14	44.3	47.2	66.6	77.1	81.1	82.6
	ViT-B/14	47.3	51.3	69.4	80.0	82.5	84.9
	ViT-L/14	47.7	**53.1**	70.3	80.9	82.1	86.0
	ViT-g/14	**49.0**	53.0	**71.3**	**81.0**	**83.0**	**86.2**

Semantic segmentation on ADE20K, CityScapes and Pascal VOC with frozen features

Depth estimation with frozen features

- **Depth estimation outperforms OpenCLIP and weakly-supervised features.**

Method	Arch.	NYUd (0.330) lin. 1	NYUd (0.330) lin. 4	NYUd (0.330) DPT	KITTI (2.10) lin. 1	KITTI (2.10) lin. 4	KITTI (2.10) DPT	NYUd → SUN RGB-D (0.421) lin. 1	NYUd → SUN RGB-D (0.421) lin. 4	NYUd → SUN RGB-D (0.421) DPT
OpenCLIP	ViT-G/14	0.541	0.510	0.414	3.57	3.21	2.56	0.537	0.476	0.408
MAE	ViT-H/14	0.517	0.483	0.415	3.66	3.26	2.59	0.545	0.523	0.506
DINO	ViT-B/8	0.555	0.539	0.492	3.81	3.56	2.74	0.553	0.541	0.520
iBOT	ViT-L/16	0.417	0.387	0.358	3.31	3.07	2.55	0.447	0.435	0.426
DINOv2	ViT-S/14	0.449	0.417	0.356	3.10	2.86	2.34	0.477	0.431	0.409
	ViT-B/14	0.399	0.362	0.317	2.90	2.59	2.23	0.448	0.400	0.377
	ViT-L/14	0.384	0.333	0.293	2.78	2.50	2.14	0.429	0.396	0.360
	ViT-g/14	**0.344**	**0.298**	**0.279**	**2.62**	**2.35**	**2.11**	**0.402**	**0.362**	**0.338**

5-5. Qualitative Results

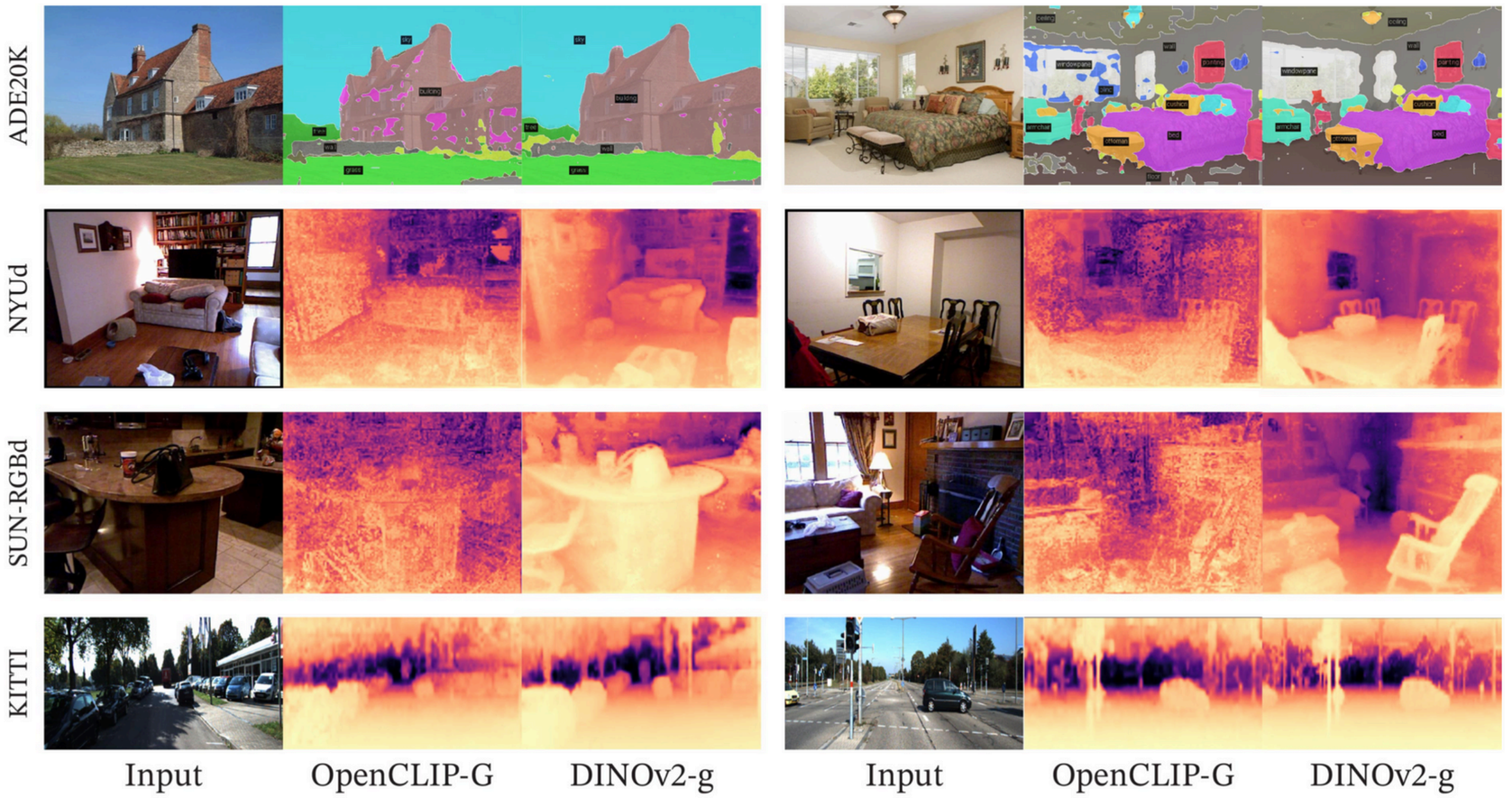


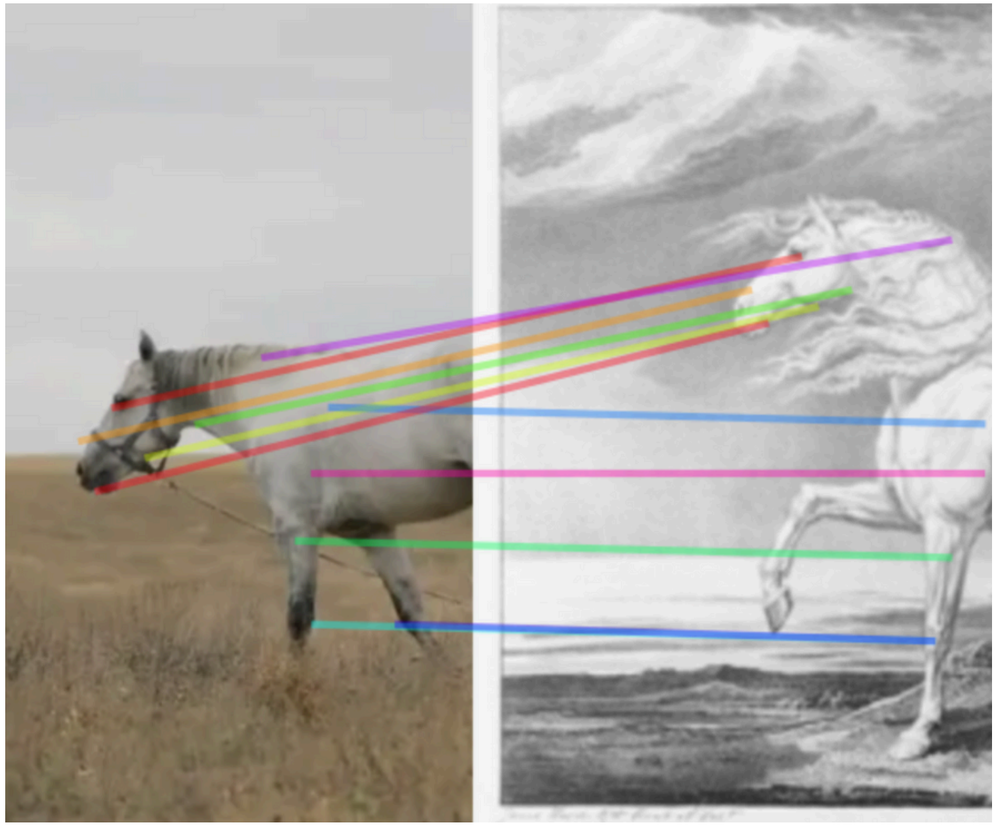
Figure 7: **Segmentation and depth estimation with linear classifiers.** Examples from ADE20K, NYUd, SUN RGB-D and KITTI with a linear probe on frozen OpenCLIP-G and DINOv2-g features.

5-5. Qualitative Results

Figure 8: **Examples of out-of-distribution examples** with frozen DINOv2-g features and a linear probe.

5-5. Qualitative Results

Dense Matching



Sparse Matching

6. Conclusion

Scalable self-supervised model that matches or outperforms weakly-supervised models.

Curated dataset (LVD-142M) improves generalization over traditional ImageNet pretraining.

Efficient training (FlashAttention, FSDP, knowledge distillation) enables scaling to billion-parameter models.

Thanks!

Q&A

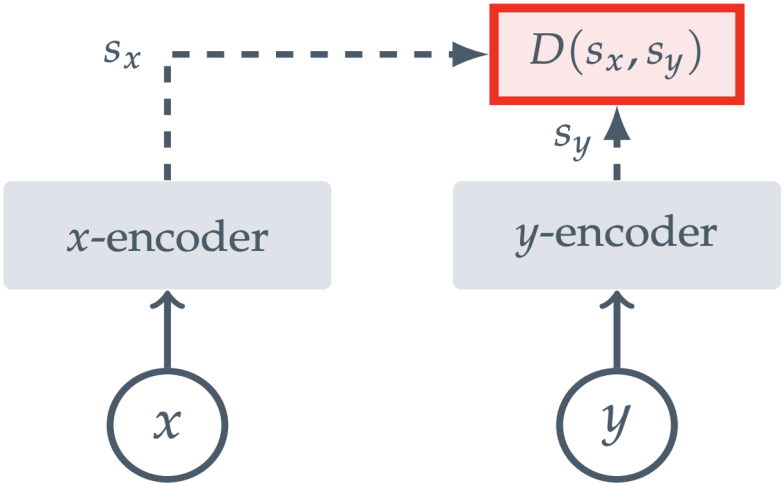
Self-Supervised Learning from Images with a Joint- Embedding Predictive Architecture (IJEPA)

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, Nicolas Ballas

Presentation by Surbhi

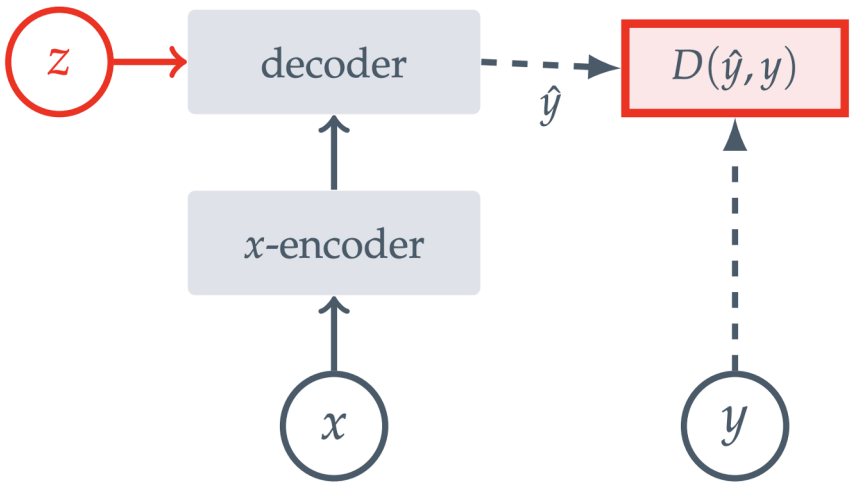
Approaches for Visual Representation Learning

01



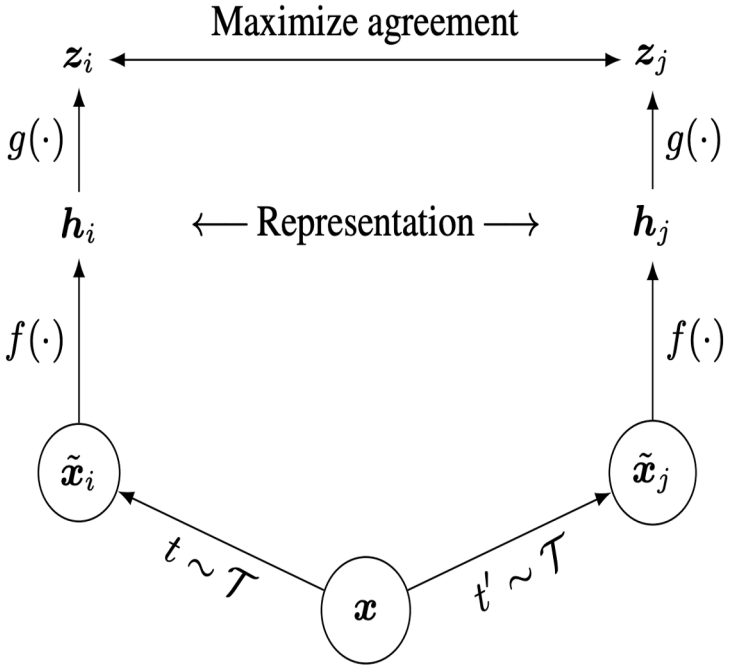
(a) **Joint-Embedding Architecture**

02



(b) **Generative Architecture**

Approach - Joint Embedding Architecture



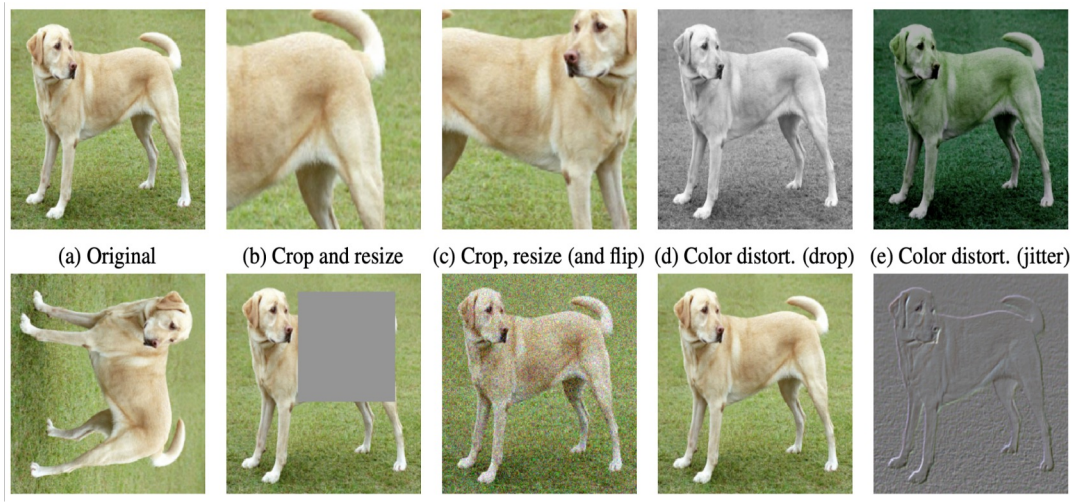
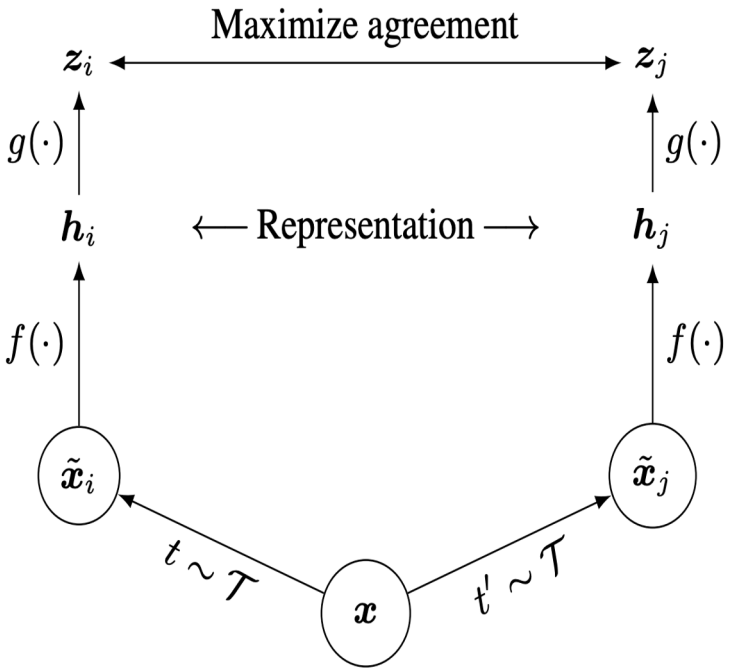


Image Credits: Chen et al., A Simple Framework for Contrastive Learning of Visual Representations, ICML 2020

Approach - Joint Embedding Architecture

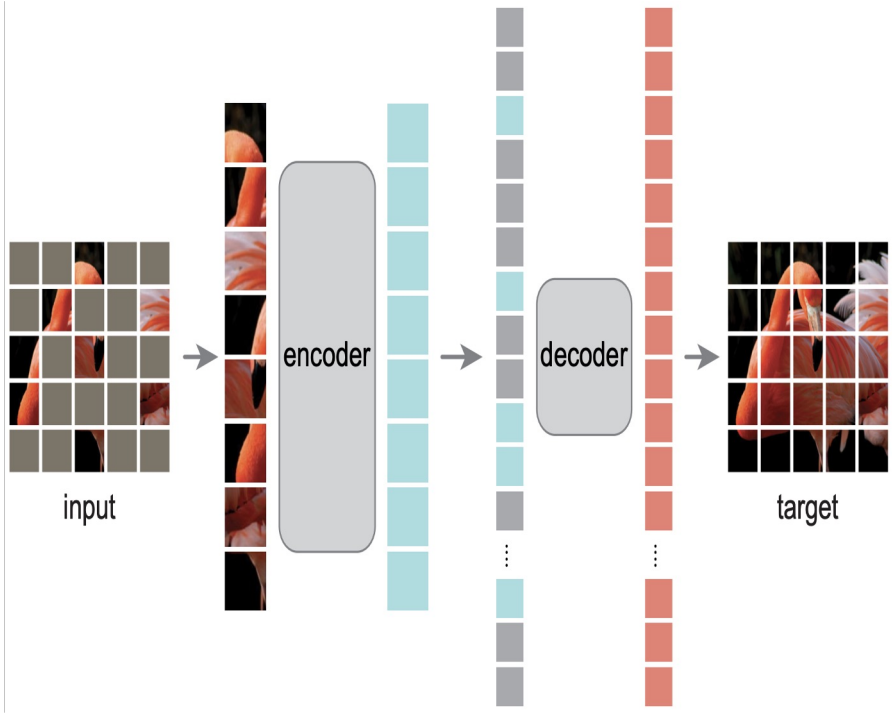


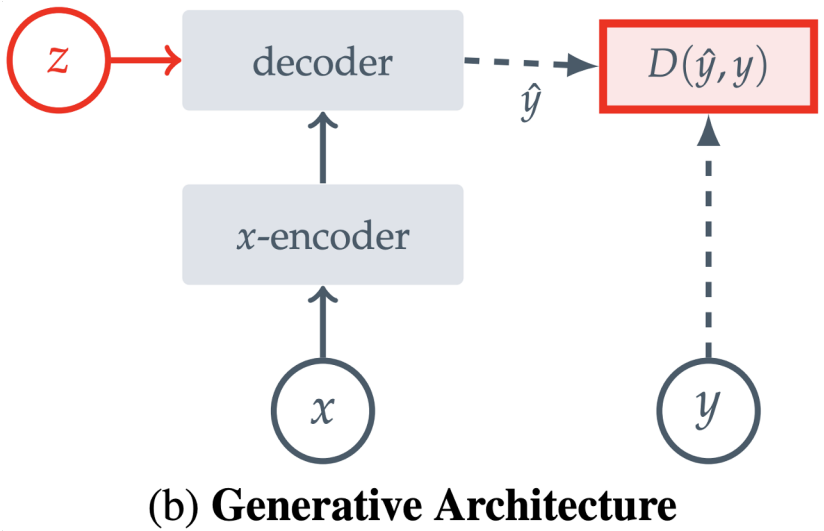
PROS

- Features of high semantic level
- Features excel off-the shelf (frozen eval.)

Image Credits: Chen et al., A Simple Framework for Contrastive Learning of Visual Representations, ICML 2020

Approach - Generative Architecture

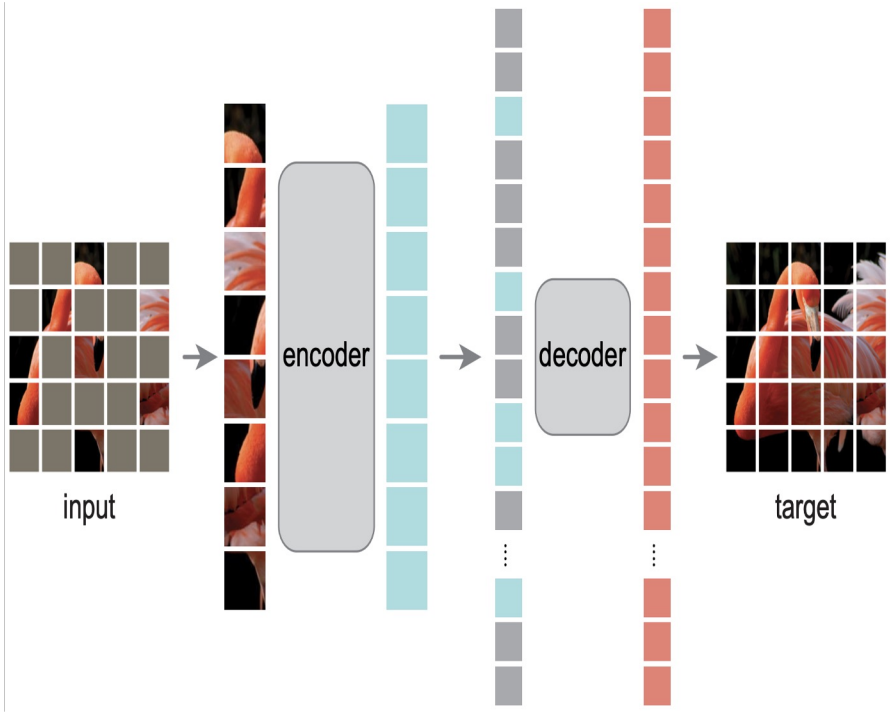




(b) **Generative Architecture**

Image Credits: He et al., Masked Autoencoders Are Scalable Vision Learners, arXiv 2021

Approach - Generative Architecture



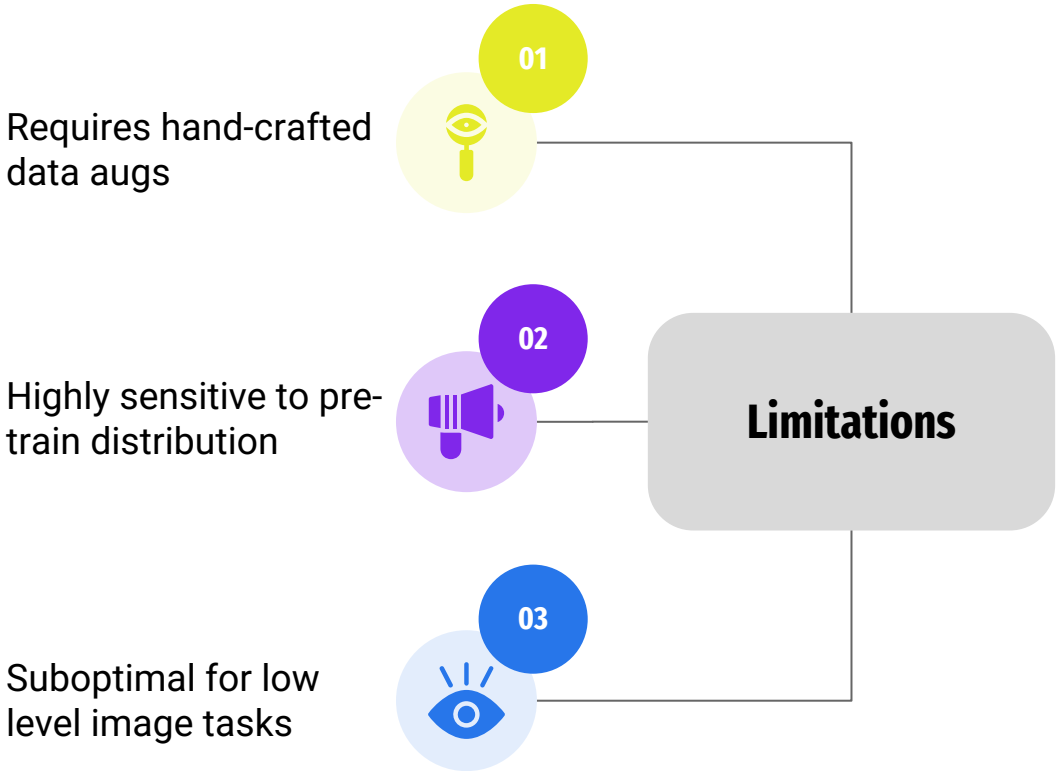
Features suitable for lower level image tasks

General Approach (no domain-specific hand-crafted data augmentations)

Less sensitive to pre-train distribution

Image Credits: He et al., Masked Autoencoders Are Scalable Vision Learners, arXiv 2021

Joint Embedding Architecture



Joint Embedding Architecture

Requires hand-crafted data augs

Highly sensitive to pre-train distribution

Suboptimal for low level image tasks

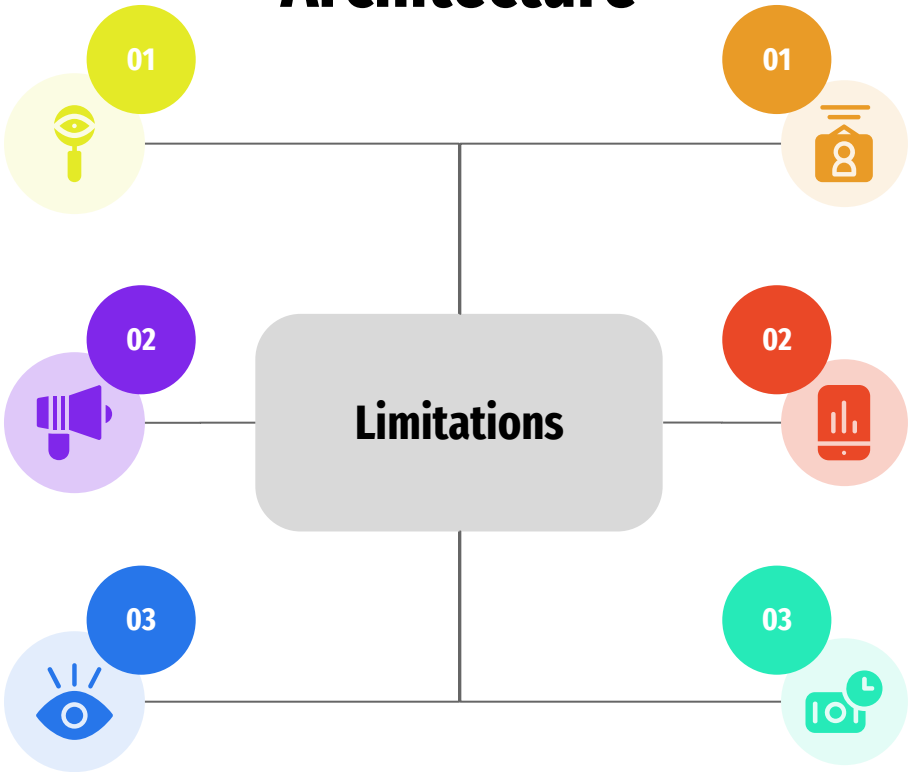
Generative Architecture

Features of relatively lower semantic level

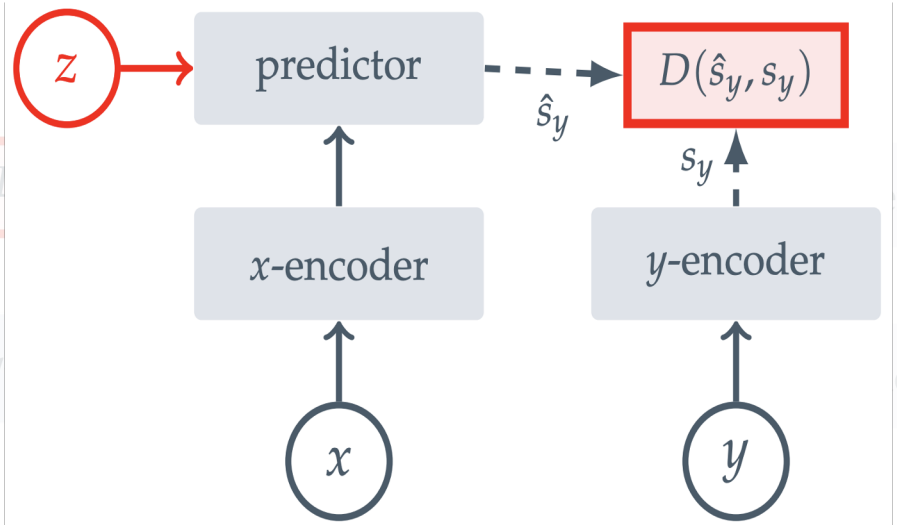
Require more involved adaptation mechanism

Pluto is considered a dwarf planet

Limitations



Approaches for Visual Representation Learning

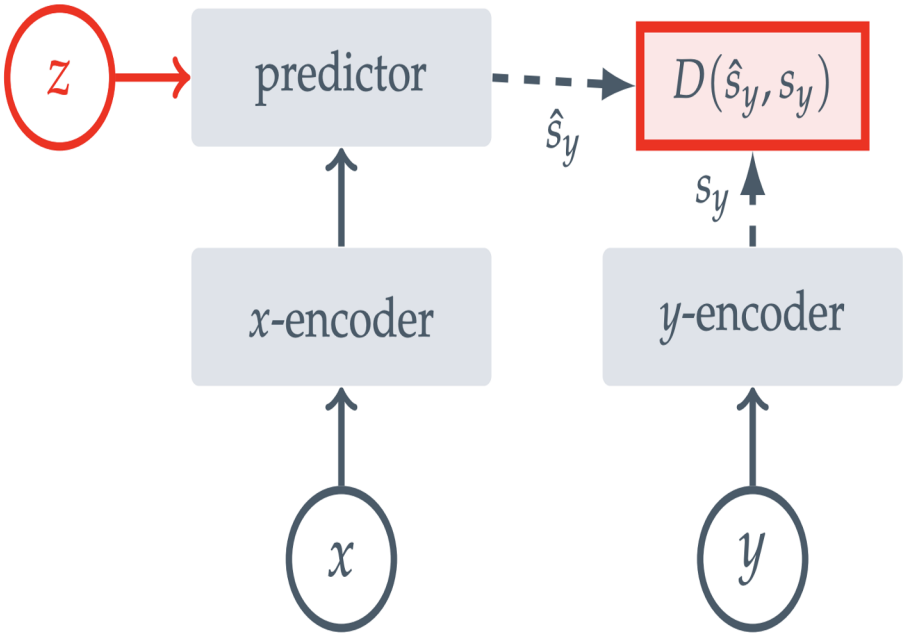


(c) **Joint-Embedding Predictive Architecture**

(a) **Joint-Embedding Architecture**

(b) **Generative Architecture**

Joint Embedding Predictive Architecture (JEPA)



Capture semantics without relying on hand-crafted data augs

Excel off-the-shelf (frozen eval.)

Perform well in both high-level and low-level image tasks

Image Joint Embedding Predictive Architecture (I-JEPA)

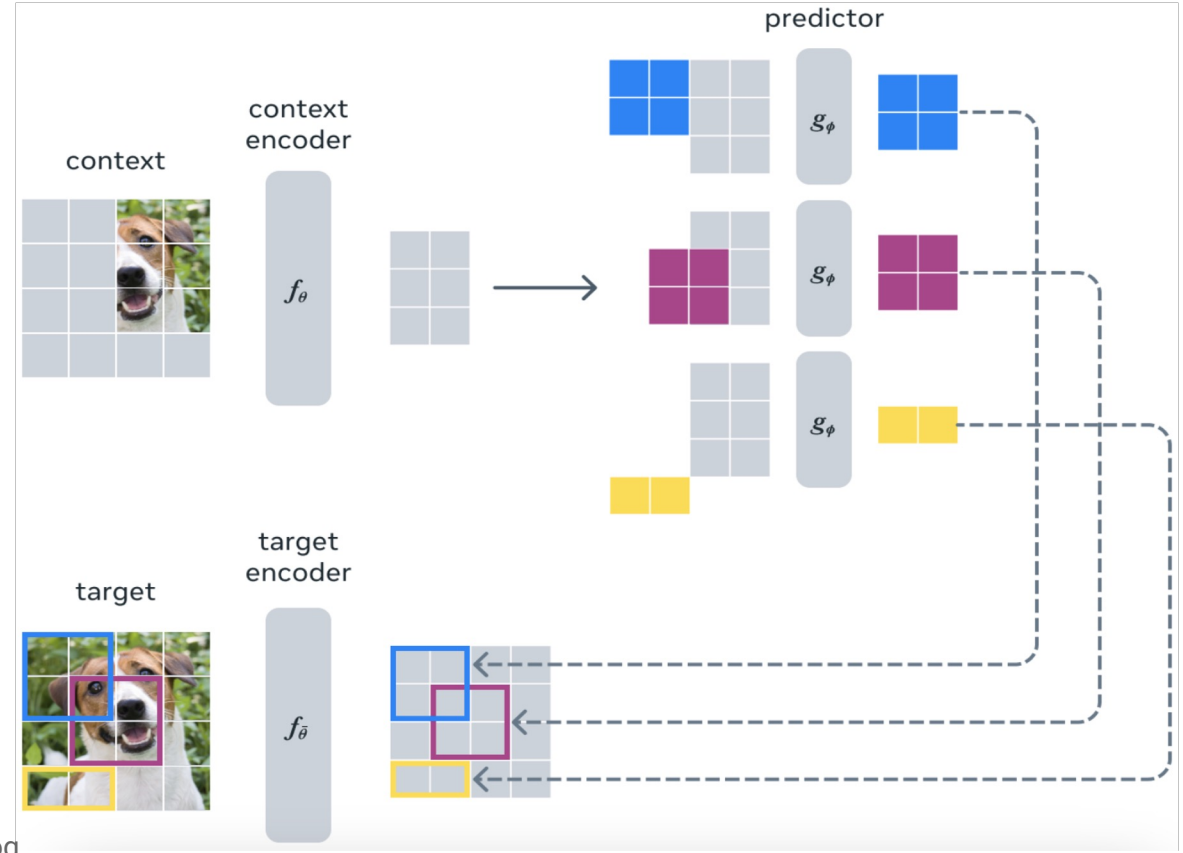


Image Credits: Meta AI Blog

Method

01 **Vision Transformer Architecture**

02 Context Encoder

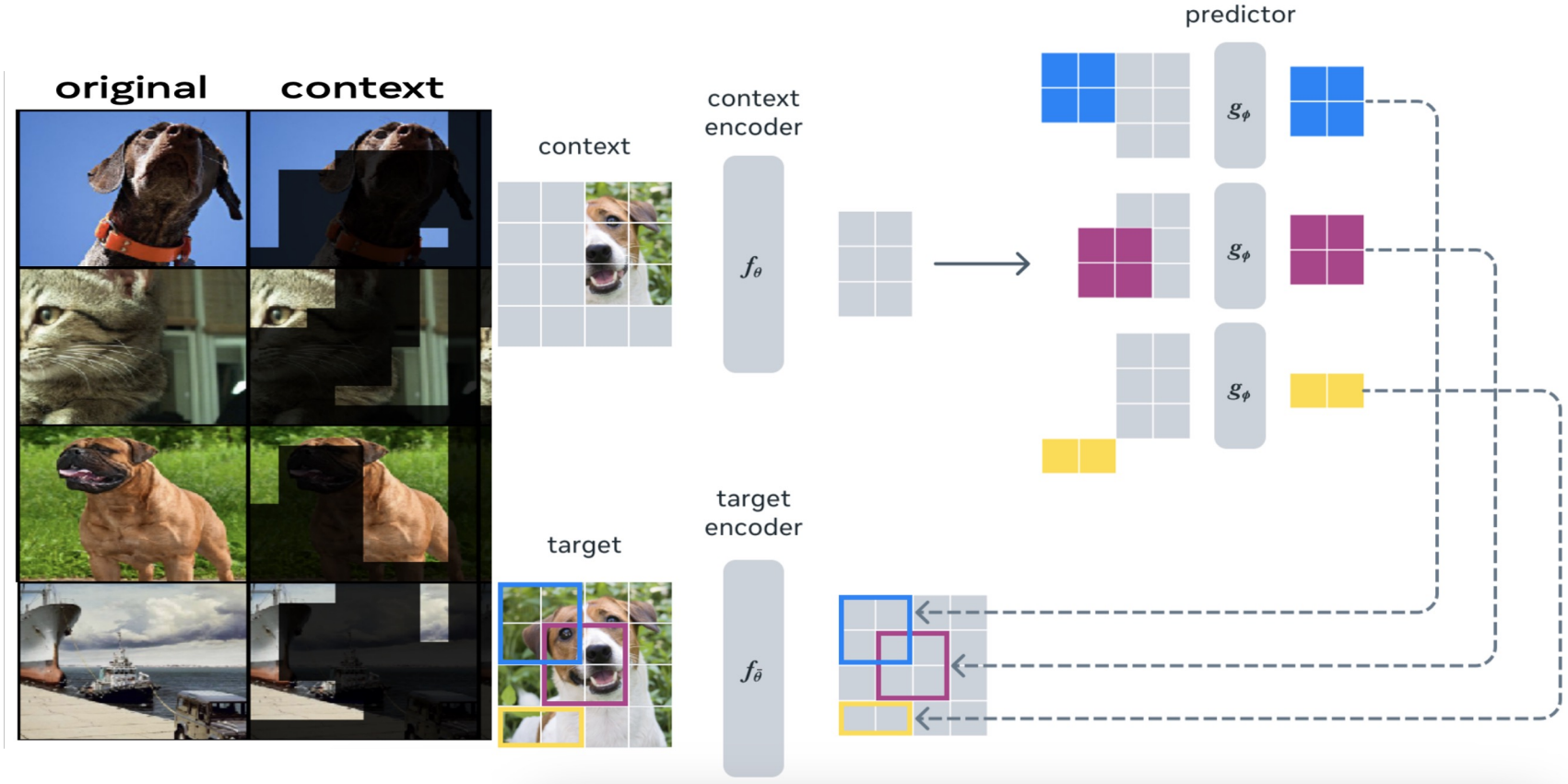
03 Target Encoder

04 Predictor

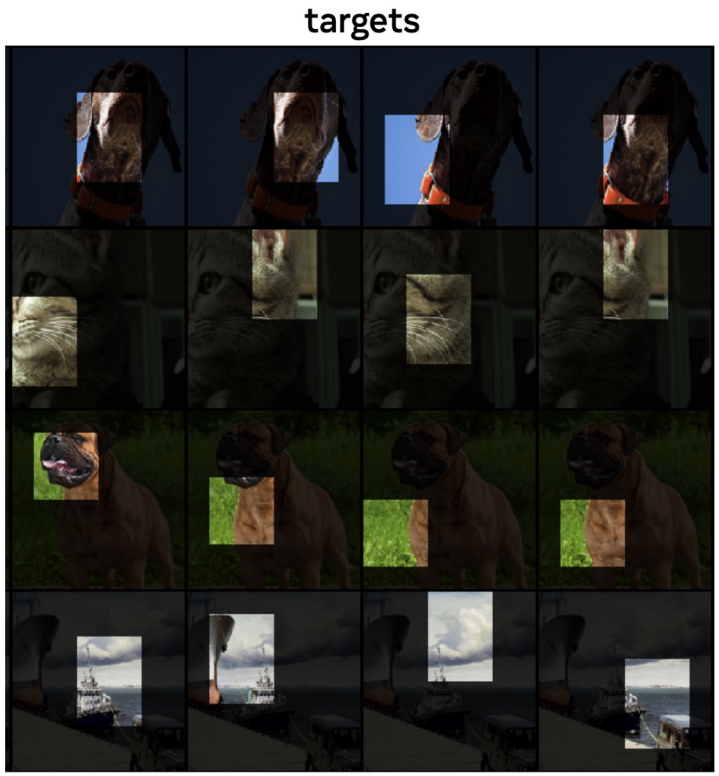
GIF Credits: Google AI Blog

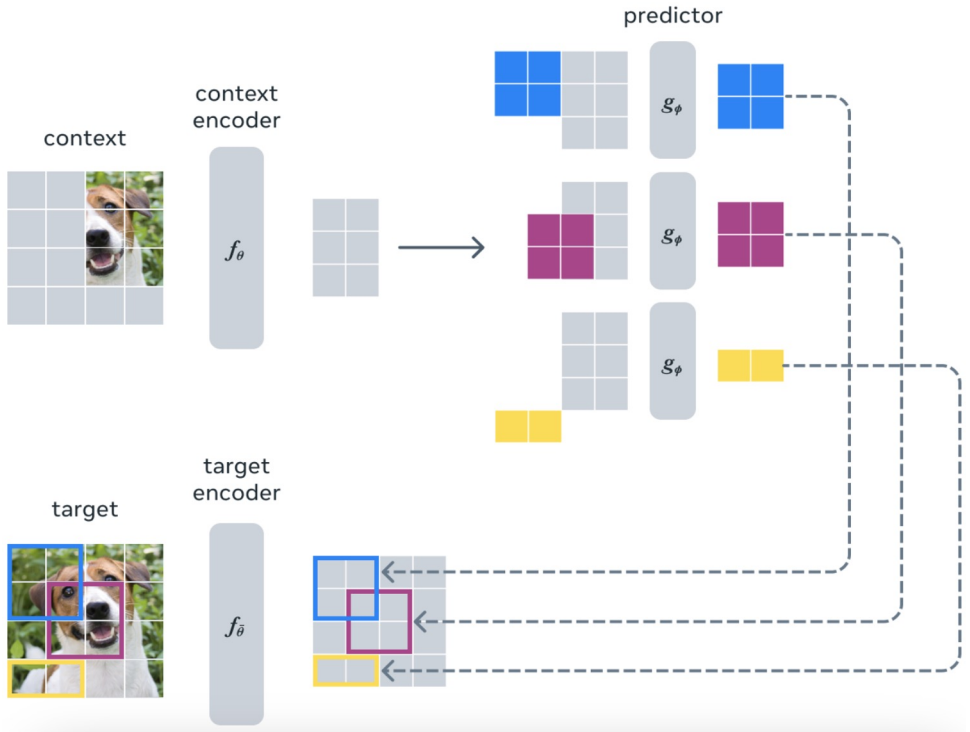
Context & Target Masking Strategy

Context Masking Strategy

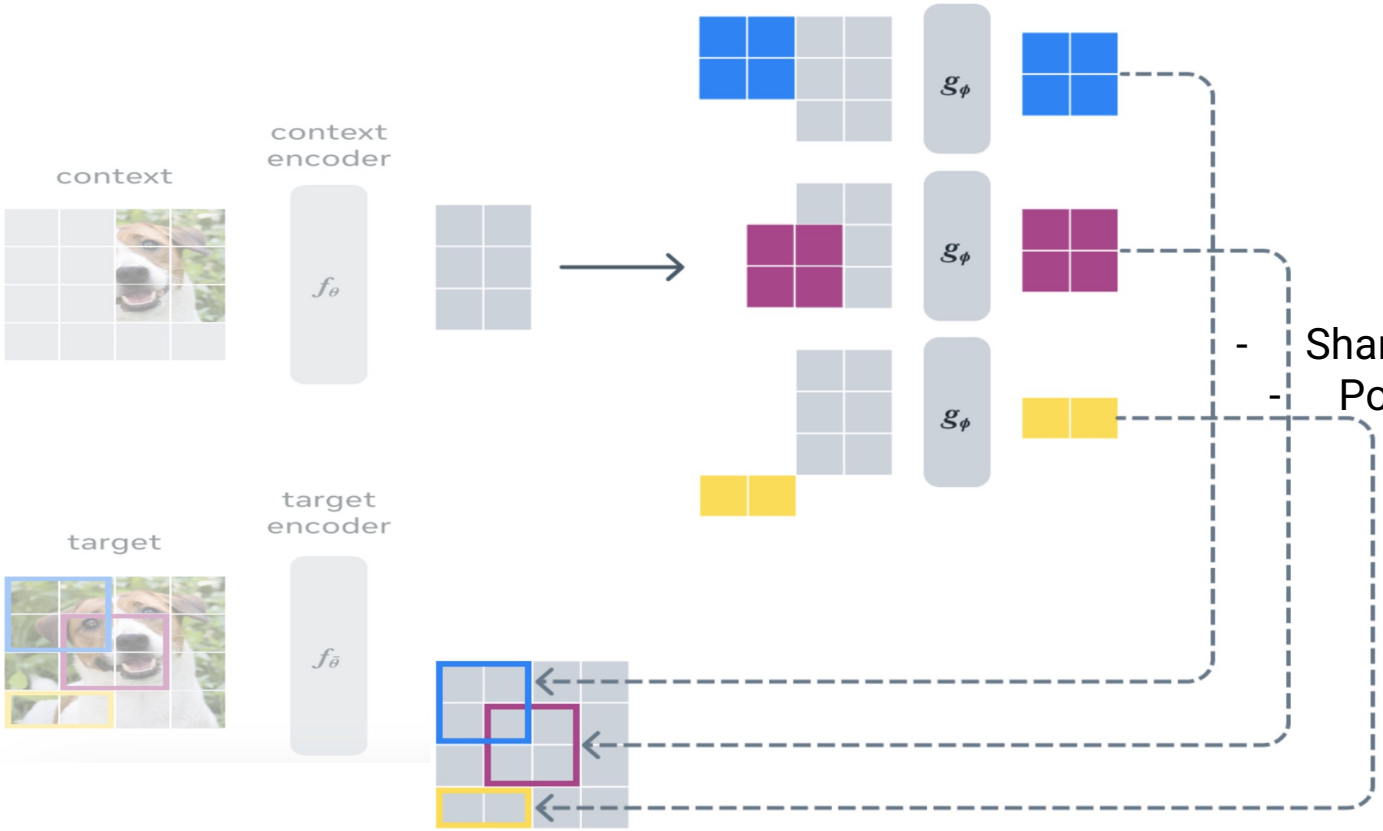


Target Masking Strategy





Predictor



Input:
Mask Tokens
- Shared learnable vector
- Positional Embedding

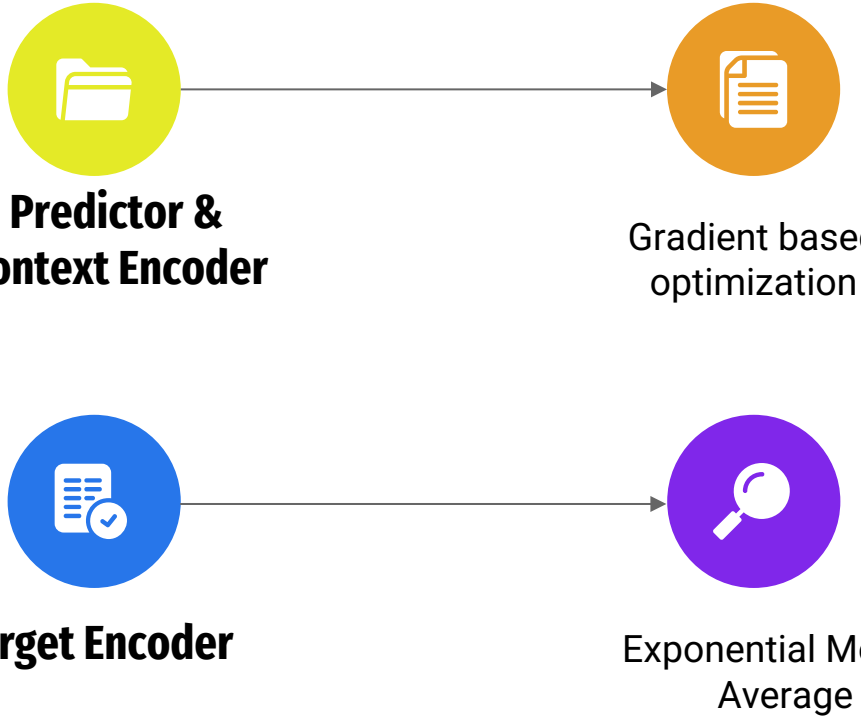
Loss

Average L2 Distance

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

Between predicted and target patch-level representations

Parameters


Predictor &
Context Encoder
Gradient based
optimization
Target Encoder
Exponential Moving
Average

Exponential Moving Average

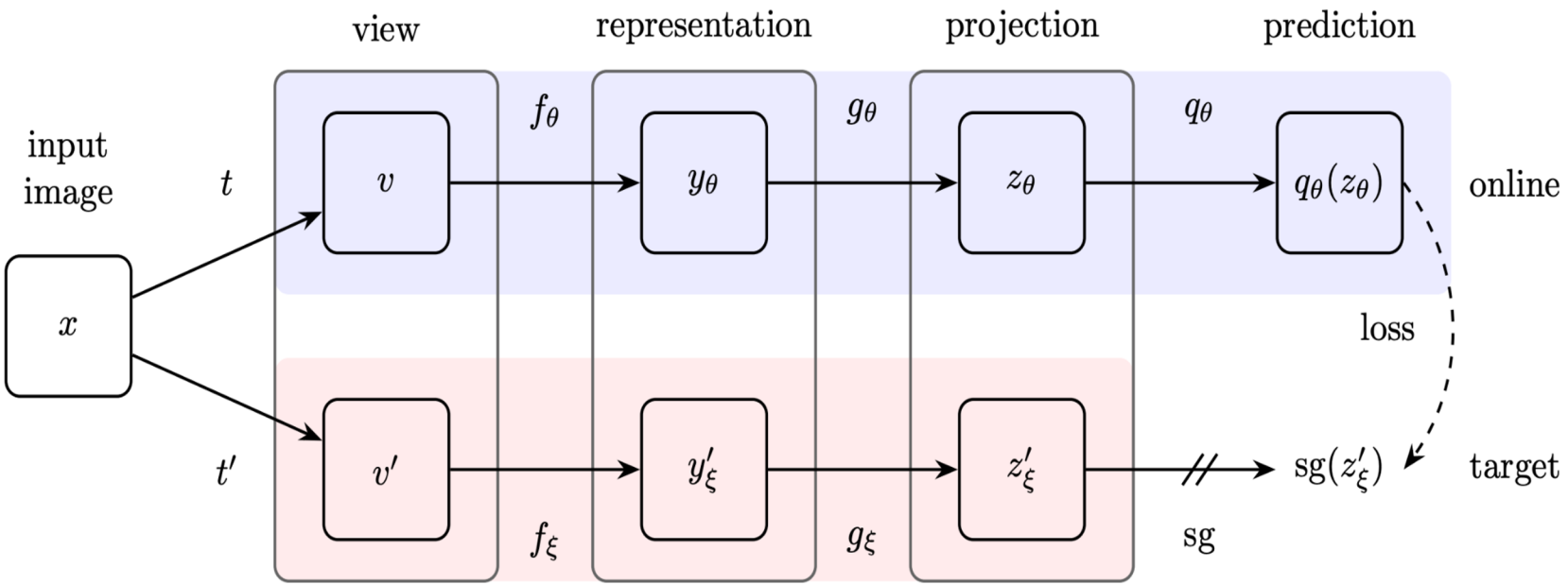


Photo Credits: Grill et al., Bootstrap your own latent space. A new approach to self supervised learning, 2020

Results (Image Classification Task)

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**

Table 1. **ImageNet**. Linear-evaluation on ImageNet-1k (the ViT-H/16_{448} is pretrained at at a resolution of 448 × 448). I-JEPA improves linear probing performance compared to other methods that do not rely on hand-crafted view data-augmentations during pretraining. Moreover, I-JEPA demonstrates good scalability — the larger I-JEPA model matches the performance of view-invariance approaches without requiring view data-augmentations.

Method	Arch.	Epochs	Top-1
Methods without view data augmentations			
data2vec [8]	ViT-L/16	1600	73.3
MAE [36]	ViT-L/16	1600	67.1
	ViT-H/14	1600	71.5
I-JEPA	ViT-L/16	600	69.4
	ViT-H/14	300	73.3
	ViT-H/16_{448}	300	**77.3**
Methods using extra view data augmentations			
iBOT [79]	ViT-B/16	400	69.7
DINO [18]	ViT-B/8	300	70.0
SimCLR v2 [35]	RN151 (2×)	800	70.2
BYOL [35]	RN200 (2×)	800	71.2
MSN [4]	ViT-B/4	300	**75.7**

Table 2. **ImageNet-1%**. Semi-supervised evaluation on ImageNet-1K using only 1% of the available labels. Models are adapted via fine-tuning or linear-probing, depending on whichever works best for each respective method. ViT-H/16_{448} is pretrained at at a resolution of 448 × 448. I-JEPA pretraining outperforms MAE which also does not rely on hand-crafted data-augmentations during pretraining. Moreover, I-JEPA benefits from scale. A ViT-H/16 trained at resolution 448 surpasses previous methods including methods that leverage extra hand-crafted data-augmentations.

Method	Arch.	CIFAR100	Places205	iNat18
Methods without view data augmentations				
data2vec [8]	ViT-L/16	81.6	54.6	28.1
MAE [36]	ViT-H/14	77.3	55.0	32.9
I-JEPA	ViT-H/14	**87.5**	**58.4**	**47.6**
Methods using extra view data augmentations				
DINO [18]	ViT-B/8	84.9	57.9	55.9
iBOT [79]	ViT-L/16	**88.3**	**60.4**	**57.3**

Table 3. **Linear-probe transfer for image classification**. Linear-evaluation on downstream image classification tasks. I-JEPA significantly outperforms previous methods that also do not use augmentations (MAE and data2vec), and decreases the gap with the best view-invariance-based methods that leverage hand-crafted data augmentations during pretraining.

Results (Local Prediction Tasks)

Method	Arch.	Clevr/Count	Clevr/Dist
Methods without view data augmentations			
data2vec [7]	ViT-L/16	85.3	71.3
MAE [35]	ViT-H/14	**90.5**	**72.4**
I-JEPA	ViT-H/14	86.7	**72.4**
Methods using extra data augmentations			
DINO [17]	ViT-B/8	86.6	53.4
iBOT [75]	ViT-L/16	85.7	62.8

Table 4. **Linear-probe transfer for low-level tasks**. Linear-evaluation on downstream low-level tasks consisting of object counting (Clevr/Count) and depth prediction (Clevr/Dist). The I-JEPA method effectively captures low-level image features during pretraining and outperforms view-invariance based methods on tasks such object counting and depth prediction.

Performance on lower-level image tasks

I-JEPA outperforms joint embedding methods in low-level image tasks

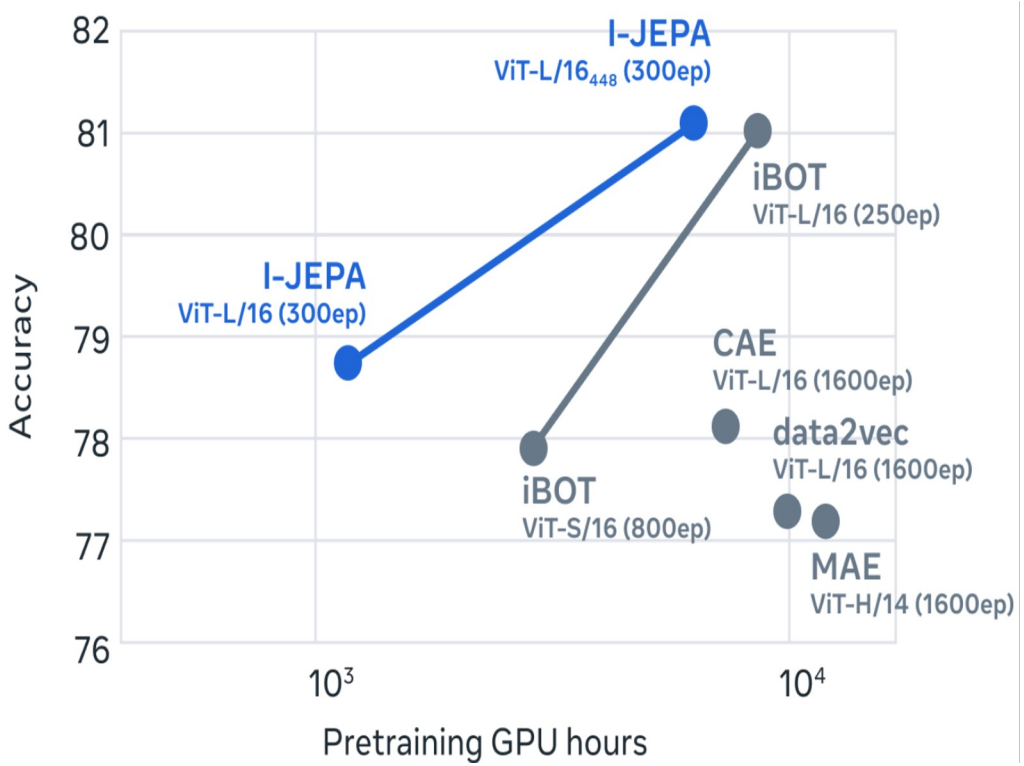
Results - Scalability

01

Very Efficient

02

Faster than iBOT



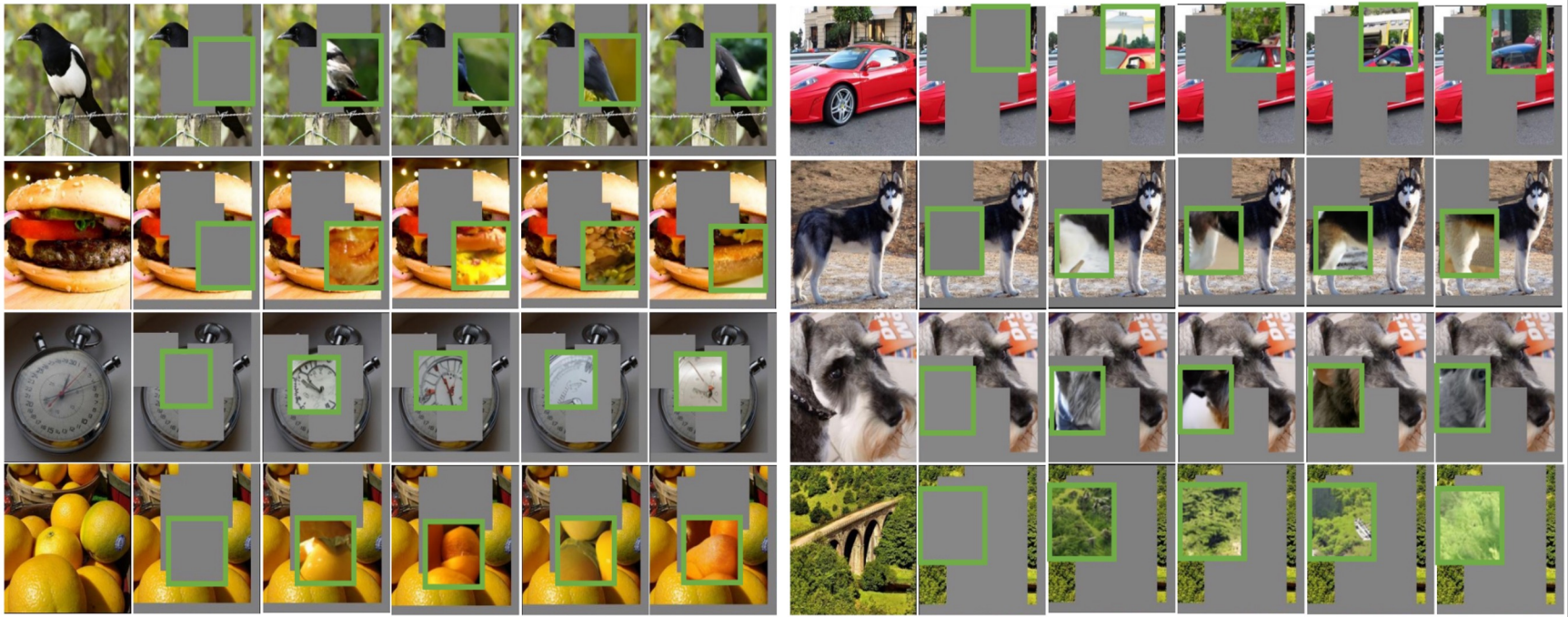
03

Converges Faster

04

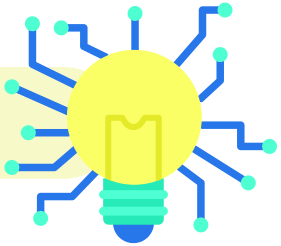
Scalable

Predictor Visualizations



Ablations

I-JEPA is non-generative



Targets	Arch.	Epochs	Top-1
Target-Encoder Output	ViT-L/16	500	**66.9**
Pixels	ViT-L/16	800	40.7

Table 7. **Ablating targets**. Linear evaluation on ImageNet-1K using only 1% of the available labels. The semantic level of the I-JEPA representations degrades significantly when the loss is applied in pixel space, rather than representation space, highlighting the importance of the target-encoder during pretraining.

Same method in pixel space performs much worse on semantic classification tasks.

Conclusion

Decoding I-JEPA predictor outputs to sketches

I-JEPA predictor captures both global semantics and spatial uncertainty.

Image Credits: Meta AI Blog

Further Work

V-JEPA	Extension of I-JEPA to videos
MC-JEPA	Motion Content JEPA: Extension of JEPA to include motion information
H-JEPA	Hierarchy JEPA, enhance JEPA's extraction capability
C-JEPA	Connecting JEPA with contrastive SSL

Thank You!

Happy Spring Break!

DS-GA.3001 Embodied Learning and Vision Topic Presentation

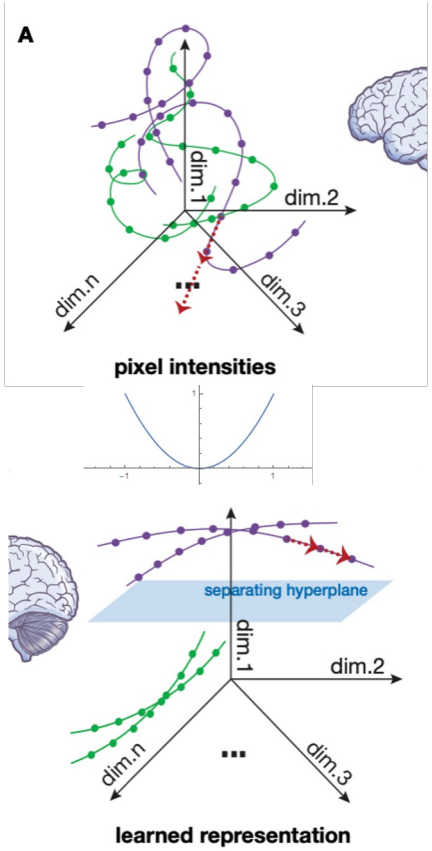
Niu et al., *"Learning predictable and robust neural representations by straightening image sequences"*, 2024.

Sal Yeung

NYU

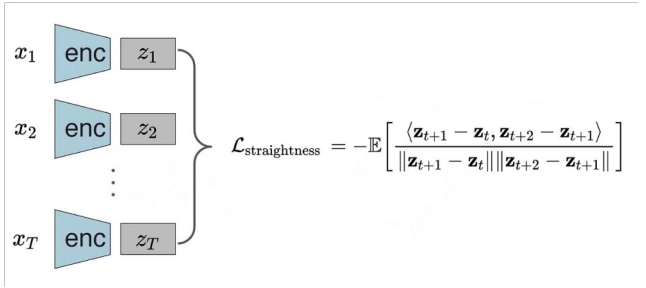
Abstract

Prediction is a fundamental capability of all living organisms, and has been proposed as an objective for learning sensory representations. Recent work demonstrates that in primate visual systems, prediction is facilitated by neural representations that follow straighter temporal trajectories than their initial photoreceptor encoding, which allows for prediction by linear extrapolation. Inspired by these experimental findings, we develop a self-supervised learning (SSL) objective that explicitly quantifies and promotes straightening. We demonstrate the power of this objective in training deep feedforward neural networks on smoothly-rendered synthetic image sequences that mimic commonly-occurring properties of natural videos. The learned model contains neural embeddings that are predictive, but also factorize the geometric, photometric, and semantic attributes of objects. The representations also prove more robust to noise and adversarial attacks compared to previous SSL methods that optimize for invariance to random augmentations. Moreover, these beneficial properties can be transferred to other training procedures by using the straightening objective as a regularizer, suggesting a broader utility of straightening as a principle for robust unsupervised learning.



NYU

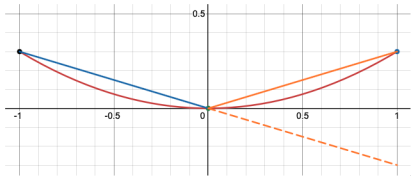
Introduction



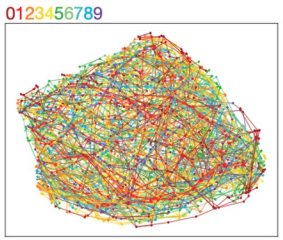
"Straightness": average cosine similarity between 2 successive difference vectors of any 3 temporally adjacent points

Contributions:

1. SSL objective + *whitening* to prevent representation collapse
2. Downstream predictions on visual attributes
3. Class separability
4. Robustness to multi-view invariance



"Whitening": transformation of random vector to be uncorrelated [1]



NYU

[1] Bardes et al., "*VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*", ICLR, 2022

Background

"Pixel intensity": dissimilarity between difference(t) of successive frames t ∈ T

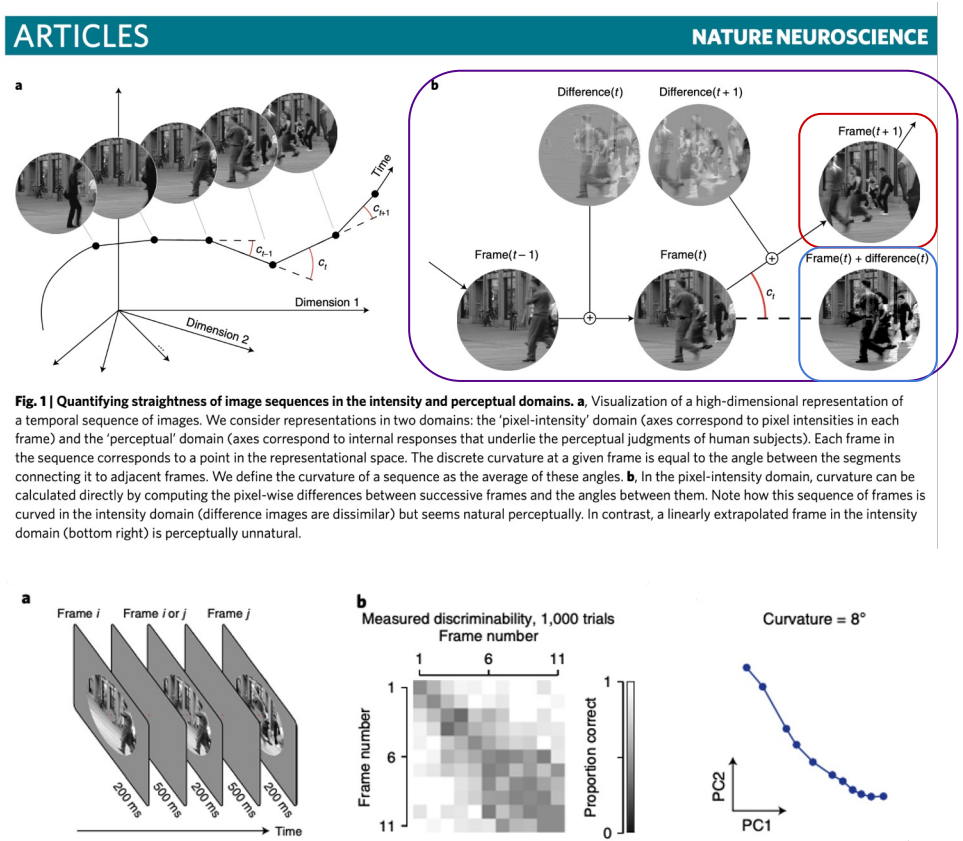
"Perceptual intensity"

1. Neural network features
2. "Psychophysical AXB task", identify outlier given set of images (A, A or B, B)

Hypotheses

1. natural sequences that are highly curved in pixel space are straighter perceptually
2. artificial sequences that are straight in pixel space are more curved perceptually

$$f_{t+1} = f_t + (f_t - f_{t-1})$$



Hénaff et al., *"Perceptual straightening of natural videos"*, Nature Neuroscience, 2019

Background

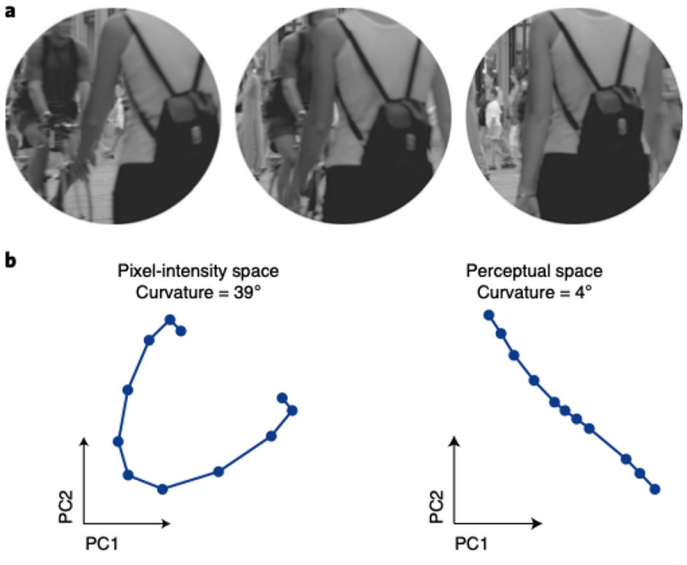


Fig. 3 | Curvature reduction for natural image sequences.

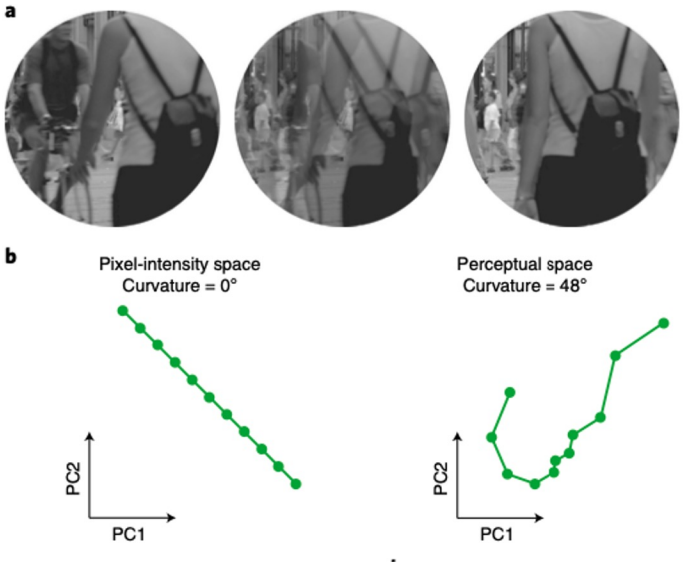


Fig. 4 | Curvature increase for artificial image sequences.

NYU

Hénaff et al., *"Perceptual straightening of natural videos"*, Nature Neuroscience, 2019

Background

Biologically-inspired transformer model

1. Center-surround filtering, local luminance and contrast gain control operations

1. Oriented filters, squared and combined responses over phase

Curvature of model features

- Biology network, more “straight curvatures” on natural videos

- Neural network, only seen in output layer

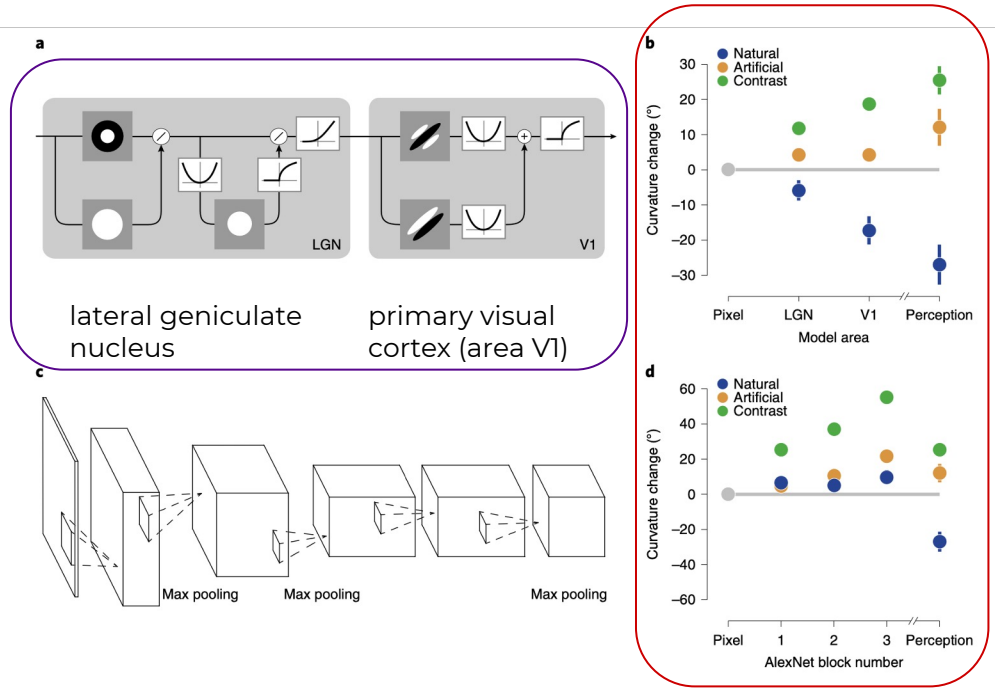


Fig. 6 | Changes in curvature induced by models of the visual system. a, Two-stage cascade model describing computations found in the retina, lateral

Hénaff et al., *“Perceptual straightening of natural videos”*, Nature Neuroscience, 2019

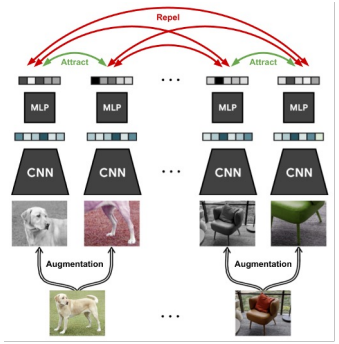
Related

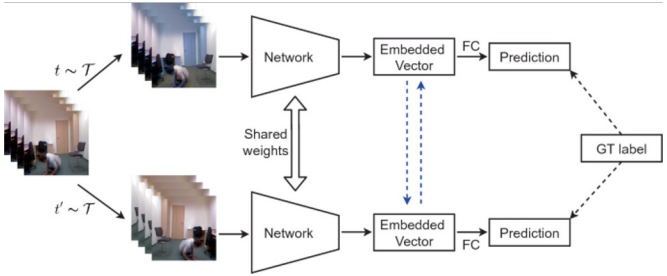
1. Temporal Invariance

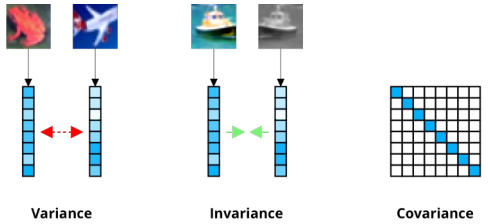
Existing SSL methods
- Contrastive
- Self distillation
- Correlation analysis (whitening)

Past
- Operates on static images, lost time varying features
  van Steenkiste et al., “Moving Off-the-Grid: Scene-Grounded Video Representations”, 2024

Current
- Straightening captures all features in spatio temporal inputs
- Predict future states







Chen et al., *“A Simple Framework for Contrastive Learning of Visual Representations”*, ICML, 2020
Grill et al., “*Bootstrap your own latent: A new approach to self-supervised Learning*”, NeurIPS, 2020
Bardes et al., “*VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*”, ICLR, 2022

Related

2. Temporal Prediction

Past
- Contrastive prediction encoding
  - parametrization scales quadratically with feature dimension
- Linearized representation
  - auxiliary architectural components and loss

Current
- parameter-free
- predictions can adapt to different contexts

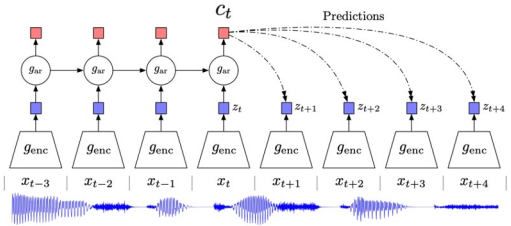


Figure 1: Overview of Contrastive Predictive Coding, the proposed representation learning approach. Although this figure shows audio as input, we use the same setup for images, text and reinforcement learning.

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

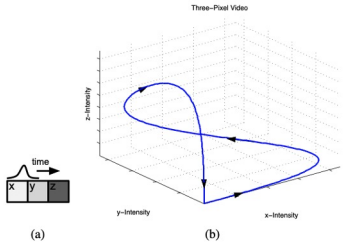


Figure 1: (a) A video generated by translating a Gaussian intensity bump over a three pixel array (x,y,z), (b) the corresponding manifold parametrized by time in three dimensional space

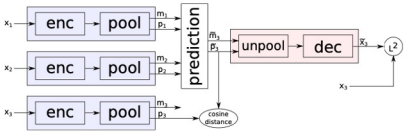


Figure 2: The basic linear prediction architecture with shared weight encoders

Van den Oord et a., "*Representation learning with contrastive predictive coding*", arXiv, 2018
Henaff et al., "*Data-efficient image recognition with contrastive predictive coding. In: Inter- national conference on machine learning*", PMLR, 2020

Related

3. Straightening and robustness

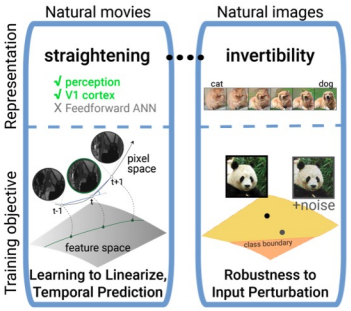
Past

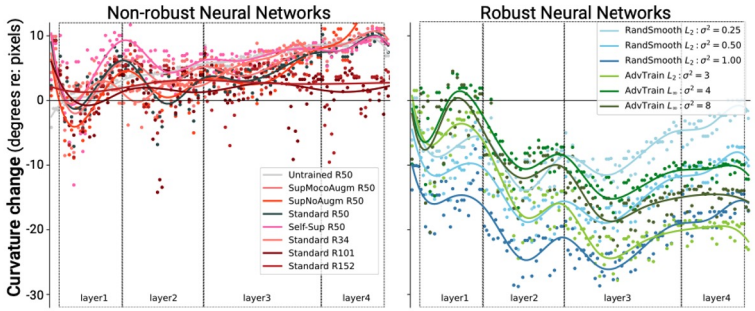
- "Straightness" found in human, macaque physiology, but not in neural networks

- Tolerance to noise or perturbations => straightened responses [1, 2]

Current

- Reverse: straightened responses => tolerance to noise or perturbations





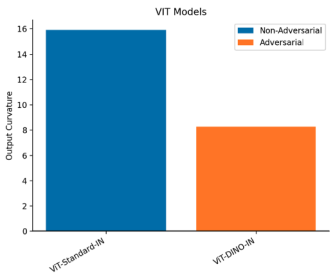
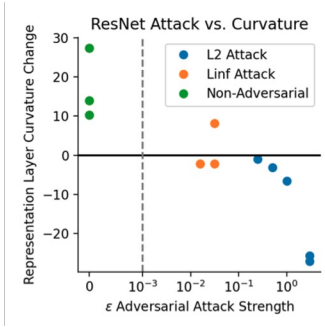
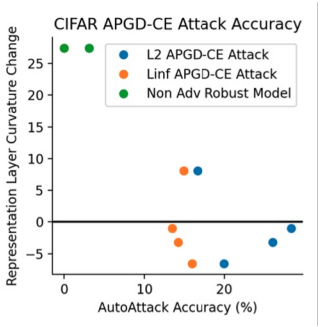


Figure 15: ViT models trained on ImageNet





NYU

[1] Toosi et al., "*Brain-like representational straightening of natural movies in robust feedforward neural networks*", ICLR, 2023
[2] Harrington et al., "*Exploring perceptual straightness in learned visual representations*", ICLR, 2023

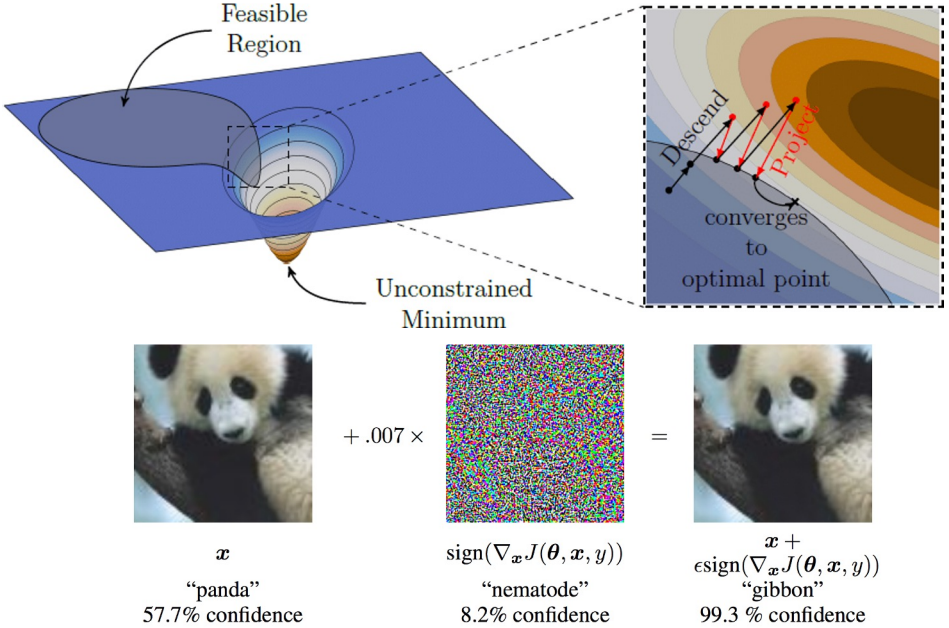
Related

3. Straightening and robustness

Past
- "Straightness" found in human, macaque physiology, but not in neural networks

- Tolerance to noise or perturbations => straightened responses [1, 2]

Current
- Reverse: straightened responses => tolerance to noise or perturbations



Madry et al., "*Towards Deep Learning Models Resistant to Adversarial Attacks*", ICLR, 2018
Goodfellow et al., "*Explaining and Harnessing Adversarial Examples*", ICLR, 2015

Model

Straightness loss

- Applied to output layer // can be any (or several) layer(s) of network
- Straightness alone can have trivial solution

VICReg

- Variance, prevent different inputs from collapsing to same output
- Covariance, decorrelates pairs of output dimensions to minimize redundancies

$$\mathcal{L}_{\text{straightness}} = -\mathbb{E}\left[\frac{\langle \mathbf{z}_{t+1} - \mathbf{z}_t, \mathbf{z}_{t+2} - \mathbf{z}_{t+1} \rangle}{\|\mathbf{z}_{t+1} - \mathbf{z}_t\| \|\mathbf{z}_{t+2} - \mathbf{z}_{t+1}\|}\right]$$

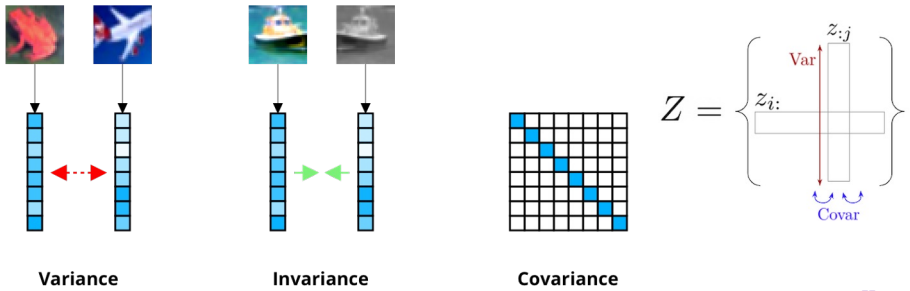
$$\mathcal{L}_{\text{variance}} = \mathbb{E}\left[\frac{1}{d}\sum_{i=1}^{d} \max\left(0, 1 - S\left(z_t^i, \epsilon\right)\right)\right]$$

$$S(x, \epsilon) = \sqrt{\text{Var}(x) + \epsilon}$$

d is the output dimensionality.

$$\mathcal{L}_{\text{covariance}} = \mathbb{E}\left[\frac{1}{d}\sum_{i \neq j} [\text{Cov}(\mathbf{z}_t)]_{i,j}^2\right]$$

$$\mathcal{L} = \mathcal{L}_{\text{straightness}} + \alpha \mathcal{L}_{\text{variance}} + \beta \mathcal{L}_{\text{covariance}}$$



Bardes et al., "*VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*", ICLR, 2022

Model

Straightness loss

- Applied to output layer // can be any (or several) layer(s) of network
- Straightness alone can have trivial solution

VICReg

- Variance, prevent different inputs from collapsing to same output
- Covariance, decorrelates pairs of output dimensions to minimize redundancies

$$\mathcal{L}_{\text{straightness}} = -\mathbb{E}\left[\frac{\langle \mathbf{z}_{t+1} - \mathbf{z}_t, \mathbf{z}_{t+2} - \mathbf{z}_{t+1} \rangle}{\|\mathbf{z}_{t+1} - \mathbf{z}_t\| \|\mathbf{z}_{t+2} - \mathbf{z}_{t+1}\|}\right]$$

$$\mathcal{L}_{\text{variance}} = \mathbb{E}\left[\frac{1}{d}\sum_{i=1}^{d} \max\left(0, 1 - S\left(z_t^i, \epsilon\right)\right)\right]$$

$$S(x, \epsilon) = \sqrt{\text{Var}(x) + \epsilon}$$

d is the output dimensionality.

$$\mathcal{L}_{\text{covariance}} = \mathbb{E}\left[\frac{1}{d}\sum_{i \neq j} [\text{Cov}(\mathbf{z}_t)]_{i,j}^2\right]$$

$$\mathcal{L} = \mathcal{L}_{\text{straightness}} + \alpha \mathcal{L}_{\text{variance}} + \beta \mathcal{L}_{\text{covariance}}$$

For control experiments

$$L = L_{\text{invariance}} + \lambda L_{\text{variance}} + \gamma L_{\text{covariance}}$$

$$L_{\text{invariance}} = \mathbb{E}\left[\frac{1}{d}\|\mathbf{z}_t - \mathbf{z}_{t_0}\|_F^2\right]$$

NYU

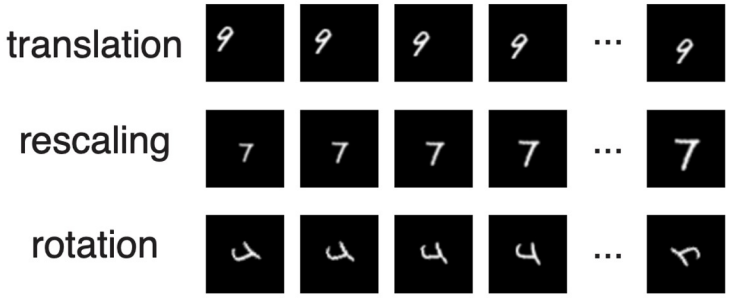
Bardes et al., "*VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*", ICLR, 2022

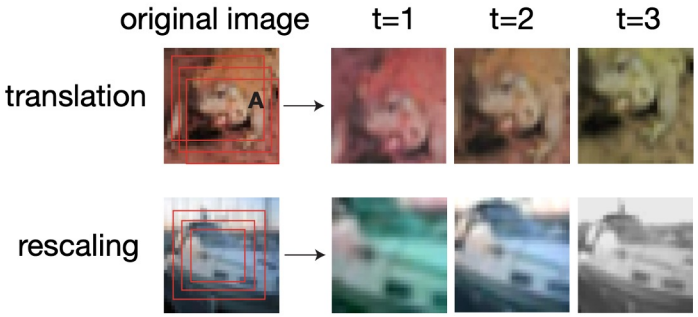
Data

Artificial videos
- temporally structured augmentations
- mimic natural transformations to static images

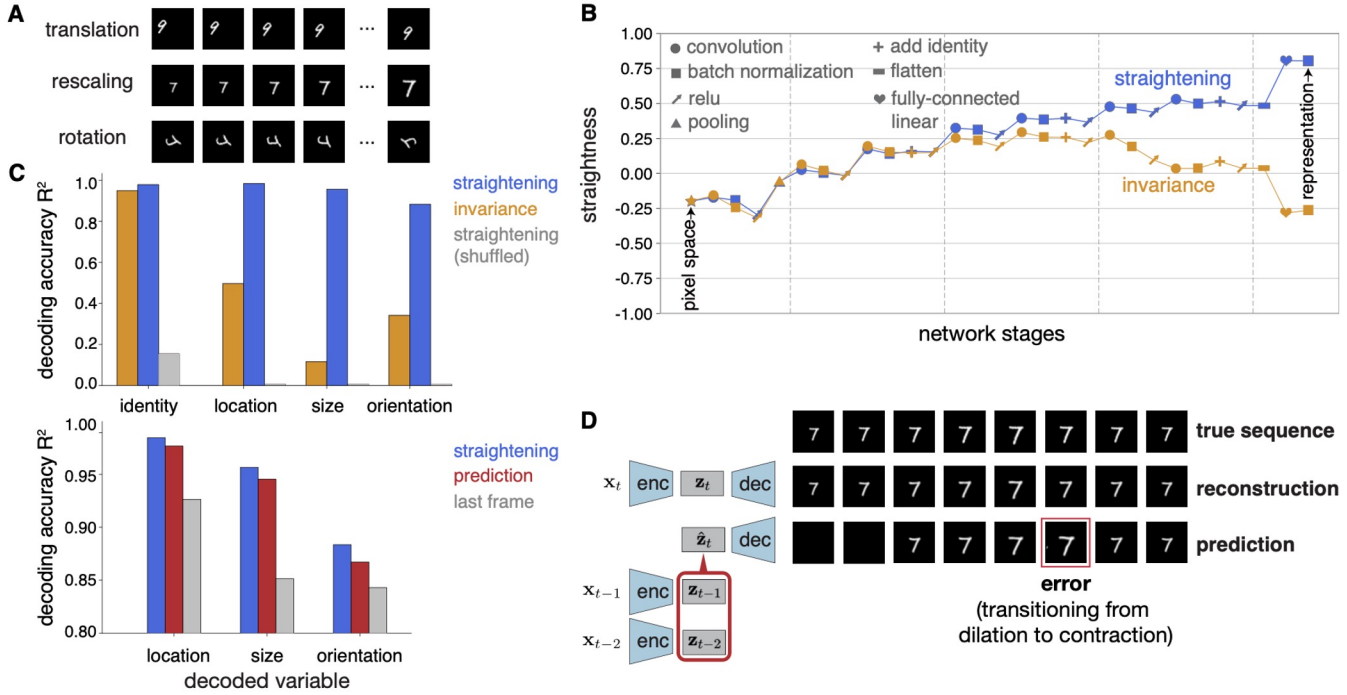
Reasons of not using natural videos
- Match image based SSL models
- Bad performance on recognition tasks due to insufficient object variety

Approach
- MNIST
  - Translation, rescaling, rotation
    Brightness, contrast, saturation, hue
- CIFAR
  - Translation, rescaling
  - Horizontal flips (all frames)
    Grayscale, solarization (per frame)

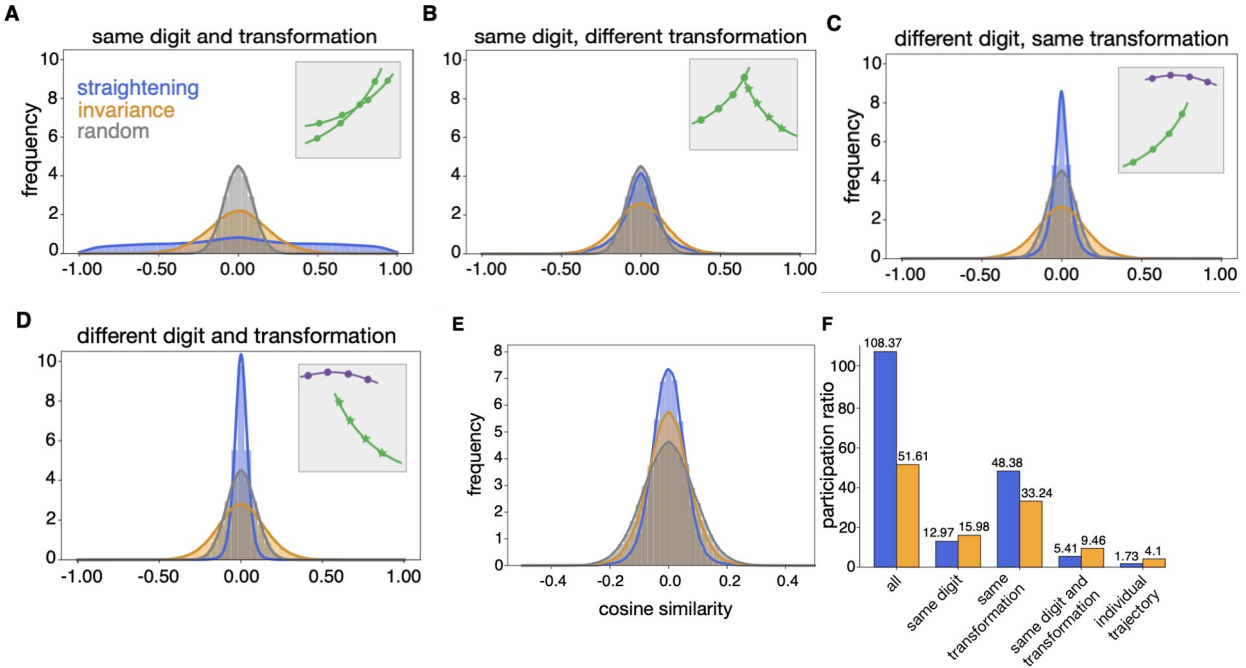




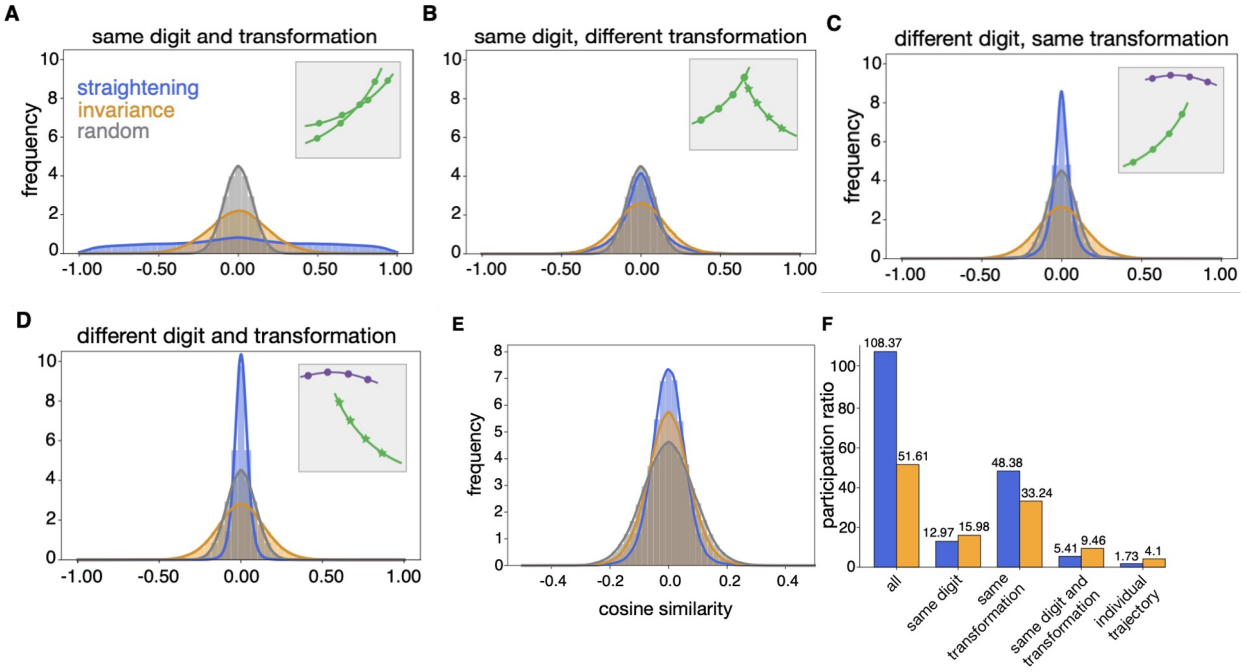
Experiments



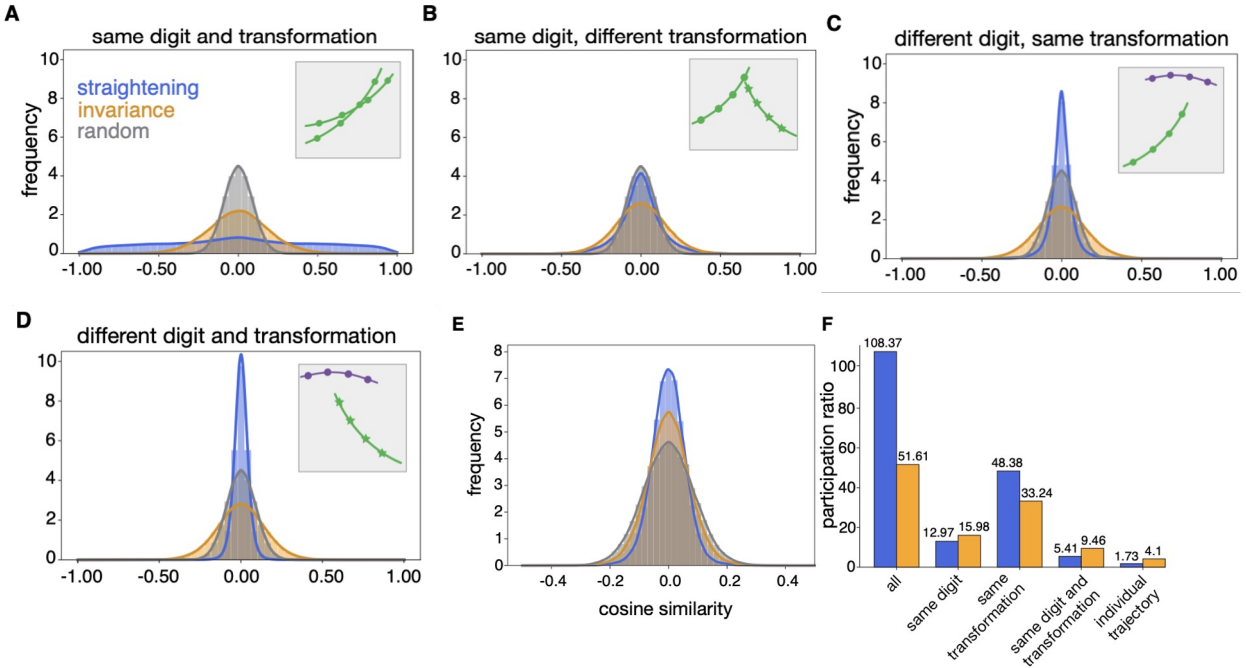
Experiments



Experiments

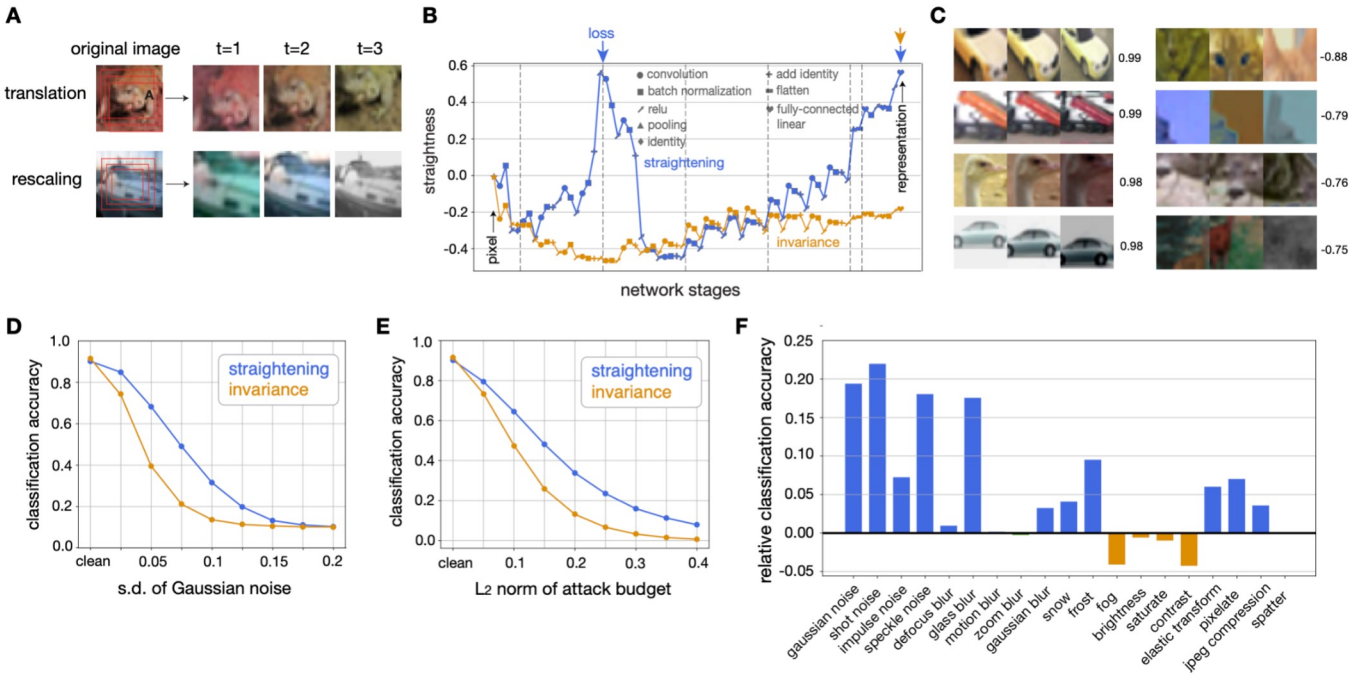


Experiments

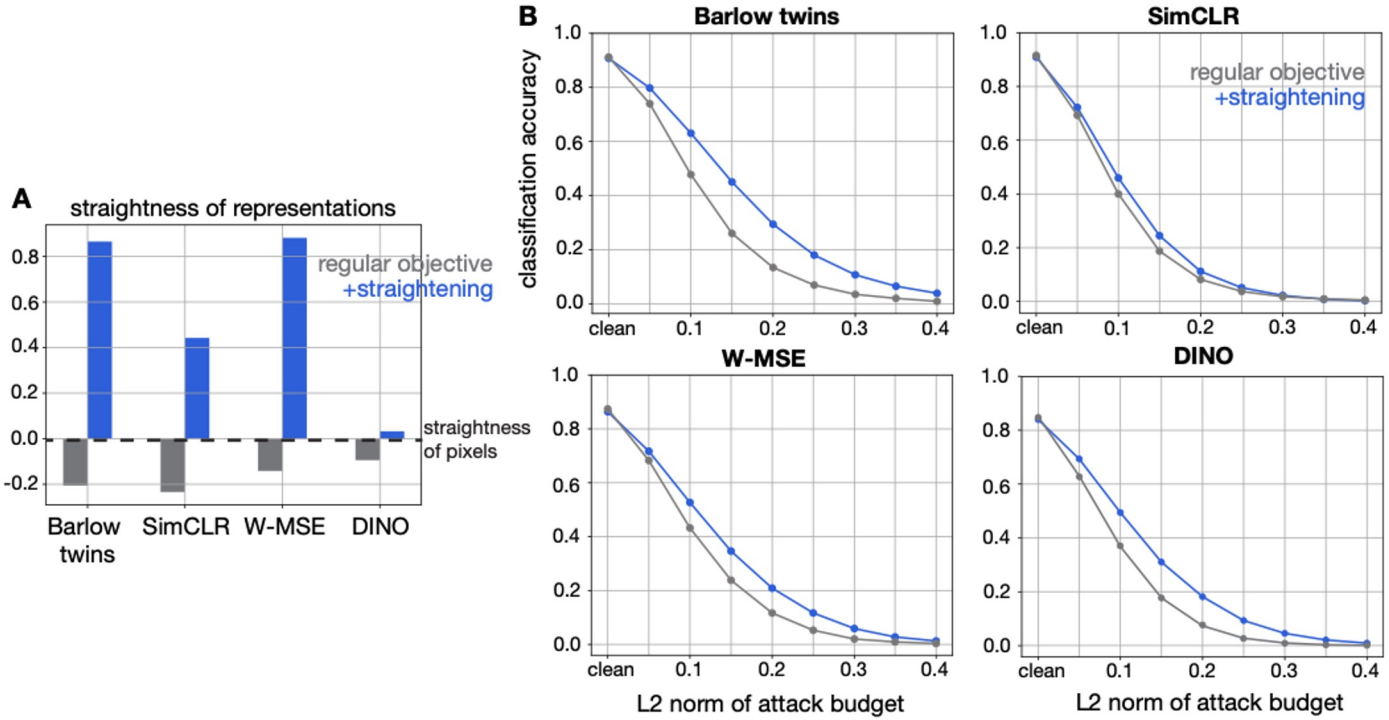


$R = \frac{(\sum_i \lambda_i)^2}{\sum_i \lambda_i^2}$, where $\{\lambda_i\}$ are the eigenvalues of the covariance matrices

Experiments



Experiments



Conclusion

Idea

- Biologically-inspired SSL objective
- Predictive neural representations

Advantages

- Easy and computationally less demanding
- Representational capacity and robustness
- Alternative for hand crafted augmentations

Extensions

- Multiple time scale
  - Hierarchical temporal structure for long horizon predictions
- Multiple network stages

NYU

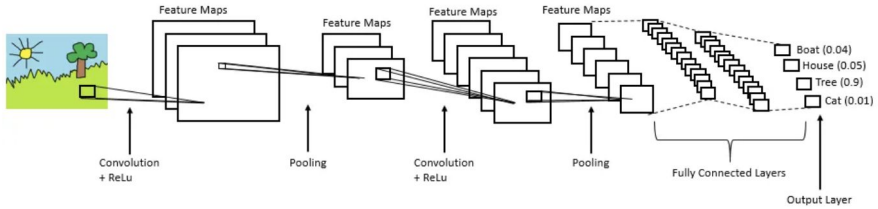
Moving Off-the-Grid: Scene-Grounded Video Representations

Name: Anurup Naskar
NetID: an4462

Traditional Vision Based Models

Convolutional Neural Networks (CNNs)

CNNs learn hierarchical feature representations using spatially localized filters (kernels) that slide over the image in a fixed grid pattern. Each convolutional filter captures local patterns such as edges, textures, and objects, which are then combined at deeper layers to form more abstract representations. CNNs exhibit *translation invariance* due to weight sharing, meaning they can recognize objects regardless of their position in the image. However, they struggle with long-range dependencies and rely on pooling layers or downsampling techniques to aggregate global information.

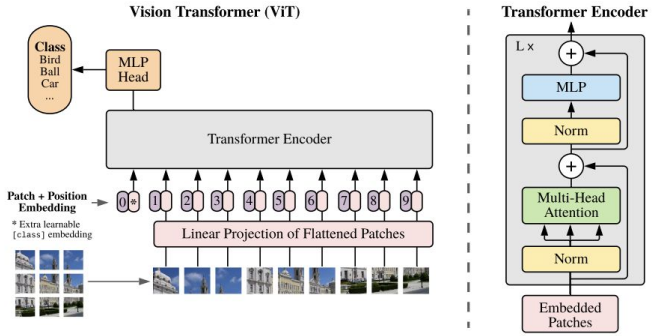


NYU

Traditional Vision Based Models

Vision Transformers (ViTs)

ViTs use self-attention instead of convolutions to model image representations. They divide an image into non-overlapping patches, which are then flattened and projected into a sequence of embeddings (tokens). A Transformer encoder applies *multi-head self-attention* to model relationships between tokens, allowing *global feature extraction* from the start. Compared to CNNs, ViTs capture long-range dependencies more effectively and enable more flexible representations. However, ViTs still maintain a fixed grid structure since tokens correspond to static image patches.



NYU

Research Gap In Traditional Vision Models

- **Rigid Token Alignment:** Struggles to track moving objects, causing inconsistent representations.
- **Grid-Induced Bias:** Tokens are constrained to static locations, limiting motion adaptability.
- **Limited Generalization:** Fixed grids fail to capture spatio-temporal variations in videos.

NYU

Objective

- MooG is a recurrent model designed for processing an arbitrary number of video frames while maintaining a consistent scene-grounded OTG representation.
- Takes as input a sequence of observed frames $\{X_t\}_{t=1}^{T}$, where $X_t \in \mathbb{R}^{H \times W \times 3}$.
- Iteratively encodes frames into a set of latent tokens.
- Separates the latent state into:
  - Corrected states $\{z_c^t\}_{t=1}^{T}$, where $z_c^t \in \mathbb{R}^{K \times D}$, obtained by encoding input frames.
  - Predicted states $\{z_p^t\}_{t=1}^{T}$, where $z_p^t \in \mathbb{R}^{K \times D}$, representing the model's internal prediction of the next frame.
- The recurrent structure allows tokens to track elements consistently across frames and anticipate their future locations.

Encoder CNN:

Features	Kernel	stride
64	3×3	1×1
128	3×3	1×1
128	3×3	1×1
256	3×3	2×2
256	3×3	1×1
512	3×3	2×2

NYU

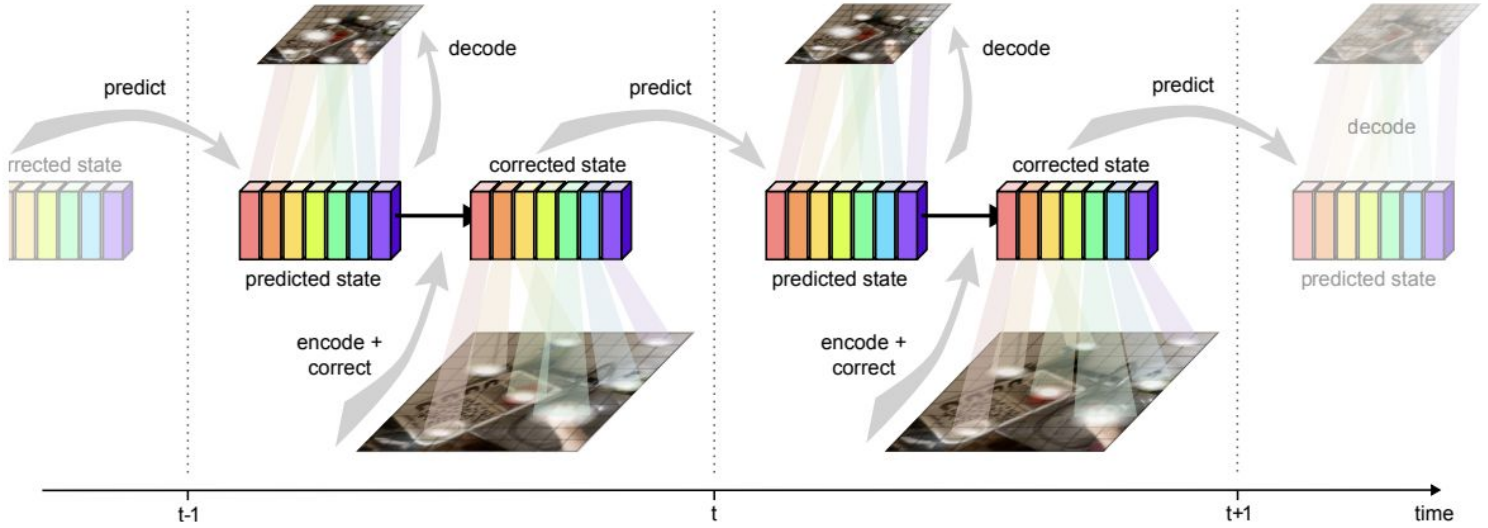
Training Objective

- The model is trained for next-frame prediction given the previous frame and model state.
- Composed of three main networks:
  - **Predictor** (P): Predicts the next state from the previous corrected state.
  - **Decoder** (D): Decodes the predicted state to reconstruct the current frame.
  - **Corrector** (C): Encodes the current frame and corrects errors in the predicted state.

Component	Type	Layers	QKV size	Heads	MLP size
Corrector	XA & SA	2	64×8	8	2048
Predictor	SA	3	64×4	4	2048
Decoder	XA	6	64×2	2	2048

NYU

Architecture



NYU

Predictor

- Takes the previous corrected state z_c^{t-1} and generates the predicted state z_p^t:

$$z_p^t = z_c^{t-1} + P(kqv = z_c^{t-1})$$

- Functions as a self-attention transformer network.
- Initial corrected state z_c^0 is initialized with Gaussian noise ($\mathcal{N}(0, 10^{-4})$).
- Ensures permutation symmetry and prevents token specialization.

NYU

Corrector

- Incorporates information from the current observation X_t to update z_p^t and form z_c^t.
- Uses a convolutional encoder E with Fourier positional embeddings.
- Outputs a feature grid $F_t \in \mathbb{R}^{H' \times W' \times D}$.
- Updates the state using cross-attention:

$$z_c^t = z_p^t + C(kv = F_t, q = z_p^t), \quad \text{where } F_t = E(X_t)$$

- The corrected state does not receive a direct loss; it serves as an improved estimate for the predictor.

NYU

Decoder

- Takes the predicted state z_p^t and decodes it into an RGB image.
- Uses cross-attention with spatial queries for pixel-based readouts.
- Decodes only a sub-sampled version of the target image for computational efficiency:

$$\tilde{X}_t = D(kv = z_p^t, q = P)$$

- Decoder attention weights reveal relationships between spatial positions and latent tokens.

NYU

Loss Calculation

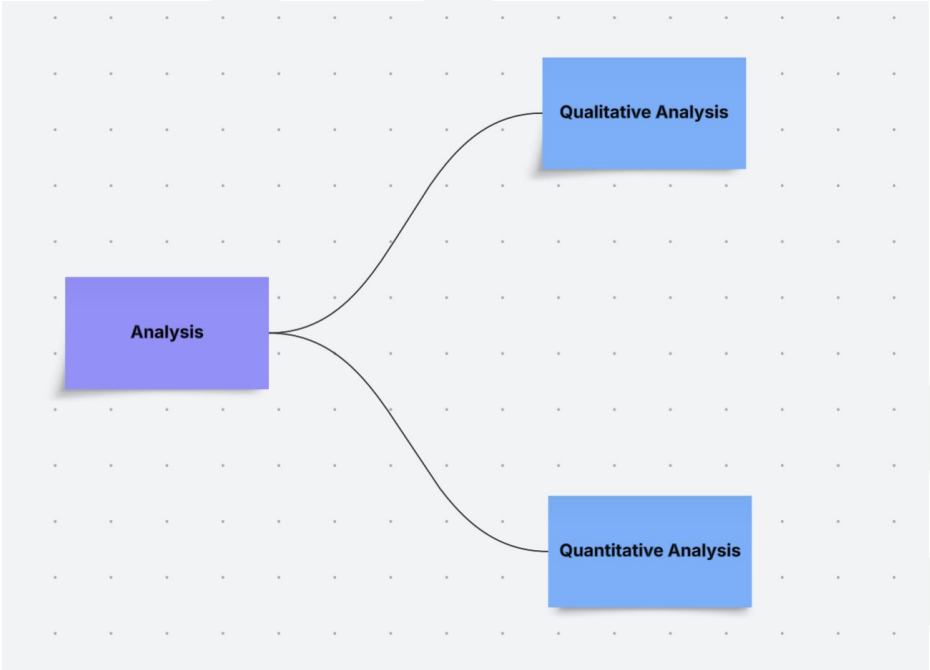
- Uses an L2 loss on image pixels.
- For each frame, the loss is computed as:

$$L_t = L2(\tilde{X}_t, X_t)$$

- The predicted state depends only on the previous frame and model state.
- Training involves unrolling the model over 8 frames.

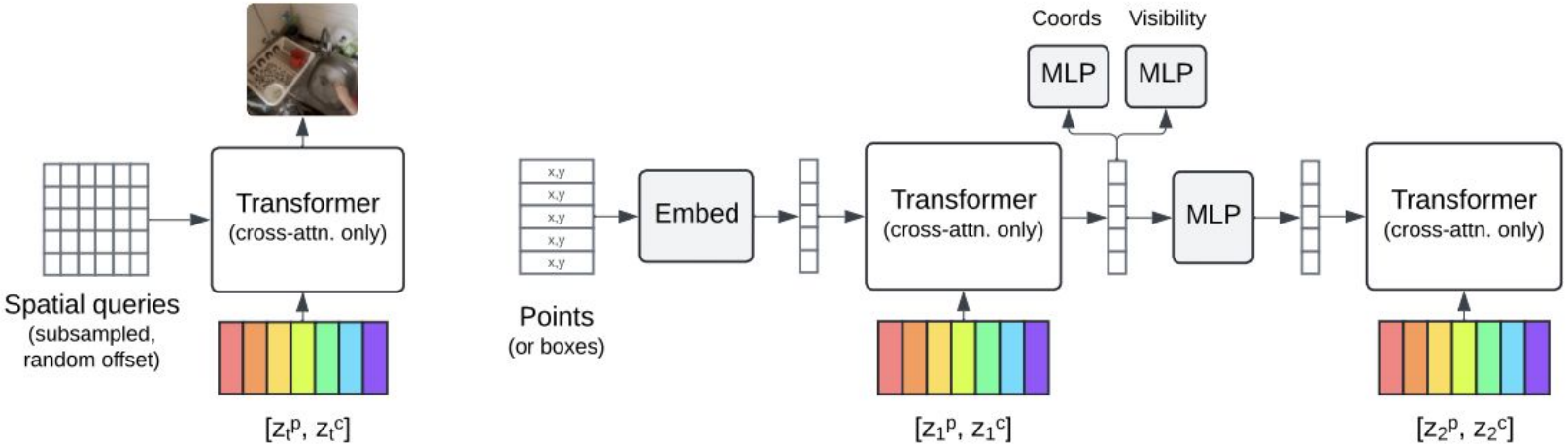
NYU

Analysis



NYU

Readout Decoders

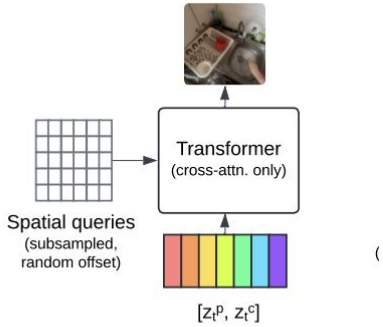


(a) Pixel readout (RGB, depth).

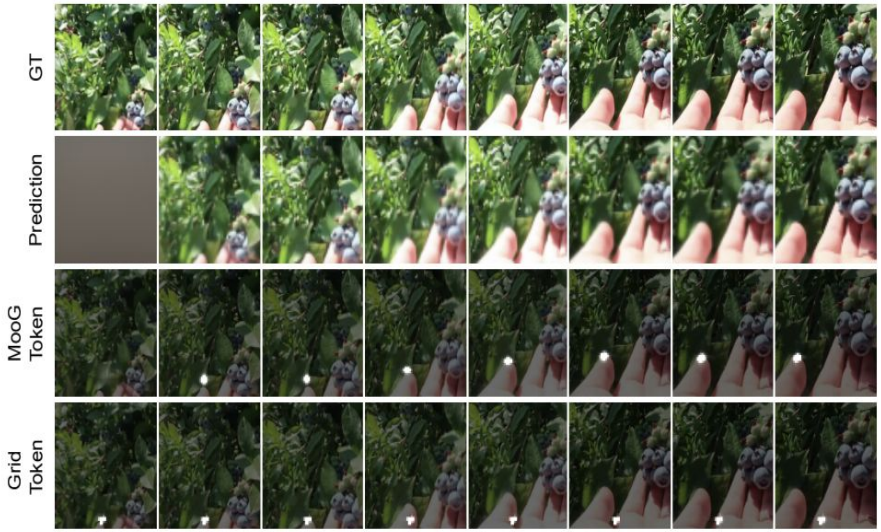
(b) Recurrent readout (points, boxes).

NYU

Qualitative Analysis



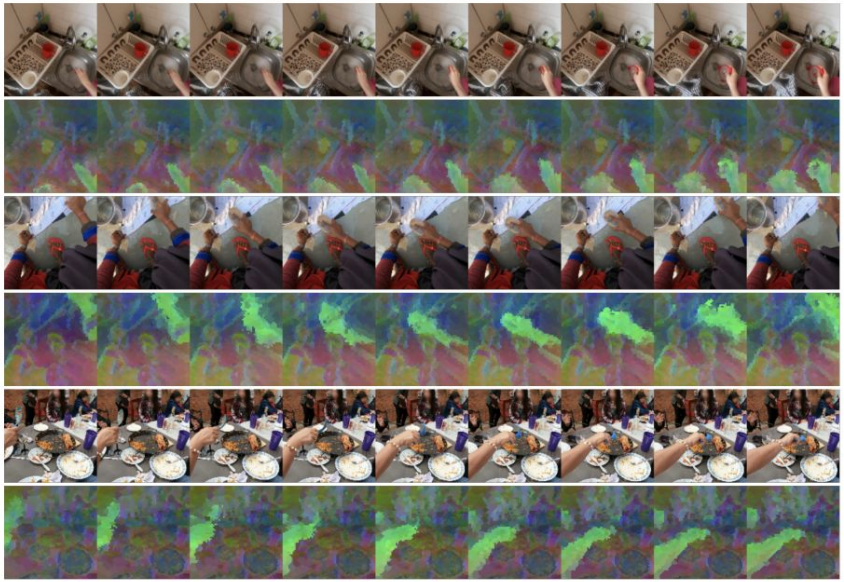
(a) Pixel readout (RGB, depth).



The MooG attention map indicates that the visualized token tracks the scene element it binds to across the full range of motion. In contrast, the grid-based token attention map demonstrates how these tokens end up being associated with a specific image location that does not track the scene content.

NYU

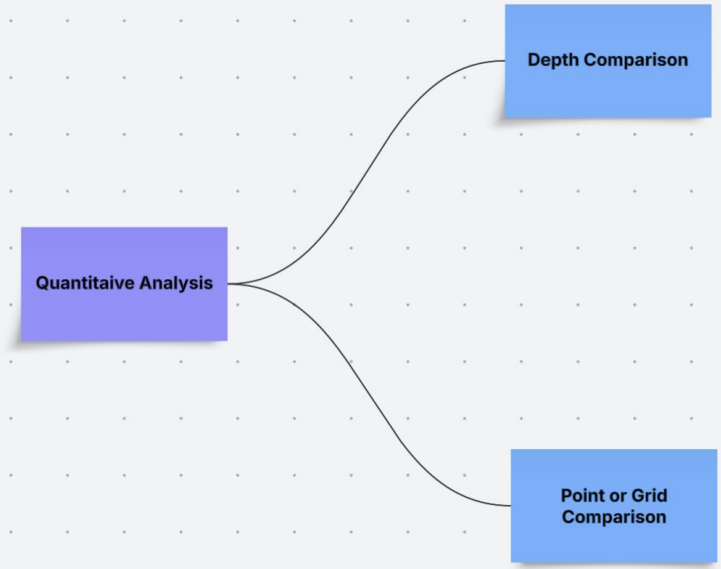
Qualitative Analysis



PCA of MooG tokens unrolled over a batch of short sequences. The model was unrolled over a batch of 24 sequences, 12 frames each. Predicted states from all time steps and batch samples were concatenated and PCA analysis was performed on the entire set jointly. We then reshape the projected set back to its original shape and use the arg-max token to visualize the result in image space.

NYU

Quantitative Analysis



Type	Type	Layers	QKV size	Heads	MLP size
Points	XA	3	64×8	8	2048
Depth	XA	3	64×8	8	2048
Boxes	XA	3	64×8	8	2048

NYU

Downstream readout performance from frozen representations

	MOVi-E			DAVIS	Waymo
Name	Points (↑AJ)	Depth (↓AbsRel)	Boxes (↑IoU)	Points (↑AJ)	Boxes (↑IoU)
MooG	**0.839**	0.0359	**0.793**	0.687	**0.730**
Grid	0.769	0.0451	0.730	0.518	0.625
Grid Rec.	0.778	0.0443	0.734	0.559	0.629
DINOv1 (B)	0.518	0.0371	0.724	0.409	0.566
DINOv2 (B)	0.544	0.0370	0.738	0.402	0.559
VMAEv2 (S)	0.595	0.0567	0.700	0.365	0.567
VMAEv2 (B)	0.681	0.0458	0.736	0.434	0.611
VMAEv2 (G)	0.822	**0.0311**	**0.793**	**0.720**	0.708

NYU

Downstream readout performance trained in an end-to-end manner

	MOVi-E			Davis	Waymo
Name	Points (↑AJ)	Depth (↓AbsRel)	Boxes (↑IoU)	Points (↑AJ)	Boxes (↑IoU)
MooG	0.886	0.0263	0.803	0.778	**0.719**
Grid	0.860	0.0264	0.775	0.644	0.615
Grid Rec.	**0.902**	**0.0233**	**0.806**	**0.779**	0.675
DINOv1	0.698	0.0381	0.728	0.578	0.557
DINOv2	0.732	0.0439	0.734	0.656	0.607

NYU

Depth comparison (End to end)

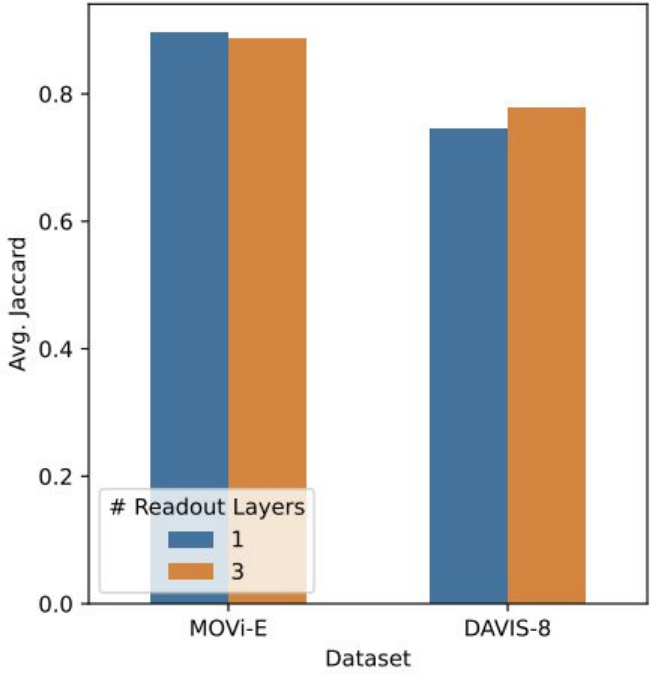
Name	Waymo (↓AbsRel)
MooG	**0.094**
DPT (ViT-L/16)	0.161
DPT (ViT-E/14)	0.158
DPT (ViT-22b)	0.154

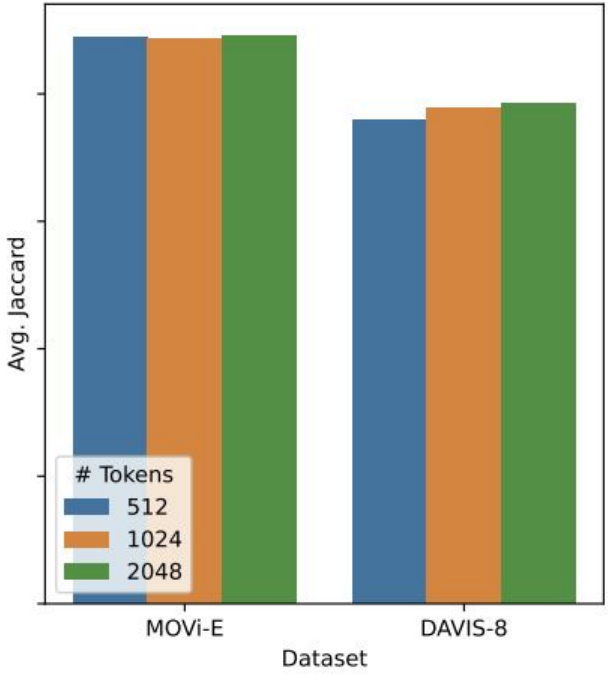
Points comparison (End to end)

Name	Davis-8 (↑AJ)	Davis-full (↑AJ)
MooG	**0.824**	0.510
TAP-Net	0.687	0.392
TAPIR	**0.823**	**0.580**

NYU

No of Tokens





NYU

Limitations

- **Task Generalization:** Evaluations focused on readout tasks (e.g., object tracking, depth prediction), but effectiveness for tasks like semantic segmentation, classification, and generation remains unclear.
- **Handling Scene Changes:** OTG representations may struggle with disappearing or reappearing scene elements, unlike grid-based representations that change more gradually.

NYU

Conclusion

- Traditional vision models (CNNs, ViTs) rely on fixed *on-the-grid* architectures tied to pixel grids.
- The real world does not exist on a pixel grid—rigid architectures limit adaptability to motion and scene changes.
- **MooG** enables representations to move *off-the-grid*, binding to scene elements dynamically.
- Demonstrates effectiveness in tasks requiring *motion understanding* and *scene geometry*.
- Current model is simple: deterministic, lacks uncertainty modeling, and uses an L2 pixel loss.

NYU

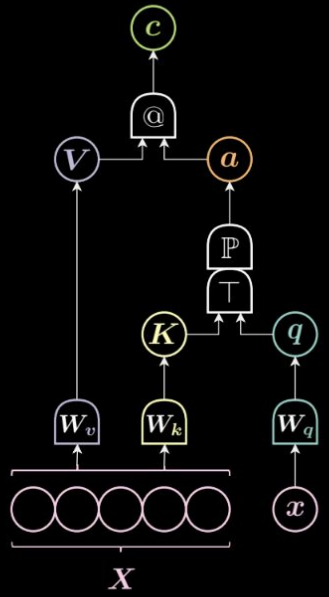
Thank You

NYU

Appendix

Self-attention

- For every $\boldsymbol{x} \in \{\boldsymbol{x}_m\}_{m=1}^{M} = \boldsymbol{X}$, compute
  - query $\boldsymbol{q} = \boldsymbol{W}_q \boldsymbol{x}$
  - key $\boldsymbol{k} = \boldsymbol{W}_k \boldsymbol{x} \quad \boldsymbol{K} = \boldsymbol{W}_k \boldsymbol{X}$
  - value $\boldsymbol{v} = \boldsymbol{W}_v \boldsymbol{x} \quad \boldsymbol{V} = \boldsymbol{W}_v \boldsymbol{X}$
- Compute the query-key match $\boldsymbol{r} = \boldsymbol{K}^\top \boldsymbol{q}$
- The context vector $\boldsymbol{c} = \boldsymbol{V}\boldsymbol{a}$
- where $\boldsymbol{a} = \mathrm{softargmax}_\beta(\boldsymbol{r})$

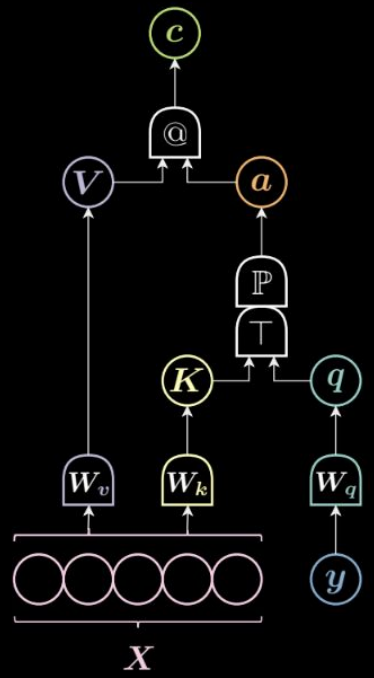


Multi-headed attention: multiple replicas of the simple attention circuit.

NYU

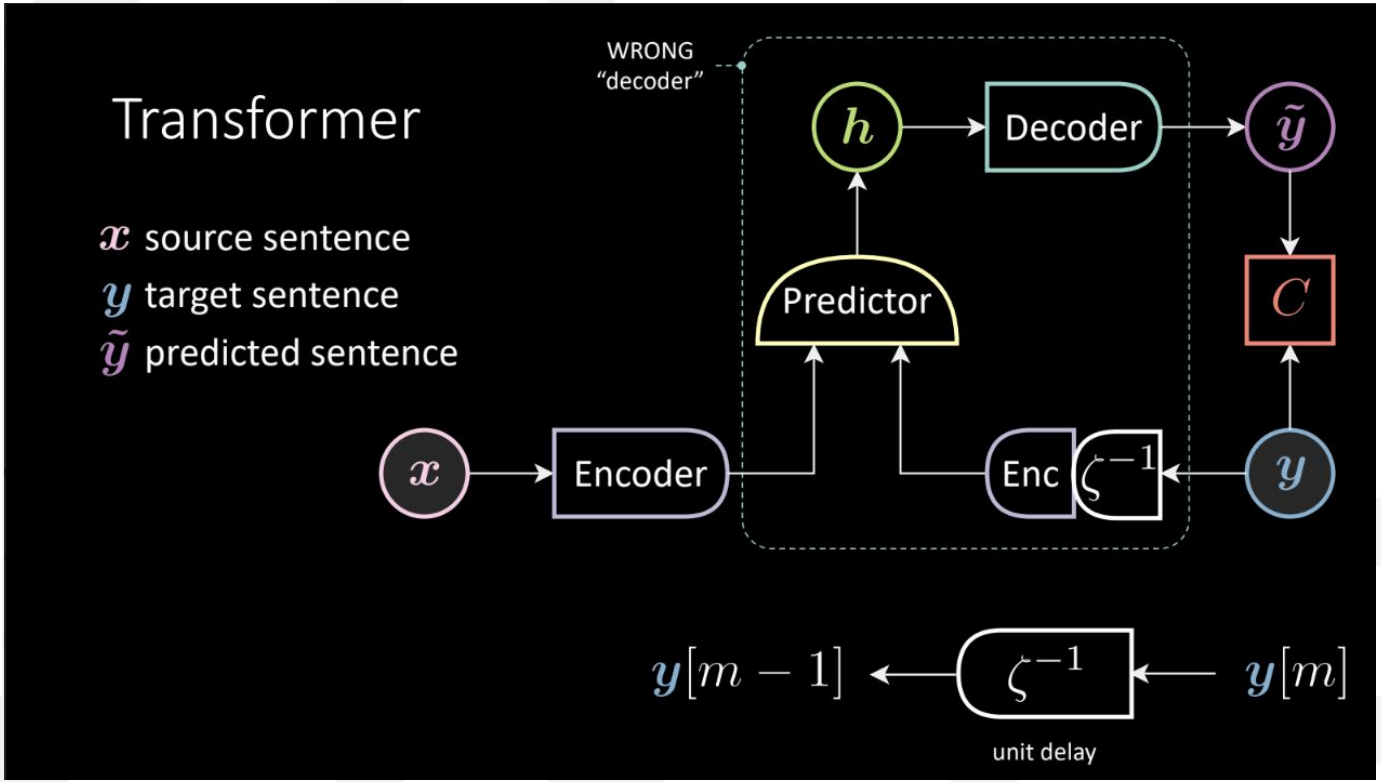
Cross-attention

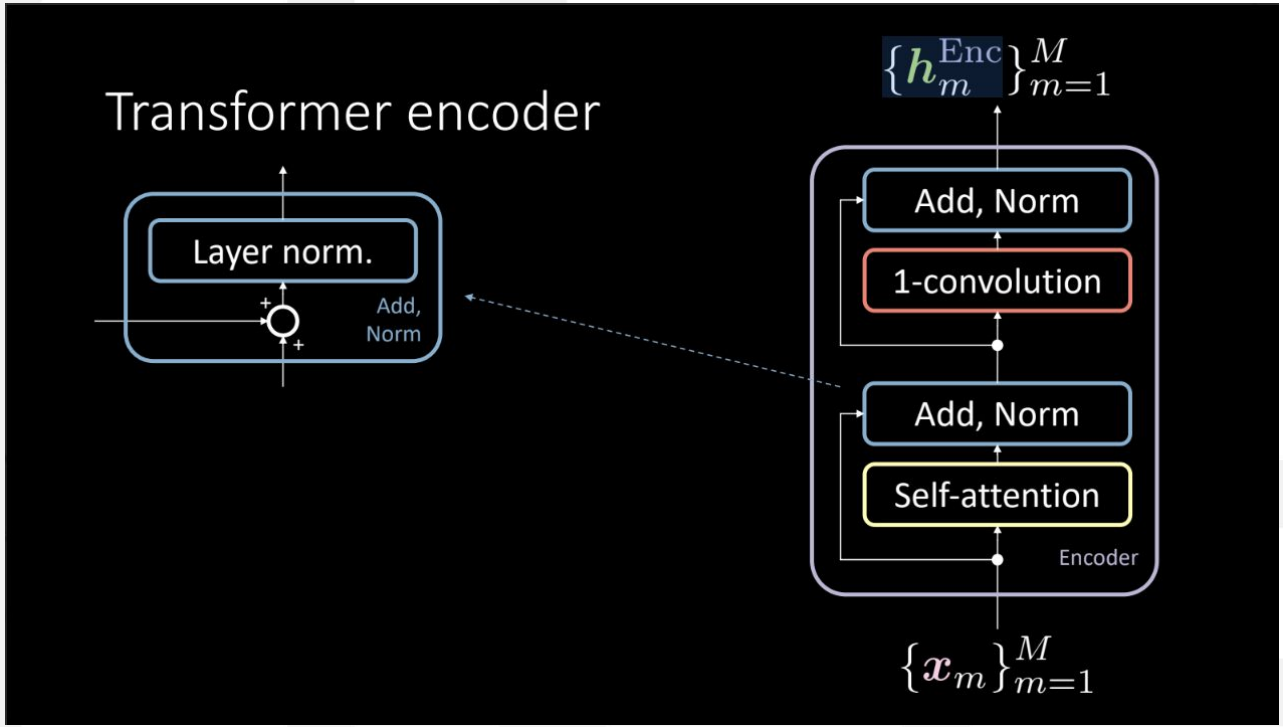
- For every $y \in \{y_n\}_{n=1}^{N}$, compute
  - query $q = W_q y$
  - key $k = W_k x$ $K = W_k X$
  - value $v = W_v x$ $V = W_v X$
- Compute the query-key match $r = K^\top q$
- The context vector $c = V a$
- where $a = \mathrm{softargmax}_\beta(r)$



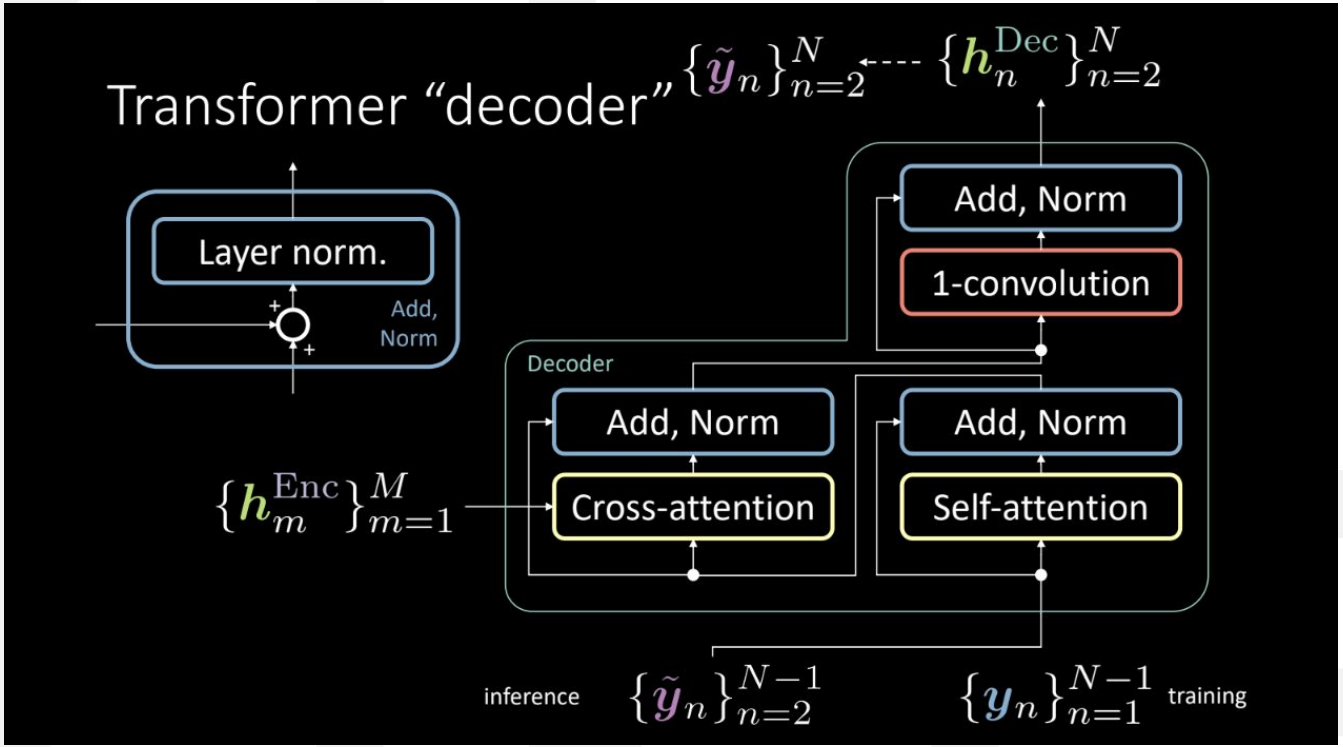
Multi-headed attention: multiple replicas of the simple attention circuit.

NYU

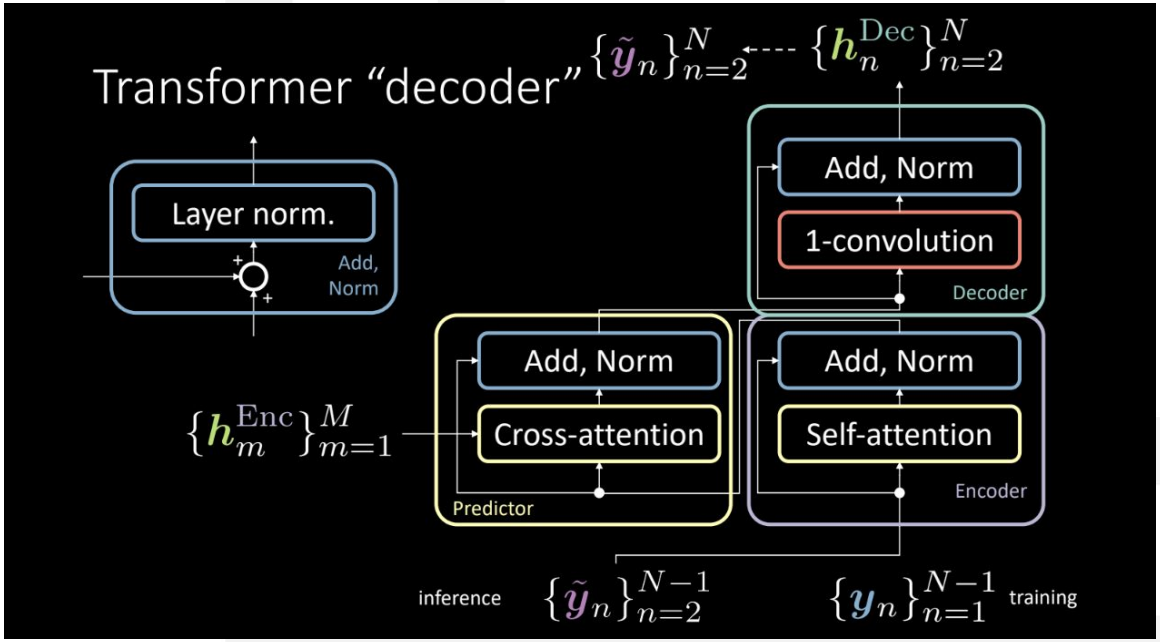

Transformer
x source sentence
y target sentence
$\tilde{y}$ predicted sentence
WRONG
"decoder"
h
Decoder
$\tilde{y}$
Predictor
C
x
Encoder
Enc
ζ^{-1}
y
$y[m-1]$
ζ^{-1}
$y[m]$
unit delay


Transformer encoder
$\{h_m^{\text{Enc}}\}_{m=1}^{M}$
Layer norm.
Add, Norm
Add, Norm
1-convolution
Add, Norm
Self-attention
Encoder
$\{x_m\}_{m=1}^{M}$

NYU


Transformer "decoder"
$\{\tilde{y}_n\}_{n=2}^{N}$
$\{h_n^{\text{Dec}}\}_{n=2}^{N}$
Layer norm.
Add, Norm
Add, Norm
1-convolution
Decoder
Add, Norm
Add, Norm
$\{h_m^{\text{Enc}}\}_{m=1}^{M}$
Cross-attention
Self-attention
inference
$\{\tilde{y}_n\}_{n=2}^{N-1}$
$\{y_n\}_{n=1}^{N-1}$
training

NYU


Transformer “decoder”
$\{\tilde{y}_n\}_{n=2}^{N}$
$\{h_n^{\text{Dec}}\}_{n=2}^{N}$
Layer norm.
Add, Norm
Add, Norm
1-convolution
Decoder
$\{h_m^{\text{Enc}}\}_{m=1}^{M}$
Add, Norm
Cross-attention
Predictor
Add, Norm
Self-attention
Encoder
inference
$\{\tilde{y}_n\}_{n=2}^{N-1}$
$\{y_n\}_{n=1}^{N-1}$
training

NYU

Self-Supervised Depth

- Appearance matching loss

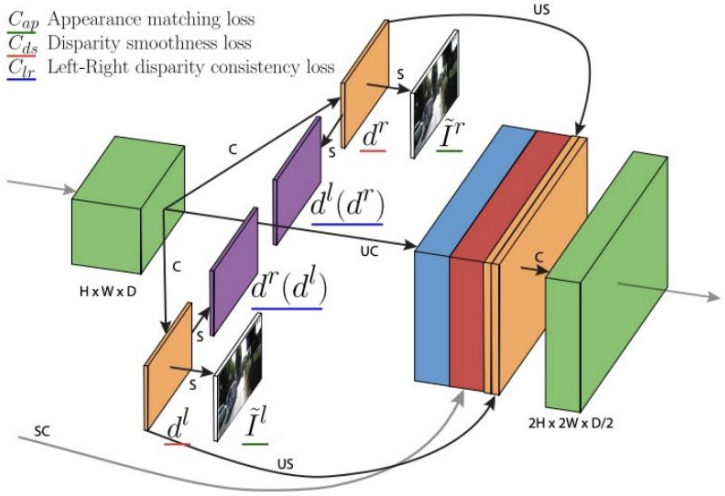
$$C_{ap}^{l}=\frac{1}{N}\sum_{i,j}\alpha\frac{1-\mathrm{SSIM}(I_{ij}^{l},\tilde{I}_{ij}^{l})}{2}+(1-\alpha)\left\|I_{ij}^{l}-\tilde{I}_{ij}^{l}\right\|.$$

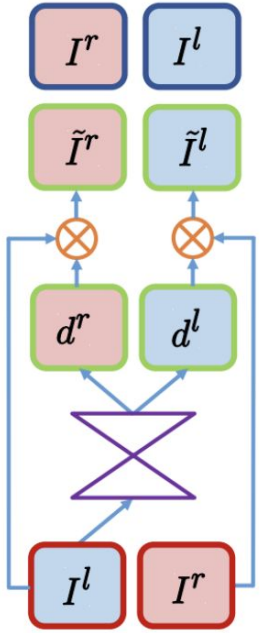
- Disparity smoothness loss

$$C_{ds}^{l}=\frac{1}{N}\sum_{i,j}\left|\partial_x d_{ij}^{l}\right|e^{-\left\|\partial_x I_{ij}^{l}\right\|}+\left|\partial_y d_{ij}^{l}\right|e^{-\left\|\partial_y I_{ij}^{l}\right\|}.$$

- Left-right disparity consistency loss

$$C_{lr}^{l}=\frac{1}{N}\sum_{i,j}\left|d_{ij}^{l}-d_{ij+d_{ij}^{l}}^{r}\right|.$$





DayDreamer

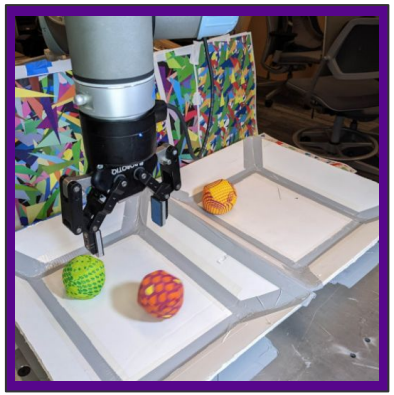
World Models for Physical Robo Learning

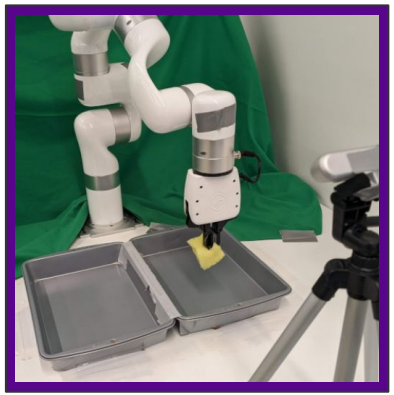
Andrew Deur

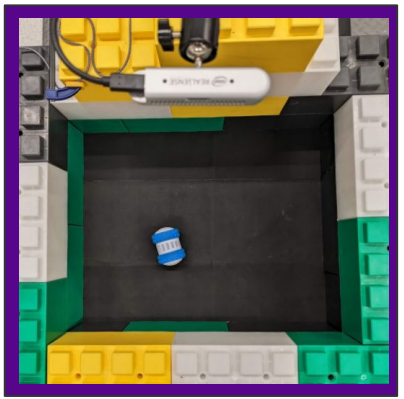
Background

Challenges in Robotic Learning

- Data Intensive: Robots require millions of interactions to learn simple behaviors
- Real-world learning is impractical due to time, wear-and-tear and safety constraints
- Simulators are required to provide robots with sufficient training trials, but, they fail to capture real-world complexity.







NYU

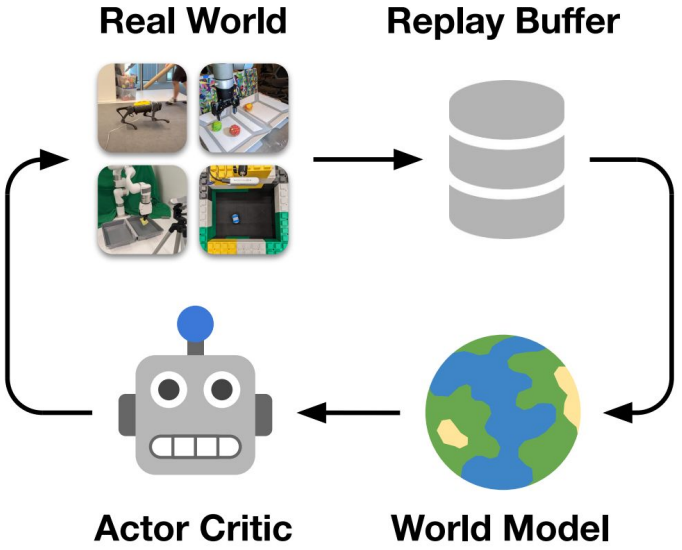
World Models & Embodied Learning

Embodied Learning

- Robots learn through physical interactions with the real world rather than pre-collected datasets or simulations

Planning from Learned World Model

- Predict the consequences actions using a learned world model to plan as opposed to learning through random interactions



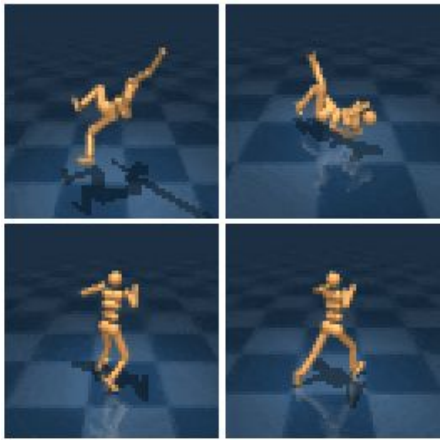
DayDreamer Motivation

Built on DreamerV2 (Hafner et al.)

- Extends Dreamer from simulation to real-world robotics without pre-training

Imagined Rollouts using a World Model

- Instead of physically trying every action, learn a World Model and ‘imagine’ outcomes of different actions
  - Greatly improves sample efficiency
  - Enables learning from real world interaction in faster than real time
  - Allows the robot to learn off-policy, improving exploration / exploitation tradeoff

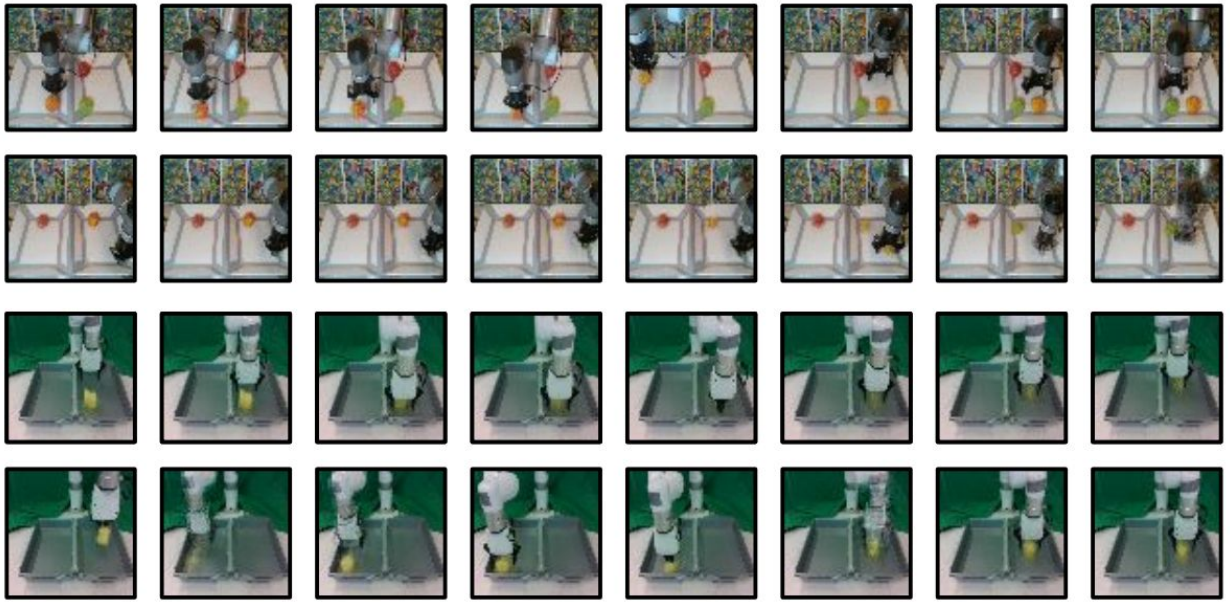


NYU

DayDreamer Approach Overview

Applies Dreamer to real-world robotics

- No simulation or pre-training. Learning is done directly on hardware with rewards from interaction with the environment

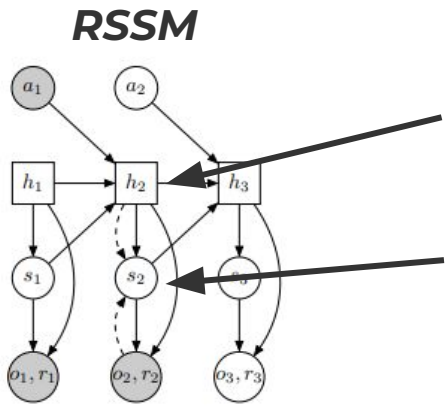


NYU

World Model Architecture

Recurrent State Space Model (RSSM)

- Combines deterministic and stochastic components to model dynamics
- Stochastic latent state models uncertainty in robot sensor subject to random noise
- Improves handling of partial observability



Deterministic State

- $h_t = f(h_{t-1}, s_{t-1}, a_{t-1})$

Stochastic State

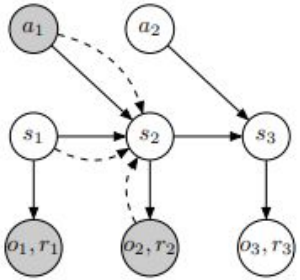
- $s_t \sim p(s_t \mid h_t)$

Observation Model

- $o_t \sim p(o_t \mid h_t, s_t)$

Reward Model:

- $r_t \sim p(r_t \mid h_t, s_t)$



Stochastic Model

NYU

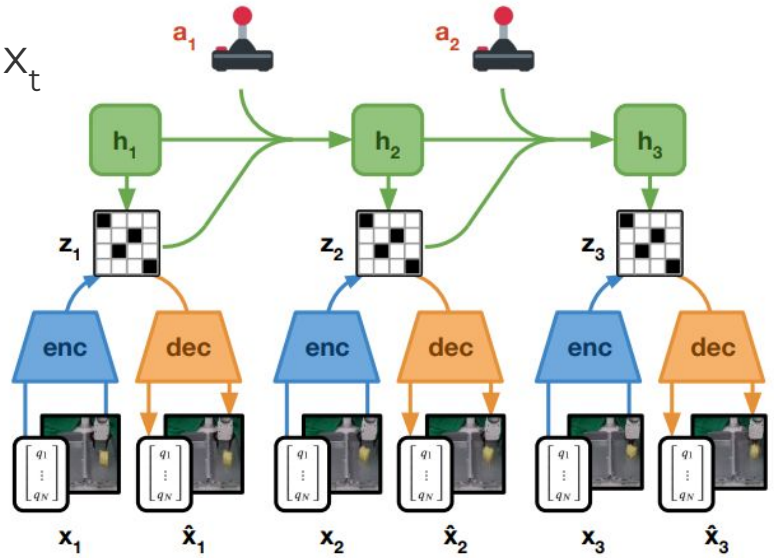
World Model Architecture (cont.)

Encoder Network
- Fuses sensors of different modalities, x_t

Dynamics Network
- Predicts sequence of stochastic representations, z_t

Decoder Network
- Reconstructs input from z_t

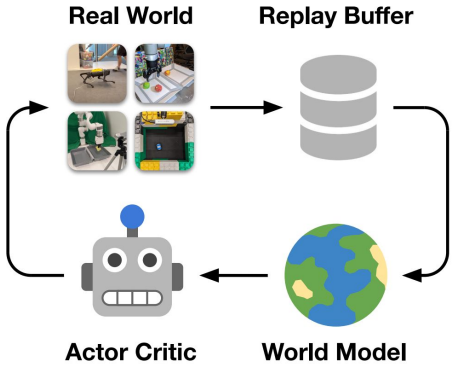
Reward Network
- Predict reward from z_t for training rollouts



World Model Learning

Discrete Latent Code Predictions Z_t

- World Model makes predictions in latent space to reduces computation and accumulating errors
  - Fast simulator of the environment that the robot learns autonomously



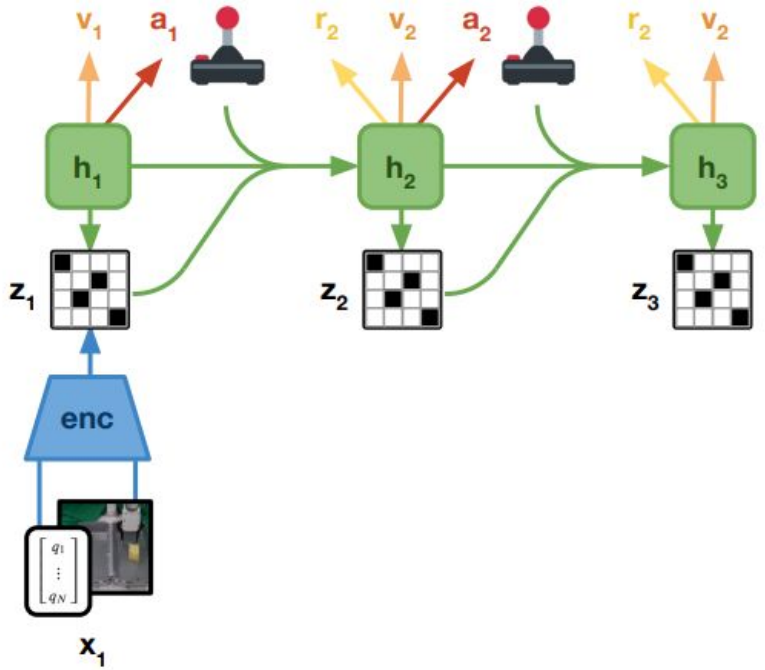
Replay Buffer

- Learns world model from buffer of past experiences
- Decouples learning from data collection to enable fast training without waiting for environment
- World Model is learned off-policy, improving exploration

Behavior Model Architecture

Actor Critic Learning

- World Model represents task-agnostic knowledge about environment dynamics
- Actor Critic Networks learns behaviors to accomplish task at hand
- Behaviors learned on rollouts predicted in latent space of world model, without decoding back to parameter space



Behavior Model Architecture (cont.)

Actor Network

- Learns a distribution over successful actions for each latent model state to maximize future predicted rewards

$$\pi \sim (a_t \mid z_t)$$

- Policy gradients estimated using Reinforce for discrete control and reparametrization for continuous control
- High entropy is also incentivized to prevent collapsing to deterministic policy.

$$\mathcal{L}(\pi) \doteq -\mathrm{E}\Big[\sum_{t=1}^{H} \ln \pi(a_t \mid s_t)\, \mathrm{sg}(V_t^{\lambda} - v(s_t)) + \eta\, \mathrm{H}\big[\pi(a_t \mid s_t)\big]\Big]$$

Behavior Model Architecture (cont.)

Critic Network

- Trained to regress the return of the trajectory. Learned through temporal-difference learning to allow taking into account rewards beyond planning horizon H = 16 steps

$$v(z_t)$$

- λ-returns averaging over all N $\in$ [1, H-1] to avoid choosing arbitrary N for TD learning. Slowly updating target critic used for computing λ-returns.

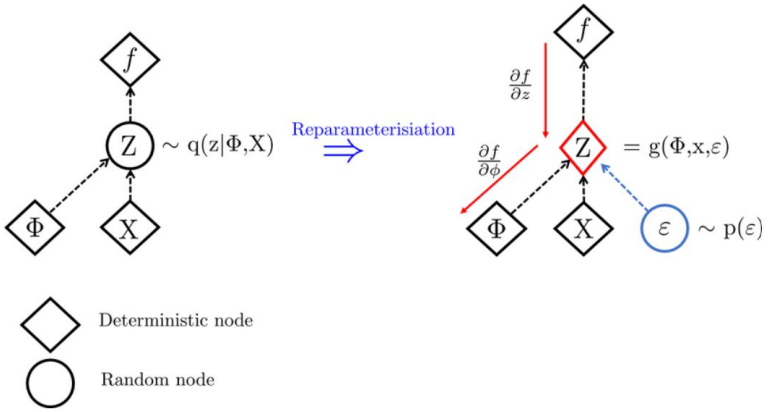
$$V_t^\lambda \doteq r_t + \gamma\Big((1-\lambda)v(s_{t+1}) + \lambda V_{t+1}^\lambda\Big), \quad V_H^\lambda \doteq v(s_H).$$

NYU

Implementation Details

Asynchronous Training

- Robot collects data in real time, storing it in replay buffer while training model and policy in parallel—no idle waiting.



Stochastic Backpropagation

- World model learns distributions over latent states ,so standard backpropagation can’t be used for training.
- Uses reparameterization trick so gradients can flow through probability distributions.

Implementation Details

Same Hyperparameters Across Robots and Tasks

- Same learning rate, network sizes, etc. across all robots and tasks,
- Underscores robust, task-agnostic generalization of DayDreamer

Name	Symbol	Value
General		
Replay capacity (FIFO)	—	10^6
Start learning	—	10^4
Batch size	B	32
Batch length	T	32
MLP size	—	4×512
Activation	—	LayerNorm + ELU
World Model		
RSSM size	—	512
Number of latents	—	32
Classes per latent	—	32
KL balancing	—	0.8
Actor Critic		
Imagination horizon	H	15
Discount	γ	0.95
Return lambda	λ	0.95
Target update interval	—	100
All Optimizers		
Gradient clipping	—	100
Learning rate	—	10^{-4}
Adam epsilon	ϵ	10^{-6}

NYU

Experiments Overview

Experiment Name	Input Space	Action Space	Reward Type	Training Time
1 - Quadruped Walking	Proprioceptive Readings	Continuous	Dense	1 hour
2 - UR5 Multi-Object Visual Pick and Place	RGB Image, Proprioceptive Readings	Discrete	Sparse	8 hours
3 - XArm Visual Pick and Place	RGB Image, Depth Map, Proprioceptive Readings	Discrete	Sparse	10 hours
4 - Sphero Navigation	RGB Image	Continuous	Dense	2 hours

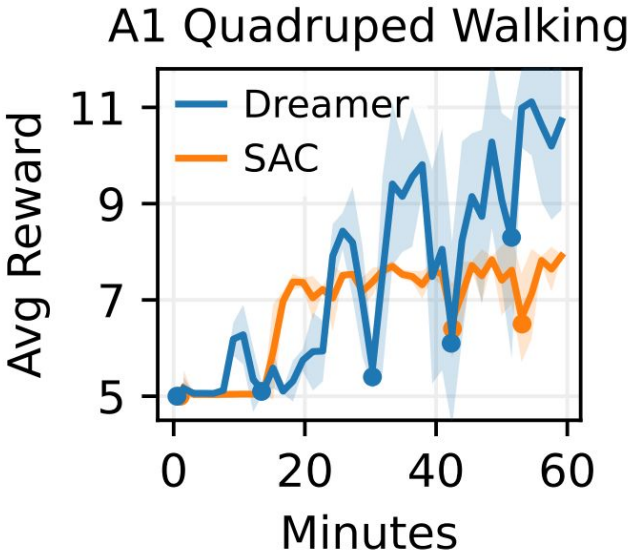
Quadruped Walking

DayDreamer learns to get up, stand and walk within 1 hour.

- Learns without environment re-starts
- 10 minutes of additional online learning and it can withstand pushes

SAC learns to roll off its back

- Can’t stand up or walk due to limited training budget

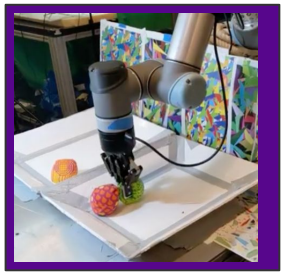


NYU

UR5 Multi-Object Visual Pick & Place

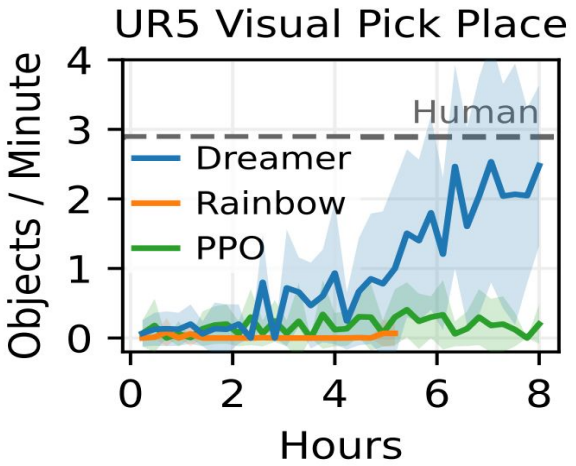
DayDreamer achieves human level performance within 8 hours

- Challenging due to sparse rewards, struggles to learn for first 2 hours


Rainbow and PPO only learn short sighted behaviors and drop items in same bin

- Rainbow and PPO fail because they require quantities of data that are infeasible to collect in real world



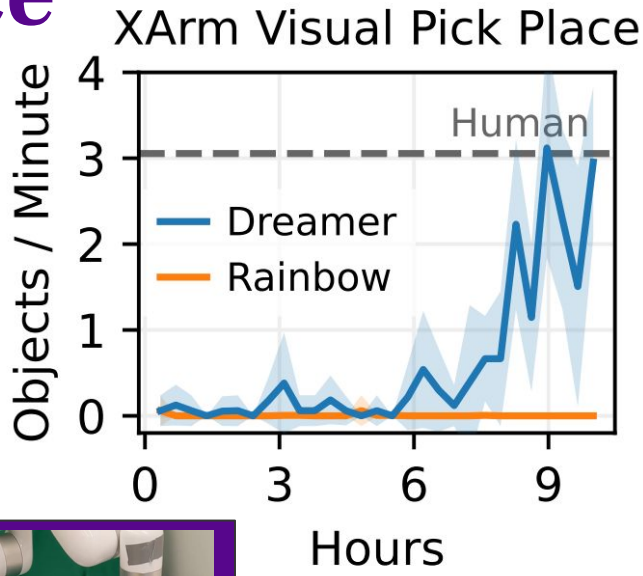
NYU

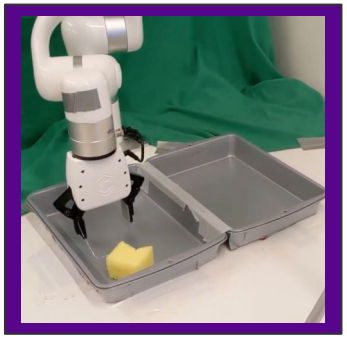
XArm Visual Pick and Place

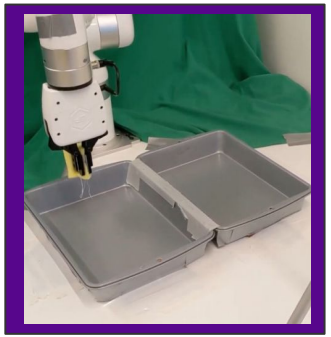
DayDreamer achieves human level performance within 10 hours

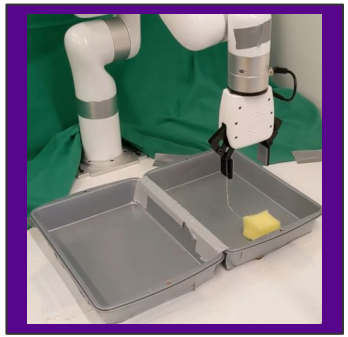
- Exhibited multimodal behaviors, using string to pull object out of corners

Rainbow fails to learn how to accomplish this task







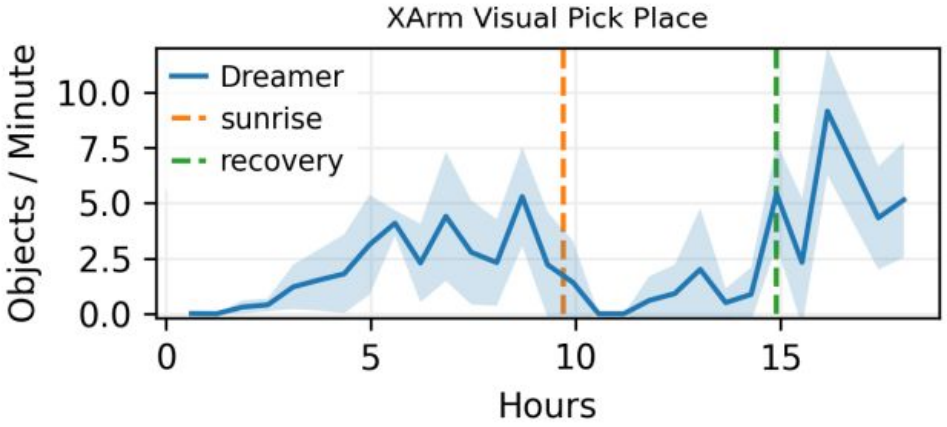


NYU

XArm Visual Pick and Place Adaptation

Continual Learning

- Extreme lighting changes, particularly shadows after sunrise, cause performance to collapse
- DayDreamer adapts and recovers to previous performance after 5 hours of additional training
  - Less than learning the task from scratch



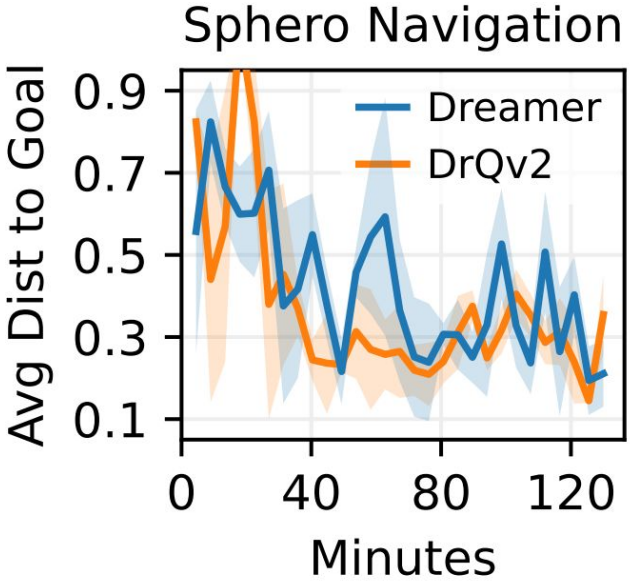
NYU

Sphero Navigation

DayDreamer matches performance with model designed for this specific problem space within 2 hours

- Infers heading direction from history of observations because it only has access to image observations

DrQv2 is a model-free algorithm specifically designed for continuous control from pixels alone



NYU

Practical Applications & Implications

- **Direct, Embodied Learning:** Trains in the real world—no simulators or pre-training. Its learned world model supports adaptive “dreaming” for robust planning.
- **Continual, Multi-Task Potential:** Handles locomotion, manipulation, and navigation with minimal tuning, enabling ongoing adaptation across tasks.
- **High Sample Efficiency:** Consistently surpasses other RL methods under the same training budget.

Conclusion & Future Work

Conclusion

- Learns complex tasks in hours on real hardware—no simulator needed.
- A single approach solves locomotion, manipulation, and navigation efficiently.

Future Work

- Extend to multi-task learning, longer training, and full autonomy.
- Integrate advanced exploration, safety constraints, or expert data; open-source fosters broader adoption.

Questions

World Models

UniSim: Learning Interactive Real-World Simulators

Sidhartha Reddy Potu

March 19, 2025

Introduction & Motivation

- **UniSim** is an action-conditioned video prediction model based on diffusion.
- Learns directly from visual & textual data.
- Overcomes limitations of traditional simulators (complexity).
- Applications: robotics control, instructional video generation.
- Leverages real-world video data for realistic, interactive simulation.

Related Work & Background

- Language models excel at text, not physical tasks.
- Video generation models focus on media, not multi-turn interactions.
- Image-based world models target games and simple simulations.
- Existing methods lack interactive control; UniSim fills that gap.

UniSim Dataset

UniSim is trained on diverse datasets:

	Dataset	# Examples	Weight
Simulation	Habitat HM3D (Ramakrishnan et al., 2021)	710	0.1
	Language Table sim (Lynch & Sermanet, 2020)	160k	0.05
Real Robot	Bridge Data (Ebert et al., 2021)	2k	0.05
	RT-1 data (Brohan et al., 2022)	70k	0.1
	Language Table real (Lynch & Sermanet, 2020)	440k	0.05
	Miscellaneous robot videos	133k	0.05
Human activities	Ego4D (Grauman et al., 2022)	3.5M	0.1
	Something-Something V2 (Goyal et al., 2017)	160k	0.1
	EPIC-KITCHENS (Damen et al., 2018)	25k	0.1
	Miscellaneous human videos	50k	0.05
Panorama scan	Matterport Room-to-Room scans (Anderson et al., 2018)	3.5M	0.1
Internet text-image	LAION-400M (Schuhmann et al., 2021)	400M	0.05
	ALIGN (Jia et al., 2021)	400M	0.05
Internet video	Miscellaneous videos	13M	0.05

Figure: Diverse datasets used to train UniSim

Conditioning on Multiple Modalities

UniSim processes multiple diverse modalities from various datasets:

Modalities as Conditioning Inputs:

- **Simulation Data**: Continuous control actions (e.g., Habitat, Language Table data).
- **Real Robot Data**: High-level task descriptions and discretized low-level controls.
- **Human Activity Videos**: Text-based action labels from activity recognition datasets.
- **Panorama Scans**: Camera pose information converted to actions (e.g., turning directions).
- **Internet Text-Image Data**: Captions treated as actions for single-frame images.

Unified Modality Alignment:

- Text tokens processed into continuous representations using T5 embeddings.
- Low-level robot actions concatenated with language embeddings to create a unified action space.

UniSim Overview

- Video diffusion model predicting future frames.
- Conditioned on past observations and actions.
- Multimodal inputs (vision, language, motor controls).

Figure 1: **A universal simulator (UniSim).** The simulator of the real-world learns from broad data with diverse information including objects, scenes, human activities, motions in navigation and manipulation, panorama scans, and simulations and renderings.

Figure: UniSim Training and Inference

UniSim Architecture

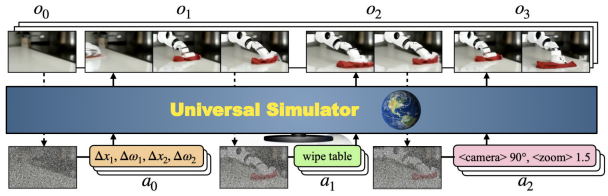


Figure: UniSim Architecture

UniSim is an action-conditioned video prediction model parameterized using diffusion models. Given a history h_{t-1} and action a_{t-1}, the model predicts the next observation frames o_t:

$$p(o_t \mid h_{t-1}, a_{t-1})$$

Where:

- h_{t-1}: Observations from the past (previous frames)
- a_{t-1}: Action input (e.g., "move left," "pick up object")

Architechture Cont'd

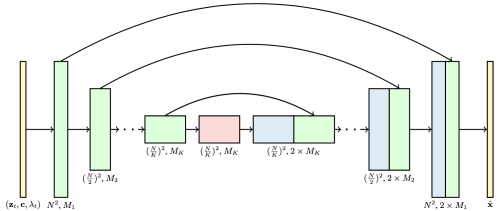


Figure: 3D-UNet

- Each block is a 4D tensor: frames $\times$ height $\times$ width $\times$ channels.
- Spatial attention treats the frame axis as a batch dimension.
- Temporal attention operates over frames, treating spatial (h, w, c) as batch tokens.
- Relative position embeddings preserve frame order.

Architecture Cont'd

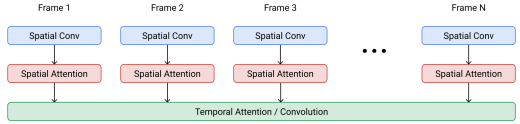


Figure: Video U-Net space-time separable block

- UniSim uses a history-conditioned base model at [16, 24, 40] resolution with temporal attention.
- Two cascaded super-resolution models upscale from [24, 40] to [48, 80] and then to [192, 320].
- The base model is conditioned on history by concatenating 4 previous frames channelwise with the U-Net's noise input.
- Temporal convolution is used in the super-resolution models for efficiency.

Training Objective

UniSim predicts future frames using a video diffusion model:
Diffusion Model Objective:

$$L_{\text{MSE}} = \|\epsilon - \epsilon_\theta(\sqrt{1-\beta^{(k)}}o_t + \sqrt{\beta^{(k)}}\epsilon, k \mid h_{t-1}, a_{t-1})\|^2$$

Classifier-Free Guidance:

$$\hat{\epsilon}_\theta(o_t^{(k)}, k \mid h_{t-1}, a_{t-1}) = (1+\eta)\,\epsilon_\theta(o_t^{(k)}, k \mid h_{t-1}, a_{t-1}) - \eta\,\epsilon_\theta(o_t^{(k)}, k \mid h_{t-1})$$

where η is the guidance scale. **Denoising step:**

$$o_t^{(k-1)} = \alpha^{(k)}\left(o_t^{(k)} - \gamma^{(k)}\epsilon_\theta(o_t^{(k)}, k|h_{t-1}, a_{t-1})\right) + \xi, \quad \xi \sim \mathcal{N}(0, \sigma_k^2 I)$$

Downstream use cases

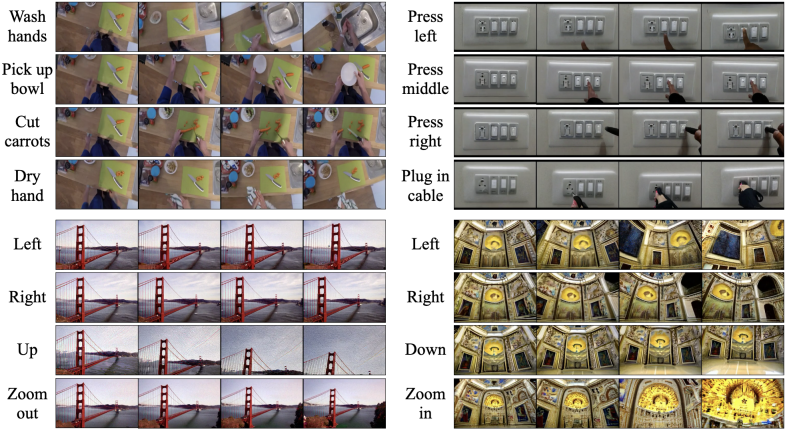


Figure 3: **Action-rich simulations.** UniSim can support manipulation actions such as "cut carrots", "wash hands", and "pickup bowl" from the same initial frame (top left) and other navigation actions.

Figure: Action rich, diverse long sequences

Downstream use cases

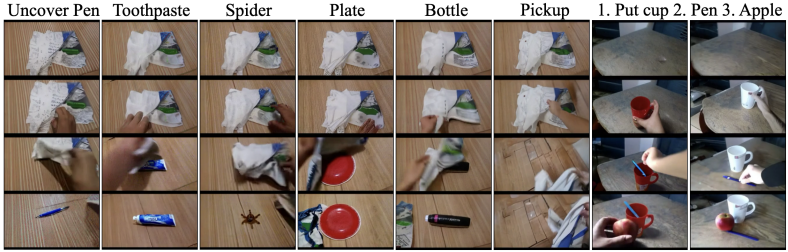


Figure 5: **Diverse and stochastic simulations.** On the left, we use text to specify the object being revealed by suffixing "uncovering" with the object name. On the right, we only specify "put cup" or "put pen", and cups and pens of different colors are sampled as a result of the stochastic sampling process during video generation.

Figure: Different simulation

Generating Synthetic Data

- Use placeholder frames (e.g., white images) with strong text guidance.
- Generate 4 videos per text from ActivityNet Captions.
- Fine-tune PaLI-X on 4× data.
- The generated videos align semantically well than AcitvityNet Caption(their claim, no examples).
- Helpful in generating rare event data.

	Activity	MSR-VTT	VATEX	SMIT
No finetune	15.2	21.91	13.31	9.22
Activity	54.90	24.88	36.01	16.91
Simulator	46.23	**27.63**	**40.03**	**20.58**

Table 4: **VLM trained in UniSim** to perform video captioning tasks. CIDEr scores for PaLI-X finetuned only on simulated data from UniSim compared to no finetuning and finetuning on true video data from ActivityNet Captions. Finetuning only on simulated data has a large advantage over no finetuning and transfers better to other tasks than finetuning on true data.

Long-Horizon Simulations

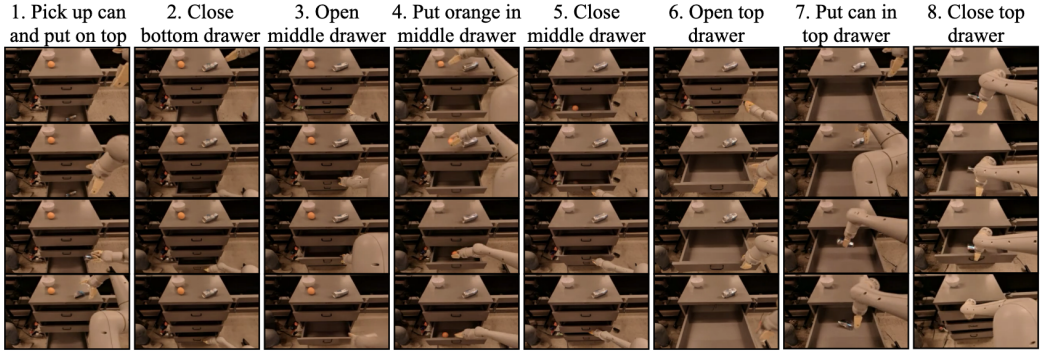


Figure 4: **Long-horizon simulations.** UniSim sequentially simulates 8 interactions autoregressively. The simulated interactions maintain temporal consistency across long-horizon interactions, correctly preserving objects and locations (can on counter in column 2-7, orange in drawer in column 4-5).

Long-Horizon Simulations

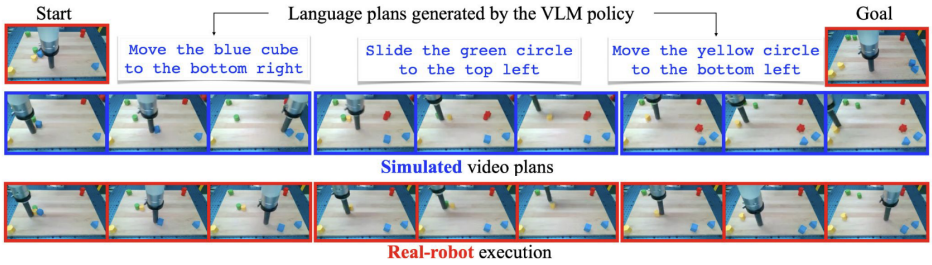


Figure: A VLM policy generating long-horizon actions, simulated video plans, and real-robot execution.

Model	RDG (moved)	RDG (all)
VLM-BC	0.11 ± 0.13	0.07 ± 0.11
Sim-Hindsight	**0.34** ± 0.13	**0.34** ± 0.13

Table: Evaluation of long-horizon actions.

Long-Horizon Simulations

- Generate long-horizon trajectories with UniSim.
- Use the final frame of the short-horizon trajectory as the initial state.
- Move some blocks, then use a VLM to generate language instructions.
- Use these instructions to generate long-horizon data.
- Train the VLM policy using the generated data.
- An inverse dynamics model produces low-level actions for robot execution.

Sim-to-Real: RL with UniSim

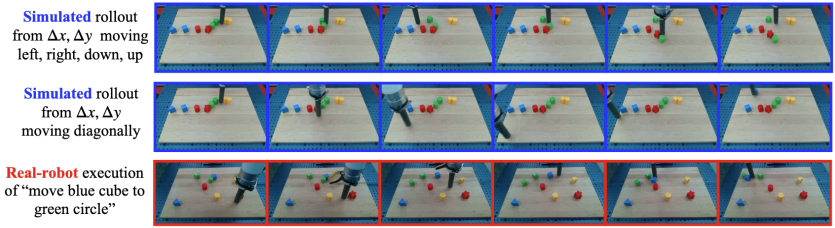


Figure 8: **[Top] Simulation from low-level controls**. UniSim supports low-level control actions as inputs to move endpoint horizontally, vertically, and diagonally. **[Bottom] Real-robot execution of an RL policy** trained in simulation and zero-shot onto the real Language Table task. The RL policy can successfully complete the task of "moving blue cube to green circle".

	Succ. rate (all)	**Succ. rate (pointing)**
VLA-BC	0.58	0.12
Simulator-RL	**0.81**	**0.71**

Table: Performance of RL policy trained with UniSim and evaluated on real-robot tasks.

Sim to Real

- UniSim is used as a realistic simulator for training RL policies.
- Compared to a baseline trained via behavior cloning, the RL policy is fine-tuned using REINFORCE on simulated rollouts (reward signals based on progress toward the goal).
- Low-level action predictions are generated from observations; the simulator produces video trajectories from which rewards are derived.
- The RL policy trained in UniSim outperforms the behavior cloning baseline and is deployed directly on the real robot.

Ablation

Condition	FID ↓	FVD ↓	IS ↑	CLIP ↑
1 frame	59.47	315.69	3.03	22.55
4 distant	34.89	237	3.43	22.62
4 recent	**34.63**	**211.3**	**3.52**	**22.63**

Table 1: **Ablations of history conditioning** using FVD, FID, and Inception score, and CLIP score on Ego4D. Conditioning on multiple frames is better than on a single frame, and recent history has an edge over distant history.

Model size	FVD ↓	CLIP ↑
500M	277.85	22.08
1.6B	224.61	22.27
5.6B	**211.30**	**22.63**

Dataset	FVD ↓	CLIP ↑
Internet only	219.62	22.27
Without internet	307.80	21.99
Universal simulator	**211.30**	**22.63**

Video examples and Failures

Unisim: **universal-simulator.github.io/unisim/**

Limitations

- Hallucinations
- Limited temporal memory only 4 frames
- struggles with out of distribution
- Not a truly universal model. only uses visual changes. not able to tell forces, touch.

Future Work

Genie: Generative Interactive Environments

Video as the New Language for Real-World Decision Making

Genie 2: A Large-Scale Foundation World Model

SORA: Video Generation Models as World Simulators

References

- Jonathan Ho et al., *Imagen Video: High Definition Video Generation with Diffusion Models*.
- Jonathan Ho et al., *Video Diffusion Models*.
- Özgün Çiçek et al., *3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation*.