ICML 2021 Tutorial
Unsupervised Learning for RL

Aravind Srinivas, Pieter Abbeel
UC Berkeley

Part 1: Representation Learning in RL

Structure

- Part 1:
  Representation Learning in RL (Aravind):
  How can you use Self(Un)-supervised Learning to improve RL

- Part 2:
  Reward-Free RL (Pieter):
  How can you train a completely unsupervised
  (or un-reinforced) agent

Why Unsupervised Learning

Why Unsupervised Learning

Please also check out Alex Graves and Marc Ranzato’s Deep Unsupervised Learning (NeurIPS 2018) tutorial

Why Unsupervised Learning

Please also check out Alex Graves and Marc Ranzato's Deep Unsupervised Learning (NeurIPS 2018) tutorial

Objective:
Introduce some of the recent UL/SSL methods and some of their RL applications to folks doing RL.

LeCake

LeCake

"If intelligence is a cake,
bulk of the cake is un(self-)supervised learning,
the icing on the cake is supervised learning, and,
the cherry on the cake is reinforcement learning."
— *Yann LeCun*

LeCake

LeCake

“If intelligence is a cake,
bulk of the cake is un(self-)supervised learning,
the icing on the cake is supervised learning, and,
the cherry on the cake is reinforcement learning.”
— *Yann LeCun*

Definitions:

1. Unsupervised Learning: Learn from *unannotated* data
2. Supervised Learning: Learn from *annotations* (human labels)
3. Reinforcement Learning: Learn from *reward* signals

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Learning is purely based on *extrinsic* reward optimization

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Examples: DQN, AlphaGo, Alphazero, OpenAI Five (Dota), AlphaStar

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Learning is purely based on *intrinsic* reward optimization

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Examples: Count-Based Exploration (DeepMind, Bellemare),
Curiosity (Pathak, Efros), Random Network Distillation (OpenAI), Go-Explore (Uber AI)

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Learning is purely based on predicting the output from an input using *labeled (annotated)* data

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Examples: AlexNet, VGG, ResNet, Neural Image Captioning, seq2seq (Transformers), DeepSpeech, WaveNet, AlphaFold, DALL-E, ……

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Learn representations, or world models, or generative models, from unlabeled (un-annotated or rewarded) data.

Table from Alex Graves

Forms of Learning

LeCake

	With teacher	Without teacher
Active	Reinforcement Learning	Intrinsic Motivation (Exploration)
Passive	Supervised Learning	Self-(un)supervised Learning

Examples: PixelCNN, GANs, VAEs, GPT-1,2,3; BERT, T5, Electra, iGPT, Contrastive Predictive Coding (CPC), Momentum Contrast (MoCo), AMDIM, CMC, SimCLR, BYOL, SimSiam, DINO, Barlow Twins, ...

Table from Alex Graves

LeCake

LeCake

"If intelligence is a cake,
bulk of the cake is un(self-)supervised learning,
the icing on the cake is supervised learning, and,
the cherry on the cake is reinforcement learning."
— *Yann LeCun*

Bits Argument:

1. Unsupervised Learning: Predict missing from given (*millions* of bits)
2. Supervised Learning: Predict human-annotations (*thousands* of bits)
3. Reinforcement Learning: Predict scalar rewards (*fewer* bits)

Practical arguments in favor of unsupervised learning

LeCake

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

2. Some domains require significant expertise in annotation (medicine, law, ethics, science, biology, etc.)

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

2. Some domains require significant expertise in annotation (medicine, law, ethics, science, biology, etc.)

3. Reward annotation is even trickier - sparse or dense, what's the right UI for scalable annotation of rewards by humans, should you use a continuous-valued reward or categorical (good/bad/neural), etc.

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

2. Some domains require significant expertise in annotation (medicine, law, ethics, science, biology, etc.)

3. Reward annotation is even trickier - sparse or dense, what's the right UI for scalable annotation of rewards by humans, should you use a continuous-valued reward or categorical (good/bad/neural), etc.

4. Collecting human demos for behavior cloning is challenging at scale

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

2. Some domains require significant expertise in annotation (medicine, law, ethics, science, biology, etc.)

3. Reward annotation is even trickier - sparse or dense, what's the right UI for scalable annotation of rewards by humans, should you use a continuous-valued reward or categorical (good/bad/neural), etc.

4. Collecting human demos for behavior cloning is challenging at scale

5. Behavior cloning for real world problems prone to distribution mismatch and compounding errors without sufficient data

Practical arguments in favor of unsupervised learning

LeCake

1. Label creation, annotation and maintenance is time-consuming and a challenging discipline in itself.

2. Some domains require significant expertise in annotation (medicine, law, ethics, science, biology, etc.)

3. Reward annotation is even trickier - sparse or dense, what's the right UI for scalable annotation of rewards by humans, should you use a continuous-valued reward or categorical (good/bad/neural), etc.

4. Collecting human demos for behavior cloning is challenging at scale

5. Behavior cloning for real world problems prone to distribution mismatch and compounding errors without sufficient data

6. Good (useful) behavioral data *even without* annotations is quite challenging to collect.

Philosophical arguments in favor of unsupervised learning

LeCake

Philosophical arguments in favor of unsupervised learning

LeCake

1. Inspired by how human infants learn

Philosophical arguments in favor of unsupervised learning

LeCake

1. Inspired by how human infants learn

2. Feels more *human-like*

Philosophical arguments in favor of unsupervised learning

LeCake

1. Inspired by how human infants learn

2. Feels more *human-like*

3. Build mental models of the world (Kenneth Craik) - "*mind is a predictive modelling engine*", "*organism carries a 'small-scale model' of external reality and of its own possible actions within its head*"

Philosophical arguments in favor of unsupervised learning

LeCake

1. Inspired by how human infants learn

2. Feels more *human-like*

3. Build mental models of the world (Kenneth Craik) - "*mind is a predictive modelling engine*", "*organism carries a 'small-scale model' of external reality and of its own possible actions within its head*"

4. Learn skills, not tasks (Satinder Singh)

Philosophical arguments in favor of unsupervised learning

LeCake

1. Inspired by how human infants learn

2. Feels more *human-like*

3. Build mental models of the world (Kenneth Craik) - “*mind is a predictive modelling engine*”, “*organism carries a ‘small-scale model’ of external reality and of its own possible actions within its head*”

4. Learn skills, not tasks (Satinder Singh)

5. Build a *general-purpose* understanding of the world (representations, world models, common sense, etc etc) to be able to **generalize** well to new scenarios and learn fast (**transfer**)

Bits Argument

LeCake

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits
3. Unsupervised: 256x256x3x8 ~ 2000 Gbits, 150000x more.

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits
3. Unsupervised: 256x256x3x8 ~ 2000 Gbits, 150000x more.
4. Flaw in the argument: Not all bits are equal, a 8-bit color-channel information conveys much less than a human-language aligned categorical classification bit.

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits
3. Unsupervised: 256x256x3x8 ~ 2000 Gbits, 150000x more.
4. Flaw in the argument: Not all bits are equal, a 8-bit color-channel information conveys much less than a human-language aligned categorical classification bit.
5. Somehow, *intrinsically capture* the bits *that matter* for *downstream tasks* and *rely less* on human-provided bits

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits
3. Unsupervised: 256x256x3x8 ~ 2000 Gbits, 150000x more.
4. Flaw in the argument: Not all bits are equal, a 8-bit color-channel information conveys much less than a human-language aligned categorical classification bit.
5. Somehow, *intrinsically capture* the bits *that matter* for *downstream tasks* and *rely less* on human-provided bits
6. Use self-supervised learning to improve the label and reward efficiency of supervised and reinforcement learning systems

Bits Argument

LeCake

1. Consider ImageNet (1.28M images) with 1000 categories, 256x256.
2. Supervised: log2(1000) *1.28M ~ 12.8 Mbits
3. Unsupervised: 256x256x3x8 ~ 2000 Gbits, 150000x more.
4. Flaw in the argument: Not all bits are equal, a 8-bit color-channel information conveys much less than a human-language aligned categorical classification bit.
5. Somehow, *intrinsically capture* the bits *that matter* for *downstream tasks* and *rely less* on human-provided bits
6. Use self-supervised learning to improve the label and reward efficiency of supervised and reinforcement learning systems

Focus of this tutorial

Forms of Unsupervised Learning

Forms of Unsupervised Learning

Generative

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Density Modeling

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Density Modeling

(PixelCNN, GPT-x,
Bigan (implicit))

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Density Modeling

(PixelCNN, GPT-x,
Bigan (implicit))

Masked Auto-Encoding

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Density Modeling

(PixelCNN, GPT-x, Bigan (implicit))

Masked Auto-Encoding

(BERT, Electra)

Forms of Unsupervised Learning

Generative

Non-Generative (Contrastive-like)

Density Modeling

(PixelCNN, GPT-x, Bigan (implicit))

Masked Auto-Encoding

(BERT, Electra)

Siamese Networks

Forms of Unsupervised Learning

Generative

- Density Modeling
  (PixelCNN, GPT-x, Bigan (implicit))
- Masked Auto-Encoding
  (BERT, Electra)

Non-Generative (Contrastive-like)

- Siamese Networks
  (SimCLR, MoCo, AMDIM, BYOL, SimSiam, DINO, Barlow Twins …)

Forms of Unsupervised Learning

- Generative
  - Density Modeling
    (PixelCNN, GPT-x, Bigan (implicit))
  - Masked Auto-Encoding
    (BERT, Electra)
- Non-Generative (Contrastive-like)
  - Siamese Networks
    (SimCLR, MoCo, AMDIM, BYOL, SimSiam, DINO, Barlow Twins …)
  - Contrastive Prediction
    (CPC)

How best to use UL for RL?

How best to use UL for RL?

Generative

Non-Generative (Contrastive-like)

How best to use UL for RL?

Generative

Non-Generative (Contrastive-like)

Learn world models;
Dyna (fake rollouts),
Model-Based RL

How best to use UL for RL?

Generative

Non-Generative (Contrastive-like)

Learn world models;
Dyna (fake rollouts),
Model-Based RL

Use as an auxiliary task to speed up (sample-efficiency) RL

How best to use UL for RL?

Generative

Non-Generative (Contrastive-like)

Learn world models;
Dyna (fake rollouts),
Model-Based RL

Use as an auxiliary task to speed up (sample-efficiency) RL

Beyond the scope of this tutorial;
but will highlight some representative work,
Refer to Mordatch/Hamrick Tutorial (ICML 20')

How best to use UL for RL?

Generative

Non-Generative (Contrastive-like)

Learn world models;
Dyna (fake rollouts),
Model-Based RL

Use as an auxiliary task to speed up (sample-efficiency) RL

Beyond the scope of this tutorial;
but will highlight some representative work,
Refer to Mordatch/Hamrick Tutorial (ICML 20')

Focus of this tutorial

Why use UL as an auxiliary task to speed up RL?

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

3. Hope: Representations learned with UL/SSL **help** for the RL task(s)

Why use UL as an auxiliary task to speed up RL?

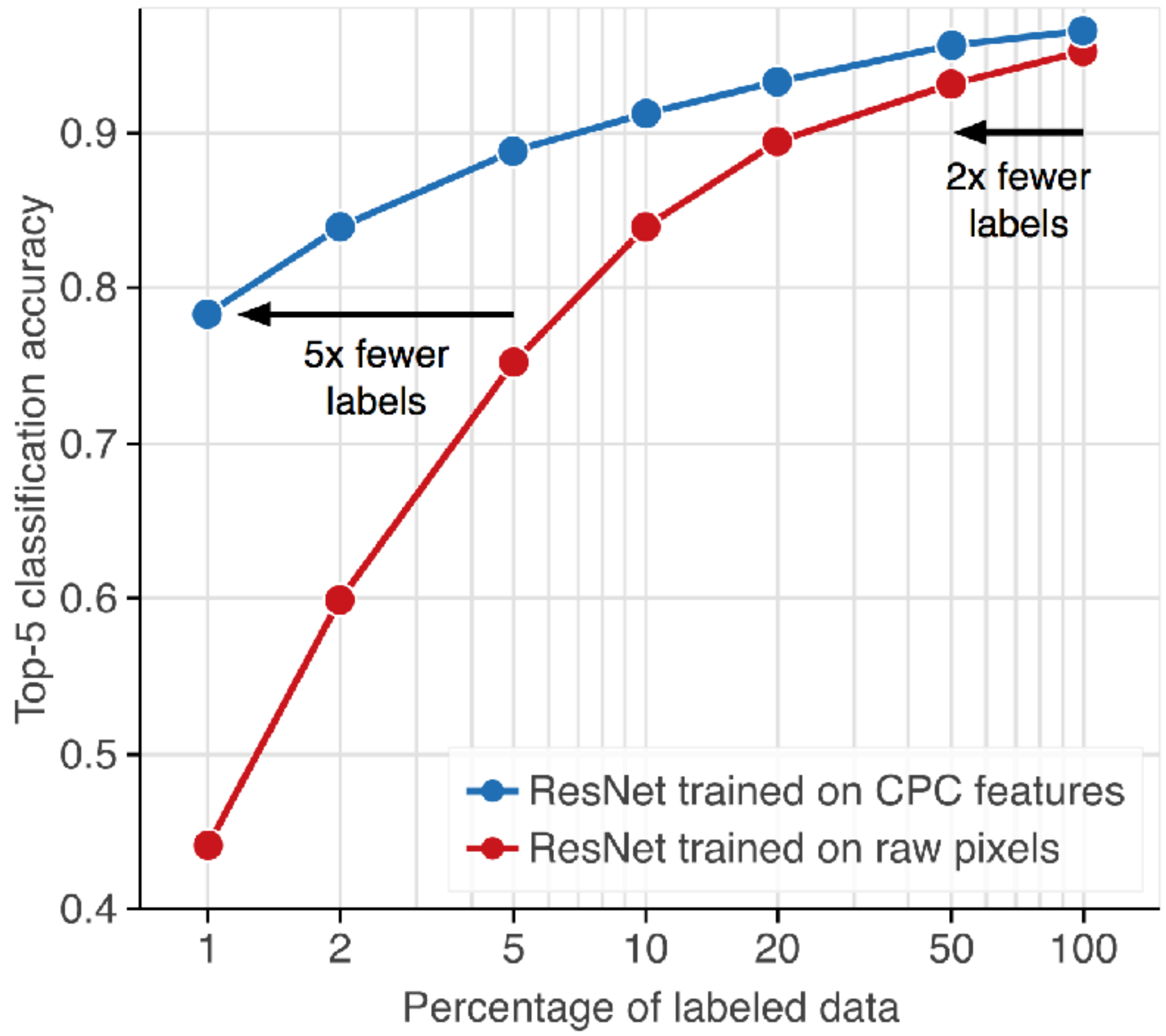
1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

3. Hope: Representations learned with UL/SSL **help** for the RL task(s)

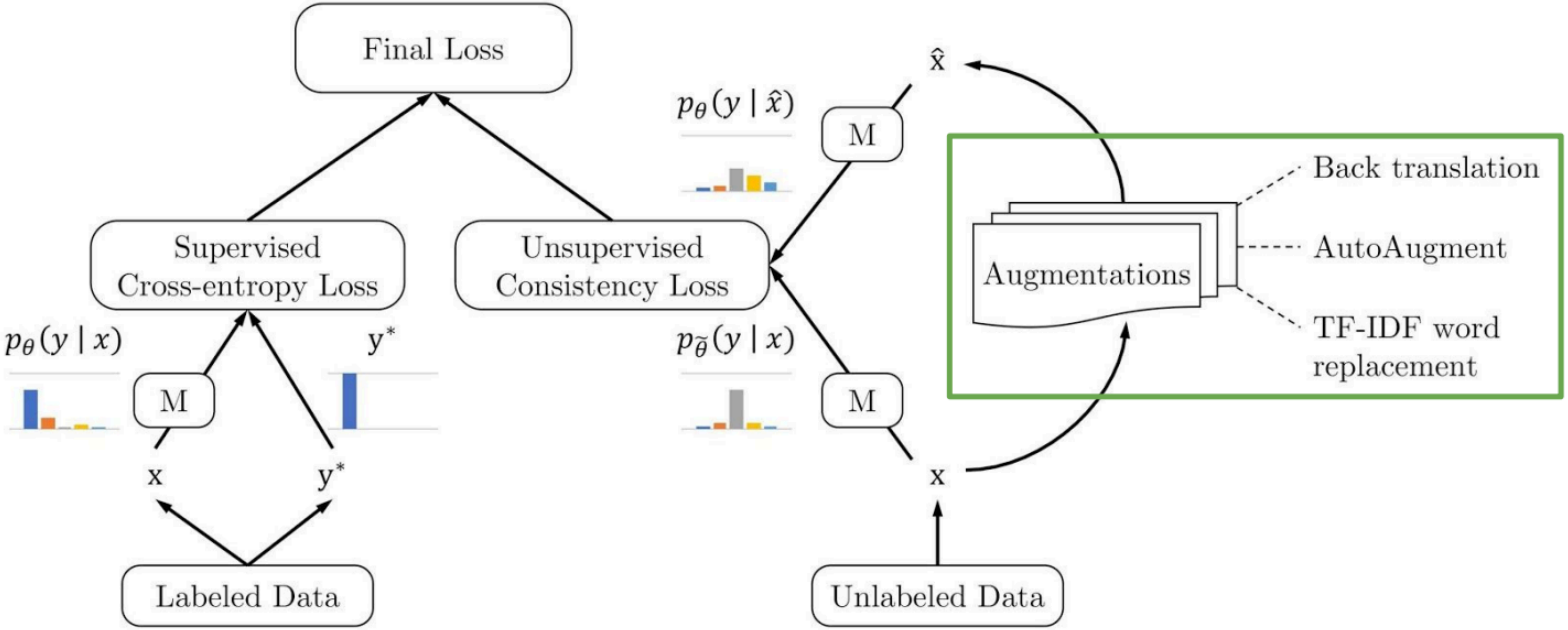
4. Inspired by: **Success** of **Un/Self/Semi**-Supervised Learning for **label-efficient** Supervised Learning

Inspiration from success in supervised learning



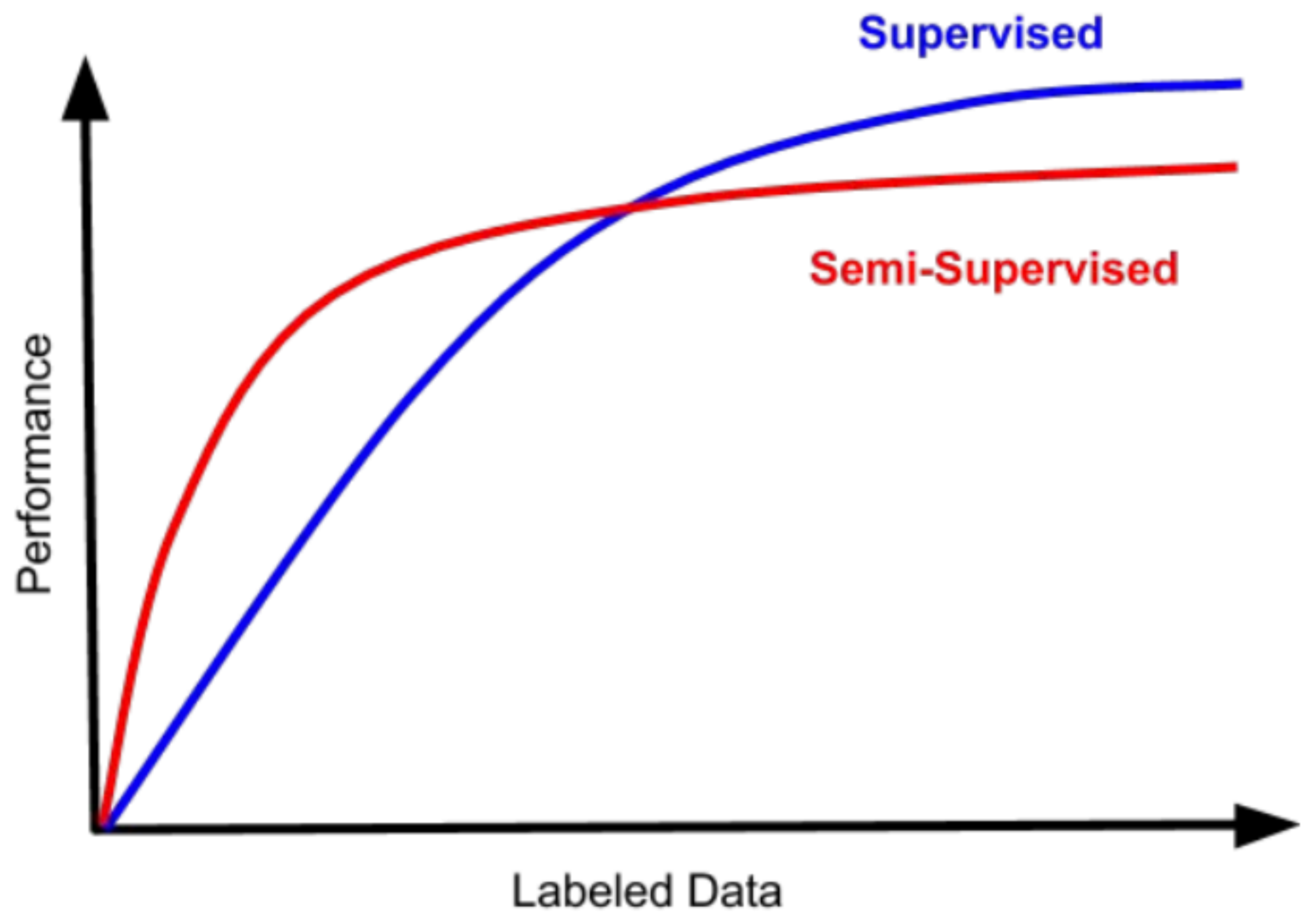
Data-Efficient Image Recognition using Contrastive Predictive Coding (Henaff et al ICML 2020)

Inspiration from success in supervised learning

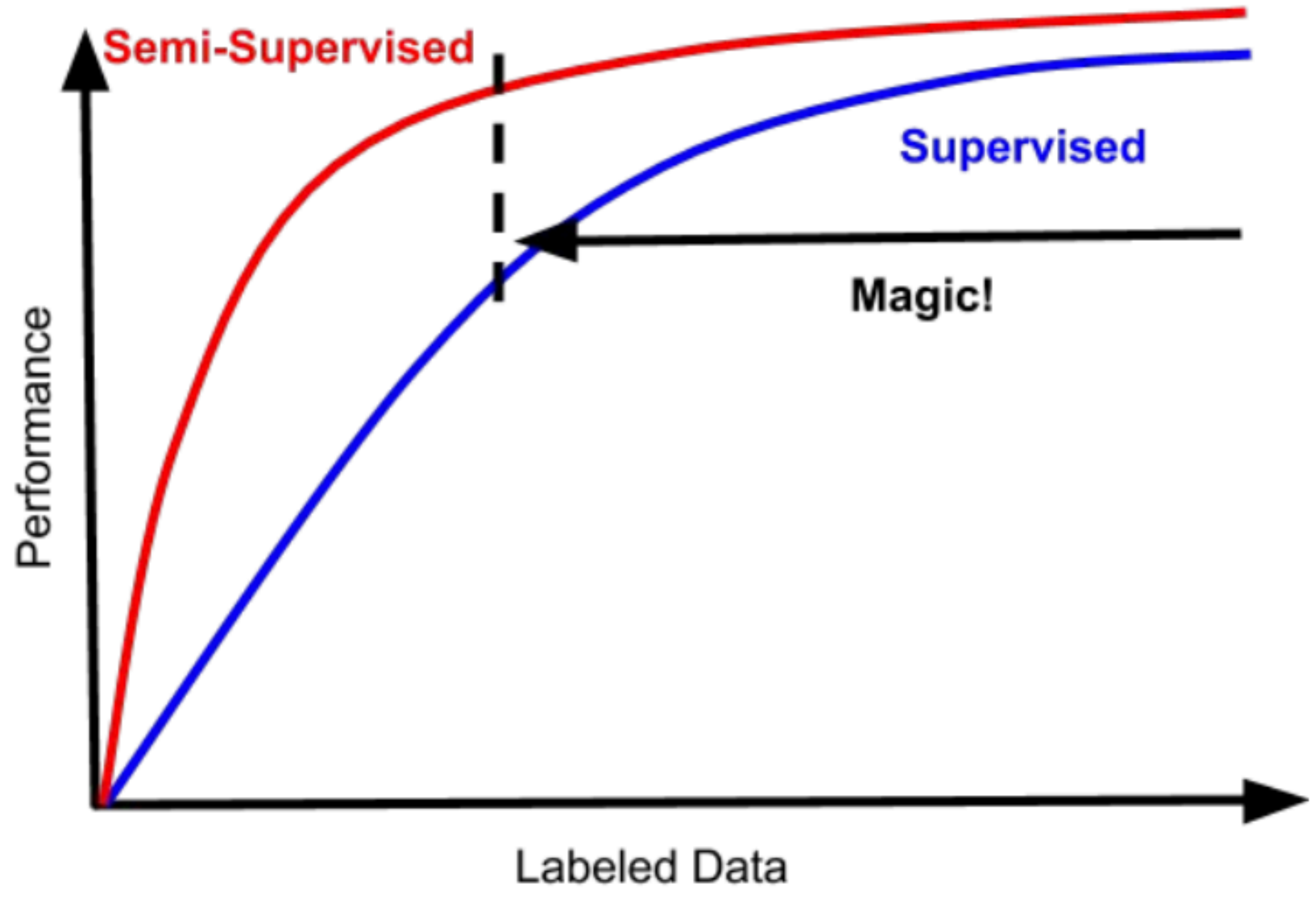


Unsupervised Data Augmentation (Xie et al 2019), figure from Thang Luong.

Inspiration from success in supervised learning



Belief of many ML practitioners



Belief of many SSL researchers

Figure credit: Vincent Vanhoucke

Unsupervised Data Augmentation (Xie et al 2019), figure from Thang Luong.

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

3. Hope: Representations learned with UL/SSL **help** for the RL task(s)

4. Inspired by: **Success** of **Un/Self/Semi**-Supervised Learning for **label-efficient** Supervised Learning

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

3. Hope: Representations learned with UL/SSL **help** for the RL task(s)

4. Inspired by: **Success** of **Un/Self/Semi**-Supervised Learning for **label-efficient** Supervised Learning

5. Challenge: What is the right kind of UL objective that will work well in tandem with RL? How to **capture** the **useful** aspects of a high dimensional sensory stream?

Why use UL as an auxiliary task to speed up RL?

1. **Simplest** way to combine UL/SSL and RL

2. Maximize reward, and learn about the world, using the **same shared network** - design choices: how many parameters for UL, how many for RL, etc. but simplest way is to just use all for both.

3. Hope: Representations learned with UL/SSL **help** for the RL task(s)

4. Inspired by: **Success** of **Un/Self/Semi**-Supervised Learning for **label-efficient** Supervised Learning

5. Challenge: What is the right kind of UL objective that will work well in tandem with RL? How to **capture** the **useful** aspects of a high dimensional sensory stream?

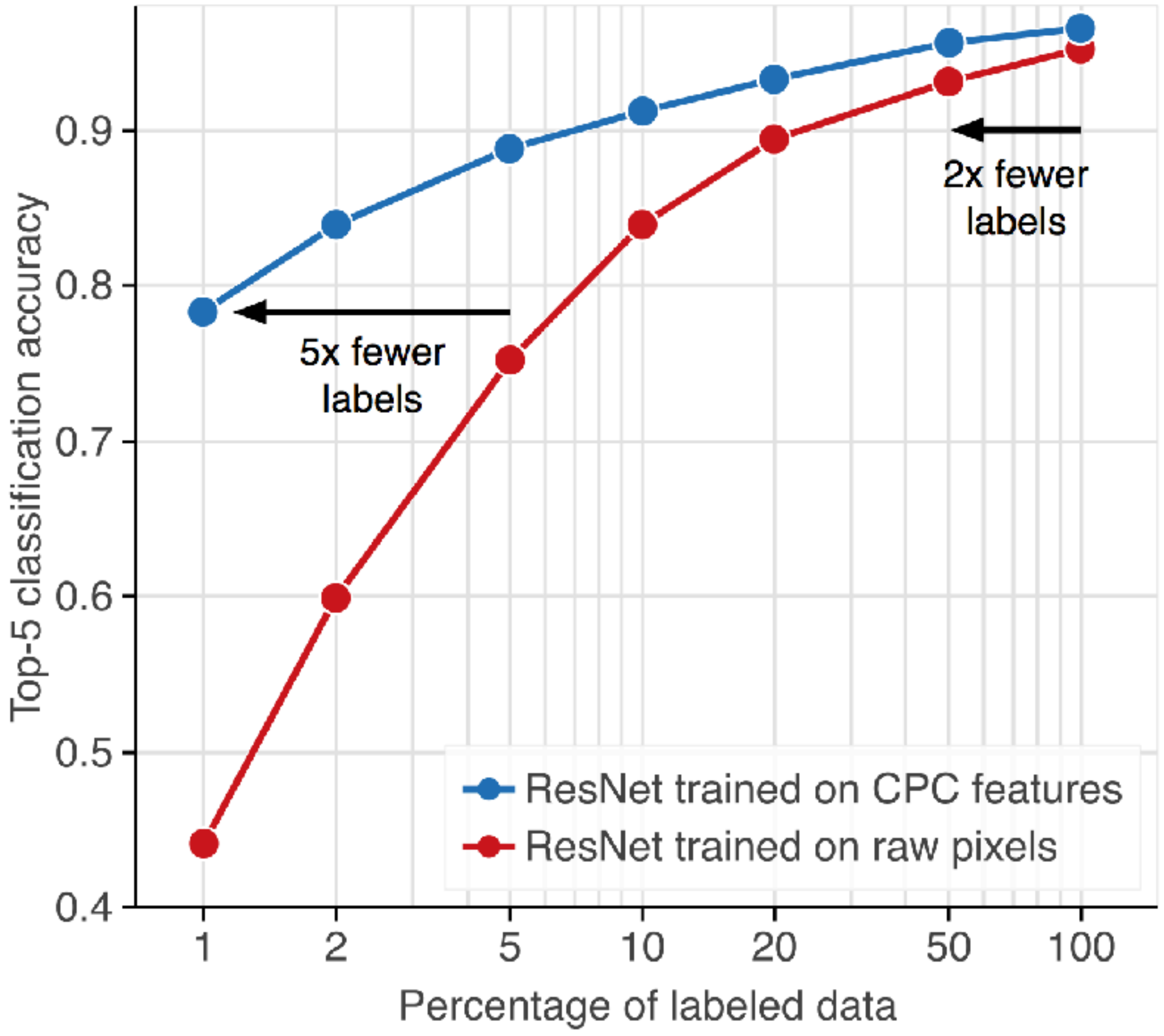
… we lived our lives under the constantly changing sky without sparing it a glance or thought. And why indeed should we? If the various formations had some meaning, if, for example, there had been some concealed signs and messages for us which it was important to decode correctly, unceasing attention to what was happening would have been inescapable

- Karl Ove Knausgaard, A Death in the Family

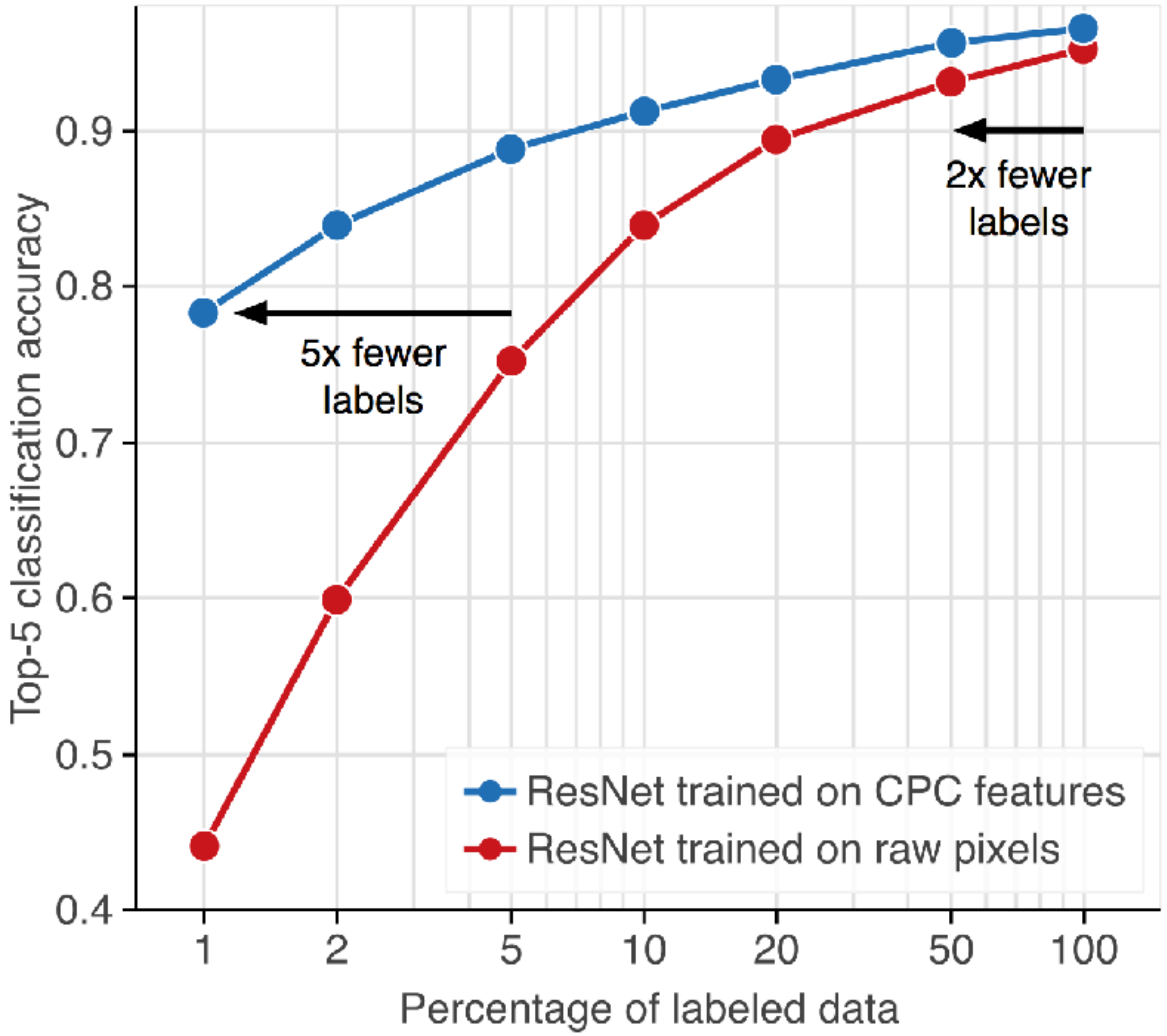
(From Alex Graves' tutorial)

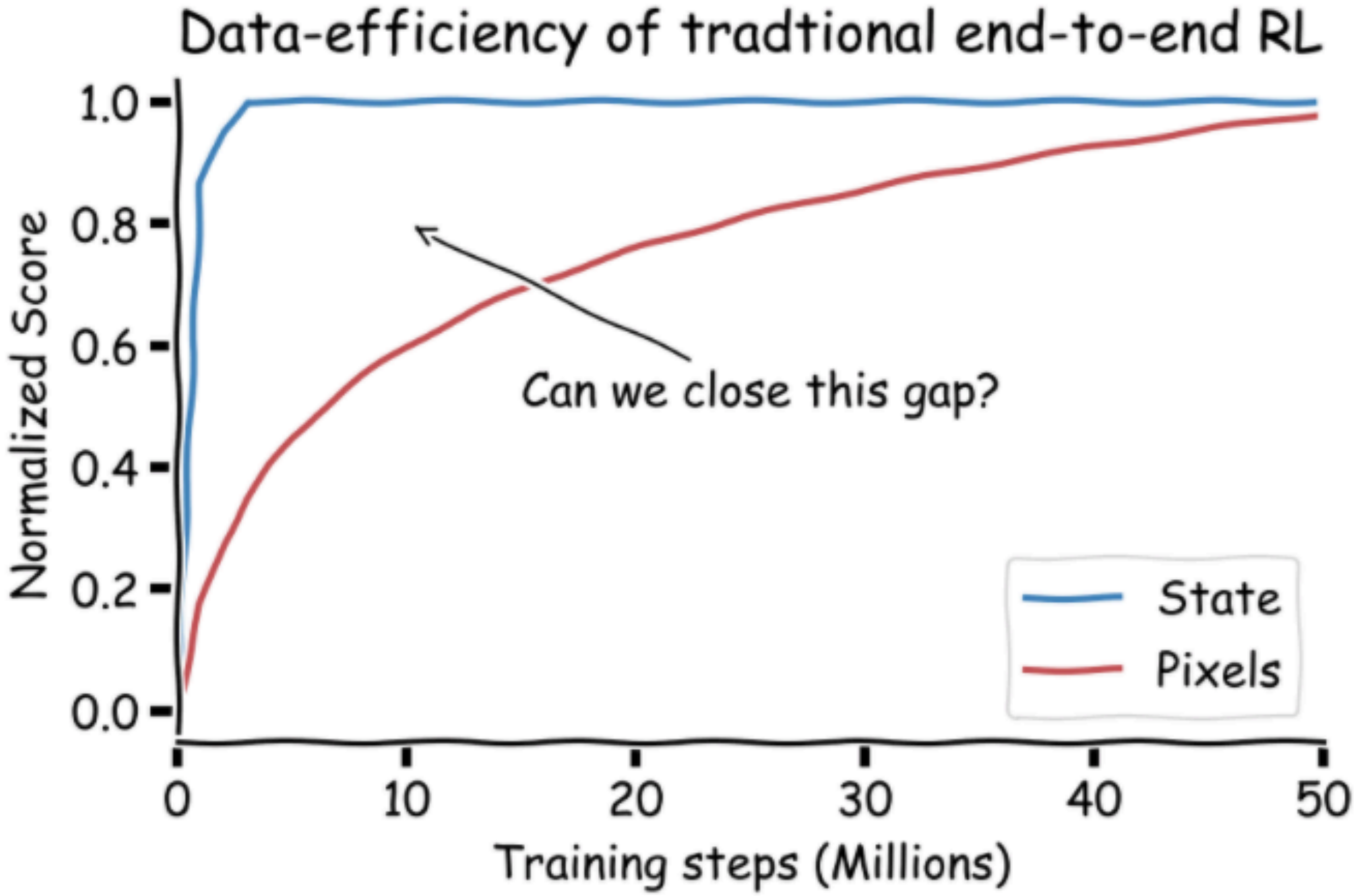
Capture useful aspects of high dimensional sensory stream

Capture useful aspects of high dimensional sensory stream



Capture useful aspects of high dimensional sensory stream





Quick background

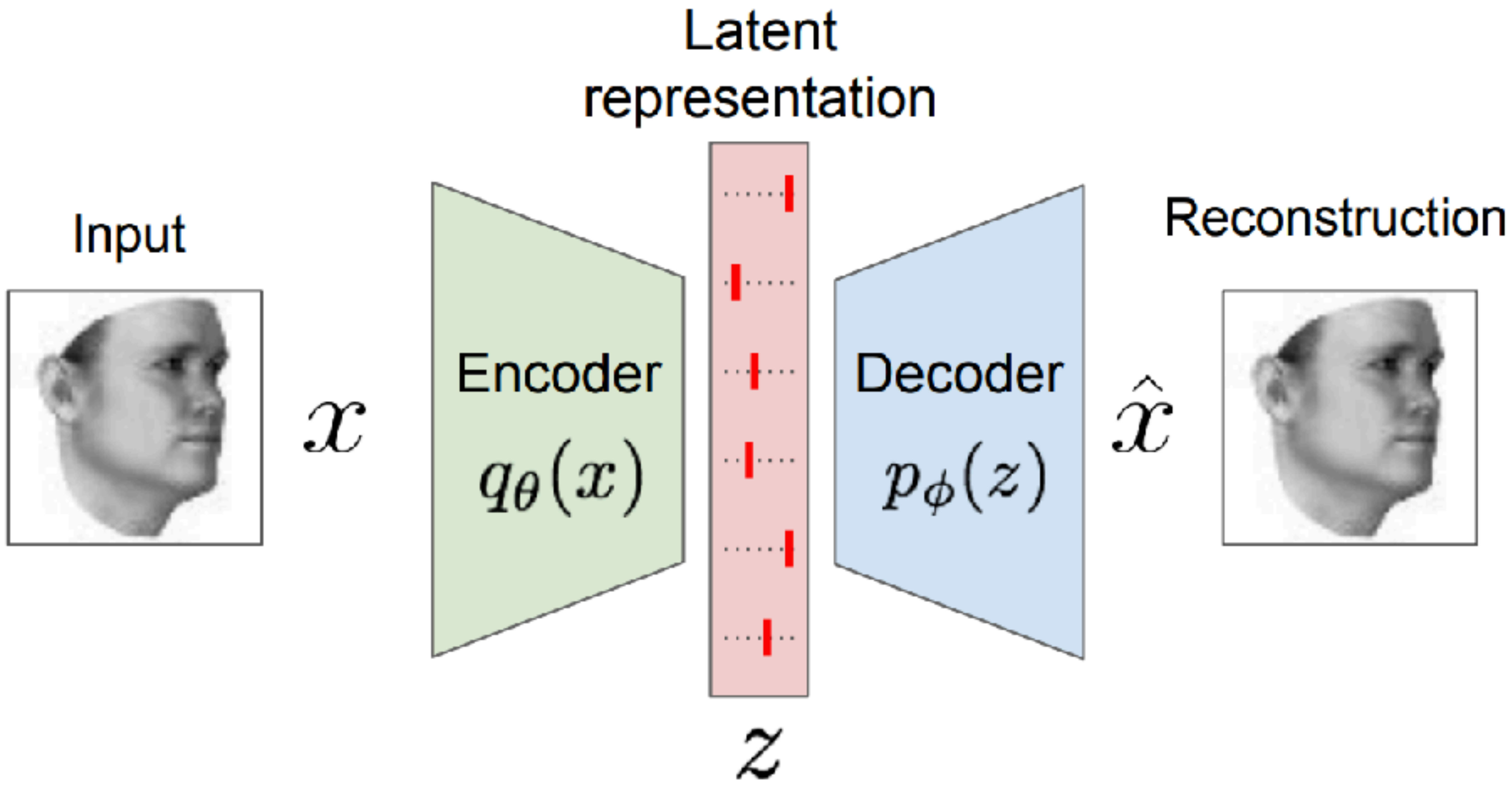
1. Autoencoder
2. Variational Autoencoder
3. Contrastive Learning
4. Siamese Networks
5. Data-Augmentations

Quick background

1. Autoencoder
2. Variational Autoencoder
3. Contrastive Learning
4. Siamese Networks
5. Data-Augmentations

Generative UL

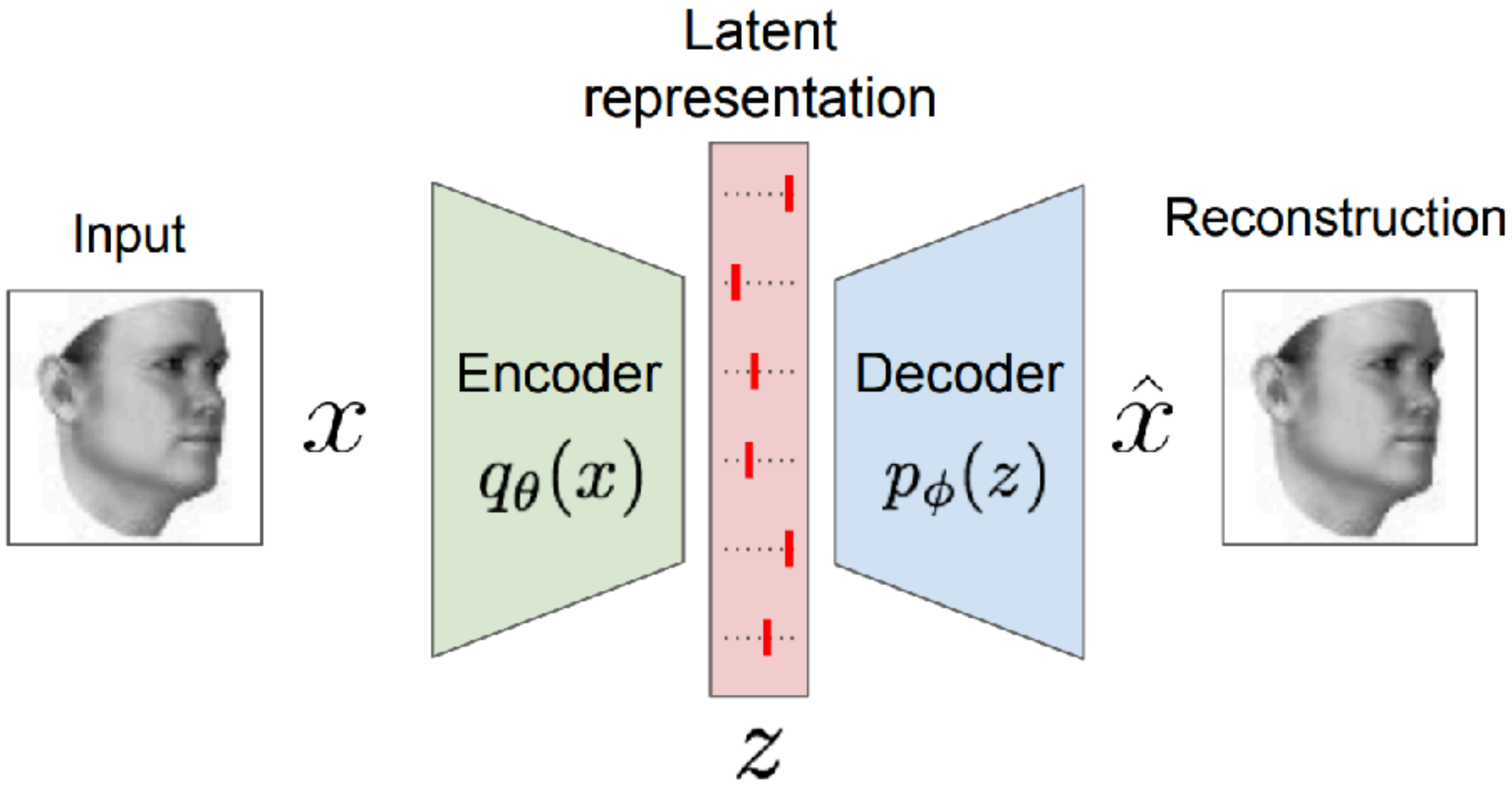
AutoEncoder



$$\mathcal{L}^{AE}(\mathbf{x};\theta,\phi) = \underbrace{[\mathbf{x} - p_\theta(q_\phi(\mathbf{x})]^2}_{\text{Reconstruction cost}}$$

Slide from Alex Graves

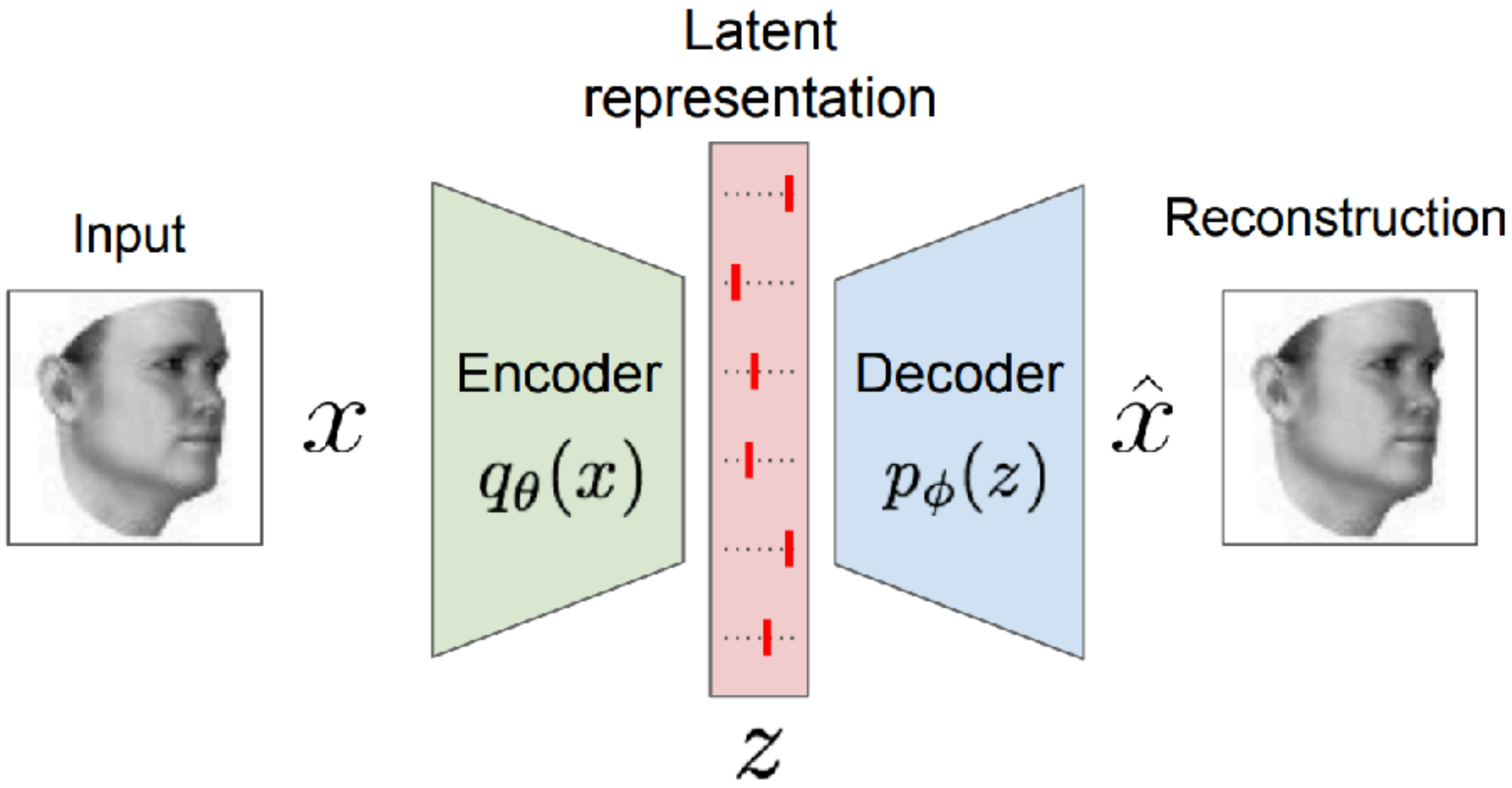
AutoEncoder



$$\mathcal{L}^{AE}(\mathbf{x};\theta,\phi) = \cancel{\left[\mathbf{x} - p_\theta(q_\phi(\mathbf{x}))\right]^2} \underbrace{- \log p_\theta(q_\phi(\mathbf{x}))}_{\text{Reconstruction cost}}$$

Slide from Alex Graves

AutoEncoder

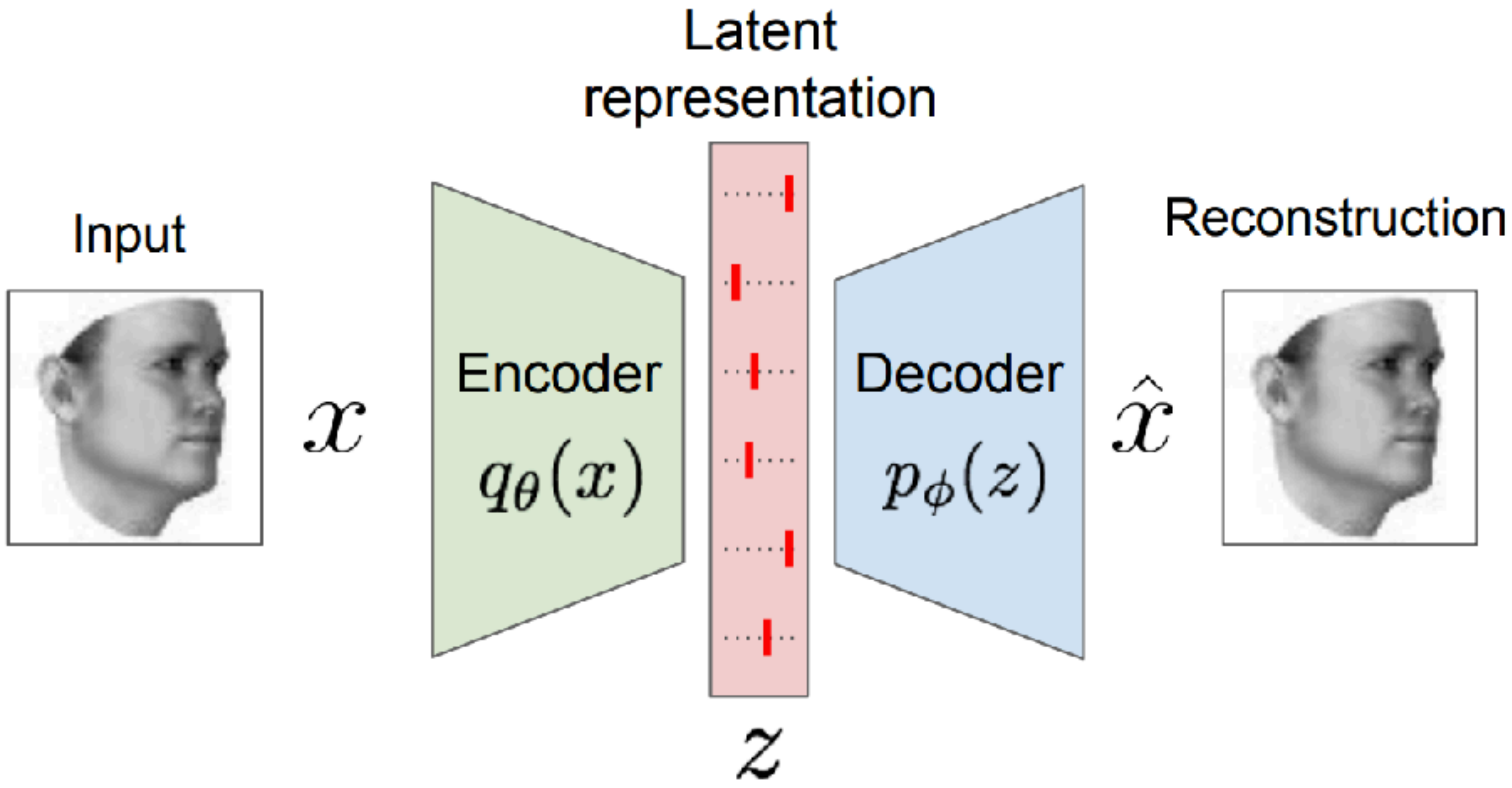


$$\mathcal{L}^{AE}(\mathbf{x};\theta,\phi) = \cancel{\left[\mathbf{x} - p_\theta(q_\phi(\mathbf{x}))\right]^2} \underbrace{- \log p_\theta(q_\phi(\mathbf{x}))}_{\text{Reconstruction cost}}$$

A lossy (bottleneck) representation of the input - but packs as much information as possible

Slide from Alex Graves

AutoEncoder



$$\mathcal{L}^{AE}(\mathbf{x};\theta,\phi) = \cancel{\left[\mathbf{x} - p_\theta(q_\phi(\mathbf{x}))\right]^2} \underbrace{- \log p_\theta(q_\phi(\mathbf{x}))}_{\text{Reconstruction cost}}$$

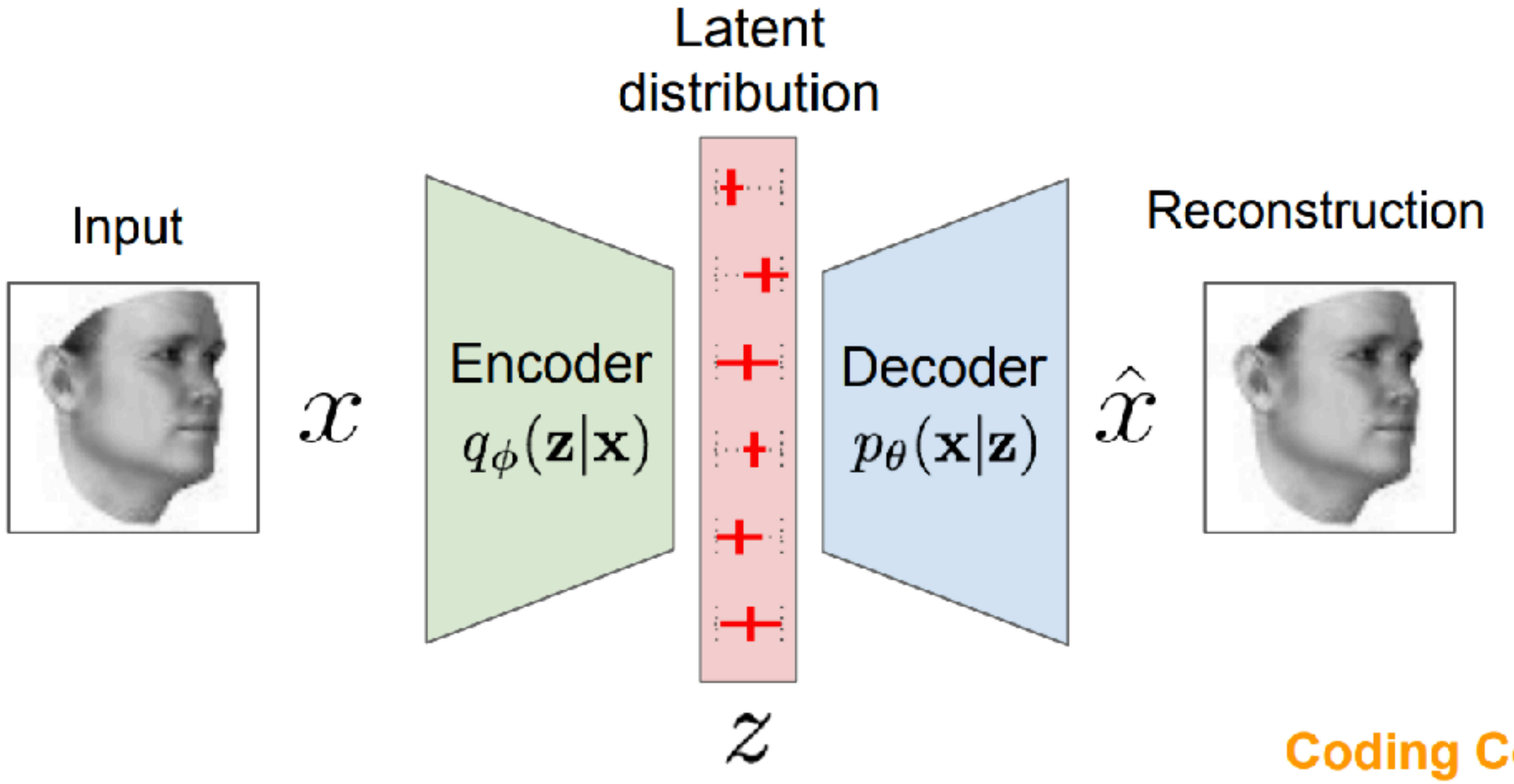
A lossy (bottleneck) representation of the input - but packs as much information as possible

Adding some structure to the latent (bottleneck) can help add more semantic meaning

Slide from Alex Graves

Variational AutoEncoder (VAE)

Kingma et al, 2014
Rezende et al, 2014

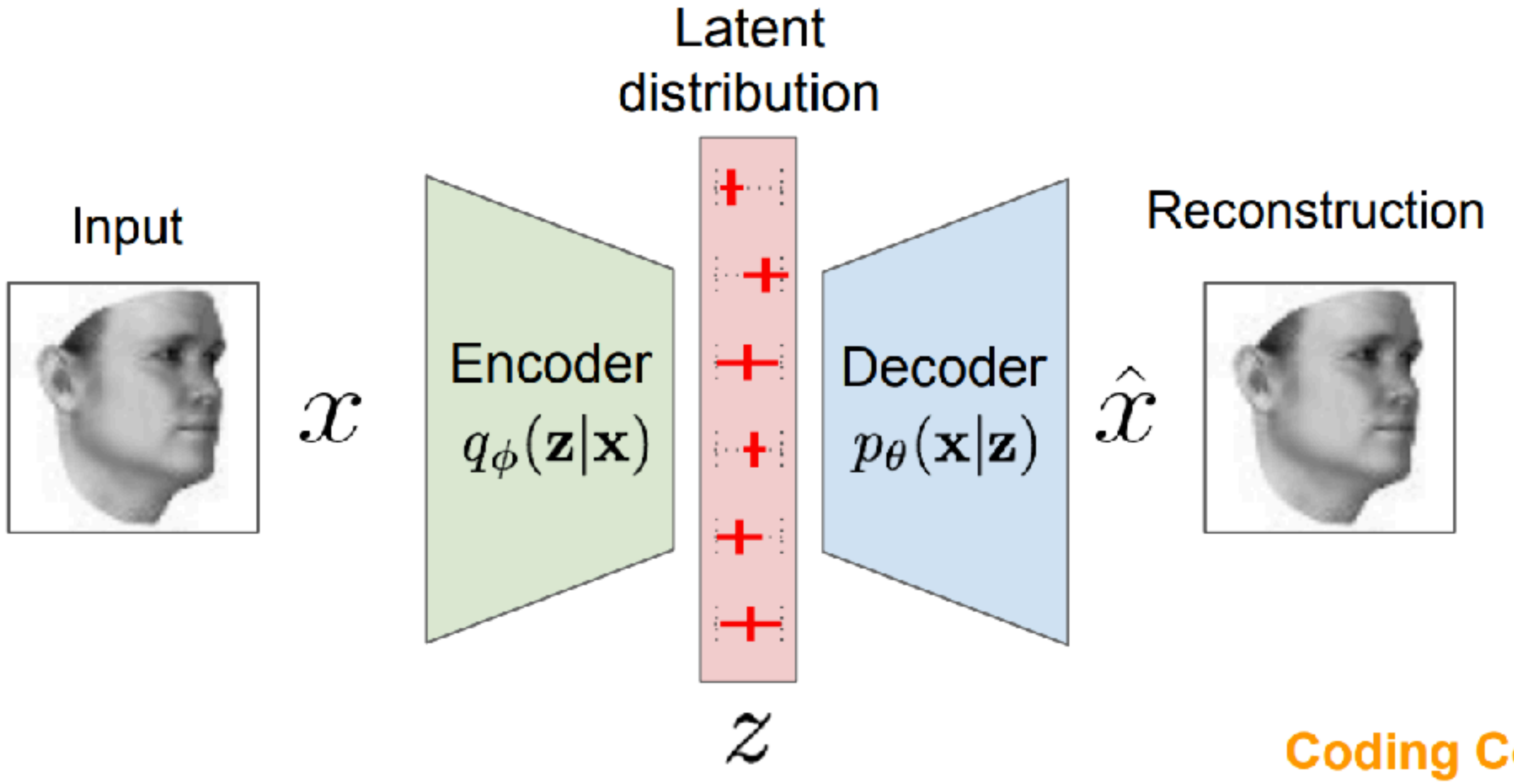


$$\mathcal{L}_{VAE}(\mathbf{x};\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log p_\theta(\mathbf{x}|\mathbf{z})\right]}_{\text{Reconstruction cost}} + \overbrace{KL\big(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big)}^{\text{Coding Cost}}$$

Slide from Alex Graves

Variational AutoEncoder (VAE)

Kingma et al, 2014
Rezende et al, 2014



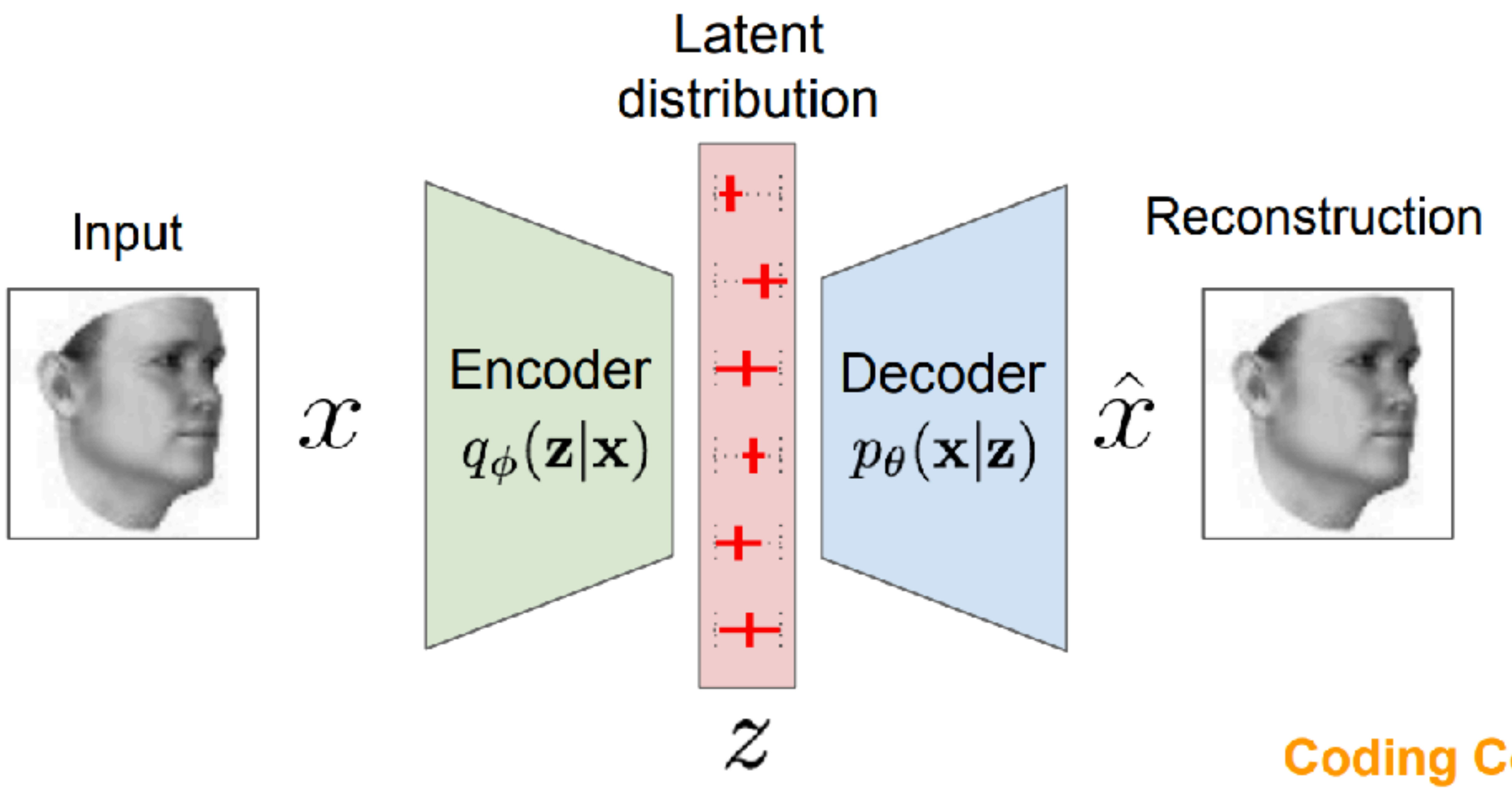
$$\mathcal{L}_{VAE}(\mathbf{x};\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log p_\theta(\mathbf{x}|\mathbf{z})\right]}_{\text{Reconstruction cost}} + \overbrace{KL\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)}^{\text{Coding Cost}}$$

Make sure there's an additional penalty for the latents (posterior) to match a prior

Slide from Alex Graves

Variational AutoEncoder (VAE)

Kingma et al, 2014
Rezende et al, 2014



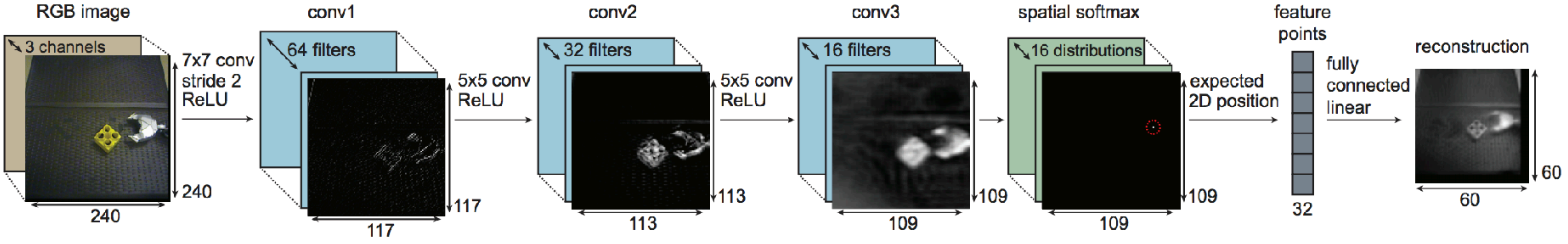
$$\mathcal{L}_{VAE}(\mathbf{x};\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log p_\theta(\mathbf{x}|\mathbf{z})\right]}_{\text{Reconstruction cost}} + \overbrace{KL\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)}^{\text{Coding Cost}}$$

Make sure there's an additional penalty for the latents (posterior) to match a prior

reconstruction pathway goes through a cost for how much information it can pack in; also has to match the prior

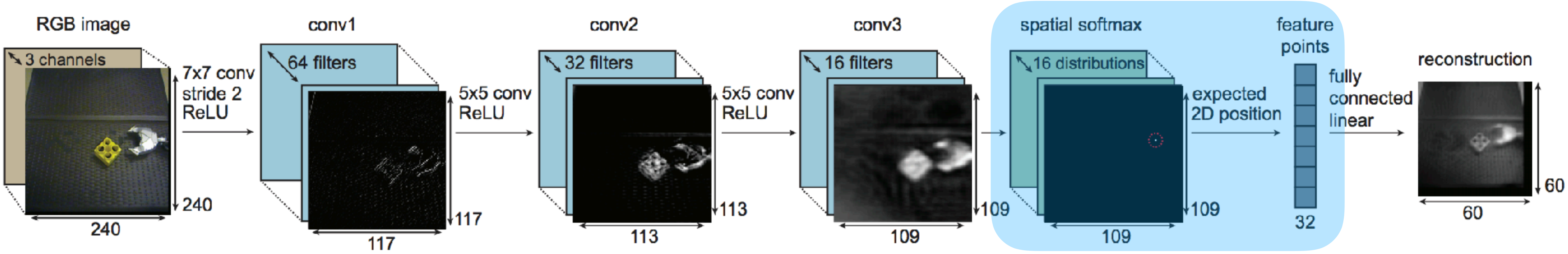
Slide from Alex Graves

Spatial AutoEncoder



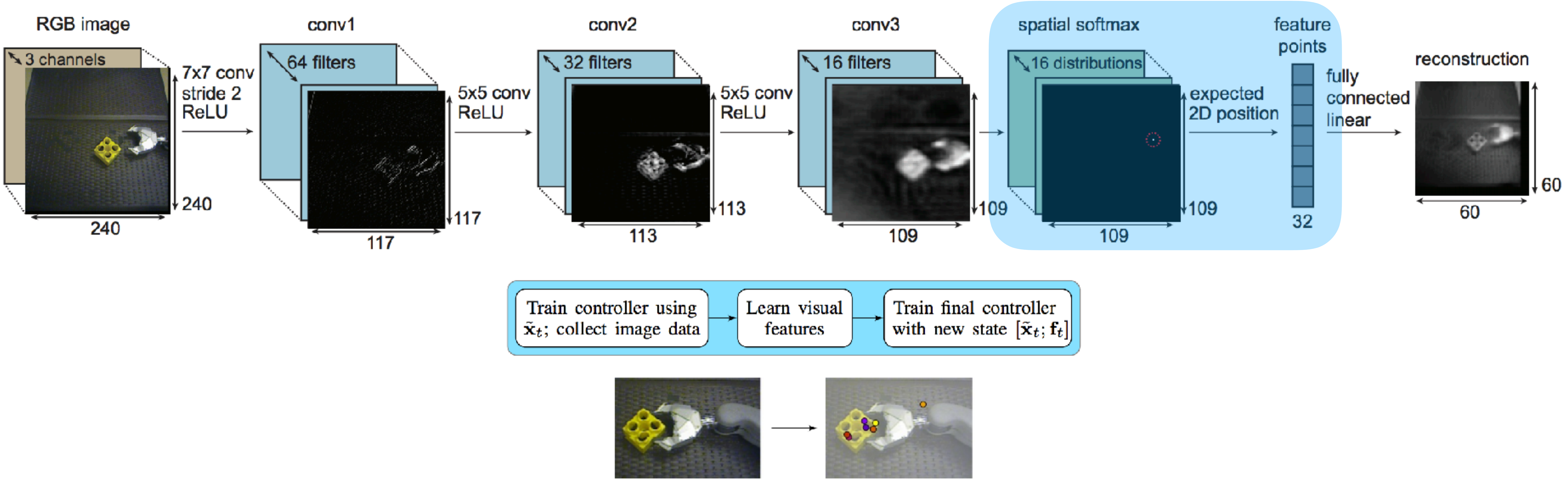
Deep Spatial Autoencoders for Visuomotor Learning (Finn et al 2015)

Spatial AutoEncoder



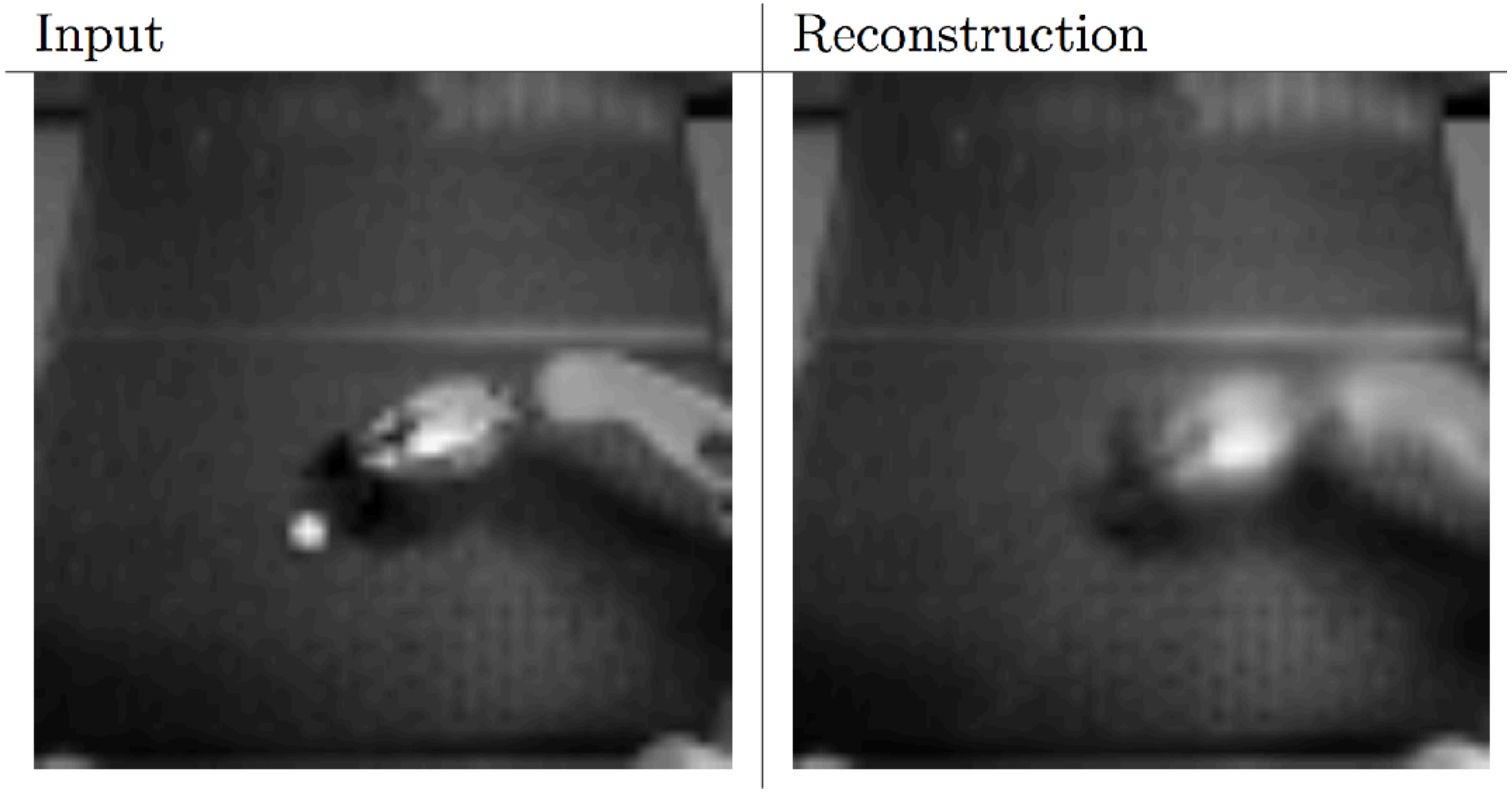
Deep Spatial Autoencoders for Visuomotor Learning (Finn et al 2015)

Spatial AutoEncoder



Deep Spatial Autoencoders for Visuomotor Learning (Finn et al 2015)

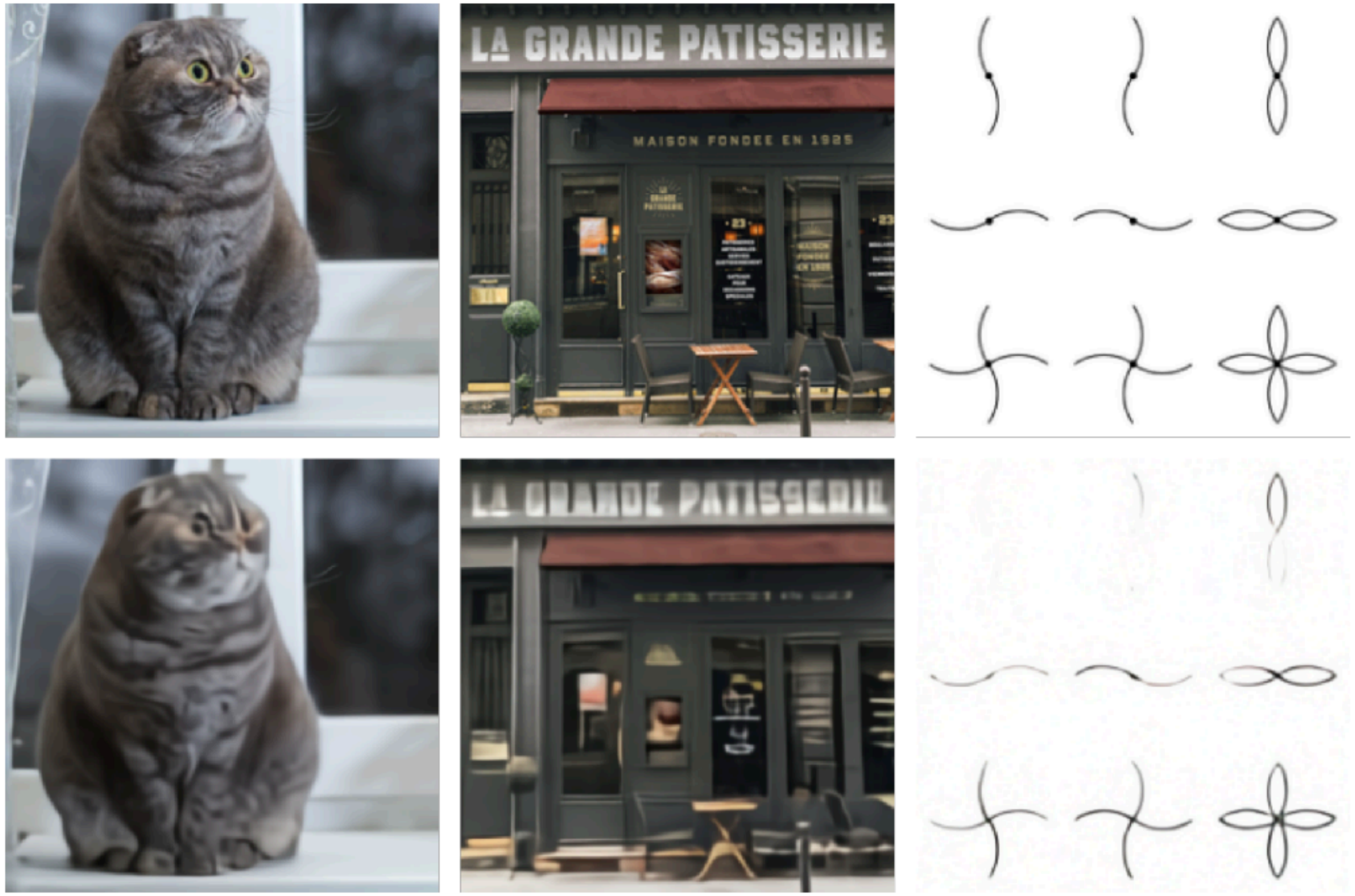
AutoEncoders can ignore relevant features for a task



Reconstruction error isn't significantly altered by presence or absence of the ping-pong ball.

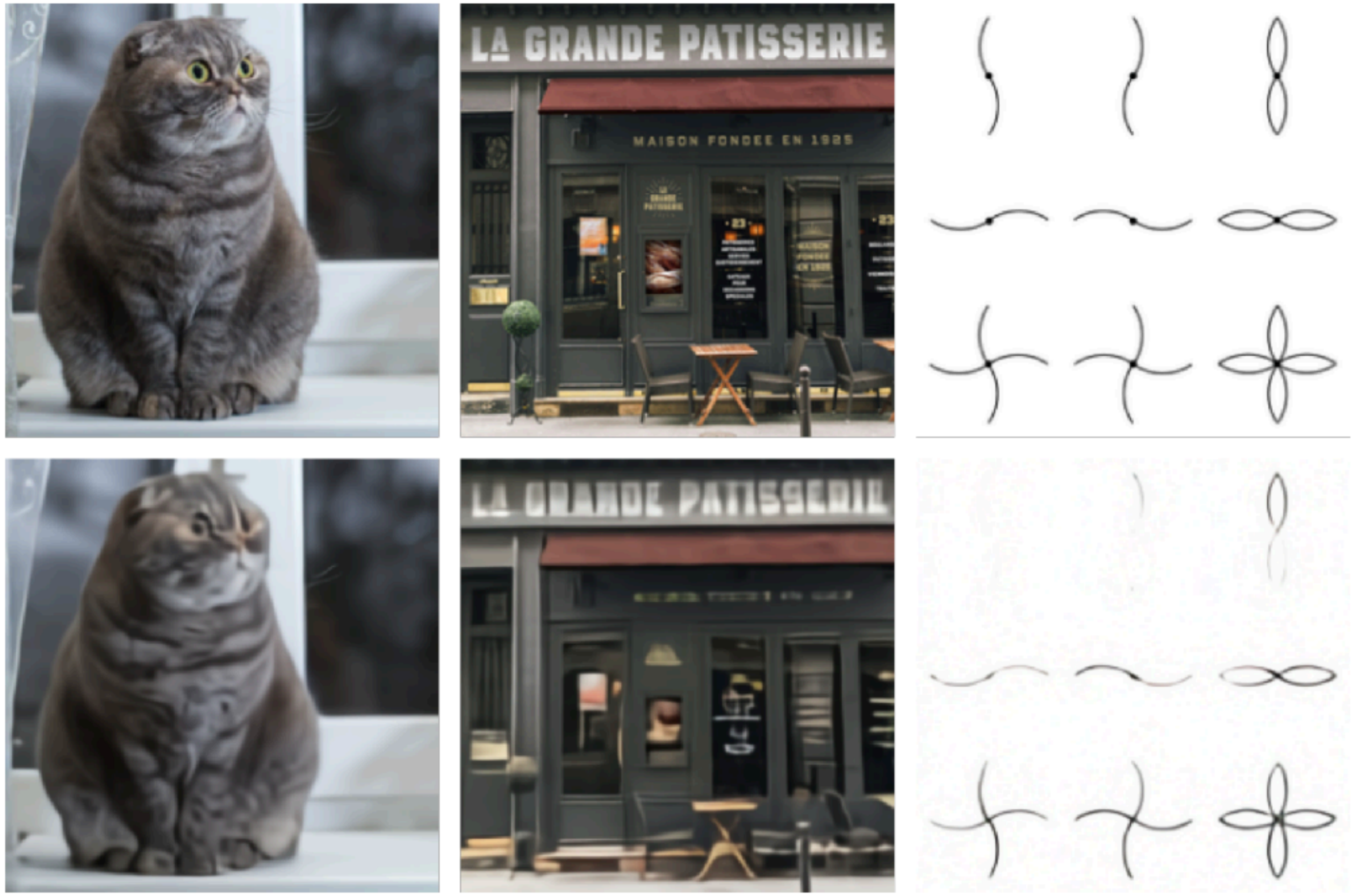
Ian Goodfellow (Chapter 13, Deep Learning Textbook, Figure from Chelsea Finn)

AutoEncoders can ignore relevant features for a task



OpenAI DALL-E Reconstructions (Ramesh et al 2021)

AutoEncoders can ignore relevant features for a task

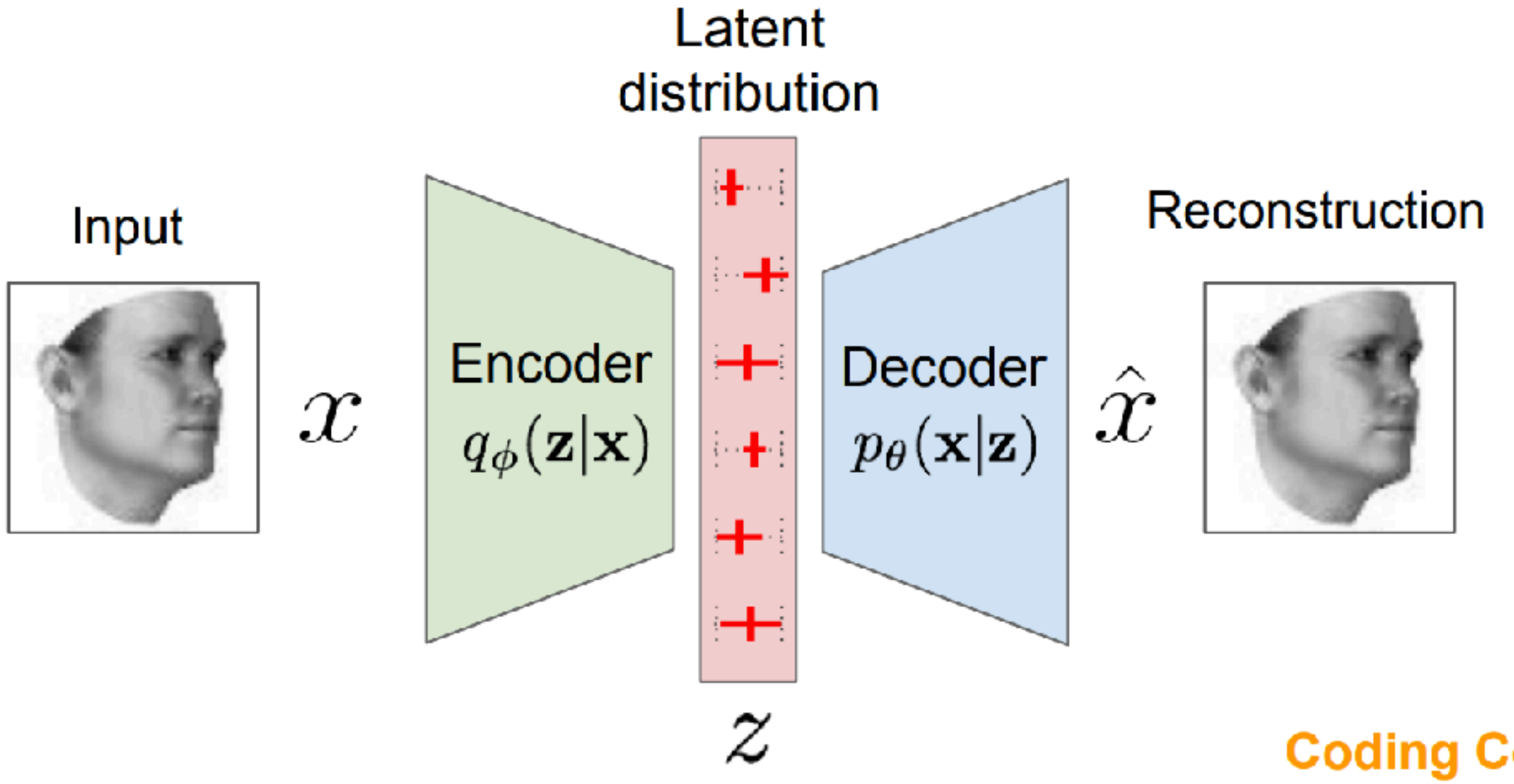


If the task is to accurately identify the pastry place, some of the details are blurry. However, future incarnations could potentially address these issues. It all depends on how much you downsample (8x in this case).

OpenAI DALL-E Reconstructions (Ramesh et al 2021)

Beta-Variational AutoEncoder (beta-VAE)

Kingma et al, 2014
Rezende et al, 2014



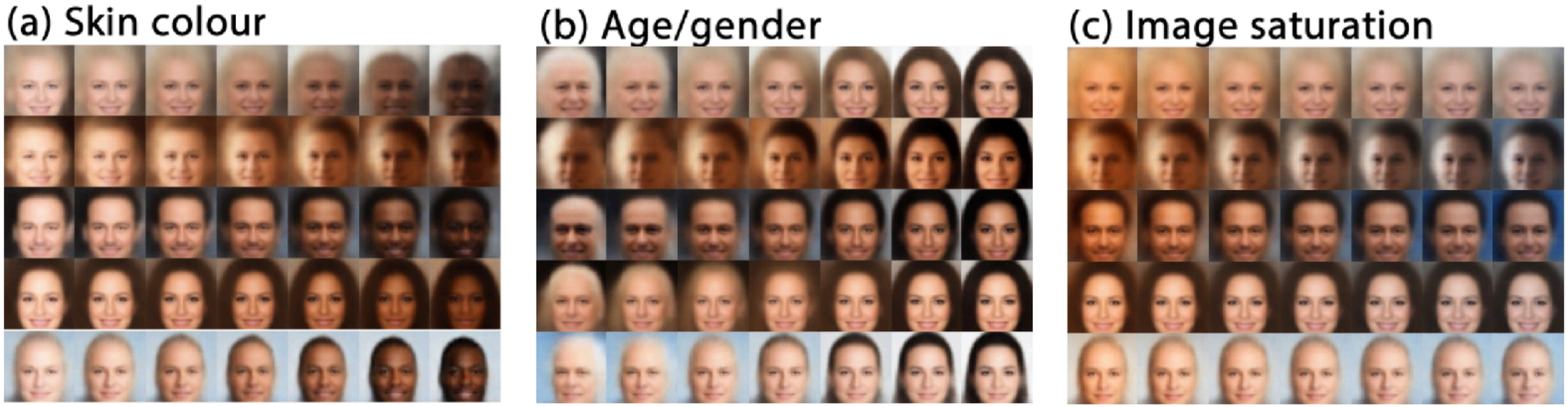
$$\mathcal{L}_{VAE}(\mathbf{x};\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log p_\theta(\mathbf{x}|\mathbf{z})\right]}_{\text{Reconstruction cost}} + \overbrace{KL\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)}^{\text{Coding Cost}} \cdot \beta$$

Increase the KL cost on the latent. Leads to more disentanglement

beta-VAE - Higgins et al 2016

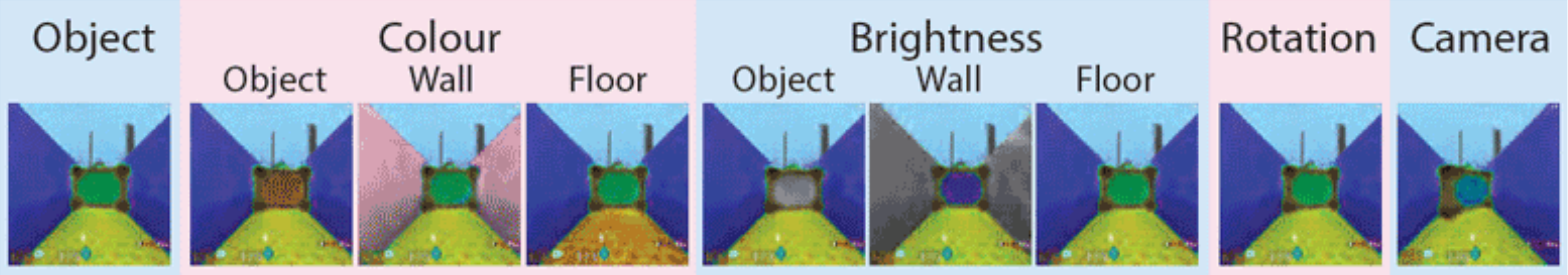
Slide from Alex Graves

Beta-Variational AutoEncoder (beta-VAE)



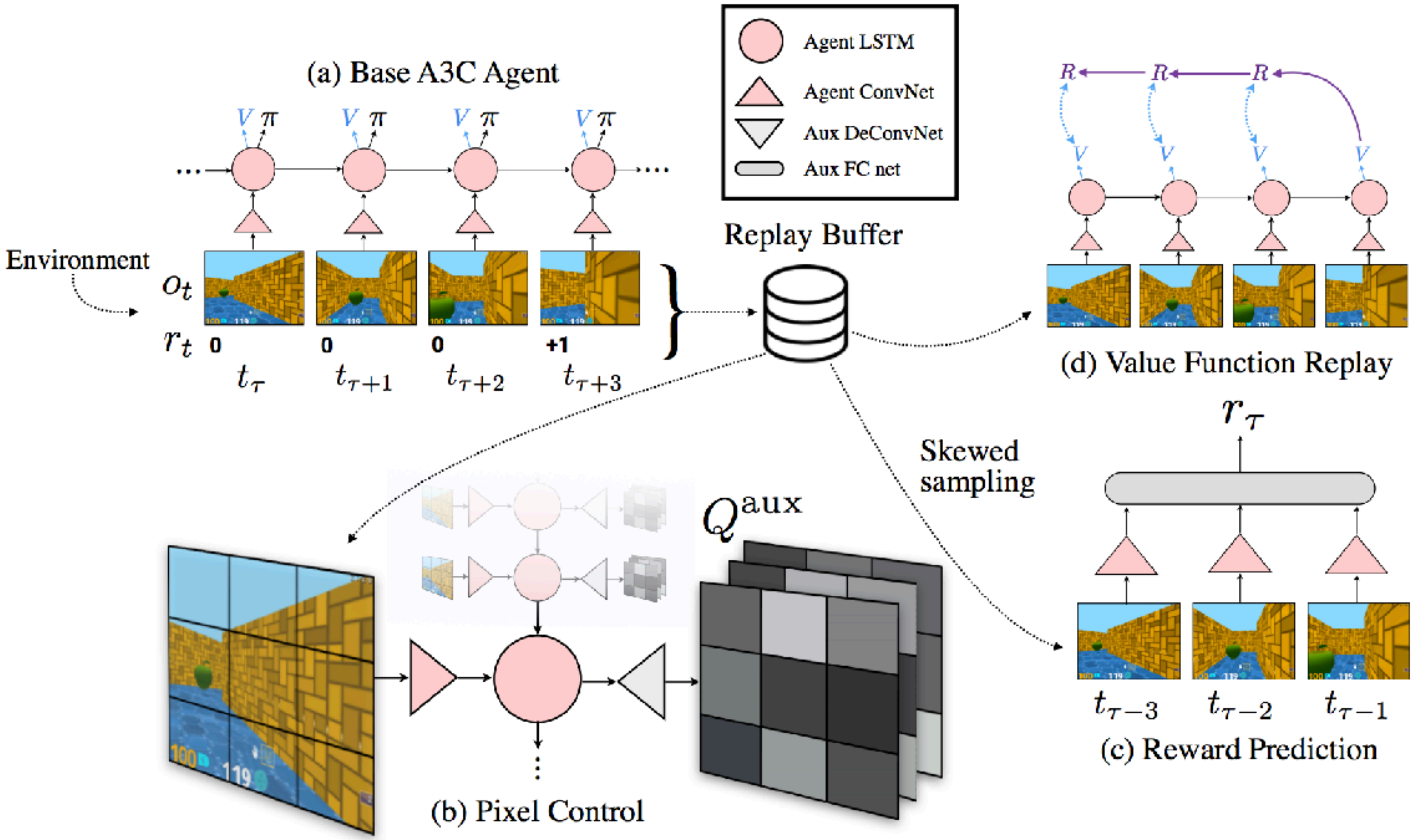
beta-VAE - Higgins et al 2016

Beta-Variational AutoEncoder (beta-VAE)



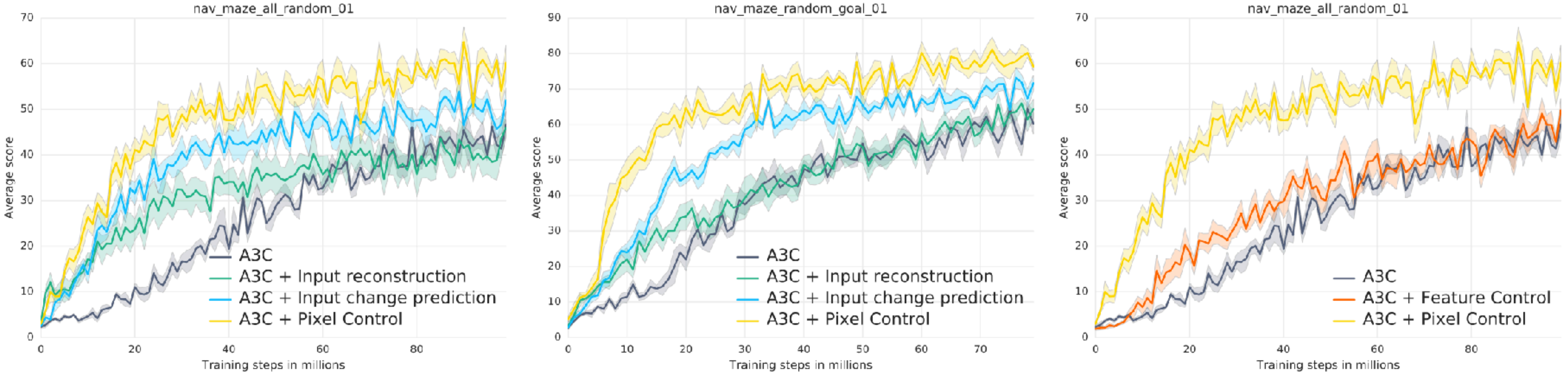
SCAN: Learning Abstract Hierarchical Compositional Visual Concepts (DeepMind Blog)

UNREAL Architecture



Reinforcement Learning with Unsupervised Auxiliary Tasks Jaderberg et al 2016

UNREAL Architecture



Reinforcement Learning with Unsupervised Auxiliary Tasks Jaderberg et al 2016

DARLA: Improving Zero-Shot Transfer in RL

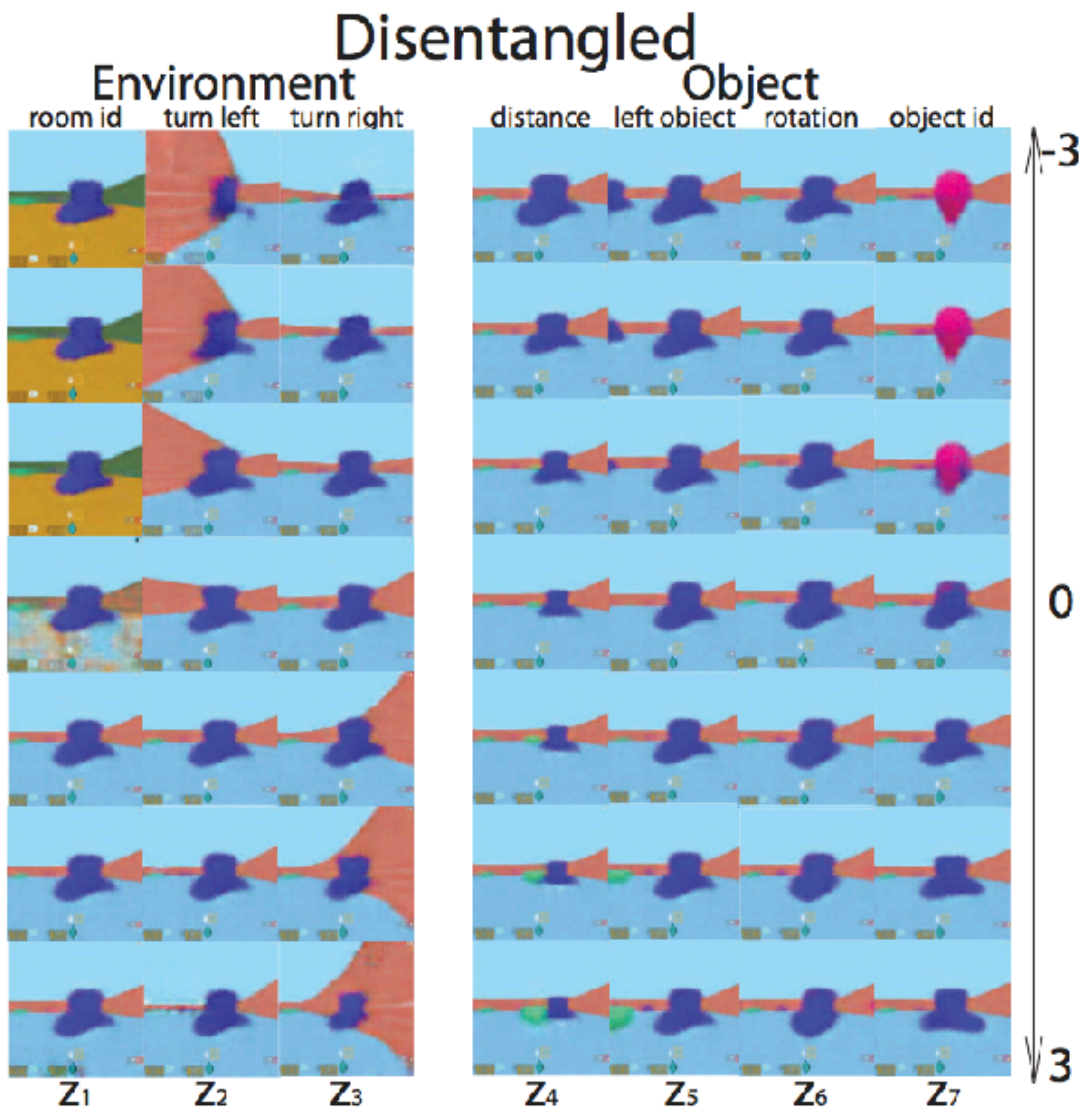


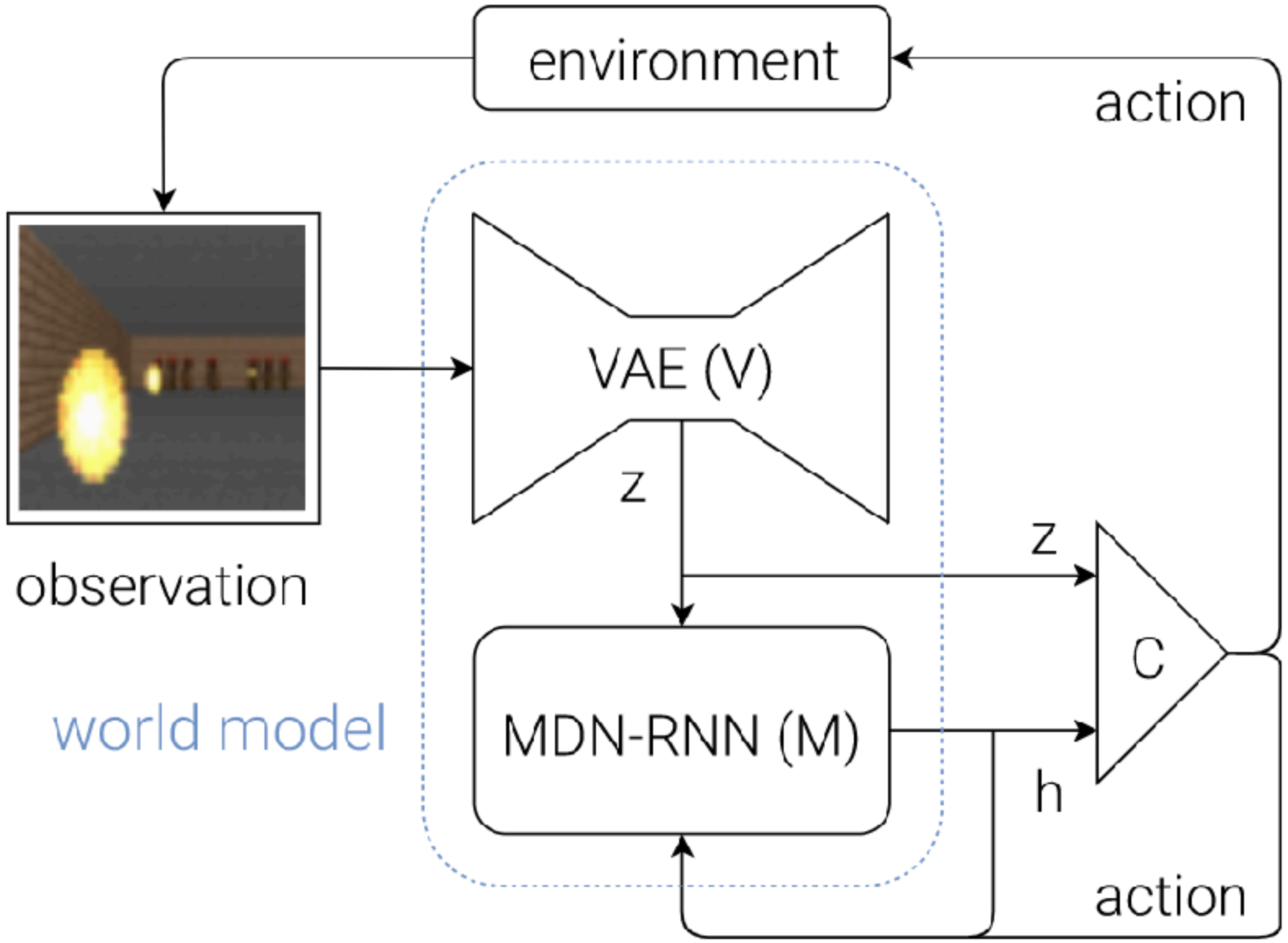
Table 1. Transfer performance

Vision Type	DeepMind Lab DQN	DeepMind Lab A3C	DeepMind Lab EC	Jaco (A3C) sim2sim	Jaco (A3C) sim2real
Baseline agent	1.86 ± 3.91	5.32 ± 3.36	-0.41 ± 4.21	97.64 ± 9.02	94.56 ± 3.55
UNREAL	-	4.13 ± 3.95	-	-	-
DARLA_{FT}	$\mathbf{13.36 \pm 5.8}$	1.4 ± 2.16	-	86.59 ± 5.53	99.25 ± 2.3
DARLA_{ENT}	3.45 ± 4.47	15.66 ± 5.19	5.69 ± 3.73	84.77 ± 4.42	59.99 ± 15.05
DARLA_{DAE}	7.83 ± 4.47	6.74 ± 2.81	5.59 ± 3.37	85.15 ± 7.43	100.72 ± 4.7
DARLA	10.25 ± 5.46	$\mathbf{19.7 \pm 5.43}$	$\mathbf{11.41 \pm 3.52}$	$\mathbf{100.85 \pm 2.92}$	$\mathbf{108.2 \pm 5.97}$

DARLA's performance is significantly different from all baselines under Welch's unequal variances t-test with $p < 0.01$ ($N \in [60, 150]$).

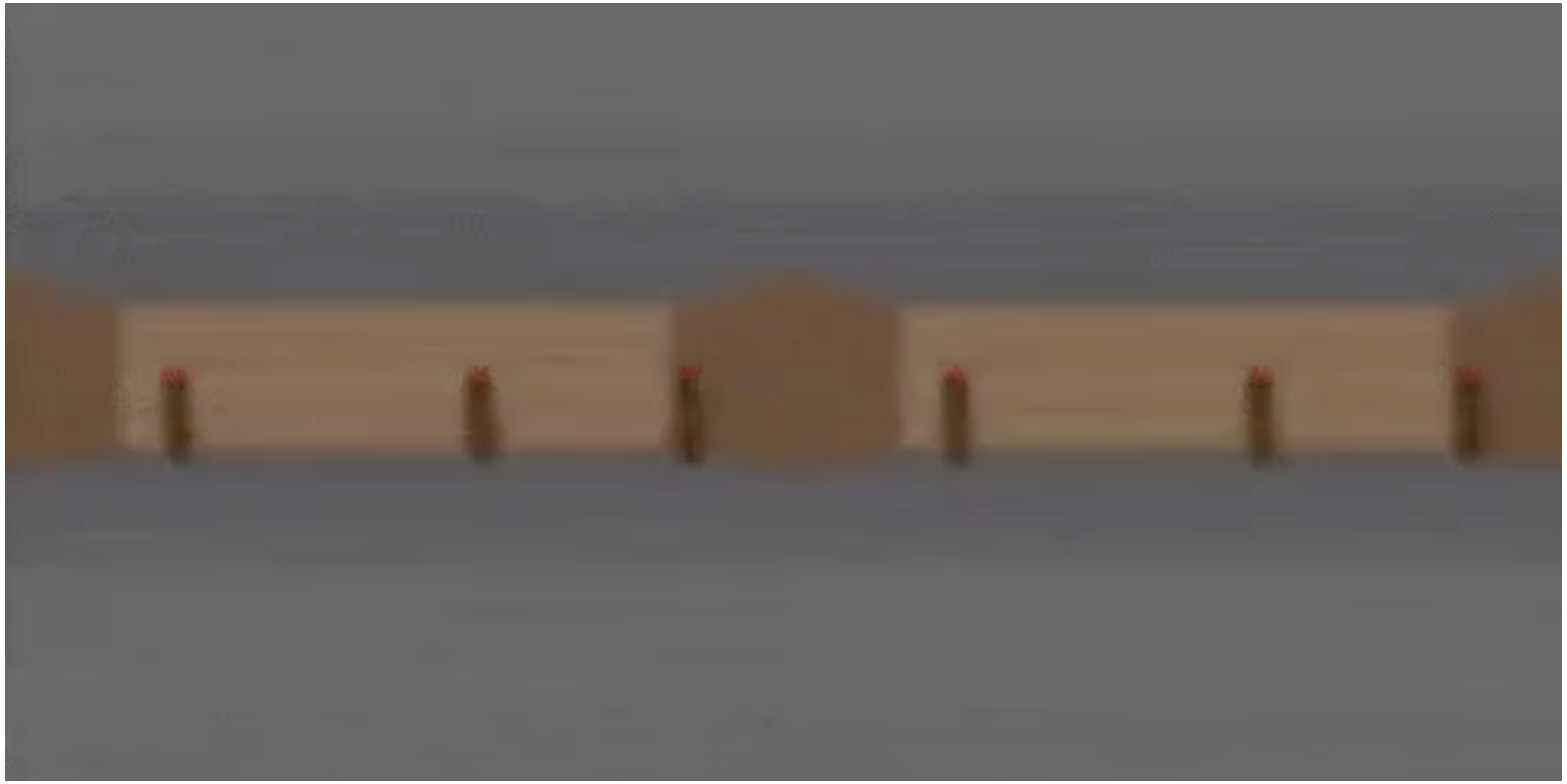
Higgins et al 2017

World Models (Ha and Schmidhuber 2018)



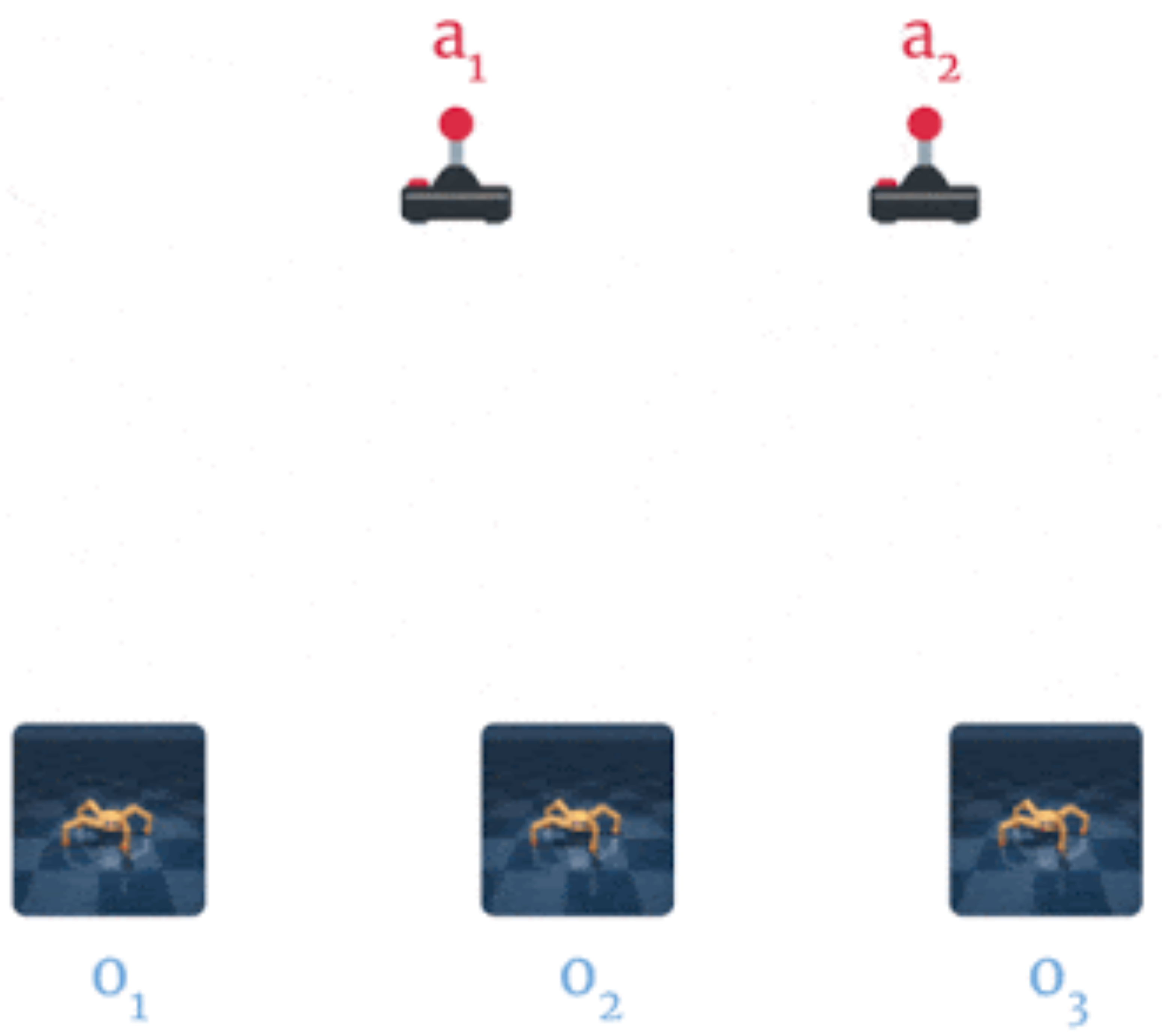
World Models - Ha & Schmidhuber 2018

World Models (Ha and Schmidhuber 2018)



World Models - Ha & Schmidhuber 2018

Dreamer (Hafner et al 2019)



Dream to Control: Learning behaviors by latent imagination - Hafner et al 2019

Dreamer (Hafner et al 2019)

o_1

Dream to Control: Learning behaviors by latent imagination - Hafner et al 2019

Quick background

1. Autoencoder
2. Variational Autoencoder
3. Contrastive Learning
4. Siamese Networks
5. Data-Augmentations

Contrastive-like UL

Siamese Networks: Invariance to (augmented) views

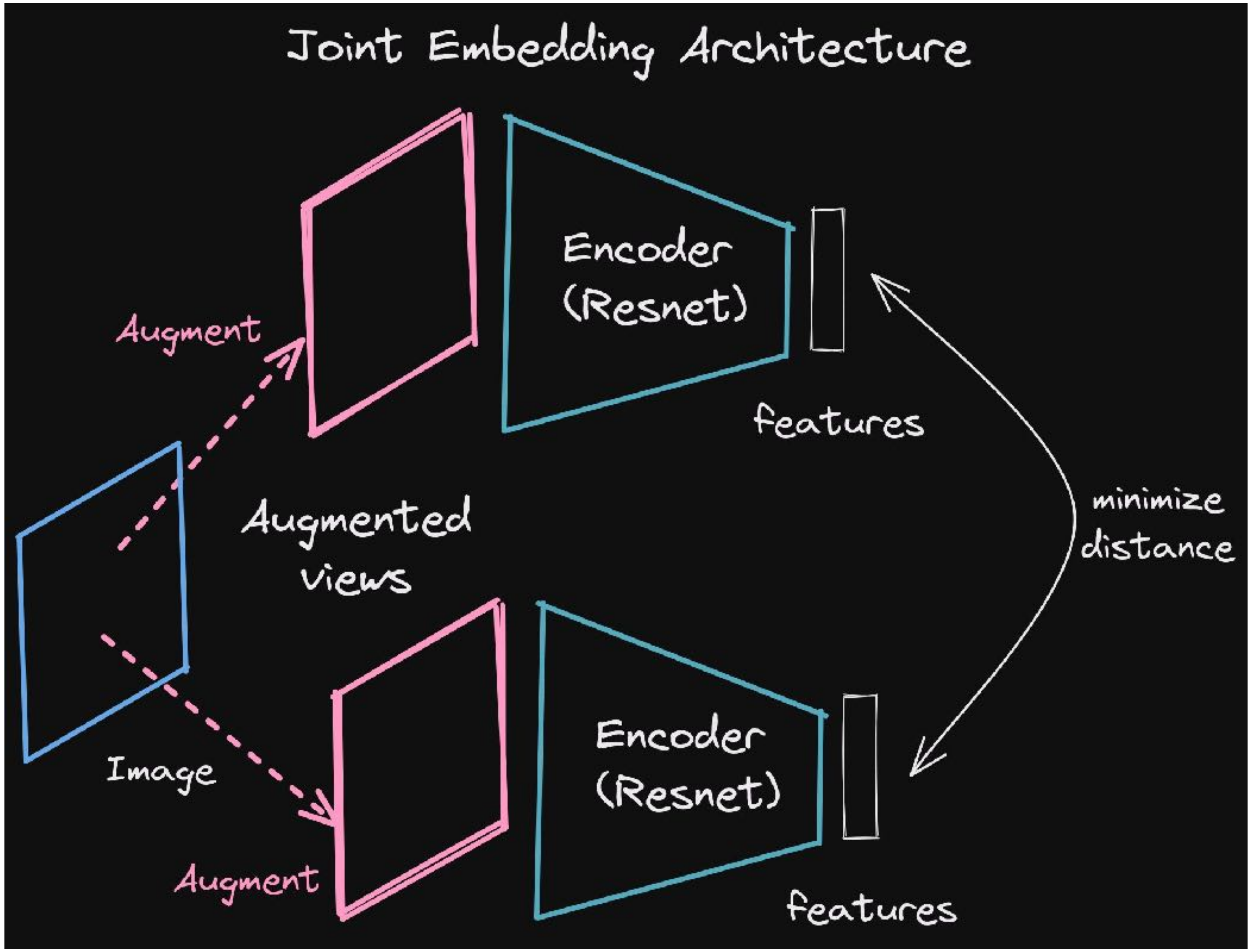
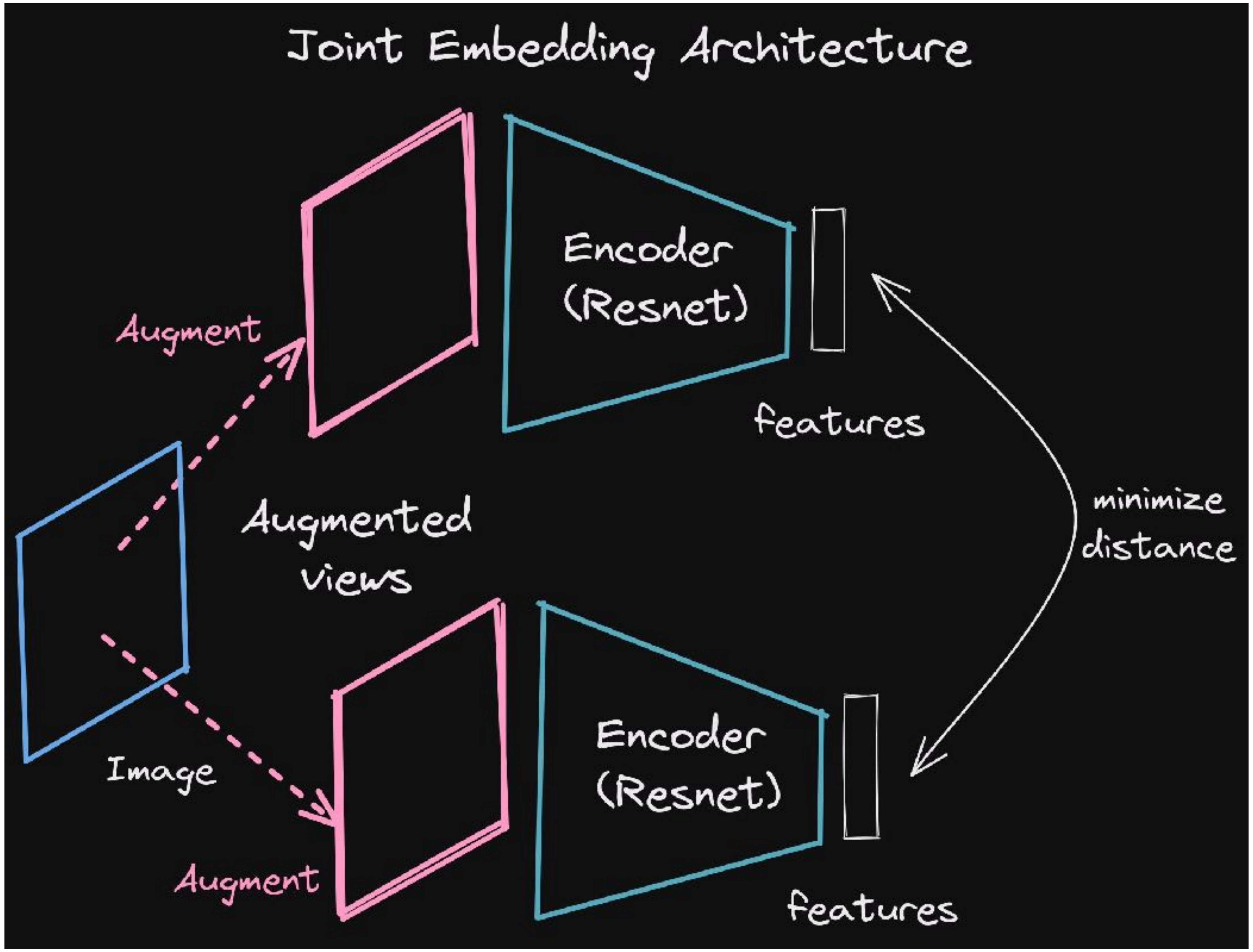


Figure from Antonin Raffin, @araffin2

Siamese Networks: Invariance to (augmented) views



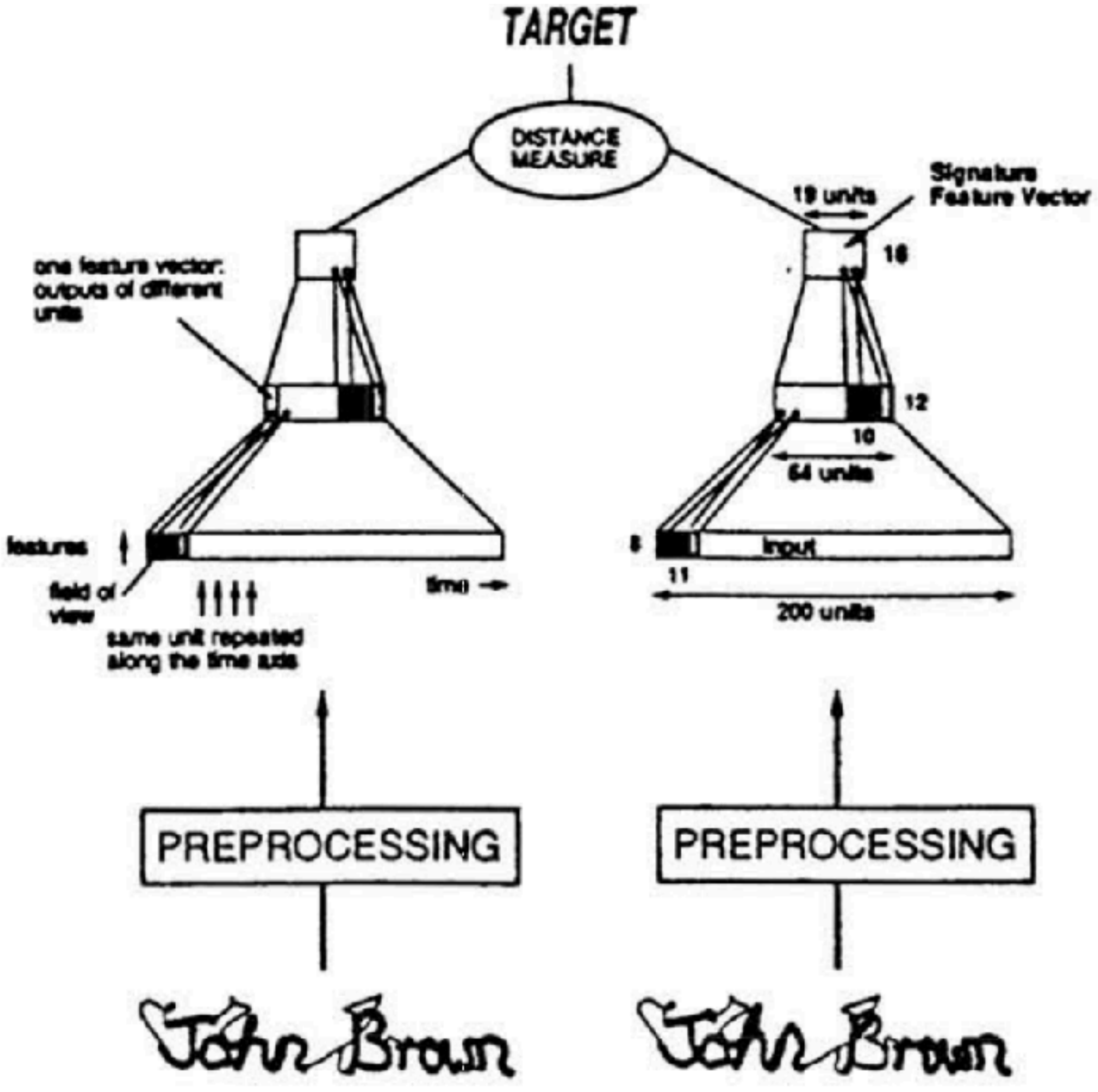
If you have some knowledge about the domain, encode it into the latent space through the form of invariances.

More often than not, this results in emergent representations that capture the most *useful aspects* of the high-dimensional inputs.

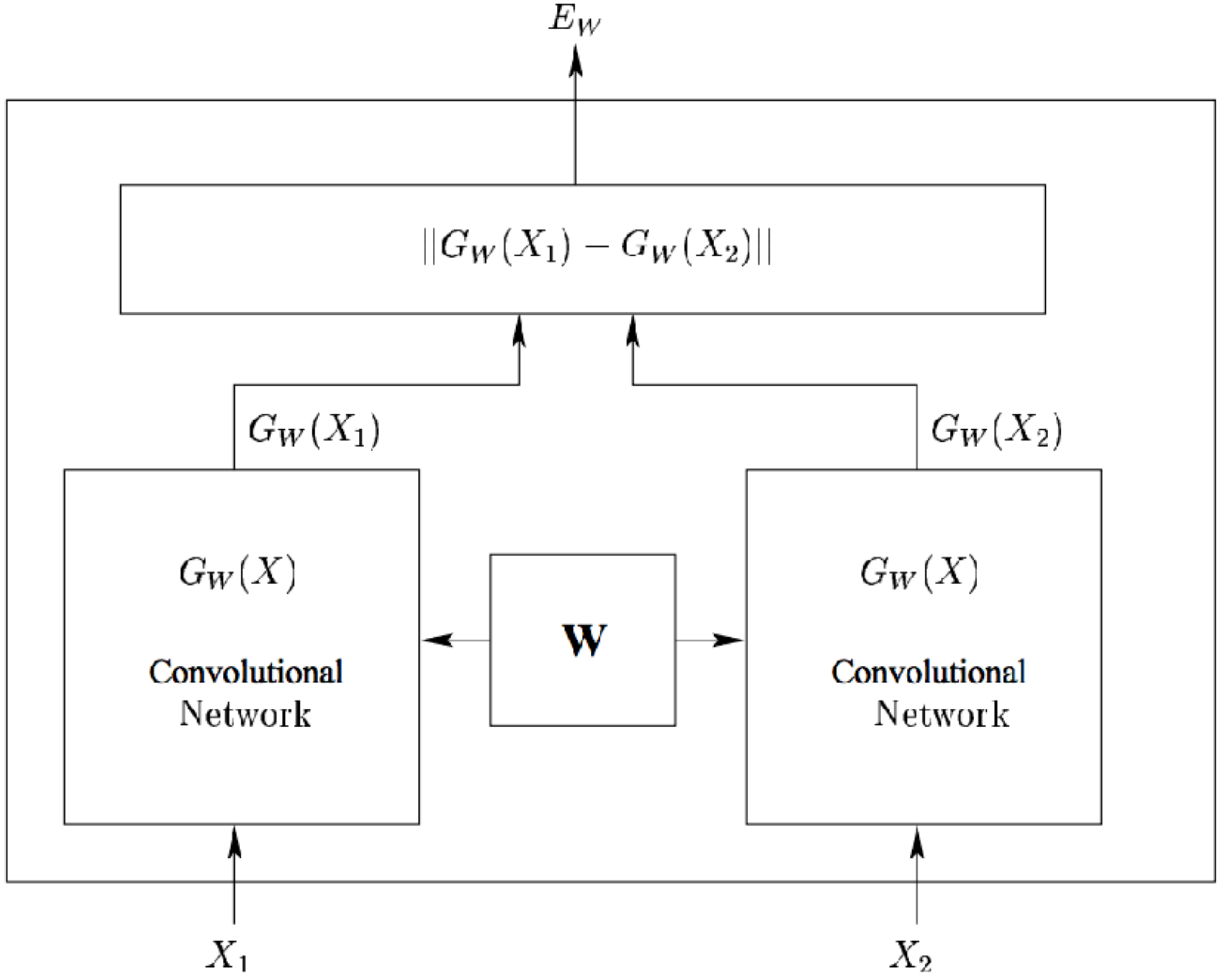
Why? Because it ignores what you ask the encoder to be invariant to - and the better you can prescribe that, the more the model knows what **not** to focus on.

Figure from Antonin Raffin, @araffin2

Siamese Networks: Invariance to (augmented) views

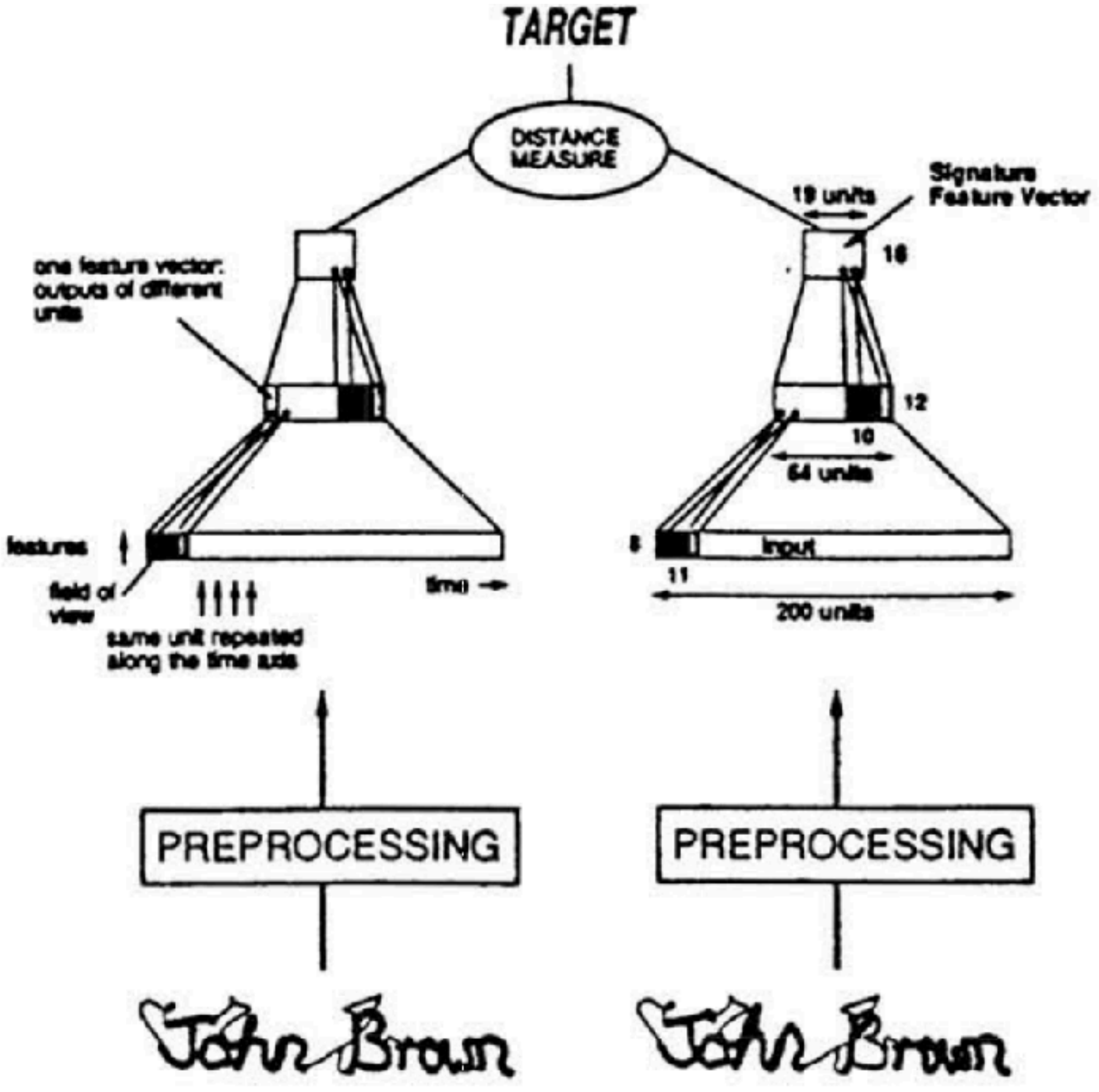


Signature Verification (Bromley, Guyon, LeCun, et al, 93)

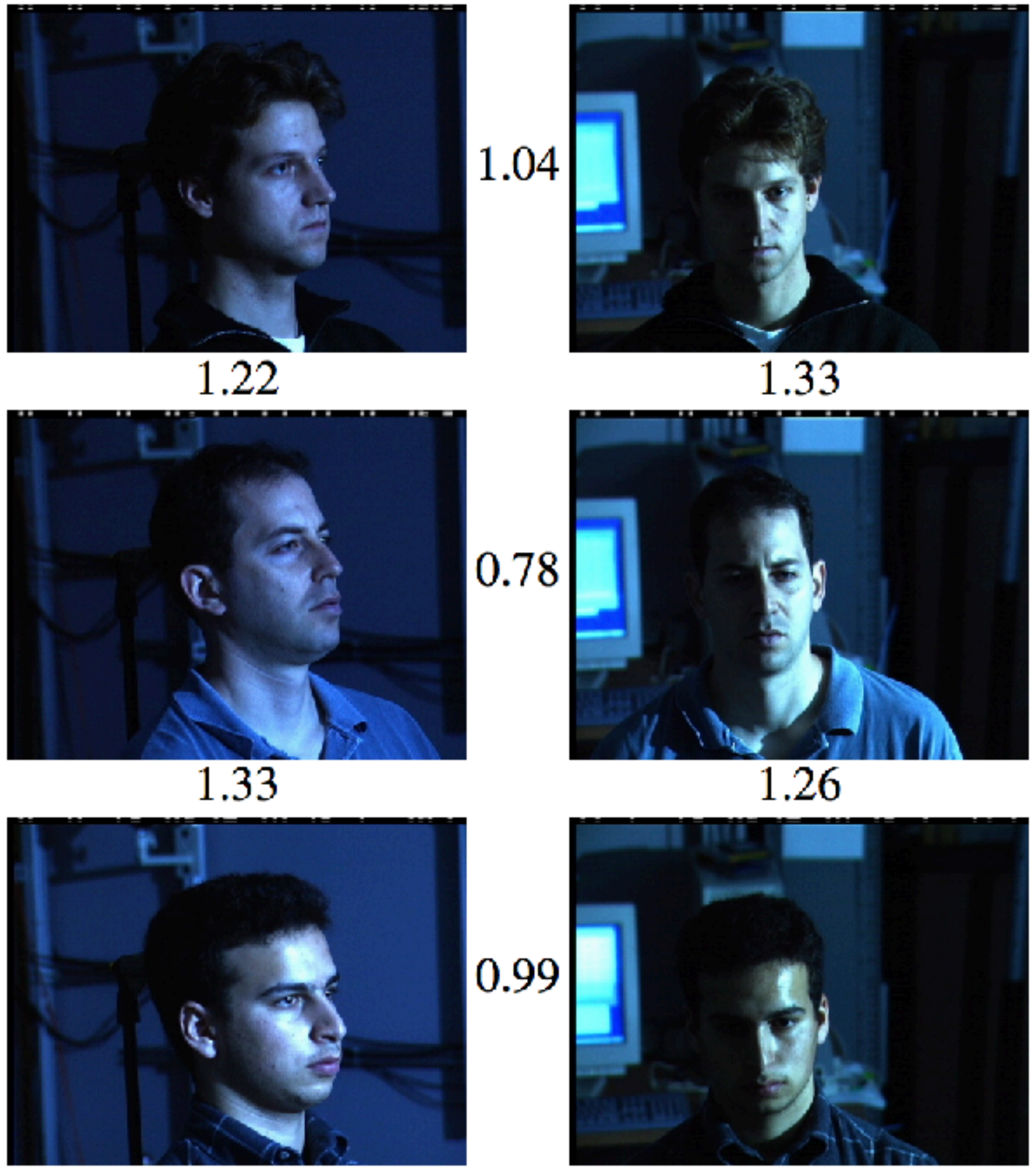


Learning a similarity metric discriminatively,
Chopra, Hadsell, LeCun, 2005

Siamese Networks: Invariance to (augmented) views



Signature Verification (Bromley, Guyon, LeCun, et al, 93)



FaceNet, Schroff et al 2015

Invariance to pose, lighting, etc - model focuses on encoding facial features

One way to train Siamese Networks is Contrastive Learning

Contrastive Learning

Take a datapoint (an image), and try to fit a scoring function to make sure it aligns more with a positive relative to a negative.

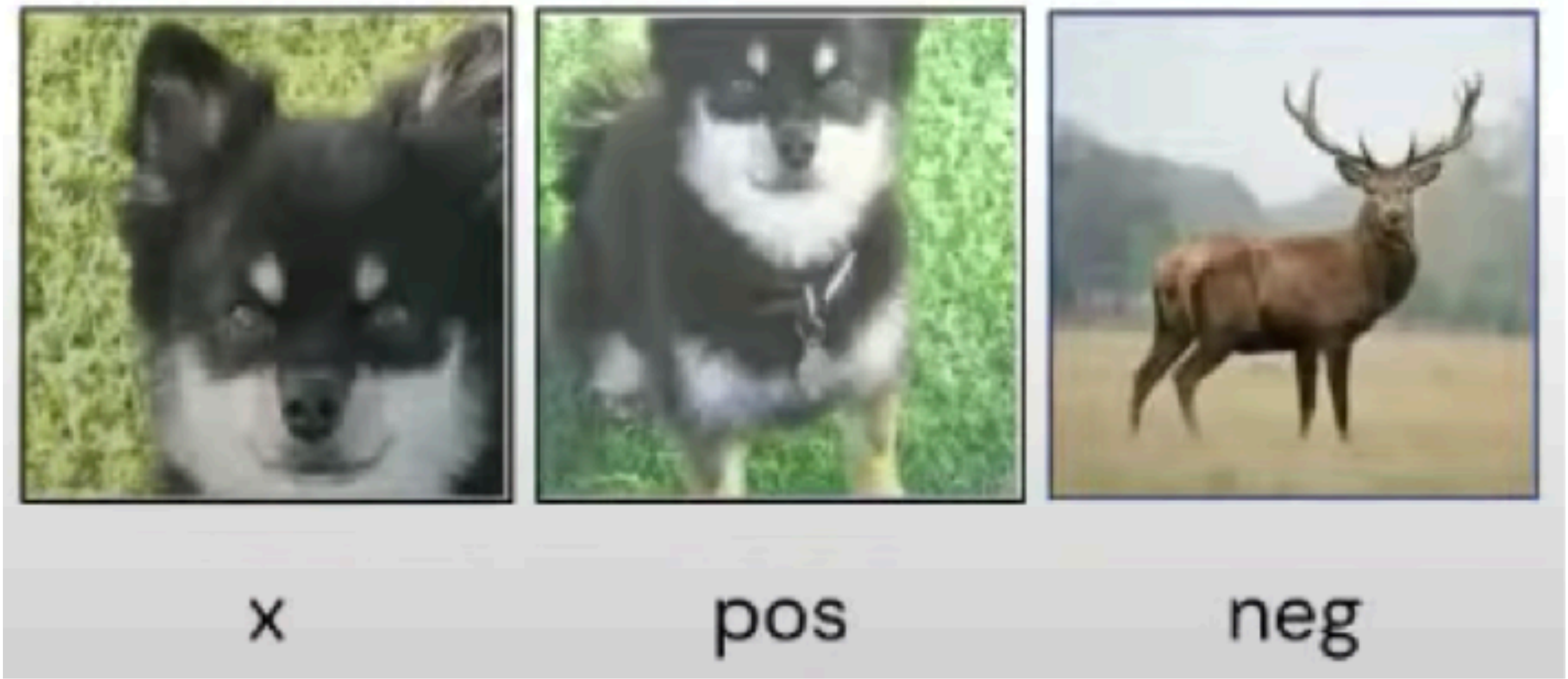
$$score(x, x_{pos}) > score(x, x_{neg})$$

Slide adapted from Aaron van den oord

Contrastive Learning

Take a datapoint (an image), and try to fit a scoring function to make sure it aligns more with a positive relative to a negative.

$$score(x, x_{pos}) > score(x, x_{neg})$$



Learn the concept of cats and dogs

Slide adapted from Aaron van den oord

Contrastive Learning

Take a datapoint (an image), and try to fit a scoring function to make sure it aligns more with a positive relative to a negative.

$$score(x, x_{pos}) > score(x, x_{neg})$$

$$L_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{s(x, x_{\text{pos}})}{s(x, x_{\text{pos}}) + \sum_{y_j \neq x_{\text{pos}}} s(x, y_j)}\right]$$

Slide adapted from Aaron van den oord

Contrastive Learning

Take a datapoint (an image), and try to fit a scoring function to make sure it aligns more with a positive relative to a negative.

$$score(x, x_{pos}) > score(x, x_{neg})$$

$$L_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{s(x, x_{\text{pos}})}{s(x, x_{\text{pos}}) + \sum_{y_j \neq x_{\text{pos}}} s(x, y_j)}\right]$$

InfoNCE (or N-Pairs) Loss

1. Representation Learning with Contrastive Predictive Coding (van den Oord et al 2018)
2. Improved Deep Metric Learning with Multi-Class N-Pairs Loss - (Sohn et al 2016)
3. Deep InfoMax, AMDIM (Hjelm, Bachman, et al 2019)

Slide adapted from Aaron van den oord

Contrastive Learning

Take a datapoint (an image), and try to fit a scoring function to make sure it aligns more with a positive relative to a negative.

$$score(x, x_{pos}) > score(x, x_{neg})$$

$$L_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{s(x,y)}{\sum_{y_j} s(x,y_j)}\right]$$

InfoNCE (or N-Pairs) Loss

1. Representation Learning with Contrastive Predictive Coding (van den Oord et al 2018)
2. Improved Deep Metric Learning with Multi-Class N-Pairs Loss - (Sohn et al 2016)
3. Deep InfoMax, AMDIM (Hjelm, Bachman, et al 2019)

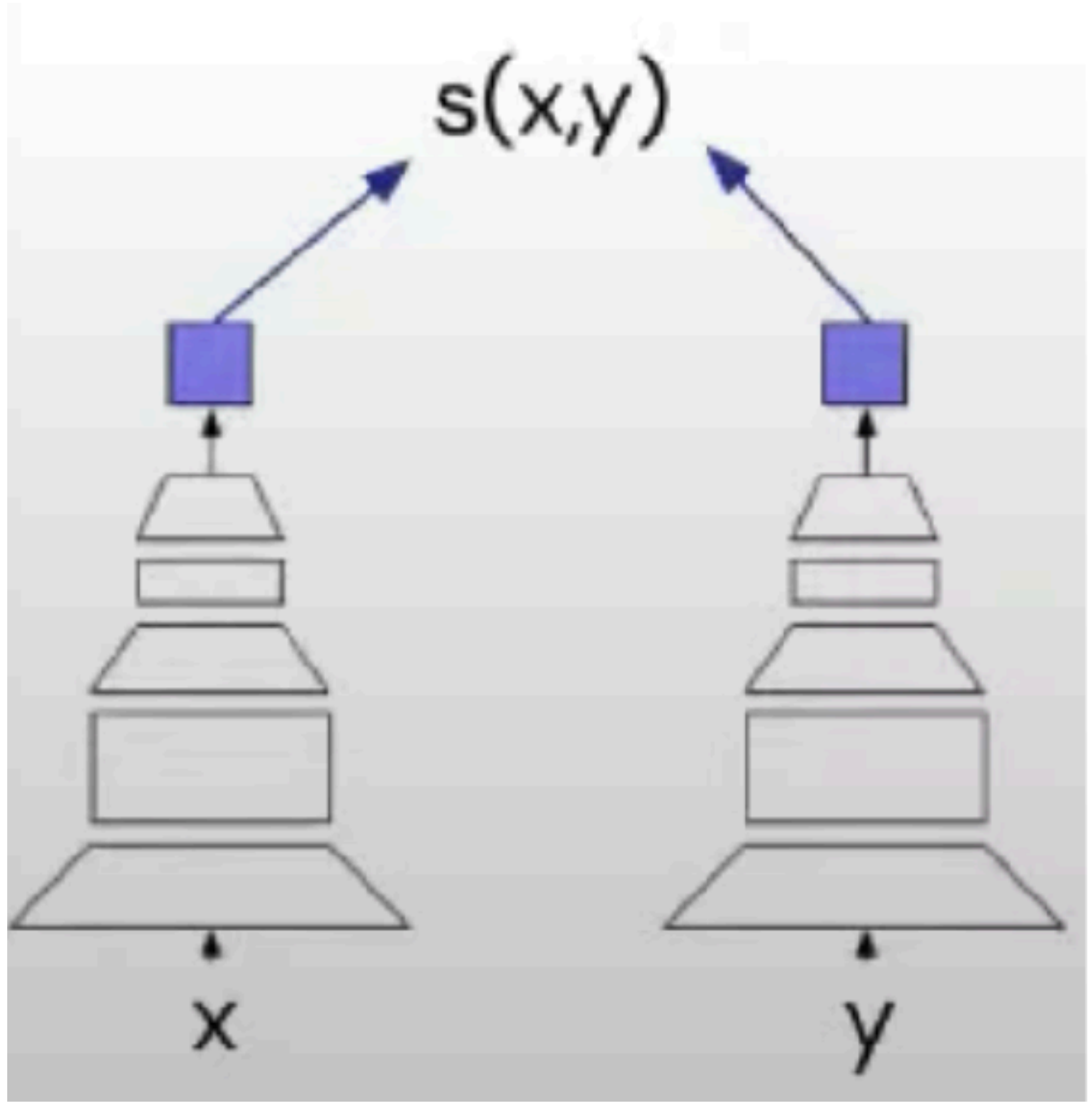
Slide adapted from Aaron van den oord

Contrastive Learning: InfoNCE Loss

$$L_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{s(x,y)}{\sum_{y_j} s(x,y_j)}\right]$$

$$s(x,y) = e^{f_X(x)^T f_Y(y)}$$

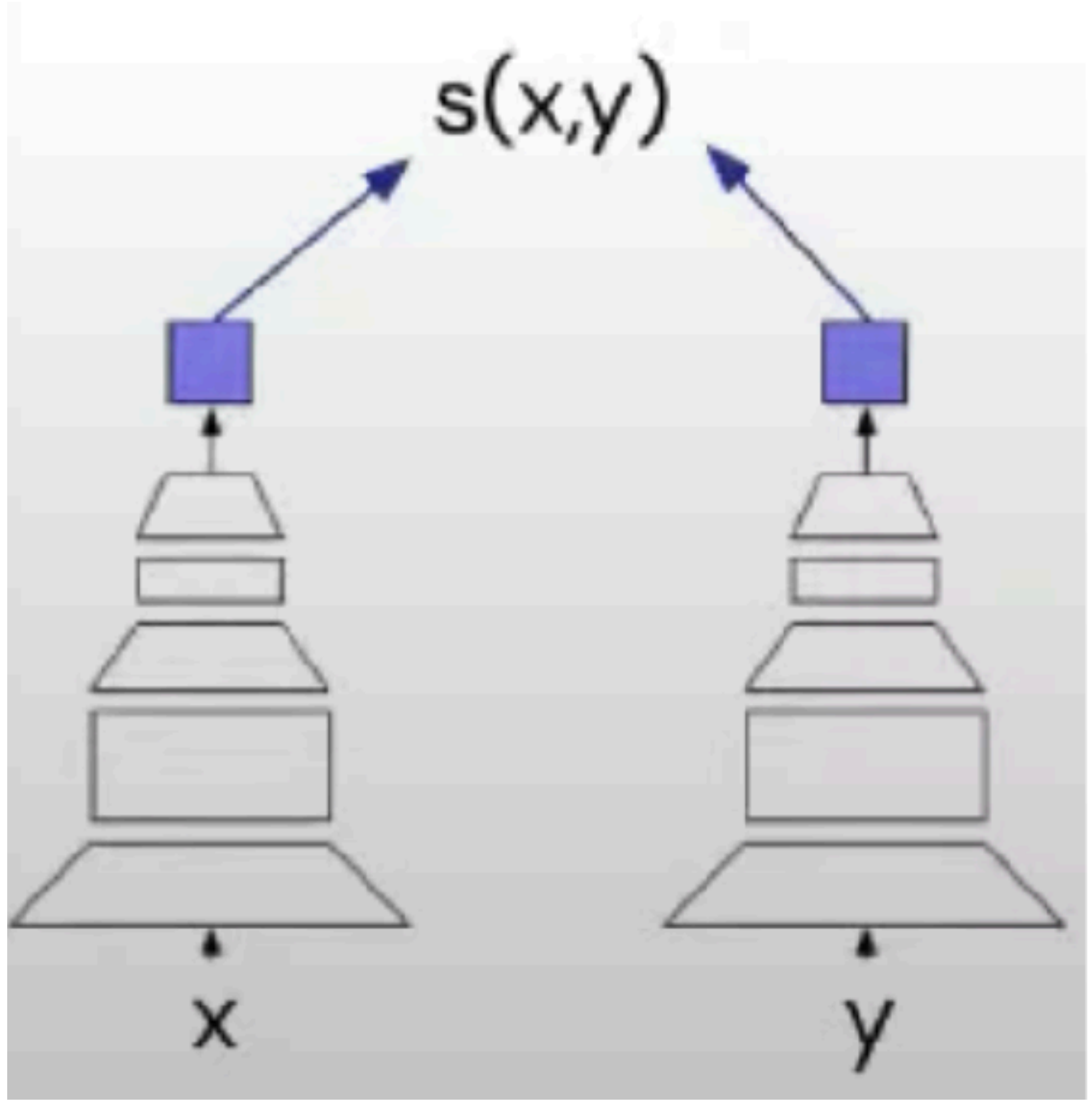
$$L_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{e^{f_X(x)^T f_Y(y)}}{\sum_{y_j} e^{f_X(x)^T f_Y(y_j)}}\right]$$



1. Representation Learning with Contrastive Predictive Coding (van den Oord et al 2018)
2. Improved Deep Metric Learning with Multi-Class N-Pairs Loss - (Sohn et al 2016)
3. Deep InfoMax, AMDIM (Hjelm, Bachman, et al 2019)

Slide adapted from Aaron van den oord

Contrastive Learning: InfoNCE Loss

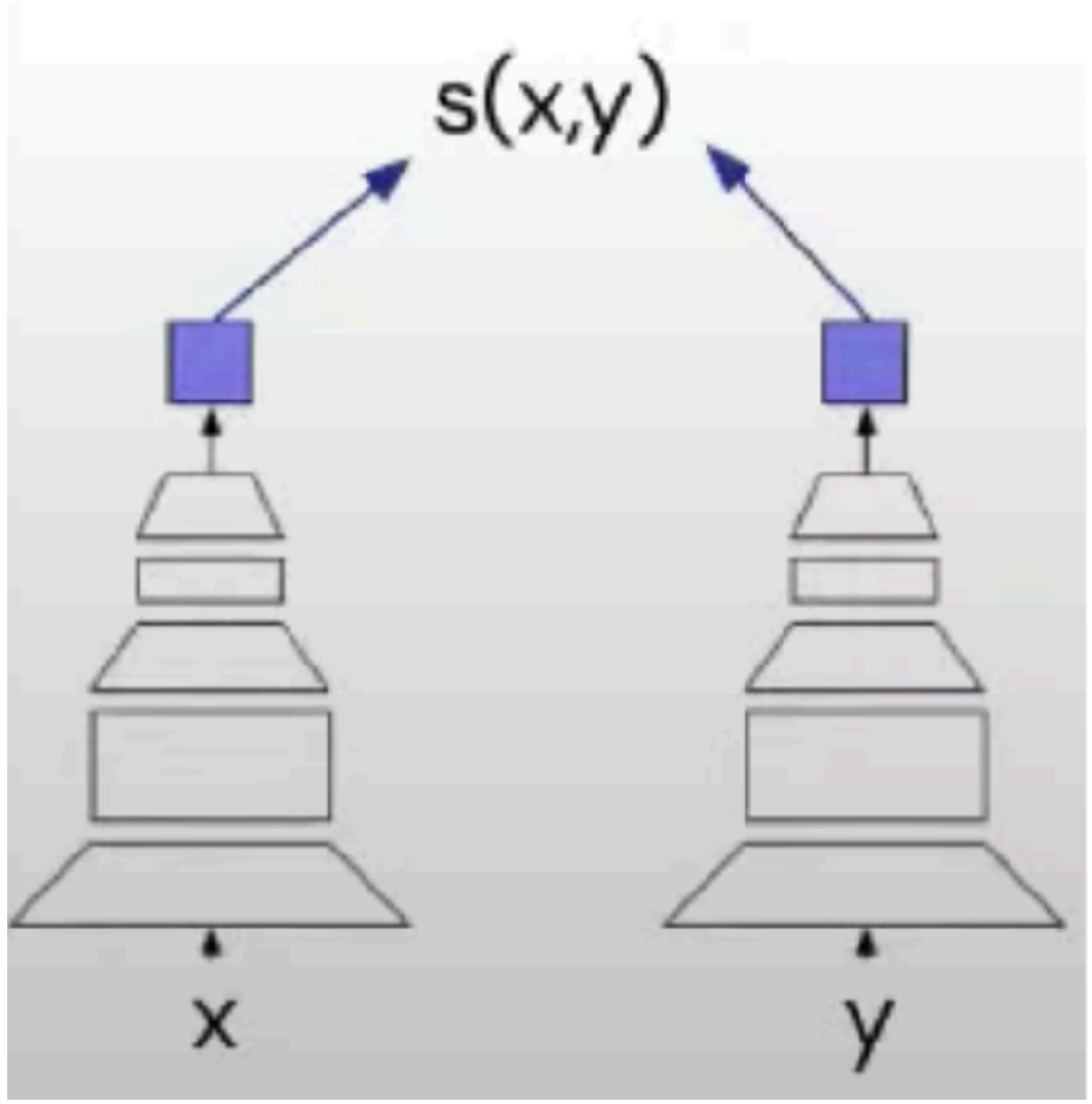


1. *Maximize mutual information* between the views x and y.
2. Convert it to a classification problem, optimized with a stable cross-entropy loss.
3. Representations capture things *common* to x and y.
4. By augmenting x and y in several ways (stuff you don't want to capture), you only capture the *relevant left-over* common things.

1. Representation Learning with Contrastive Predictive Coding (van den Oord et al 2018)
2. Improved Deep Metric Learning with Multi-Class N-Pairs Loss - (Sohn et al 2016)
3. Deep InfoMax, AMDIM (Hjelm, Bachman, et al 2019)

Slide adapted from Aaron van den oord

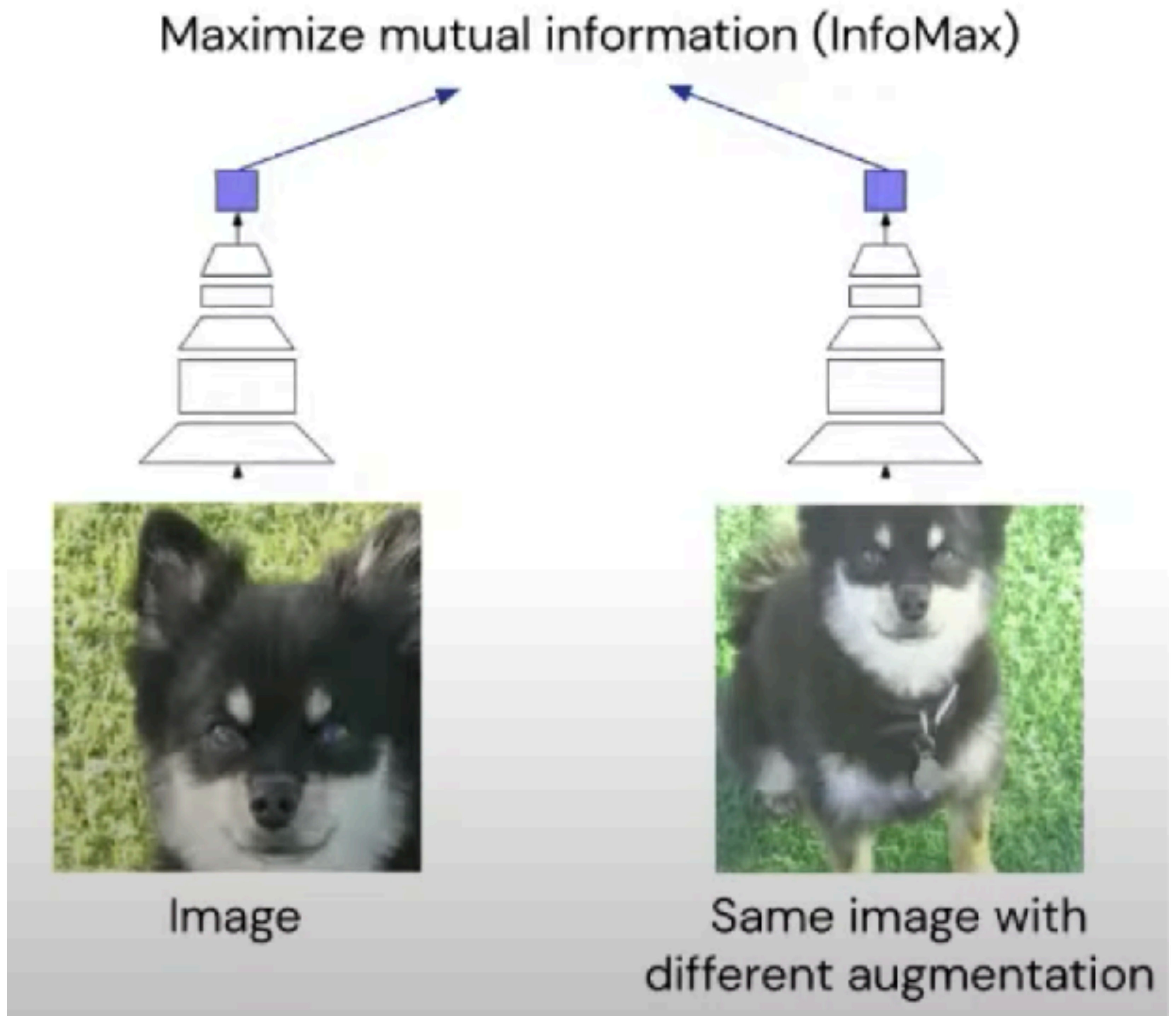
Contrastive Learning: InfoNCE Loss



1. Plenty of choices for what x and y can be.
2. x and y could be two augmented views of the same image.
3. x could be the past frame (aug), y could be the future frame (aug)
4. x could be past frame (aug) + action, y could be future (aug)…
5. Whatever it is, you are optimizing for MI between x and y (lower bound)

Slide adapted from Aaron van den oord

Contrastive Learning: InfoNCE Loss



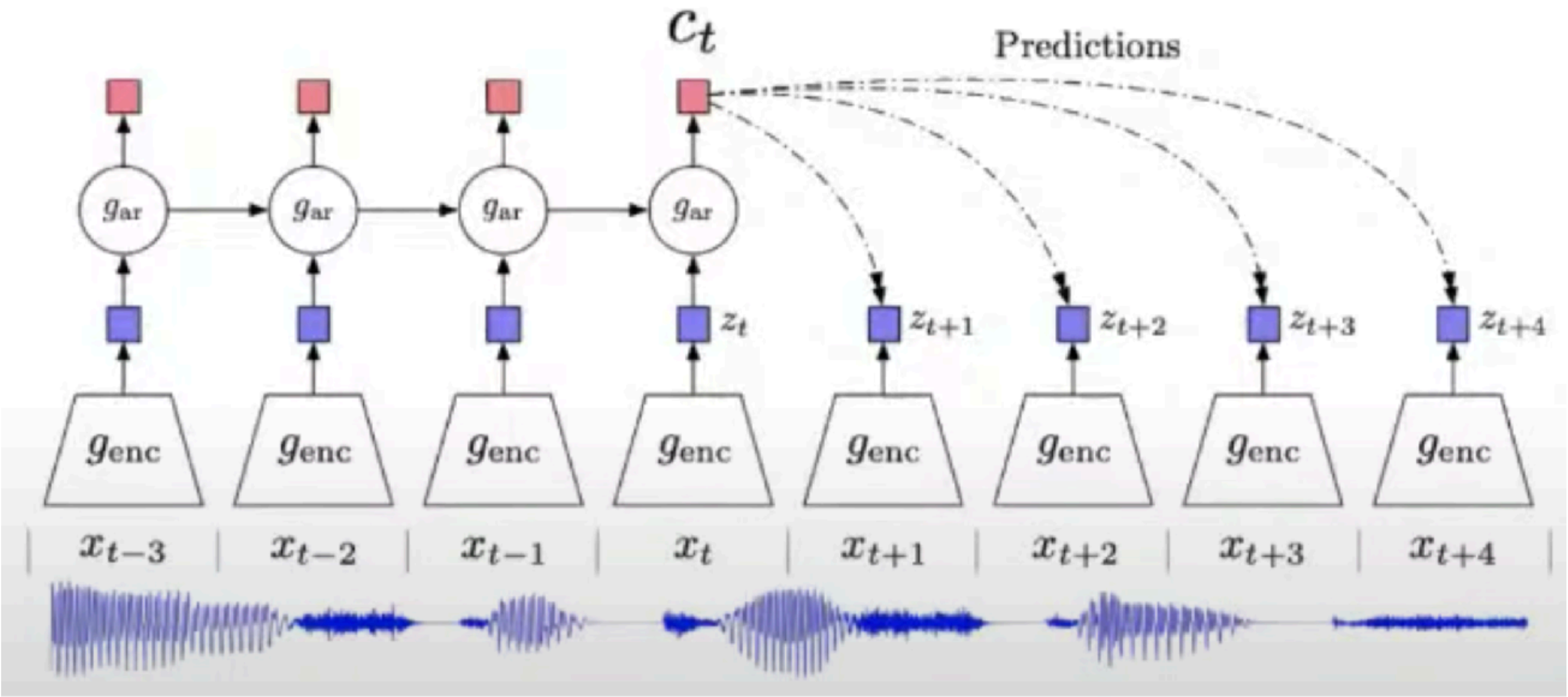
1. Plenty of choices for what x and y can be.
2. x and y could be two augmented views of the same image.
3. x could be the past frame (aug), y could be the future frame (aug)
4. x could be past frame (aug) + action, y could be future (aug)…
5. Whatever it is, you are optimizing for MI between x and y (lower bound)

Slide adapted from Aaron van den oord

Contrastive Learning: SimCLR

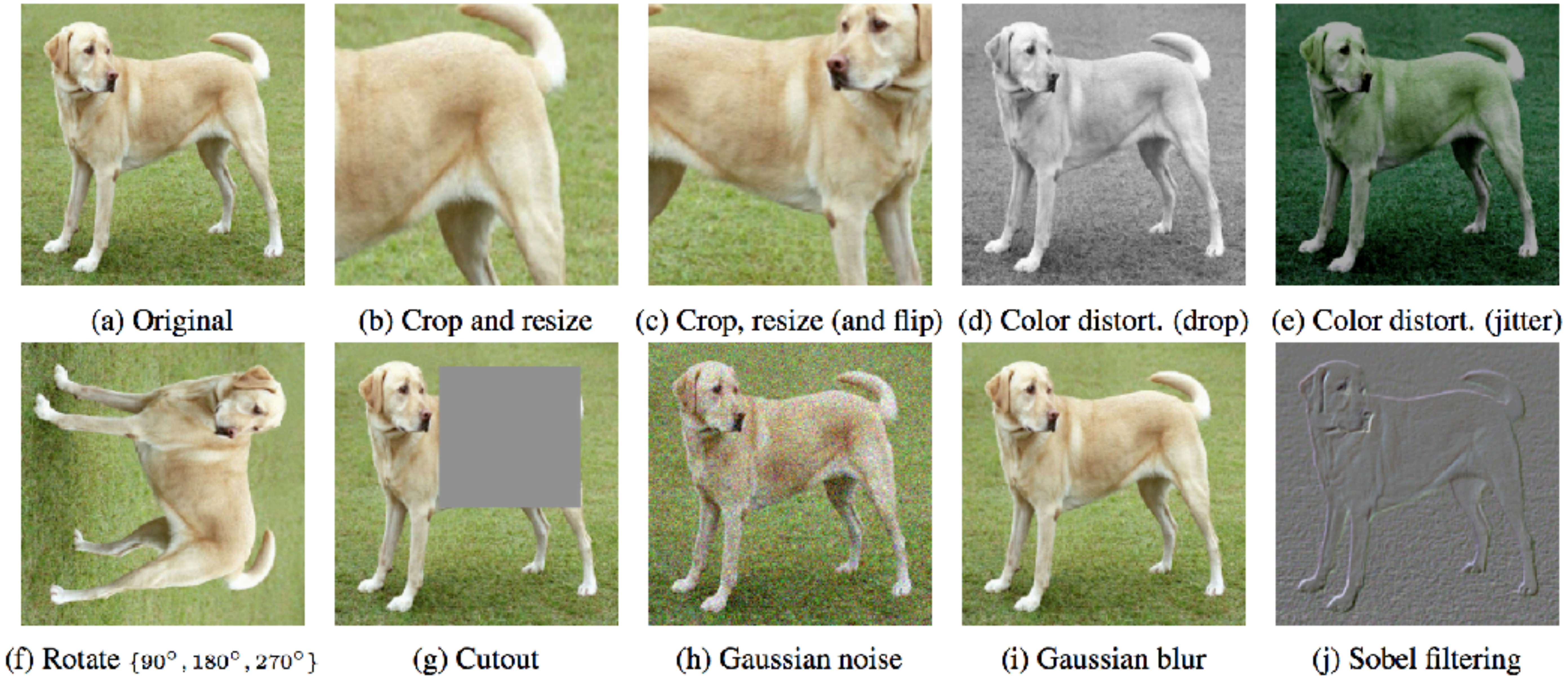
A Simple Framework for Contrastive Learning of Visual Representations, Chen et al 2020

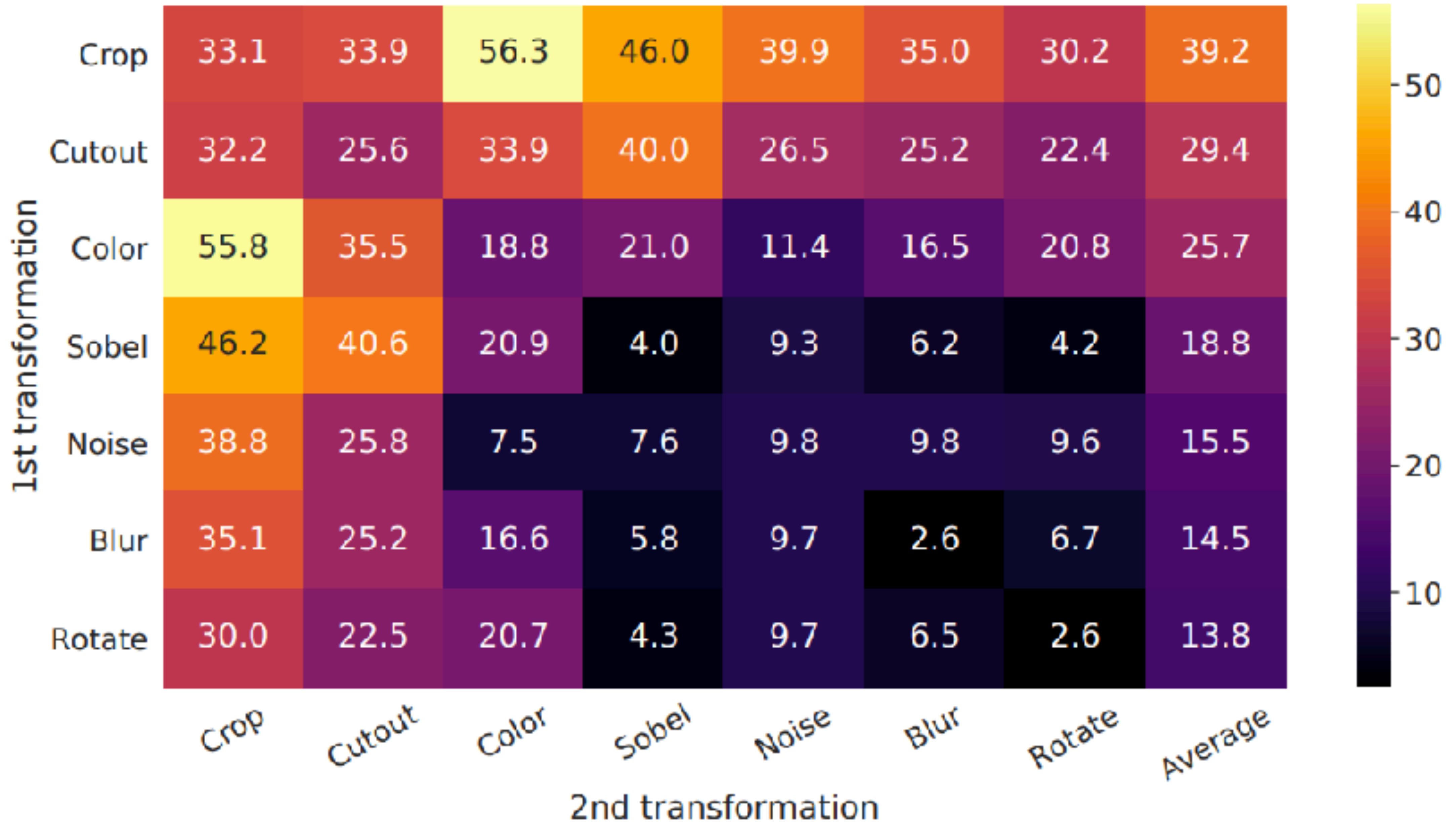
Contrastive Learning: Contrastive Predictive Coding (CPC)



1. Representation Learning using Contrastive Predictive Coding, van den Oord et al 2018
2. Data-Efficient Image Recognition using Contrastive Predictive Coding, Henaff et al 2020

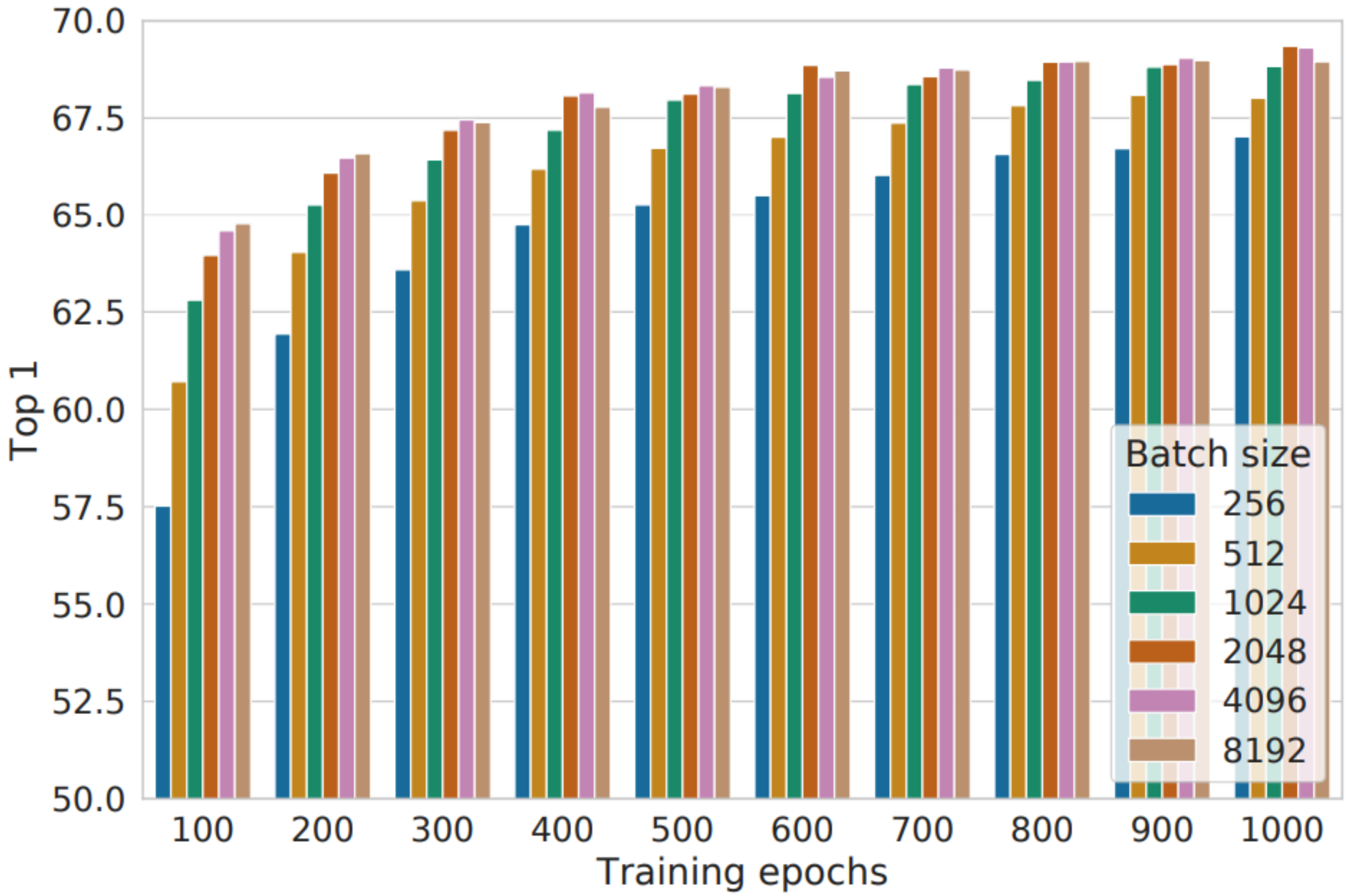
SimCLR: Importance of Augs





A Simple Framework for Contrastive Learning of Visual Representations, Chen et al 2020

SimCLR: Importance of Negatives



A Simple Framework for Contrastive Learning of Visual Representations, Chen et al 2020

Challenges in Contrastive Learning

Contrastive Learning: Negatives are tricky

x pos neg

Contrastive Learning: Negatives are tricky

x

pos

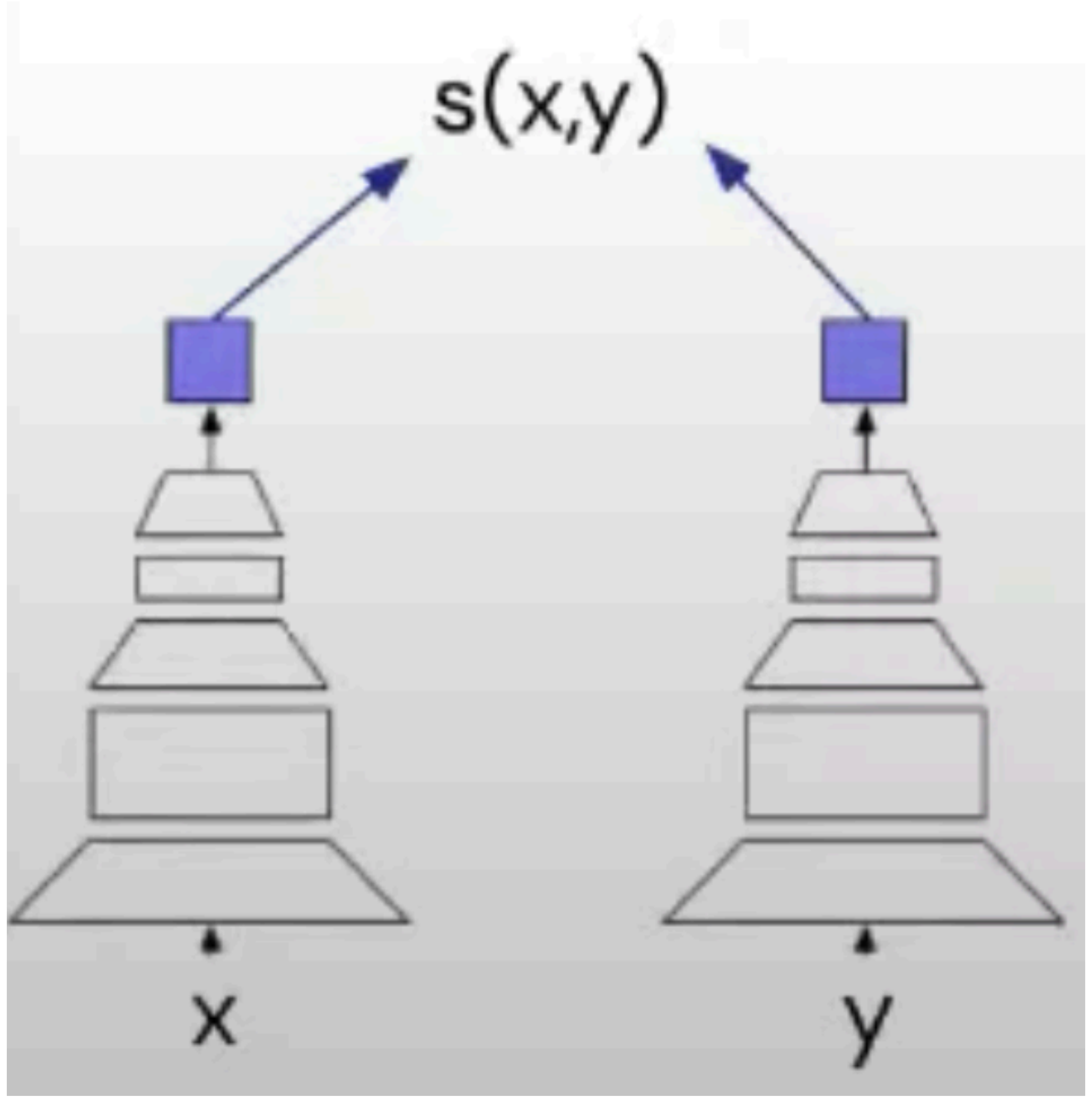
neg

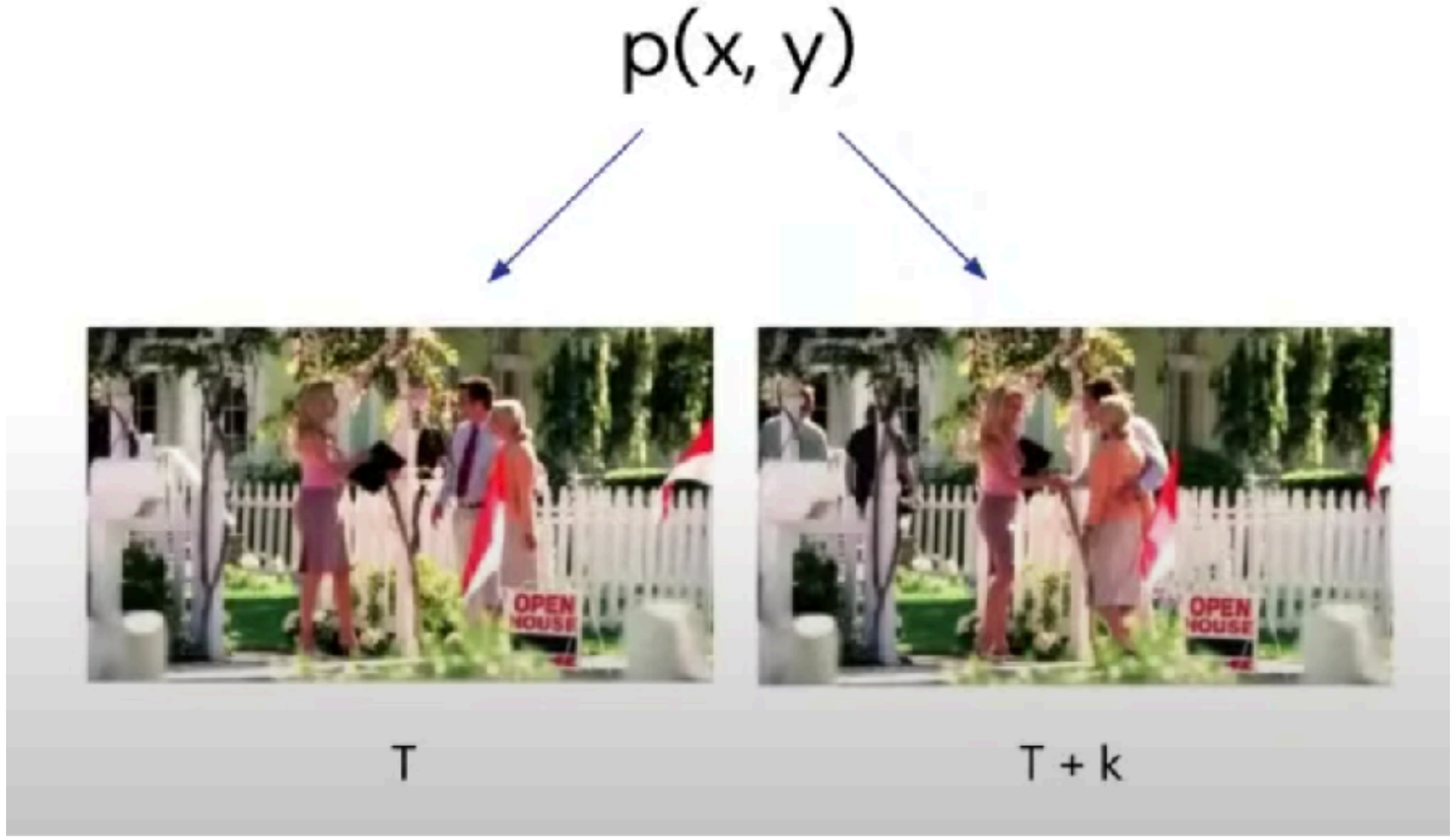
Contrastive Learning: Negatives are tricky

Contrastive Learning: Negatives are tricky

x pos neg

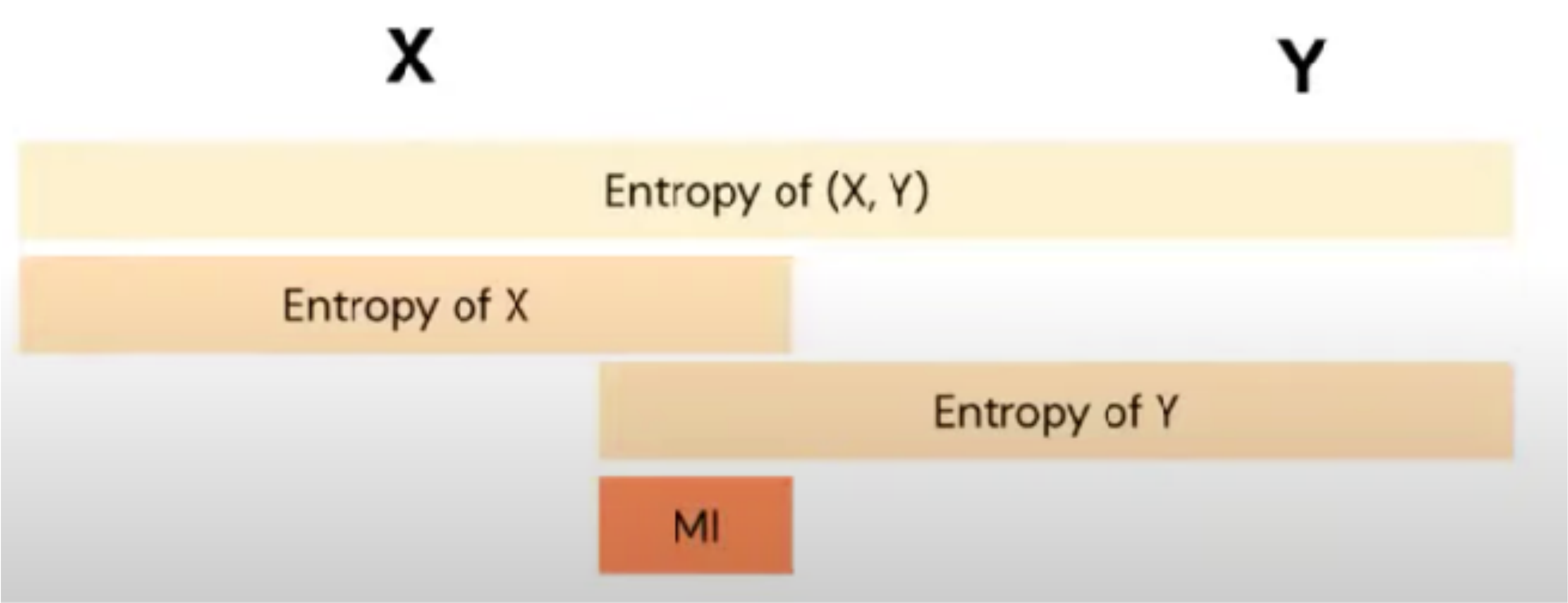
High MI between anchor and positive

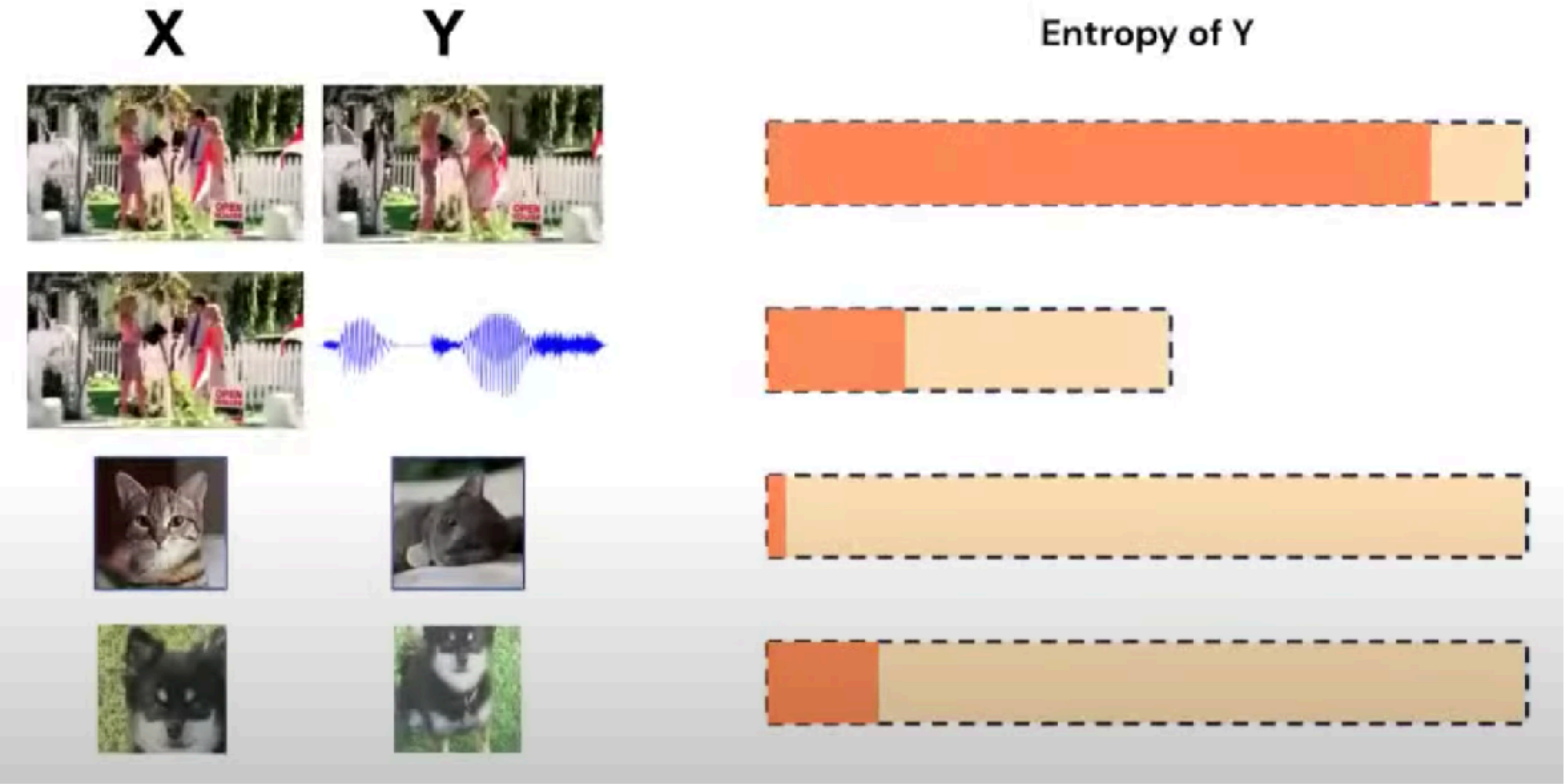




Slide adapted from Aaron van den Oord

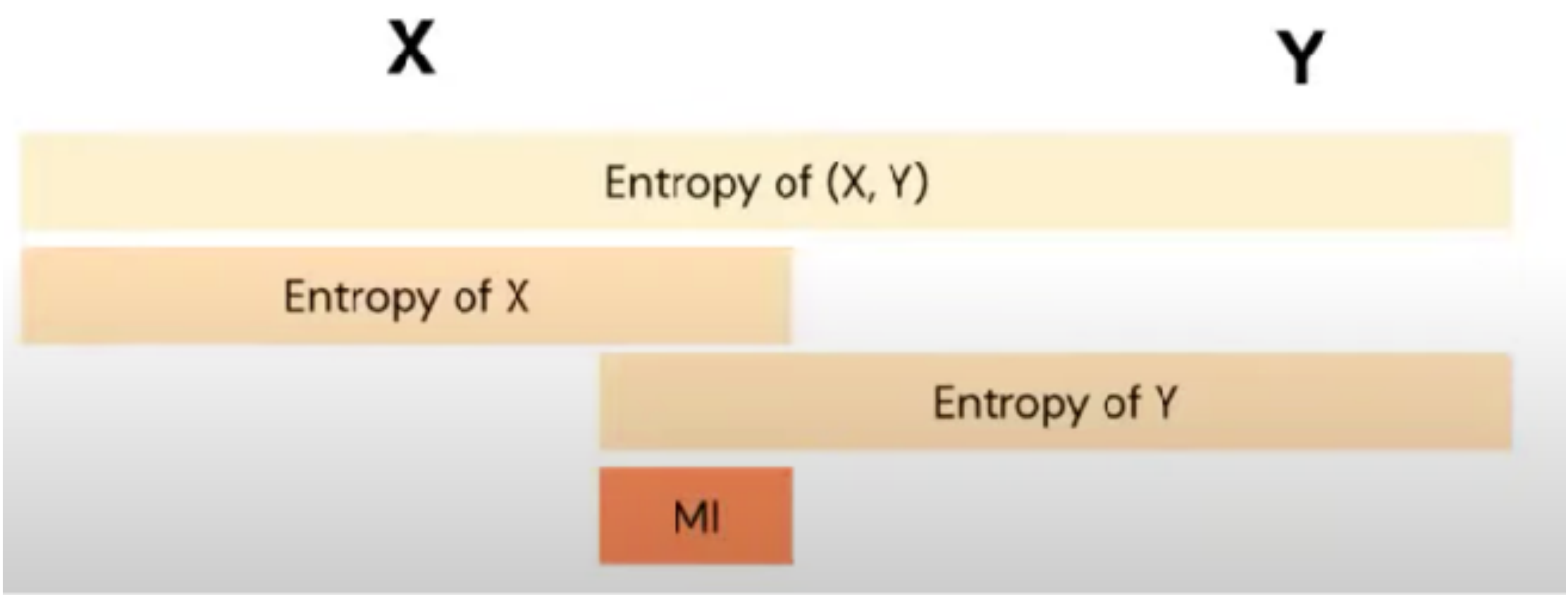
High MI between anchor and positive

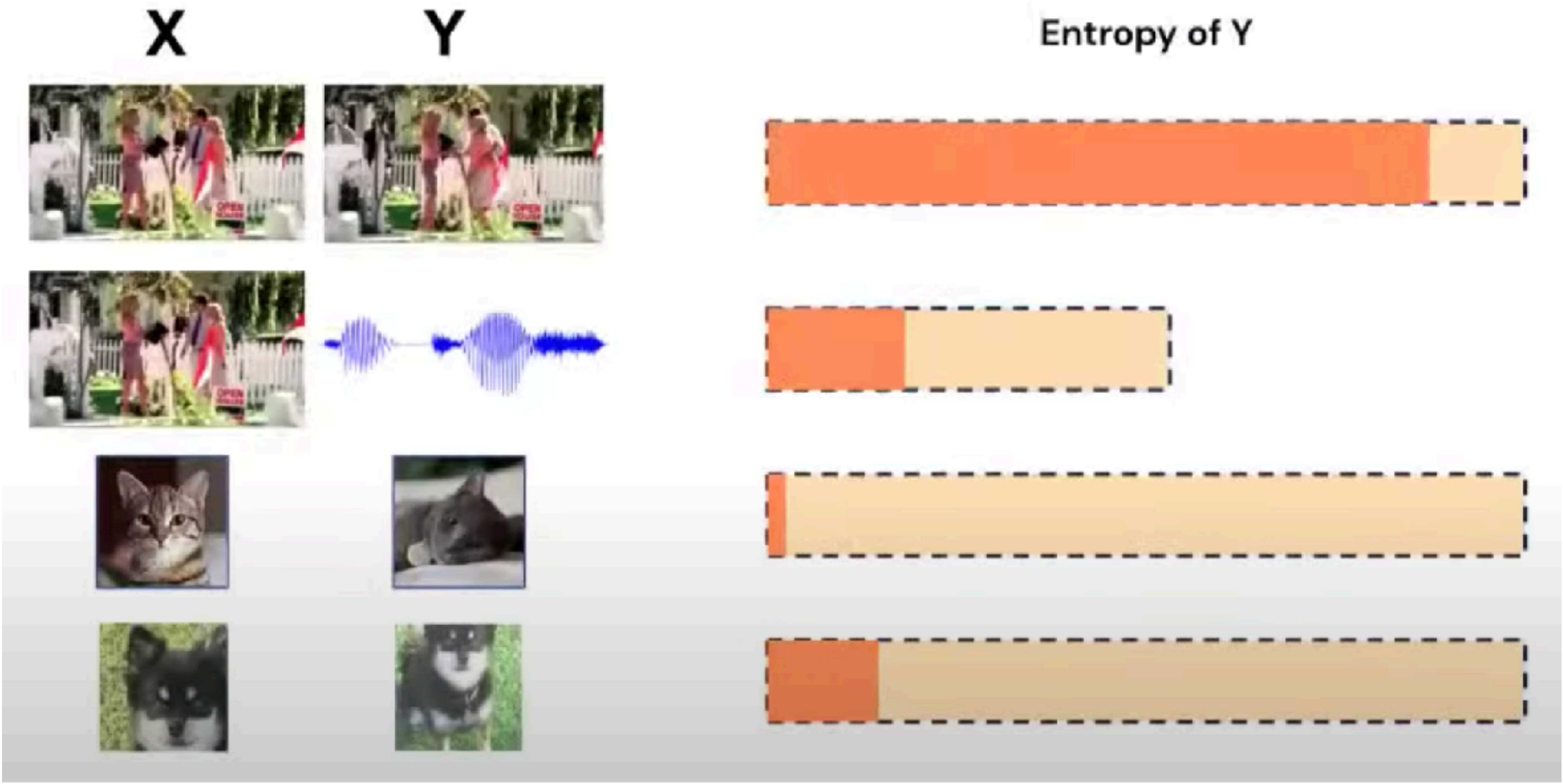




Slide adapted from Aaron van den Oord

High MI between anchor and positive





Please do check out Aaron Van den Oord's talks on this topic

Slide adapted from Aaron van den Oord

Contrastive Learning is just one way to learn Siamese-style Networks

There are approaches that work without negatives (BYOL, SimSiam, DINO, Barlow Twins)

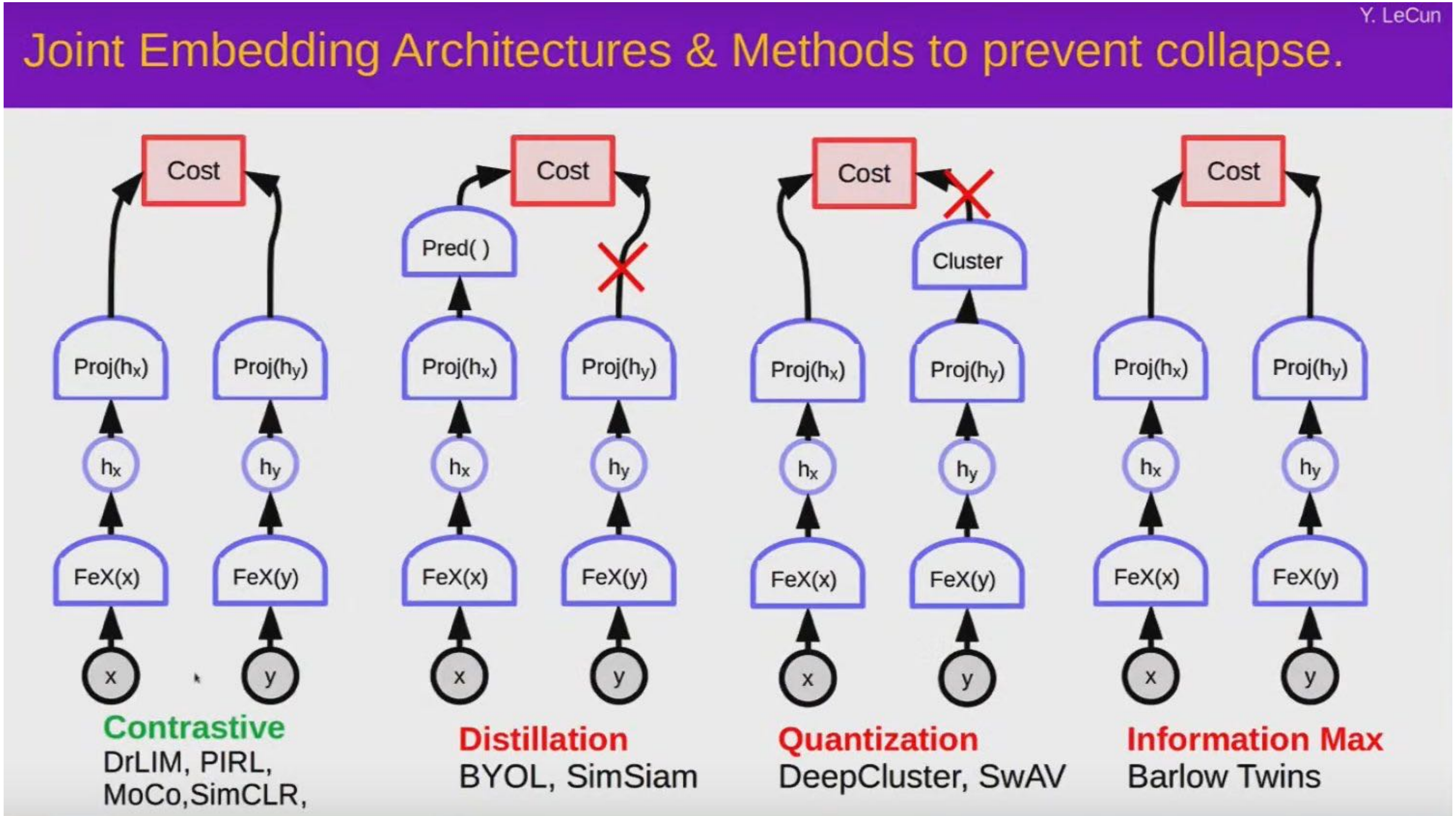
Cluster Swapping using Augmentations (SwaV)

1. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments (Caron et al 2020)

BYOL, SimSiam, DINO

1. Bootstrap Your Own Latent (Grill et al 2020)
2. Exploring Simple Siamese Representation Learning - (Chen & He 2021)
3. Emerging properties in self-supervised vision transformers (Caron et al 2021)

Yann LeCun's summary of all Siamese approaches



Quick background

1. Autoencoder

2. Variational Autoencoder

3. Contrastive Learning

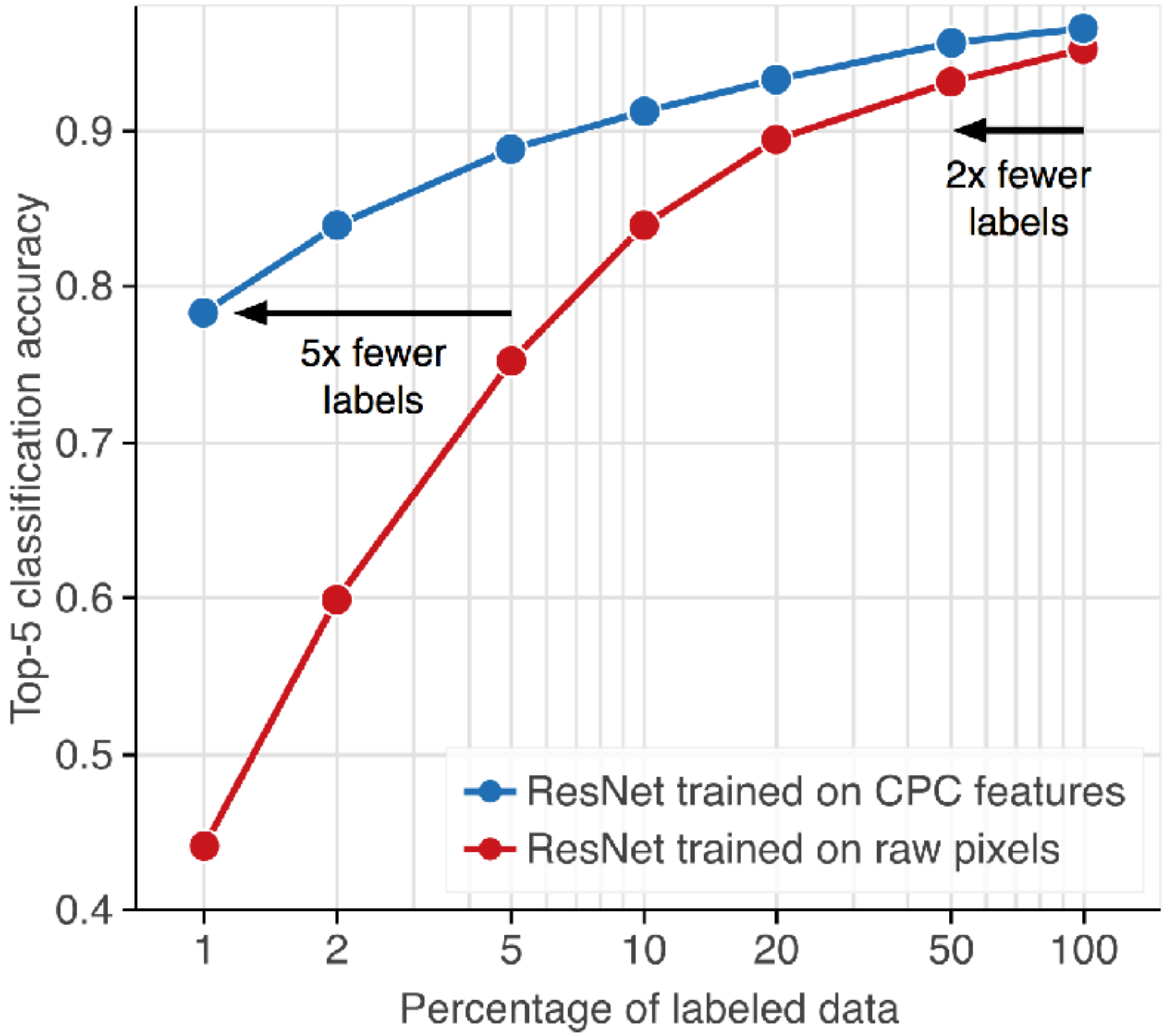
4. Siamese Networks

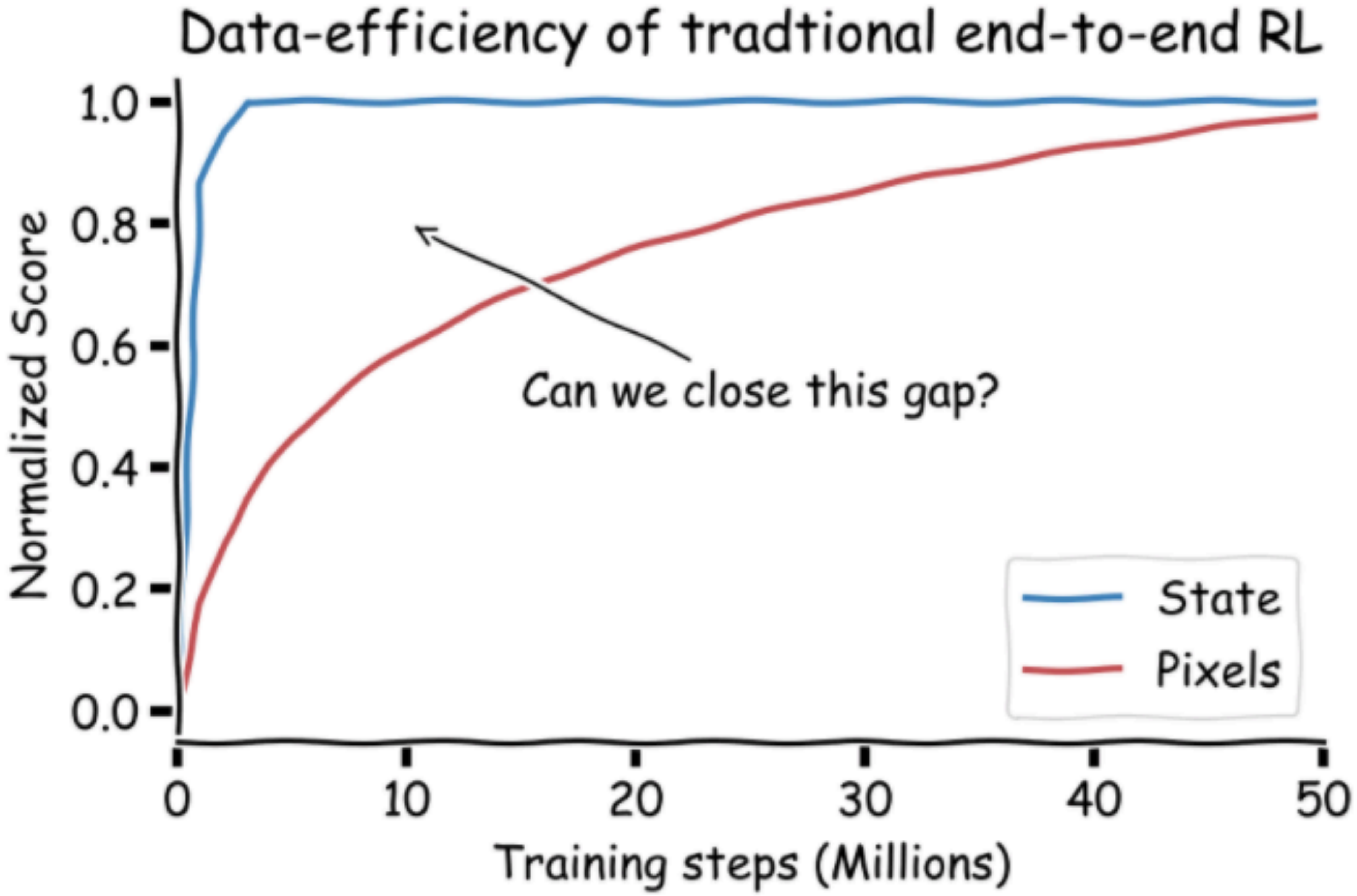
5. Data-Augmentations

Contrastive-like UL

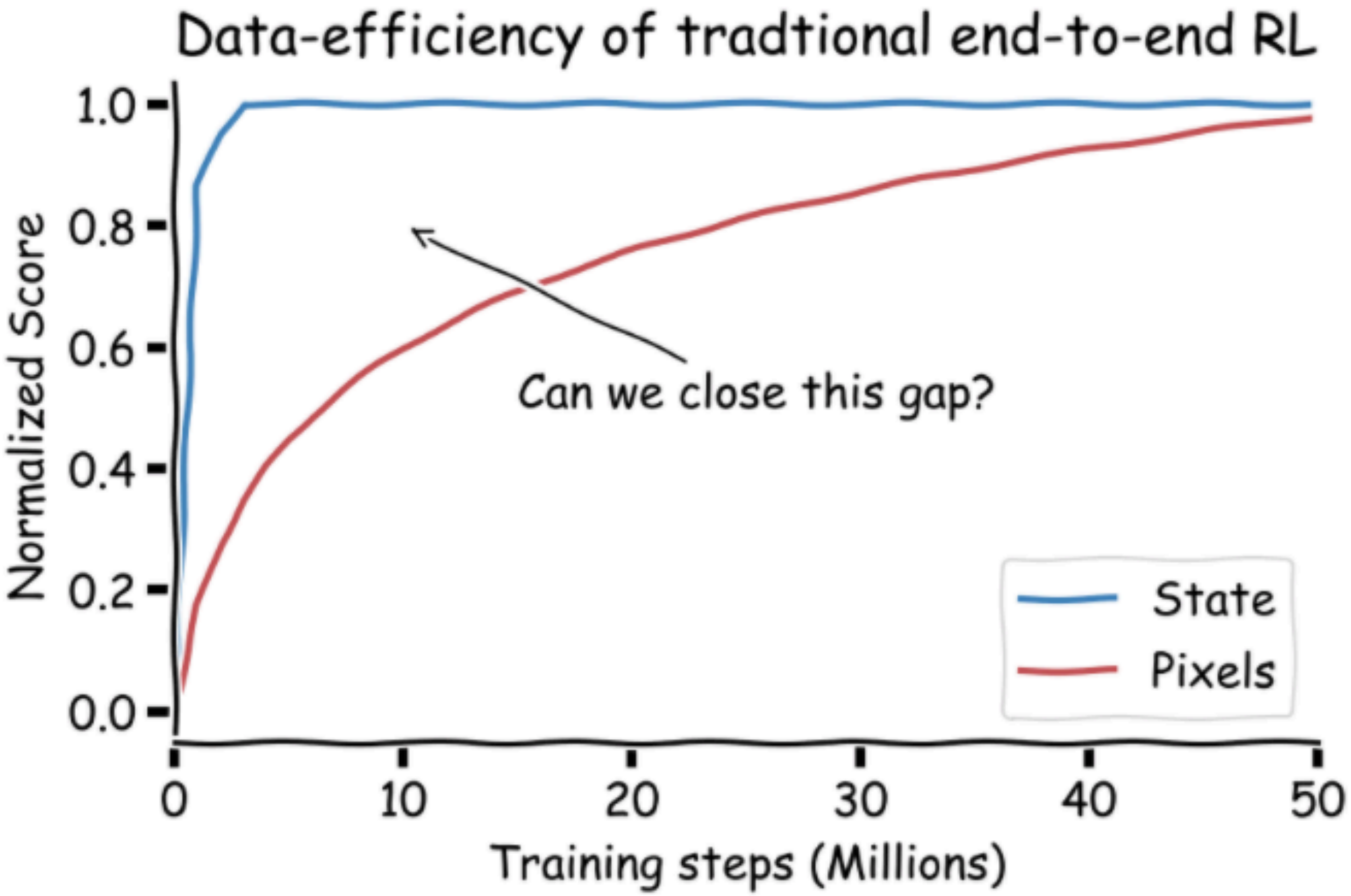
How have people applied Contrastive Learning, Siamese Nets, Data Augmentations, etc in Reinforcement Learning?

Capture useful aspects of high dimensional sensory stream

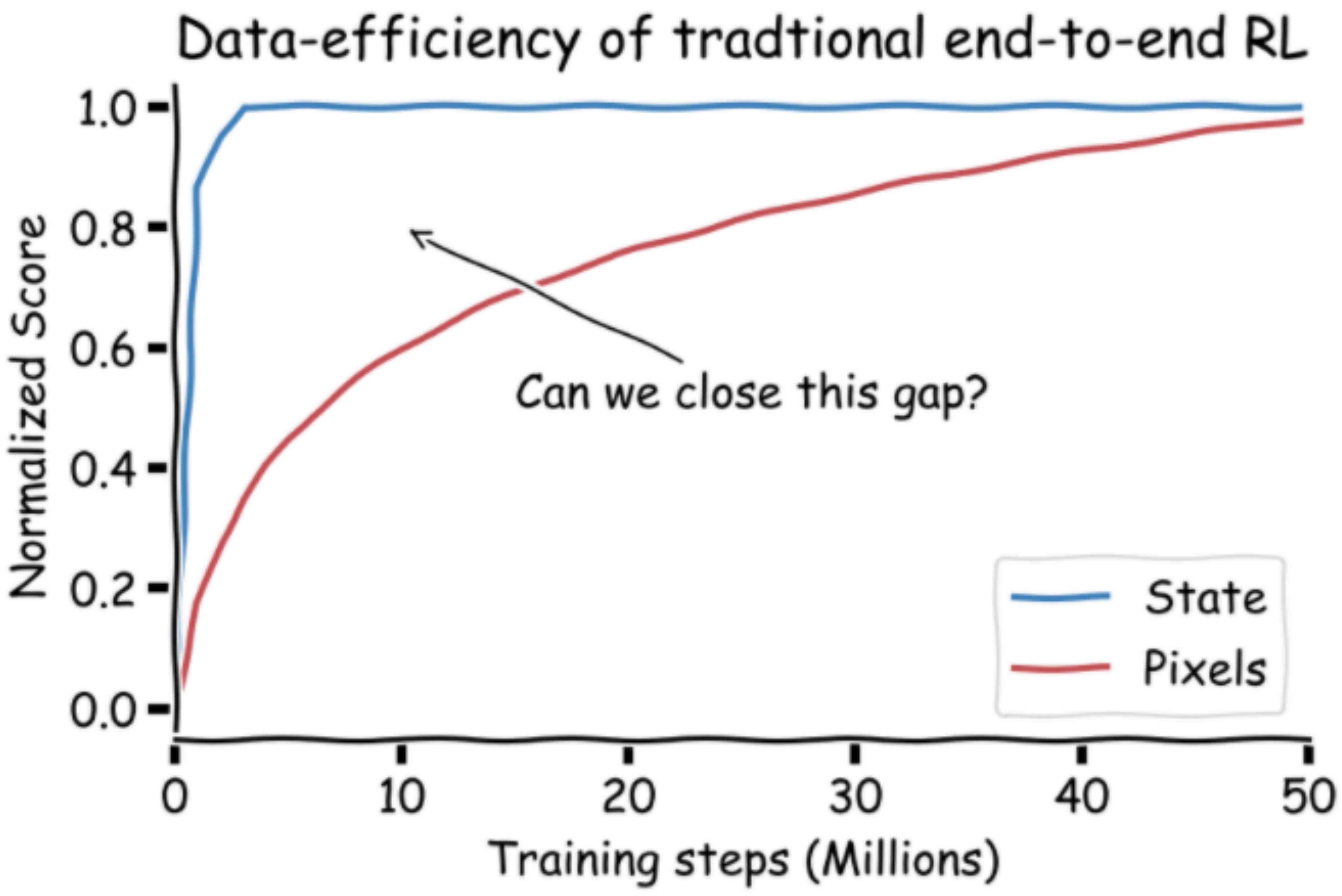




Data-Efficiency in Reinforcement Learning



Data-Efficiency in Reinforcement Learning

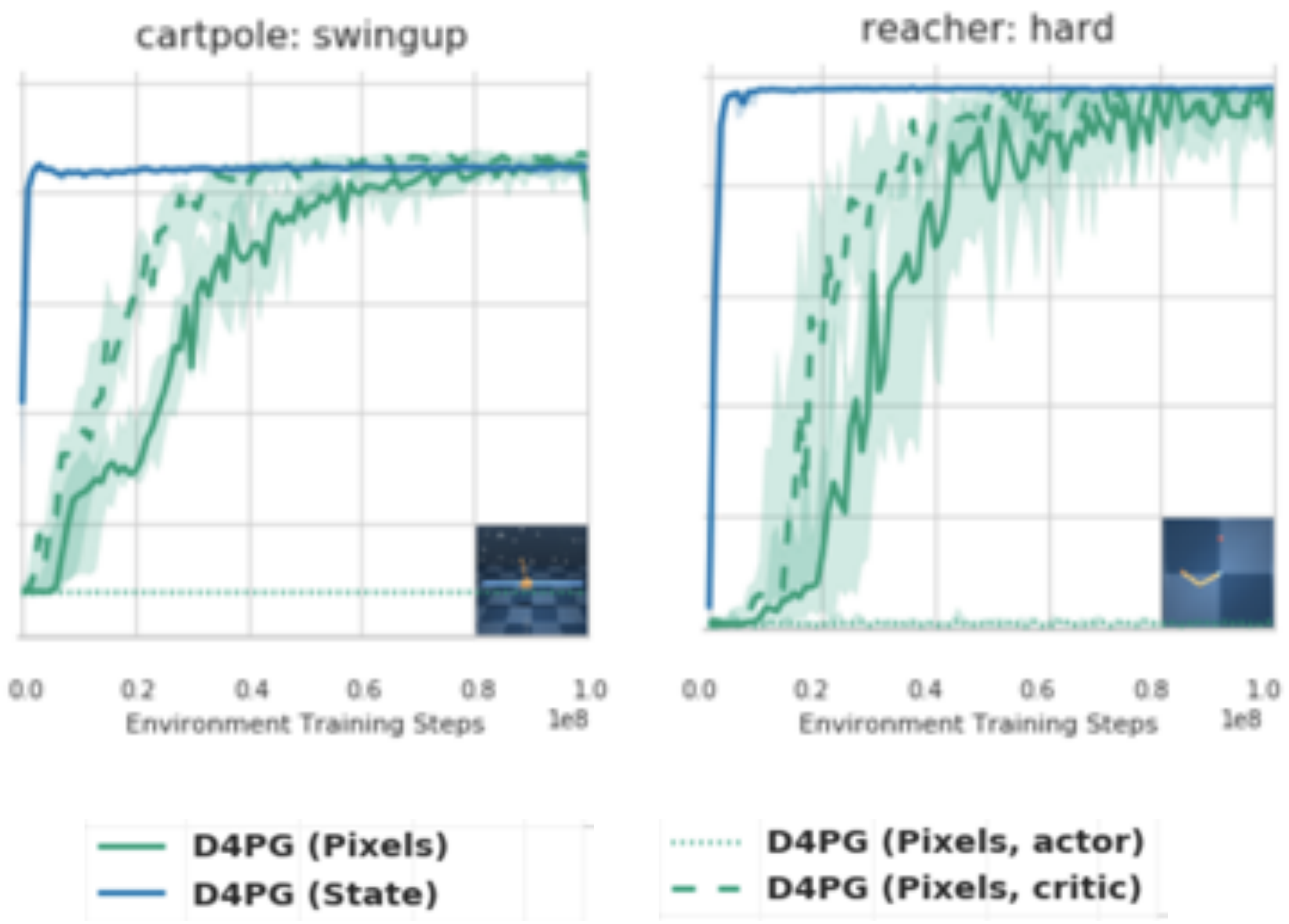


1. Code becomes a lot simpler. Less feature engineering.
2. Calibration, sim2real etc become a lot simpler.
3. Take advantage of all the DL scale infra that's been built for vision and NLP, for robotics.
4. Sensor costs.
5. MLPs are not as parameter and FLOP efficient as transformers and ConvNets.

Data-Efficiency in Reinforcement Learning

60M training steps

60M training steps

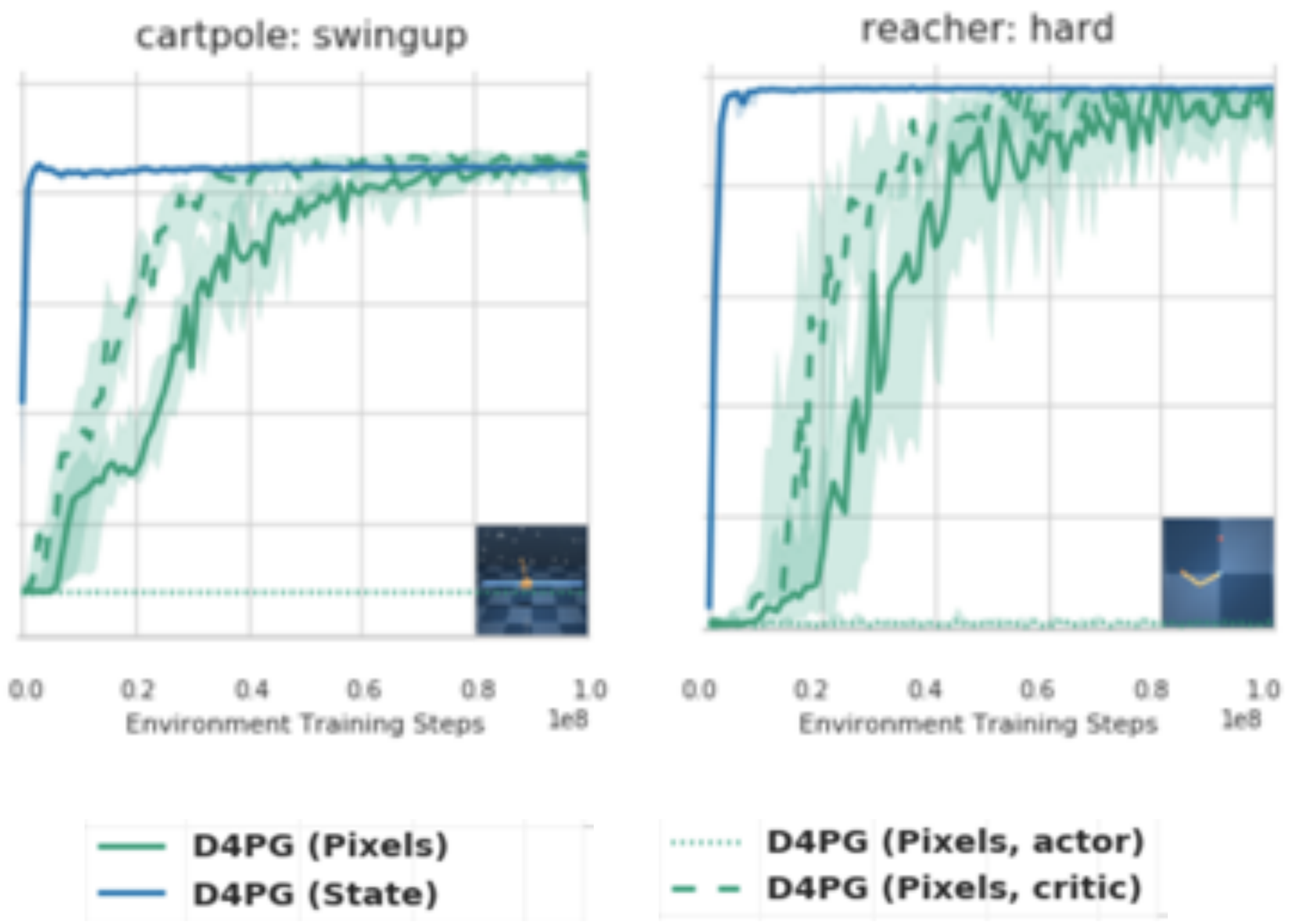


[Tassa et al., 2018] Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D.D.L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A. and Lillicrap, T DeepMind Control Suite, arxiv:1801.00690, 2018.

Data-Efficiency in Reinforcement Learning

60M training steps

60M training steps



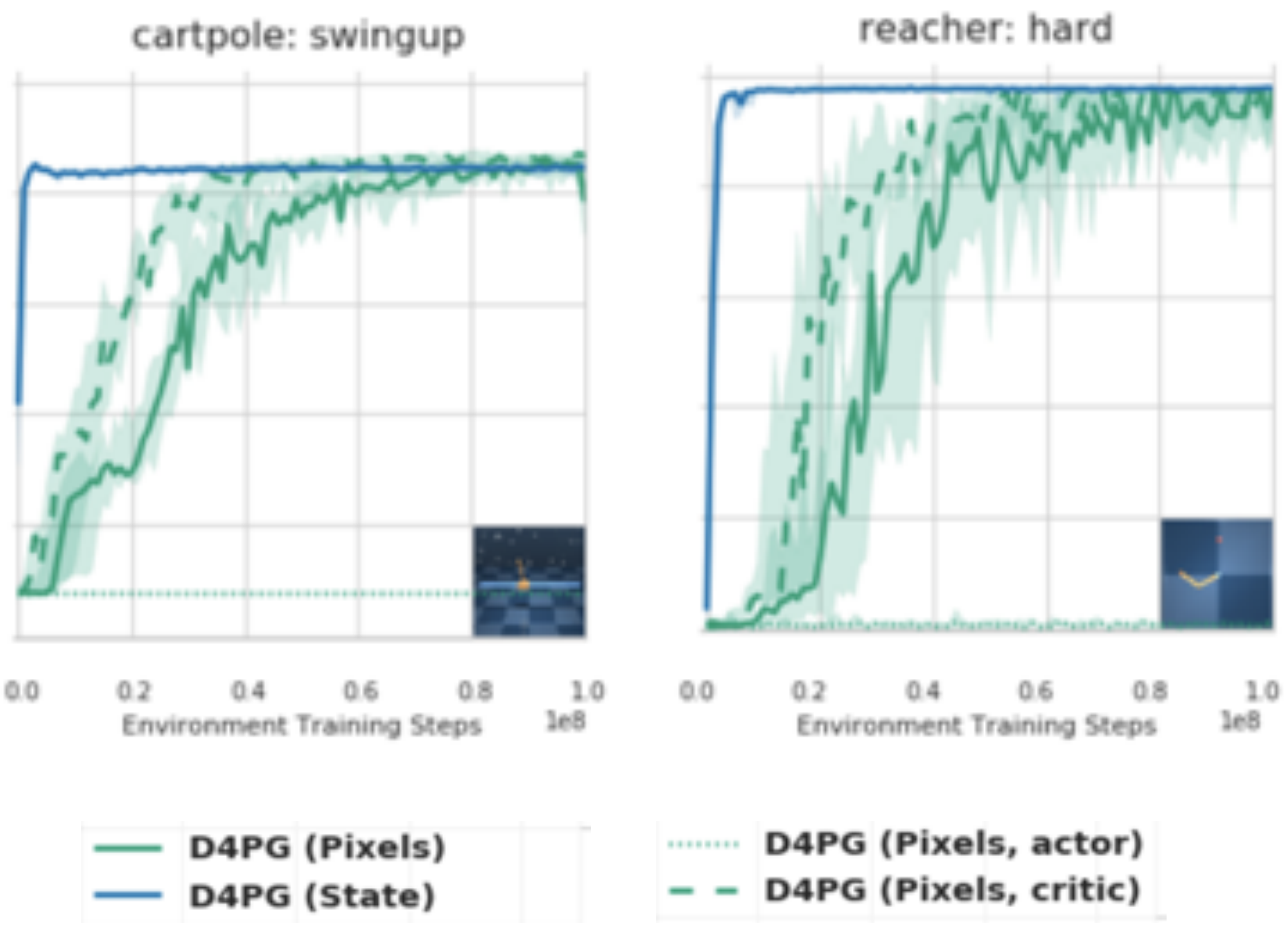
Blue: RL from state,
Green: RL from pixels

[Tassa et al., 2018] Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D.D.L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A. and Lillicrap, T [DeepMind Control Suite](), arxiv:1801.00690, 2018.

Data-Efficiency in Reinforcement Learning

60M training steps

60M training steps



Blue: RL from state,
Green: RL from pixels

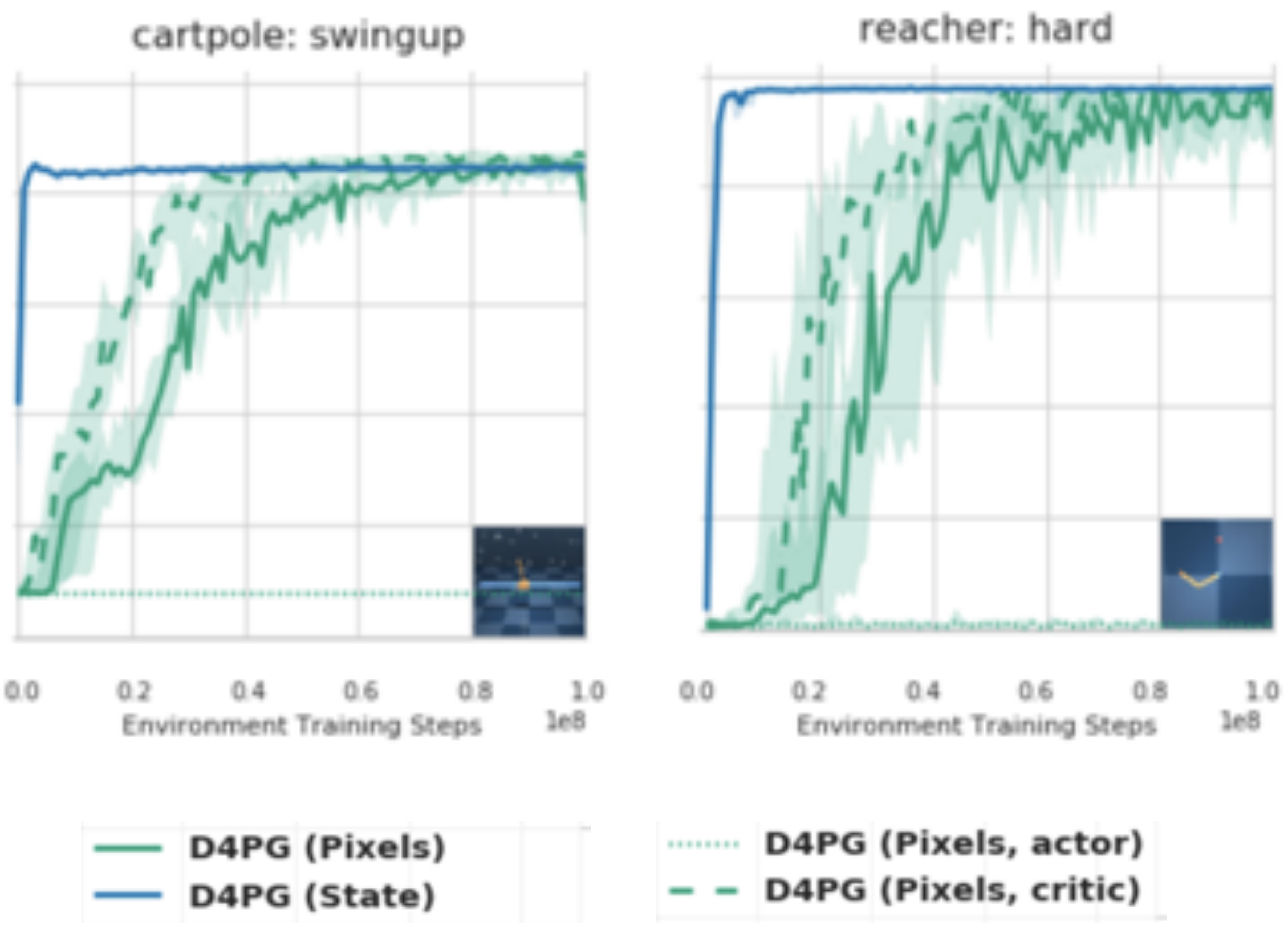
Difference in sample complexity is at least > 50M training steps

[Tassa et al., 2018] Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D.D.L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A. and Lillicrap, T DeepMind Control Suite, arxiv:1801.00690, 2018.

Data-Efficiency in Reinforcement Learning

60M training steps

60M training steps



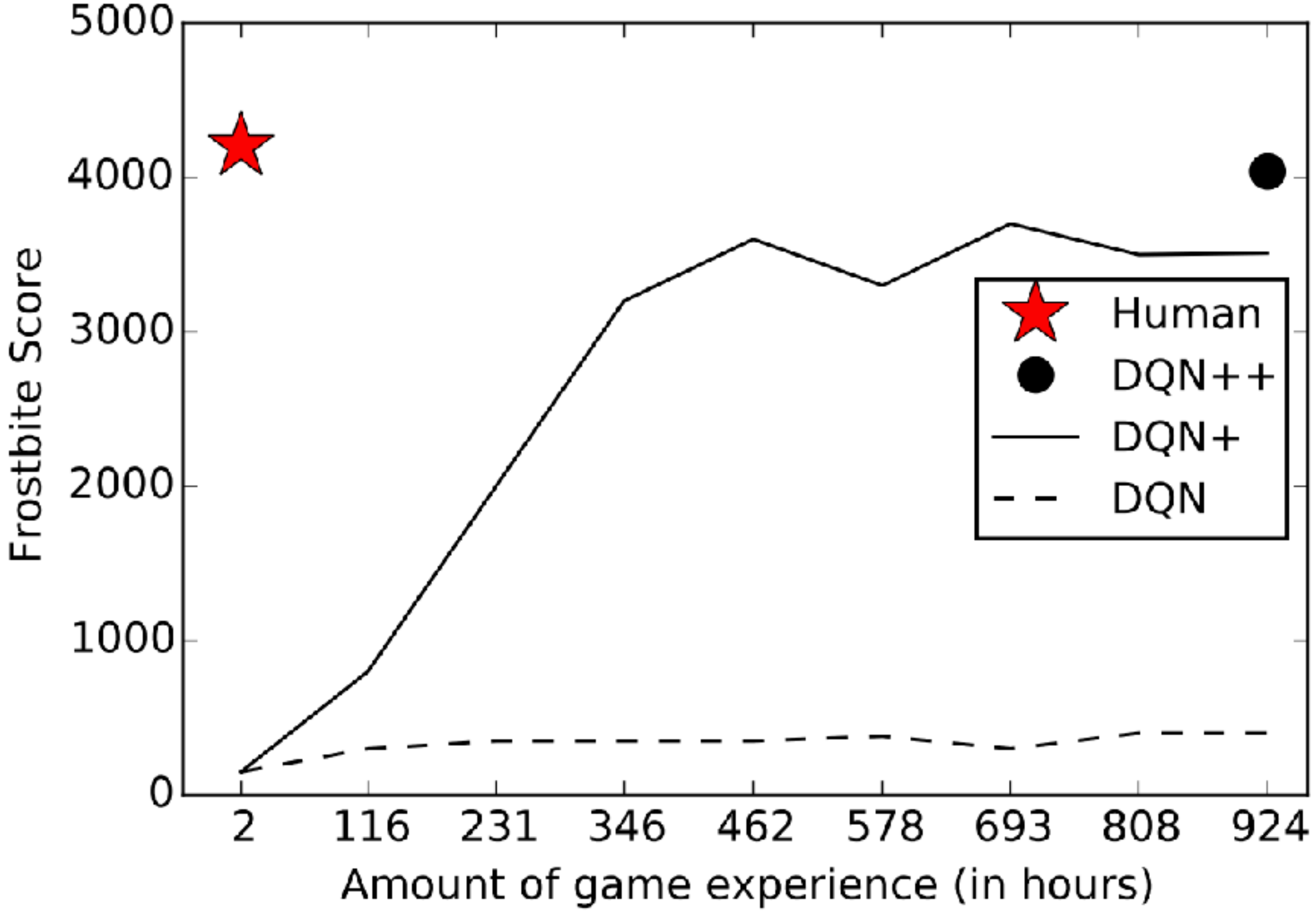
Blue: RL from state,
Green: RL from pixels

Difference in sample complexity is at least > 50M training steps

Can contrastive learning fix this?

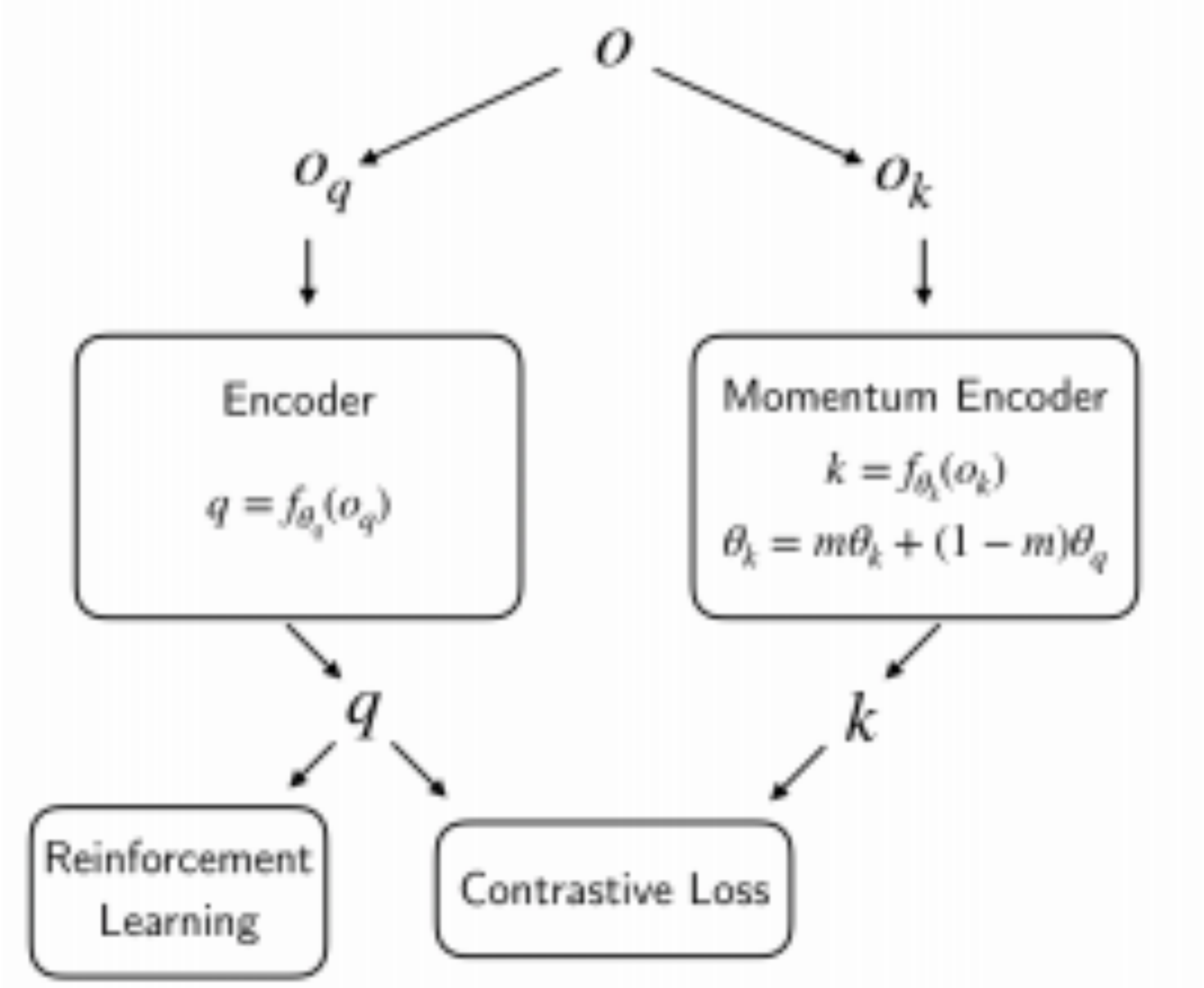
[Tassa et al., 2018] Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D.D.L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A. and Lillicrap, T DeepMind Control Suite, arxiv:1801.00690, 2018.

Data-Efficiency in Reinforcement Learning



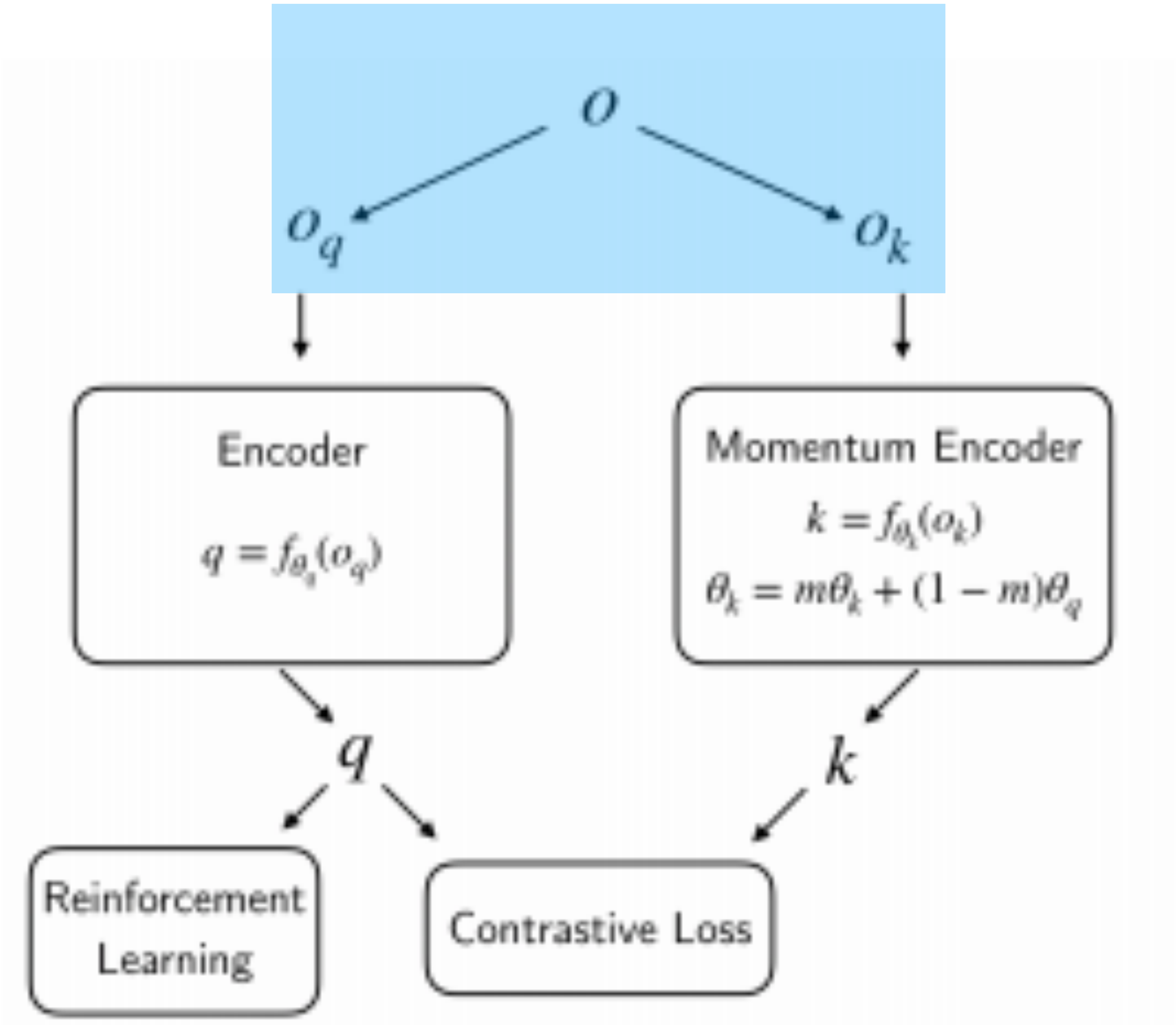
Building machines that can learn and think like people, Lake et al 2016

Contrastive Learning + Reinforcement Learning



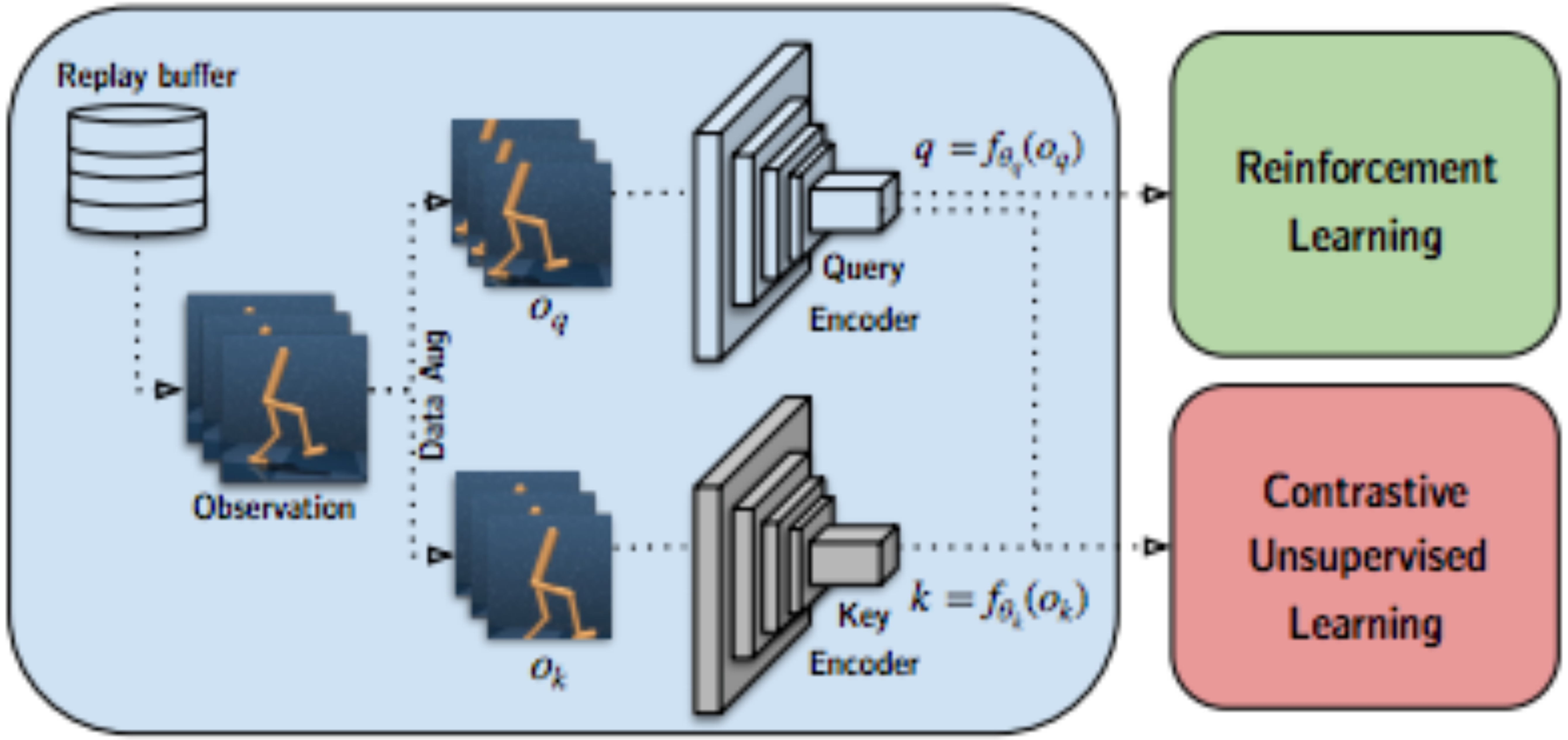
Contrastive Unsupervised Representations for Reinforcement Learning (Srinivas, Laskin, Abbeel; ICML 2020)

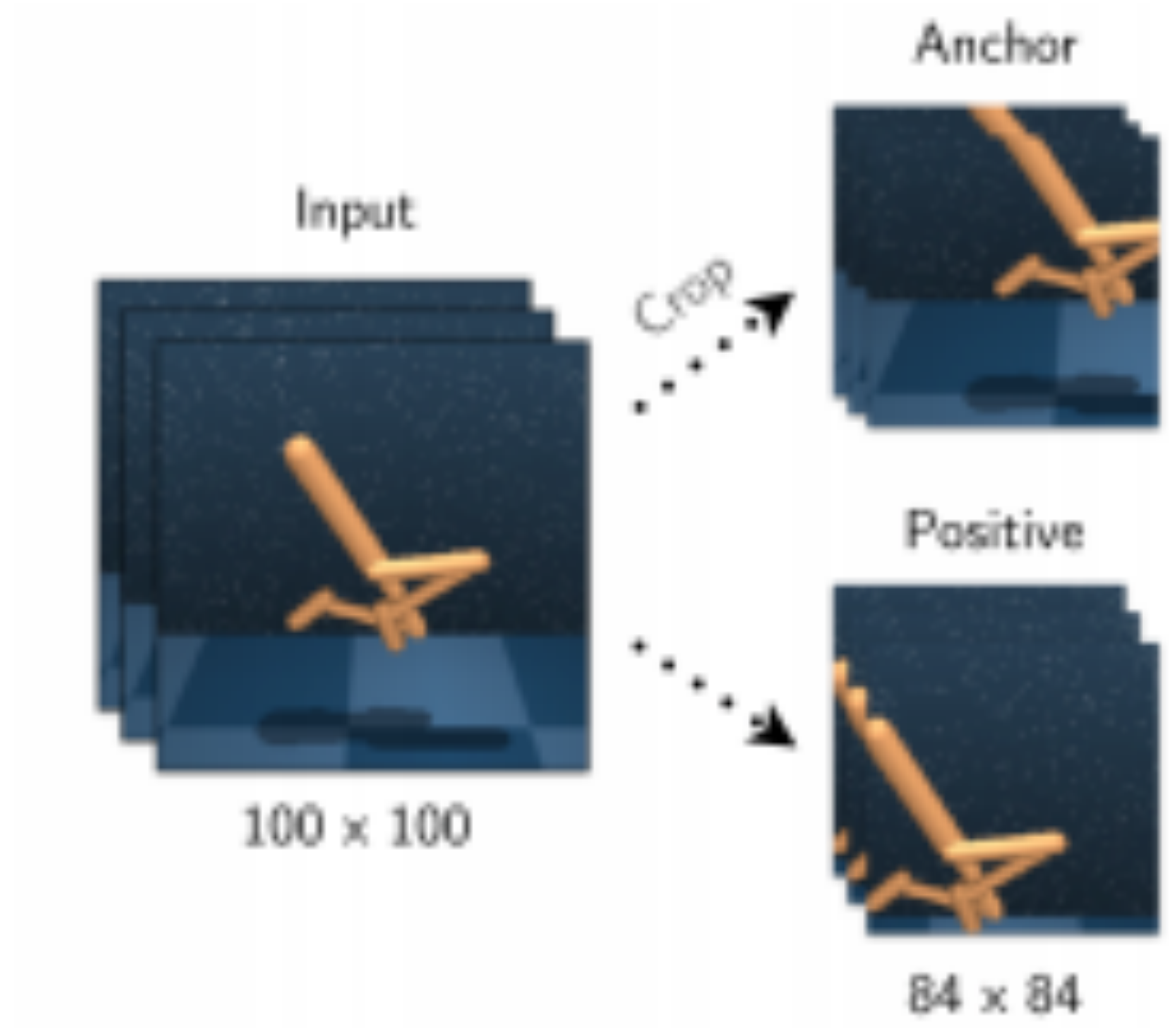
Contrastive Learning + Reinforcement Learning



Contrastive Unsupervised Representations for Reinforcement Learning (Srinivas, Laskin, Abbeel; ICML 2020)

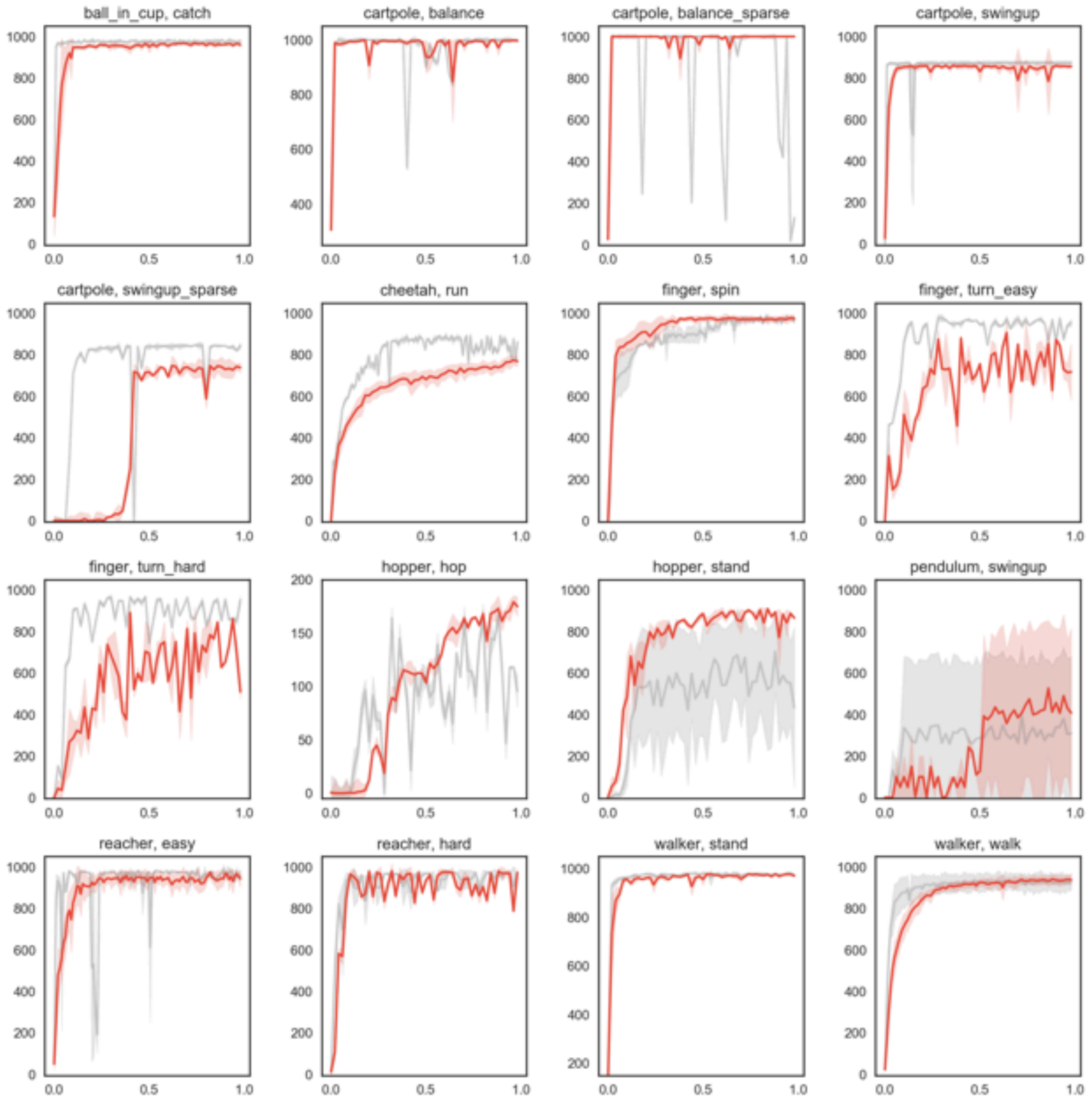
Contrastive Learning + Reinforcement Learning





1. Stacked Frames instead of single image
2. Temporally Consistent Spatial Random Crop
3. Common practice in Video Recognition

Contrastive Learning + Reinforcement Learning



GRAY: SAC State
RED: CURL

Contrastive Learning + Reinforcement Learning

500K step scores	CURL	PlaNet	Dreamer	SAC+AE	SLAC	Pixel SAC	State SAC
Finger, spin	**971 ± 18**	693 ± 27	343 ± 43	884 ± 128	892 ± 130	509 ± 148	932 ± 32
Cartpole, swingup	**853 ± 10**	794 ± 14	688± 207	735 ± 63	-	382 ± 79	870 ± 11
Reacher, easy	**945 ± 27**	833 ± 101	480 ± 128	627 ± 58	-	201 ± 94	944 ± 30
Cheetah, run	**694 ± 42**	608 ± 20	628 ± 32	550 ± 34	617 ± 14	292 ± 31	826 ± 22
Walker, walk	**925 ± 21**	912 ± 35	865 ± 44	847 ± 48	877 ± 54	226 ± 15	935 ± 31
Ball in cup, catch	**956 ± 18**	725 ± 309	955 ± 19	794± 58	900 ± 181	118± 92	984 ± 16
100K step scores							
Finger, spin	**845 ± 42**	632 ± 112	141 ± 73	740 ± 64	693 ± 141	403 ± 67	741±54
Cartpole, swingup	**855±8**	498±95	288±167	311±11	-	119±22	869±10
Reacher, easy	**819±72**	336±122	134±64	274±14	-	120±23	966±29
Cheetah, run	**495±31**	294±98	113±25	267±24	296±34	49±14	617±9
Walker, walk	**715±45**	290±111	102±12	394±22	529±47	76±19	884±64
Ball in cup, catch	**942 ± 27**	405 ± 375	298 ± 40	391± 82	834 ± 128	131± 52	979±14

Contrastive Learning + Reinforcement Learning

500K STEP SCORES	CURL	PLANET	DREAMER	SAC+AE	SLAC	PIXEL SAC	STATE SAC
FINGER, SPIN	**971** ± **18**	693 ± 27	343 ± 43	884 ± 128	892 ± 130	509 ± 148	932 ± 32
CARTPOLE, SWINGUP	**853** ± **10**	794 ± 14	688± 207	735 ± 63	-	382 ± 79	870 ± 11
REACHER, EASY	**945** ± **27**	833 ± 101	480 ± 128	627 ± 58	-	201 ± 94	944 ± 30
CHEETAH, RUN	**694** ± **42**	608 ± 20	628 ± 32	550 ± 34	617 ± 14	292 ± 31	826 ± 22
WALKER, WALK	**925** ± **21**	912 ± 35	865 ± 44	847 ± 48	877 ± 54	226 ± 15	935 ± 31
BALL IN CUP, CATCH	**956** ± **18**	725 ± 309	955 ± 19	794± 58	900 ± 181	118± 92	984 ± 16
100K STEP SCORES							
FINGER, SPIN	**845** ± **42**	632 ± 112	141 ± 73	740 ± 64	693 ± 141	403 ± 67	741±54
CARTPOLE, SWINGUP	**855**±**8**	498±95	288±167	311±11	-	119±22	869±10
REACHER, EASY	**819**±**72**	336±122	134±64	274±14	-	120±23	966±29
CHEETAH, RUN	**495**±**31**	294±98	113±25	267±24	296±34	49±14	617±9
WALKER, WALK	**715**±**45**	290±111	102±12	394±22	529±47	76±19	884±64
BALL IN CUP, CATCH	**942** ± **27**	405 ± 375	298 ± 40	391± 82	834 ± 128	131± 52	979±14

Planet, Dreamer, SLAC, SAC+AE are strong but relatively complex baselines.

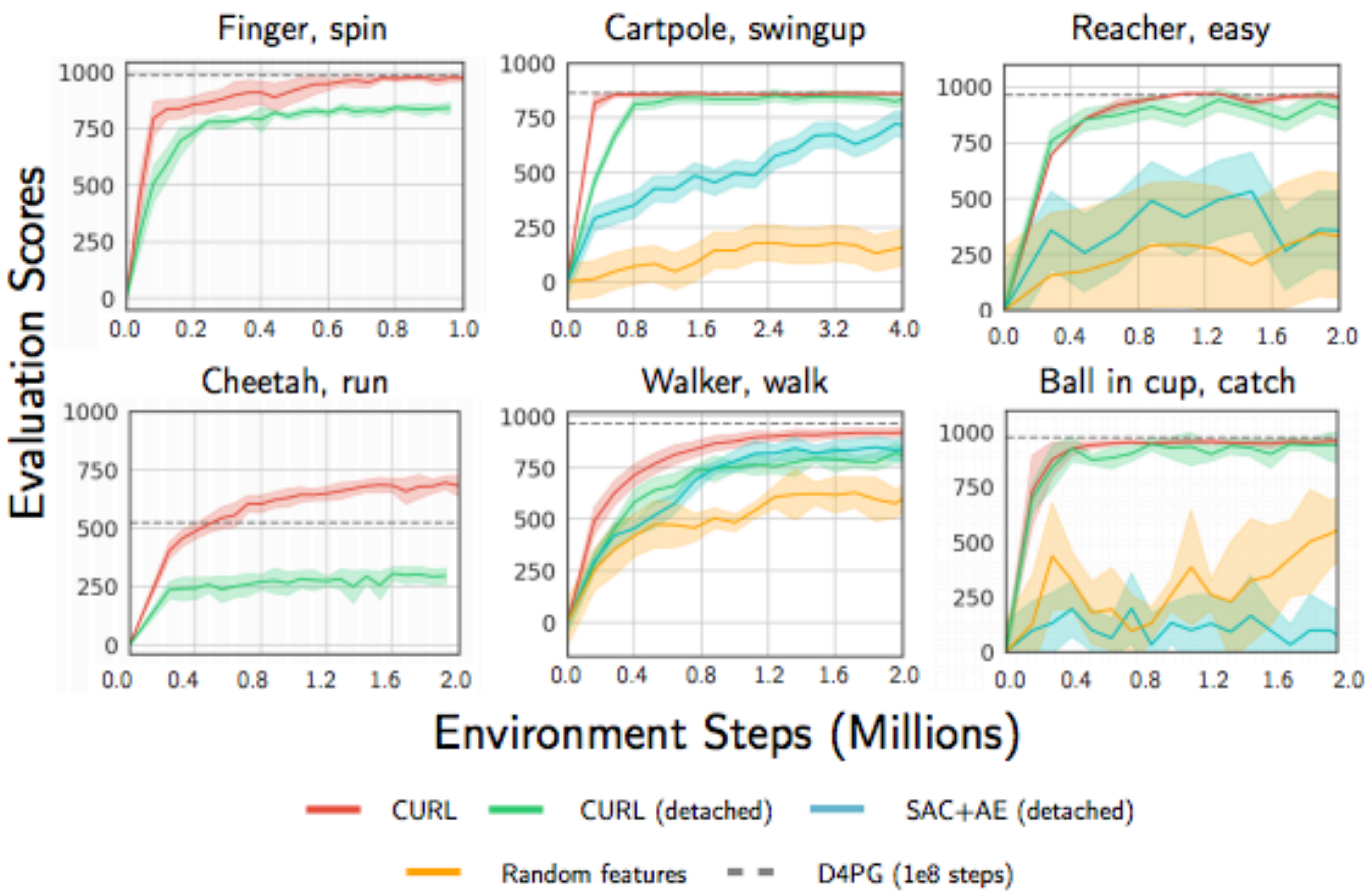
All of them reconstruct pixels (either autoencoder or future frames). Contrastive just learns what's necessary and high-level.

CURL beats Planet, Dreamer, SLAC (model-based methods) despite being model-free and minimal.

Decoupled Representation Learning

Just learn the encoder with contrastive, and do not backpropogate the RL loss to the encoder.
Towards a Feature Layer Moment (promise, not yet happened) for RL from pixels.

The Le Cake (Figure from David Ha)

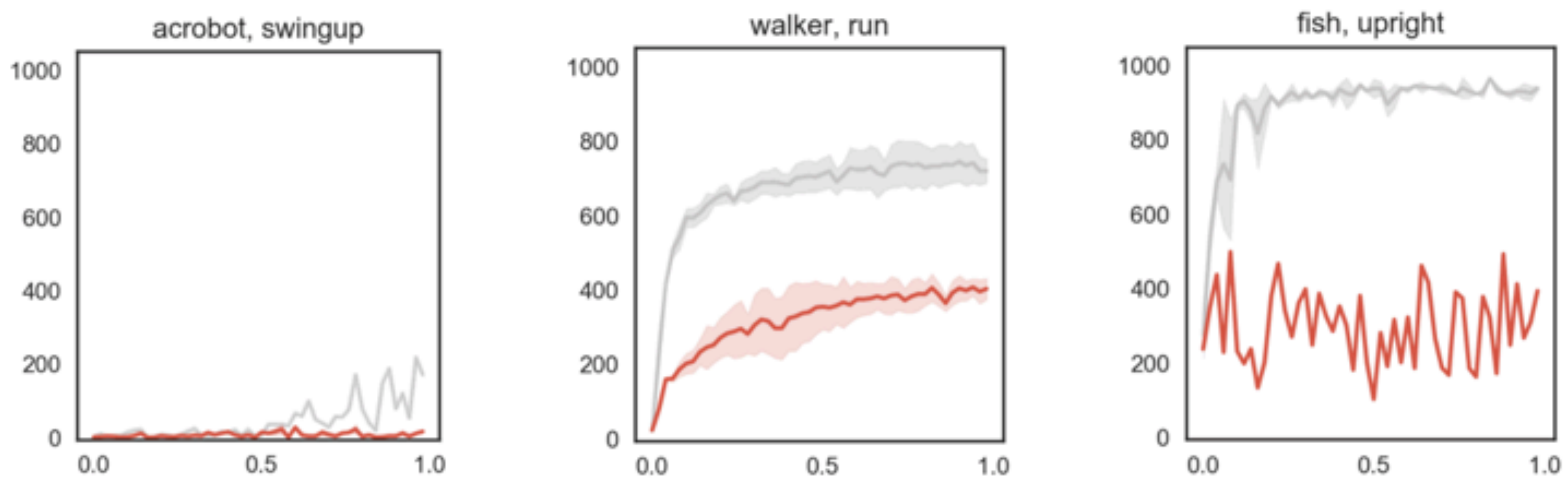


On 4/6 tasks, it is almost as good. On remaining 2, it is not (ex Cheetah Run, Walk Walk).

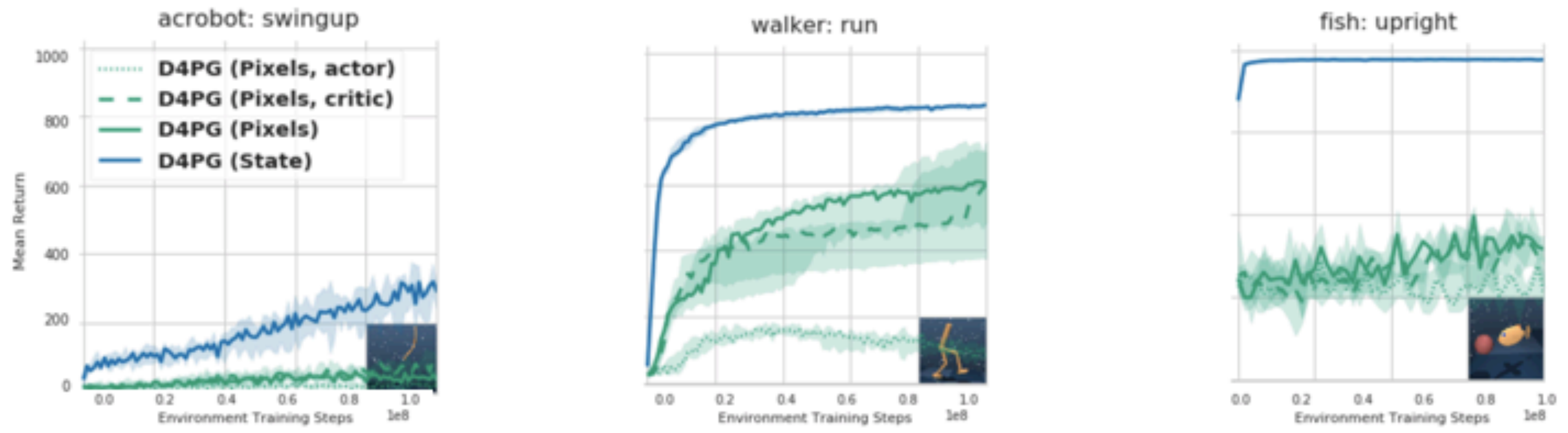
Failure Cases

GRAY: SAC State
RED: CURL

Environment training steps 1 = 1M

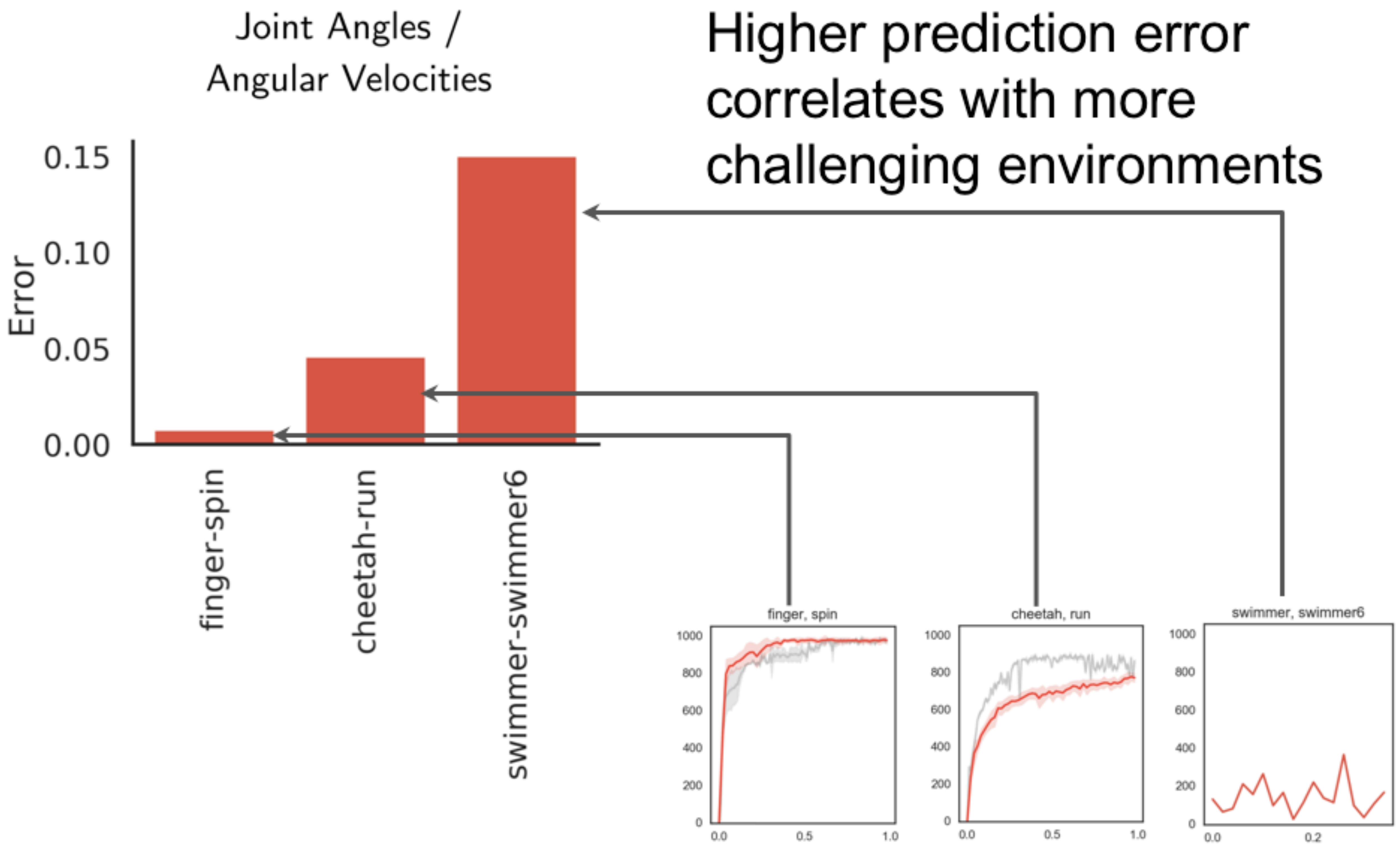


Environment training steps 1 = 100M



Not good on tasks with high frequency dynamics, for example, running, swimming, acrobot, etc.

Failure Cases



The representation learning struggles on these harder environments.

Self-Predictive Representations for RL

Data-Efficient Reinforcement Learning with Self-Predictive Representations

Max Schwarzer* Mila, Université de Montréal

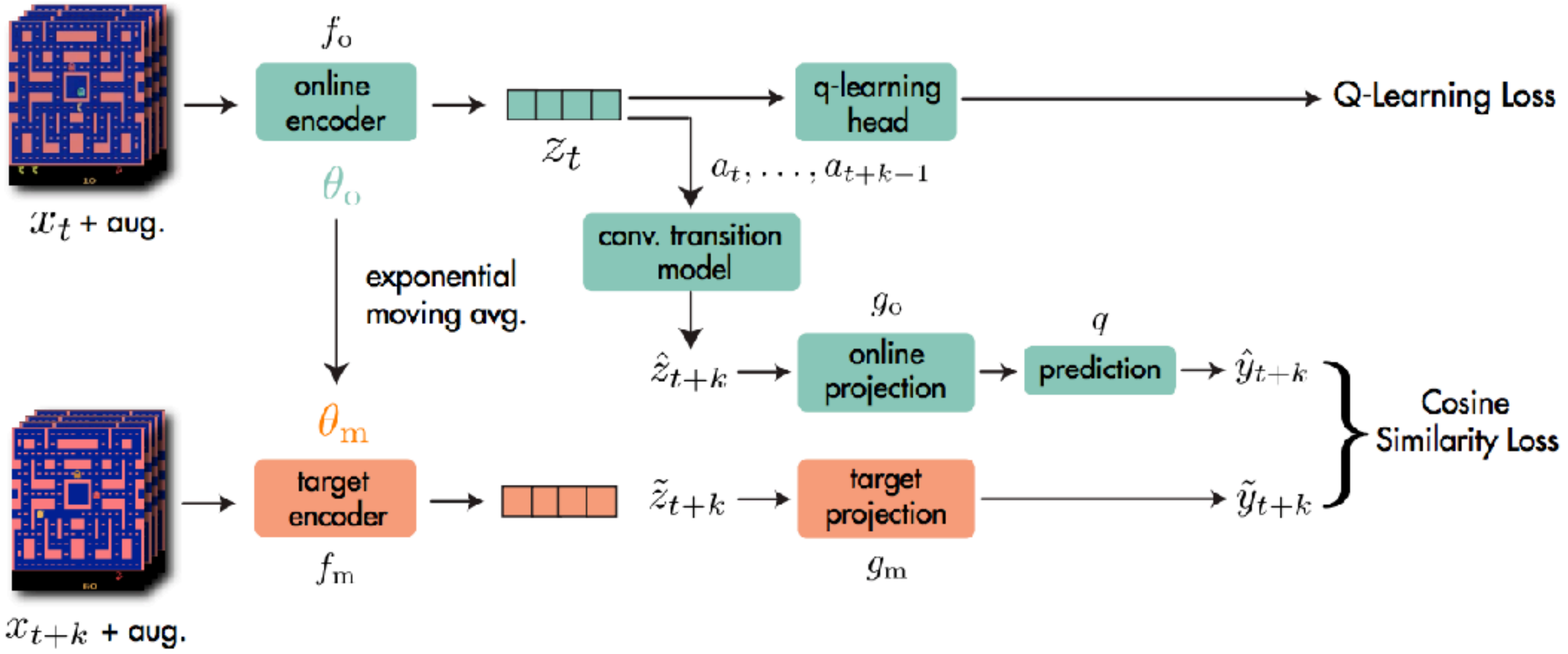
Ankesh Anand* Mila, Université de Montréal, Microsoft Research

Rishab Goel Mila

R Devon Hjelm Microsoft Research, Mila, Université de Montréal

Aaron Courville Mila, Université de Montréal, CIFAR Fellow

Philip Bachman Microsoft Research



Self-Predictive Representations for RL

Data-Efficient Reinforcement Learning with Self-Predictive Representations

Max Schwarzer*
Mila, Université de Montréal

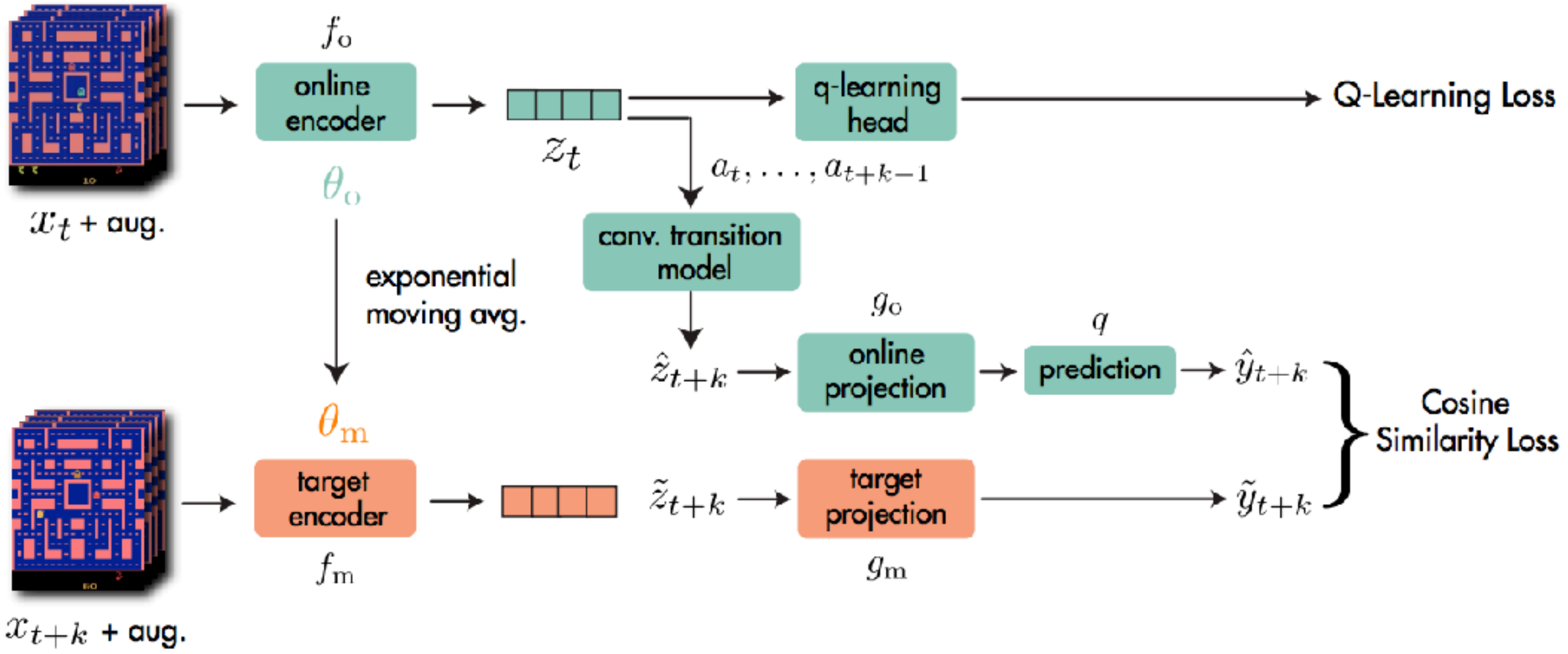
Ankesh Anand*
Mila, Université de Montréal
Microsoft Research

Rishab Goel
Mila

R Devon Hjelm
Microsoft Research
Mila, Université de Montréal

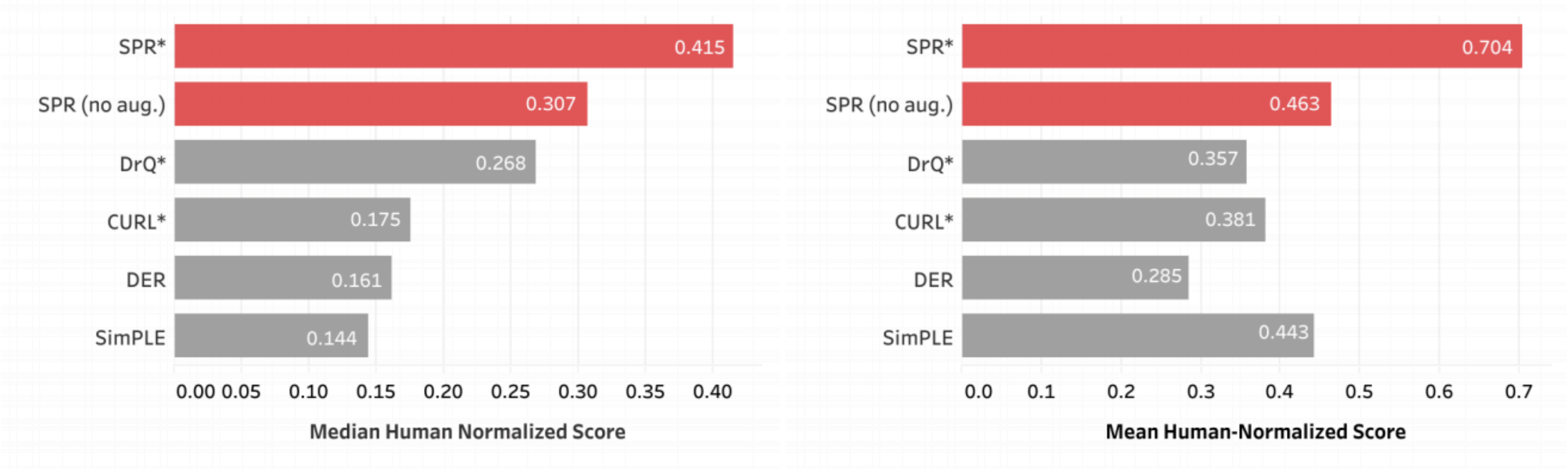
Aaron Courville
Mila, Université de Montréal
CIFAR Fellow

Philip Bachman
Microsoft Research



Temporal information helps. Predict the future from past. (CPC-like, but executed in BYOL fashion)

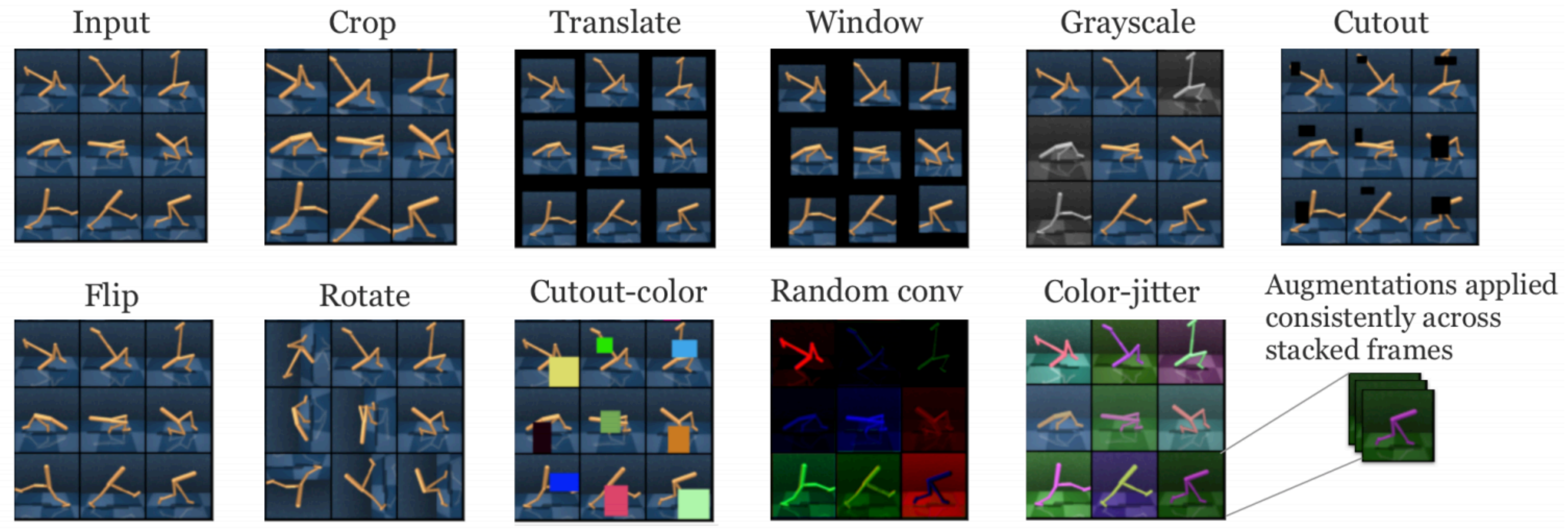
Self-Predictive Representations for RL



On an average, contrastive learning (or like methods) help get to 70% data-efficiency as a human.

Data-Efficient Reinforcement Learning with Self-Predictive Representations, Schwarzer & Anand 2020

Reinforcement Learning with Augmented Data

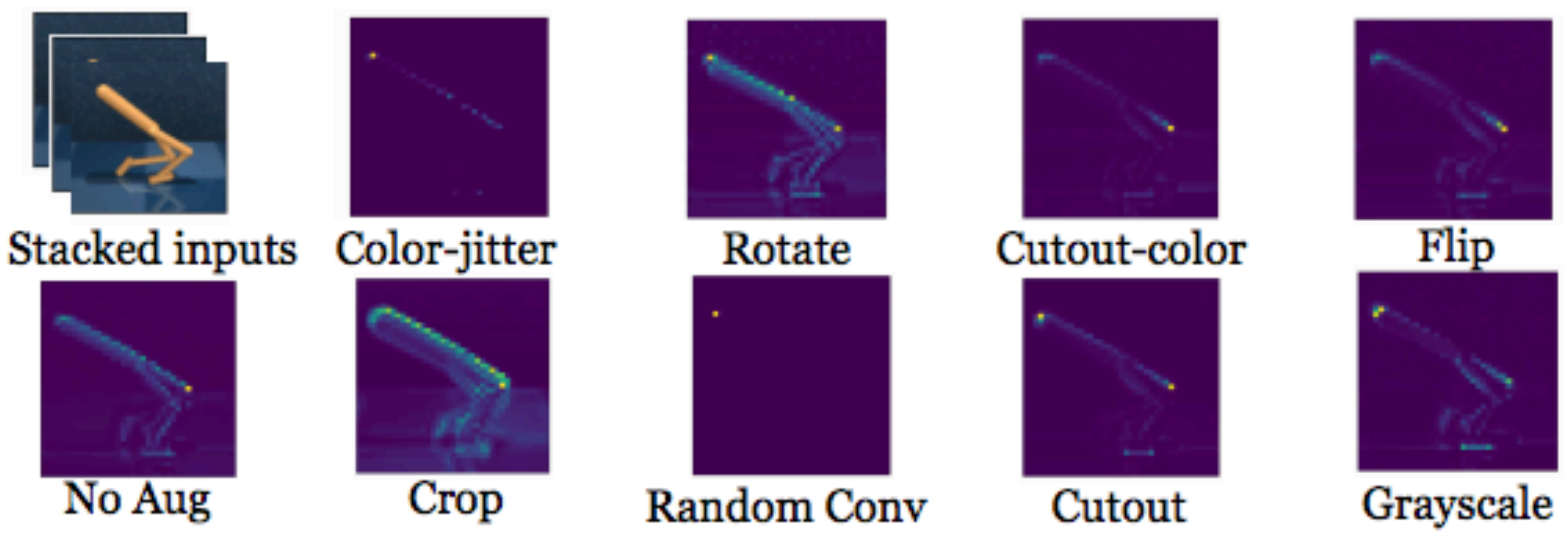


Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, Srinivas, NeurIPS 2020
Image Augmentation is all you need; Kostrikov, Yarats, Fergus; ICLR 2021

Reinforcement Learning with Augmented Data

	Crop	Gray scale	Rotate	Cutout	Color-jitter	Flip
Crop	920	849	635	855	797	650
Grayscale	856	175	349	187	214	231
Rotate	604	403	268	293	392	394
Cutout	722	206	376	215	31	284
Color-jitter	831	227	420	407	194	265
Flip	828	210	391	244	264	223

(a) Scores on DMControl500k for Walker, walk.

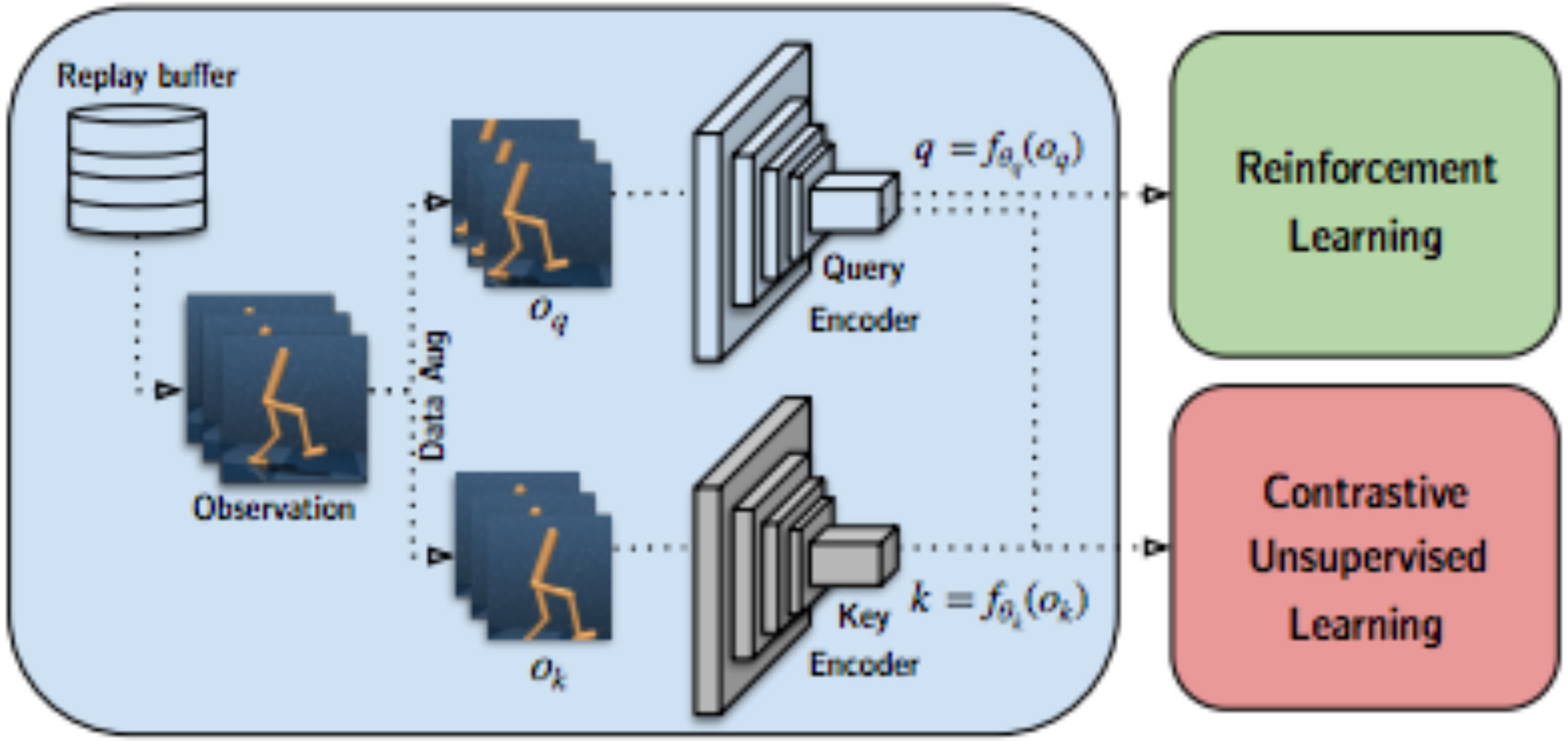


(b) Spatial attention map for Walker, walk.

No surprises, random crops helps. And the network focuses on the right saliency regions.

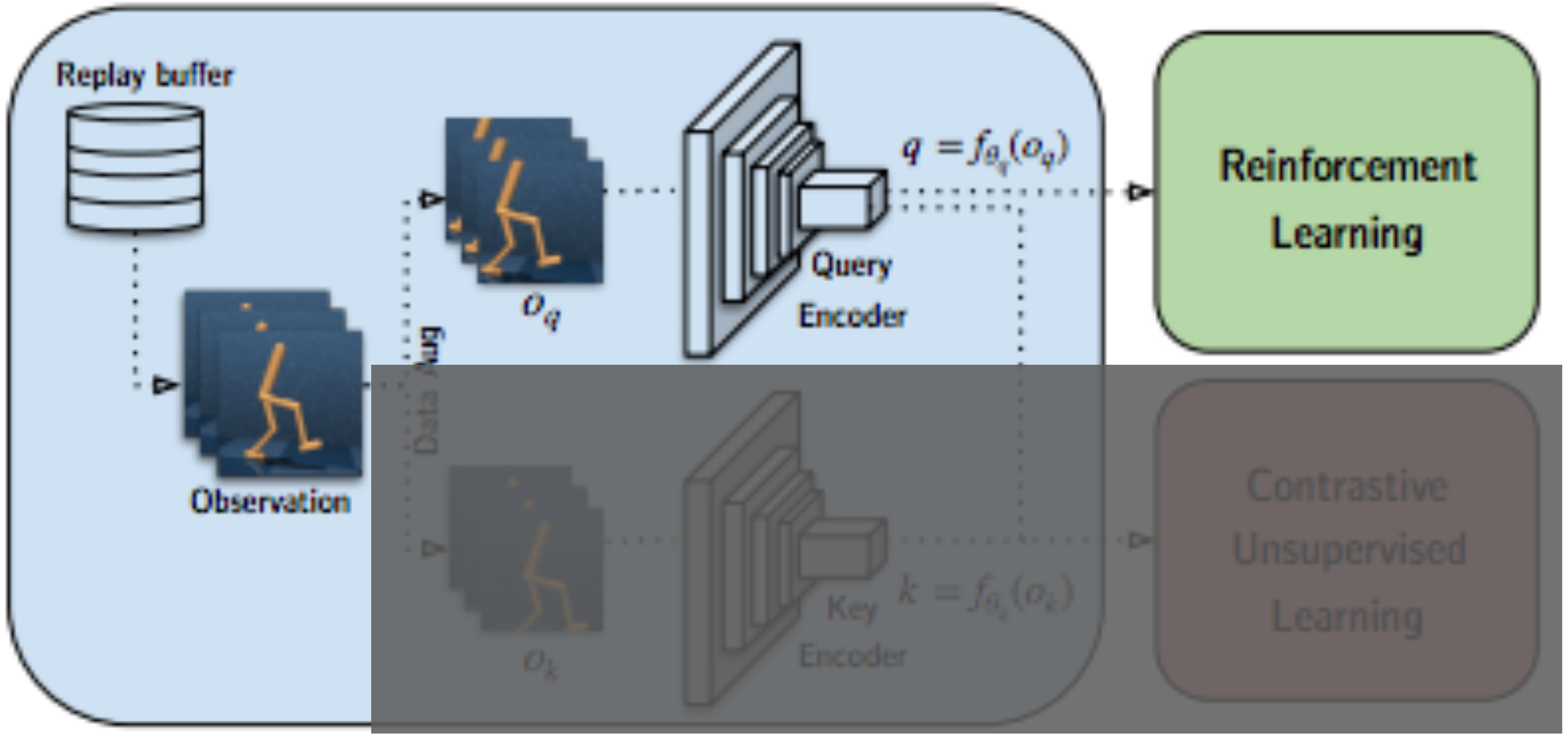
Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, Srinivas, NeurIPS 2020

Reinforcement Learning with Augmented Data



Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, Srinivas, NeurIPS 2020

Reinforcement Learning with Augmented Data



Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, Srinivas, NeurIPS 2020
Image Augmentation is all you need; Kostrikov, Yarats, Fergus; ICLR 2021

Reinforcement Learning with Augmented Data

500K STEP SCORES	RAD	CURL	PLANET	DREAMER	SAC+AE	SLACV1	PIXEL SAC	STATE SAC
FINGER, SPIN	**947** ± 101	926 ± 45	561 ± 284	796 ± 183	884 ± 128	673 ± 92	192 ± 166	923 ± 211
CARTPOLE, SWING	**863** ± 9	845 ± 45	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
REACHER, EASY	**955** ± 71	929 ± 44	210 ± 44	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
CHEETAH, RUN	**728** ± 71	518 ± 28	305 ± 131	570 ± 253	550 ± 34	640 ± 19	197 ± 15	795 ± 30
WALKER, WALK	**918** ± 16	902 ± 43	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
CUP, CATCH	**974** ± 12	959 ± 27	460 ± 380	879 ± 87	794 ± 58	852 ± 71	312 ± 63	974 ± 33
100K STEP SCORES								
FINGER, SPIN	**856** ± 73	767 ± 56	136 ± 216	341 ± 70	740 ± 64	693 ± 141	224 ± 101	811 ± 46
CARTPOLE, SWING	**828** ± 27	582 ± 146	297 ± 39	326 ± 27	311 ± 11	-	200 ± 72	835 ± 22
REACHER, EASY	**826** ± 219	538 ± 233	20 ± 50	314 ± 155	274 ± 14	-	136 ± 15	746 ± 25
CHEETAH, RUN	**447** ± 88	299 ± 48	138 ± 88	235 ± 137	267 ± 24	319 ± 56	130 ± 12	616 ± 18
WALKER, WALK	**504** ± 191	403 ± 24	224 ± 48	277 ± 12	394 ± 22	361 ± 73	127 ± 24	891 ± 82
CUP, CATCH	**840** ± 179	769 ± 43	0 ± 0	246 ± 174	391 ± 82	512 ± 110	97 ± 27	746 ± 91

Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, Srinivas**,** NeurIPS 2020
Image Augmentation is all you need; Kostrikov, Yarats, Fergus; ICLR 2021

Reinforcement Learning with Augmented Data

500K step scores	RAD	CURL	PlaNet	Dreamer	SAC+AE	SLACv1	Pixel SAC	State SAC
Finger, spin	**947** ± 101	926 ± 45	561 ± 284	796 ± 183	884 ± 128	673 ± 92	192 ± 166	923 ± 211
Cartpole, swing	**863** ± 9	845 ± 45	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
Reacher, easy	**955** ± 71	929 ± 44	210 ± 44	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
Cheetah, run	**728** ± 71	518 ± 28	305 ± 131	570 ± 253	550 ± 34	640 ± 19	197 ± 15	795 ± 30
Walker, walk	**918** ± 16	902 ± 43	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
Cup, catch	**974** ± 12	959 ± 27	460 ± 380	879 ± 87	794 ± 58	852 ± 71	312 ± 63	974 ± 33
100K step scores								
Finger, spin	**856** ± 73	767 ± 56	136 ± 216	341 ± 70	740 ± 64	693 ± 141	224 ± 101	811 ± 46
Cartpole, swing	**828** ± 27	582 ± 146	297 ± 39	326 ± 27	311 ± 11	-	200 ± 72	835 ± 22
Reacher, easy	**826** ± 219	538 ± 233	20 ± 50	314 ± 155	274 ± 14	-	136 ± 15	746 ± 25
Cheetah, run	**447** ± 88	299 ± 48	138 ± 88	235 ± 137	267 ± 24	319 ± 56	130 ± 12	616 ± 18
Walker, walk	**504** ± 191	403 ± 24	224 ± 48	277 ± 12	394 ± 22	361 ± 73	127 ± 24	891 ± 82
Cup, catch	**840** ± 179	769 ± 43	0 ± 0	246 ± 174	391 ± 82	512 ± 110	97 ± 27	746 ± 91

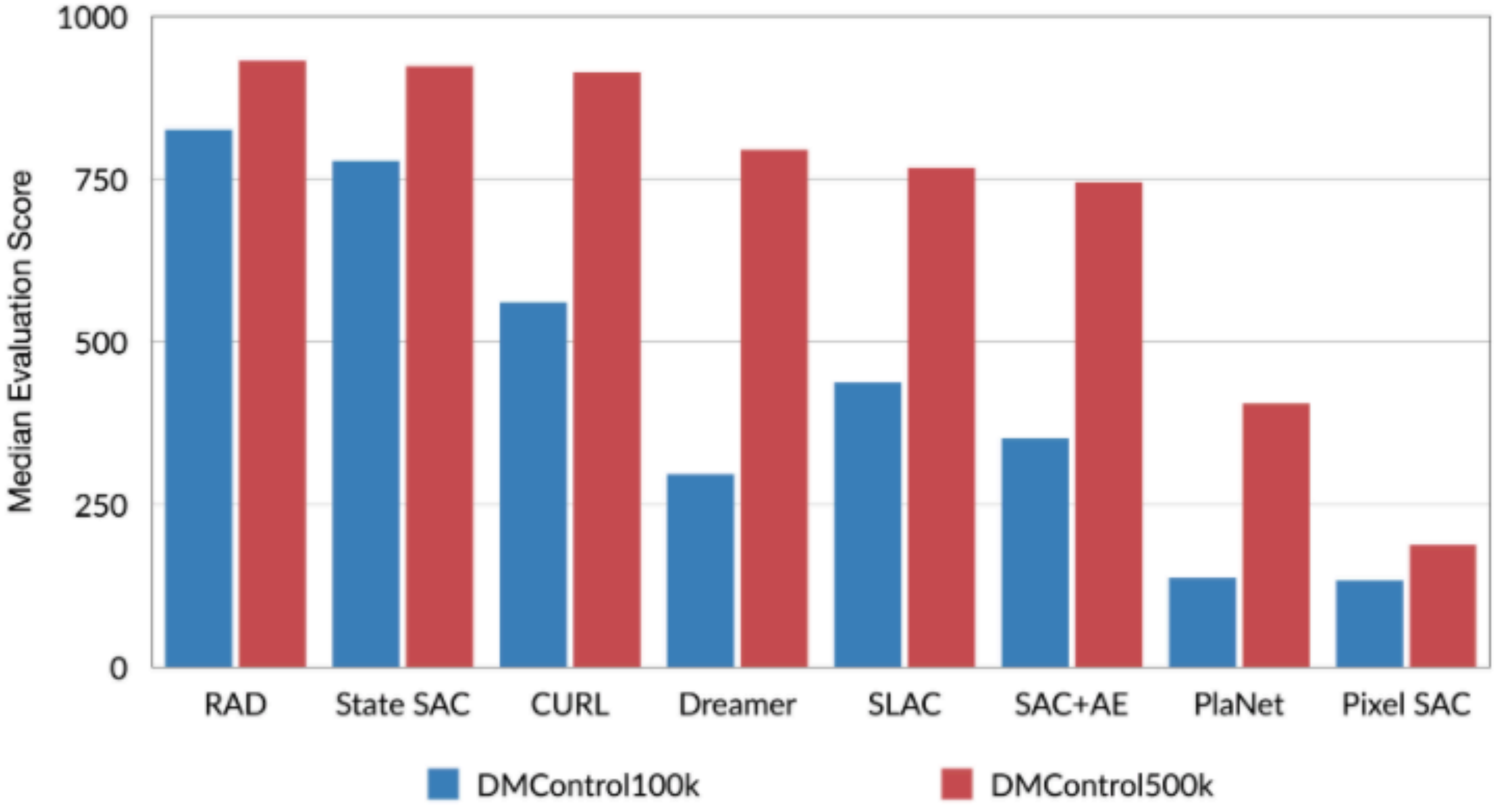
Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, **Srinivas,** NeurIPS 2020
Image Augmentation is all you need; Kostrikov, Yarats, Fergus; ICLR 2021

Reinforcement Learning with Augmented Data

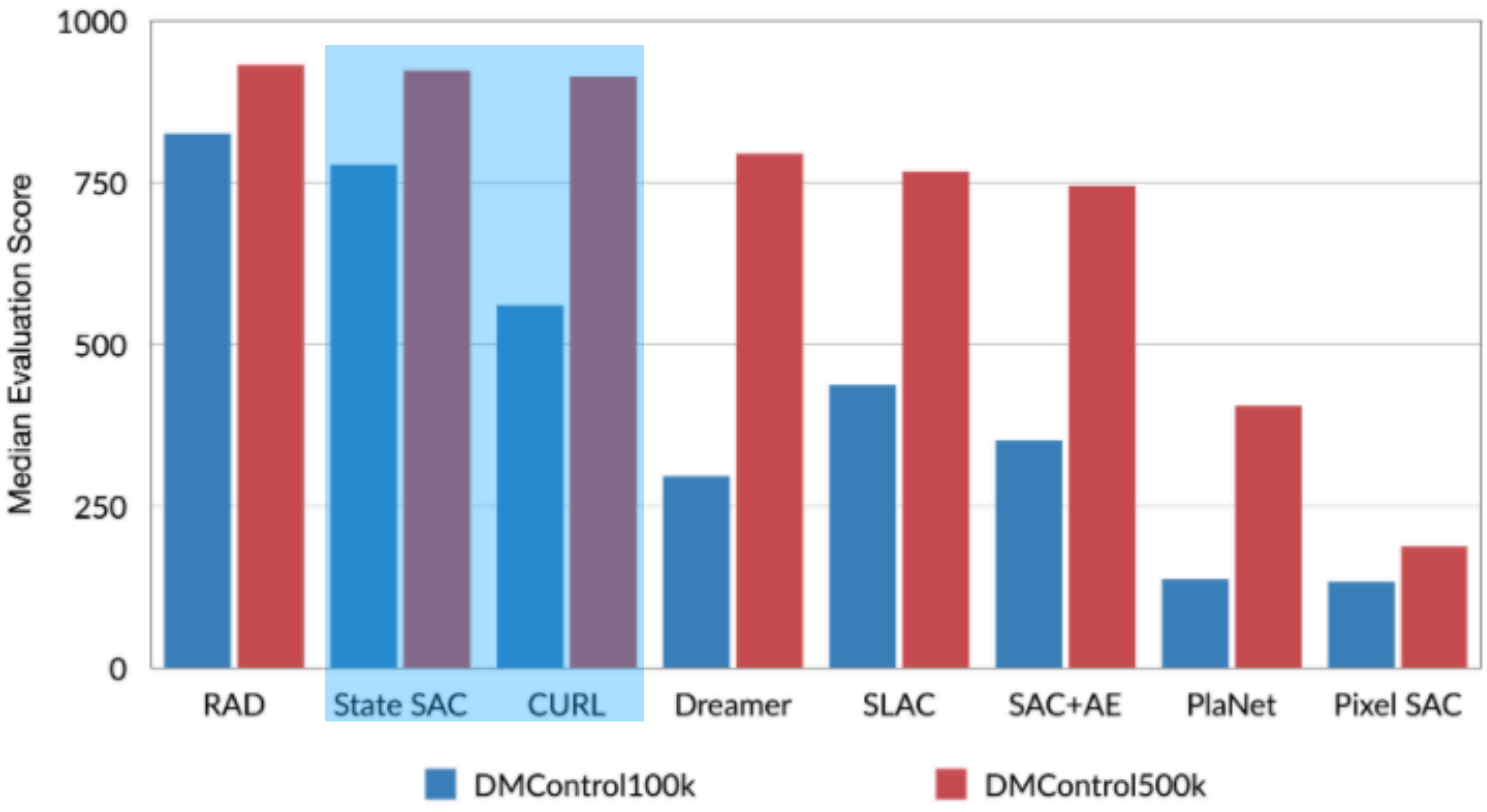
500K step scores	RAD	CURL	PlaNet	Dreamer	SAC+AE	SLACv1	Pixel SAC	State SAC
Finger, spin	**947** ± 101	926 ± 45	561 ± 284	796 ± 183	884 ± 128	673 ± 92	192 ± 166	923 ± 211
Cartpole, swing	**863** ± 9	845 ± 45	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
Reacher, easy	**955** ± 71	929 ± 44	210 ± 44	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
Cheetah, run	**728** ± 71	518 ± 28	305 ± 131	570 ± 253	550 ± 34	640 ± 19	197 ± 15	795 ± 30
Walker, walk	**918** ± 16	902 ± 43	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
Cup, catch	**974** ± 12	959 ± 27	460 ± 380	879 ± 87	794 ± 58	852 ± 71	312 ± 63	974 ± 33
100K step scores								
Finger, spin	**856** ± 73	767 ± 56	136 ± 216	341 ± 70	740 ± 64	693 ± 141	224 ± 101	811 ± 46
Cartpole, swing	**828** ± 27	582 ± 146	297 ± 39	326 ± 27	311 ± 11	-	200 ± 72	835 ± 22
Reacher, easy	**826** ± 219	538 ± 233	20 ± 50	314 ± 155	274 ± 14	-	136 ± 15	746 ± 25
Cheetah, run	**447** ± 88	299 ± 48	138 ± 88	235 ± 137	267 ± 24	319 ± 56	130 ± 12	616 ± 18
Walker, walk	**504** ± 191	403 ± 24	224 ± 48	277 ± 12	394 ± 22	361 ± 73	127 ± 24	891 ± 82
Cup, catch	**840** ± 179	769 ± 43	0 ± 0	246 ± 174	391 ± 82	512 ± 110	97 ± 27	746 ± 91

Reinforcement Learning with Augmented Data, Laskin, Lee, Stooke, Pinto, Abbeel, **Srinivas,** NeurIPS 2020
Image Augmentation is all you need; Kostrikov, Yarats, Fergus; ICLR 2021

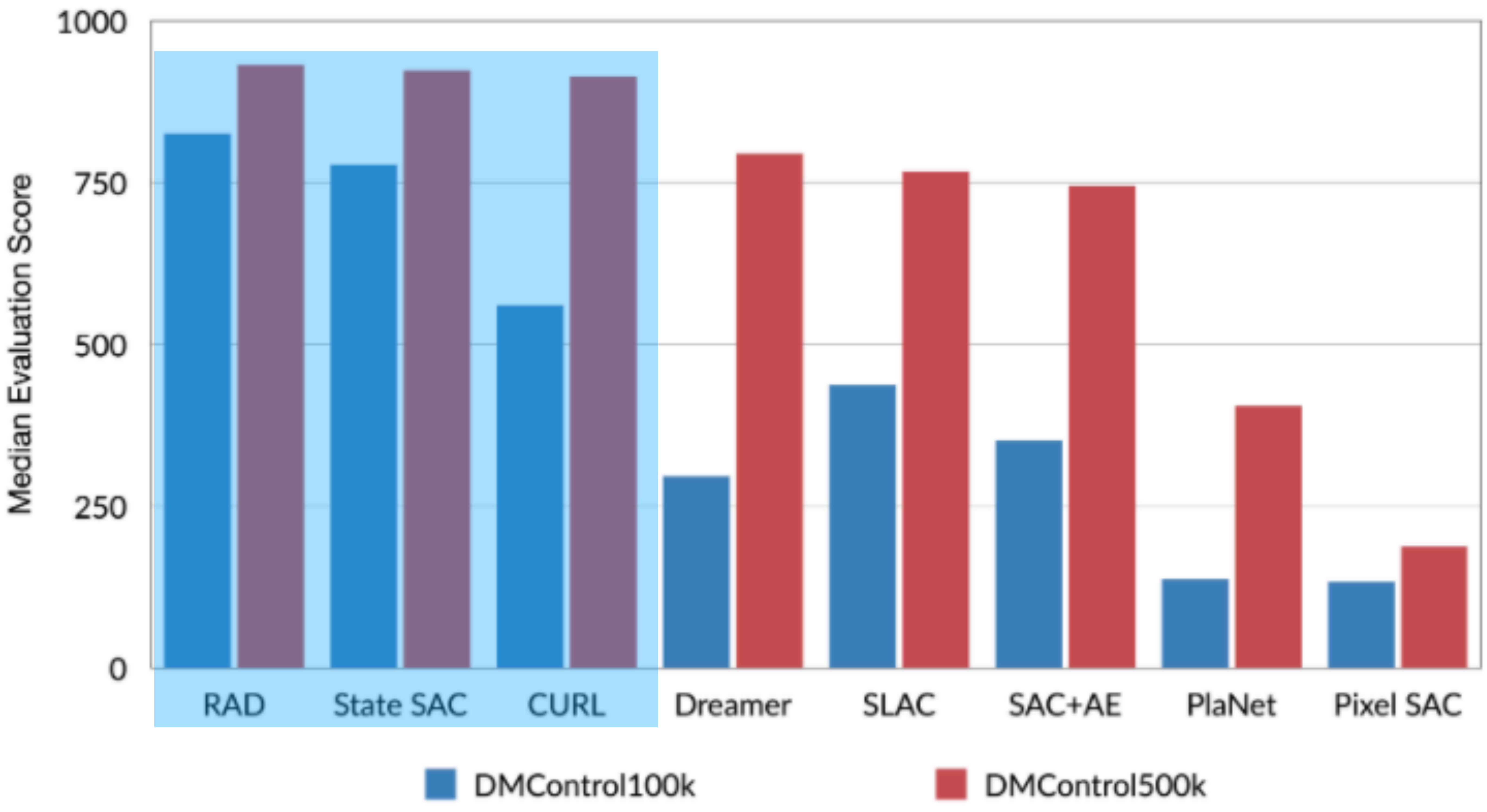
Reinforcement Learning with Augmented Data



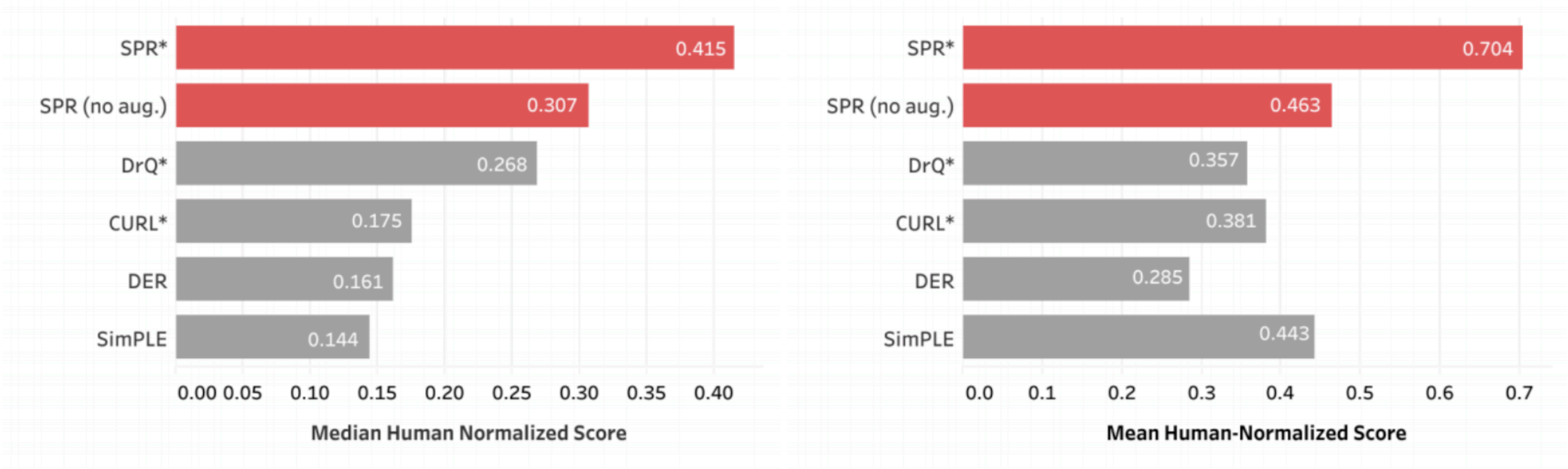
Reinforcement Learning with Augmented Data



Reinforcement Learning with Augmented Data



Data Augmentation vs Auxiliary Losses

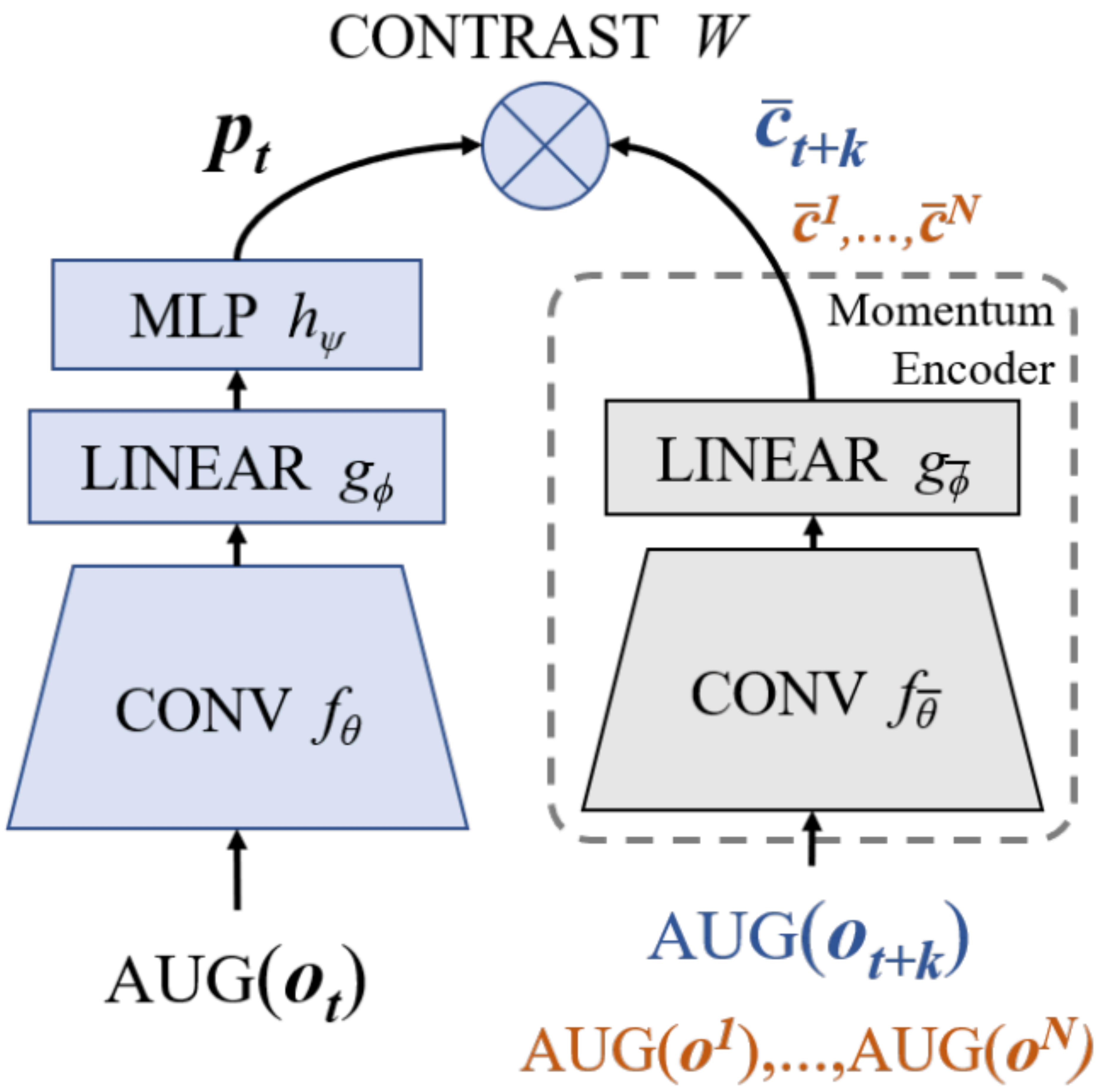


Data-Efficient Reinforcement Learning with Self-Predictive Representations, Schwarzer & Anand 2020

Interpretation of augmentations in auxiliary and direct form

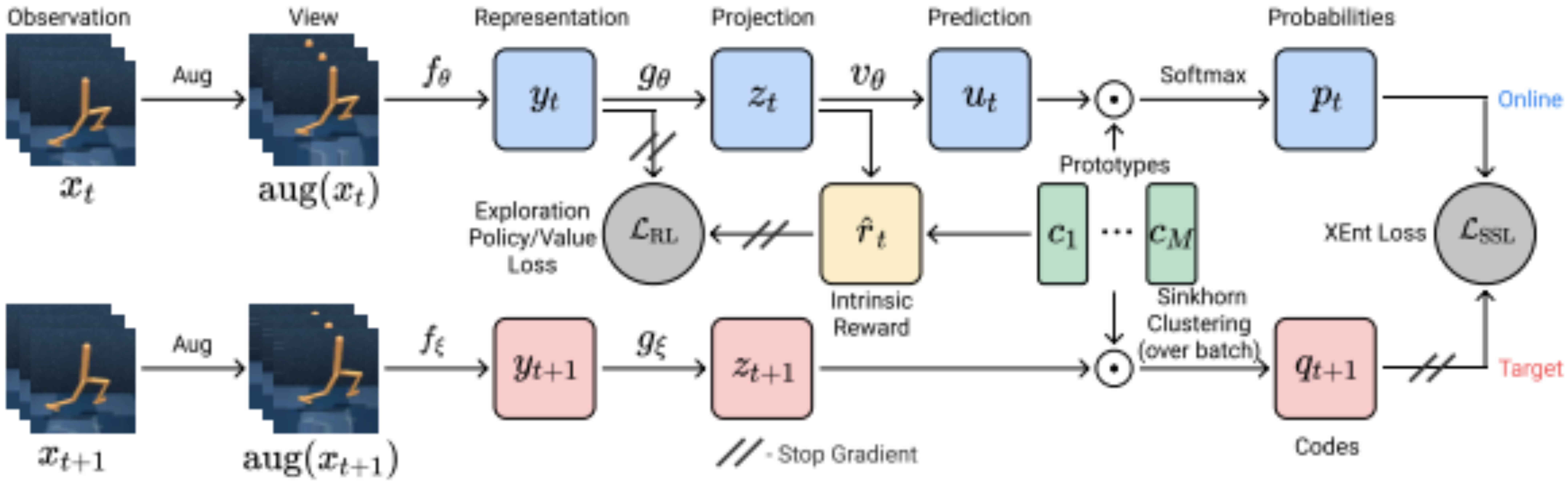
- CURL = Un-annotated (can be done w/o rewards or tasks; flexible with respect to positive & anchors)
- RAD = Annotated (only for a specific reward or task)
- CURL can help you learn one general purpose encoder for many tasks in the decoupled setup.
- Lots of followup work combining both the ideas now.

Decoupling Representation and Reinforcement Learning



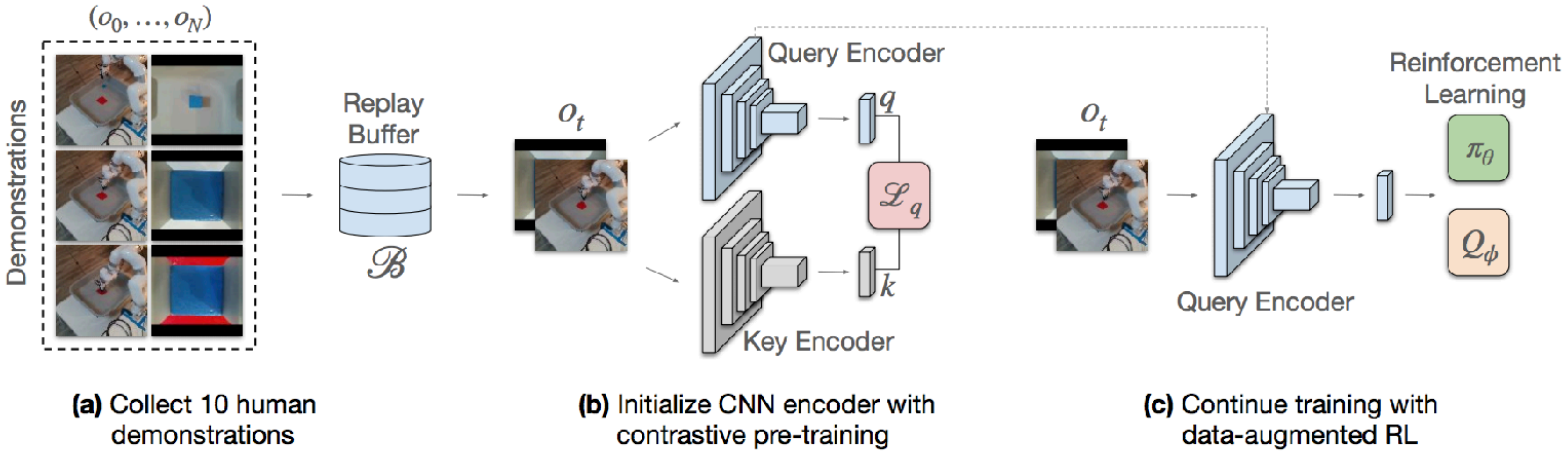
Stooke et al, ICML 2021

SwaV + DrQ (Proto-RL) - Yarats



Proto-RL, Yarats et al 2021

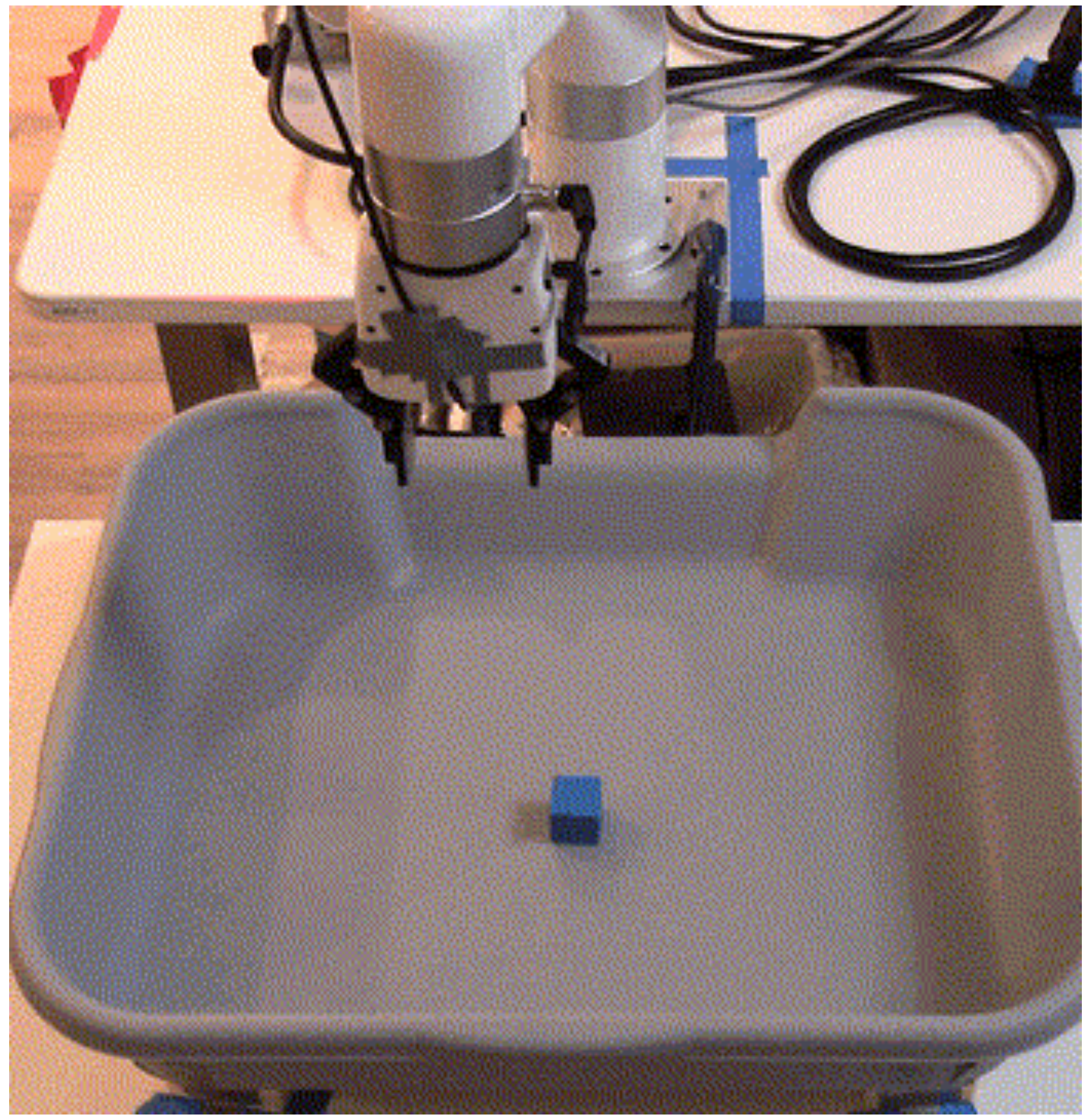
FERM: CURL + RAD for Robotics

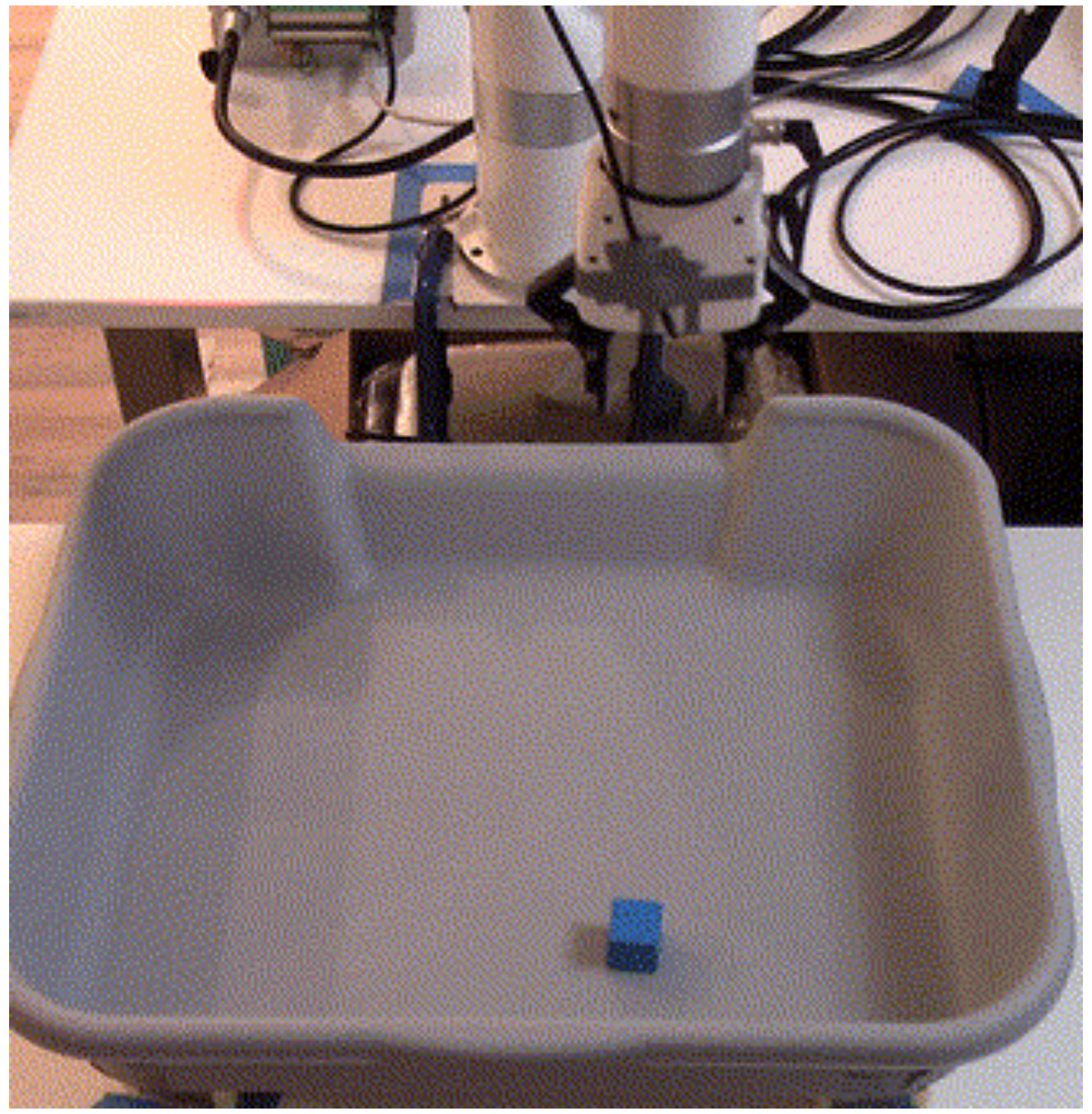


Robot Learning from Pixels for Sparse Reward Tasks, within 30 minutes of real time training

Framework for Efficient Robot Manipulation - Zhan et al, 2020

FERM: CURL + RAD for Robotics

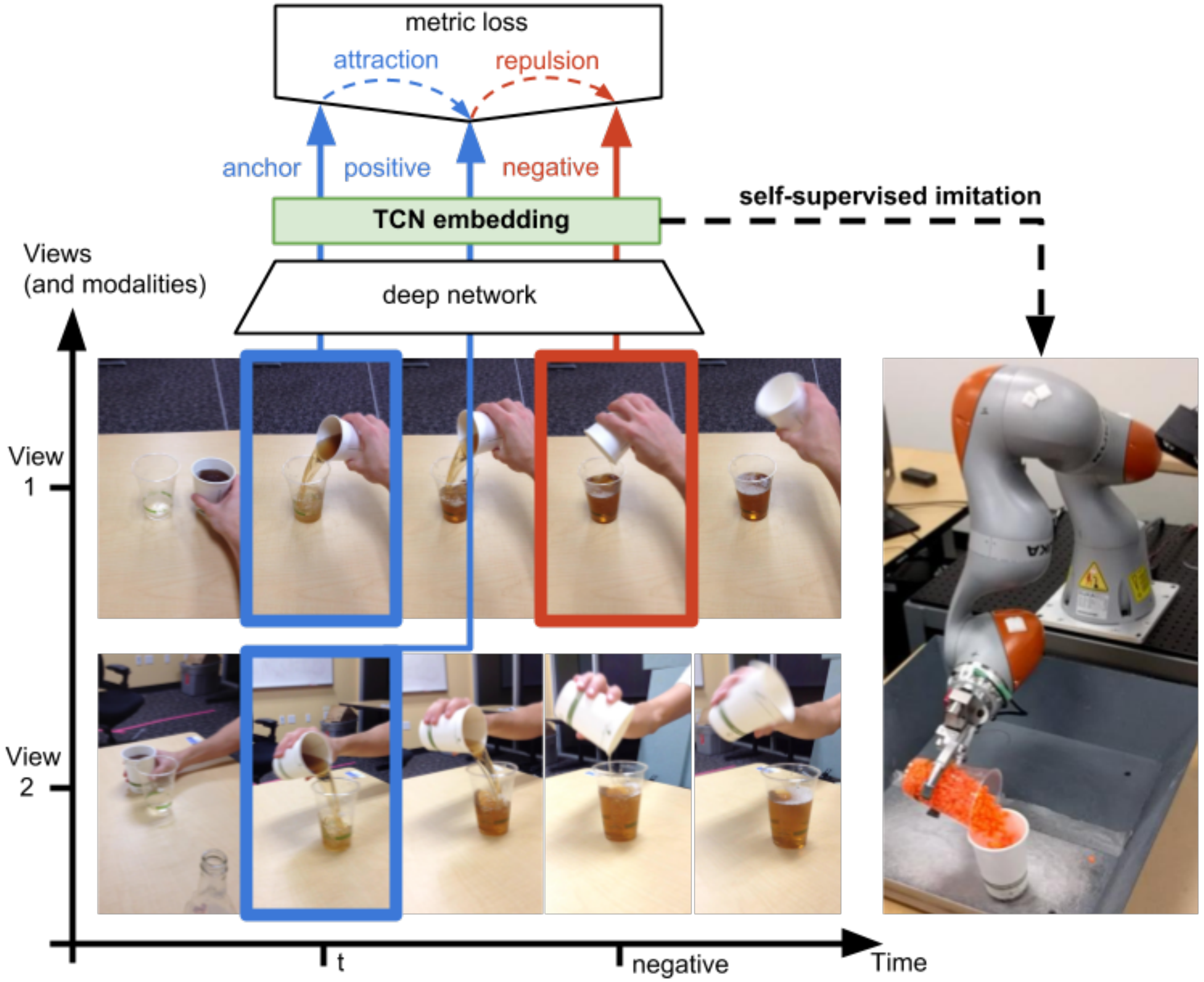




Robot Learning from Pixels for Sparse Reward Tasks, within 30 minutes of real time training

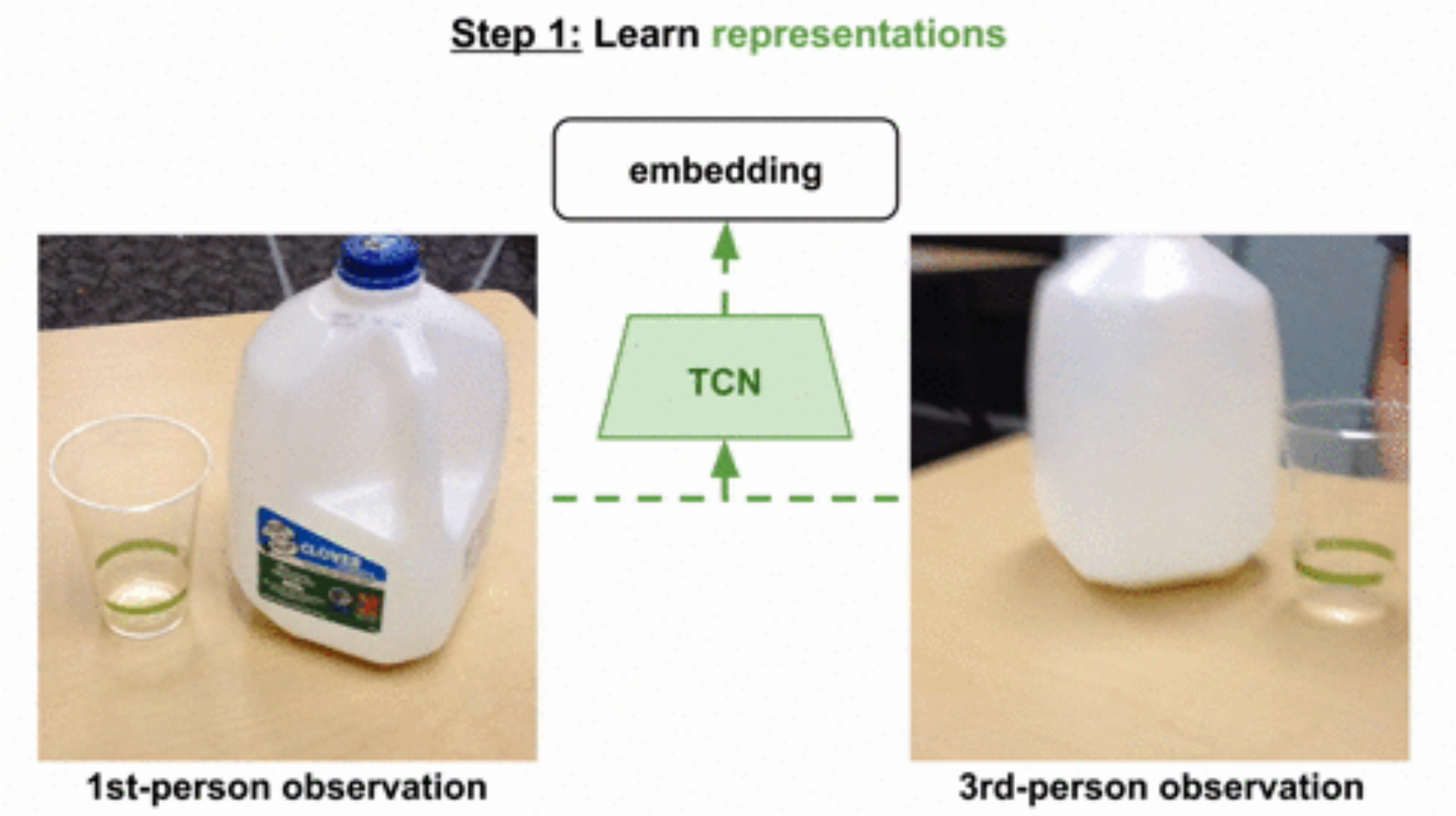
Framework for Efficient Robot Manipulation - Zhan et al, 2020

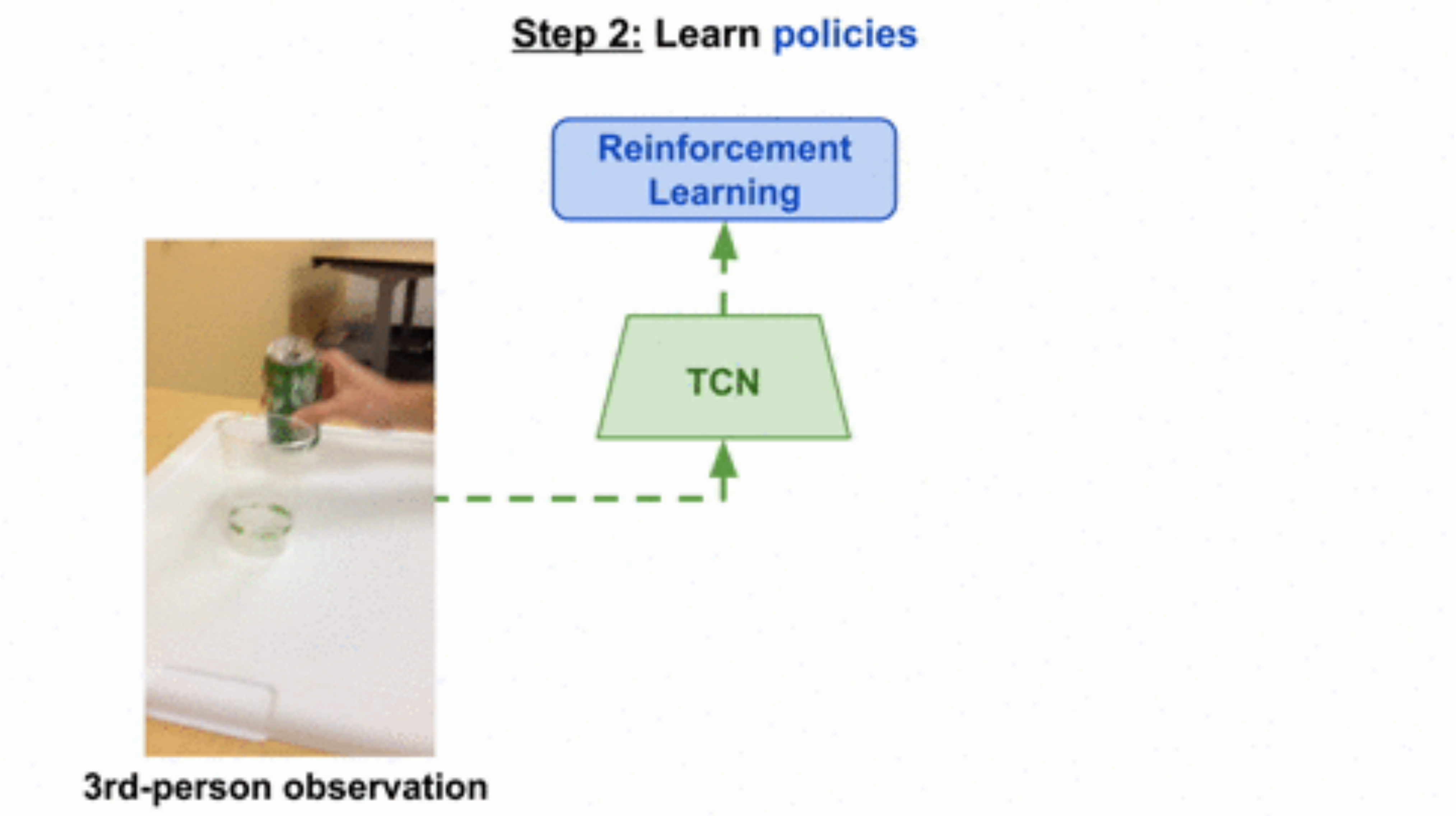
Time Contrastive Networks (TCN)



Time Contrastive Networks: Self-Supervised Learning from Video, Sermanet et al 2018

Time Contrastive Networks (TCN)





Time Contrastive Networks: Self-Supervised Learning from Video, Sermanet et al 2018

Representative Work (not exhaustive)

Generative UL

1. UNREAL: Reinforcement Learning with Unsupervised Auxiliary Tasks (Jaderberg et al 2016)
2. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning (Higgins et al 2017)
3. World Models (Ha & Schmidhuber, 2018)
4. Learning to summarize from human feedback (Stiennon et al 2020)

Contrastive-like UL

5. CPC: Contrastive Predictive Coding (Van den Oord et al 2018)
6. CURL: Contrastive Unsupervised Representations for RL (Srinivas & Laskin et al 2020)
7. DrQ: Image Augmentation is all you need - (Kostrikov & Yarats et al 2020),
   RAD: Reinforcement Learning with Augmented Data (Laskin et al 2020)
8. SPR: Self-Predictive Representations (Schwarzer & Anand et al 2020)
9. Decoupling Representation Learning from Reinforcement Learning (Stooke et al 2021)
10. Reinforcement Learning with Prototypical Representations (Yarats et al 2021)
11. Pre-training representations for data-efficient RL (Schwarzer & Rajkumar et al 2021)

Robotics Applications

12. Deep Spatial Autoencoders for Visuomotor Learning (Finn et al 2015)
13. TCN: Time Contrastive Networks (Sermanet et al 2018)
14. FERM: A Framework for Efficient Robotic Manipulation (Zhan et al 2020)

Representative Work (not exhaustive)

1. UNREAL: Reinforcement Learning with Unsupervised Auxiliary Tasks (Jaderberg et al 2016)
2. DARLA: Improving Zero-Shot Transfer in Reinforcement Learning (Higgins et al 2017)
3. World Models (Ha & Schmidhuber, 2018)
4. Learning to summarize from human feedback (Stiennon et al 2020)

5. CPC: Contrastive Predictive Coding (Van den Oord et al 2018)
6. CURL: Contrastive Unsupervised Representations for RL (Srinivas & Laskin et al 2020)
7. DrQ: Image Augmentation is all you need - (Kostrikov & Yarats et al 2020),
   RAD: Reinforcement Learning with Augmented Data (Laskin et al 2020)
8. **SPR: Self-Predictive Representations (Schwarzer & Anand et al 2020)**
9. Decoupling Representation Learning from Reinforcement Learning (Stooke et al 2021)
10. Reinforcement Learning with Prototypical Representations (Yarats et al 2021)
11. **Pre-training representations for data-efficient RL (Schwarzer & Rajkumar et al 2021)**

Significant improvements

12. Deep Spatial Autoencoders for Visuomotor Learning (Finn et al 2015)
13. TCN: Time Contrastive Networks (Sermanet et al 2018)
14. FERM: A Framework for Efficient Robotic Manipulation (Zhan et al 2020)

Research Ideas

1. So many opportunities for future research in this space
2. Exploring objectives like DINO, Barlow Twins for RL - connections to Policy Distillation
3. Applying these ideas to indoor navigation in Habitat
4. Real-world results in robot manipulation
5. Observational Imitation (Revisiting TCN with the modern tools)
6. Learning world models in latent spaces discovered by Siamese Nets
7. Larger networks pre-training
8. Multi-task learning with shared unsupervised pre-trained backbones
9. ……..
10. ……..

Thanks!

aravindsrinivas@gmail.com

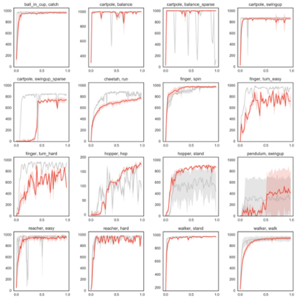
ICML 2021 Tutorial on Unsupervised Learning for RL: Part II: Reward-Free RL

Pieter Abbeel & Aravind Srinivas
UC Berkeley

Thanks to Marc Bellemare, Yang Gao, Misha Laskin, Sergey Levine, Hao Liu, Vlad Mnih, Pierre-Yves Oudeyer, Deepak Pathak, Lerrel Pinto, Aravind Rajeswaran for valuable feedback and suggestions.

Tutorial Overview

- Part I: Representation Learning in RL

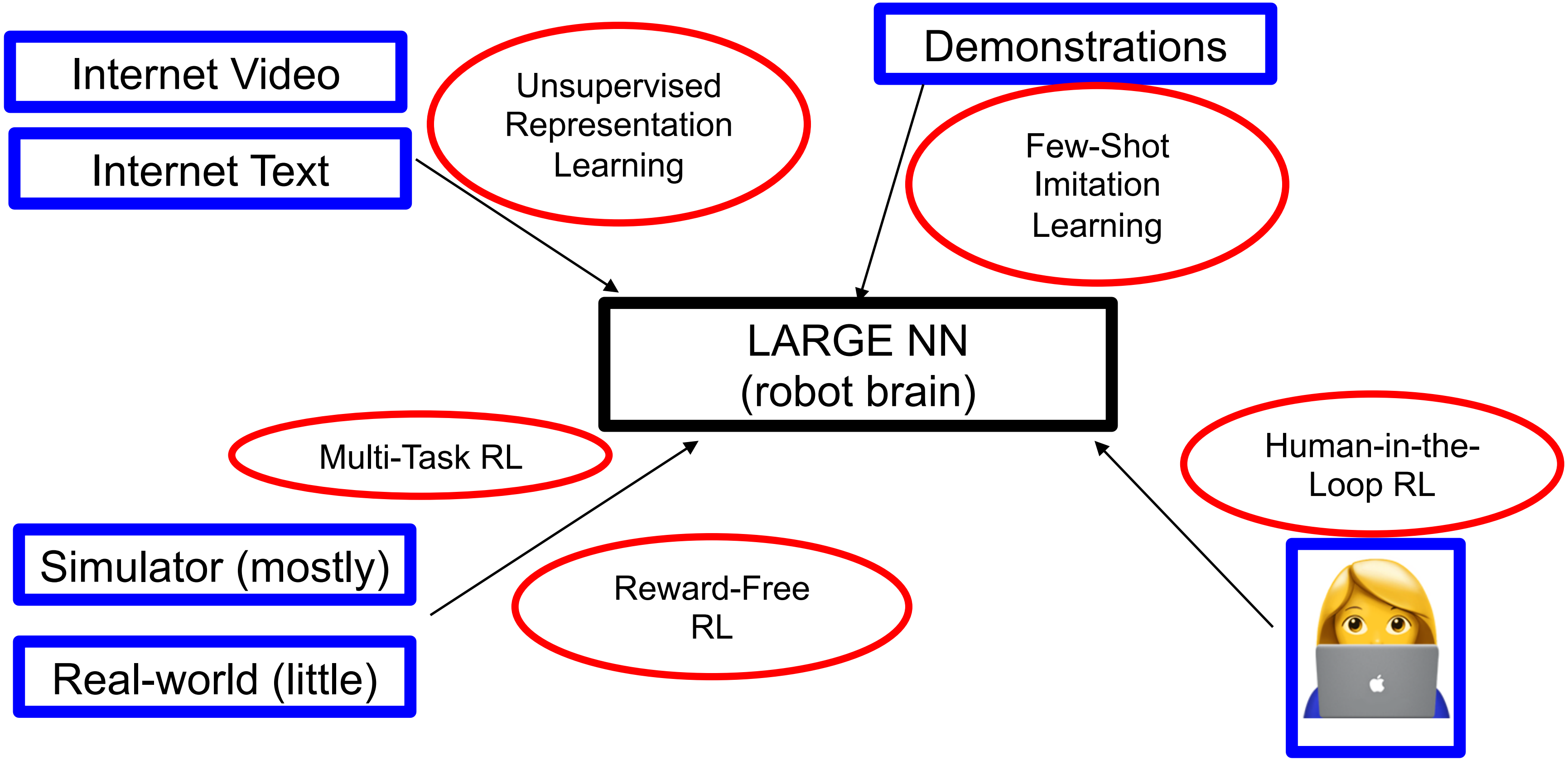


- Part II: Reward-Free RL

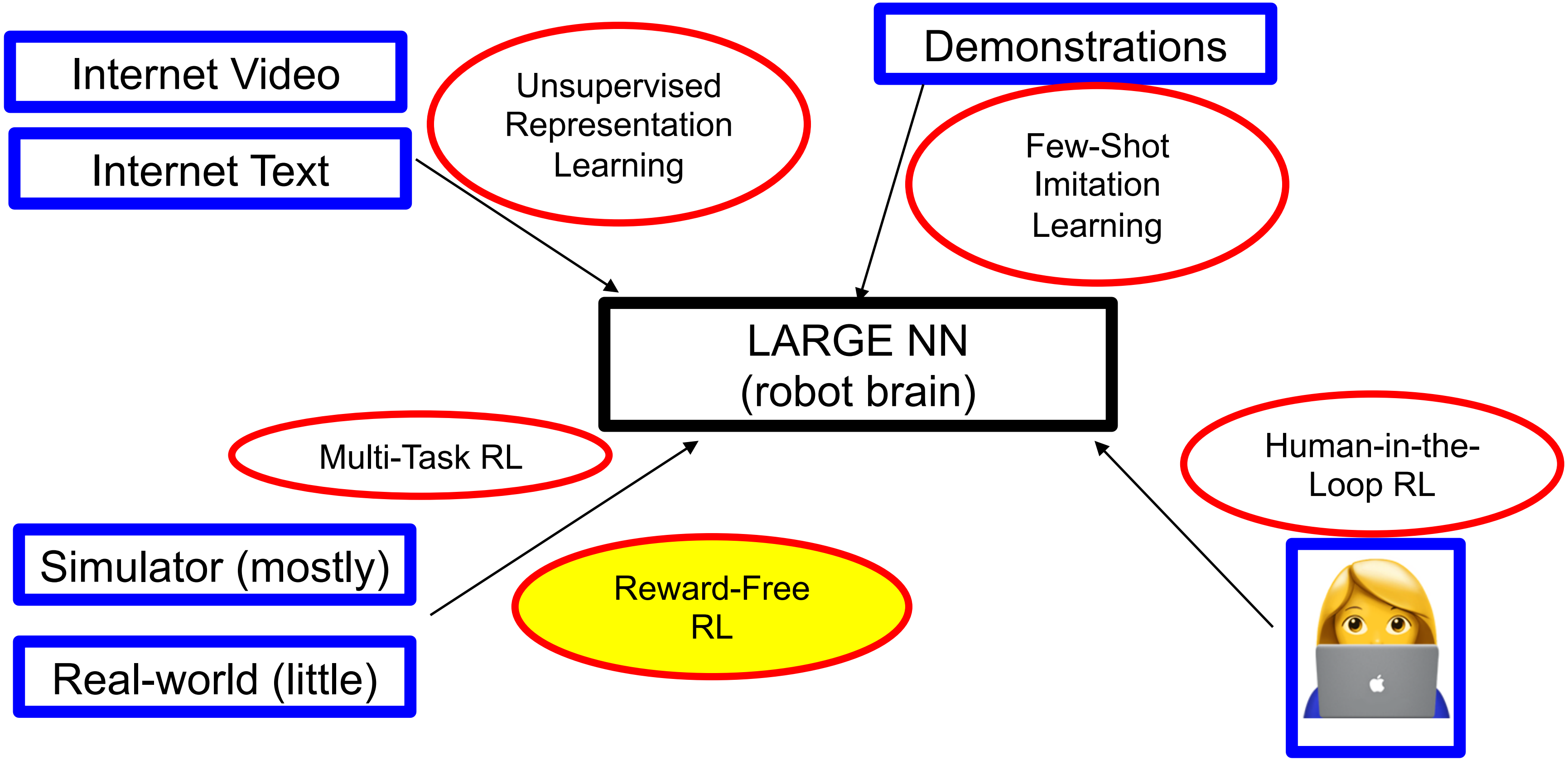
But Wait: Lots of Passive Data Out There…

- So, shouldn’t we leverage that?
  - YES, absolutely
  - But, IN ADDITION, we want our agents to be able to ***intelligently collect their own data***, and that’s what ***this tutorial*** will cover

Interlude: An Attempt at a Complete Picture



Interlude: An Attempt at a Complete Picture

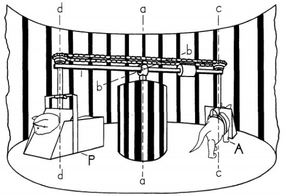


Why Study Agents Collecting Their Own Data (1/2)

- In real world, reward can be tedious to provide
- Alleviate need for bootstrapping from demonstrations
- Superhuman solutions
  - Debug a new game for unexpected issues
  - Circuit design
  - Scientific discovery RL
  - …

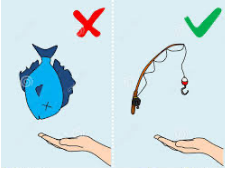
Why Study Agents Collecting Their Own Data (2/2)

- Might lead to more robust learning (or not)



[Held & Hein, 1969]
also: [Walk, Shepherd, Miller, 1988]

- "general brain" for learning decision making



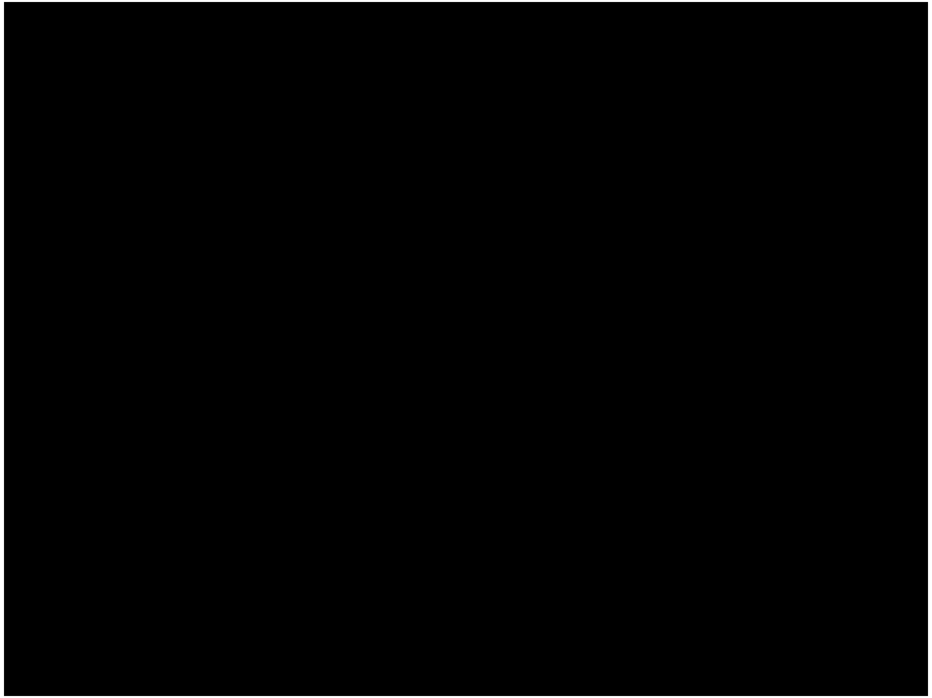
"Give an AI agent a task-reward, they can learn for a day,
give an AI agent intrinsic reward, they can learn for a lifetime."

- AI motivation

[image source: Francis Vachon]

It's how humans/babies learn
lots of play (+ some supervision)

Play



- What's going on here?
  [see, e.g., Gopnik (Berkeley), Schulz (MIT)]
- How to formalize into machine learning / silicon compute?
- What does success even mean?
  Robot rolling on the floor with toys??

Formalizing "Play": "Reward-Free Pre-Training (RFPT)"

Reward-free Pre-Training

Sample complexity typically not too heavily scrutinized here

⇨

Possible measures of success:

- $\pi(a|s)$: 0-shot
- $\pi(a|s)$: fine-tune
- $\pi(a|s,g)$: Goal-conditioned policy
- $\pi(a|s,z)$: Latent-conditioned policy
- $\pi(a|s,z)$: Diverse skill set
- $\{(s,a,s')\}$: Data set
- $f_\theta(s'|s,a)$: Dynamics model

Further challenge: some works are in image space, some use hardcoded image encodings, some in state space, some use hand-selected subsets of state variables, etc.. Same video demo could have very different assumptions…

Is Reward-Free Pre-Training Realistic?

- We don’t simply leave a baby/child on its own for 18 years, then expect them to quickly fine-tune into adult life

- Similarly, it’s likely our AI agents will benefit from a more interleaved approach between training on their own *and* getting access to side-information, e.g.
    - human providing some extrinsic rewards
    - human providing some rewards for good exploration
    - passive data being leveraged for representation learning
    - passive data being leveraged for guiding what’s ”interesting” to explore
    - etc…

- Such variants are beyond the scope of this tutorial

- Such variants will likely still greatly benefit from progress in “pure” RFPT regime

Reward-Free Pre-Training

- First:
  pre-train without task reward available
- Then:
  leverage pre-training to learn faster once task reward is available

Exploration

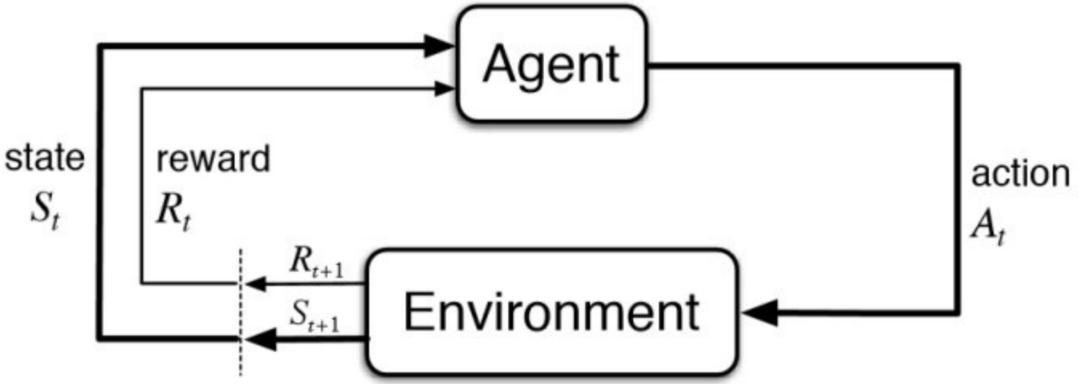
- Task reward is available from the very beginning
- But: often task reward provides very little signal in early learning stages, hence AI Agent needs exploration to find out how to achieve task reward
- Note: as long as no task reward has been experienced, the AI Agent is forced to learn reward-free

- Yes, very related, and so are research ideas/results for each
- Often arbitrary in which of the 2 contexts an idea ends up (and sometimes both)
- Ideas often complementary and by combining can amplify each other

Outline

- Problem Motivation
- ***Baseline RL Algorithms Refresher***
- Intrinsic Rewards for Reward-Free Pre-Training and Exploration
- Algorithmic Approaches to Exploration (can complement intrinsic reward RFPT!)
- Algorithmic Approaches to Reward-Free Pre-Training

Reinforcement Learning (RL)



- Popular RL Algorithms
  - PPO
  - SAC / TD3
  - DDQN
  - Model-Based RL (MBRL)

Popular RL Algorithms are Mostly Optimizers, with Modest Exploration

- PPO, SAC, TD3, DDQN, MBRL in their standard forms
  - use experience so far to optimize expected reward in next roll-out
  - yet also have a *very basic exploration mechanism* built-in:
    - Epsilon-greedy
    - Boltzmann exploration (direct in DDQN, indirect in SAC)
    - Parameter perturbation [Plappert et al, 2017]
    - Noisy-nets [Fortunato et al, 2017]

→ Since so simple, remains very popular to just rely on these, even if clearly not very exploratory

→ But, *hopefully*, this statement about popularity and about superior simplicity can be untrue a few years from now ☺

Outline

- Problem Motivation
- Baseline RL Algorithms Refresher
- ***Reward-Free Pre-Training and Exploration through Baseline RL Algorithms + Intrinsic Rewards***
- Algorithmic Approaches to Exploration (can complement intrinsic reward RFPT!)
- Algorithmic Approaches to Reward-Free Pre-Training

Main Idea behind Intrinsic Rewards

- Intrinsic Reward = generic reward signal that encourages experiencing diversity
- IF we can design such Intrinsic Reward
- THEN we can simply use our existing RL Algorithms as Optimizers to yield exploratory / pre-training behaviors

Reward-Free Pre-Training with Intrinsic Rewards

Run PPO / SAC / TD3 / DDQN / etc.:

$$\max \quad E\,[\,r_{\{INTRINSCIC\}}]$$

Possible measures of success:

- $\pi(a|s)$: 0-shot
- $\pi(a|s)$: fine-tune
- $\pi(a|s,g)$: Goal-conditioned policy
- $\pi(a|s,z)$: Latent-conditioned policy
- $\pi(a|s,z)$: Diverse skill set
- $\{(s,a,s')\}$: Data set
- $f_\theta(s'|s,a)$: Dynamics model

Exploration with Intrinsic Rewards

Run PPO / SAC / TD3 / DDQN / etc.:

$$\max \quad E\,[\;r_{\{TASK\}} + \lambda\;r_{\{INTRINSCIC\}}]$$

Measure of success:

- performance on $r_{\{TASK\}}$

Intrinsic Rewards

CURIOUS MODEL-BUILDING CONTROL SYSTEMS

In Proc. International Joint Conference on Neural Networks, Singapore, volume 2, pages 1458-1463. IEEE, 1991.

Jürgen Schmidhuber*
Department of Computer Science
University of Colorado
Campus Box 430, Boulder, CO 80309, USA

[Schmidhuber, 1991]

What is intrinsic motivation? A typology of computational approaches

Pierre-Yves Oudeyer[1,2,*] and Frederic Kaplan[3]

1. Sony Computer Science Laboratory Paris, Paris, France
2. INRIA Bordeaux-Sud-Ouest, France
3. Ecole Polytechnique Federale de Lausanne, EPFL – CRAFT, Lausanne, Switzerland

[Oudeyer and Kaplan, 2007]

Intrinsic Rewards

					Homeostatic (-) vs Heterostatic (+)	Motivation	Exploration potential	Organization potential	Computational cost	Existing models
Internal	Intrinsic	Adaptive	Knowledge-based	Information theoretic	+	UM	***	*	***	**
					+	IGM	***	***	***	**
					+	DSM	**	***	***	*
					−	DFM	*	***	***	*
				Predictive	+	NM	***	*	*	***
					−	ILNM	**	**	*	**
					+	LPM	***	***	**	**
					+	SM	**	**	**	*
					−	FM	*	***	**	**
			Competence-based		+	IM	***	*	**	*
					+	CPM	***	***	**	*
					−	CM	*	***	**	*
		Fixed	Morphological		−	SyncM	*	***	**	**
					−	StabM	*	***	*	**
					+	VarM	***	*	*	*
	Extrinsic				−	SocM	/	/	*	***
					−	EnerM	/	/	*	***

[Oudeyer and Kaplan, 2007]

Three Main Types of Intrinsic Reward

- Knowledge-based: Surprise / unpredictability / how much learned about world from experience
- Competence-based: Empowerment / Skills
- Data-based: Entropy (i.e. coverage) of data collected

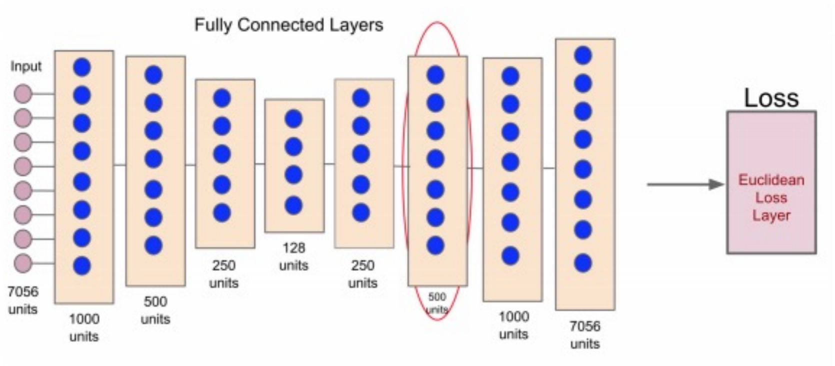
<u>Note 1</u>: Not the only way to categorize, but it's a categorization that can help us understand some of the most prominent work
<u>Note 2</u>: Data can give us Knowledge which can give us Competence, so one could possibly think of Data as the most fundamental, however, it's also least informed during data collection about the value of the data (for knowledge, for building competences)…

Three Main Types of Intrinsic Reward

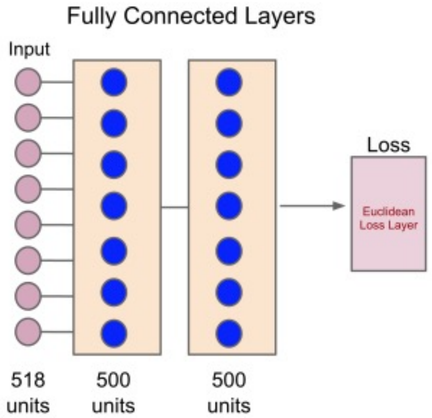
- ***Knowledge-based*: Surprise / unpredictability / how much learned about world from experience**
- Competence-based: Empowerment / Skills
- Data-based: Entropy (i.e. coverage) of data collected

Intrinsic Reward: Error in Learned Dynamics Model

- Key Idea:
  - Train dynamics model on data collected by agent
  - Intrinsic reward = prediction error of the learned dynamics model



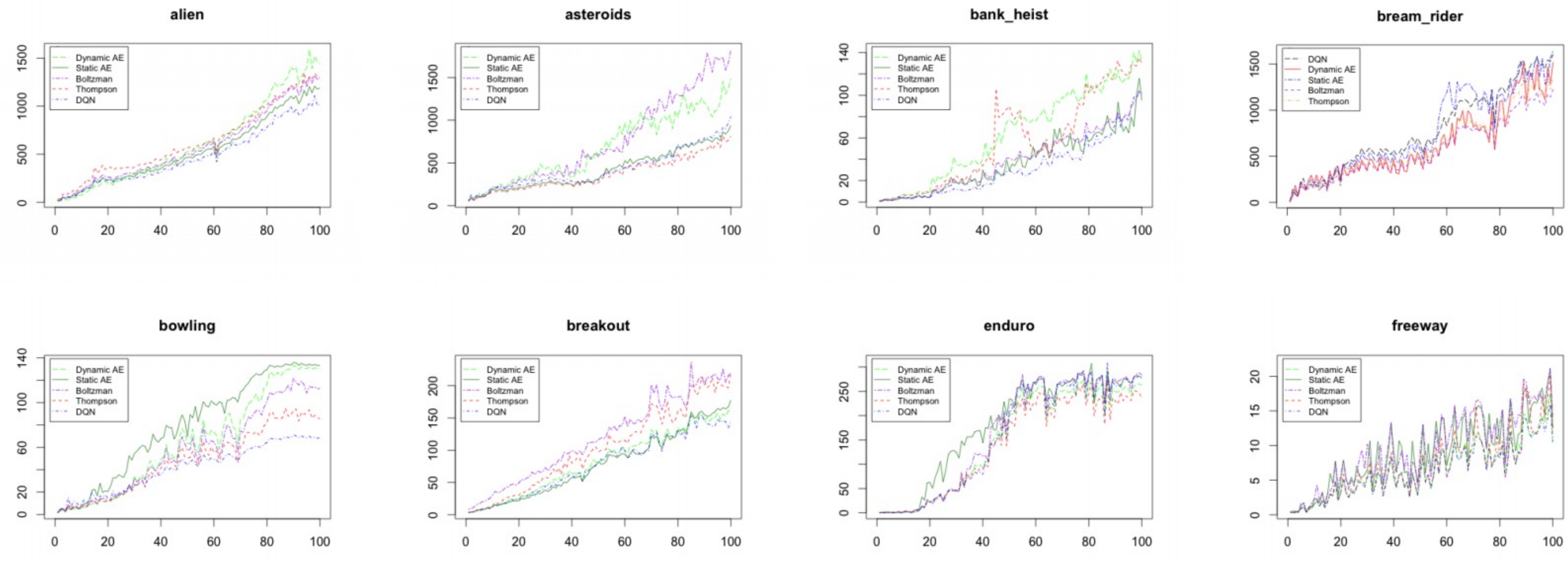
Auto-encoder



Dynamics model in latent space

[Stadie, Levine, Abbeel, 2015; Achiam, Sastry, 2017]

Intrinsic Reward: Error in Learned Dynamics Model



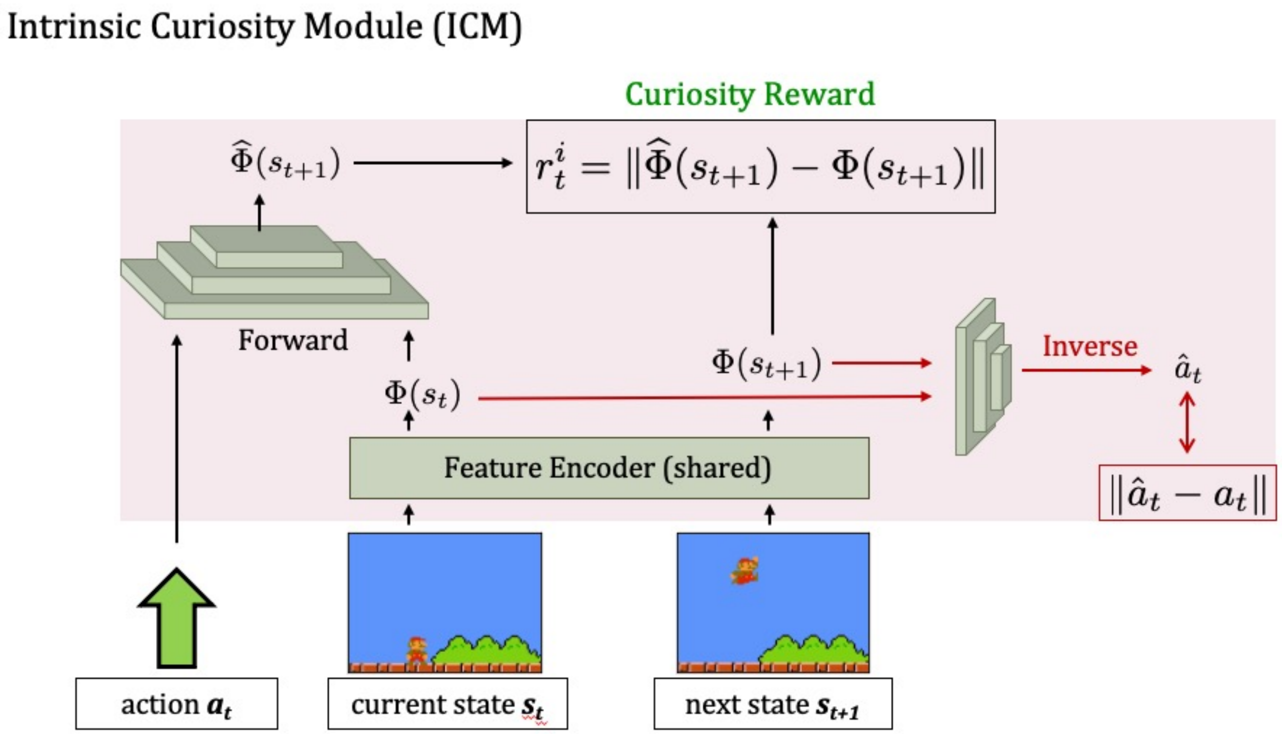
Experimental Findings: tends to often aid exploration / learning speed in Atari

[Stadie, Levine, Abbeel, 2015; Achiam, Sastry, 2017]

Why not doing even better?

- Maybe:
  - VAE spends time modeling things the agent doesn't control
  - Dynamics model makes prediction errors on things the agent doesn't control ("noisy TV")

Inverse Dynamics Model for Encoder



[Pathak, Agrawal, Efros, Darrell, 2017] [See also: Progress Drive by Kaplan and Oudeyer, 2005 for study of noisy TV with remote / solutions]

Inverse Dynamics Model for Encoder

Key findings:

- Inverse dynamics helps focus learning
- 0-shot performance in many games

[Pathak, Agrawal, Efros, Darrell, 2017]

Deeper dive on feature space?

Large-Scale Study of Curiosity-Driven Learning

Yuri Burda* — OpenAI
Harri Edwards* — OpenAI
Deepak Pathak* — UC Berkeley
Amos Storkey — Univ. of Edinburgh
Trevor Darrell — UC Berkeley
Alexei A. Efros — UC Berkeley

- Pixels
- Random features
- VAE
- Inverse Dynamics

Findings:

- Random features do well
- Learned features generalize better

Current Status

- Intrinsic Reward:

$$\boxed{r_t^i = \|\widehat{\Phi}(s_{t+1}) - \Phi(s_{t+1})\|}$$

(with features trained with inverse dynamics)

- How can it still break down?
  - What if agent encounters dice and keeps rolling them?
    - It'll keep getting intrinsic reward, b/c the forward model keeps failing
  - More generally, need to distinguish:
    - Epistemic Uncertainty
    - Aleatoric Uncertainty

Intrinsic Reward for Reduction in Epistemic Uncertainty

- Planning to be Surprised [Sun, Gomez, Schmidhuber, 2011]

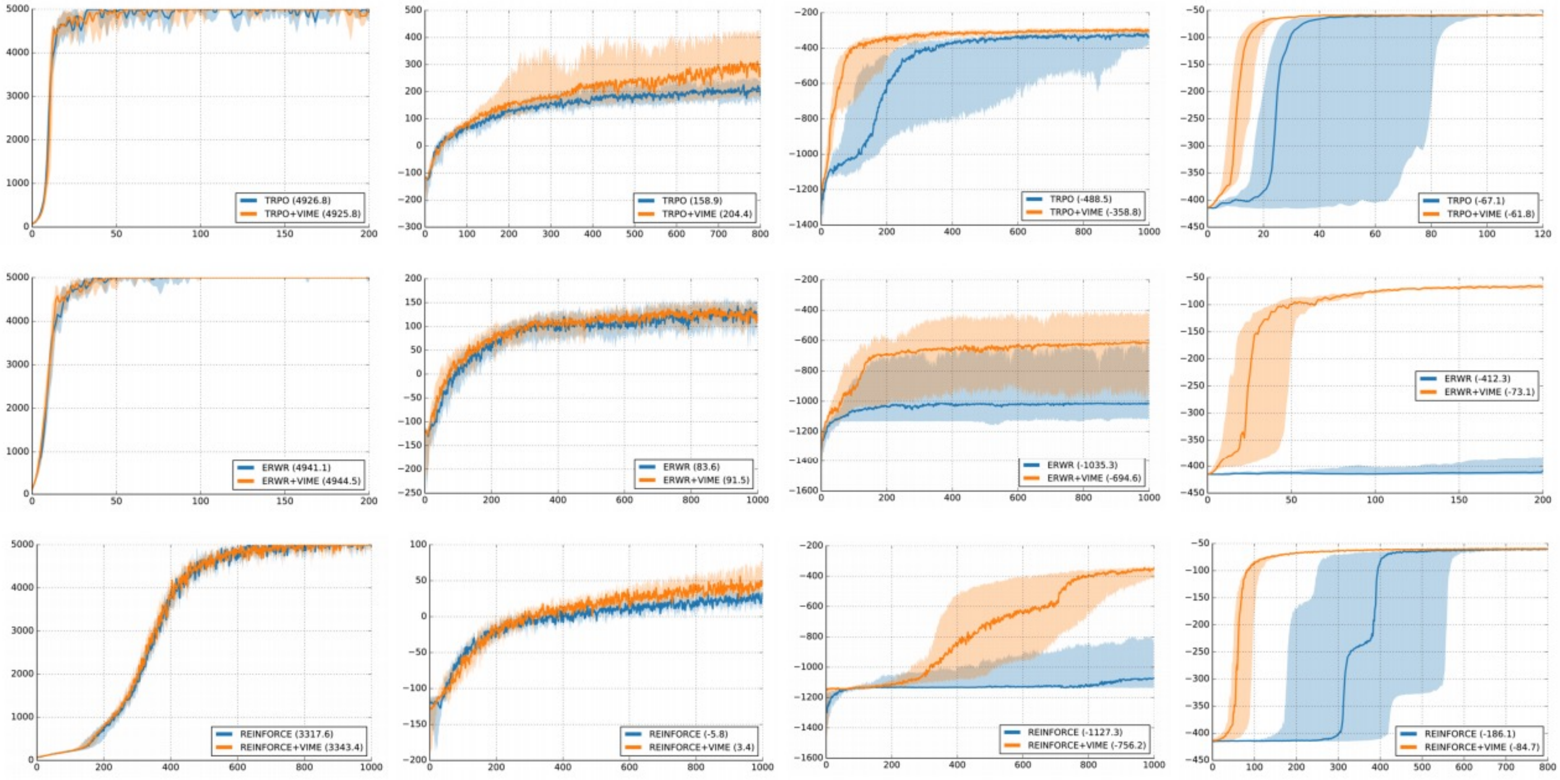
$$\sum_t \left(H(\Theta|\xi_t, a_t) - H(\Theta|S_{t+1}, \xi_t, a_t)\right)$$

Reward for reducing entropy of posterior distribution over possible dynamics models

- VIME: Variational Information Maximization Exploration [Houthooft, Chen, Duan, Schulman, De Turck, Abbeel, 2017]
  - Variational Bayes NN approximation to above (intractable objective)
- Self-Supervised Exploration via Disagreement [Pathak, Gandhi, Gupta, 2019]
  - Learn ensemble of NN dynamics models → disagreement signals epistemic uncertainty

Intrinsic Reward for Reduction in Epistemic Uncertainty

VIME

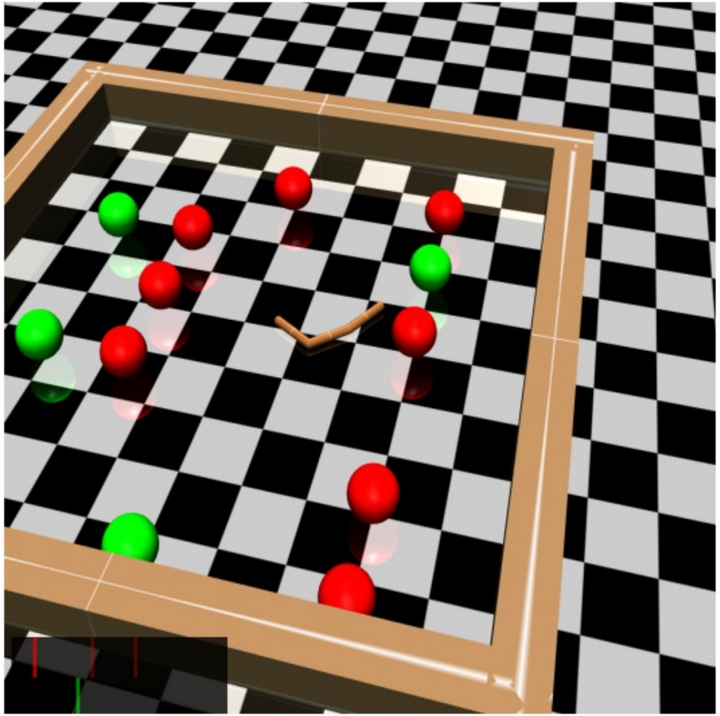


(a) CartPole (b) CartPoleSwingup (c) DoublePendulum (d) MountainCar

[Houthooft, Chen, Duan, Schulman, De Turck, Abbeel, 2017]

Intrinsic Reward for Reduction in Epistemic Uncertainty

VIME



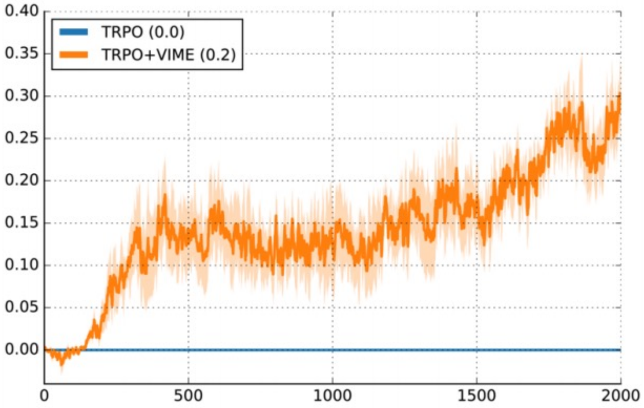


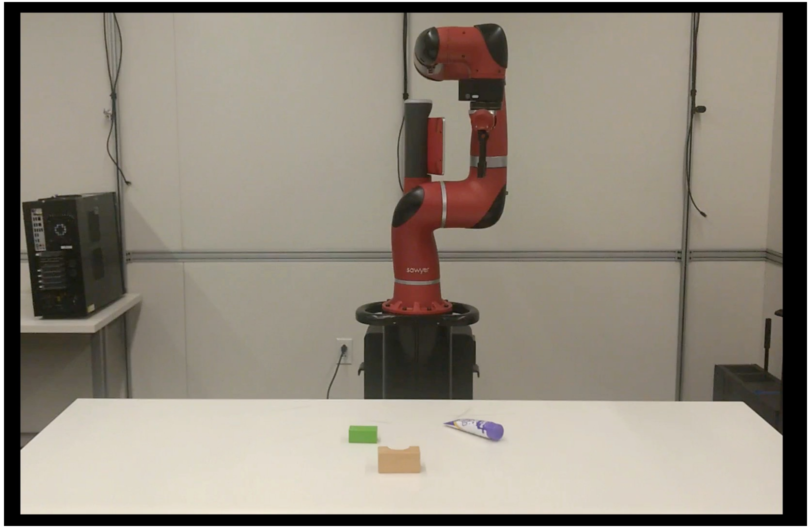
Figure 5: Performance of TRPO with and without VIME on the challenging hierarchical task SwimmerGather.

[Houthooft, Chen, Duan, Schulman, De Turck, Abbeel, 2017]

Intrinsic Reward for Reduction in Epistemic Uncertainty

Disagreement

Baseline: random exploration



Disagreement

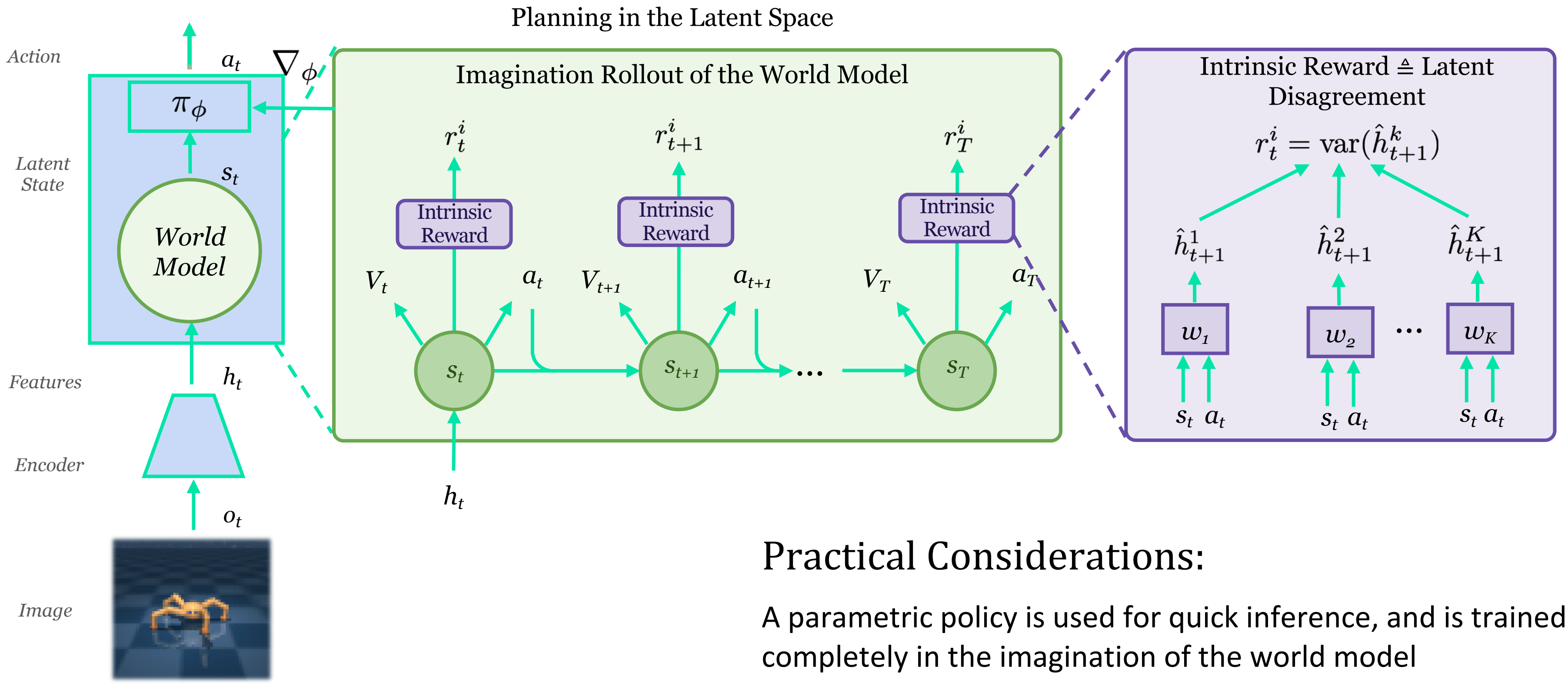
Setup: Overhead Camera; Action space = (position, direction, gripper angle, gripper opening)

[Pathak*, Gandhi*, Gupta, 2019]

Current Status

- Intrinsic Reward = (reduction in) epistemic uncertainty
- As agent collects high intrinsic reward data, the epistemic uncertainty gets driven down
- What might still cause agent inefficiency?
  - Agent only gets rewarded for (reduction in) epistemic uncertainty *after* it's achieved it. I.e. agent needs to be lucky, and will then be encouraged to learn from that luck.
  - More generally, can distinguish:
    - Retrospective signals
    - Prospectively seeking out signals

Planning to Achieve Disagreement

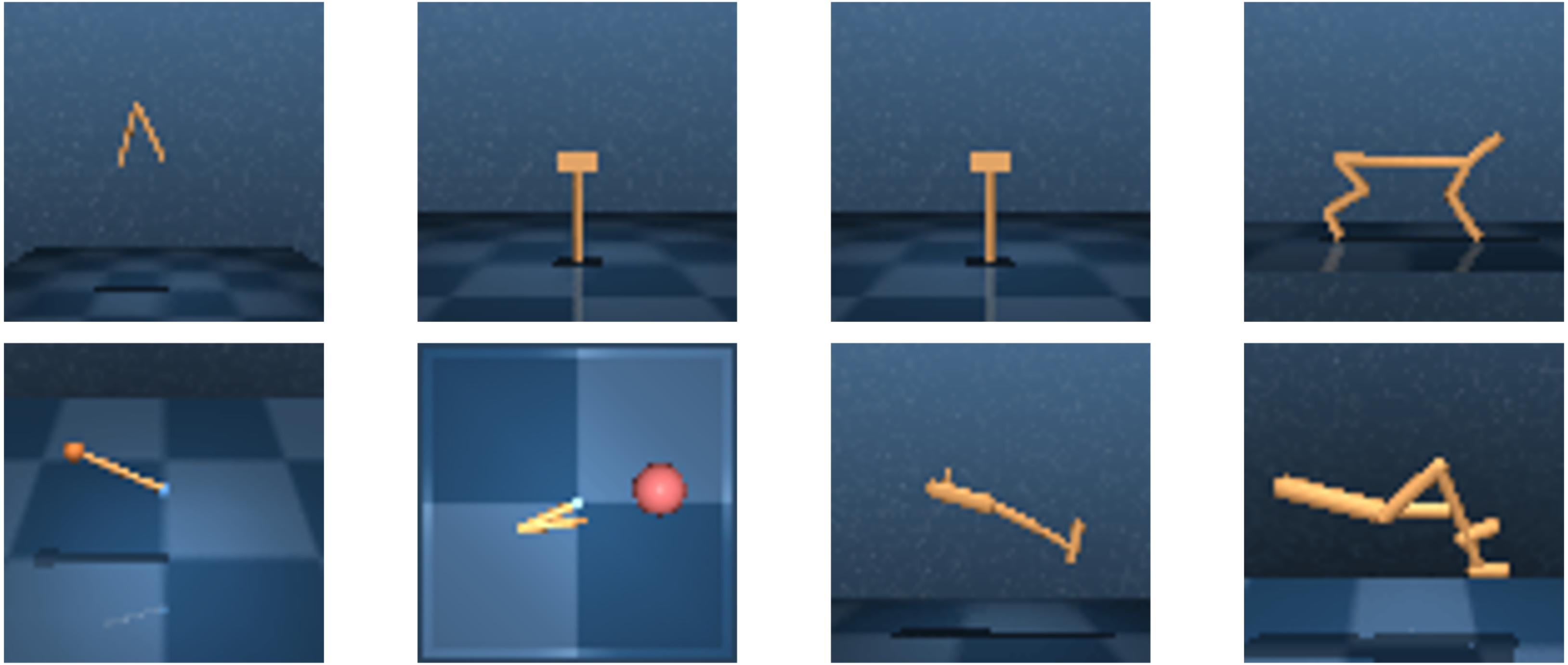


Practical Considerations:

A parametric policy is used for quick inference, and is trained completely in the imagination of the world model

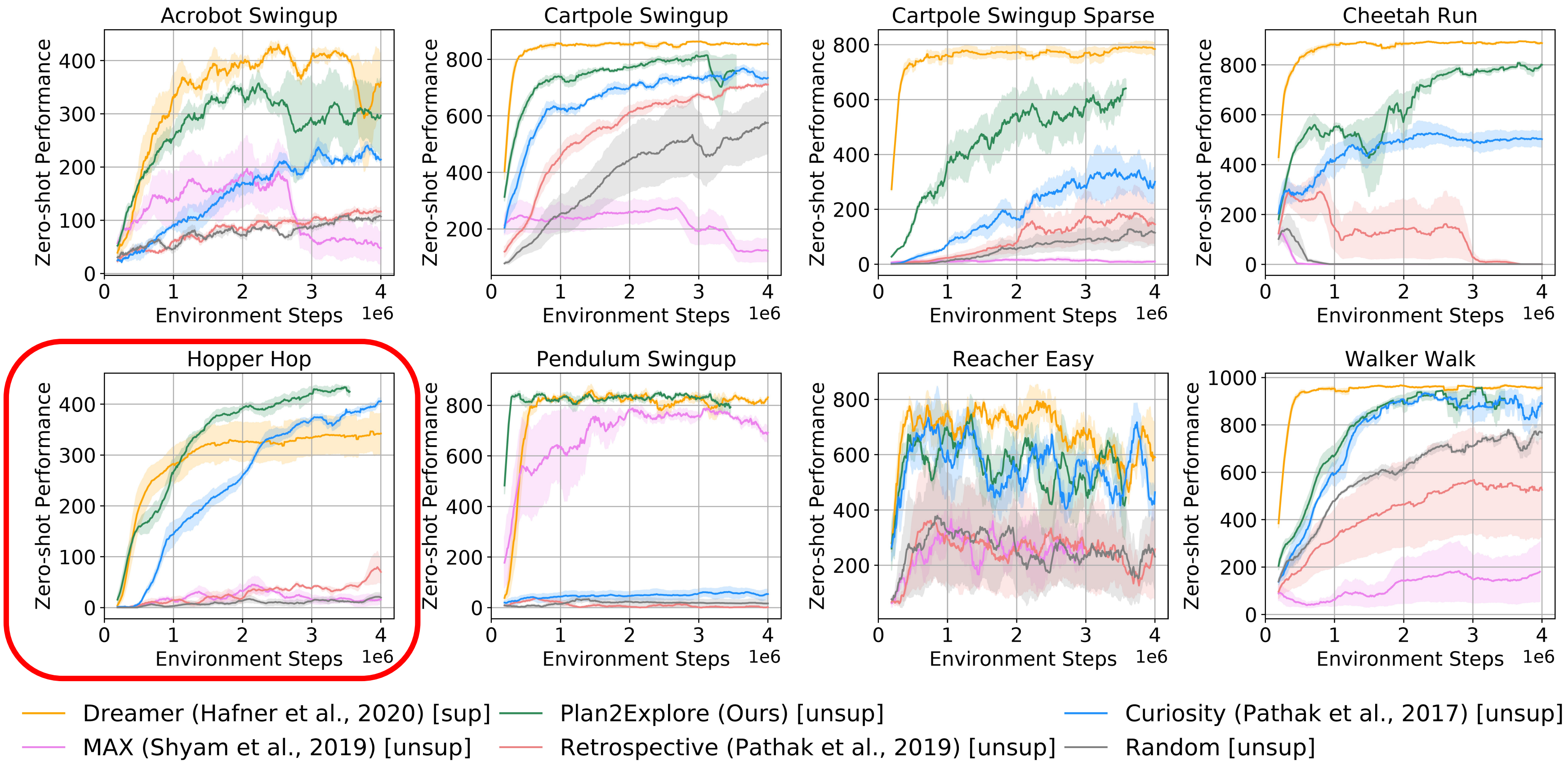
[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore] [Also: Shyam, Jaskowski, Gomez, 2018: MAX]

Self-Supervised Exploration Results



[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore]

Zero-Shot Reinforcement Learning



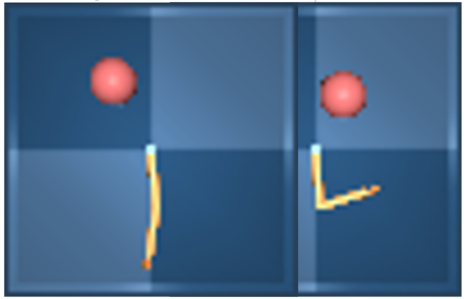
[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore]

Few-Shot Adaptation

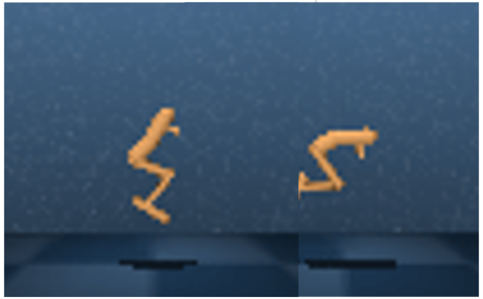
Our Agent (few-shot) Oracle (supervised)

Cheetah Run

Our Agent (few-shot) Oracle (supervised)



Reacher Easy

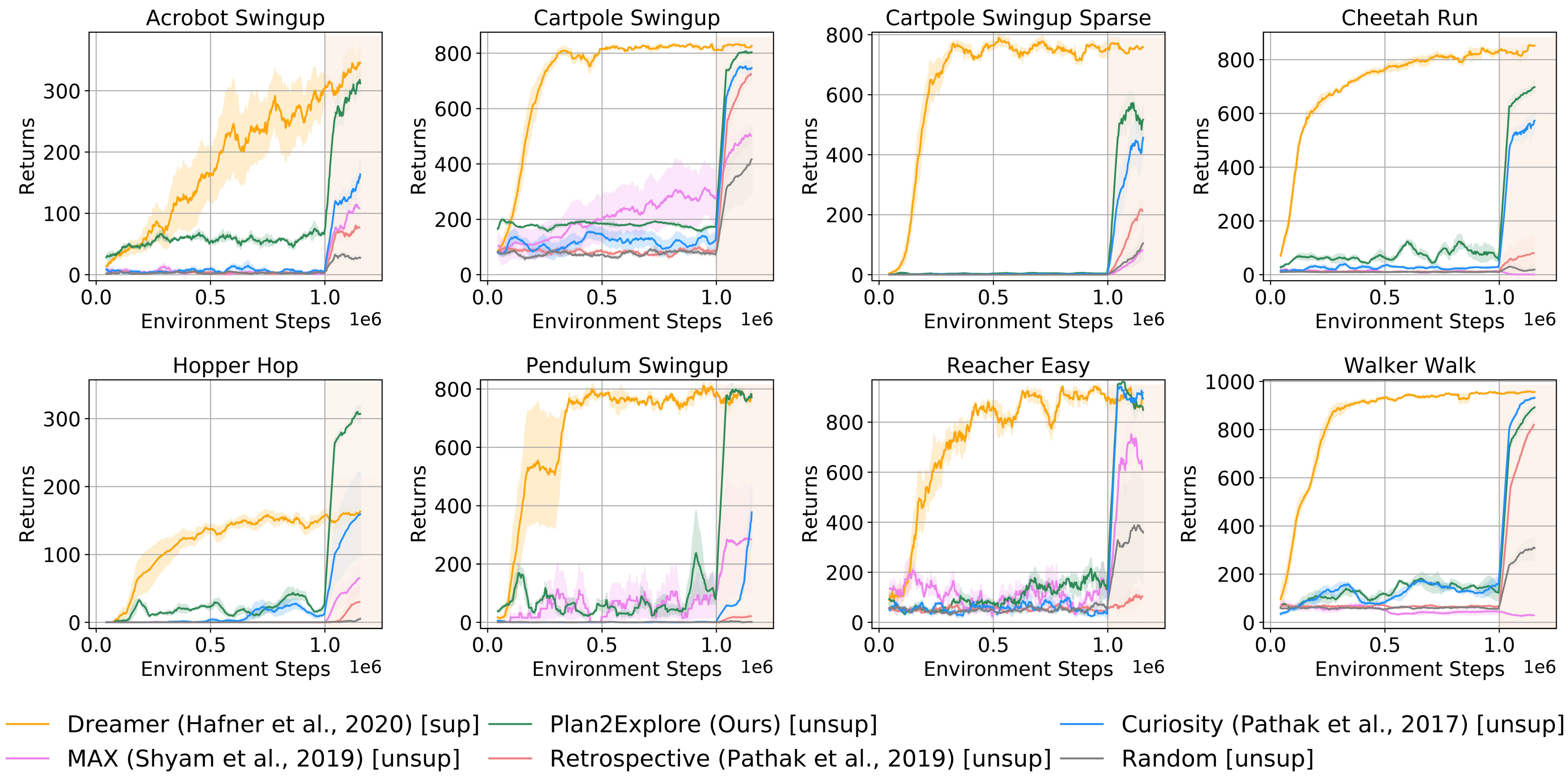


Hopper Hop

Walker Walk

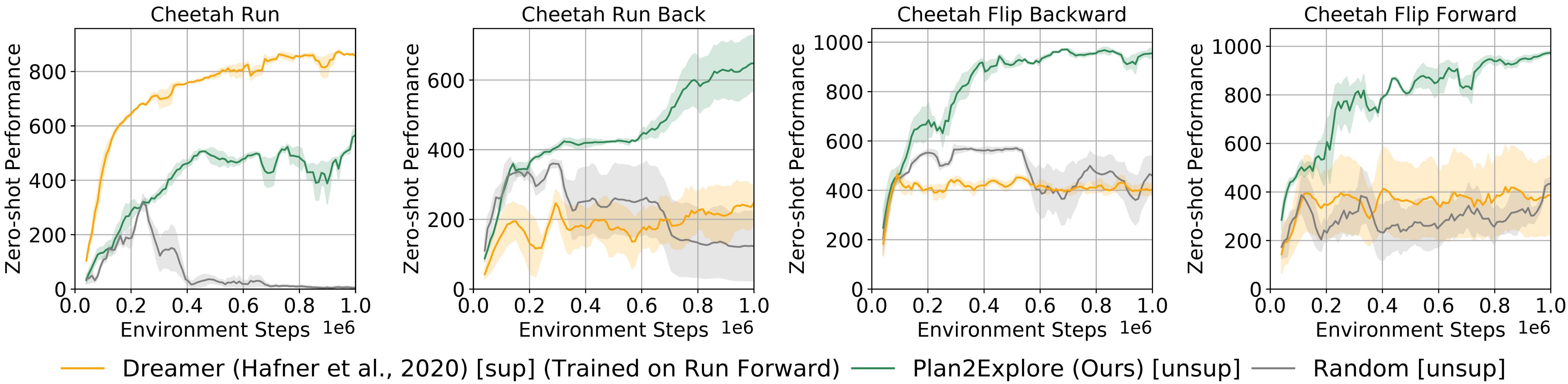
[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore]

Few-Shot Adaptation



[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore]

Can one model be used for multiple tasks?



[Sekar*, Rybkin*, Daniilidis, Abbeel, Hafner, Pathak, 2020: Planning to Explore]

Three Main Types of Intrinsic Reward

- Knowledge-based: Surprise / unpredictability / how much learned about world from experience

- ***<u>Competence-based</u>*: Empowerment / Skills**

- Data-based: Entropy (i.e. coverage) of data collected

Competences

- Why care about the agent's competences?
  - High competence is a natural desideratum after pre-training
  - Competence-based RFPT can be seen as end-to-end optimizing for the competence desideratum during RFPT
  - Other desiderata are possible, of course, e.g. 0-shot, fast fine-tuning, dataset coverage, high quality dynamics model
  - High competence might be achievable in "Phase 2" assuming great dataset coverage or great learned dynamics model

Formalizing Competence: Empowerment

- Formalizes and quantifies the degrees of freedom (or options) that an organism or agent has as a proxy for "preparedness"

- Behavioral Empowerment Hypothesis
  - "The adaptation brought about by natural evolution produced organisms that in absence of specific goals behave as if they were maximising their empowerment."

- Evolutionary Empowerment Hypothesis
  - "The adaptation brought about by natural evolution increases the empowerment of the resulting organism."

- AI Empowerment Hypothesis
  - "Empowerment provides a task-independent motivation that generate AI behaviour which is beneficial for a range of goal-oriented behaviour."

[Klyubin, Polani, Nehaniv, 2005: Empowerment: A Universal Agent-Centric Measure of Control] [Salge, Glackin, Polani, 2013: Empowerment: An Introduction]

Empowerment

$$\mathfrak{E}_t = C\Big(p(s_{t+n}|a_t^n)\Big) = \max_{p(a_t^n)} I(A_t^n; S_{t+n})$$

A: the agent's actuator[3] which takes values $a \in \mathcal{A}$
S: the agent's sensor which takes values $s \in \mathcal{S}$
M: the agent's internal state (or memory) which takes values $m \in \mathcal{M}$
R: the state of the environment which takes values $r \in \mathcal{R}$

- Measures the optionality a (perfect) agent can create over its future
- As phrased, is a number associated with a state + environment
- Empowerment can be increased by:
  - (1) going to more empowered states
  - (2) changing the agent/environment

[Klyubin, Polani, Nehaniv, 2005: Empowerment: A Universal Agent-Centric Measure of Control] [Salge, Glackin, Polani, 2013: Empowerment: An Introduction]

Empowerment

$$\mathfrak{E}_t = C\Big(p(s_{t+n}|a_t^n)\Big) = \max_{p(a_t^n)} I(A_t^n; S_{t+n})$$

So, what does this have to do with Exploration / RFPT / Competences?

An agent has maximal competence if it has figured out a policy that achieves the ”empowerment / channel capacity” available in its environment

→ Intrinsic Reward that measures the ***empowerment level of the agent’s policy***

(1) going to more empowered states
(2) changing the agent/environment

[Klyubin, Polani, Nehaniv, 2005: Empowerment: A Universal Agent-Centric Measure of Control] [Salge, Glackin, Polani, 2013: Empowerment: An Introduction]

Empowerment through Mutual Information Maximization

- Find policy $\pi_\theta(a \mid s, z)$ such that $MI(z\,;\tau)$ is maximized
- How to compute MI?
- $MI(z\,;\tau) = H(z) - H(z \mid \tau) = H(\tau) - H(\tau \mid z)$
- Both decompositions have been investigated (though mostly the first one)

Variational Approximation to Mutual Information

$$MI(z\,;\tau) = H(z) - H(z\mid\tau)$$

$\rightarrow$ $r_{\{intrinsic\}} = \log q_\phi(z\mid\tau) - \log p(z)$

- Intuition: need to be able to recover the "intent"/"latent" z from the trajectory; i.e. different z results in distinctly different trajectories
- For z discrete: this means training a classifier, and can set p(z) equal to uniform to maximize entropy

Variational Approximation to Mutual Information: $MI(z\,;\tau) = H(z) - H(z\,|\,\tau)$

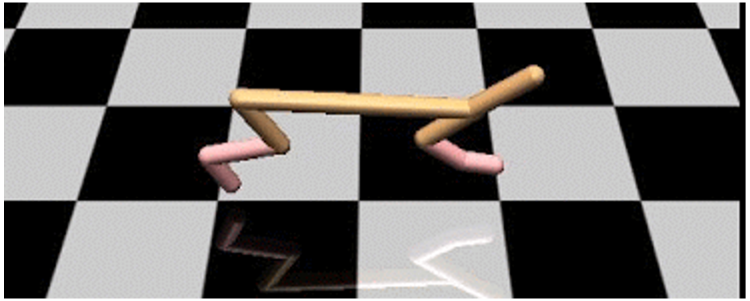
- SSN4HRL: $r_{\{intrinsic\}} = \log q_\phi(z \mid s_t) - \log p(z)$
- VIC: $r_{\{intrinsic\}} = \log q_\phi(z \mid s_H) - \log p(z)$
- DIAYN / VISR: $r_{\{intrinsic\}} = \log q_\phi(z \mid s_t) - \log p(z)$
- VALOR: $r_{\{intrinsic\}} = \log q_\phi(z \mid s_{1:H}) - \log p(z)$
- DISCERN: $r_{\{intrinsic\}} = \log q_\phi(z \mid s_{1:H}) - \log p(z)$

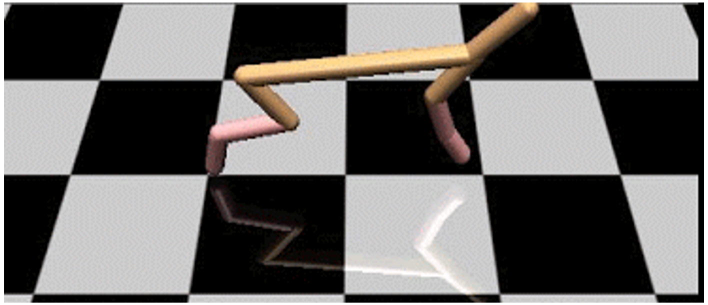
DISCERN is the only one to use continuous z, which doesn't allow for a classifier, so it uses a contrastive approximation to the first term

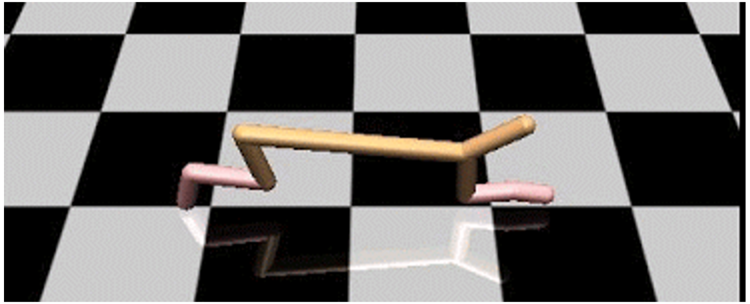
References for Previous Slide

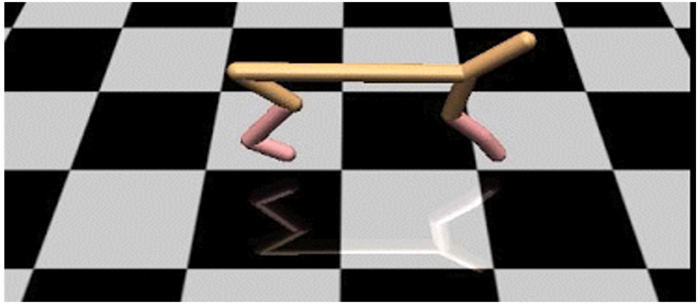
- SSN4HRL: Florensa, Duan, Abbeel, 2017
- VIC: Gregor, Rezende, Wierstra, 2016
- DIAYN: Eysenbach, Gupta, Ibarz, Levine, 2018
- VALOR: Achiam, Edwards, Amodei, Abbeel, 2018
- DISCERN: Warde-Farley, Van de Wiele, Kulkarni, Ionesu, Hansen, Mnih, 2018
- VISR: Hansen, Dabney, Barreto, Warde-Farley, Van de Wiele, Mnih, 2019

Results Highlights: DIAYN

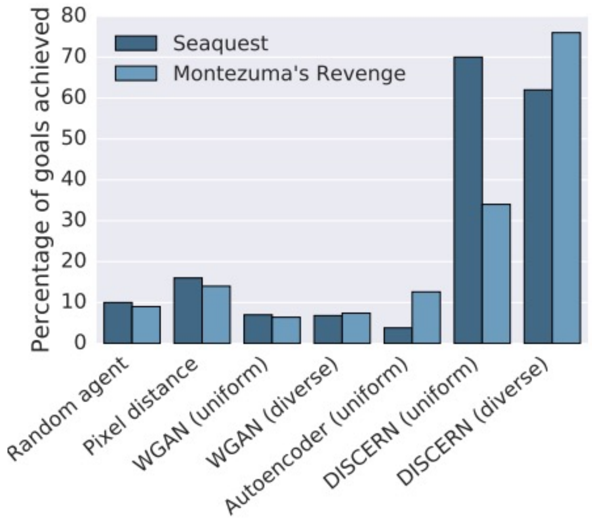




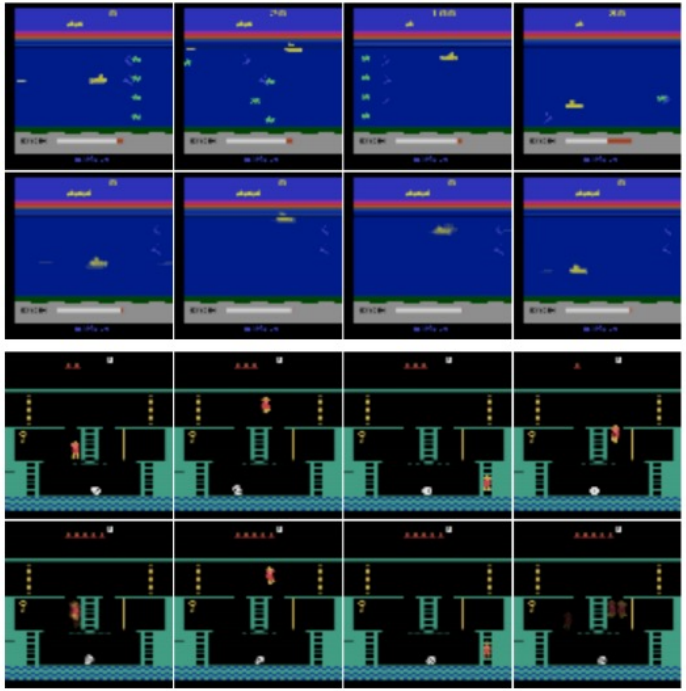




Results Highlights: DISCERN



(a)



(b)

Results Highlights: DISCERN

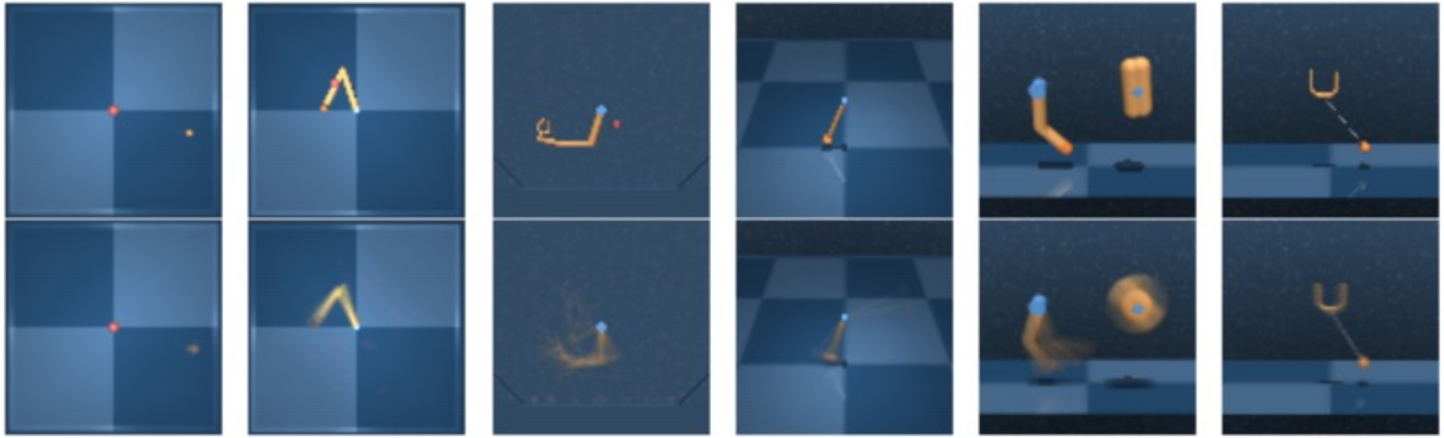


Figure 2: Average achieved frames for `point mass` (task `easy`), `reacher` (task `hard`), `manipulator` (task `bring ball`), `pendulum` (task `swingup`), `finger` (task `spin`) and `ball in cup` (task `catch`) environments. The goal is shown in the top row and the achieved frame is shown in the bottom row.

VISR

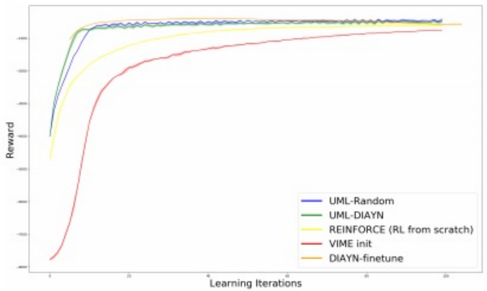
- Addresses slowness of fine-tuning
- Main ideas:
  - Same MI objective as DIAYN
  - Add success features [Barreto et al, 2017] to the agent

[Hansen et al, 2019]

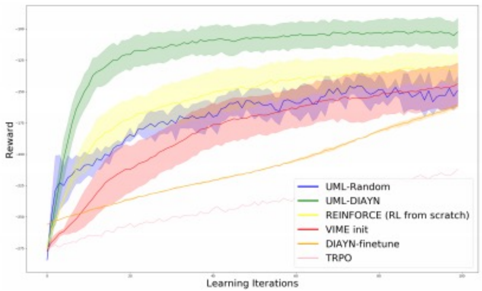
Unsupervised Meta-RL

- Challenge: how to quickly adapt to new rewards?
- Key Ideas:
  - Train with DIAYN (or similar)
  - After training, we can consider z as indexing into tasks, with reward log q(z|s)

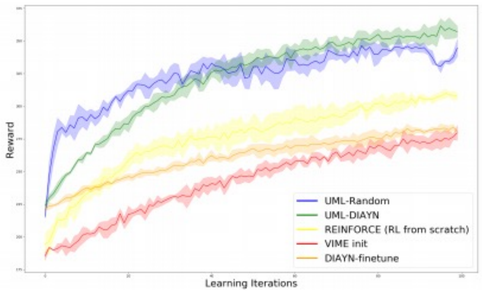
  → use this as tasks for Meta-RL training



2D navigation



Half-Cheetah



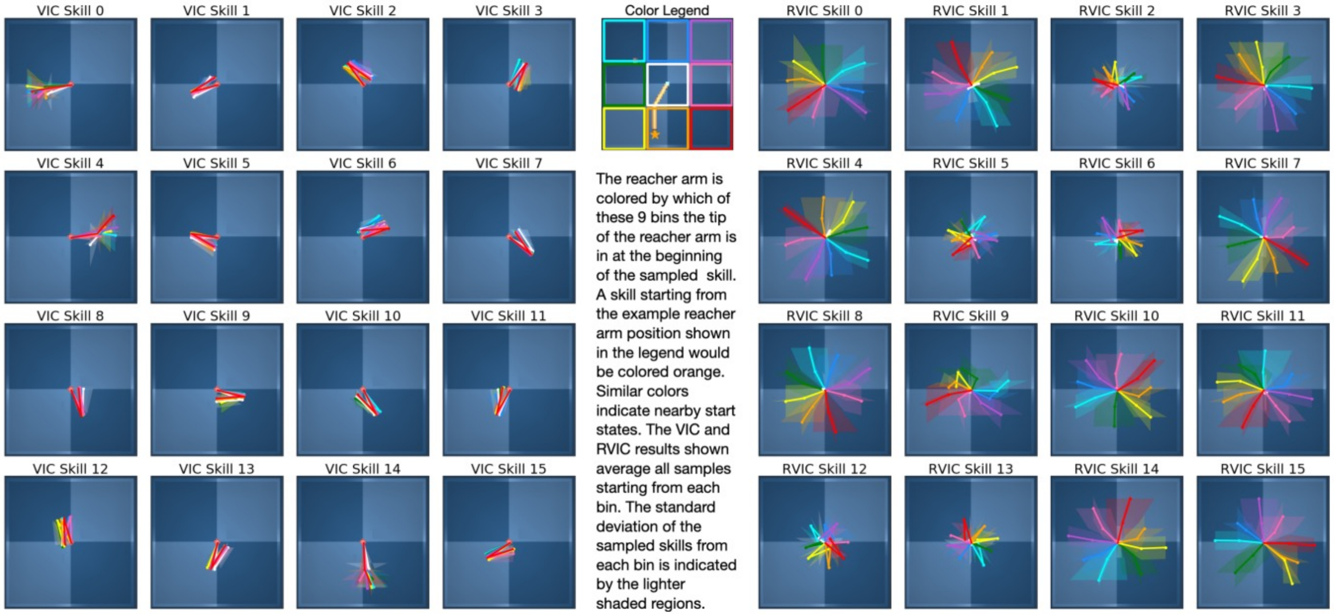
Ant

[Gupta*, Eysenbach*, Finn, Levine, 2018]

Relative VIC

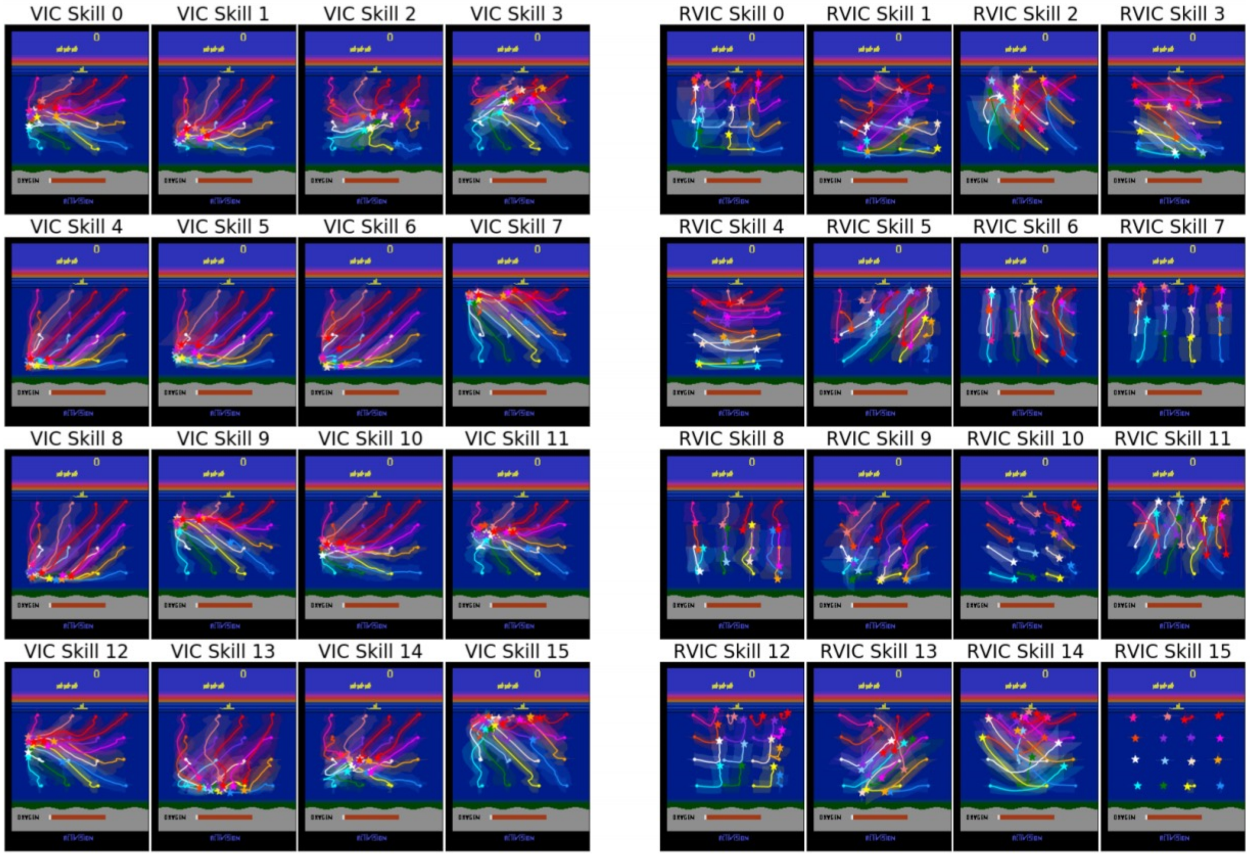
- Main idea: z (omega) more readily recoverable from s0 and sT than just sT

$$\log q_\phi(\Omega|s_T, s_0) - \log q_\psi^{\text{abs}}(\Omega|s_T)$$



[Baumli et al, 2020]

Relative VIC



[Baumli et al, 2020]

DADS

$$r_{\{intrinsic\}} = \log q_\phi(s_{t+1}|s_t, z) - \log q_\phi(s_{t+1} \mid s_t)$$

[DADS: Sharma, Gu, Levine, Kumar, Hausman, 2019]

Some Extra Perspective on MI(z; tau) = H(z) – H(z|tau)

- Different papers treat tau slightly differently, but the bigger difference might be in other experiment design factors:
  - Which RL algorithm is used?
  - State-based or image-based (only DISCERN is image based)
  - Termination conditions of episodes affect learning signal
  - When state-based, which variables to include in tau?
    - E.g. all state variables, vs. just x-y coordinates of ant robot?

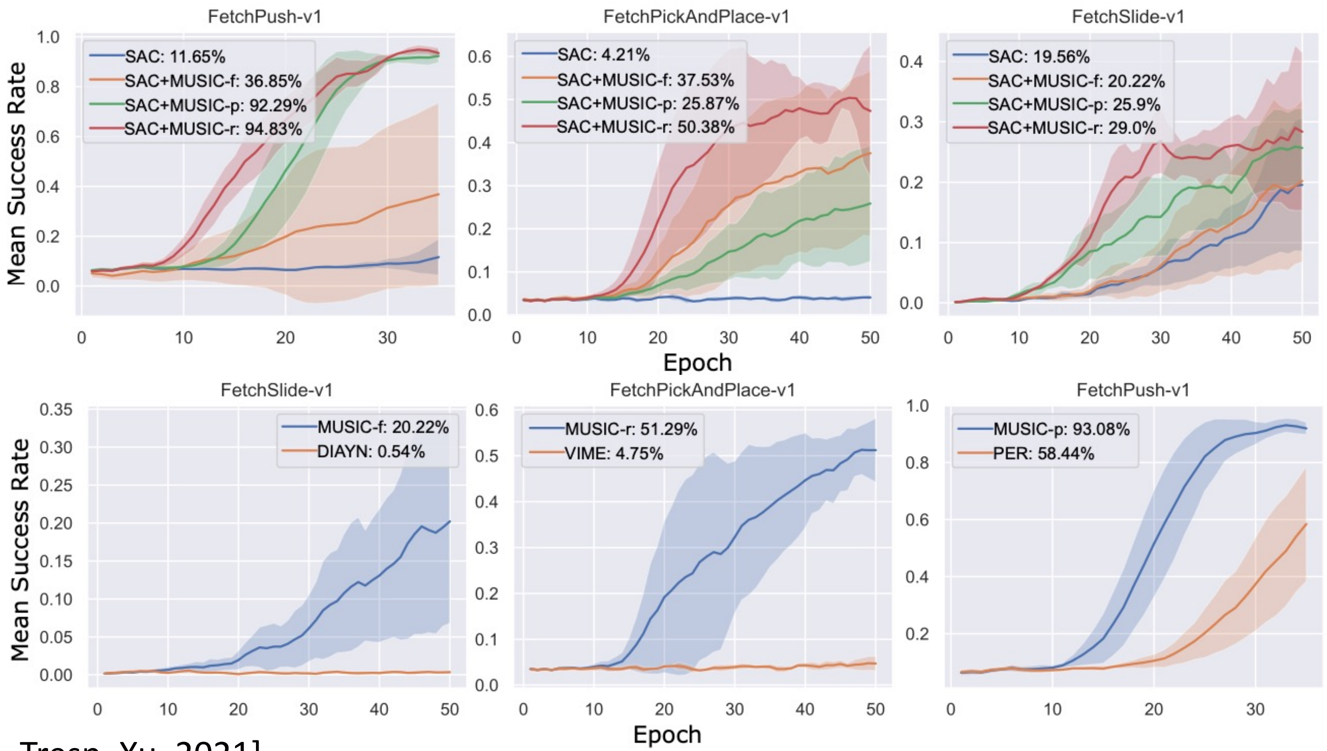
MUSIC: Mutual Info State Intrinsic Control

- Embraces there is a natural split of the state:
  - State of the agent (body)
  - State of the environment around the agent
- Optimizes MI(agent state ; env state)
  - i.e. incentivize the agent to visit agent states that ***affect*** the environment

[Zhao, Gao, Abbeel, Tresp, Xu, 2021]

MUSIC: Mutual Info State Intrinsic Control

- The proposed intrinsic reward help the agent to quickly learn to solve different downstream tasks.



[Zhao, Gao, Abbeel, Tresp, Xu, 2021]

More Perspective on MI(z; tau) = H(z) – H(z|tau)

- Limited exploration signal:
  - H(z) is maximized by simply sampling the latent from a high entropy distribution at the start of the trajectory
  - H(z|tau) encourages re-visiting states for which the classifier can recover z, which in turn likely leads to also visiting nearby states and gradual expansion, but not a very strong signal

  → Might want to combine with other exploration

How about the other decomposition of MI

- $MI(z\,;\tau) = H(\tau)\; - H(\tau\,|\,z)$
  - $H(\tau)$ directly encourages coverage / exploration
  - $-\,H(\tau\,|\,z)$ encourages predictability of the trajectory once we have decided on the latent z
- How to estimate entropy and conditional entropy over trajectories?
  - → recent work showing promising results, we'll look at this a bit later, when studying entropy based exploration/RFPT

Reverse Question: Can there be too much MI available?

- In case of very open-ended environments, info-theoretic approaches presented so far might keep busy forever
  - Showcase humanoid on floor in many configurations ,but never bothers to stand up…
- How to know what data is relevant to collect?
  - Option 1: passive data / human feedback
  - Option 2: try to more directly measure “skill” in terms of what the neural net policy could learn

    → ASP

Asymmetric Self-Play (ASP)

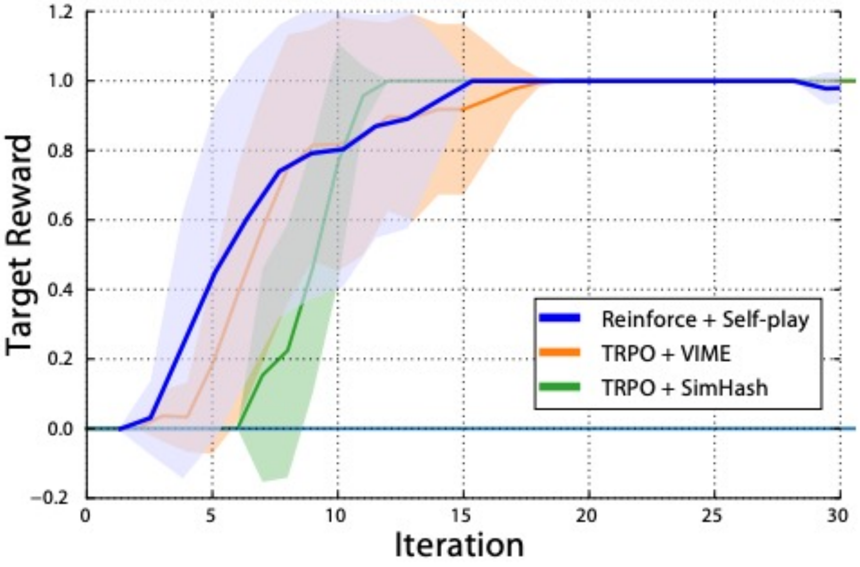
- Key Idea: two-player game, Alice challenges Bob, which encourages Alice to try new things (explore) and Bob to acquire skill
- Algorithm:
  - Alice does roll-out, gives final state as goal to Bob
  - Bob learns goal-conditioned policy with goals set by Alice
  - Alice intrinsic reward = $\max(0, t_B - t_A)$
  - Bob reward = $-t_B$

  t_A *and* t_B are completion times of Alice and Bob

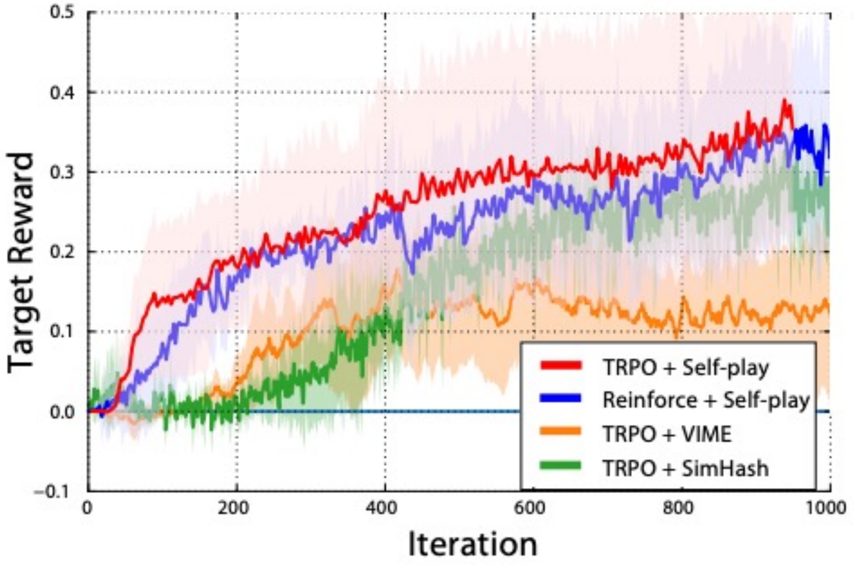
[Sikhbaatar, Lin, Kostrikov, Synnaeve, Szlam, Fergus, 2017]

Asymmetric Self-Play (ASP)

Mountain car

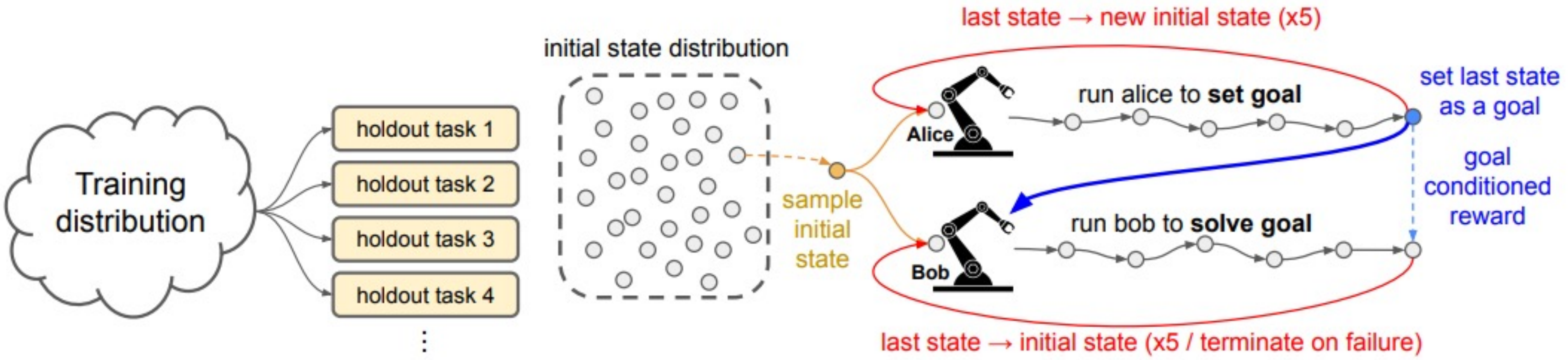


Swimmer Gather



[Sikhbaatar, Lin, Kostrikov, Synnaeve, Szlam, Fergus, 2017]

Scaled-up ASP

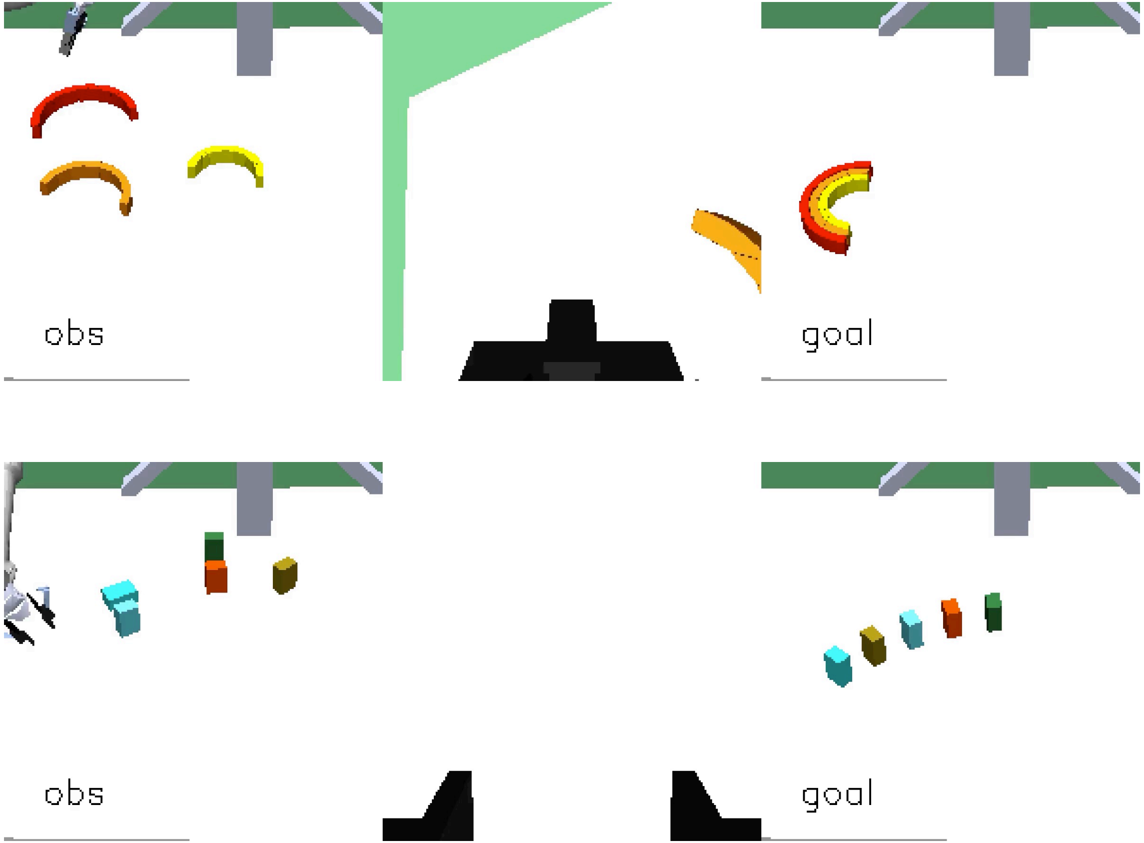


(a) Overall framework (b) Training distribution based on asymmetric self-play

- Goal setting filter: Goals only valid if an object was moved
- 1 billion steps

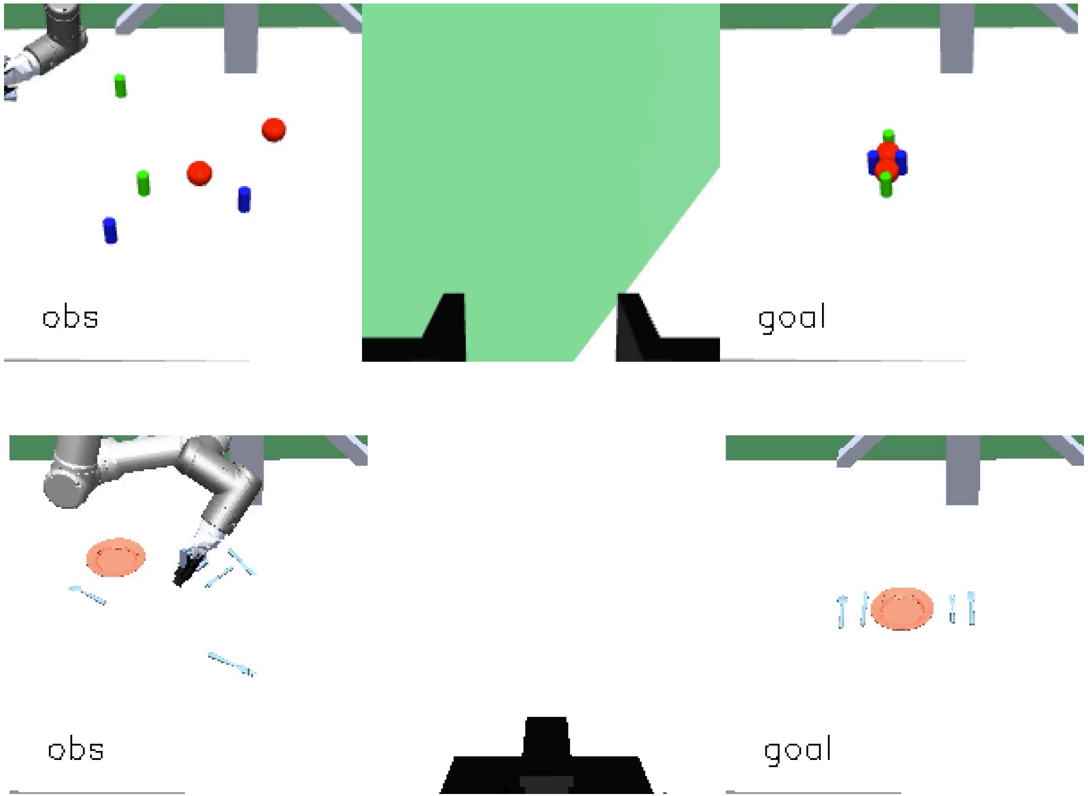
[OpenAI et al 2021: Asymmetric Self-Play for Automatic Goal Discovery in Robotic Manipulation]

Scaled-up ASP: 0-shot generalization



[OpenAI et al 2021: Asymmetric Self-Play for Automatic Goal Discovery in Robotic Manipulation]

Scaled-up ASP: 0-shot generalization



[OpenAI et al 2021: Asymmetric Self-Play for Automatic Goal Discovery in Robotic Manipulation]

Three Main Types of Intrinsic Reward

- Knowledge-based: Surprise / unpredictability / how much learned about world from experience

- Competence-based: Empowerment / Skills

- ***<u>Data-based:</u> Entropy (i.e. coverage) of data collected***

Tabular MDPs: Count-based Exploration Bonus

Model-Based Interval Estimation Exploration Bonus (MBIE-EB)

$$V(x) = \max_{a \in \mathcal{A}} \left[\hat{R}(x,a) + \gamma \, \mathbb{E}_{\hat{P}} \left[V(x') \right] + \beta N(x,a)^{-1/2} \right]$$

[Strehl & Littman, 2008]

High-D Spaces: Pseudo-Counts

- Bellemare et al 2016: Unifying Count-based Exploration and Intrinsic Motivation
  - learn P(s), change in P(s) can be connected to a pseudo-count
- Ostrovski et al 2017: Count-Based Exploration with Neural Density Models
  - Similar to above, significantly improved performance thanks to better density models
- Tang et al 2017: #exploration
  - NN hashes high-D observation into lower-dimensional space, use counts there
- Burda et al 2018: RND
  - Train a neural net to match the classification of a random neural net on observations encountered; as long as discrepancy on a new observation, count considered low

Density-based Pseudo-Counts

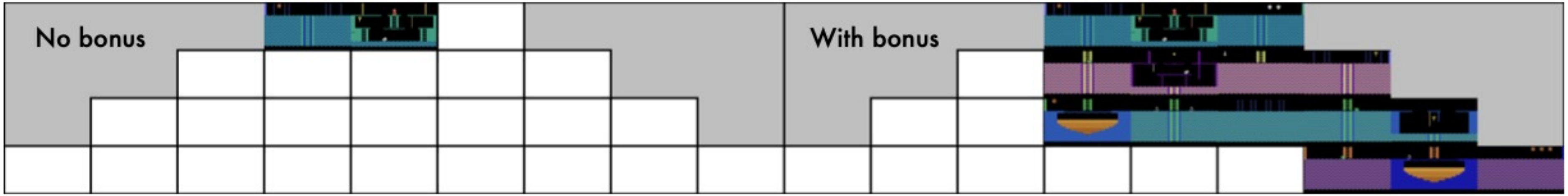


Figure 3: "Known world" of a DQN agent trained for 50 million frames with (**right**) and without (**left**) count-based exploration bonuses, in MONTEZUMA'S REVENGE.

[Bellemare, et al 2016]

Density-based Pseudo-Counts

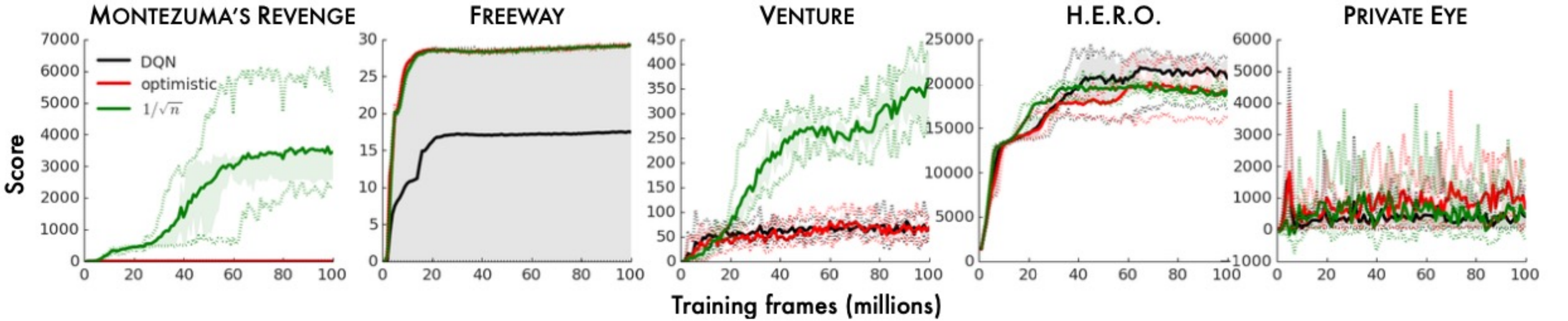


Figure 2: Average training score with and without exploration bonus or optimistic initialization in 5 Atari 2600 games. Shaded areas denote inter-quartile range, dotted lines show min/max scores.

[Bellemare, et al 2016]

Directly Optimizing Entropy of Data Collected

- Incentivizing exploration by introducing intrinsic rewards based on a measure of state novelty

- State entropy as intrinsic reward

$$r^{\texttt{intrinsic}} = \mathcal{H}(s) = -\mathbb{E}_{s \sim p(s)} \left[\log p(s) \right]$$

  - Maximizing state entropy ~= good state coverage

[MEPOL – Mutti, Pratissoli, Restelli, 2020]

Directly Optimizing Entropy of Data Collected

- Incentivizing exploration by introducing intrinsic rewards based on a measure of state novelty

- State entropy as intrinsic reward

$$r^{\texttt{intrinsic}} = \mathcal{H}(s) = -\mathbb{E}_{s \sim p(s)} [\log p(s)]$$

  - Maximizing state entropy ~= good state coverage

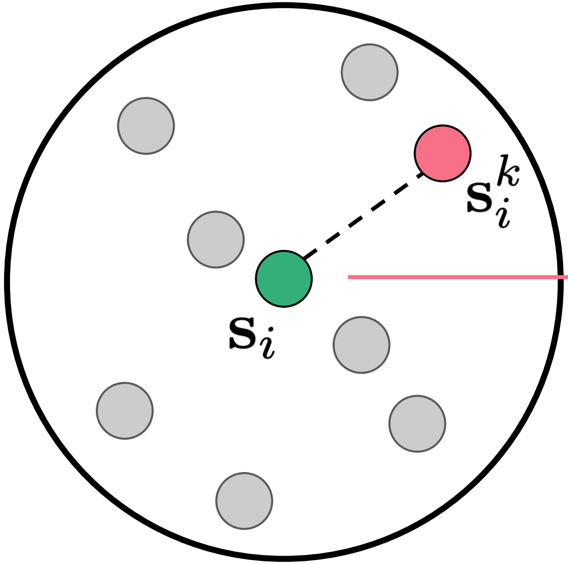
⇨ **Measuring state entropy is intractable** to compute in most setting

[MEPOL – Mutti, Pratissoli, Restelli, 2020]

K-Nearest-Neighbor Entropy Estimator

- *K*-nearest entropy estimator

$$\mathcal{H}(s) = -\mathbb{E}_{s \sim p(s)} [\underline{\log p(s)}]$$

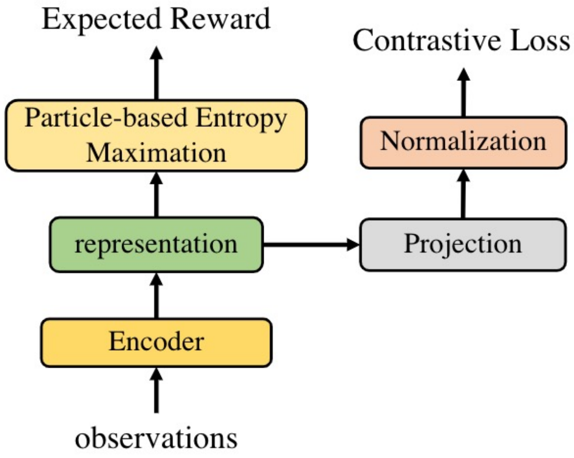


$$\widehat{\mathcal{H}}(\mathbf{s}) \propto \sum_i \boxed{\log(||\mathbf{s}_i - \mathbf{s}_i^k||)}$$

- Distribution → Store N number of visited states
- Compute the distance between each state and its K-NN

Singh, H., et al., 2003. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, *23*(3-4), pp.301-321.
MEPOL – Mutti, Pratissoli, Restelli, 2020

APT: Active Pre-Training

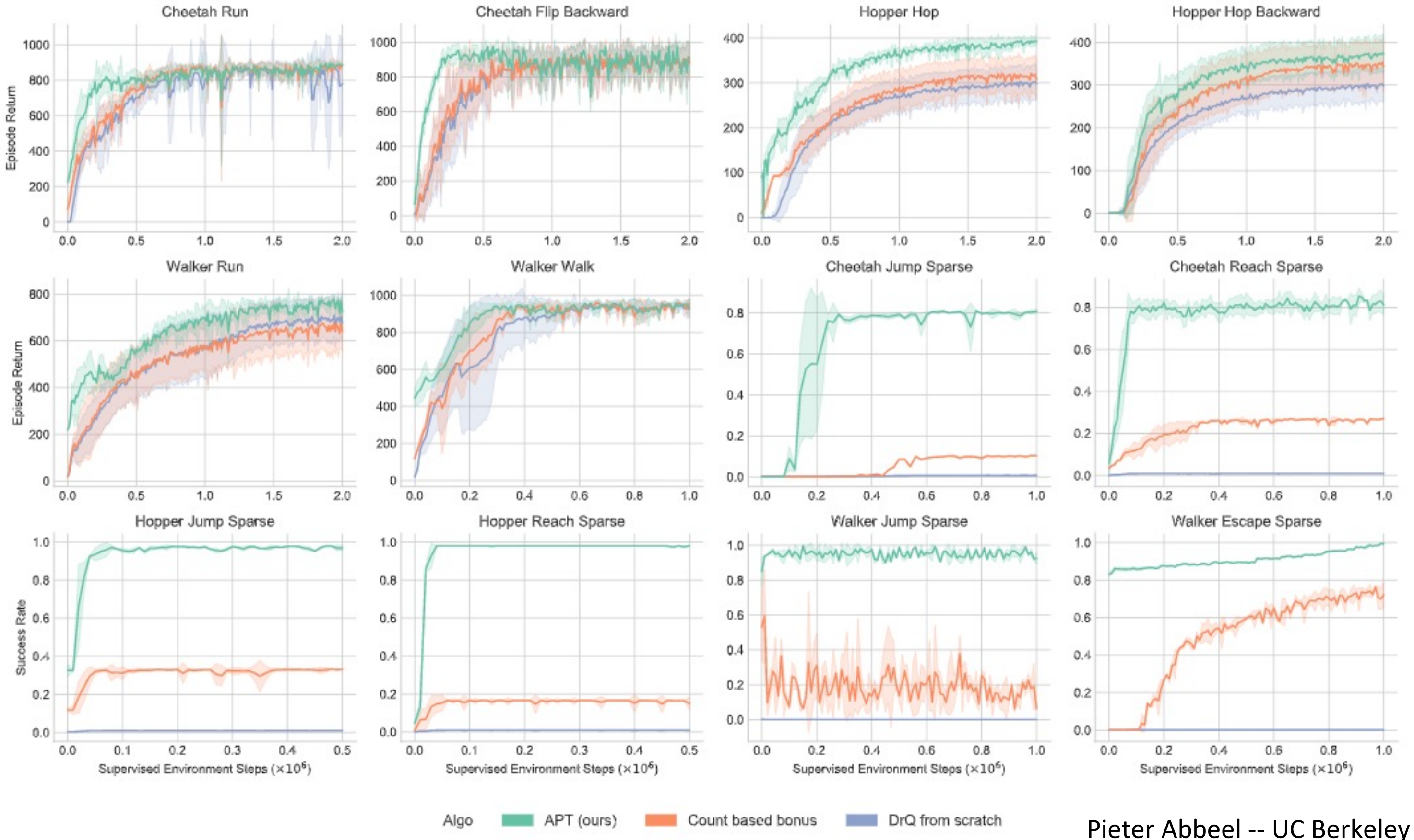


Algorithm 1: Training APT

Randomly Initialize f encoder
Randomly Initialize π and Q networks
for $e := 1, \infty$ **do**
  for $t := 1, T$ **do**
    Receive observation s_t from environment
    Take action $a_t \sim \pi(\cdot|s_t)$, receive observation s_{t+1} and $\cancel{r_t}$ from environment
    $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, \cancel{r_t}, s_t')$
    $\{(s_i, a_i, \cancel{r_i}, s_i')\}_{i=1}^{N} \sim \mathcal{D}$ // sample a mini batch
    Train neural encoder f on mini batch // representation learning
    for *each* $i = 1..N$ **do**
      $a_i' \sim \pi(\cdot|s_i')$
      $\hat{Q}_i = Q_{\theta'}(s_i', a_i')$
      Compute r_{APT} with equation (5) // particle-based entropy reward
      $y_i \leftarrow r_{\text{APT}} + \gamma \hat{Q}_i$
    end
    $loss_Q = \sum_i (Q(s_i, a_i) - y_i)^2$
    Gradient descent step on Q and π // standard Q-learning
  end
end

[H Liu & P Abbeel, 2020]

Experiments: DM Control Suite

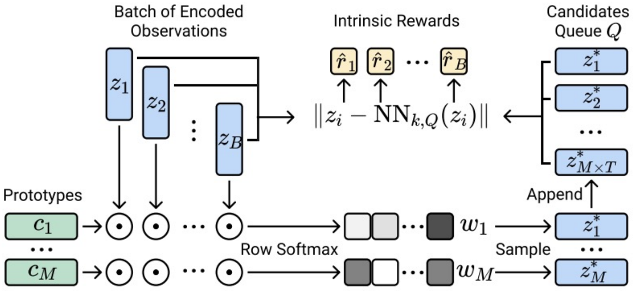


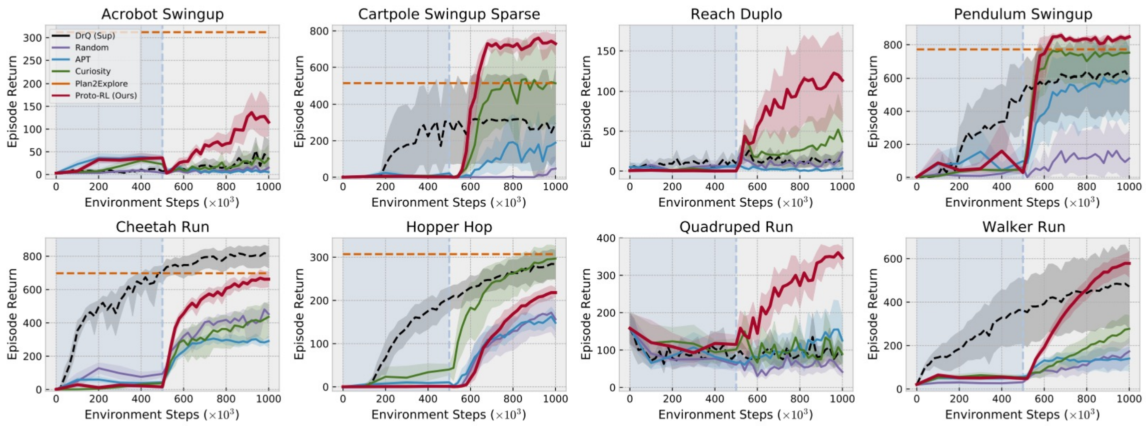
Experiments: Atari

Game	Random	Human	SimPLe	DER	CURL	DrQ	SPR	VISR	APT (ours)
Alien	227.8	7127.7	616,9	739.9	558.2	771.2	801.5	364.4	**2614.8**
Amidar	5.8	1719.5	88.0	188.6	142.1	102.8	176.3	186.0	**211.5**
Assault	222.4	742.0	527.2	431.2	600.6	452.4	571.0	**12091.1**	891.5
Asterix	210.0	8503.3	1128.3	470.8	734.5	603.5	977.8	**6216.7**	185.5
Bank Heist	14.2	753.1	34.2	51.0	131.6	168.9	380.9	71.3	**416.7**
BattleZone	2360.0	37187.5	5184.4	10124.6	14870.0	12954.0	**16651.0**	7072.7	7065.1
Boxing	0.1	12.1	9.1	0.2	1.2	6.0	**35.8**	13.4	21.3
Breakout	1.7	30.5	16.4	1.9	4.9	16.1	17.1	**17.9**	10.9
ChopperCommand	811.0	7387.8	**1246.9**	861.8	1058.5	780.3	974.8	800.8	317.0
Crazy Climber	10780.5	23829.4	**62583.6**	16185.2	12146.5	20516.5	42923.6	49373.9	44128.0
Demon Attack	107805	35829.4	62583.6	16185.3	12146.5	20516.5	42923.6	**8994.9**	5071.8
Freeway	0.0	29.6	20.3	27.9	26.7	9.8	24.4	-12.1	**29.9**
Frostbite	65.2	4334.7	254.7	866.8	1181.3	331.1	**1821.5**	230.9	1796.1
Gopher	257.6	2412.5	771.0	349.5	669.3	636.3	715.2	498.6	**2590.4**
Hero	1027.0	30826.4	2656.6	6857.0	6279.3	3736.3	**7019.2**	663.5	6789.1
Jamesbond	29.0	302.8	125.3	301.6	471.0	236.0	365.4	**484.4**	356.1
Kangaroo	52.0	3035.0	323.1	779.3	872.5	940.6	**3276.4**	1761.9	412.0
Krull	1598.0	2665.5	**4539.9**	2851.5	4229.6	4018.1	2688.9	3142.5	2312.0
Kung Fu Master	258.5	22736.3	17257.2	14346.1	14307.8	9111.0	13192.7	16754.9	**17357.0**
Ms Pacman	307.3	6951.6	1480.0	1204.1	1465.5	960.5	1313.2	558.5	**2827.1**
Pong	-20.7	14.6	**12.8**	-19.3	-16.5	-8.5	-5.9	-26.2	-8.0
Private Eye	24.9	69571.3	58.3	97.8	218.4	-13.6	**124.0**	98.3	96.1
Qbert	163.9	13455.0	1288.8	1152.9	1042.4	854.4	669.1	666.3	**17671.2**
Road Runner	11.5	7845.0	5640.6	9600.0	5661.0	8895.1	**14220.5**	6146.7	4782.1
Seaquest	68.4	42054.7	683.3	354.1	384.5	301.2	583.1	706.6	**2116.7**
Up N Down	533.4	11693.2	3350.3	2877.4	2955.2	3180.8	**28138.5**	10037.6	8289.4
Mean HNS	0.000	1.000	44.3	28.5	38.1	35.7	**70.4**	64.31	69.55
Median HNS	0.000	1.000	14.4	16.1	17.5	26.8	41.5	12.36	**47.50**
# Superhuman	0	N/A	2	2	2	2	**7**	6	**7**

How about size of replay buffer for entropy estimates?

→ Keep around cluster representatives for entropy estimation





Reinforcement Learning with Prototypical Representations, Yarats, Fergus, Lazaric, Pinto, 2021

Active Pre-Training with Successor Features (APS)

- Similar to how VISR added Successor Features to DIAYN for faster adaptation

- APS add Successor Features to APT

[APS: Liu & Abbeel, 2021]

APS on Atari

Game	Random	Human	SimPLe	DER	CURL	DrQ	SPR	VISR	APT	APS (ours)
Alien	227.8	7127.7	616,9	739.9	558.2	771.2	801.5	364.4	**2614.8**	934.9
Amidar	5.8	1719.5	88.0	188.6	142.1	102.8	176.3	186.0	**211.5**	178.4
Assault	222.4	742.0	527.2	431.2	600.6	452.4	571.0	**12091.1**	891.5	413.3
Asterix	210.0	8503.3	1128.3	470.8	734.5	603.5	977.8	**6216.7**	185.5	1159.7
Bank Heist	14.2	753.1	34.2	51.0	131.6	168.9	380.9	71.3	**416.7**	262.7
BattleZone	2360.0	37187.5	5184.4	10124.6	14870.0	12954.0	16651.0	7072.7	7065.1	**26920.1**
Boxing	0.1	12.1	9.1	0.2	1.2	6.0	35.8	13.4	21.3	**36.3**
Breakout	1.7	30.5	16.4	1.9	4.9	16.1	17.1	17.9	10.9	**19.1**
ChopperCommand	811.0	7387.8	1246.9	861.8	1058.5	780.3	974.8	800.8	317.0	**2517.0**
Crazy Climber	10780.5	23829.4	62583.6	16185.2	12146.5	20516.5	42923.6	49373.9	44128.0	**67328.1**
Demon Attack	107805	35829.4	62583.6	16185.3	12146.5	20516.5	42923.6	**8994.9**	5071.8	7989.0
Freeway	0.0	29.6	20.3	27.9	26.7	9.8	24.4	-12.1	**29.9**	27.1
Frostbite	65.2	4334.7	254.7	866.8	1181.3	331.1	**1821.5**	230.9	1796.1	496.5
Gopher	257.6	2412.5	771.0	349.5	669.3	636.3	715.2	498.6	**2590.4**	2386.5
Hero	1027.0	30826.4	2656.6	6857.0	6279.3	3736.3	7019.2	663.5	6789.1	**12189.3**
Jamesbond	29.0	302.8	125.3	301.6	471.0	236.0	365.4	484.4	356.1	**622.3**
Kangaroo	52.0	3035.0	323.1	779.3	872.5	940.6	3276.4	1761.9	412.0	**5280.1**
Krull	1598.0	2665.5	**4539.9**	2851.5	4229.6	4018.1	2688.9	3142.5	2312.0	4496.0
Kung Fu Master	258.5	22736.3	17257.2	14346.1	14307.8	9111.0	13192.7	16754.9	17357.0	**22412.0**
Ms Pacman	307.3	6951.6	1480.0	1204.1	1465.5	960.5	1313.2	558.5	**2827.1**	2092.3
Pong	-20.7	14.6	**12.8**	-19.3	-16.5	-8.5	-5.9	-26.2	-8.0	12.5
Private Eye	24.9	69571.3	58.3	97.8	218.4	-13.6	**124.0**	98.3	96.1	117.9
Qbert	163.9	13455.0	1288.8	1152.9	1042.4	854.4	669.1	666.3	17671.2	**19271.4**
Road Runner	11.5	7845.0	5640.6	9600.0	5661.0	8895.1	**14220.5**	6146.7	4782.1	5919.0
Seaquest	68.4	42054.7	683.3	354.1	384.5	301.2	583.1	706.6	2116.7	**4209.7**
Up N Down	533.4	11693.2	3350.3	2877.4	2955.2	3180.8	**28138.5**	10037.6	8289.4	4911.9
Mean Human-Norm'd	0.000	1.000	44.3	28.5	38.1	35.7	70.4	64.31	69.55	**99.04**
Median Human-Norm'd	0.000	1.000	14.4	16.1	17.5	26.8	41.5	12.36	47.50	**58.80**
# Superhuman	0	N/A	2	2	2	2	7	6	7	**8**

[APS: Liu & Abbeel, 2021]

Never Give Up

- Main idea:
  - Short-term exploration through particle-based entropy
  - Long-term exploration through RND

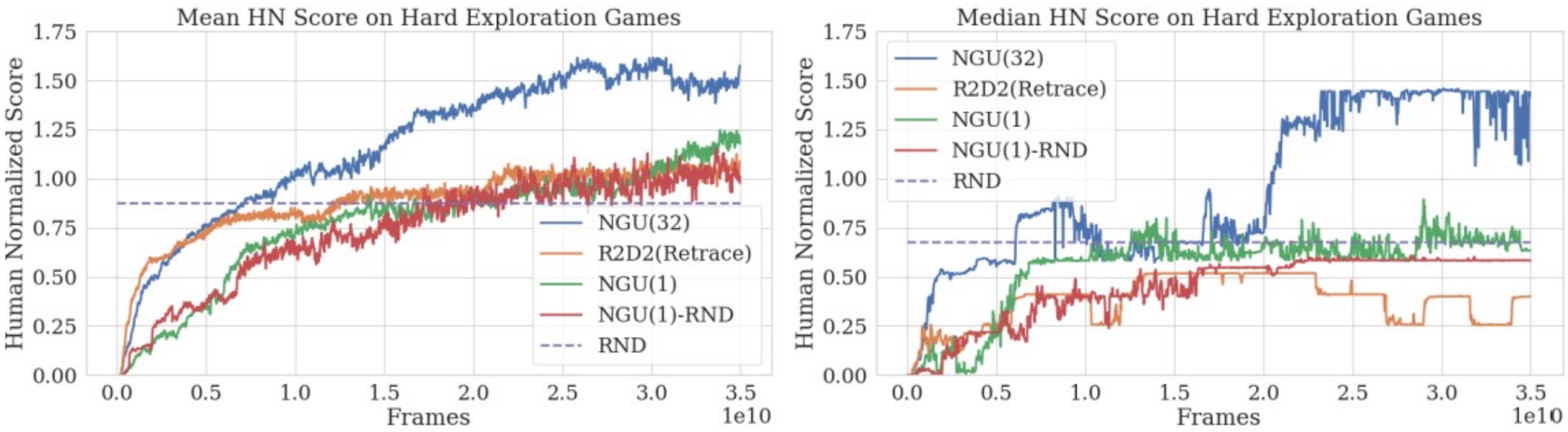


Figure 4: Human Normalized Scores on the 6 hard exploration games.

[Never Give Up: Learning Directed Exploration Strategies, Badia et al, 2020]

Some Theory

Provably Efficient Maximum Entropy Exploration

Elad Hazan[1 2] Sham M. Kakade[1 3 4] Karan Singh[1 2] Abby Van Soest[2]

[1] Google AI Princeton
[2] Department of Computer Science, Princeton University
[3] Allen School of Computer Science and Engineering, University of Washington
[4] Department of Statistics, University of Washington
{ehazan,karans,asoest}@princeton.edu, sham@cs.washington.edu

Kinematic State Abstraction and Provably Efficient Rich-Observation Reinforcement Learning

Dipendra Misra, Mikael Henaff, Akshay Krishnamurthy, and John Langford

Microsoft Research, New York, NY

Outline

- Problem Motivation
- Baseline RL Algorithms Refresher
- Intrinsic Rewards for Reward-Free Pre-Training and Exploration
- ***Algorithmic Approaches to Exploration*** ***(can complement intrinsic reward RFPT!)***
- Algorithmic Approaches to Reward-Free Pre-Training

Key Idea: Optimism in Face of Uncertainty

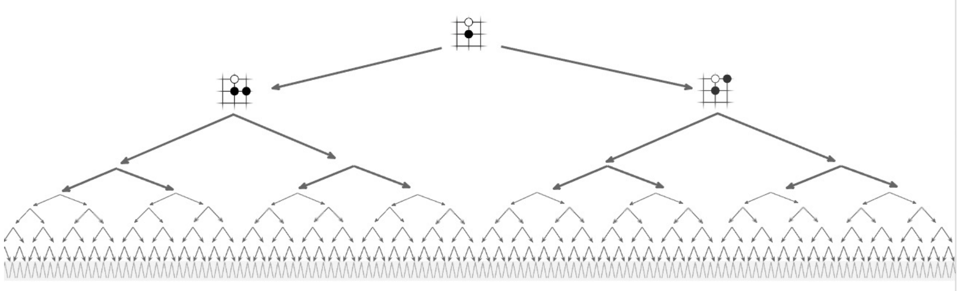
- Key Idea permeating most works: Optimism in Face of Uncertainty
  - If haven't visited a state, let's assume it might have high reward, until we experience otherwise

- Classic references (experiments largely tabular / linear)

Lai and Robbins' Upper Confidence Bounds (UCB) 1985;
Sutton's Dyna, 1990;
Schmidhuber's Curiosity 1991;
Kaelbling's Interval Exploration 1993;
Moore & Atkeson's Prioritized Sweeping 1993;
Kaelbling, et al, RL Survey 1996;
Kearns and Singh, E3 2002;
Brafman and Tennenholtz, RMax 2002
Peter Auer's UCB regret bounds, 2002;
Strehl and Littman, Model-based Interval Estimation (MBIE), 2008

Algorithmic Exploration: MCTS

AlphaGo



- MCTS Assumes: (i) model-based RL OR (ii) sim with resets to any previously experienced state
- PUCB value: $$u(a) = v(a) + p(a) \cdot pb_c$$

$$pb_c = \frac{\sqrt{visit_{parent}}}{visit_{child}+1} \cdot \left(log\left(\frac{visit_{parent} + pb_{base} + 1}{pb_{base}}\right) + pb_{init} \right)$$

→ Favor less frequently visited children in the tree

[Coulom 2008; Kocsis & Szepesvari 2006; Gelly, Wang, Munos, Teytaud, 2006; Silver et al 2015]

Algorithmic Exploration: Q Ensembles

Q: How to more directly represent posterior over value functions?

A: DQN with an ensemble of Q functions

Q: How to use this posterior?

A1: Posterior Sampling / Thompson Sampling / “Bootstrap”

A2: Create Upper Confidence Bound (UCB)

Algorithmic Exploration: Q Ensembles: Thompson Sampling / Bootstrap

Key idea: Rather than independent action selection in each step, use the same member of the Q-ensemble for the entire roll-out

→ more consistent behavior, which aids exploration

Figure 3: Scalable environments that requires deep exploration.

[Osband, Blundell, Pritzel, Van Roy, Deep Exploration via Bootstrapped DQN, 2016]

Algorithmic Exploration: Q Ensembles: Thompson Sampling / Bootstrap

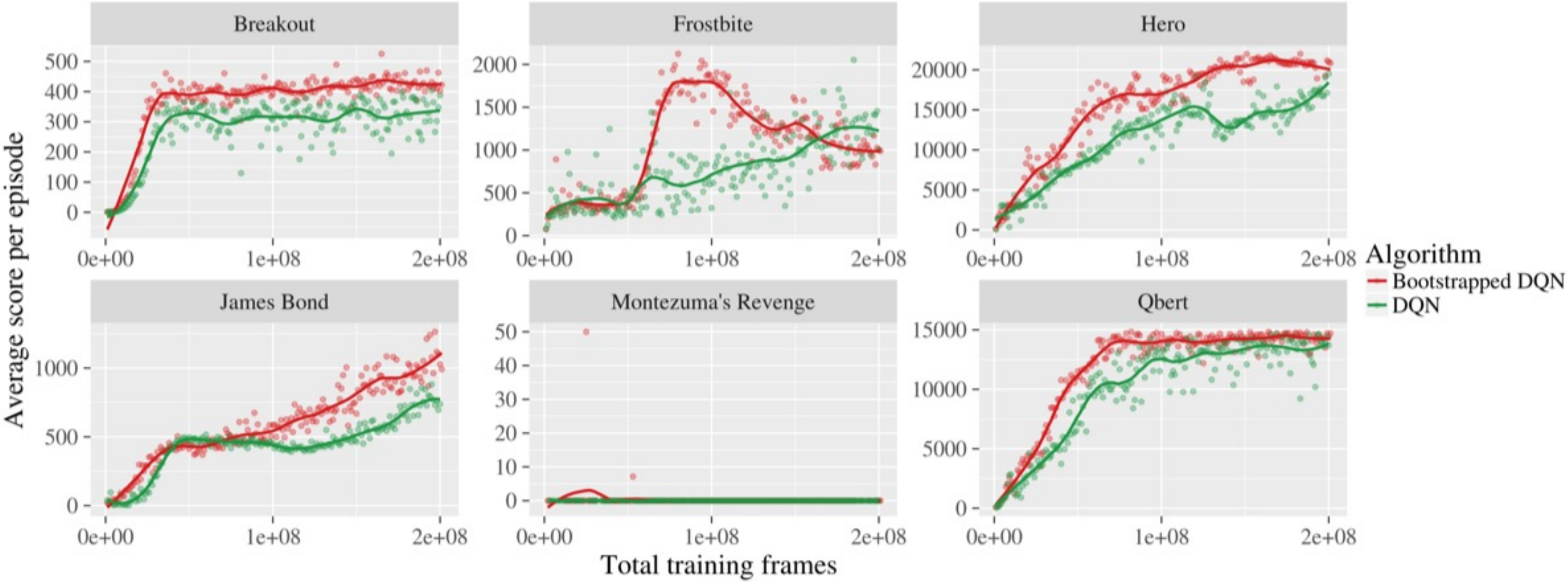


Figure 6: Bootstrapped DQN drives more efficient exploration.

[Osband, Blundell, Pritzel, Van Roy, Deep Exploration via Bootstrapped DQN, 2016]

Algorithmic Exploration: Q-Ensembles: Q-UCB

- How to add UCB to deep Q-learning (DQN)?

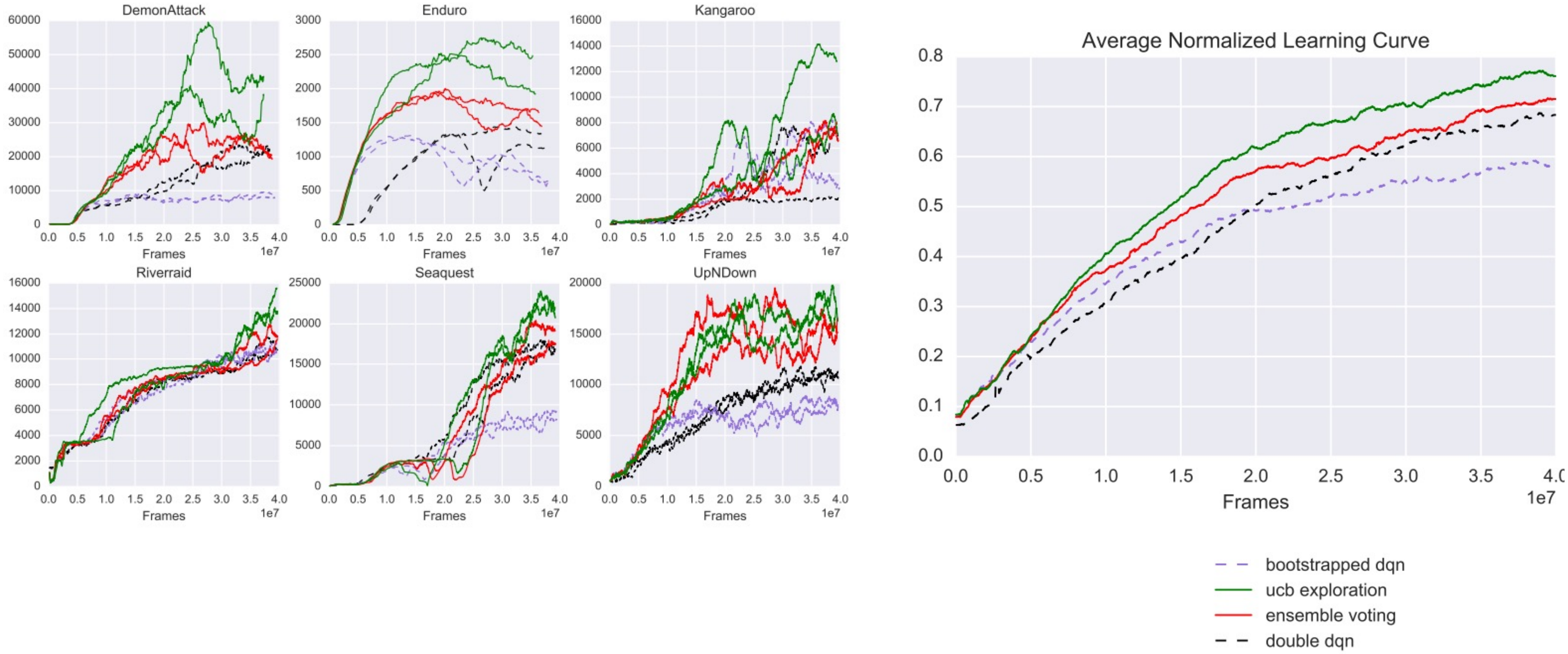
Algorithm 2 UCB Exploration with Q-Ensembles

Input: Value function networks Q with K outputs $\{Q_k\}_{k=1}^{K}$. Hyperparameter λ.
Let B be a replay buffer storing experience for training.
for each episode **do**
    Obtain initial state from environment s_0
    for step $t = 1, \ldots$ until end of episode **do**
        Pick an action according to $a_t \in \mathrm{argmax}_a \boxed{\{\tilde{\mu}(s_t, a) + \lambda \cdot \tilde{\sigma}(s_t, a)\}}$
        Receive state s_{t+1} and reward r_t from environment, having taken action a_t
        Add (s_t, a_t, r_t, s_{t+1}) to replay buffer B
        At learning interval, sample random minibatch and update $\{Q_k\}$ according to (12)
    end for
end for

mean and std across
ensemble of Q-functions

[Chen, Sidor, Abbeel, Schulman Q-UCB, 2017]

Algorithmic Exploration: Q-Ensembles: Q-UCB



[Chen, Sidor, Abbeel, Schulman Q-UCB, 2017]

Algorithmic Exploration: V-Ensemble + MPC: POLO

- Key idea: leverage value function ensemble to *plan* for *exploration*

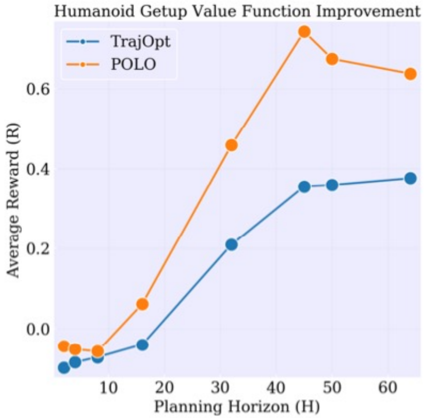
$$\hat{\pi}_{MPC}(s) := \arg\max_{\pi_{0:H-1}} \mathbb{E}\left[\sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H r_f(s_H)\right]$$

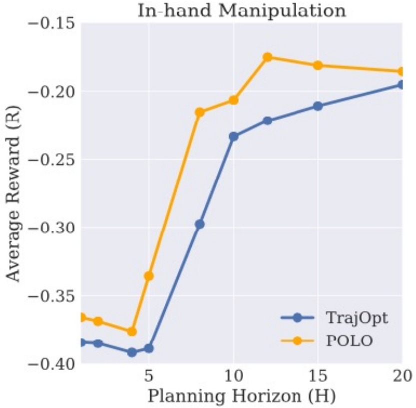
$$\boxed{\hat{V}(s) = \log\left(\sum_{k=1}^{K} \exp\left(\kappa\, \hat{V}_{\theta_k}(s)\right)\right)}$$

MPC with optimistic final state value through softmax over ensemble

[Lowrey*, Rajeswaran*, Kakade, Todorov, Mordatch, POLO, 2019]

Algorithmic Exploration: V-Ensemble + MPC: POLO





[Lowrey*, Rajeswaran*, Kakade, Todorov, Mordatch, POLO, 2019]

Algorithmic Exploration: Q-Ensembles: SUNRISE

Key Contributions beyond Q-UCB:

- Q: If we use Q-UCB, can the off-policy action selection affect learning stability?
- A: Yes. (see also recent work on stabilizing offline RL, investigating same)

- Q: If we have a Q-ensemble, can we leverage it to stabilize/improve the Q-learning update?
- A: Yes, and it can more generally mitigate noise propagation that happens in regular DQN/Q-learning (which is especially present when Q-learning is run off-policy*)

*Related: Chen, Schulman, Abbeel, Boltzmann Exploration Q-learning is secretly on-policy actor critic…

[Lee, Laskin, Srinivas, Abbeel, SUNRISE 2020]

Algorithmic Exploration: Q-Ensembles: SUNRISE

- Error propagation issue in Q-learning

unseen (s,a) → high error

$$Q(s_t, a_t) \leftarrow r_t + \gamma \max_a Q(s_{t+1}, a)$$

error propagates

- Reweightt each term in Bellman backup loss

$$w(s,a)\left(Q(s,a) - [r(s,a) + \gamma \widehat{Q}(s',a')]\right)^2$$

confidence score about target value based on variance of Q-ensemble

[Lee, Laskin, Srinivas, Abbeel, SUNRISE 2020]

Algorithmic Exploration: Q-Ensembles: SUNRISE

- Performance on DeepMind Control Suite at 100K and 500K environment steps

500K step	PlaNet [16]	Dreamer [17]	SLAC [31]	CURL [41]	DrQ [25]	RAD [30]	SUNRISE
Finger-spin	561 ± 284	796 ± 183	673 ± 92	926 ± 45	938 ± 103	975 ± 16	**983** ±1
Cartpole-swing	475 ± 71	762 ± 27	-	845 ± 45	868 ± 10	873 ± 3	**876** ± 4
Reacher-easy	210 ± 44	793 ± 164	-	929 ± 44	942 ± 71	916 ± 49	**982** ± 3
Cheetah-run	305 ± 131	570 ± 253	640 ± 19	518 ± 28	660 ± 96	624 ± 10	**678** ± 46
Walker-walk	351 ± 58	897 ± 49	842 ± 51	902 ± 43	921 ± 45	938 ± 9	**953** ± 13
Cup-catch	460 ± 380	879 ± 87	852 ± 71	959 ± 27	963 ± 9	966 ± 9	**969** ± 5
100K step							
Finger-spin	136 ± 216	341 ± 70	693 ± 141	767 ± 56	901 ± 104	811 ± 146	**905** ± 57
Cartpole-swing	297 ± 39	326 ± 27	-	582 ± 146	**759** ± 92	373 ± 90	591 ± 55
Reacher-easy	20 ± 50	314 ± 155	-	538 ± 233	601 ± 213	567 ± 54	**722** ± 50
Cheetah-run	138 ± 88	235 ± 137	319 ± 56	299 ± 48	344 ± 67	381 ± 79	**413** ± 35
Walker-walk	224 ± 48	277 ± 12	361 ± 73	403 ± 24	612 ± 164	641 ± 89	**667** ± 147
Cup-catch	0 ± 0	246 ± 174	512 ± 110	769 ± 43	**913**± 53	666 ± 181	633 ± 241

- SUNRISE consistently improves the performance of RAD

Algorithmic Exploration: Q-Ensembles: SUNRISE

- Performance on Atari games at 100K interactions

Game	Human	Random	SimPLe [23]	CURL [41]	Rainbow [47]	SUNRISE
Alien	7127.7	227.8	616.9	558.2	789.0	**872.0**
Amidar	1719.5	5.8	88.0	**142.1**	118.5	122.6
Assault	742.0	222.4	527.2	**600.6**	413.0	594.8
Asterix	8503.3	210.0	**1128.3**	734.5	533.3	755.0
BankHeist	753.1	14.2	34.2	131.6	97.7	**266.7**
BattleZone	37187.5	2360.0	5184.4	14870.0	7833.3	**15700.0**
Boxing	12.1	0.1	**9.1**	1.2	0.6	6.7
Breakout	30.5	1.7	**16.4**	4.9	2.3	1.8
ChopperCommand	7387.8	811.0	**1246.9**	1058.5	590.0	1040.0
CrazyClimber	35829.4	10780.5	**62583.6**	12146.5	25426.7	22230.0
DemonAttack	1971.0	152.1	208.1	817.6	688.2	**919.8**
Freeway	29.6	0.0	20.3	26.7	28.7	**30.2**
Frostbite	4334.7	65.2	254.7	1181.3	1478.3	**2026.7**
Gopher	2412.5	257.6	771.0	**669.3**	348.7	654.7
Hero	30826.4	1027.0	2656.6	6279.3	3675.7	**8072.5**
Jamesbond	302.8	29.0	125.3	**471.0**	300.0	390.0
Kangaroo	3035.0	52.0	323.1	872.5	1060.0	**2000.0**
Krull	2665.5	1598.0	**4539.9**	4229.6	2592.1	3087.2
KungFuMaster	22736.3	258.5	**17257.2**	14307.8	8600.0	10306.7
MsPacman	6951.6	307.3	1480.0	1465.5	1118.7	**1482.3**
Pong	14.6	-20.7	**12.8**	-16.5	-19.0	-19.3
PrivateEye	69571.3	24.9	58.3	**218.4**	97.8	100.0
Qbert	13455.0	163.9	1288.8	1042.4	646.7	**1830.8**
RoadRunner	7845.0	11.5	5640.6	5661.0	9923.3	**11913.3**
Seaquest	42054.7	68.4	**683.3**	384.5	396.0	570.7
UpNDown	11693.2	533.4	3350.3	2955.2	3816.0	**5074.0**

Consistently outperform Rainbow

SOTA on 13 out of 26 environments

Outline

- Problem Motivation
- Baseline RL Algorithms Refresher
- Intrinsic Rewards for Reward-Free Pre-Training and Exploration
- Algorithmic Approaches to Exploration (can complement intrinsic reward RFPT!)
- ***Algorithmic Approaches to Reward-Free Pre-Training***

Algorithmic RFPT: Frontier Approaches

- Main idea:
  - keep track of frontier of where the AI Agent has been
  - then set goals near this frontier and/or explicitly revisit past state on frontier and randomly explore from there

Algorithmic RFPT: Frontier: HER

- Train policy $\pi(a \mid s, g)$
- Key idea:
  - To improve learning signal / alleviate exploration needs: Hindsight-relabel 50% of goals g in the replay buffer to match the final achieved state in the corresponding trajectory
  - Leads to natural expansion of goals that can be achieved

[Andrychowicz et al, 2017: Hindsight Experience Replay]
[Schaul et al, 2015: Universal Value Function Approximators]
[Kaelbling 1993: Learning to Achieve Goals]

Algorithmic RFPT: Frontier: HER

- Train policy $\pi(a \mid s, g)$
- Key idea:
  - To improve learning signal / alleviate exploration needs:
    Hindsight-relabel 50% of goals g in the replay buffer to match the final achieved state in the corresponding trajectory
  - Leads to natural expansion of goals that can be achieved

[Andrychowicz et al, 2017: Hindsight Experience Replay]
[Schaul et al, 2015: Universal Value Function Approximators]
[Kaelbling 1993: Learning to Achieve Goals]

Quantitative Evaluation

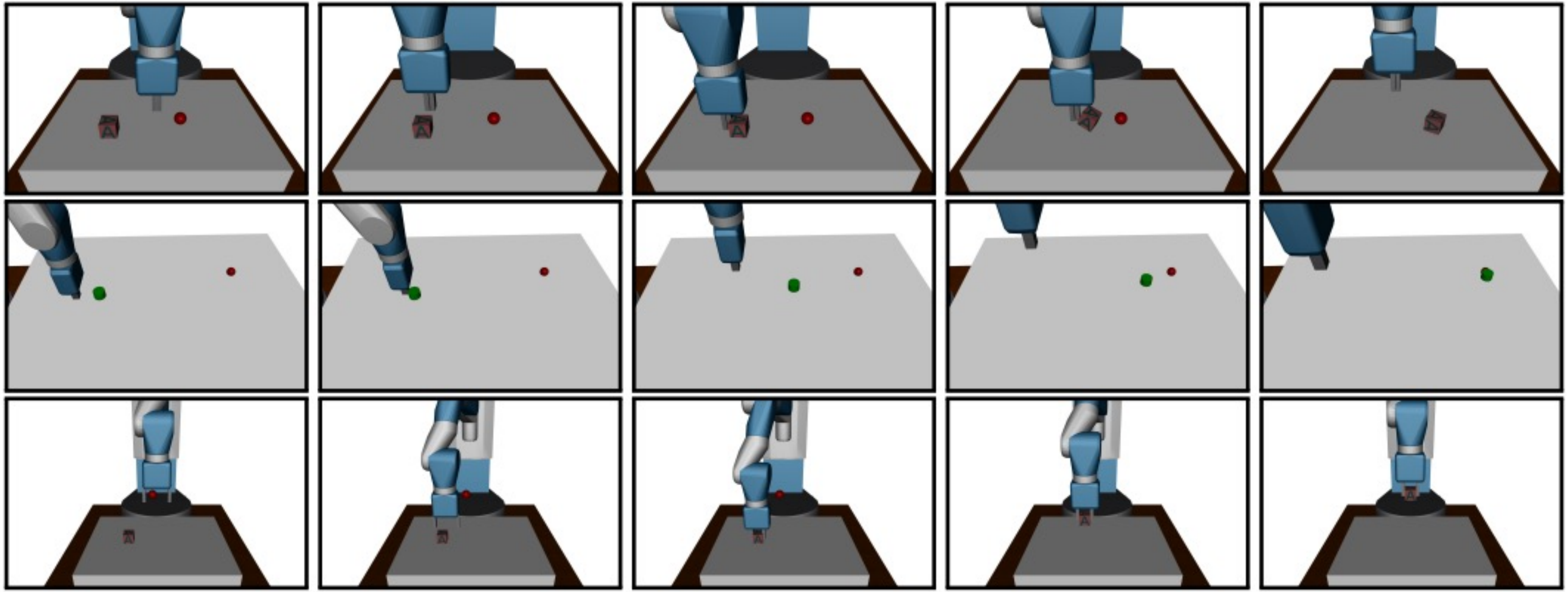
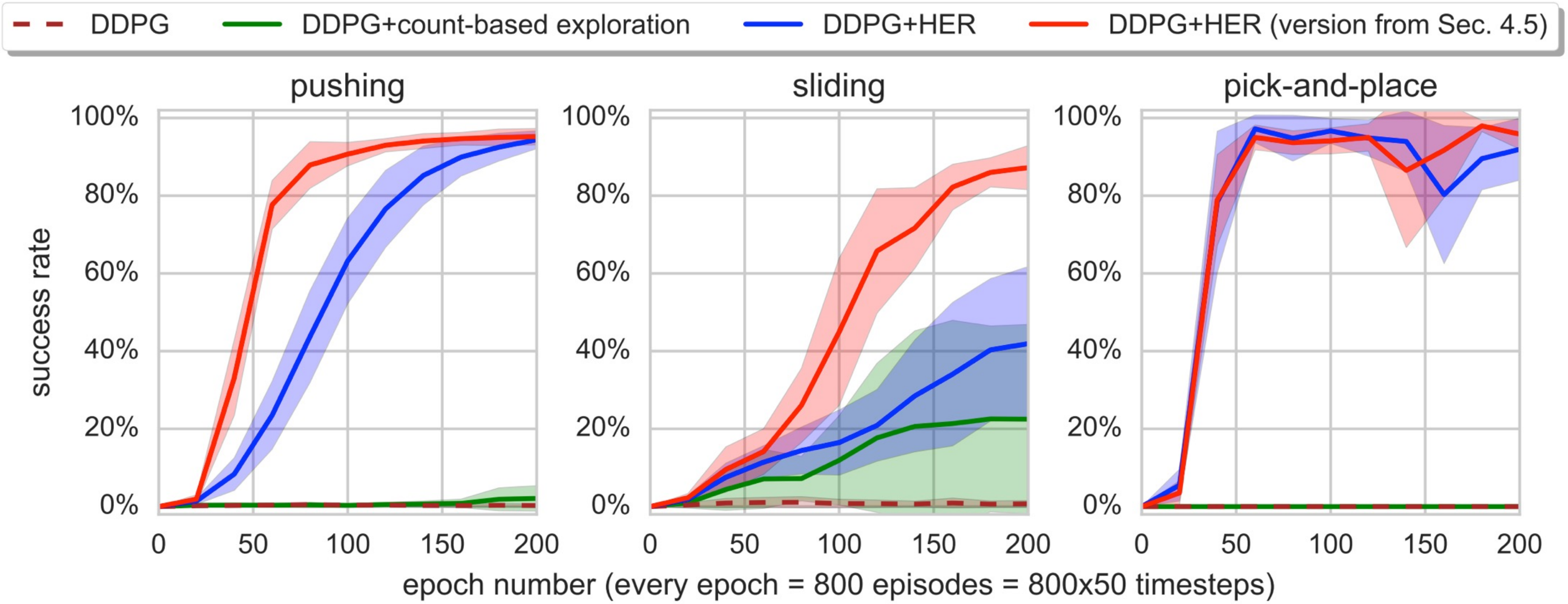


Figure 2: Different tasks: *pushing* (top row), *sliding* (middle row) and *pick-and-place* (bottom row). The red ball denotes the goal position.

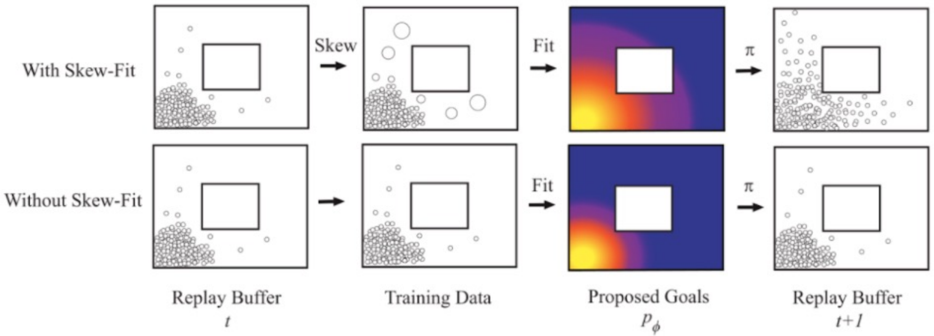
Quantitative Evaluation



~3s/episode → 1.5 days for 50 epochs, 6 days for 200 epochs

Algorithmic RFPT: Frontier: RIG / SkewFit

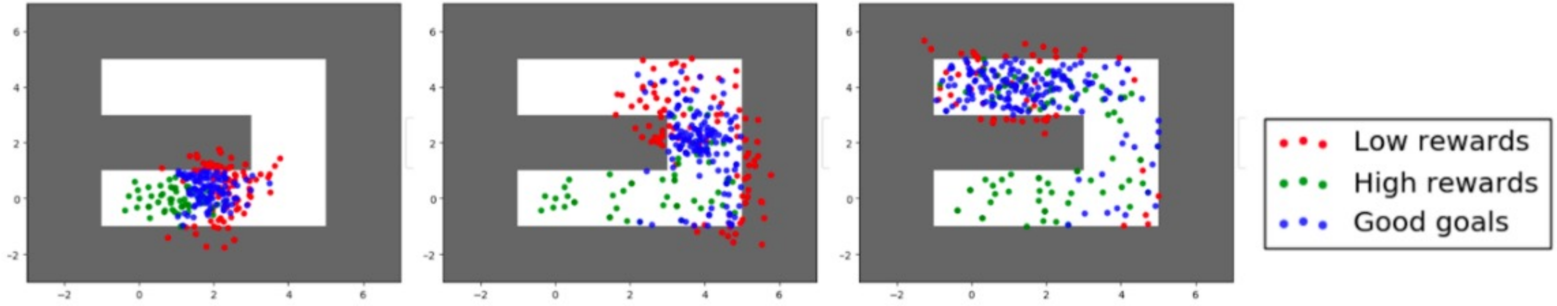
- HER assumes access to underlying state
- RIG adapts HER to Image Inputs
  - operates in latent space of a VAE
  - can use latent space for new goal setting
- SkewFit shifts RIG / HER goal sampling



[RIG: Nair, Pong, Dalal, Bahl, Lin, Levine, 2018] [Skew-Fit: Pong*, Dalal*, Lin*, Nair, Bahl, Levine, 2019]

Algorithmic RFPT: Frontier: GoalGAN

- Key ideas:
  - Train goal-conditioned policy / Q-function (~HER)
  - Sample Goals of Intermediate Difficulty – goal generator = GAN

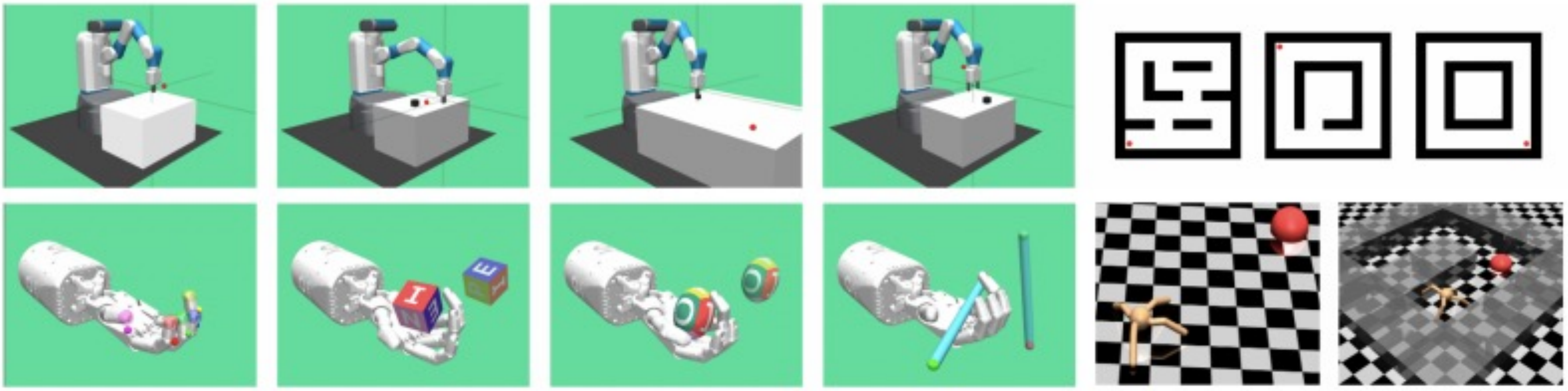


(a) Iteration 5 (b) Iteration 90 (c) Iterartion 350

[GoalGAN: Florensa*, Held*, Geng*, Abbeel, 2017: Automatic Goal Generation for RL Agents]
see also: [CURIOUS: Colas, Fournier, Sigaud, Chetouani, Oudeyer, 2019] – considers learning progress (vs. GoalGAN considers learning status)

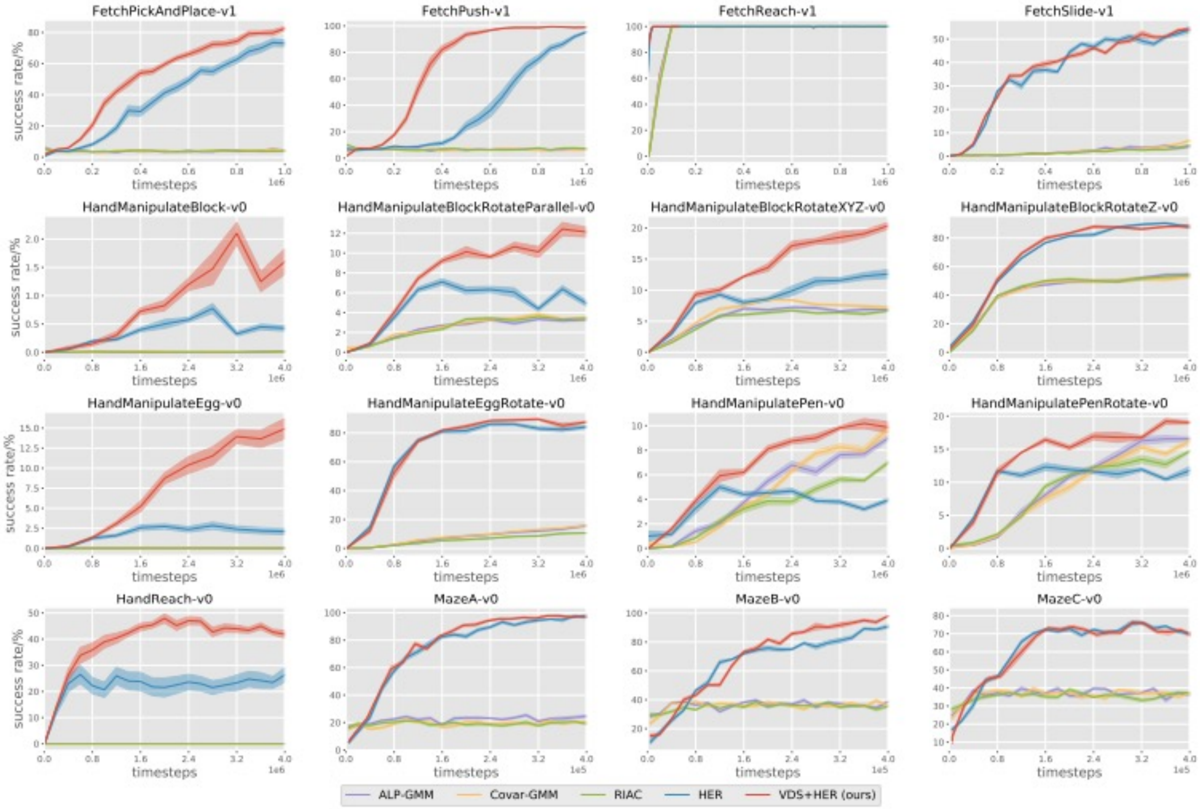
Algorithmic RFPT: Frontier: Value Disagreement

- Key ideas:
  - Train goal-conditioned policy / Q-function (~HER)
  - Sample Goals based on Q-Value Disagreement



[Zhang, Abbeel, Pinto, 2020: Automatic Curriculum Learning through Value Disagreement]

Algorithmic RFPT: Frontier: Value Disagreement



[Zhang, Abbeel, Pinto, 2020: Automatic Curriculum Learning through Value Disagreement]

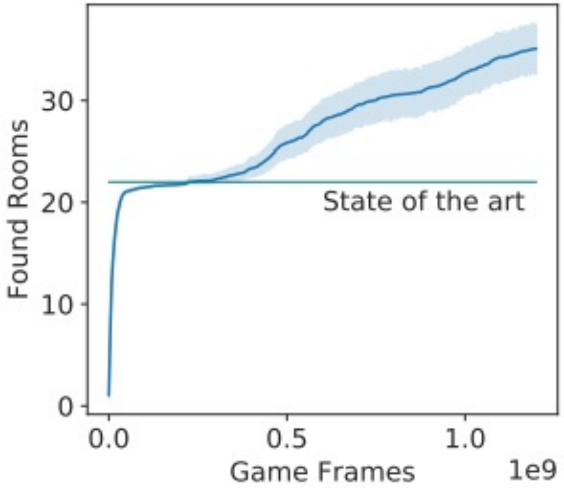
Algorithmic RFPT: Frontier: GoExplore

- Key idea:
  - When visiting a state with low visitation count, specifically go to that state again in future roll-outs + randomly explore around that state, which will likely yield new such exploration goal states
  - Original paper: assume ability to deterministically return or access to resets; later versions train a goal conditioned policy
  - Key assumption: reasonable way to divide state space into cells, and not too many cells to be able to explore them all --- done by low-res image in Atari
- Main result: breakthrough/"solve" Atari hard-exploration games

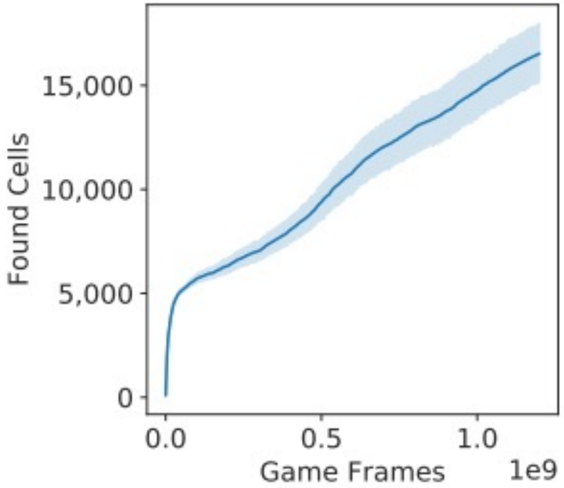
[Go-Explore: Ecoffet, Huizinga, Lehman, Stanley, Clune, 2019; also: First Return, Then Explore: Ecoffet, H, L, S, C, 2020]

Algorithmic RFPT: Frontier: GoExplore

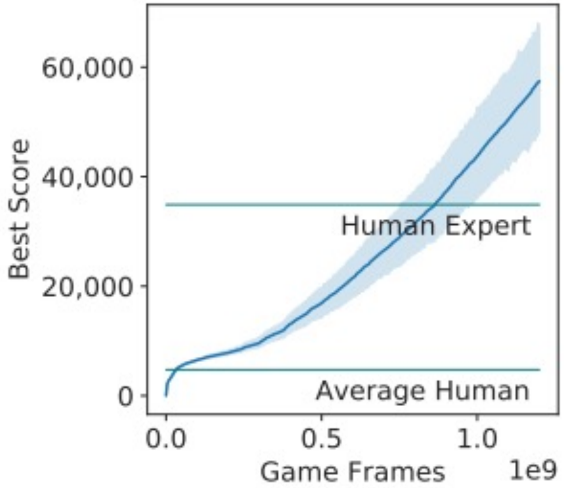
Montezuma's Revenge Atari Game Performance



(a) Number of rooms found



(b) Number of cells found

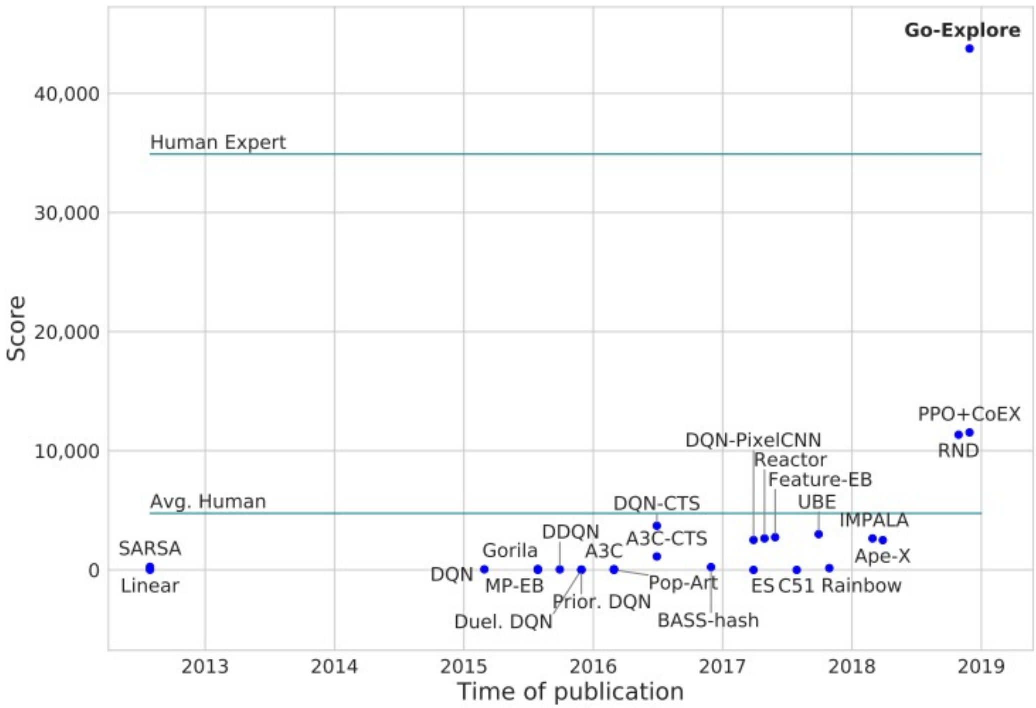


(c) Maximum score in archive

[Go-Explore: Ecoffet, Huizinga, Lehman, Stanley, Clune, 2019; also: First Return, Then Explore: Ecoffet, H, L, S, C, 2020]

Algorithmic RFPT: Frontier: GoExplore

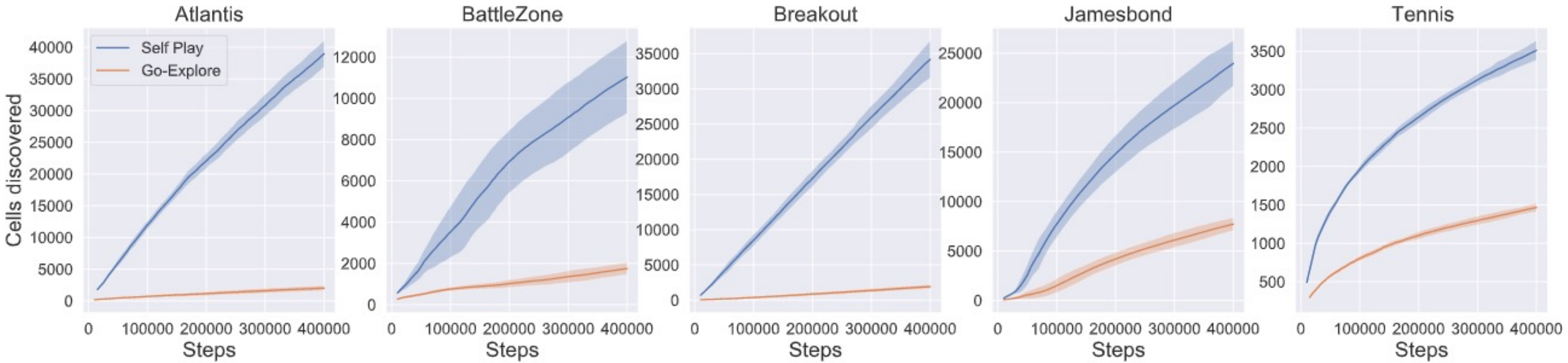
Montezuma's Revenge Atari Game Performance



[Go-Explore: Ecoffet, Huizinga, Lehman, Stanley, Clune, 2019; also: First Return, Then Explore: Ecoffet, H, L, S, C, 2020]

Algorithmic RFPT: Frontier: SelfPlayer

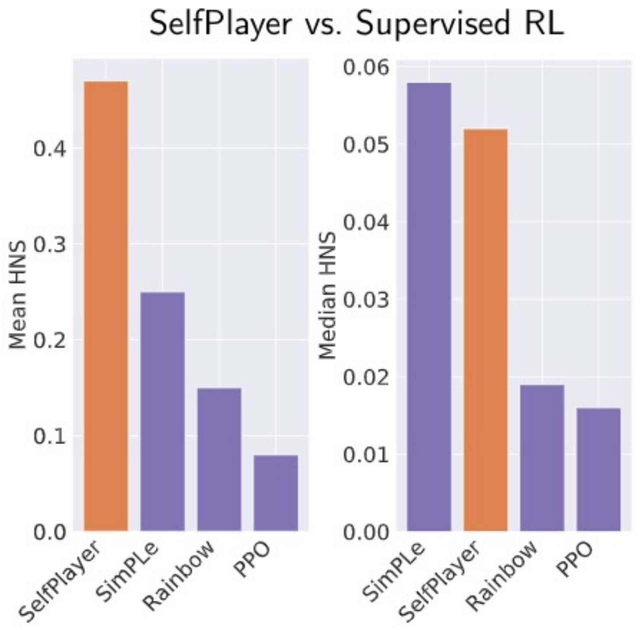
- Brings together ideas from ASP and GoExplore

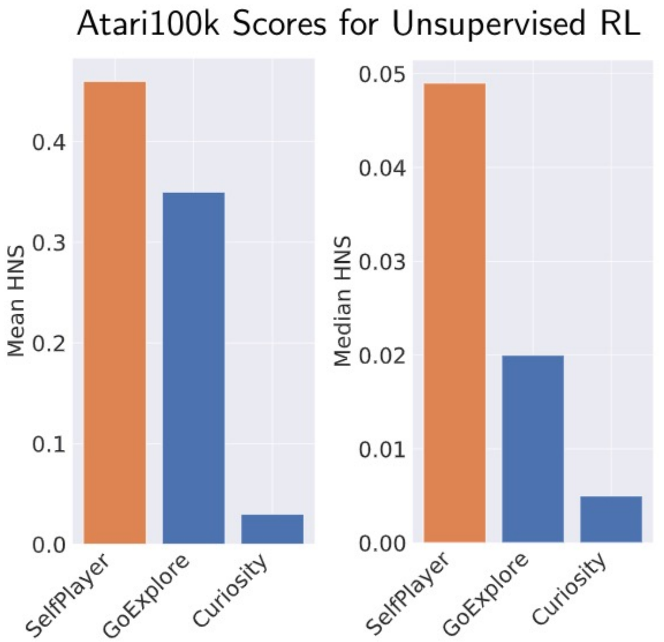


[SelfPlayer: Laskin, Rudes, Cang, Abbeel, 2021]

Algorithmic RFPT: Frontier: SelfPlayer

- Brings together ideas from ASP and GoExplore





[SelfPlayer: Laskin, Rudes, Cang, Abbeel, 2021]

Algorithmic RFPT: Frontier: SelfPlayer

- Brings together ideas from ASP and GoExplore

Env	SelfPlayer	GoExplore	Curiosity	Random	Human
Alien	**551.41** ± 41.8	**550.32** ± 54.5	217.0	227.8	7127.7
Amidar	**270.84** ± 17.9	166.92 ± 11.9	18.1	5.8	1719.5
Assault	**305.88** ± 17.0	180.67 ± 14.6	214.0	222.4	742.0
Asterix	743.43 ± 315.0	**1007.14** ± 47.8	223.0	210.0	8503.3
Asteroids	718.69 ± 101.5	217.78 ± 14.0	**953.0**	719.1	47388.7
Atlantis	6990.91 ± 1011.0	1220.63 ± 195.3	**17400.0**	12850.0	29028.1
BankHeist	**68.99** ± 5.8	26.19 ± 3.9	12.9	14.2	753.1
BattleZone	**10679.01** ± 604.4	6412.7 ± 636.6	2560.0	2360.0	37187.5
BeamRider	294.26 ± 14.8	285.08 ± 27.1	311.0	**363.9**	16926.5
Bowling	**40.53** ± 3.0	20.22 ± 1.2	20.2	23.1	160.7
Breakout	**87.31** ± 39.2	9.9 ± 0.8	3.3	1.7	30.5
Centipede	**4024.76** ± 373.1	3308.33 ± 576.1	2100.0	2090.9	12017.0
Chopper	**850.0** ± 65.1	719.05 ± 51.8	597.0	**811.0**	7387.8
CrazyClimber	3574.75 ± 337.0	1326.98 ± 59.4	9690.0	**10780.5**	35829.4
DemonAttack	**455.33** ± 33.0	371.9 ± 26.2	161.0	152.1	1971.0
DoubleDunk	-23.64 ± 0.3	-15.87 ± 0.7	-18.0	-18.6	-16.4
Enduro	0.38 ± 0.1	**0.46** ± 0.2	0.0	0.0	860.5
FishingDerby	-88.62 ± 0.5	-73.65 ± 1.9	-92.3	-91.7	-38.7
Freeway	4.77 ± 0.2	**5.16** ± 0.3	0.73	0.0	29.6
Frostbite	**403.09** ± 59.0	86.03 ± 8.9	149.0	65.2	4334.7
Gopher	142.96 ± 14.3	53.97 ± 7.8	**397.0**	257.6	2412.5
Gravitar	**276.54** ± 22.0	103.97 ± 25.4	251.0	173.0	3351.4
IceHockey	-15.02 ± 0.3	-14.02 ± 0.7	-13.1	-11.2	0.9
Jamesbond	**5483.89** ± 148.4	4663.49 ± 155.2	38.2	29.0	302.8
Kangaroo	**866.67** ± 127.5	**892.06** ± 234.8	59.3	52.0	3035.0
Krull	83.52 ± 5.0	37.56 ± 11.2	814.0	**1598.0**	2665.5
KungFuMaster	77.78 ± 33.9	17.46 ± 6.5	**400.0**	258.5	22736.3
Montezuma	**1369.14** ± 140.4	26.98 ± 6.3	0.0	0.0	4753.3
MsPacman	**1381.73** ± 79.9	1071.11 ± 78.5	246.0	307.3	6951.6
NameThisGame	1274.32 ± 64.2	1120.63 ± 62.9	2070.0	**2292.3**	8049.0
Pitfall	-446.57 ± 107.3	-430.62 ± 44.3	-39.1	-229.4	6463.7
Pong	-20.21 ± 0.2	-11.71 ± 0.7	-19.1	-20.7	14.6
PrivateEye	**2694.78** ± 764.7	**2565.24** ± 893.4	-608.0	24.9	69571.3
Qbert	**854.01** ± 168.7	687.7 ± 47.7	334.0	163.9	13455.0
Riverraid	**2149.63** ± 203.9	1016.98 ± 54.7	1490.0	1338.5	17118.0
RoadRunner	**2271.6** ± 312.5	1103.17 ± 239.7	60.0	11.5	7845.0
Robotank	**4.61** ± 0.3	3.62 ± 0.3	2.67	2.2	11.9
Seaquest	**252.89** ± 26.3	179.52 ± 45.1	237.0	68.4	42054.7
SpaceInvaders	**330.06** ± 29.1	304.13 ± 29.2	191.0	148.0	1668.7
StarGunner	146.67 ± 7.6	19.05 ± 5.7	533.0	**664.0**	10250.0
Tennis	-23.08 ± 0.1	-15.79 ± 0.5	-23.5	-23.8	-8.3
TimePilot	1122.22 ± 271.7	120.63 ± 17.5	2520.0	**3568.0**	5229.2
Tutankham	**16.2** ± 2.0	5.94 ± 1.1	7.41	11.4	167.6
UpNDown	1213.11 ± 114.5	1176.83 ± 107.3	**1510.0**	533.4	11693.2
Venture	23.33 ± 15.8	15.87 ± 8.9	**83.3**	0.0	1187.5
VideoPinball	10555.21 ± 1434.2	10454.87 ± 1549.6	**35300.0**	0.0	17667.9
WizardOfWor	**7813.33** ± 1012.8	1206.35 ± 66.1	817.0	563.5	4756.5
Zaxxon	**391.11** ± 41.4	184.13 ± 34.8	1.68	32.5	9173.3
# Envs Best	**26**	6	7	7	-
Average HNS	**.47**	.35	0.03	-	-
Median HNS	**.05**	.02	0.01	-	-

[SelfPlayer: Laskin, Rudes, Cang, Abbeel, 2021]

Summary

- Problem Motivation
- Baseline RL Algorithms Refresher
- Intrinsic Rewards for Reward-Free Pre-Training and Exploration
- Algorithmic Approaches to Exploration (can complement intrinsic reward RFPT!)
- Algorithmic Approaches to Reward-Free Pre-Training

Many Research Opportunities

- Clearer evaluations / comparisons
- Right combination of components?
- Simpler / more robust versions → supplant existing simple SAC/TD3/PPO/DQN baselines
- Open-ended environments are the frontier; but even regular vision-based environments still challenging
- Better fine-tuning/adaptation
- Can it lead to more robust vision
- Showcase truly unexpected solutions (e.g. circuit design / ..)

Some Additional Perspective

Intrinsically Motivated Goal-Conditioned Reinforcement Learning: a Short Survey

Cédric Colas CEDRIC.COLAS@INRIA.FR
INRIA and Univ. de Bordeaux; Bordeaux (FR)

Tristan Karch TRISTAN.KARCH@INRIA.FR
INRIA and Univ. de Bordeaux; Bordeaux (FR)

Olivier Sigaud OLIVIER.SIGAUD@UPMC.FR
Sorbonne Université; Paris (FR)

Pierre-Yves Oudeyer PIERRE-YVES.OUDEYER@INRIA.FR
INRIA; Bordeaux (FR) and ENSTA Paris Tech; Paris (FR)

Thank you!

pabbeel@cs.berkeley.edu