Action-based Unsupervised State Representation Learning in Atari

Joongkyu Lee
Seoul National University
jklee0717@snu.ac.kr

Dongyoung Kim
Seoul National University
kimdy0324@snu.ac.kr

Ji Yong Kim
Seoul National University
jykim00324@snu.ac.kr

Jisub Kwak
Seoul National University
kjs675252@snu.ac.kr

Joonseok Lee
Seoul National University
Google Research
joonseok@google.com

Abstract

State representation learning aims to extract useful features from the observations received by a Reinforcement Learning agent interacting with an environment. These features allow the agent to take advantage of the low-dimensional and informative representation to improve the efficiency in solving tasks. Especially, recent contrastive state representation learning methods have shown impressive performances. While these contrastive methods mainly focus on generating invariant features by minimizing the distance between two consecutive states, they are prone to overlook action relationship between them.

In this work, we introduce novel method that learns state representations by not only maximizing mutual information across spatially and temporally distinct features of a neural encoder of the observations but also training an auxiliary network to predict the action taken by the agent to go from one observation to the next. Our method shows significant performance improvements over state-of-the-art generative and contrastive representation learning methods 7

1. Introduction

One major problem of current state-of-the-art Reinforcement Learning (RL) algorithms is still the need for millions of training examples to learn a good or near-optimal policy to solve the given task. To mitigate this problem, One idea, the researchers came up with is decoupling representation learning from the actual policy learning for the RL agents. Representations that precisely capture the true state of the environment should empower agents to effectively transfer knowledge across different tasks in the environment, and enable learning with fewer interactions.

Modern image-recognition systems learn image representations from large collections of images and corresponding semantic annotations. These annotations can be provided in the form of class labels [35], hashtags [31], bounding boxes [29], etc. Pre-defined semantic annotations scale poorly to the long tail of visual concepts [23], which hampers further improvements in image recognition.

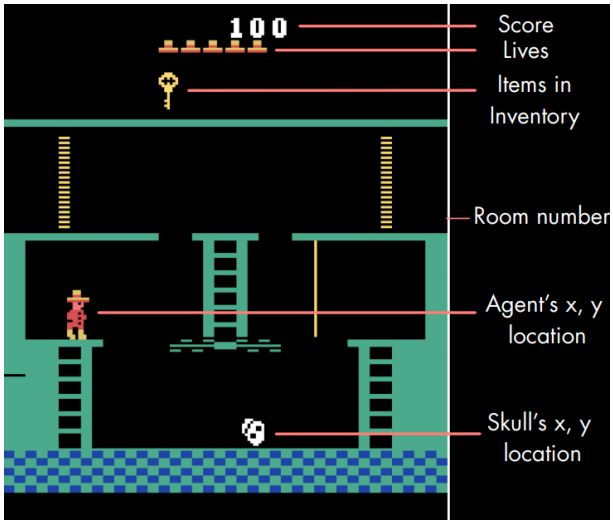


Figure 1. We use Atari 2600 games to evaluate state representations. We leveraged the source code of the games to annotate the RAM states with important state variables such as the location of various objects in the game. We compare various unsupervised representation learning techniques based on how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

Self-supervised learning tries to address these limitations by learning image representations from the pixels themselves without relying on pre-defined semantic annotations. In the context of learning state representations [28], current unsupervised methods rely on generative decoding of the

data using either VAEs [12,18,21,39] or prediction in pixel-space [15, 33]. 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. Spatiotemporal Deep Infomax (ST-DIM) [1] is a self-supervised state representation learning technique which exploits the global-local contrastive task of visual observations in a reinforcement learning setting. However, ST-DIM does not utilize the action-related information. Also, certain action may change the state significantly, which means x_t and x_{t+1} are no longer similar.

In this work, motivated by the limitaion of ST-DIM, we assumed that by exploiting action-related information, we could learn better state representation. Action-based Spatiotemporal Deep Infomax (ABST-DIM) improves ST-DIM by considering action relationship between consecutive observations. The objective of ABST-DIM is maximizing not only the mutual information between global and local representations in consecutive time steps, but also conditional likelihood $P(a_t \mid x_t, x_{t+1})$, given a triplet $\{x_t, a_t, x_{t+1}\}$ composed of two consecutive observations, x_t and x_{t+1}, and the action a_t taken by the agent. Even though predicting action a_t is not directly related with learning representations, we showed that it is still beneficial to use the information if it is somewhat related.

To evaluate ABST-DIM, we used the Arcade Learning Environment [4] benchmark based on Atari 2600 games (See Fig. 1.)

Contributions.

- We propose a new self-supervised state representation learning technique, named ABST-DIM, which exploits action relationship between consecutive observations in a reinforcement learning setting.
- Based on a baseline algorithm ST-DIM, we add a new loss term L_a to maximize conditional likelihood $P(a_t \mid x_t, x_{t+1})$.
- A variety of experimental results show clear advantages of ABST-DIM over the existing state-of-the-art representation learning methods.
- Our new approach is applicable to any kinds of RL contrastive representation learning architectures.

2. Related works

Unsupervised representation learning via mutual information objectives. Recent work 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. [3] estimate the mutual information with neural networks using the Donsker-Varadhan representation of the KL divergence [11], while Chen et al. [7] use the variational bound from Barber and Agakov [2] to learn discrete latent representations. Hjelm et al. [22] learn representations by maximizing the Jensen-Shannon divergence between joint and product of marginals of an image and itspatches. van den Oord et al. [36] maximize mutual information using a multi-sample version of noise contrastive estimation [17, 30]. 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. Recently, Cuccu et al. [9] and Eslami et al. [13] show that visual processing and policy learning can be effectively decoupled in pixel-based environments. Jonschkowski and Brock [25] and Jonschkowski et al. [26] 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 [20,37,39]. Single-view TCN [30] and TDC [37] learn state representations using self-supervised objectives that leverage temporal information in demonstrations. ST-DIM [1] can be considered as an extension of TDC and TCN that also leverages the local spatial structure .

A few works have focused on learning state representations that capture factors of an environment that are under the agent's control in order to guide exploration [8, 27] or unsupervised control [38]. EMI [27] harnesses mutual information between state embeddings and actions to learn representations that capture just the controllable factors of the environment, like the agent's position. ST-DIM [1] captures every temporally evolving factors (not just the controllable ones) in an environment, like the position of enemies, score, balls, missiles, moving obstacles, and the agent position. Lastly, ST-DIM uses an InfoNCE objective instead of the JSD one used in EMI. Our work is also closely related to recent work in learning object-oriented representations. [5, 16, 41]

Inverse prediction. Inverse models rely on a loss function that computes the prediction error on the action a_t taken by the agent to move from state s_t to s_{t+1}. We can use two consecutive states s_t and s_{t+1} to predict which action a_t made the transition happen. The inverse model g is implemented by Pathak et al. [14] and defined as $\hat{a_t} = g(s_t, s_{t+1}; \theta_I)$. where, $\hat{a_t}$ is the predicted estimate of the action a_t. This inverse model parameters θ_I are trained to optimize, $\min_{\theta_I} L_I(\hat{a_t}, a_t)$,where L_I is the loss function that measures the discrepancy between the predicted and actual actions. In case a_t is discrete, the output of g is a soft-max distribution across all possible actions and mini-

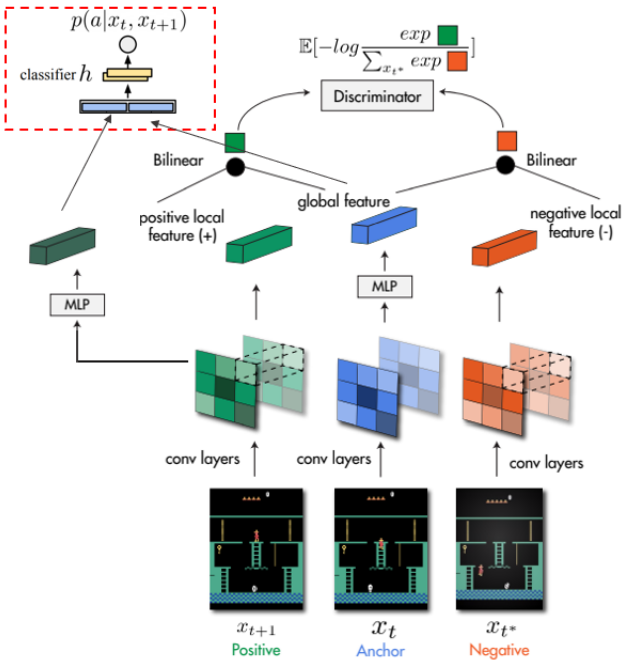


Figure 2. A schematic overview of Action-based SpatioTemporal DeepInfoMax(ABST-DIM). The red-dotted part is our new approach to predict action made the state trasition happen. Without this part, the model is just ST-DIM where there are different mutual information objectives: local-local infomax and global-local infomax. However, The detailed structure varies from method to method.

mizing L_I amounts to maximum likelihood estimation of θ_I under a multinomial distribution.

3. Method

3.1. ABST-DIM: Action-based Spatiotemporal Deep Info-max

For improved representation learning, we aim to exploit not only the mutual information between features embedded from states across different time and space, but also the action between the states. This is based on the belief that employing actions would aid to capturing observations related to movement of the agent rather than variation of the environment independent of the agent.

We follow ST-DIM [1] for mutual information estimation, which uses spatial and temporal relationship of locally, and globally embedded features. We assume a set of observations as state space $S = \{x_1, x_2, ..., x_n\}$ and a representation network $f : S \rightarrow \mathbb{R}^p$ which maps the states into p-dimensional vector. Global features can be denoted as $f(x_t)$ where x_t is an observation at time t, while local features are denoted as $f_l(x_t)$ where $l = (m, n)$ is a location on intermediate layer of f at which the local feature is produced. Score function of features, $f(x_i)$ and $f(x_j)$ for example, is defined as a bilinear model $f(x_i)^T W f(x_j)$. Scores between consecutive observations (x_t, x_{t+1}) and non-consecutive observations (x_t, x_{t^*}) is combined with infoNCE to set objective of maximizing mutual information among consecutive states. As like ST-DIM, we construct two losses: the global-local objective (GL) and the local-local objective (LL). The global-local objective is as follows This can be formulated as follows:

$$L_{GL} = \sum_l -log\frac{exp(f(x_t)^T W_g f_l(x_{t+1}))}{\sum_{x_{t^*} \in X_{next}} exp(f(x_t)^T W_g f_l(x_{t^*}))} \quad (1)$$

$$L_{LL} = \sum_l -log\frac{exp(f_l(x_t)^T W_l f_l(x_{t+1}))}{\sum_{x_{t^*} \in X_{next}} exp(f_l(x_t)^T W_l f_l(x_{t^*}))} \quad (2)$$

where X_{next} indicates set of next states, L_{GL} and L_{LL} denote objective according to mutual information with global and local features respectively.

We leverage this method by adding action estimation to the objective. For this, action is collected in addition to the states in the data acquisition process which is done by an agent exploring the environment following certain policy, in this case random policy(steps through the environment by selecting actions randomly). This collection of actions is used as a ground truth for training action classifiers. In our evaluations, we compare the following methods:

1. Fully Connected Discriminator
2. Region Sensitive Module
3. Attention Mask

Combining all the objectives, the final loss function is as follows:

$$L = L_{GL} + L_{LL} + \lambda L_a \quad (3)$$

where L_{GL} and L_{LL} denote objective according to mutual information with global and local features respectively, and L_a is a maximum likelihood loss between estimated and ground truth distribution of action. Degree of significance of the action estimation is regulated via factor λ. Overall architecture is described at Figure 2. Based on existing ST-DIM architecture, we added auxiliary network to predict the action a_t. Note that we tried three different methods to build the structures of the action predicting networks. The model is forced to learn not only semantic meaning of states(images), but also action-relationship between consecutive states(temporally consecutive images). In other words, it is not a simple unsupervised learning task anymore, because we use subsidiary labeled information.

3.1.1 Auxiliary Fully Connected(AFC) Discriminator

First, we simply added Auxiliary Fully Connected(AFC) layer to discriminate the action. Detailed network architecture is described at Figure 3.(a). Given a triplet $\{x_t, a_t, x_{t+1}\}$ composed of two consecutive observations, x_t and x_{t+1}, and the action taken by the agent a_t, we parameterise the conditional likelihood as $P(a \mid x_t, x_{t+1}) = h(f(x_t), f(x_{t+1}))$, where h is a one fully connected layer followed by a softmax. Here, $f(x_t)$ and $f(x_{t+1})$ are global feature embedding of current and next state respectively. The result can be denoted as $h(f(x_t), f(x_{t+1}))$ which is trained by maximum likelihood

The loss function of Auxiliary Fully Connected Discriminator is as follows:

$$L_a = L_a(h(f(x_t), f(x_{t+1})), a_t) \tag{4}$$

3.1.2 Auxiliary Region Sensitive Module(ARSM)

We also present Auxiliary Region Sensitive module(ARSM) based on two considerations. First, humans tend to look at some regions directly related to rewards rather than looking at the entire game screen. Second, important information in the game is concentrated on each essential object rather than in the background that is not directly related to the game.

Detailed network architecture is described at Figure 3.(b). To design ARSM, we employ two convolutions with ELU activation function [40]. ARSM's input $f_{final}(x_t)$ is the output of last convolutional layer of the encoder at time t, and ARSM's output $A = RS(f_{final}(x_t))$ is a score map of the same size as the input $f_{final}(x_t)$, where ARSM is $RS()$. Each element on a score map A corresponds to a spatial location on $f_{final}(x_t)$ and means each local pixel's importance to represent the action from entire image. We apply sigmoid function to normalize the score map A, where A is the learned probability distributions.

We generate the 2-D image embedding vector F, defined as the element-wise multiplication of A and $f_{final}(x_t)$, as $F = A \bigotimes f_{final}(x_t)$. To obtain one dimensional vector, we apply additional fully-connected layer. These 1-D vectors coming from two different time domain encoders are concatenated to estimate action of agent.

The loss function of Auxiliary Region Sensitive module is as follows:

$$L_a = L_a(h(FC(F(x_t)), FC(F(x_{t+1}))), a_t) \tag{5}$$

where F is the element-wise multiplication of score map A and $f_{final}(x_t)$, and FC is a fully connected layer.

3.1.3 Attention Mask Module(AMM)

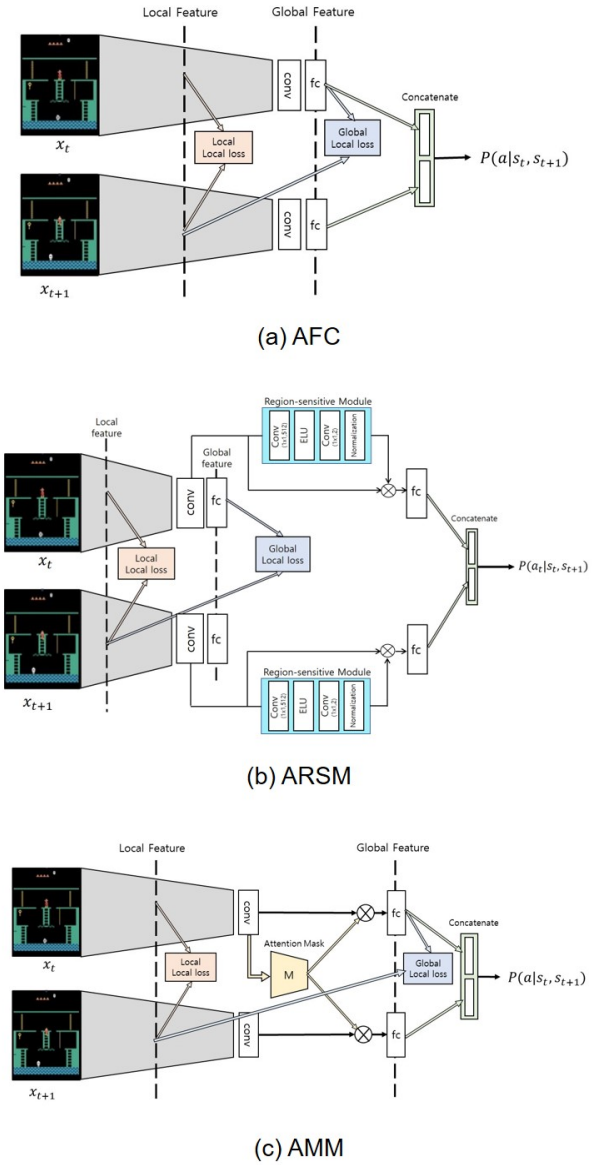


Figure 3. Detailed structure of each network for the three methodologies. (a) is Auxiliary Fully Connected Discriminator(AFC), simple discriminator for action estimation. (b) is Auxiliary Region Sensitive module(ARSM), considering spatial location to represent the action from entire image. (c) is Attention Mask Module(AMM), predicting the action and embedding global features.

Lastly, we applied Attention Mask Module(AMM) before the last fully connected layer of the encoder to not only predict the action but also embed global features. Detailed network architecture is described at Figure 3.(c). We aimed to mask out insignificant regions in terms of both preserving generative features and action inference, in the embedding stage. Before reducing 2D convolutional fea-

tures into a 1D latent vector at the last stage of embedding network, the final convolutional feature map $f_{final}(x_t)$ goes through attention mask module [19] (same architecture as the region sensitive module) which outputs a single channel layer of normalized weights for each spatial coordinates. This weights are multiplied element-wise to the feature map from which it originated and also to the final feature map of Siamese encoder $f_{final}(x_{t+1})$, reflecting regional interest in both global-global contrastive learning and action inference process. In other words, the final embeddings $f(x_t)$ and $f(x_{t+1})$ in the total loss function L is now $f_{final}(x_t) \bigotimes m(f_{final}(x_t))$ and $f_{final}(x_{t+1}) \bigotimes m(f_{final}(x_t))$ reduced to a vector through a fully connected hidden layer, where m denotes the attention mask network. The loss function of Attention Mask Module is same as equation(5). However, L_{GL} is slightly changed because Global features at time t, f_t is no longer same. Global feature f_t can be reformulated as follows:

$$f_t(x_t) = f_{final}(x_t) \bigotimes m(f_{final}(x_t)) \tag{6}$$

$$f_t(x_{t+1}) = f_{final}(x_{t+1}) \bigotimes m(f_{final}(x_t)) \tag{7}$$

3.2. The Atari Annotated RAM Interface (Atari-ARI)

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.

Annotating Atari RAM. We used the same benchmark Arcade Learning Environment(ALE, [4]) as Ankesh Anand et al. [1]. 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 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. 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.

3.3. Implementation Details

We evaluate the performance of different representation learning methods on our benchmark. Our experimental pipeline consists of first training an encoder and an auxiliary network, then freezing their weights and evaluating their 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 our network, and only used to train the linear classifier on top of the representation.

Dataset. We collect the data using a random agent (steps through the environment by selecting actions randomly). Note that learning representation is agnostic to the policy. The reason why we collect data by a agent is to utilize a triplet $\{x_t, a_t, x_{t+1}\}$ composed of two consecutive observations including action, not only $\{x_t, x_{t+1}\}$. We ensure there is enough data diversity by collecting data using 8 differently initialized workers. We train the model with 35,000 frames and use 5,000 and 10,000 frames each for validation and test respectively.

Network Architecture. Network used in encoding observations consists of 4 sequential convolution layers each activated by ReLU, outcome of which is flattened and fed into a dense layer to output a global feature vector. Local feature is the intermediate result from third convolution layer. In detail, $(channels, kernelsize, stride)$ of each convolution layer is $(32, 8, 4)$, $(32, 64, 4)$, $(64, 128, 4)$, $(128, 64, 3)$, respectively. Global feature and local feature is resized into a same sized vector by a dense layer before loss evaluation. For action estimation, two resized vectors are concatenated and processed with another dense layer to match the size of the one-hot encoded action space before performing MLE.

For Auxiliary Region Sensitive Module(ARSM), two 1×1 convolution layers are used with ELU activation function. Each convolution layer is (512,9,6), (2,9,6) in that order. An embedding vector for the image x_t is calculated by element-wise multiplying the input and output of ARSM. For AMM, the network architecture is the same as for ARSM.

Hyperparameters. As mentioned, we give variation to the weight of the loss function L_a, which measures the discrepancy between the predicted and actual actions, by applying hyperparameter λ. In our experiment we use three values, $(0.5, 1, 3)$ to observe the impact of action estimation in each Atari environment.

GAME	ST-DIM(Baseline)	ABST-DIM(empirical best λ) AFC(Method 1)	ARSM(Method 2)	AMM(Method 3)
Berzerk	0.51	**0.56**(3)	0.53(1)	0.49(1)
Pitfall	0.59	**0.72**(3)	0.70(0.5)	0.61(3)
Pong	0.82	**0.84**(0.5)	0.83(0.5)	0.78(0.5)
Qbert	0.72	**0.77**(1)	0.74(1)	0.70(0.5)
Mean	0.66	**0.72**	0.70	0.64

Table 1. F1 scores of each method averaged across categories for each game: a) Auxiliary Fully Connected(AFC) Discriminator b) Auxiliary Region Sensitive Module(ARSM) c) Attention Mask Module(AMM)

GAME	VAE	CPC	ST-DIM(baseline)	ABST-DIM AFC(λ) 0.5	1	3	Max(vs ST-DIM)
Asteroids	0.36	0.42	0.45	0.45	**0.47**	0.43	0.47(+0.02)
Berzerk	0.45	**0.56**	0.51	0.50	0.52	**0.56**	0.56(+0.05)
Boxing	0.20	0.29	0.62	**0.67**	0.62	0.58	0.67(+0.05)
DemonAttack	0.26	0.57	0.66	0.66	**0.69**	0.65	0.69(+0.03)
Hero	0.69	0.90	0.90	0.91	**0.93**	0.91	0.93(+0.02)
Mspacman	0.56	0.65	0.70	**0.71**	0.70	0.70	0.71(+0.01)
Pitfall	0.35	0.46	0.59	0.70	0.68	**0.72**	0.72(+0.13)
Pong	0.09	0.71	0.82	**0.84**	0.82	0.80	0.84(+0.02)
PrivateEye	0.71	0.81	0.88	**0.91**	0.90	**0.91**	0.91(+0.03)
Qbert	0.49	0.65	0.72	0.74	**0.77**	0.74	0.77(+0.05)
Tennis	0.29	**0.60**	0.56	0.57	0.59	0.57	0.59(+0.03)
Venture	0.38	0.51	0.57	**0.61**	0.57	0.58	0.61(+0.04)
YarsRevenge	0.08	0.39	0.40	0.41	**0.45**	0.41	0.45(+0.05)
Mean	0.38	0.58	0.64	0.67	0.67	0.66	**0.69**(+0.05)

Table 2. F1 scores of Auxiliary Fully Connected(AFC) Discriminator averaged across categories for each game.(data collected by random agents)

4. Experiments

4.1. Baselines

An important baseline is the ST-DIM [1] as it proposes self-supervised state representation learning technique which exploits the spatial-temporal nature of visual observations in a reinforcement learning. It magnify mutual information of representations across spatial and temporal domains. Based on ST-DIM architecture, we tried three different approaches to attach an auxiliary action-predicting network.

4.2. Training details

We preprocess frames primarily in the same way as described in [32], with the key difference being we use the full 210x160 images for all our experiments instead of downsampling to 84x84. We use early stopping and a learning rate scheduler based on plateaus in the validation loss. 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.

4.3. Evaluation: Linear probing

Evaluating representation learning methods is a challenging open problem. 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 or in transfer learning tasks. 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. [6, 10, 36] We train a different 256-way[1] linear classifier with the representation under consideration as input. Specifically, to evaluate a representation we train linear classifiers predicting each state variable, and we report the mean F1 score.

4.4. Results

As introduced in section 3, we applied three different auxiliary networks for action inference task. The results are

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

CATEGORY	ST-DIM(Baseline)	ABST-DIM AFC
SMALL LOC	0.46	0.50
AGENT LOC	0.48	0.51
OTHER LOC	0.60	0.64
SOCRE/CLOCK/LIVES/DISPLAY	0.89	0.92
MISC	0.78	0.80

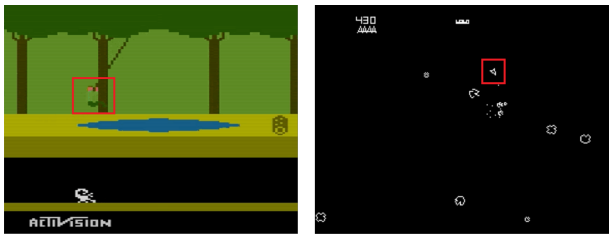
Table 3. F1 scores for different methods averaged across all games for each category (data collected by random agents)

shown in Table 1. As suggested by the result, action inference in general improved the representation performance. However, sophisticated methods such as ARSM and AMM did not shown any performance advantage over relatively simple AFC. We believe this is caused by the bigger emphasis on action inference network and focusing more on the action classification itself, not maximizing the similarity between the positive samples.

We employed the best performing network AFC for action inference, and collected the F1 score averaged across all categories for each existing method and ABST-DIM with different λ for each game in Table 2. 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. The results show that ABST-DIM largely outperforms other methods in terms of F1 score in every category. For the best λ, the coefficient of L_a, ABST-DIM AFC improved ST-DIM by 5% in overall. Specifically, in case of Pitfall, Berzerk and Boxing, the proposed algorithm indicates superior performance, increasing the performance 13%, 5% and 5% respectively. Conversely, in the case of Mspacman and Asteroids, the performance was not significantly improved.

The F1 score according to the different λ values indicates a different tendency to each Atari environment, suggesting that it is related to the dependency of the agent's behavior for each game. For the larger λ, the agent's action is more considered during self-supervised learning process rather than the mutual information on the spatial and temporal axes. Therefore, when an environment has few variability independent of the agent's actions(such as Pitfall and Berzerk), ABST-DIM shows significant performance improvements. Otherwise, when there are a lot of the variability in the environment that is not affected by the action taken by the agent(such as Mspacman and Asteroids), our new action-based approach has a little effect. To be more specific, the obstacles in Pitfall tends to move in a certain range with a consistent pattern while in Asteroids, movement of the enemies is stochastic and cannot be intuitively distinguished from the agent's action. Refer Figure 4 for more details. The Figure 4 shows that our new model improved a lot better when there are few uncontrollable objects in the environments. For example, in Asteroids, variety size of asteroids are drifting in various directions on the screen, which means predicting action is little helpful to learning representations.

To summarize, using action inference as an auxiliary objective in addition to contrastive loss is helpful to some degree, but does have negative effect to it's performance if unnecessary effort is put on to our model. Additionally, the performance advantage may vary depending on the movements independent of the action inside the environment.



(a) Pitfall (b) Asteroids

Figure 4. Two environments with significant differences in performance improvement.(13% and 2% respectively) (a) has few uncontrollable obstacles in the environment. In contrast, (b) includes multiple objects that is not affected by the agent's action.

5. Discussion

5.1. Ablation

We investigate two ablations of our ABST-DIM model: ABST-DIM without local-local objective(L_{LL}), which only use global-local objective(L_{GL}) and Global-ABST-DIM, which only maximizes the mutual information between the global representations (similar in construction to PCL [24]). We report results from these ablations in Table 4 and Table 5 respectively. We see from the results in that Auxiliary Fully Connected(AFC) Discriminator for action prediction also improves the performance of both an ablation of ST-DIM that removes local-local objective(L_{LL}) and Global ST-DIM. This means our new approach could be expandable to any constrastive learning framework. Note that there should be a subsidiary labeled information for auxiliary learning. More specifically, in case of Qbert at Table 4 and Pong at Table 5, our proposed algorithm indicates bigger performance gain, increasing the performance 10% and

GAME	ST-DIM(baseline)	ST-DIM(GL)	ABST-DIM(GL) AFC(λ) 0.5	1	3	Max(vs ST-DIM(GL))
Berzerk	0.51	0.51	0.51	0.51	**0.53**	0.53(+0.02)
Pitfall	0.59	0.71	**0.73**	0.72	0.72	0.73(+0.02)
Pong	0.82	0.82	0.82	**0.83**	0.83	0.83(+0.01)
Qbert	0.72	0.65	**0.75**	0.73	0.72	0.75(+0.10)
Mean	0.66	0.67	0.70	0.70	0.70	**0.71**(+0.04)

Table 4. Comparison of F1 Scores for ST-DIM, Global-Local ST-DIM(an ablation of ST-DIM that removes local-local objective) and Global-Local-ABST-DIM

GAME	ST-DIM(baseline)	ST-DIM(GG)	ABST-DIM(GG) AFC(λ) 0.5	1	3	Max(vs ST-DIM(GL))
Berzerk	0.51	0.51	0.49	0.47	**0.52**	0.52(+0.01)
Pitfall	0.59	0.60	**0.64**	0.55	0.64	0.64(+0.04)
Pong	0.82	0.62	**0.70**	0.68	0.70	0.70(+0.08)
Qbert	0.72	0.59	0.58	0.60	**0.61**	0.61(+0.02)
Mean	0.66	0.58	0.60	0.58	0.61	**0.62**(+0.04)

Table 5. Comparison of F1 Scores for ST-DIM, Global ST-DIM and Global-ABST-DIM

METHOD	ST-DIM(GG)	ABST-DIM(GG) AFC(λ=1)
CLOCK	**0.66**	0.62
ENEMY_SCORE	**0.51**	0.39
ENEMY_X	**0.16**	0.15
ENEMY_Y	0.14	**0.16**
PLAYER_SCORE	0.51	**0.55**
PLAYER_X	0.18	**0.34**
PLAYER_Y	0.16	**0.21**
Mean	0.33	**0.35**

Table 6. Breakdown of F1 Scores for every state variable in Boxing for Global ST-DIM and Global-ABST-DIM

8% respectively. Based on this results, we could conclude that when the performance gap between baseline ST-DIM and the ablation methods is large, our new approach has a significant impact.

5.2. Capturing the position of Agent

As we can see in Table 6, ABST-DIM performs better at capturing the position of the agents than other methods. The F1 scores of PLAYER X and PLAYER Y of Global-ABST-DIM was improved by 16% and 5% respectively, while overall F1 scores are slightly improved by 2%. This is consistent with our intuition that action-predicting task is helpful to represent the agent's position.

6. Conclusions

This paper presents new algorithm(ABST-DIM) learning state representations by agent's action and mutual information across spatially and temporally distinct features. We tried three different action inference architectures and evaluated performance of each architecture in terms of F1 score over generative features. We discovered that for auxiliary objective, unnecessary complexity leads to negative effect in representation performance as the result favored the relatively simple solution, the AFC.

We then extensively evaluated the performance of ABST-DIM based on AFC, and in most cases, ABST-DIM shows improved performance. This emphasize that action between observations gives certain intuition on semantics of the observation itself. Also, auxiliary action inference is implementable to any Reinforcement Learning setting.

However, the performance is greatly influenced by the surrounding environment as well as by the agent's behavior. Our new model(ABST-DIM) did not achieve dramatic performance improvement when the game environment has many uncontrollable object. As a future research, we intend to explore robust negative sampling and contrastive learning methods.

References

[1] Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Yoshua Bengio, and R Devon Hjelm. Unsupervised state representation learning in atari. *NeurIPS*, 2010. 2, 3, 5, 6

[2] David Barber and Felix Agakov. The im algorithm: A variational approach to information maximization. *Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS'03*, page 201–208, 2003. 2

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. *Proceedings of the 35th International Conference on Machine Learning*, page 531–540, 2018. 2

[4] 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. 2, 5

[5] 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. 2

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018. 6

[7] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, page 2172–2180, 2016. 2

[8] Jongwook Choi, Yijie Guo, Marcin Moczulski, Junhyuk Oh, Neal Wu, Mohammad Norouzi, and Honglak Leer. Contingency-aware exploration in reinforcement learning. *Contingency-aware exploration in reinforcement learning*, 2018. 2

[9] Giuseppe Cuccu, Julian Togelius, and Philippe Cudré-Mauroux. Playing atari with six neurons. *International Conference on Autonomous Agents and Multiagent Systems*, 2019. 2

[10] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2061–2060, 2017. 6

[11] Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time. iv. *Communications on Pure and Applied Mathematics*, 4(36):183–212, 1983. 2

[12] Wuyang Duan. Learning state representations for robotic control: Information disentangling and multi-modal learning. *Master's thesis, Delft University of Technology*, 2017. 2

[13] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari SMorcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, and et al. Neural scene representation and rendering. *Science, 360(6394):1204–1210*, 2018. 2

[14] Deepak Pathak et al. Curiosity-driven exploration by self-supervised prediction. *ICML*, 2017. 2

[15] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *Advances in neural information processing systems*, pages 64–72, 2016. 2

[16] Klaus Greff, Raphaël Lopez Kaufmann, Rishab Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 2

[17] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, page 297–304, 2020. 2

[18] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in Neural Information Processing Systems*, page 2450–2462, 2018. 2

[19] Jansel Herrera-Gerena, Ramakrishnan Sundareswaran, John Just, Matthew Darr, and Ali Jannesari. Claws: Contrastive learning with hard attention and weak supervision. *Association for the Advancement of Artificial Intelligence*, 2021. 5

[20] 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. 2

[21] 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. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, page 1480–1490, 2017. 2

[22] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *International Conference on Learning Representations (ICLR)*, 2019. 2

[23] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *arXiv preprint arXiv:1709.01450*, 2017. 1

[24] AJ Hyvarinen and Hiroshi Morioka. Nonlinear ica of temporally dependent stationary sources. *Proceedings of Machine Learning Research*, 2017. 7

[25] Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. autonomous robots. *39(3):407–428*, 2015. 2

[26] 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. 2

[27] Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi:exploration with mutual information. *International Conference on Machine Learning*, page 3360–3369, 2019. 2

[28] Timothée Lesort, Natalia Díaz-Rodríguez, Jean Franois-Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 2018. 1

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence ´ Zitnick. Microsoft coco: Common objects in context. *ECCV*, 2014. 1

[30] Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. *arXiv preprint arXiv:1809.018122018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 2

[31] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, , and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. *ECCV*, 2018. 1

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *NIPS Deep Learning Workshop*, 2013. 6

[33] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. *Advances in neural information processing systems*, page 2863–2871, 2015. 2

[34] Ben Poole, Sherjil Ozair, Aäron Van den Oord, Alexander A Alemi, and George Tucker. On variational bounds of mutual information. *International Conference on Machine Learning*, 2019. 2

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fe. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 1

[36] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2, 6

[37] Herke van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. Stable reinforcement learning with autoencoders for tactile and visual data. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3928–3934, 2016. 2

[38] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018. 2

[39] 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. 2

[40] Zhao Yang, Song Bai, Li Zhang, and Philip H.S. Torr. Learn to interpret atari agents. *arXiv:1812.11276v2*, 2019. 4

[41] Guangxiang Zhu, Zhiao Huang, and Chongjie Zhang. Object-oriented dynamics predictor. *Guangxiang Zhu, Zhiao Huang, and Chongjie Zhang*, pages 9804–9815, 2018. 2