Object-Oriented Dynamics Predictor

Guangxiang Zhu, Chongjie Zhang
Institute for Interdisciplinary Information Sciences
Tsinghua University
Beijing, China
guangxiangzhu@outlook.com, zhangchongjie@gmail.com

Abstract

Generalization has been one of the major challenges for learning dynamics models in model-based reinforcement learning. However, previous work on action-conditioned dynamics prediction focuses on learning the pixel-level motion and thus does not generalize well to novel environments with different object layouts. In this paper, we present a novel object-oriented framework, called *object-oriented dynamics predictor* (OODP), which decomposes the environment into objects and predicts the dynamics of objects conditioned on both actions and object-to-object relations. It is an end-to-end neural network and can be trained in an unsupervised manner. To enable the generalization ability of dynamics learning, we design a novel CNN-based relation mechanism that is class-specific (rather than object-specific) and exploits the locality principle. Empirical results show that OODP significantly outperforms previous methods in terms of generalization over novel environments with various object layouts. OODP is able to learn from very few environments and accurately predict dynamics in a large number of unseen environments. In addition, OODP learns semantically and visually interpretable dynamics models.

1 Introduction

Recently, model-free deep reinforcement learning (DRL) has been extensively studied for automatically learning representations and decisions from visual observations to actions. Although such approach is able to achieve human-level control in games [1, 2, 3, 4, 5], it is not efficient enough and cannot generalize across different tasks. To improve sample-efficiency and enable generalization for tasks with different goals, model-based DRL approaches (e.g., [6, 7, 8]) are proposed to learn the environmental dynamics model and then plan or learn policies based on the learned model.

Existing work on learning action-conditioned dynamics models [9, 10, 11] has achieved significant progress, which learns dynamics models and achieves remarkable performance in training environments [9, 10], and further makes a step towards generalization over object appearances [11]. However, these models focus on learning pixel-level motions and thus their learned dynamics models does not generalize well to novel environments with different object layouts. Cognitive studies show that objects are the basic units of decomposing and understanding the world through natural human senses [12, 13, 14, 15], which indicates object-based models are useful for generalization [16, 17, 18].

In this paper, we develop a novel object-oriented framework, called *object-oriented dynamics predictor* (OODP). It is an end-to-end neural network and can be trained in an unsupervised manner. It follows an object-oriented paradigm, which decomposes the environment into objects and learns the object-level dynamic effects of actions conditioned on object-to-object relations. To enable the generalization ability of OODP's dynamics learning, we design a novel CNN-based relation mechanism that is class-specific (rather than object-specific) and exploits the locality principle. This mechanism induces neural networks to distinguish objects based on their appearances, as well as their effects on an agent's dynamics.

Empirical results show that OODP significantly outperforms previous methods in terms of generalization over novel environments with different object layouts. OODP is able to learn from very few environments and accurately predict the dynamics of objects in a large number of unseen environments. In addition, OODP learns semantically and visually interpretable dynamics models, and demonstrates robustness to some changes of object appearance.

2 Related Work

Action-conditioned dynamics learning approaches have been proposed for learning the dynamic effects of an agent's actions from raw visual observations and achieves remarkable performance in training environments [9, 10]. CDNA [11] further makes a step towards generalization over object appearances, and demonstrates its usage for model-based DRL [8]. However, these approaches focus on learning pixel-level motions and do not explicitly take object-to-object relations into consideration. As a result, they cannot generalize well across novel environments with different object layouts. An effective relation mechanism can encourage neural networks to focus attentions on the moving objects whose dynamics needs to be predicted, as well as the static objects (e.g., walls and ladders) that have an effect on the moving objects. The lack of such a mechanism also accounts for the fact that the composing masks in CDNA [11] are insensitive to static objects.

Relation-based nets have shown remarkable effectiveness to learn relations for physical reasoning [18, 19, 20, 21, 22]. However, they are not designed for action-conditioned dynamics learning. In addition, they have either manually encoded object representations [18, 19] or not demonstrated generalization across unseen environments with novel object layouts [19, 20, 21, 22]. In this paper, we design a novel CNN-based relation mechanism. First, this mechanism formulates a spatial distribution of a class of objects as a class-specific object mask, instead of representing an individual object by a vector, which allows relation nets to handle a vast number of object samples and distinguish objects by their specific dynamic properties. Second, we use neighborhood cropping and CNNs to exploit the locality principle of object interactions that commonly exists in the real world.

Object-oriented reinforcement learning has been extensively studied, whose paradigm is that the learning is based on object representations and the effects of actions are conditioned on object-to-object relations. For example, *relational MDPs* [16] and *Object-Oriented MDPs* [17] are proposed for efficient model-based planning or learning, which supports strong generalization. Kulkarni et al. [23] develop a model-free object-oriented learning algorithm to speed up classic Q-learning with compact state representations. However, these models require explicit encoding of the object representations [16, 17, 23] and relations [16, 17]. In contrast, our work aims at automatically learning object representations and object-to-object relations.

3 Object-Oriented Dynamics Predictor

To enable the generalization ability over object layouts for dynamics learning, we develop a novel unsupervised end-to-end neural network framework, called *Object-Oriented Dynamics Predictor* (OODP), which takes the video frames and agents' actions as input and learns the dynamics of objects conditioned on the actions and object-to-object relations. As shown in Figure 1, the OODP framework consists of three main components: Object Detector, Dynamics Net, and Background Splitter. Object Detector and Dynamic Net are designed for predicting the motions of dynamic objects. Object Detector learns to identify dynamic objects as well as static objects that have effects on the motions of dynamic objects. Dynamic Net predicts the motions of dynamics objects based on both actions of an agent and their interactions with other objects. Background Splitter extracts the background of the input image frame, which is not changing over time (e.g., walls, roads, and stairs). The extracted background will then be combined with the predicted motions of dynamic objects to predict the next frame. OODP assumes the environment is Markovian and deterministic, so it is possible to predict the next frame by the current frame and action.

OODP follows an object-oriented dynamics learning paradigm. It uses object masks to bridge object perception and dynamics learning and to form a tensor bottleneck, which allows only the object-level information to flow through. Each object mask has its own dynamics learner, which forces the perception network to act as a detector for the object of interest as well as a classifier for distinguishing which object has a specific dynamics. In addition, we add an entropy loss to reduce

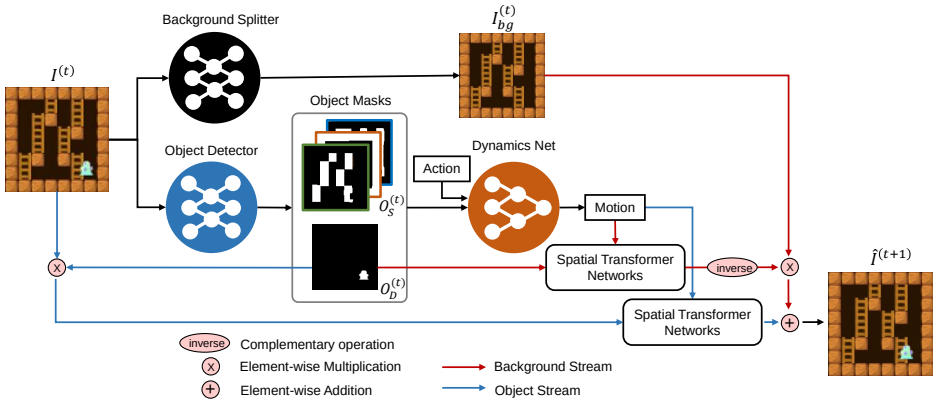

Figure 1: Overall framework of OODP.

the uncertainty of object masks, thus encouraging attention on key parts and learning invariance to irrelevances.

3.1 Object Detector

Object Detector decomposes the input image into objects and represents the spatial distributions of the objects by object masks $M_O^{(t)} \in [0,1]^{H\times W\times n_O}$, where H and W denote the height and width of the input image $I^{(t)} \in \mathbb{R}^{H\times W\times 3}$, n_O denotes the total number of object masks, and t denotes the current time step. To simplify our model, we group the objects (denoted as O) into static and dynamic objects (denoted as S and D) so that we only need to focus on predicting the motions of the dynamic objects. The amount and variety of static objects are usually greater than those of dynamic objects. Thus, we use one static mask $M_{S_i}^{(t)}$ $(1 \le i \le n_S$, where n_S denotes the class number of static objects) to represent each class of static objects and one dynamic mask $M_{D_j}^{(t)}$ $(1 \le j \le n_D$, where n_D denotes the individual number of dynamic objects) to describe each individual dynamic object. Specifically, the entry $M_{S_i}^{(t)}(u,v)$ $(1 \le u \le H, 1 \le v \le W)$ indicates the probability that the pixel $I^{(t)}(u,v)$ belongs to the i-th class of static objects, while $M_{D_j}^{(t)}(u,v)$ indicates the probability that the pixel $I^{(t)}(u,v)$ belongs to the j-th dynamic object.

Figure 2 depicts the architecture of Object Detector. As shown in the figure, the pixels of the input image go through different CNNs to obtain different object masks. There are totally n_O CNNs owning the same architecture but not sharing weights. The architecture details of these CNNs can be found in Appendix. Then, the output layers of these CNNs are concatenated with each other across channels and a pixel-wise softmax is applied on the concatenated feature map $F \in \mathbb{R}^{H\times W\times n_O}$. Let $f(u,v,c)$ denote the value at position (u,v) in the c-th channel of F. The entry $M_{O_c}(u,v)$ of the c-th object mask which represents the probability that the pixel $I(u,v)$ of the input image belongs to the c-th object O_c, can be calculated as,

$$M_{O_c}(u,v) = p\left(O_c|I(u,v)\right) = \frac{e^{f(u,v,c)}}{\sum_i^{n_O} e^{f(u,v,i)}}.$$

To reduce the uncertainty of the affiliation of each pixel $I(u,v)$ and encourage the object masks to obtain more discrete attention distributions, we introduce a pixel-wise entropy loss to limit the entropy of the object masks, which is defined as,

$$\mathcal{L}_{\text{entropy}} = \sum_{u,v}\sum_{c}^{n_O} -p\left(O_c|I(u,v)\right)\log p\left(O_c|I(u,v)\right).$$

3.2 Dynamics Net

Dynamics Net aims at learning the motion of each dynamic object conditioned on actions and object-to-object relations. Its architecture is illustrated in Figure 3. In order to improve the computational

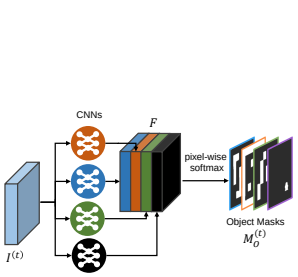

Figure 2: Architectures of Object Detector.

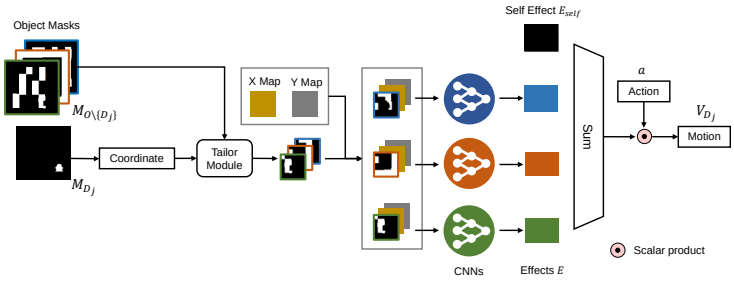

Figure 3: Architecture of Dynamics Net. We illustrate predicting the motion of M_{D_j} as an example. $O\backslash\{D_j\}$ denotes O excluding D_j.

efficiency and generalization ability, the Dynamics Net architecture incorporates a tailor module to exploit the locality principle and employs CNNs to learn the effects of object-to-object relations on the motions of dynamic objects. As the locality principle commonly exists in object-to-object relations in the real world, the tailor module enables the inference on the dynamics of objects focusing on the relations with neighbour objects. Specifically, it crops a "horizon" window of size w from the object masks centered on the position of the dynamic object whose motion is being predicted, where w indicates the maximum effective range of object-to-object relations. The tailored local objects are then fed into the successive network layers to reason their effects. Unlike most previous work which uses fully connected networks for identifying relations [18, 19, 20, 21, 22], our dynamics inference employs CNNs. This is because CNNs provide a mechanism of strongly responding to spatially local patterns, and the multiple receptive fields in CNNs are capable of dealing with complex spatial relations expressed in distribution masks.

To demonstrate the detailed pipeline in Dynamics Net (Figure 3), we take as an example the computation of the predicted motion of the j-th dynamic object D_j. First, we describe the cropping process of the tailor module, which crops the object masks near to the dynamic object D_j.

For each dynamic object D_j, its position $(\bar{u}_{D_j}, \bar{v}_{D_j})$ is defined as the expected location of its object mask M_{D_j}, where $\bar{u}_{D_j} = \left(\sum_u^H \sum_v^W M_{D_j}(u,v)\right)^{-1} \sum_u^H \sum_v^W u \cdot M_{D_j}(u,v)$, $\bar{v}_{D_j} = \left(\sum_u^H \sum_v^W M_{D_j}(u,v)\right)^{-1} \sum_u^H \sum_v^W v \cdot M_{D_j}(u,v)$. The "horizon" window of size w centered on $(\bar{u}_{D_j}, \bar{v}_{D_j})$ is written as $B_w = \{(u,v) : \bar{u}_{D_j} - w/2 \le u \le \bar{u}_{D_j} + w/2, \bar{v}_{D_j} - w/2 \le v \le \bar{v}_{D_j} + w/2\}$. The tailor module receives B_w and performs cropping on other object masks excluding M_{D_j}, that is, $M_{O\backslash\{D_j\}}$. This cropping process can be realized by bilinear sampling [24], which is a sub-differentiable sampling approach. Taking cropping M_{O_1} as an example, the pairwise transformation of sampling grid is $(u_2, v_2) = (u_1 + \bar{u}_{D_j} - w/2, v_1 + \bar{v}_{D_j} - w/2)$, where (u_1, v_1) are coordinates of the cropped object mask C_{O_1} and $(u_2, v_2) \in B_w$ are coordinates of the original object mask M_{O_1}. Then applying bilinear sampling kernel on this grid can compute the cropped object mask C_{O_1},

$$C_{O_1}(u_1, v_1) = \sum_u^H \sum_v^W \Big(M_{O_1}(u,v) \cdot \max(0, 1 - |u_2 - u|) \max(0, 1 - |v_2 - v|) \Big), \qquad (1)$$

Note that the gradients wrt $(\bar{u}_{D_j}, \bar{v}_{D_j})$ are frozen to force the cropping module focus on what to crop rather than where to crop, which is different to the vanilla bilinear sampling [24].

Then, each cropped object mask C_{O_i} is concatenated with the constant x-coordinate and y-coordinate meshgrid map, which makes networks more sensitive to the spatial information. The concatenated map is fed into its own CNNs to learn the effect of i-th object on j-th dynamic object $E(O_i, D_j) \in \mathbb{R}^{2 \times n_a}$, where n_a represents the number of actions. There are totally $(n_O - 1) \times n_D$ CNNs for $(n_O - 1) \times n_D$ pairs of objects. Different CNNs working for different objects forces the object mask to act as a classifier for distinguishing each object with the specific dynamics. To predict the motion vector $V_{D_j}^{(t)} \in \mathbb{R}^2$ for dynamic object D_j, all the object effects and a self effect $E_{\text{self}}(D_j) \in \mathbb{R}^{2 \times n_a}$ representing the natural bias are summed and multiplied by the one-hot coding of action $a^{(t)} \in \{0,1\}^{n_a}$, that is,

$$V_{D_j}^{(t)} = \Big(\big(\sum_{O_i \in O\backslash\{D_j\}} E(O_i, D_j) \big) + E_{\text{self}}(D_j) \Big) \cdot a^{(t)}.$$

In addition to the conventional prediction error $\mathcal{L}_{\text{prediction}}$ (described in Section 3.4), here we introduce a regression loss to guide the optimization of $M_O^{(t)}$ and $V_D^{(t)}$, which serves as a highway to provide early feedback before reconstructing images. This regression loss is defined as follows,

$$\mathcal{L}_{\text{highway}} = \sum_j^{n_D} \left\| (\bar{u}_{D_j}, \bar{v}_{D_j})^{(t)} + V_{D_j}^{(t)} - (\bar{u}_{D_j}, \bar{v}_{D_j})^{(t+1)} \right\|_2^2.$$

3.3 Background Splitter

To extract time-invariant background, we construct a Background Splitter with neural networks. The Background Splitter takes the form of a traditional encoder-decoder structure. The encoder alternates convolutions [25] and Rectified Linear Units (ReLUs) [26] followed by two fully-connected layers, while the decoder alternates deconvolutions [27] and ReLUs. To avoid losing information in pooling layers, we replace all the pooling layers by convolutional layers with increased stride [28, 29]. Further details of the architecture of Background Splitter can be found in Appendix.

Background Splitter takes the current frame $I^{(t)} \in \mathbb{R}^{H \times W \times 3}$ as input and produces the background image $I_{\text{bg}}^{(t)} \in \mathbb{R}^{H \times W \times 3}$, whose pixels remain unchanged over times. We use $\mathcal{L}_{\text{background}}$ here to force the network to satisfy such a property of time invariance, given by $\mathcal{L}_{\text{background}} = \left\| I_{\text{bg}}^{(t+1)} - I_{\text{bg}}^{(t)} \right\|_2^2$.

3.4 Converging Streams

Finally, two streams of pixels are converged to produce the prediction of the next frame. One stream carries the pixels of dynamic objects which can be obtained by transforming the masked pixels of dynamic objects from the current frame. The other stream provides the rest pixels generated by Background Splitter.

Spatial Transformer Network (STN) [24] provides neural networks capability of spatial transformation. In the first stream, a STN accepts the learned motion vectors V and performs spatial transforms (denoted by $\text{STN}(*, V)$) on dynamic pixels. The bilinear sampling (similar as Equation 1) is also used in STN to compute the transformed pixels in a sub-differentiable manner. The difference is that, we allow the gradients of loss backpropagate to the object masks as well as the motion vectors. Then, we obtain the pixel frame $\hat{I}_1^{(t+1)}$ of dynamic objects in the next frame $\hat{I}^{(t+1)}$, that is, $\hat{I}_1^{(t+1)} = \sum_j^{n_D} \text{STN}(M_{D_j}^{(t)} \cdot I^{(t)}, V_{D_j}^{(t)})$, where $\cdot$ denotes element-wise multiplication. The other stream aims at computing the rest pixels $\hat{I}_2^{(t+1)}$ of $\hat{I}^{(t+1)}$, that is, $\hat{I}_2^{(t+1)} = \big(1 - \sum_j^{n_D} \text{STN}(M_{D_j}^{(t)}, V_{D_j}^{(t)})\big) \cdot I_{\text{bg}}^{(t)}$. Thus, the output end of OODP, $\hat{I}^{(t+1)}$, is calculated by,

$$\hat{I}^{(t+1)} = \hat{I}_1^{(t+1)} + \hat{I}_2^{(t+1)} = \sum_j^{n_D} \text{STN}(M_{D_j}^{(t)} \cdot I^{(t)}, V_{D_j}^{(t)}) + \big(1 - \sum_j^{n_D} \text{STN}(M_{D_j}^{(t)}, V_{D_j}^{(t)})\big) \cdot I_{\text{bg}}^{(t)}.$$

We use l_2 pixel loss to restrain image prediction error, which is given by, $\mathcal{L}_{\text{prediction}} = \left\| \hat{I}^{(t+1)} - I^{(t+1)} \right\|_2^2$. We also add a similar l_2 pixel loss for reconstructing the current image, that is,

$$\mathcal{L}_{\text{reconstruction}} = \left\| \sum_j^{n_D} M_{D_j}^{(t)} \cdot I^{(t)} + \big(1 - \sum_j^{n_D} M_{D_j}^{(t)}\big) \cdot I_{\text{bg}}^{(t)} - I^{(t)} \right\|_2^2.$$

In addition, we add a loss $\mathcal{L}_{\text{consistency}}$ to link the visual perception and dynamics prediction of the objects, which enables learning object by integrating vision and interaction,

$$\mathcal{L}_{\text{consistency}} = \sum_j^{n_D} \left\| M_{D_j}^{(t+1)} - \text{STN}\big(M_{D_j}^{(t)}, V_{D_j}^{(t)}\big) \right\|_2^2.$$

3.5 Training Procedure

OODP is trained by the following loss given by combining the previous losses with different weights,

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{highway}} + \lambda_p \mathcal{L}_{\text{prediction}} + \lambda_e \mathcal{L}_{\text{entropy}} + \lambda_r \mathcal{L}_{\text{reconstruction}} + \lambda_c \mathcal{L}_{\text{consistency}} + \lambda_{bg} \mathcal{L}_{\text{background}}$$

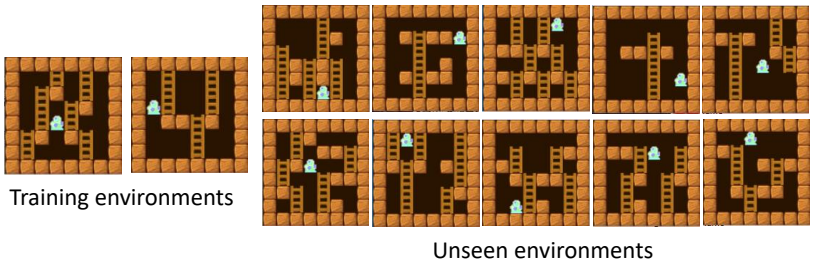

Figure 4: An example of 2-to-10 generalization problem.

Considering that signals derived from foreground detection may benefit Object Detector to produce more accurate object masks, we use the simplest unsupervised foreground detection approach [30] to calculate a rough proposal dynamic region and then add a l_2 loss to encourage the dynamic object masks to concentrate more attentions on this region, that is, $\mathcal{L}_{\text{proposal}} = \left\| \left(\sum_j^{n_D} M_{D_j}^{(t)} \right) - M_{\text{proposal}}^{(t)} \right\|_2^2$, where $M_{\text{proposal}}^{(t)}$ represents the proposal dynamic region. With this additional loss, the augmented version of OODP is called OODP+p.

4 Experiments

4.1 Experiment Setting

We evaluate our model on *Monster Kong* from the Pygame Learning Environment [31], which offers various scenes for testing generalization abilities across object layouts (e.g., different number and spatial arrangement of objects). Across different scenes, the same underlying physics engine that simulates the real-world dynamics mechanism is shared. For example, in each scene, an agent can move upward using a ladder, it will be stuck when hitting the walls, and it will fall in free space. The agent explores various environments with a random policy over actions including *up*, *down*, *left*, *right*, and *noop* and its gained experiences are used for learning dynamics.

To evaluate the generalization ability, we compare OODP and OODP+p with state-of-the-art action-conditioned dynamics learning approaches, AC Model [9], and CDNA [11]. We evaluate all models in k-to-m generalization problems (Figure 4), where they learn dynamics with k different training environments and are then evaluated in m different unseen testing environments with different object layouts. In this paper, we use m = 10 and k = 1, 2, 3, 4, and 5, respectively. The smaller the value k, the more difficult the generalization problem. In this setting, truly achieving generalization to new scenes requires learners' full understanding of the object-level abstraction, object relationships and dynamics mechanism behind the images, which is quiet different from the conventional video prediction task and crucially challenging for the existing learning models. In addition, we will investigate whether OODP can learn semantically and visually interpretable knowledge and is robustness to some changes of object appearance.

4.2 Generalization of Learned Models

To demonstrate the generalization ability, we evaluate the prediction accuracy of the learned dynamics model in unseen environments with novel object layouts without re-training. Table 1 shows the performance of all models on predicting the dynamics of the agent, where n-error accuracy is defined as the proportion that the difference between the predicted and ground-true agent locations is less than n pixel ($n = 0, 1, 2$).

From Table 1, we can see our model significantly outperforms the previous methods under all circumstances. This demonstrates our object-oriented approach is beneficial for the generalization over object layouts. As expected, as the number of training environments increases, the learned object dynamics can generalize more easily over new environments and thus the accuracy of dynamics prediction tends to grow. OODP achieves reasonable performance with 0.86 0-error accuracy only trained in 3 environments, while the other methods fail to get satisfactory scores (about 0.2). We observe that the AC Model achieves extremely high accuracy in training environments but cannot

	Models	Training environments					Unseen environments				
		1-10	2-10	3-10	4-10	5-10	1-10	2-10	3-10	4-10	5-10
0-error accuracy	OODP+p	0.90	0.94	0.92	0.93	0.93	**0.32**	**0.78**	0.73	0.79	0.82
	OODP	0.96	0.98	0.98	0.98	0.97	0.22	**0.78**	**0.86**	**0.90**	**0.95**
	AC Model	0.99	0.99	0.99	0.99	0.99	0.01	0.17	0.22	0.44	0.70
	CDNA	0.20	0.13	0.14	0.19	0.17	0.33	0.18	0.20	0.25	0.19
1-error accuracy	OODP+p	0.98	0.98	0.99	0.98	0.98	**0.71**	0.90	0.90	0.94	0.95
	OODP	0.98	0.98	0.99	0.99	0.99	0.61	**0.91**	**0.94**	**0.96**	**0.97**
	AC Model	0.99	0.99	0.99	0.99	0.99	0.01	0.31	0.31	0.57	0.77
	CDNA	0.30	0.29	0.30	0.33	0.30	0.53	0.49	0.47	0.52	0.55
2-error accuracy	OODP+p	0.99	0.99	0.99	0.99	0.99	**0.87**	**0.96**	0.96	**0.98**	**0.99**
	OODP	0.99	0.99	0.99	0.99	0.99	0.82	0.94	**0.97**	**0.98**	0.98
	AC Model	0.99	0.99	0.99	0.99	0.99	0.02	0.37	0.34	0.64	0.80
	CDNA	0.36	0.44	0.47	0.45	0.45	0.56	0.55	0.56	0.62	0.62

Table 1: Accuracy of the dynamics prediction. k-m means the k-to-m generalization problem.

make accurate predictions in novel environments, which implies it overfits the training environments severely. This is partly because the AC Model only performs video prediction at the pixel level and learns few object-level knowledge. Though CDNA includes object concepts in their model, it still performs pixel-level motion prediction and does not consider object-to-object relations. As a result, CDNA also fails to achieve accurate predictions in unseen environments with novel object layouts (As the tuning of hyper parameters does not improve the prediction performance, we use the default settings here).

4.3 Interpretability of Learned Knowledge

Interpretable deep learning has always been a significant but vitally hard topic [32, 33, 34]. Unlike previous video prediction frameworks [9, 11, 35, 36, 37, 38, 22], most of which use neural networks with uninterpretable hidden layers, our model has informative and meaningful intermediate layers containing the object-level representations and dynamics.

To interpret the intermediate representations learned by OODP, we illustrate its object masks in unseen environments, as shown in Figure 5. Intriguingly, the learned dynamic object masks accurately capture the moving agents, and the static masks successfully detect the ladders, walls and free space that lead to different action conditioned dynamics of the agent. Each object mask includes one class of objects, which implies that the common characteristics of this class are learned and the knowledge that links visual features and dynamics properties is gained. While the learned masks are not as fine as those derived from the supervised image segmentation, they clearly demonstrate visually interpretable representations in the domain of unsupervised dynamics learning.

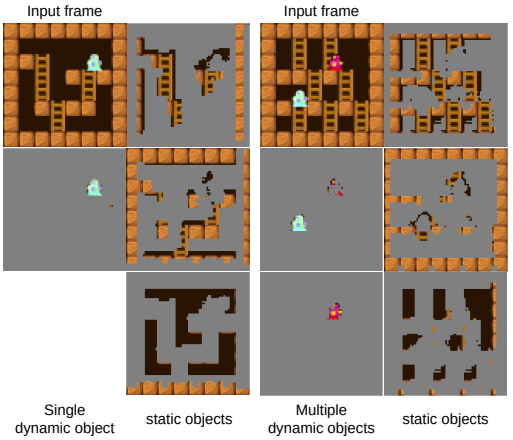

Figure 5: Visualization of the masked images in unseen environments with single dynamic object (left) and multiple dynamics objects (right). To demonstrate the learned attentions of object masks, the raw input images are multiplied by binarized object masks.

To interpret the learned object dynamics behind frame prediction, we evaluate the root-mean-square errors (RMSEs) between the predicted and ground-truth motions. Table 2 shows the RMSEs averaged over 10000 samples. From Table 2, we can observe that motions predicted by OODP are very

	Models	Number of training envs				
		1	2	3	4	5
Training envs	OODP+p	0.28	0.24	0.23	0.23	0.23
	OODP	0.18	0.17	0.19	0.14	0.15
Unseen envs	OODP+p	1.04	0.52	0.51	0.43	0.40
	OODP	1.09	0.53	0.38	0.35	0.29

Table 2: RMSEs between predicted and ground-truth motions. The unit of measure is pixel.

	Object appearance						
	S0	S1	S2	S3	S4	S5	S6
Acc	0.94	0.92	0.94	0.94	0.92	0.88	0.93
RMSE	0.29	0.35	0.31	0.28	0.31	0.40	0.30

Table 3: The performance (accuracy and RMSE in 5-to-10 generalization problem) of OODP in novel environments with different object layouts and appearances.

accurate, with the RMSE close or below one pixel, in both training and unseen environments. Such a small error is visually indistinguishable since it is less than the resolution of the input video frame (1 pixel). As expected, as the number of training environments increases, this prediction error rapidly descends. Further, we also provide some intuitive prediction examples (see Appendix) and a video (`https://goo.gl/BTL2wH`) for better perceptual understanding of the prediction performance.

These interpretable intermediates demystify why OODP is able to generalize across novel environments with various object layouts. While the visual perceptions of novel environments are quite different, the underlying physical mechanism based on object relations keeps invariant. As shown in Figure 5, OODP learns to decompose a novel scene into understandable objects and thus can reuse object-level knowledge (features and relations) acquired from training environments to predict the effects of actions.

4.4 Robustness to changes of object appearance

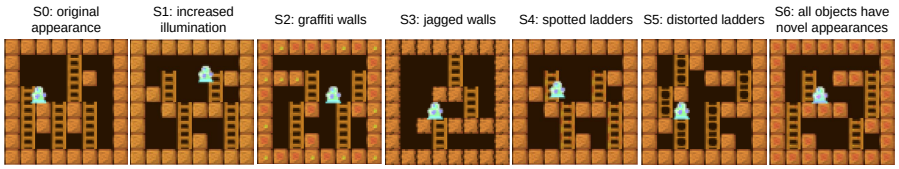

Figure 6: Illustration of the configurations of the novel testing environments. Compared to the training environments, the testing ones have different object layouts (S0-S6), and their objects have some appearance differences (S1-S6).

To demonstrate the robustness to object appearances, we evaluate the generalization performance of OODP in testing environments including objects with appearance differences from those in training environments, as shown in Figure 6. As shown in Table 3, OODP provides still high prediction performance in all these testing environments, which indicates that it can still generalize to novel object layouts even when the object appearances have some differences. This robustness is partly because the Object Detector in OODP employs CNNs that are capable of learning essential patterns over appearances. Furthermore, we provide the learned masks (see Appendix) and a video (`https://goo.gl/ovupdn`) to show our results on S2 environments.

5 Conclusion

In this paper, we present an object-oriented end-to-end neural network framework. This framework is able to learn object dynamics conditioned on both actions and object-to-object relations in an unsupervised manner. Its learned dynamics model exhibits strong generalization and interpretability. In addition, our framework demonstrates that object perception and dynamics can be mutually learned and it reveals a promising way to learn object-level knowledge by integrating both vision and interaction. We make one of the first steps in investigating whether object-oriented modeling is beneficial for generalization and interpretability in deep reinforcement learning. Our future work includes extending our model to partial-observation settings, long-term predictions, and domains

with more complex transformations, and developing model-based policy search or planning methods utilizing our learned dynamics model.

References

[1] 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.

[2] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2016.

[3] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[4] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

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

[6] Sergey Levine and Vladlen Koltun. Guided policy search. In *International Conference on Machine Learning*, pages 1–9, 2013.

[7] Silvia Chiappa, Sébastien Racaniere, Daan Wierstra, and Shakir Mohamed. Recurrent environment simulators. *International Conference on Learning Representations*, 2017.

[8] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2786–2793. IEEE, 2017.

[9] 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.

[10] 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.

[11] 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.

[12] Jean Piaget. Piaget's theory. 1970.

[13] Renee Baillargeon, Elizabeth S Spelke, and Stanley Wasserman. Object permanence in five-month-old infants. *Cognition*, 20(3):191–208, 1985.

[14] Renee Baillargeon. Object permanence in $3 1/2$-and $4 1/2$-month-old infants. *Developmental psychology*, 23(5):655, 1987.

[15] Elizabeth S Spelke. Where perceiving ends and thinking begins: The apprehension of objects in infancy. In *Perceptual development in infancy*, pages 209–246. Psychology Press, 2013.

[16] Carlos Guestrin, Daphne Koller, Chris Gearhart, and Neal Kanodia. Generalizing plans to new environments in relational mdps. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pages 1003–1010. Morgan Kaufmann Publishers Inc., 2003.

[17] Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 240–247. ACM, 2008.

[18] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

[19] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, pages 4502–4510, 2016.

[20] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks. In *Advances in Neural Information Processing Systems*, pages 4540–4548, 2017.

[21] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4974–4983, 2017.

[22] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, pages 152–163, 2017.

[23] Luis C Cobo, Charles L Isbell, and Andrea L Thomaz. Object focused q-learning for autonomous agents. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1061–1068. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[24] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015.

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[26] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[27] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010.

[28] Viren Jain, Joseph F Murray, Fabian Roth, Srinivas Turaga, Valentin Zhigulin, Kevin L Briggman, Moritz N Helmstaedter, Winfried Denk, and H Sebastian Seung. Supervised learning of image restoration with convolutional networks. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[29] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[30] BPL Lo and SA Velastin. Automatic congestion detection system for underground platforms. In *Intelligent Multimedia, Video and Speech Processing, 2001. Proceedings of 2001 International Symposium on*, pages 158–161. IEEE, 2001.

[31] Norman Tasfi. Pygame learning environment. `https://github.com/ntasfi/PyGame-Learning-Environment`, 2016.

[32] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. *arXiv preprint arXiv:1710.00935*, 2017.

[33] Quan-shi Zhang and Song-Chun Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

[34] Quanshi Zhang, Yu Yang, Ying Nian Wu, and Song-Chun Zhu. Interpreting cnns via decision trees. *arXiv preprint arXiv:1802.00121*, 2018.

[35] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852, 2015.

[36] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *International Conference on Learning Representations*, 2016.

[37] William Lotter, Gabriel Kreiman, and David Cox. Unsupervised learning of visual structure using predictive generative networks. *Computer Science*, 2015.

[38] Marc Aurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *Eprint Arxiv*, 2014.

[39] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456, 2015.

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

6 Appendix

6.1 Additional results of OODP

We supplement the learned masks (for Section 4.4 in the main body) in the unseen environments with novel object layouts and object appearance S2, which is shown in Figure 7. In addition, we further provide intuitive results of our performance on learned dynamics for perceptual understanding. Figure 8, 9 and 10 depict representative cases of our predictions in training and novel environments.

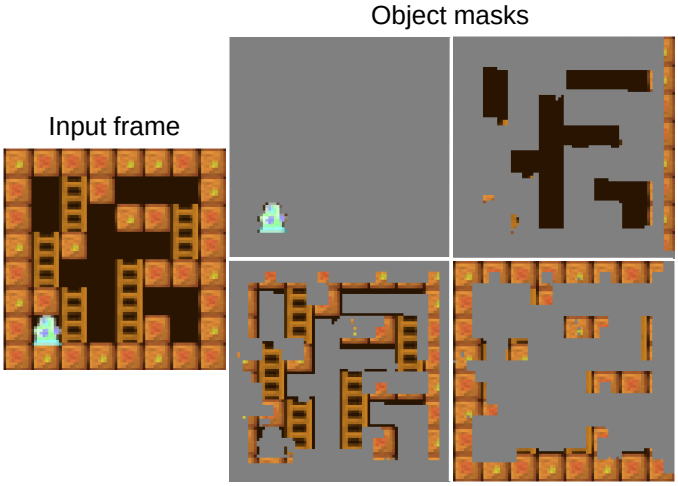

Figure 7: Visualization of the learned masked images in S2 environments.

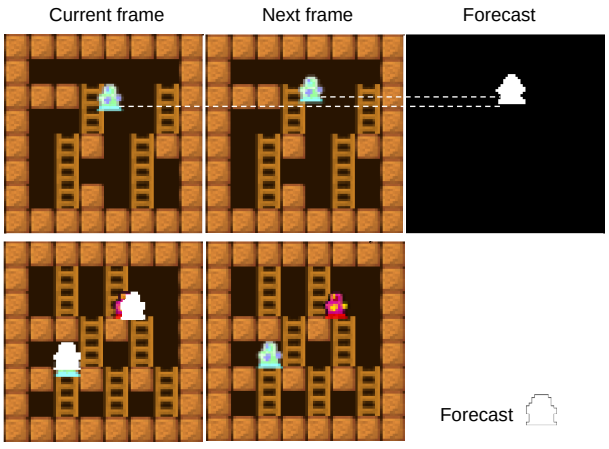

Figure 8: Cases to illustrate the performance of our model in novel environments including single (top) and multiple (bottom) dynamic objects.

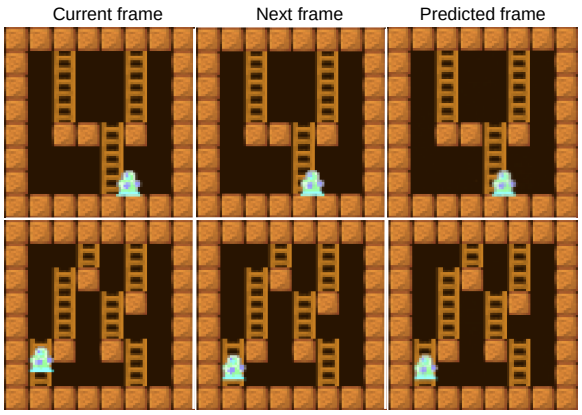

Figure 9: Two examples of predictions in training environments including one dynamic object.

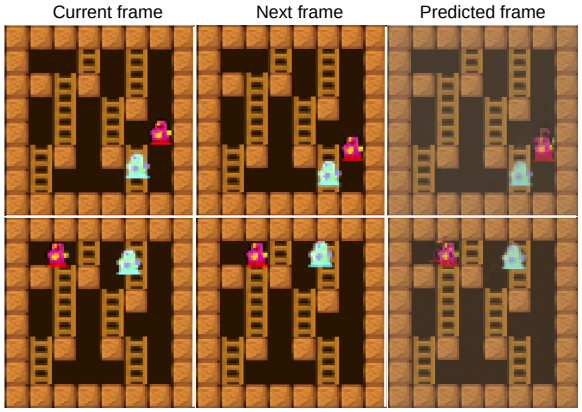

Figure 10: Two examples of predictions in training environments including multiple dynamic and static objects.

6.2 Validation on greater maximum of object masks

OODP does not require the actual number of objects in an environment, but needs to set a maximum number. When they does not match, some learned object masks may be redundant, which does not affect the accuracy of predictions. We conduct an experiment to confirm this. As shown in Figure 11, when the maximum number is larger than the actual number, some learned object masks are redundant. Walls are detected by the two masks shown in the top right corner of the figure, while ladders are detected by the two masks in the bottom right corner. Furthermore, the accuracies of dynamics prediction are similar when upper bounds of object masks are different (0.840 and 0.842 for the maximum number 3 and 5, respectively).

6.3 Architecture details

6.3.1 Background Splitter

As shown in Figure 12, the Background Splitter takes the form of a traditional encoder-decoder structure. The encoder alternates convolutions and ReLUs followed by two fully connected layers, while the decoder alternates deconvolutions and ReLUs. For all the convolutions and deconvolutions, the kernel size, stride, number of channels are 3, 2, and 64, respectively. The dimension of the hidden layer between the encoder and the decoder is 128. When OODP is trained in a large number of environments, convolutions with more channels (e.g., 64 or 128 channels) are needed to improve

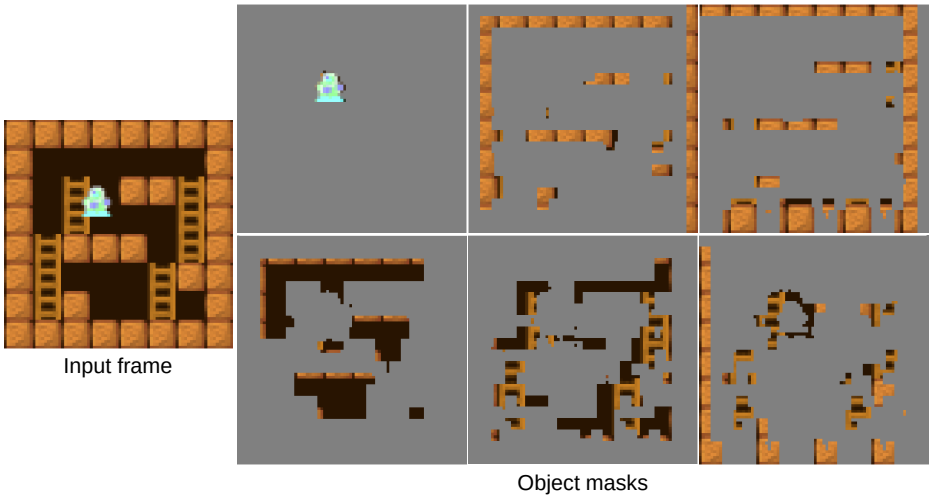

Figure 11: Visualization of the masked images in unseen environments with the maximum number of object masks equal to 5.

the expressiveness of Background Splitter. In addition, we replace the ReLU layer after the last deconvolution with tanh to output values ranged from -1 to 1.

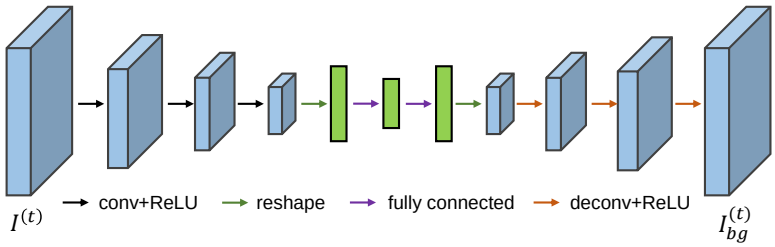

Figure 12: Architecture of Background Splitter.

6.3.2 CNNs in Object Detector and Dynamics Net

Figure 13 illustrates the architecture of the CNNs in Object Detector. Denote $Conv(F, K, S)$ as the convolutional layer with the number of filters F, kernel size K and stride S. Let $R()$ and $BN()$ denote the ReLU layer and batch normalization layer [39]. The 5 convolutional layers in the figure can be indicated as $R(BN(Conv(64, 5, 2)))$, $R(BN(Conv(64, 3, 2)))$, $R(BN(Conv(64, 3, 1)))$, $R(BN(Conv(32, 1, 1)))$, and $R(BN(Conv(1, 3, 1)))$, respectively.

The CNNs in Dynamics Net are connected in the order: $R(BN(Conv(16, 3, 2)))$, $R(BN(Conv(32, 3, 2)))$, $R(BN(Conv(64, 3, 2)))$, and $R(BN(Conv(128, 3, 2)))$. The last convolutional layer is reshaped and fully connected by the 128-dimensional hidden layer and the 2-dimensional output layer successively.

6.4 Implementation details

For OODP, OODP+p, and all baseline models, the training images are collected by an agent with random policy exploring the environment and then the changed and changeless images are balanced via sampling. The images are down-sampled to size $80 \times 80 \times 3$. The parameters for training DDOP and OODP+p are listed as follows:

- In OODP, the weights for $\mathcal{L}_{\text{prediction}}$, $\mathcal{L}_{\text{entropy}}$, $\mathcal{L}_{\text{reconstruction}}$, $\mathcal{L}_{\text{consistency}}$, and $\mathcal{L}_{\text{background}}$ are 100, 0.1, 1, 1, and 1, respectively. While in OODP+p, we can remove the auxiliary losses

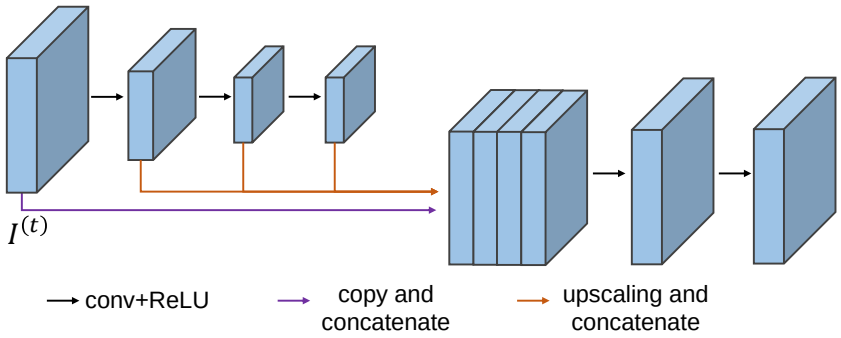

Figure 13: Architecture of the CNNs in Object Detector.

($\mathcal{L}_{\text{reconstruction}}$, $\mathcal{L}_{\text{consistency}}$, and $\mathcal{L}_{\text{background}}$) that have the similar functions to $\mathcal{L}_{\text{proposal}}$, and the weights for $\mathcal{L}_{\text{prediction}}$, $\mathcal{L}_{\text{entropy}}$, and $\mathcal{L}_{\text{proposal}}$ are 10, 1, and 1, respectively. In addition, all the l_2 losses are divided by HW to keep invariance to the image size.

- Bath size is 16 and the maximum number of training steps is set to be 1×10^6. The size of the horizon window w is 33.
- The optimizer is Adam [40] with learing rate 1×10^{-4}.