Unsupervised State Representation Learning in Atari

Ankesh Anand[*]
Mila, Université de Montréal
Microsoft Research

Evan Racah[*]
Mila, Université de Montréal

Sherjil Ozair[*]
Mila, Université de Montréal

Yoshua Bengio
Mila, Université de Montréal

Marc-Alexandre Côté
Microsoft Research

R Devon Hjelm
Microsoft Research
Mila, Université de Montréal

Abstract

State representation learning, or the ability to capture latent generative factors of an environment, is crucial for building intelligent agents that can perform a wide variety of tasks. Learning such representations without supervision from rewards is a challenging open problem. We introduce a method that learns state representations by maximizing mutual information across spatially and temporally distinct features of a neural encoder of the observations. We also introduce a new benchmark based on Atari 2600 games where we evaluate representations based on how well they capture the ground truth state variables. We believe this new framework for evaluating representation learning models will be crucial for future representation learning research. Finally, we compare our technique with other state-of-the-art generative and contrastive representation learning methods.

1 Introduction

The ability to perceive and represent visual sensory data into useful and concise descriptions is considered a fundamental cognitive capability in humans [1, 2], and thus crucial for building intelligent agents [3]. Representations that succinctly reflect the true state of the environment should allow agents to learn to act in those environments with fewer interactions, and effectively transfer knowledge across different tasks in the environment.

Recently, deep representation learning has led to tremendous progress in a variety of machine learning problems across numerous domains [4, 5, 6, 7, 8]. Typically, such representations are often learned via end-to-end learning using the signal from labels or rewards, which makes such techniques often very sample-inefficient. In contrast, human learning in the natural world appears to require little to no explicit supervision for perception [9].

Unsupervised [10, 11, 12] and self-supervised representation learning [13, 14, 15] have emerged as an alternative to supervised versions which can yield useful representations with reduced sample complexity. In the context of learning state representations [16], current unsupervised methods rely on generative decoding of the data using either VAEs [17, 18, 19, 20] or prediction in pixel-space [21, 22]. Since these objectives are based on reconstruction error in the pixel space, they are not incentivized to capture abstract latent factors and often default to capturing pixel level details.

In this work, we leverage recent advances in self-supervision that rely on scalable estimation of mutual information [23, 24, 25, 26], and propose a new contrastive state representation learning method named Spatiotemporal DeepInfomax (ST-DIM), which maximizes the mutual information across both the spatial and temporal axes.

[*] Equal contribution. {anandank, racaheva, ozairs}@mila.quebec

Preprint. Under review.

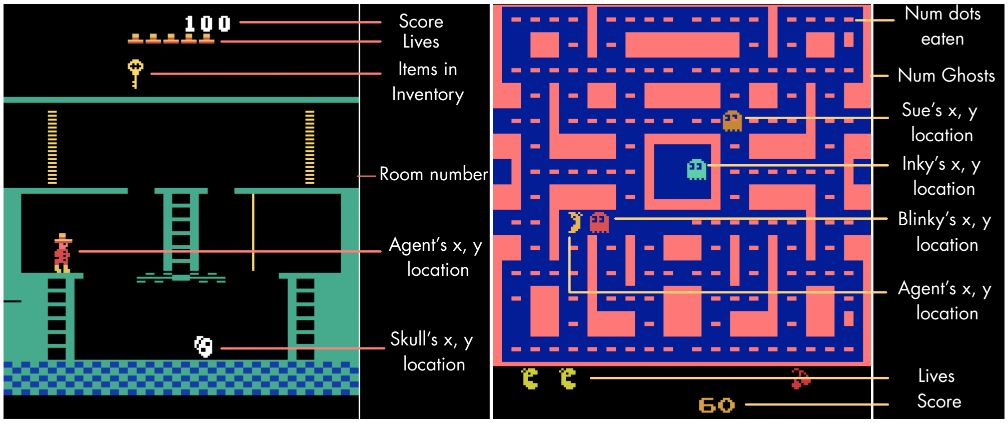


Figure 1: We use a collection of 22 Atari 2600 games to evaluate state representations. We leveraged the source code of the games to annotate the RAM states with important state variables such as the location of various objects in the game. We compare various unsupervised representation learning techniques based on how well the representations linearly-separate the state variables. Shown above are examples of state variables annotated for Montezuma's Revenge and MsPacman.

To systematically evaluate the ability of different representation learning methods at capturing the true underlying factors of variation, we propose a benchmark based on Atari 2600 games using the Arcade Learning Environment [ALE, 27]. A simulated environment provides access to the underlying generative factors of the data, which we extract using the source code of the games. These factors include variables such as the location of the player character, location of various items of interest (keys, doors, etc.), and various non-player characters, such as enemies (see figure 1). Performance of a representation learning technique in the Atari representation learning benchmark is then evaluated using *linear probing* [28], i.e. the accuracy of linear classifiers trained to predict the latent generative factors from the learned representations.

Our contributions are the following

1. We propose a new self-supervised state representation learning technique which exploits the spatial-temporal nature of visual observations in a reinforcement learning setting.
2. We propose a new state representation learning benchmark using 22 Atari 2600 games based on the Arcade Learning Environment (ALE).
3. We conduct extensive evaluations of existing representation learning techniques on the proposed benchmark and compare with our proposed method.

2 Related Work

Unsupervised representation learning via mutual information estimation: Recent works in unsupervised representation learning have focused on extracting latent representations by maximizing a lower bound on the mutual information between the representation and the input. Belghazi et al. [23] estimate the mutual information with neural networks using the Donsker-Varadhan representation of the KL divergence [29], while Chen et al. [30] use the variational bound from Barber and Agakov [31] to learn discrete latent representations. Hjelm et al. [25] learn representations by maximizing the Jensen-Shannon divergence between joint and product of marginals of an image and its patches. van den Oord et al. [24] maximize mutual information using a multi-sample version of noise contrastive estimation [32, 33]. See [34] for a review of different variational bounds for mutual information.

State representation learning: Learning better state representations is an active area of research within robotics and reinforcement learning. Jonschkowski and Brock [35] and Jonschkowski et al. [36] propose to learn representations using a set of handcrafted robotic priors. Several prior works use a VAE and its variations to learn a mapping from observations to state representations [37, 17, 38].

Thomas et al. [39] aims to learn the representations that maximize the causal relationship between the distributed policies and the representation of changes in the state. Recently, Cuccu et al. [40] shows that visual processing and policy learning can be effectively decoupled in Atari games. Nachum et al. [41] connects mutual information estimators to representation learning in hierarchical RL. Our work is also closely related to recent work in learning object-oriented representations [42].

Evaluation frameworks of representations: Evaluating representations is an open problem, and doing so is usually domain specific. In vision tasks, it is common to evaluate based on the presence of linearly separable label-relevant information, either in the domain the representation was learned on [43] or in transfer learning tasks [44, 45]. In NLP, the SentEval [46] and GLUE [47] benchmarks provide a means of providing a more linguistic-specific understanding of what the model has learned, and these have become a standard tool in NLP research. Our evaluation framework can be thought of as a GLUE-like benchmarking tool for RL, providing a fine-grained understanding of how well the RL agent perceives the objects in the scene. Analogous to GLUE in NLP, we anticipate that our benchmarking tool will be useful in RL research for better designing components of agent learning.

3 Spatiotemporal Deep Infomax

We assume a setting where an agent interacts with an environment and observes a set of high-dimensional observations $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ across several episodes. Our goal is to learn an abstract representation of the observation that captures the underlying latent generative factors of the environment.

This representations should focus on high-level semantics (e.g., the concept of agents, enemies, objects, score, etc.) and ignore the low-level details such as the precise texture of the background, which warrants a departure from the class of methods that rely on a generative decoding of the full observation. Prior work in neuroscience [48, 49] has suggested that the brain maximizes *predictive information* [50] at an abstract level to avoid sensory overload. Predictive information, or the mutual information between consecutive states, has also been shown to be the organizing principle of retinal ganglion cells in salamander brains [51]. Thus our representation learning approach relies on maximizing an estimate based on a lower bound on the mutual information over consecutive observations x_t and x_{t+1}.

3.1 Maximizing mutual information across space and time

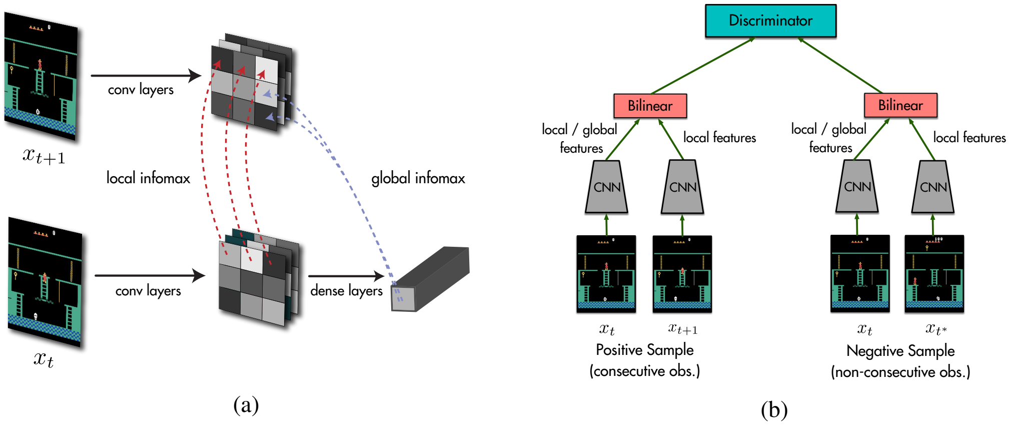


Figure 2: A schematic overview of SpatioTemporal DeepInfoMax (ST-DIM). (a) shows the two different mutual information objectives: local infomax and global infomax. (b) shows a simplified version of the contrastive task we use to estimate mutual information. In practice, we use multiple negative samples.

Given a mutual information estimator, we follow DIM [25] and maximize a sum of patch-level mutual information objectives. The global objectives maximize the mutual information between the full observation at time t with small patches of the observation at time $t + 1$. The representations of the small image patches are taken to be the hidden activations of the convolutional encoder applied to the full observation. The layer is picked appropriately to ensure that the hidden activations only have a limited receptive field corresponding to $1/16^{th}$ the size of the full observations. The local objective maximizes the mutual information between the local feature at time t with the corresponding local feature at time $t+1$. Figure 2 is a visual depiction of our model which we call Spatiotemporal Deep Infomax (ST-DIM).

It has been shown that mutual information bounds can be loose for large values of the mutual information [52] and in practice fail to capture all the relevant features in the data [53] when used to learn representations. To alleviate this issue, our approach constructs multiple small mutual information objectives (rather than a single large one) which are easier to estimate via lower bounds, which has been concurrently found to work well in the context of semi-supervised learning [54].

For the mutual information estimator, we use infoNCE [24], a multi-sample variant of noise-contrastive estimation [32] that was also shown to work well with DIM. Let $\{(x_i, y_i)\}_{i=1}^N$ be a paired dataset of N samples from some joint distribution $p(x, y)$. For any index i, (x_i, y_i) is a sample from the joint $p(x, y)$ which we refer to as *positive examples*, and for any $i \neq j$, (x_i, y_j) is a sample from the product of marginals $p(x)p(y)$[2], which we refer to as *negative examples*. The InfoNCE objective learns a score function $f(x, y)$ which assigns large values to positive examples and small values to negative examples by maximizing the following bound [see 24, 34, for more details on this bound],

$$\mathcal{I}_{NCE}(\{(x_i, y_i)\}_{i=1}^N) = \sum_{i=1}^N \log \frac{\exp f(x_i, y_i)}{\sum_{j=1}^N \exp f(x_i, y_j)} \tag{1}$$

The above objective has also been referred to as *multi-class n-pair loss* [55, 56] and *ranking-based NCE* [33], and is similar to MINE [23] and the JSD-variant of DIM [25].

Following van den Oord et al. [24] we use a bilinear model for the score function $f(x, y) = \phi(x)^T W \phi(y)$, where ϕ is the representation encoder. The bilinear model combined with the InfoNCE objective forces the encoder to learn *linearly predictable* representations, which we believe helps in learning representations at the semantic level. In our context, the positive examples correspond to pairs of consecutive observations (x_t, x_{t+1}) and negative samples correspond to pair to pair of non-consecutive observations (x_t, x_{t^*}), where t^* is a randomly sampled time index from the episode. For ST-DIM, the final score function for the global objective is $f_g(x_t, x_{t+1}) = \phi(x_t)^T W_g \phi_{l,m,n}(x_{t+1})$ and the score function of the local objective is $f_l(x_t, x_{t+1}) = \phi_{l,m,n}(x_t)^T W_l \phi_{l,m,n}(x_{t+1})$, where $\phi_{l,m,n}$ is the feature map at the l^{th} layer at the (m, n) spatial location.

4 The Atari Annotated RAM Interface (AARI)

Measuring the usefulness of a representation is still an open problem, as a core utility of representations is their use as feature extractors in tasks that are different from those used for training (e.g., *transfer learning*). Measuring classification performance, for example, may only reveal the amount of class-relevant information in a representation, but may not reveal other information useful for segmentation. It would be useful, then, to have a more *general* set of measures on the usefulness of a representation, such as ones that may indicate more general utility across numerous real-world tasks. In this vein, we assert that in the context of dynamic, visual, interactive environments, the capability of a representation to capture the underlying high-level factors of the state of an environment will be generally useful for a variety of downstream tasks such as prediction, control, and tracking.

We find video games to be a useful candidate for evaluating visual representation learning algorithms primarily because they are spatiotemporal in nature, which is (1) more realistic compared to static i.i.d. datasets and (2) prior work [57, 58] have argued that without temporal structure, recovering the true underlying latent factors is undecidable. Apart from this, video games also provide ready access to the underlying ground truth states, unlike real-world datasets, which we need to evaluate performance of different techniques.

Annotating Atari RAM: ALE does not explicitly expose any ground truth state information. However, ALE does expose the RAM state (128 bytes per timestep) which are used by the game

[2] For convenience, ignoring those that are in the support of the joint.

Table 1: Number of ground truth labels available in the benchmark for each game across each category. Localization is shortened for local. See section 4 for descriptions and examples for each category.

		SMALL		SCORE/CLOCK		
	AGENT	OBJECT	OTHER	LIVES		
GAME	LOCAL.	LOCAL.	LOCAL.	DISPLAY	MISC	OVERALL
ASTEROIDS	2	4	30	3	3	41
BERZERK	2	4	19	4	5	34
BOWLING	2	2	0	2	10	16
BOXING	2	0	2	3	0	7
BREAKOUT	1	2	0	1	31	35
DEMONATTACK	1	1	6	1	1	10
FREEWAY	1	0	10	1	0	12
FROSTBITE	2	0	9	4	2	17
HERO	2	0	0	3	3	8
MONTEZUMAREVENGE	2	0	4	4	5	15
MSPACMAN	2	0	10	2	3	17
PITFALL	2	0	3	0	0	5
PONG	1	2	1	2	0	6
PRIVATEEYE	2	0	2	4	2	10
QBERT	3	0	2	0	0	5
RIVERRAID	1	2	0	2	0	5
SEAQUEST	2	1	8	4	3	18
SPACEINVADERS	1	1	2	2	1	7
TENNIS	2	2	2	2	0	8
VENTURE	2	0	12	3	1	18
VIDEOPINBALL	2	2	0	2	0	6
YARSREVENGE	2	4	2	0	0	8
TOTAL	39	27	124	49	70	308

programmer to store important state information such as the location of sprites, the state of the clock, or the current room the agent is in. To extract these variables, we consulted commented disassemblies [59] (or source code) of Atari 2600 games which were made available by Engelhardt [60] and Jentzsch and CPUWIZ [61]. We were able to find and verify important state variables for a total of 22 games. Once this information is acquired, combining it with the ALE interface produces a wrapper that can automatically output a state label for every example frame generated from the game. We make this available with an easy-to-use *gym* wrapper, which returns this information with no change to existing code using *gym* interfaces. Table 1 lists the 22 games along with the categories of variables for each game. We describe the meaning of each category in the next section.

State variable categories: We categorize the state variables of all the games among six major categories: agent localization, small object localization, other localization, score/clock/lives/display, and miscellaneous. **Agent Loc.** (agent localization) refers to state variables that represent the x or y coordinates on the screen of any sprite controllable by actions. **Small Loc.** (small object localization) variables refer to the x or y screen position of small objects, like balls or missiles. Prominent examples include the ball in Breakout and Pong, and the torpedo in Seaquest. **Other Loc.** (other localization) denotes the x or y location of any other sprites, including enemies or large objects to pick up. For example, the location of ghosts in Ms Pacman or the ice floes in Frostbite. **Score/Clock/Lives/Display** refers to variables that track the score of the game, the clock, or the number of remaining lives the agent has, or some other display variable, like the oxygen meter in Seaquest. **Misc.** (Miscellaneous) consists of state variables that are largely specific to a game, and don't fall within one of the above mentioned categories. Examples include the existence of each block or pin in Breakout and Bowling, the room number in Montezuma's Revenge, or Ms. Pacman's facing direction.

Probing: Evaluating representation learning methods is a challenging open problem. The notion of *disentanglement* [62, 63] has emerged as a way to measure the usefulness of a representation [64, 37]. In this work, we focus only on *explicitness*, i.e the degree to which underlying generative factors can be recovered using a *linear* transformation from the learned representation. This is standard methodology in the self-supervised representation learning literature [14, 24, 65, 15, 25]. Specifically, to evaluate a representation we train linear classifiers predicting each state variable, and we report the mean F1 score.

5 Experimental Setup

We evaluate the performance of different representation learning methods on our benchmark. Our experimental pipeline consists of first training an encoder, then freezing its weights and evaluating its performance on linear probing tasks. For each identified generative factor in each game, we construct a linear probing task where the representation is trained to predict the ground truth value of that factor. Note that the gradients are not backpropagated through the encoder network, and only used to train the linear classifier on top of the representation.

5.1 Data preprocessing and acquisition

We consider two different modes for collecting the data: (1) using a random agent (steps through the environment by selecting actions randomly), and (2) using a PPO [66] agent trained for 50M timesteps. For both these modes, we ensure there is enough data diversity by collecting data using 8 differently initialized workers. We also add additional stochasticity to the pretrained PPO agent by using an ϵ-greedy like mechanism wherein at each timestep we take a random action with probability ϵ [3].

5.2 Methods

In our evaluations, we compare the following methods:

1. Randomly-initialized CNN encoder (RANDOM-CNN).
2. Variational autoencoder (VAE) [11] on raw observations.
3. Next-step pixel prediction model (PIXEL-PRED) inspired by the "No-action Feedforward" model from [21].
4. Contrastive Predictive Coding (CPC) [24], which maximizes the mutual information between current latents and latents at a future timestep.
5. SUPERVISED model which learns the encoder and the linear probe using the labels. The gradients are backpropagated through the encoder in this case, so this provides a base-case performance bound.

All methods use the same base encoder architecture, which is the CNN from [67], but adapted for the full 160x210 Atari frame size. To ensure a fair comparison, we use a representation size of 256 for each method. As a sanity check, we include a blind majority classifier (MAJ-CLF), which predicts label values based on the mode of the train set. More details in Appendix, section A.

5.3 Probing

We train a different 256-way[4] linear classifier with the representation under consideration as input. We ensure the distribution of realizations of each state variable has high entropy by pruning any variable with entropy less than 0.6. We also ensure there are no duplicates between the train and test set. We train each linear probe with 35,000 frames and use 5,000 and 10,000 frames each for validation and test respectively. We use early stopping and a learning rate scheduler based on plateaus in the validation loss.

6 Results

We report the F1 averaged across all categories for each method and for each game in Table 2 for data collected by random agent. In addition, we provide a breakdown of probe results in each category, such as small object localization or score/lives classification in Table 3 for the random agent. We

[3] For all our experiments, we used $\epsilon = 0.2$.

[4] Each RAM variable is a single byte thus has 256 possible values ranging from 0 to 255.

Table 2: Probe F1 scores averaged across categories for each game (data collected by random agents)

GAME	MAJ-CLF	RANDOM-CNN	VAE	PIXEL-PRED	CPC	ST-DIM	SUPERVISED
ASTEROIDS	0.28	0.34	0.36	0.34	0.42	**0.49**	N/A
BERZERK	0.18	0.43	0.45	**0.55**	**0.56**	0.53	0.68
BOWLING	0.33	0.48	0.50	0.81	0.90	**0.96**	0.95
BOXING	0.01	0.19	0.20	0.44	0.29	**0.58**	0.83
BREAKOUT	0.17	0.51	0.57	0.70	0.74	**0.88**	0.94
DEMONATTACK	0.16	0.26	0.25	0.32	0.57	**0.69**	0.83
FREEWAY	0.01	0.50	0.26	**0.81**	0.47	**0.81**	0.98
FROSTBITE	0.08	0.57	0.01	0.72	**0.76**	**0.75**	0.85
HERO	0.22	0.75	0.51	0.74	0.90	**0.93**	0.98
MONTEZUMAREVENGE	0.08	0.68	0.69	0.74	0.75	**0.78**	0.87
MSPACMAN	0.10	0.48	0.38	**0.74**	0.65	0.70	0.87
PITFALL	0.07	0.34	0.56	0.44	0.46	**0.60**	0.83
PONG	0.10	0.17	0.09	0.70	0.71	**0.81**	0.87
PRIVATEEYE	0.23	0.70	0.71	0.83	0.81	**0.91**	0.97
QBERT	0.29	0.49	0.49	0.52	0.65	**0.73**	0.76
RIVERRAID	0.04	0.34	0.26	**0.41**	**0.40**	0.36	0.57
SEAQUEST	0.29	0.57	0.56	0.62	**0.66**	**0.67**	0.85
SPACEINVADERS	0.14	0.41	0.52	**0.57**	0.54	**0.57**	0.75
TENNIS	0.09	0.41	0.29	0.57	**0.60**	**0.60**	0.81
VENTURE	0.09	0.36	0.38	0.46	0.51	**0.58**	0.68
VIDEOPINBALL	0.09	0.37	0.45	0.57	0.58	**0.61**	0.82
YARSREVENGE	0.01	0.22	0.08	0.19	0.39	**0.42**	0.74
MEAN	0.14	0.44	0.39	0.58	0.60	**0.68**	0.83

Table 3: Probe F1 scores for different methods averaged across all games for each category (data collected by random agents)

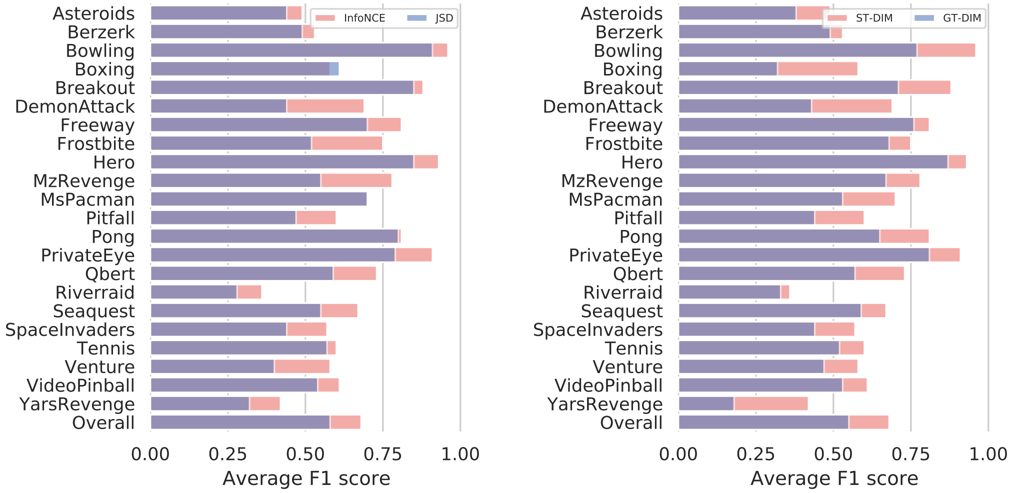
		RANDOM					
CATEGORY	MAJ-CLF	CNN	VAE	PIXEL-PRED	CPC	ST-DIM	SUPERVISED
SMALL LOC.	0.14	0.19	0.17	0.31	0.42	**0.51**	0.69
AGENT LOC.	0.12	0.31	0.30	0.48	0.43	**0.58**	0.83
OTHER LOC.	0.14	0.50	0.36	0.61	0.66	**0.69**	0.81
SCORE/CLOCK/LIVES/DISPLAY	0.13	0.58	0.53	0.76	0.83	**0.86**	0.93
MISC.	0.26	0.59	0.65	0.70	0.71	**0.74**	0.86

include the corresponding tables for these results with data collected by a pretrained PPO agent in tables 6 and 7. The results in table 2 show that ST-DIM largely outperforms other methods in terms of mean F1 score. In general, contrastive methods (ST-DIM and CPC) methods seem to perform better than generative methods (VAE and PIXEL-PRED) at these probing tasks. We find that RandomCNN is a strong prior in Atari games as has been observed before [68], possibly due to the inductive bias captured by the CNN architecture empirically observed in [69]. We find similar trends to hold on results with data collected by a PPO agent. Despite contrastive methods performing well, there is still a sizable gap between ST-DIM and the fully supervised approach, leaving room for improvement from new unsupervised representation learning techniques for the benchmark.

7 Discussion

Ablations: We investigate two ablations of our ST-DIM model: Global-T-DIM, which only maximizes the mutual information between the global representations and JSD-ST-DIM, which uses the NCE loss [70] instead of the InfoNCE loss, which is equivalent to maximizing the Jensen Shannon Divergence between representations. We report results from these ablations in Figure 3. We see from the results in that 1) the InfoNCE loss performs better than the JSD loss and 2) contrasting spatiotemporally (and not just temporally) is important across the board for capturing all categories of latent factors.

We found ST-DIM has two main advantages which explain its superior performance over other methods and over its own ablations. It captures small objects much better than other methods, and is



(a) InfoNCE vs JSD

(b) Effect of Spatial Loss

Figure 3: Different ablations for the ST-DIM model

more robust to the presence of easy-to-exploit features which hurts other contrastive methods. Both these advantages are due to ST-DIM maximizing mutual information of patch representations.

Capturing small objects: As we can see in Table 3, ST-DIM performs better at capturing small objects than other methods, especially generative models like VAE and pixel prediction methods. This is likely because generative models try to model every pixel, so they are not penalized much if they fail to model the few pixels that make up a small object. Similarly, ST-DIM holds this same advantage over Global-T-DIM (see Table 9), which is likely due to the fact that Global-T-DIM is not penalized if its global representation fails to capture features from some patches of the frame.

Robust to presence of easy-to-exploit features: Representation learning with mutual information or contrastive losses often fail to capture all salient features if a few easy-to-learn features are sufficient to saturate the objective. This phenomenon has been linked to the looseness of mutual information lower bounds [52, 53] and *gradient starvation* [71]. We see the most prominent example of this phenomenon in Boxing. The observations in Boxing have a clock showing the time remaining in the round. A representation which encodes the shown time can perform near-perfect predictions without learning any other salient features in the observation. Table 4 shows that CPC, Global T-DIM, and ST-DIM perform well at predicting the clock variable. However only ST-DIM does well on encoding the other variables such as the score and the position of the boxers.

We also observe that the best generative model (PIXEL-PRED) does not suffer from this problem. It performs its worst on high-entropy features such as the clock and player score (where ST-DIM excels), and does slightly better than ST-DIM on low-entropy features which have a large contribution in the pixel space such as player and enemy locations. This sheds light on the qualitative difference between contrastive and generative methods: contrastive methods prefer capturing high-entropy features (irrespective of contribution to pixel space) while generative methods do not, and generative methods prefer capturing large objects which have low entropy. This complementary nature suggests hybrid models as an exciting direction of future work.

8 Conclusion

We present a new representation learning technique which maximizes the mutual information of representations across spatial and temporal axes. We also propose a new benchmark for state representation learning based on the Atari 2600 suite of games to emphasize learning multiple generative factors. We demonstrate that the proposed method excels at capturing the underlying latent factors of a state even for small objects or when a large number of objects are present, which prove difficult for generative and other contrastive techniques, respectively. We have shown that

Table 4: Breakdown of F1 Scores for every state variable in Boxing for ST-DIM, CPC, and Global-T-DIM, an ablation of ST-DIM that removes the spatial contrastive constraint for the game Boxing

METHOD	VAE	PIXEL-PRED	CPC	GLOBAL-T-DIM	ST-DIM
CLOCK	0.03	0.27	0.79	0.81	**0.92**
ENEMY_SCORE	0.19	0.58	0.59	**0.74**	0.70
ENEMY_X	0.32	0.49	0.15	0.17	**0.51**
ENEMY_Y	0.22	**0.42**	0.04	0.16	0.38
PLAYER_SCORE	0.08	0.32	0.56	0.45	**0.88**
PLAYER_X	0.33	0.54	0.19	0.13	**0.56**
PLAYER_Y	0.16	**0.43**	0.04	0.14	0.37

our proposed benchmark can be used to study qualitative and quantitative differences between representation learning techniques, and hope that it will encourage more research in the problem of state representation learning.

Acknowledgements

We are grateful for the collaborative research environment provided by Mila and Microsoft Research. We thank Aaron Courville, Chris Pal, Remi Tachet, Eric Yuan, Chinwei-Huang, Khimya Khetrapal and Tristan Deleu for helpful discussions and feedback during the course of this work. We would also like to thank the developers of PyTorch [72] and Weights&Biases.

References

[1] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc., 1982. ISBN 0716715678.

[2] Robert D Gordon and David E Irwin. What's in an object file? evidence from priming studies. *Perception & Psychophysics*, 58(8):1260–1277, 1996.

[3] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[5] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182, 2016.

[6] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[7] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[8] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[9] Charles G Gross. Learning, perception, and the brain, 1968.

[10] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *International Conference on Learning Representations (ICLR)*, 2017.

[11] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations (ICLR)*, 2014.

[12] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *International Conference on Learning Representations (ICLR)*, 2017.

[13] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[14] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

[15] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. *arXiv preprint arXiv:1901.09005*, 2019.

[16] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 2018.

[17] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pages 2746–2754, 2015.

[18] Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1480–1490. JMLR. org, 2017.

[19] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, pages 2450–2462, 2018.

[20] Wuyang Duan. Learning state representations for robotic control: Information disentangling and multi-modal learning. Master's thesis, Delft University of Technology, 2017.

[21] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pages 2863–2871, 2015.

[22] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

[23] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning*, pages 531–540, 2018. URL `http://proceedings.mlr.press/v80/belghazi18a.html`.

[24] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[25] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *International Conference on Learning Representations (ICLR)*, 2019.

[26] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[27] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[28] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *International Conference on Learning Representations (Workshop Track)*, 2017.

[29] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time. iv. *Communications on Pure and Applied Mathematics*, 36 (2):183–212, 1983.

[30] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.

[31] David Barber and Felix Agakov. The im algorithm: A variational approach to information maximization. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pages 201–208, Cambridge, MA, USA, 2003. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2981345.2981371`.

[32] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.

[33] Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. *arXiv preprint arXiv:1809.01812*, 2018.

[34] Ben Poole, Sherjil Ozair, Aäron Van den Oord, Alexander A Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, 2019.

[35] Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39(3):407–428, 2015.

[36] Rico Jonschkowski, Roland Hafner, Jonathan Scholz, and Martin Riedmiller. Pves: Position-velocity encoders for unsupervised learning of structured state representations. *arXiv preprint arXiv:1705.09805*, 2017.

[37] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

[38] Herke van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3928–3934. IEEE, 2016.

[39] Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. Independently controllable factors. *arXiv preprint arXiv:1708.01289*, 2017.

[40] Giuseppe Cuccu, Julian Togelius, and Philippe Cudré-Mauroux. Playing atari with six neurons. *International Conference on Autonomous Agents and Multiagent Systems*, 2019.

[41] Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Near-optimal representation learning for hierarchical reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2019.

[42] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[43] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[44] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2018. ISSN 1939-3539. doi: 10.1109/tpami.2018.2857768. URL `http://dx.doi.org/10.1109/TPAMI.2018.2857768`.

[45] Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens, 2017.

[46] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, 2018.

[47] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rJ4km2R5t7`.

[48] Karl Friston. A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences*, 360(1456):815–836, 2005.

[49] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79, 1999.

[50] William Bialek and Naftali Tishby. Predictive information. *arXiv preprint cond-mat/9902341*, 1999.

[51] Stephanie E Palmer, Olivier Marre, Michael J Berry, and William Bialek. Predictive information in a sensory population. *Proceedings of the National Academy of Sciences*, 112(22):6908–6913, 2015.

[52] David McAllester and Karl Statos. Formal limitations on the measurement of mutual information. *arXiv preprint arXiv:1811.04251*, 2018.

[53] Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron Van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. *arXiv preprint arXiv:1903.11780*, 2019.

[54] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

[55] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.

[56] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141. IEEE, 2018.

[57] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

[58] Francesco Locatello, Stefan Bauer, Mario Lucic, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *International Conference on Machine Learning*, 2019.

[59] Zach Whalen and Laurie N Taylor. Playing the past. *History and Nostalgia in Video Games. Nashville, TN: Vanderbilt University Press*, 2008.

[60] Steve Engelhardt. BJARS.com Atari Archives. `http://bjars.com`, 2019. [Online; accessed 1-March-2019].

[61] Thomas Jentzsch and CPUWIZ. Atariage atari 2600 forums, 2019. URL `http://atariage.com/forums/forum/16-atari-2600/`.

[62] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[63] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 35(8): 1798–1828, 2013.

[64] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. *International Conference on Learning Representations (ICLR)*, 2018.

[65] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[66] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[67] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. MIT Press, 2013.

[68] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *International Conference on Learning Representations (ICLR)*, 2019.

[69] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[70] Aapo Hyvärinen and Petteri Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999.

[71] Remi Tachet des Combes, Mohammad Pezeshki, Samira Shabanian, Aaron Courville, and Yoshua Bengio. On the learning dynamics of deep neural networks. *arXiv preprint arXiv:1809.06848*, 2018.

[72] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[73] Ken Kansky, Tom Silver, David A Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1809–1818. JMLR. org, 2017.

[74] Amy Zhang, Yuxin Wu, and Joelle Pineau. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018.

A Architecture Details

All architectures below use the same encoder architecture as a base, which is the one used in [67] adapted to work for the full 160x210 frame size as shown in figure 4.

1. **Linear Probe**:
   The linear probe is a linear layer of width 256 with a softmax activation and trained with a cross-entropy loss.
2. **Majority Classifier** (maj-clsf):
   The majority classifier is parameterless and just uses the mode of the distribution of classes from the training set for each state variable and guesses that mode for every example on the test set at test time.

3. **Random-CNN**:
   The Random-CNN is the base encoder with randomly initialzied weights and no training

4. **VAE and Pixel-Pred**:
   The VAE and Pixel Prediction model use the base encoder plus each have an extra 256 wide fully connected layer to parameterize the log variance for the VAE and to more closely resemble the *No Action Feed Forward* model from [21]. In addition bith models have a deconvolutional network as a decoder, which is the exact transpose of the base encoder in figure 4.

5. **CPC**:
   CPC uses the same architecture as described in [24] with our base encoder from figure 4 being used as the image encoder g_{enc}.

6. **ST-DIM (and its ablations)**:
   ST-DIM and the two ablations, JSD-ST-DIM and Global-T-DIM, all use the same architecture which is the base encoder plus a 1x256x256 bilinear layer.

7. **Supervised**:
   The supervised model is our base encoder plus our linear probe trained end-to-end with the ground truth labels.

8. **PPO Features** (section E):
   The PPO model is our base encoder plus two linear layers for the policy and the value function, respectively.

B Preprocessing and Hyperparameters

We preprocess frames primarily in the same way as described in [67], with the key difference being we use the full 160x210 images for all our experiments instead of downsampling to 84x84. Table 5 lists the hyper-parameters we use across all games. For all our experiments, we use a learning rate scheduler based on plateaus in the validation loss (for both contrastive training and probing).

Table 5: Preprocessing steps and hyperparameters

Parameter	**Value**
Image Width	160
Image Height	210
Grayscaling	Yes
Action Repetitions	4
Max-pool over last N action repeat frames	2
Frame Stacking	None
End of episode when life lost	Yes
No-Op action reset	Yes
Batch size	64
Sequence Length (CPC)	100
Learning Rate (Training)	3e-4
Learning Rate (Probing, non supervised)	5e-2
Learning Rate (Probing, supervised)	3e-4
Entropy Threshold	0.6
Encoder training steps	70000
Probe training steps	35000
Probe test steps	10000

Compute infrastructure: We run our experiments on a autoscaling-cluster with multiple P100 and V100 GPUs. We use 8 cores per machines to distribute data collection across different workers.

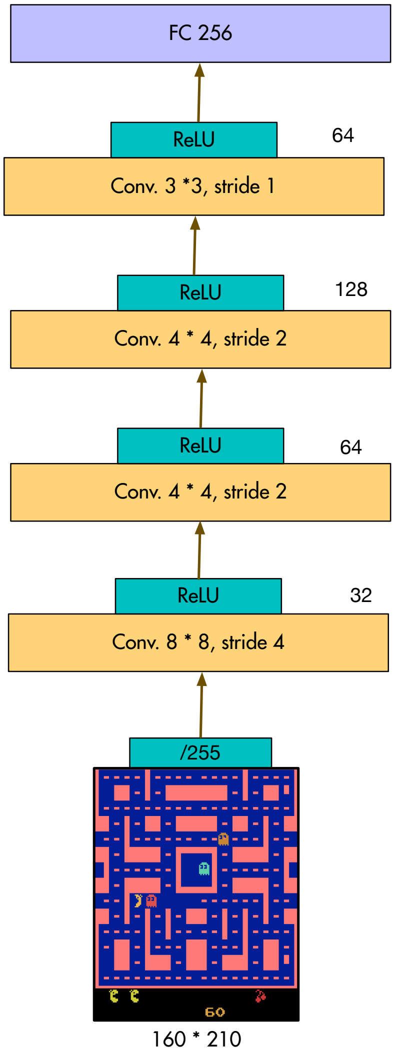


Figure 4: The base encoder architecture used for all models in this work

C Results with Probes Trained on Data Collected By a Pretrained RL agent

In addition to evaluating on data collected by a random agent, we also evaluate different representation learning methods on data collected by a pretrained PPO [66] agent. Specifically, we use a PPO agent trained for 50M steps on each game. We choose actions stochastically by sampling from the PPO agent's action distribution at every time step, and inject additional stochasticity by using an ϵ-greedy mechanism with $\epsilon = 0.2$. Table 6 shows the game-by-game breakdown of mean F1 probe scores obtained by each method in this evaluation setting. Table 7 additionally shows the category-wise breakdown of results for each method. We observe a similar trend in performance as observed earlier with a random agent.

D More Detailed Ablation Results

We expand on the results reported on different ablations (JSD-ST-DIM and Global-T-DIM) of STDIM in the main text, and provide a game by game breakdown of results in Table 8, and a category-wise breakdown in Table 9.

Table 6: Probe F1 scores for all games for data collected by a pretrained PPO (50M steps) agent

	MAJ-CLF	RANDOM-CNN	VAE	PIXEL-PRED	CPC	ST-DIM	SUPERVISED
ASTEROIDS	0.23	0.31	0.35	0.31	0.38	**0.40**	0.98
BERZERK	0.13	0.33	0.35	0.39	0.38	**0.43**	0.87
BOWLING	0.23	0.61	0.51	0.81	0.90	**0.98**	0.87
BOXING	0.05	0.30	0.32	0.57	0.32	**0.66**	0.87
BREAKOUT	0.09	0.34	0.59	0.47	0.55	**0.66**	0.76
DEMONATTACK	0.03	0.19	0.18	0.26	0.43	**0.58**	0.76
FREEWAY	0.01	0.36	0.02	**0.60**	0.38	**0.60**	0.85
FROSTBITE	0.13	0.57	0.46	0.70	**0.74**	0.69	0.96
HERO	0.12	0.54	0.60	0.68	**0.86**	0.77	0.88
MONTEZUMAREVENGE	0.08	0.68	0.58	0.72	**0.77**	**0.76**	0.70
MSPACMAN	0.06	0.34	0.36	**0.52**	0.45	0.49	0.92
PITFALL	0.16	0.39	0.37	0.53	0.69	**0.74**	0.87
PONG	0.02	0.10	0.24	0.67	0.63	**0.79**	0.99
PRIVATEEYE	0.24	0.71	0.69	0.87	0.83	**0.91**	0.65
QBERT	0.06	0.36	0.38	0.39	**0.51**	0.48	0.59
RIVERRAID	0.04	0.25	0.21	**0.34**	0.31	0.22	0.90
SEAQUEST	0.29	0.64	0.58	**0.75**	0.69	**0.75**	0.65
SPACEINVADERS	0.02	0.28	0.30	**0.41**	0.32	**0.41**	0.61
TENNIS	0.15	0.25	0.13	**0.65**	0.63	**0.65**	0.69
VENTURE	0.05	0.32	0.36	0.37	0.50	**0.59**	0.79
VIDEOPINBALL	0.13	0.36	0.42	**0.56**	**0.57**	0.54	0.77
YARSREVENGE	0.03	0.14	0.26	0.23	0.38	**0.43**	0.74
MEAN	0.11	0.38	0.38	0.54	0.56	**0.61**	0.80

Table 7: Probe F1 scores for different methods averaged across all games for each category (data collected by a pretrained PPO (50M steps) agent

	MAJ-CLF	RANDOM-CNN	VAE	PIXEL-PRED	CPC	ST-DIM	SUPERVISED
SMALL LOC.	0.10	0.13	0.14	0.27	0.31	**0.41**	0.63
AGENT LOC.	0.11	0.34	0.34	0.48	0.45	**0.54**	0.85
OTHER LOC.	0.14	0.47	0.38	0.56	0.58	**0.61**	0.74
SCORE/CLOCK/LIVES/DISPLAY	0.05	0.44	0.50	0.71	0.74	**0.80**	0.90
MISC.	0.19	0.53	0.57	0.62	0.65	**0.67**	0.85

E Probing Pretrained RL Agents

We make a first attempt at examining the features that RL agents learn. Specifically, we train linear probes on the representations from PPO agents that were trained for 50 million frames. The architecture of the PPO agent is described in section A. As we see from table 10, the features perform poorly in the probing tasks compared to the baselines. Kansky et al. [73], Zhang et al. [74] have also argued that model-free agents have trouble encoding high level state information. However, we note that these are preliminary results and require thorough investigation over different policies and models.

Table 8: Probe F1 scores for different ablations of ST-DIM for all games averaged across each category (data collected by random agents)

	JSD-ST-DIM	GLOBAL-T-DIM	ST-DIM
ASTEROIDS	0.44	0.38	**0.49**
BERZERK	0.49	0.49	**0.53**
BOWLING	0.91	0.77	**0.96**
BOXING	**0.61**	0.32	0.58
BREAKOUT	0.85	0.71	**0.88**
DEMONATTACK	0.44	0.43	**0.69**
FREEWAY	0.70	0.76	**0.81**
FROSTBITE	0.52	0.68	**0.75**
HERO	0.85	0.87	**0.93**
MONTEZUMAREVENGE	0.55	0.67	**0.78**
MSPACMAN	**0.70**	0.53	**0.70**
PITFALL	0.47	0.44	**0.60**
PONG	**0.80**	0.65	**0.81**
PRIVATEEYE	0.79	0.81	**0.91**
QBERT	0.59	0.57	**0.73**
RIVERRAID	0.28	0.33	**0.36**
SEAQUEST	0.55	0.59	**0.67**
SPACEINVADERS	0.44	0.44	**0.57**
TENNIS	0.57	0.52	**0.60**
VENTURE	0.40	0.47	**0.58**
VIDEOPINBALL	0.54	0.53	**0.61**
YARSREVENGE	0.32	0.18	**0.42**
MEAN	0.58	0.55	**0.68**

Table 9: Different ablations of ST-DIM. F1 scores for for each category averaged across all games (data collected by random agents)

	JSD-ST-DIM	GLOBAL-T-DIM	ST-DIM
Small Loc.	0.44	0.37	**0.51**
Agent Loc.	0.47	0.43	**0.58**
Other Loc.	0.64	0.53	**0.69**
Score/Clock/Lives/Display	0.69	0.76	**0.86**
Misc.	0.64	0.66	**0.74**

Table 10: Probe results on features from a PPO agent trained on 50 million timesteps compared with a majority classifier and random-cnn baseline. The probes for all three methods are trained with data from the PPO agent that was trained for 50M frames

	MAJ-CLF	RANDOM-CNN	PRETRAINED-RL-AGENT
ASTEROIDS	0.23	**0.31**	**0.31**
BERZERK	0.13	**0.33**	0.30
BOWLING	0.23	**0.61**	0.48
BOXING	0.05	**0.30**	0.12
BREAKOUT	0.09	**0.34**	0.23
DEMONATTACK	0.03	**0.19**	0.16
FREEWAY	0.01	**0.36**	0.26
FROSTBITE	0.13	**0.57**	0.43
HERO	0.12	**0.54**	0.42
MONTEZUMAREVENGE	0.08	**0.68**	0.07
MSPACMAN	0.06	**0.34**	0.26
PITFALL	0.16	**0.39**	0.23
PONG	0.02	**0.10**	0.09
PRIVATEEYE	0.24	**0.71**	0.31
QBERT	0.06	**0.36**	0.34
RIVERRAID	0.04	**0.25**	0.10
SEAQUEST	0.29	**0.64**	0.50
SPACEINVADERS	0.02	**0.28**	0.19
TENNIS	0.15	0.25	**0.66**
VENTURE	0.05	**0.32**	0.08
VIDEOPINBALL	0.13	**0.36**	0.21
YARSREVENGE	0.03	**0.14**	0.09
MEAN	0.11	**0.38**	0.27