Object-Dreamer

Object-detection Priors with DreamerV2

Vadim Kudlay, Shreyas Sundara-Raman, Anoop Reddi

21 December 2021

Abstract

'Attention' and scene segmentation are key components of both visual systems in neuroscience and convolution-based architectures in deep learning, that bring saliency to the most important or relevant features of visual inputs. To this end, this paper aims to explore the effects of incorporating different forms of object-detection as a prior to the DreamerV2 architecture, on the ATARI benchmark performance; we hypothesize this might support DreamerV2 via generalization and faster convergence.

The work explores unsupervised object-detection towards this goal, in the ATARI observation space, using spatial attention and decomposition (SPACE) and weakly-supervised deep detection networks (WSDDNs); we attempt to integrate these (end-to-end) pipelines or their predicted detections within the DreamerV2 architecture. Our results show that the architecture doesn't significantly benefit from our attempted use of external object-detection, which is consistent with some previous findings.

1 Introduction

Biomimicry and the bias-variance (model complexity) tradeoff lie at the heart of all deep learning approaches. High bias in artificial intelligence and deep learning architectures can sometimes be useful in reducing the variational complexity of large inputs to the *most relevant* features, thereby providing the model with a useful prior that supplements learning and generalization. Multiple previous works make use of PCA or other dimensionality reduction mechanisms to enhance a model's generalizability and learning.

In the spirit of re-integrating the separated 'verticals' of current approaches in AI, we find a natural intersection between computer vision (CV) and reinforcement learning (RL) inspired by the neuroscientific perspective on human/primate vision. Previous work by Papale et. al. has found evidence that non-human primates exhibit foreground-background segmentation during passive visual processing; the data presented in this work highlights that passive

viewing involves the enhancement of the foreground and, at the same time, a suppression of the background visual stimulus – suggesting that biological vision undergoes a preferential visual process [11].

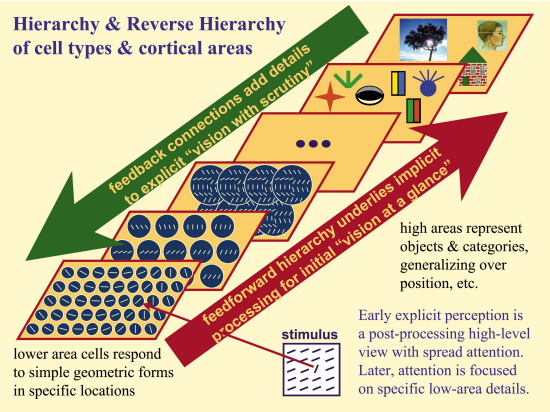

Figure 1: Classical Hierarchy and Reverse Hierarchy of visual system [6]

This raises the question of whether an 'attention based' approach that performs preferential selection and/or foreground-background segmentation might serve as an effective prior (to enhance learning speed or quality) for an agent observing in the visual domain. Our approach particularly relies on 'DreamerV2' as a RL baseline that samples visual inputs into a latent space (world model) and propagates this world model over time-steps [4]. We attempt to inject an object-detection scheme into this architecture by processing the ATARI-image input into DreamerV2 (before the encoder) during the refinement of the 'world model' through observations.

DreamerV2 is of particular interest, since the approach employs model-based learning using an internal '*discrete world model*'. This creates an active and recurrent representation of the environment (visual observation or stimulus) that can be modified using object-detection. Additionally, learning to generalize across ATARI games is complex; many previous attempts (DQN [9] or DreamerV1 [3]) achieved sub-human performance, with DreamerV2 being the *most recent* model-based approach to surpass human performance on the Atari benchmark. To explore the influence of attention and object-detection (CV) on learning in the RL setting, we acquired a baseline *mean reward* and *number of steps* in the `MsPacman-v0` (Pacman) environment in DreamerV2. We collected and compared the same metrics against this baseline after incorporating different object-detection pipelines into DreamerV2 – namely SPACE [8] and WSDDN [2]. Our in-going hypothesis proposed that object detection would parse the most important features of the input signal, thereby potentially allowing DreamerV2 to process the most relevant information, leading to faster converge and reward maximization. The results of this work (however) show

limited success with using the augmentations to improving upon DreamerV2. Thus, we find that an object-detection prior is not necessarily conducive to faster convergence or better performance for the ATARI-benchmark.

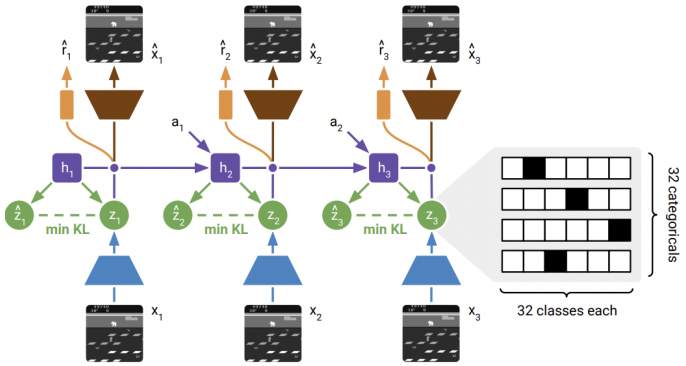

Figure 2: Google Brain's DreamerV2: discrete 32×32 world model[4]

2 Background & Related Work

2.1 Related Work

The interaction between computer-vision (object detection) and reinforcement learning is common in literature. Nayak et al. proposed 'Object RL': an algorithm that uses an auxiliary reinforcement learning agent to predict real-time transformations (pre-processing) to apply to different images, in order to improve performance on a primary pre-trained detection network [10]. Le et al. wrote an expansive survey paper, assessing recent advancements in RL and CV alike; the paper theoretically discusses current approaches used for landmark detection and object detection using deep reinforcement learning (DRL) [7]. Of note is their proposal to use a DRL (within the framework of an MDP) to predict the shifts for region proposal (up, down, left, right, bigger, smaller, taller, thinner) required to increase the IoU with ground truth region i.e. learning a region-proposal-layer (RPN) using RL. Aljalbout et al. directly explores vision-based RL in an effort to solve control tasks with image-based observations [1]. The work finds that most state-of-the-art RL algorithms struggle with sample efficiency, especially when using image observations. The work incorporates state-representation learning (SRL) into the RL pipeline to improve sample efficiency/diversity as well as the quality of samples used for CV models.

2.2 Background

DreamerV2 (DV2) is a model-based reinforcement learning architecture proposed by 'Google Brain' in 2021 that extends the 'DreamerV1' model [3] [4]. The hallmark of DV2 is the use of a *discrete* 32×32 *world model* that captures categorical latent dynamics. The pipeline consists of learning the 'world model' to get an internal understanding of the environment upon which an 'actor-critic model' can be used to that constructs an optimal policy. The first component includes a VAE-type architecture that encodes an input image (a 48 image) to a multi-modal categorical distribution; this distribution is sampled to obtain the stochastic latent state z_i which is concatenated to a propagating deterministic latent state h_i in the GRU backbone. This forms a new latent representation of the observation space. Learning the 'world model' is evaluated by the reconstruction of the input image (using a DCNN) and an estimation of the stochastic latent state $z_i^{'}$ (from the deterministic latent state). These are all optimized using a combined image-reward-discount reconstruction loss and a KL-divergence loss comparing z_i and $z_i^{'}$.

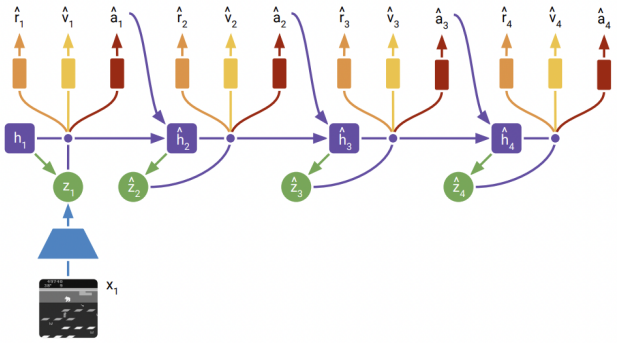

Figure 3: Google Brain's DreamerV2 Actor-Critic Model [4]

Given that an accurate and generalizable world model is constructed in the exploration buffer, the second component only requires a single image of the observation space. From that, an accurate estimation of the latent-state z_{i+1} can be propagated through the recurrent network, effectively allowing the agent to 'dream' about unobserved future states, hence the name.

Weakly Supervised Deep Detection Network (WSDDN) is a classic weakly-supervised architecture for generating bounding-box proposals from image-level class labels [2]. It first uses as pre-trained convolutional architecture (i.e. VGG-16) for feature extraction [12]. From this, region-proposal mechanisms like edge-boxes (EB) and sliding-window techniques (SSW) can be used to generate region proposals. These are then filtered by an objectness score before being pooled (i.e. spatial-pyramidal) and flattened/vectorized [5].

The vectorized 'regions of interest' (ROIs) pass through a sequence of fully-

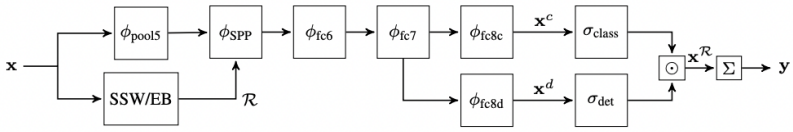

Figure 4: WSDDN high-level architecture [2]

connected layers (`fc6`,`fc7`,`fc8c`,`fc8d`) resulting in a `R`×`C` tensor i.e. a class-label assignment (in `C`) for each ROI region in `R`. Softmax is applied across class-labels in `fc8c` to generate the probability that each region (`R`) belongs to a given class (`C`); softmax is applied across regions in `fc8d` to generate the probability that each class (`C`) is found in a given region (`R`). The final score is the hadamard product of the softmax-activated `fc8d` and `fc8c` which is summed across regions to estimate an *image-level* class. This is evaluated using a C-binary log-loss term defined below.

$$E(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n}\sum_{k=1}^{C}\log(y_{ki}(\phi_k^{\mathbf{y}}(\mathbf{x}_i|\mathbf{w}) - \frac{1}{2}) + \frac{1}{2}),$$

$$\frac{1}{nC}\sum_{k=1}^{C}\sum_{i=1}^{N_k^+}\sum_{r=1}^{|\bar{R}|}\frac{1}{2}(\phi_{k*i}^{\mathbf{y}})^2(\phi_{k*i}^{\text{fc7}} - \phi_{kri}^{\text{fc7}})^{\top}(\phi_{k*i}^{\text{fc7}} - \phi_{kri}^{\text{fc7}})$$

where N_k^+ is the number of positive images for the class k and $* = \arg\max_r \phi_{kri}^{\mathbf{y}}$ is the highest scoring region in

An additional *spatial regularizer* (above) is included to penalize feature map discrepancies (at the second fully connected layer `fc7`) between the highest scoring region and other regions with high IoU overlap

Spatially Parallel Attention and Component Extraction (SPACE) is another method that aims to segment a scene into a foreground and background decomposition [8]. Given a scene x, it derives a set of foreground components $z^{fg} = z_i^{fg}$ and a sequence of background components $z^{bg} = z_{1:k}^{bg}$. For the foreground component, the model uses an encoder to train up a matrix of features corresponding to an object presence, focused object relative location, and focused object relative depth. From there, is uses a parallel spatial attention layer to compute the object identifier value which, when combined with the previous encoder outputs, constructs the foreground distribution via a variational autoencoder. In contrast, the background is broken down into a number of components, each with a generated latent pair z^m, z^c representing the component mixing probability and RGB color distribution respectively. This is also reconstructed to form just the background via a VAE. The end result is a model capable of differentiating between the foreground and the background while also proposing bounding boxes as a side effect of its reconstruction.

3 Methods

Triplet-loss Distance-based WSDDN: We propose an extension to WSDDN for situations (like ATARI games) where *all* objects of interest are present in *every* image i.e. predicting image-level labels in WSDDN is redundant. The extension incorporates a weighted *similarity loss* and *difference triplet loss* in addition to existing losses in WSDDN. Both utilize the `fc8c` softmax predictions and the flattened *features* for each ROI. The *similarity* loss minimizes the mean euclidean distance between all ROIs predicted in the same class. The *difference triplet loss* finds (for each class) largest euclidean distance between feature-vectors in the same class and the smallest euclidean distance to a vector in any other class; by minimizing the difference of these distances for each class, we emulate a triplet loss that collates feature-vectors for regions in the same class and separates feature-vectors for regions in different classes. We hoped this approach would assist in *naturally* separating/grouping ROIs by their feature maps, without the need for image-level label predictions.

Foreground-Separating Input Augmentation: We also propose using SPACE's foreground/background segmentation features to see if we can't incorporate the results to aid DreamerV2's training start-up performance. Specifically, we proposed weighing the inputs by the presence of foreground elements and thereby creating a feature-space gap between the state foreground and background elements. This is a mathematically simple operation when we consider that SPACE can be used to generate a mask $M_{x,y} = \mathbb{1}(p_{x,y} \in \bigcup z^{fg})$ which predicts the locations of the image which are occupied by a foreground object. Given such a mask M, we can use it to scale the RGB channels C of a state image simply by allowing $C' = M \otimes C$.

4 Experiments & Results

For experimentation, we deconstructed the DreamerV2 authors' Tensorflow implementation and made precision edits to modify the functionality. Much of the running was hosted on Google Colab using either a A100 or P100 NVIDIA GPU (depending on the session). Experiments on the WSDDN model took advantage of Google Cloud Platform (GCP) virtual environment with GPU support. To establish a baseline for DreamerV2 on MsPacman, we ran the authors' default model on the MsPacman-v0 gym environment. The experiment borrowed hyperparameters from the authors' Atari example defaults with a few key exceptions; namely, the environment was resized from 64x64 to 70x70 (and the encoder/decoder specification modified accordingly) to remove ambiguities where smaller objects would fail to occupy a pixel and disappear from the view. The baseline experiment ran for ∼1M game steps, ∼50K of which were preloaded before the imagination phase of training.

Initial experiments with the WSDDN model were unsuccessful. Training and

Figure 5: WSDDN results: loss by batch and generated bounding boxes

validation losses decrease sharply, which shows the potential for this model to generalize on ATARI images. However, the image-level prediction that WSDDN enforces through training is not necessarily applicable in this context, where every frame of the ATARI game contains instances of all classes i.e. by simply predicting a label $[1, 1]$, the model achieves minimum loss on all samples. The novel adaptation of WSDDN network, using distance based triplet-loss, was attempted but observed losses during training were quite unstable, so the results have been omitted; we still see strong potential for this model, perhaps involving a reformulation of the loss or tweaking hyperparamters to stabilize loss.

For the SPACE model, we were able to use the authors' PyTorch implementation which was jointly trained on a selection of 10 Atari games. Though PyTorch and Tensorflow cannot be readily incorporated together to train in an end-to-end fashion, we were able to use the pre-trained model as-is to generate an array of features in preprocessing. SPACE provides, among other things, bounding-box predictions and foreground/background predictions which we considered incorporating for DreamerV2. To avoid increasing the complexity of the state space (and due to some persisting technical issues), we avoided passing more than 3 channels and considered ways to leverage this limitation effectively.

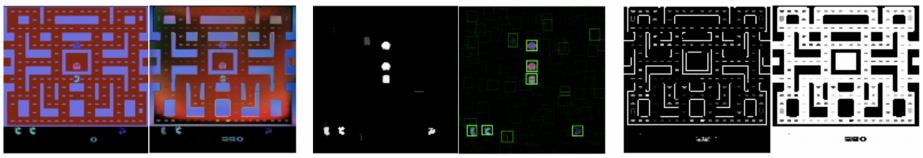

Figure 6: Some relevant inputs/outputs of the pre-trained SPACE model. Left to right: Original input and its reconstruction; foreground feature alpha mask and bounding box; background feature maps

The pre-trained SPACE model had near-constant background predictions regardless of state configuration while the foreground predictions accurately followed the moving agents and ignoring backgrounds. None of the SPACE outputs captured the state of the coins on the board, nor did any of them provide a

clear differentiator between the player and the ghost agents. To leverage these predictions, we weighed the RGB representation of the game state by the mask alpha indicating whether the object was in the background or foreground. To keep a semblance of the boundaries, we clipped the alpha values to be above 0.2, which was sufficient to differentiate the foreground/background while allowing differentiation between the game walls and pathways. Our hope with the experiment was that it would improve early-stage training by grouping the entries explicitly and so we only collected data for ∼265K state steps. Early results from the experiments suggest that, despite the scaling, the training was largely similar to the baseline with respect to the agent reward metrics. By the ∼250K step mark, both agents were comparatively similar in their performance as judged by the trend of average game length and reward mean/std-dev.

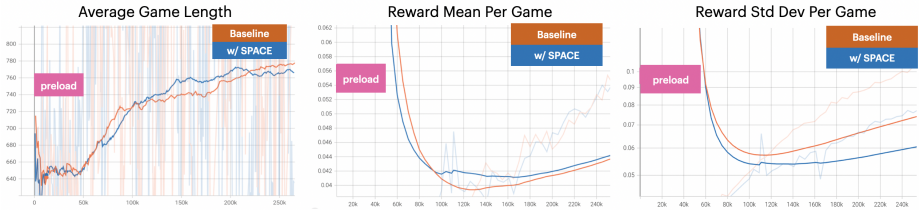

Figure 7: Tangible performance metrics as a function of number of game steps. Length represented in game steps; reward represented in abstract units.

5 Conclusion

From our attempted experiments, we have not found an effective way of incorporating object-detection schemes to boost early concept acquisition in DreamerV2. It is well-documented that DreamerV2 has had unprecedented success in image-trained model-based reinforcement learning, and close investigation does reveal that a good amount of consideration has been given to enable the model to pick up on object relationships. This is not the first time such techniques have failed; notably, the authors of SPACE also noted problems with applying their model for reinforcement learning applications, which may imply that there is a disconnect between the domains that has not yet been identified.

Despite the negative results, the interactive files used to break down and evaluate DreamerV2 may prove useful for those interested in experimenting with the model regardless of local environments and limitations (Our project codebase can be found at `https://github.com/ShreyasRaman-01/CS2951X_Final_Project`). Additionally, some of the frameworks may motivate further study where our attempts left off.

References

[1] Elie Aljalbout, Maximilian Ulmer, and Rudolph Triebel. Making curiosity explicit in vision-based rl. *arXiv*, (28), Sep 2021.

[2] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. *arXiv*, (9), Nov 2015.

[3] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *CoRR*, abs/1912.01603, 2019. URL `http://arxiv.org/abs/1912.01603`.

[4] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv*, (3), May 2021.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *CoRR*, abs/1406.4729, 2014. URL `http://arxiv.org/abs/1406.4729`.

[6] Shaul Hochstein and Merav Ahissar. View from the top: Hierarchies and reverse hierarchies in the visual system. *Cell*, 36(5):791–804, Dec 2002.

[7] Ngan Le, Vidhiwar Rathour Singh, Kashu Yamazaki, Khoa Luu, and Marios Savvides. Deep reinforcement learning in computer vision: A comprehensive survey. *arXiv*, (25), Aug 2021.

[8] Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. *arXiv*, (15), Mar 2020.

[9] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL `http://arxiv.org/abs/1312.5602`.

[10] Siddharth Nayak and Balaraman Ravindran. Reinforcement learning for improving object detection. *arXiv*, (18), Aug 2020.

[11] Paolo Papale, Andrea Leo, Luca Cecchetti, Giacomo Handjaras, Kendrick N. Kay, Pietro Pietrini, and Emiliano Ricciardi. Foreground-background segmentation revealed during natural image viewing. *eNeuro*, (28), Jun 2018.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.