Evaluating Predictive Coding Video Prediction Algorithm, Prednet, For Evidence Of Structure Learning

by

Alex Ledbetter

In partial fulfilment of the requirements for the degree of
Master of Science
at Delft University of Technology,
to be defended publicly on 26 August 2024

Faculty: Mechanical Engineering
Department: Cognitive Robotics
Programme: Artificial Intelligence for Robotics

Mentors / Supervisors: Martijn Wisse
Graduation committee: Martijn Wisse
Holger Caesar
Reza Sabzevari

TU Delft

Keywords:

Artificial Intelligence
Cognitive Robotics
Neuroscience
Structure Learning
Prednet
Object-centric
Unsupervised Learning

Abstract

Humans are our best example of the ability to learn a structure of the world through observation of environmental regularities. Specifically, humans can learn about different objects, different classes of objects, and different class-specific behaviors. Fundamental to these human abilities is evolved sensory hardware and automatic pattern recognition systems thought to be powered in part by the leading neuroscience theory of predictive coding. Artificial intelligence research is often inspired by neuroscience and algorithms already exist that implement predictive coding. In this paper, we seek to evaluate a leading predictive coding video-prediction algorithm, PredNet, for its ability to perform human-like learning of the types mentioned. By successfully training PredNet on a custom Simple Shape Motion (SSM) video dataset that explicitly requires structure learning to occur in order to accurately predict the next frame, we establish that PredNet is capable of rudimentary structure learning. We investigate PredNet filters and feature maps but find scant evidence of truly symbolic knowledge, and propose instead that PredNet performs semi-symbolic learning. We perform ablation studies that reveal the aspects of PredNet that critically contribute to its structure learning ability. Finally, we detail a set of modifications made to PredNet to allow object-centric processing as a promising step change towards human-like structure learning. Evaluation results and investigations are provided. Performance was slightly worse than Baseline, likely due to a noted implementation flaw. Code and instructions to reproduce dataset creation and model training / evaluation are available at https://github.com/ofSingularMind/parallel_prednet.

EVALUATING PREDICTIVE CODING VIDEO PREDICTION ALGORITHM, PREDNET, FOR EVIDENCE OF STRUCTURE LEARNING

Alex H. Ledbetter
Department of Cognitive Robotics
Delft University of Technology
Delft, The Netherlands

Supervisors
M. Wisse
Holger Caesar
R. Sabzevari

ABSTRACT

Humans are our best example of the ability to learn a structure of the world through observation of environmental regularities. Specifically, humans can learn about different objects, different classes of objects, and different class-specific behaviors. Fundamental to these human abilities is evolved sensory hardware and automatic pattern recognition systems thought to be powered in part by the leading neuroscience theory of predictive coding. Artificial intelligence research is often inspired by neuroscience and algorithms already exist that implement predictive coding. In this paper, we seek to evaluate a leading predictive coding video-prediction algorithm, PredNet, for its ability to perform human-like learning of the types mentioned. By successfully training PredNet on a custom Simple Shape Motion (SSM) video dataset that explicitly requires structure learning to occur in order to accurately predict the next frame, we establish that PredNet is capable of rudimentary structure learning. We investigate PredNet filters and feature maps but find scant evidence of truly symbolic knowledge, and propose instead that PredNet performs semi-symbolic learning. We perform ablation studies that reveal the aspects of PredNet that critically contribute to its structure learning ability. Finally, we detail a set of modifications made to PredNet to allow object-centric processing as a promising step change towards human-like structure learning. Evaluation results and investigations are provided. Performance was slightly worse than Baseline, likely due to a noted implementation flaw. Code and instructions to reproduce dataset creation and model training / evaluation are available at `https://github.com/ofSingularMind/parallel_prednet`.

1 INTRODUCTION

1.1 BACKGROUND

Humans are our best example of intelligent behavior and the ability to learn the structure of the world through observation of environmental regularities. Specifically, humans can perceive objects in the environment, learn that these objects belong to a class hierarchy, notice the states of objects and the world (for example, a car is out of gas, and it is daytime or nighttime), and learn to associate specific behaviors with learned classes (for example, cars generally move in the direction their front tires are pointing). These are fundamental skills that humans apply in order to understand and navigate the world we live in. These human abilities arise in part from a combination of highly-evolved sensory hardware, and a brain network architecture that facilitates automatic pattern recognition (Frensch & Rünger, 2003) (Farroni et al., 2013) (Filippetti et al., 2013). This automatic pattern recognition allows us to recognize objects that we've seen before, and to identify new objects as belonging to a class identity that we are familiar with. The leading theory for how the brain accomplishes this automatic pattern recognition is called predictive coding (Rao & Ballard, 1999) (Huang & Rao, 2011) (Millidge et al., 2022). Predictive coding suggests that the brain is fundamentally a sensory input prediction machine. This means that at all times, the brain is forming predictions about what

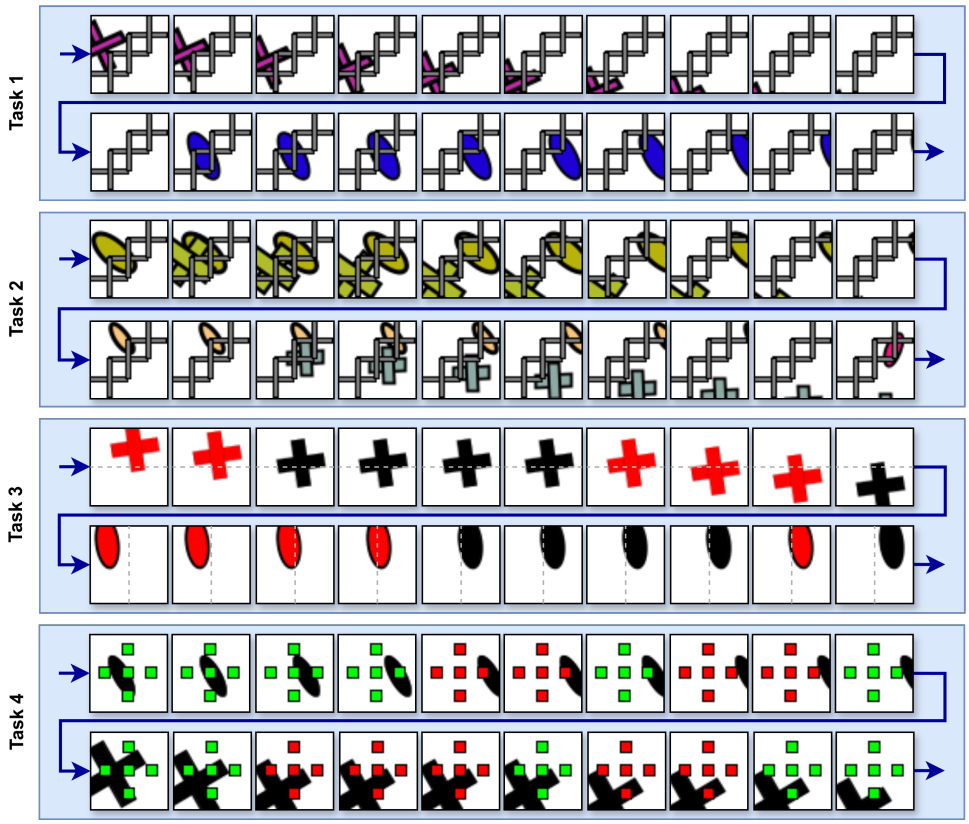


Figure 1: The Simple Shape Motion (SSM) dataset comprising four video tasks was developed in this work. Together, they enable proof of claim for a video-prediction model's ability to represent objects, classes, states, and behavioral associations at a rudimentary level. See Section 3.2 for the task details. Note that the dotted grey lines in Task 3 are added for clarity.

sensory inputs it expects to be receiving next. Through a process of prediction-error minimization, the brain adapts to form an internal representation of the world that allows ever more accurate predictions for future sensory input.

In this paper, then, we seek to evaluate an existing machine learning model that applies the principles of predictive coding for performance against human-like abilities. This effort seeks to answer the question, "*Can machine learning algorithms based on our leading neuroscience theory for how the human brain works actually learn about the same environmental regularities that humans can?*" To this end, a search was performed for a viable candidate model to evaluate against a small subset of human abilities. Our constraints for model selection included that the model learn in an unsupervised or self-supervised fashion (similar to how human infants learn without labels), the model should apply principles of predictive coding, and the model should operate over visual sensory data. This search resulted in choosing PredNet due to its simplicity, status as a well-known implementation of visual predictive coding, and due to its impressive performance on synthetic and real-world video-prediction datasets (Lotter et al., 2017).

Having chosen a model to evaluate, we establish specifically that the human abilities we are looking to evaluate against include the following: object detection in a cluttered environment, object-class recognition, perception of object and world states, and the ability to associate class-specific behaviors to perceived class instances, all at a nearly simplest-case level. For brevity we term this set of abilities, object, class, state, and behavior association representation-learning (OCSBA-RL). In order to evaluate PredNet for these abilities, we attempted to find in the literature existing methods that capture this learning for an unsupervised video-prediction model. After a thorough search, we noted that validation of OCSBA-RL occuring in video-prediction models was primarily treated as

an implicit goal while the primary and measured goals were to perform well on downstream tasks such as action recognition, person or object tracking, pose estimation, segmentation forecasting, object perception and state-transition learning, among others (See Section 2). The datasets that supported performance measurement on these tasks were deemed ill-suited for the explicit evaluation of OCSBA-RL in PredNet as we sought. Therefore, we needed to create a new dataset that allowed explicit evaluation of these skills.

In so developing such a dataset that allows for explicit testing of OCSBA-RL occuring in a video-prediction model, we first determine success criteria in order to make claims that the model has learned about these components, namely, objects, classes, states, and associated behaviors.

1.2 Success criteria to claim OCSBA-RL

1.2.1 Objects

We propose that:

- Objects exist in a world
- Objects are separate from the world
- Objects can be occluded by parts of the world
- Objects have properties: shape, color, size, position, rotation, and motion

In order to say that a video-prediction model has effectively learned to recognize, understand, and predict futures for objects, predicted futures should correctly isolate the objects from the world, correctly predict the future values of their changing-properties, and correctly maintain their static-properties. Additionally, previously occluded portions of objects that have now come into view should be in-filled with a reasonable estimate given the model's experience with similar objects. Perfectly predicting the deterministic next-frame properties while providing appropriate estimates for stochastic or previously unseen properties indicates an effective understanding of objects and their transformations over short time-horizons.

1.2.2 Classes

We propose that:

- Classes refer to a set of objects with shared class-specific properties and/or behaviors

In order to say that a video-prediction model has effectively learned to recognize and understand classes and their significance (i.e. learned about classes), without using representation-to-class-label prediction, we must rely on visually discernible class-specific properties or behaviors. With these present, we can say that the model has learned about classes when next-frame predictions display discrete and correct future values for the class-specific changing-properties, or when transformations according to class-specific behaviors are discretely obeyed. Note that by "discrete", we mean that the predictions are unambiguous. For example, in the dataset developed and presented in this paper, cross-shape class-objects are presented visually and shown to move down, while ellipse-shape class-objects move to the right. In this setup, an ambiguous prediction for a presented class-object would apply behavioral transformations that blur the line between downwards movement and rightwards movement, e.g. the next frame might show the shape having moved both down and to the right, or the class-object may have lost shape fidelity by drawing portions of the shape to be both down and to the right while the total shape is now distorted.

Additionally, the video-prediction model's recognition and predictions should be position and property-invariant (within the learned bounds of each class property including size, aspect ratio, color, edge thickness, and rotation) and generalize to previously-unseen class-shape sizes and aspect ratios within and beyond the training ranges. This is important because to claim the ability to recognize members of a class, the model should be able to recognize all possible variants of that class.

1.2.3 STATES

We propose that:

- A state reflects the current values of an object's changing-properties.
- Values for certain states have the ability to alter how objects change their properties over time. For example, when one's energy level is low, they may be less inclined to go for a run.

In order to say that the model has effectively learned to recognize and understand the effect of an object's state, predicted futures should display object transformations that discretely and correctly obey the effect of these object states.

1.2.4 ASSOCIATED BEHAVIORS

We propose that:

- A behavior refers to the manner in which changing-properties adjust over time.
- A behavior can be dependent on an object- or world-state.
- A certain behavior can be specific to and/or associated with a particular object class. For example, among 2D shape classes, only the circle rolls with a constant centroid height.

In order to say that the model has effectively learned to recognize, understand, and associate behaviors to particular objects and classes, predicted futures should display the correct behaviors considering the presented object and/or class.

1.2.5 GENERAL CONSTRAINT - NO TIME HISTORY

Finally, we propose that in order to truly demonstrate recognition and prediction for objects, classes, states, and associated behaviors, predictions should be formed without the assistance of *time-history*. This means that from a single view of the scene, the video-prediction model should be able to predict the deterministic aspects of the next frame. This additional constraint is what allows us to separate true OCSBA-oriented predictions from simple appearance and motion modeling of seen motion history, in which it is not clear if predictions are only "smearing pixels" based on how they have been moving in the last few frames, or if predictions are also accounting for the perceived objects, classes, states, and associated behaviors.

Having defined success criteria in Section 1.2 that allow claim of OCSBA-RL, a set of requirements for the design of a dataset were drafted (See Section 3.1). Following these requirements resulted in a set of synthetic videos that allow for *proof-by-demonstration* of the claim of OCSBA-RL (see Figure 1 for a visualization). This means that the videos are designed such that successful next-frame prediction explicitly hinges on an *effective* understanding of all aspects of OCSBA-RL. This effective understanding of the aspects of OCSBA-RL is developed within the internal weights of PredNet as it watches videos containing visual examples of the environmental regularities we intend for it to perceive and learn to predict. Without this developed effective understanding, the model would be unable to predict accurately what will occur in the next frame. For reference, we will refer to this dataset as the Simple Shape Motion OCSBA-RL Dataset, or the SSM dataset for short.

Equipped with this new SSM dataset, which serves as a tool to allow proof of OCSBA-RL, we evaluate PredNet on the test videos and find that PredNet is able to successfully pass the tests. The result of this effort, then, is the conclusion that **PredNet is capable of performing structure learning of objects, classes, and class-specific behaviors, including those conditioned on both object and world states, at a rudimentary level**. Following successful test completion, we perform a series of model investigations and ablation studies in order to understand the representations being formed when the model makes a prediction, and to seek evidence for symbolic learning. These investigations and ablation studies allow us to conclude that *no single element or feature map in the hierarchical representations explicitly indicate either the class being recognized or the associated behaviors to be predicted*. Nevertheless, due to the explicitly discrete predictions formed upon perception of a

scene for object, class, state, and futures dependent on class-specific behaviors, we *hypothesize* that PredNet performs semi-symbolic learning. This implies that subsets of model neurons are activating to form distributed representations that indicate proper recognition and prediction of existing scene elements, instead of only single neurons activating to indicate the presence of an object-class, the proper behavioral transformation to apply, etc.

1.3 Motivation

Finally, one might ask, "So what?" Well, a trend is noticed in the deep-learning community to progressively hyper-specialize into narrower and narrower tasks. Consider former research goals of action recognition now being divided into action detection, action localization, action segmentation, and actionness ranking (Xu et al., 2015). An alternative approach is to focus on ever more general and capable representation learning and continual world model development followed by querying for data required for the specific task at hand; "model and query". Machine learning models such as PredNet process observations into useful representations that can be queried for reliable estimates about the world it has observed (Lotter et al., 2017). This is what predictive coding suggests for human brain sensory processing, too, namely that our minds develop an exquisite world model by means of prediction-error minimization, and that querying based on the demands of the situations we find ourselves in occurs automatically, or through conscious deliberation. By extension, then, further development of model-and-query algorithms like PredNet could play a key role in advancing towards AI systems with human-like cognitive abilities.

1.4 Research Contributions

This paper makes the following contributions:

1. We identify, justify, and fill an existing gap for datasets testing for OCSBA-RL in unsupervised video-prediction algorithms.
2. We conclude that well-known video-prediction algorithm, PredNet, is capable of performing structure learning of objects, classes, and class-specific behaviors, including those conditioned on both object state and world state, at a rudimentary level, as evidenced by successfully passing a set of video-prediction tasks designed to test for exactly that.
3. We provide further evidence in support of predictive coding as a realistic neuroscientific postulate for the high-level functioning of the human brain capable of recognizing, learning, and predicting environmental regularities from visual data alone.

1.5 Paper Organization

The remainder of the paper is organized as follows. In Section 2, we review existing work that learn models related to OCSBA-RL, and the datasets used in these works. In Section 3, we discuss the justification and creation of the SSM dataset that we test on in order to prove claim of OCSBA-RL. In Section 4, we examine the experiments carried out, specifically PredNet's performance on our newly created dataset. In Section 5, we discuss several points including the justification of proof of claim of OSCBA-RL for PredNet by demonstration of successful results on our newly created dataset, further evidence in support of predictive coding, and the most-likely best next-steps for model improvement. In Section 6, we reiterate the findings presented in this paper and conclude. In the Appendix, in Section A.1, we review the model investigations undertaken to better understand the functioning of PredNet over the SSM dataset. In Section A.2 we review the ablation study performed to identify the key architectural components of PredNet required for the successful performance demonstrated by the baseline model. In Section A.3, we discuss possible future work. Finally, in Section A.4 we review a number of architectural modifications that we feel may offer PredNet some performance gains.

2 Related Work

In our search for existing work proving claim of OCSBA-RL, we encountered many datasets and learning tasks that at first glance appear to be requiring similar learning to occur. These include

Moving-MNIST (Kosiorek et al., 2018) (Jaques et al., 2020) (Hsieh et al., 2018), 2D dSprites (Higgins et al., 2017), the three-body physics problem (Jaques et al., 2020) (Kipf et al., 2020) (Ehrhardt et al., 2019), 2D bouncing balls with occlusions (Lin et al., 2020) and without (Hsieh et al., 2018) (van Steenkiste et al., 2018) (Lotter et al., 2016), physics learning for primitive 3D shapes (Lin et al., 2020), prediction for 2D object-based narrative tasks (Kumar et al., 2019), novel viewpoint estimation in stationary (Eslami et al., 2018) (Kumar et al., 2019) (Nanbo et al., 2020) (Nanbo et al., 2021) (Yan et al., 2023) and moving 3D scenes (Singh et al., 2019) (Chen et al., 2021), scene decomposition of static 2D and 3D scenes (Burgess et al., 2019) (Emami et al., 2021) (Eslami et al., 2016), planning and prediction in multi-agent vehicle intersection and infantry combat (Sukhbaatar et al., 2016) and basketball scenes (Minderer et al., 2019), prediction for Atari games (Xu et al., 2019) (Kipf et al., 2020) (van Steenkiste et al., 2018), structure modeling of action-conditioned interacting 2D and 3D shape objects (Kipf et al., 2020) (Watters et al., 2019), prediction for single agents with many behaviors, for example robotic arm motions (Finn et al., 2016) and single humans performing various actions (Minderer et al., 2019), class-based real-world action recognition (Xu et al., 2015), and pedestrian intention prediction considering only past trajectory and current position (Hoy et al., 2018) or also considering estimated pedestrian age and gender for a more class-based approach (Ma et al., 2017). These more novel dataset tasks stand in addition to a wide body of synthetic and real-world video prediction datasets that offer annotations for a number of downstream tasks such as next-frame prediction, segmentation forecasting, gaze prediction, trajectory prediction, activity recognition, video sentiment analysis, occupancy grid-map prediction (Rasouli, 2020), and video object segmentation and tracking (Gao et al., 2023) (Yao et al., 2020).

Certainly, the goals between these dataset tasks and ours have similarities, but for various reasons the complete set comprising OCSBA-RL does not appear to come together prior to this work. To demonstrate this, we would like to present a closer look at a few learning tasks that come close to what we attempt to accomplish in proving claim of OCSBA-RL. These include learning and prediction for the 2D object-based narrative tasks (Kumar et al., 2019), Section 2.1, structure modeling of action-conditioned interacting 2D and 3D shape objects (Kipf et al., 2020), Section 2.2, and class-based video action recognition via the A2D dataset (Xu et al., 2015), Section 2.3.

2.1 Narrative Tasks - Traveling Sales Person

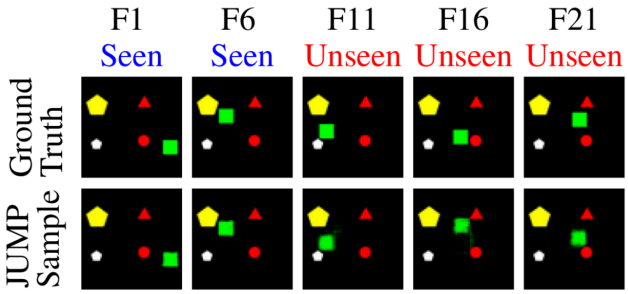


Figure 2: The Traveling Sales Person (TSP) narrative task dataset is an example of a dataset that meets some but not all requirements in order to test for OCSBA-RL. In this dataset, one shape (in this case the green square) sequentially moves towards and visits the other four shapes in some stochastic order. (Text and image reproduced with edits from Kumar et al. (2019)).

See Figure 2 for a dataset description. It depends on how we interpret the narrative test, i.e. whether there is a single object/class comprised of five shapes and a single, group behavior, or if there are five shapes belonging to two classes, agents and locations, where locations have no behavior, and agents have TSP behavior, or if there is just a single object/class, the agent, in a cluttered environment with locations to visit. However, taking the last case as the most reasonable, the Authors demonstrate that their model can identify a scene element as being an isolated object pertaining to a class (agent) and exhibiting a complicated (TSP) behavior. Successfully predicted behavior is demonstrated occasionally per the published results. We do, however, see the model struggle to capture the green square's true shape perfectly, a similar issue with our results, discussed further in Section 5. This differs from our work because there is arguably only one class-behavior association

being made, and thus no object-class distinctions need be performed and only one object's behavior is predicted in time.

2.2 Object-Action Structure Modeling

(a) 2D Shapes (b) 3D Blocks

Figure 3: The two grid world environments shown here are other examples of datasets that meet some but not all requirements in order to test for OCSBA-RL. These two grid world environments (2D shapes and 3D blocks) involve multiple interacting objects that can be manipulated independently by an agent. (Text and image reproduced with edits from Kipf et al. (2020)).

See Figure 3 for a dataset visualization. These two tasks ask a model, serving as an agent with the ability to apply actions to objects to move them, to predict what future state the objects will be in if it applies a given action. This dataset has similarities in that the model must learn to recognize and isolate objects in the scene from raw image data, and learn to predict behaviors for those objects, but the dataset differs in that there is only a single object class and action-conditioning labels are required for accurate predictions thus violating the unsupervised requirement. For these reasons, the dataset was deemed ill-suited for proof-by-demonstration for OCSBA-RL.

2.3 Class-based Video Action Recognition

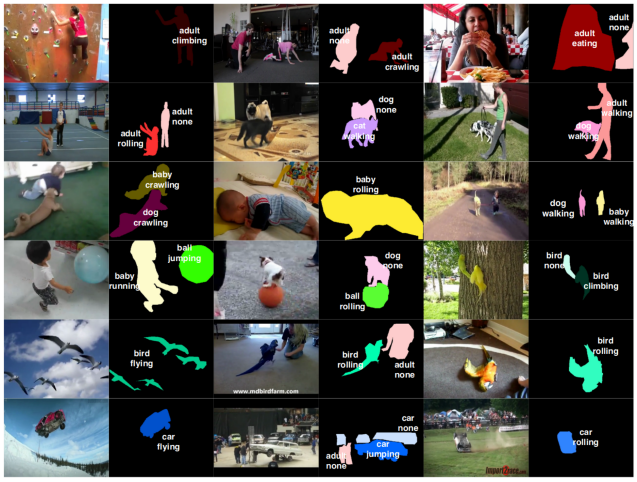


Figure 4: The actor-action dataset (A2D) is a third example of a dataset that meets some but not all of the requirements in order to test for OCSBA-RL. The image shows examples of single actor-action instances as well as multiple actors doing different actions in the actor-action dataset. (Text and image reproduced with edits from Xu et al. (2015)).

The A2D dataset (shown in Figure 4) consists of 3782 videos from YouTube with both pixel-level actors, classes, and actions labelled in each video (Xu et al., 2015). This dataset, then, offers models a chance to demonstrate learning of class-specific behaviors. For example, ball- and car-type classes can both "jump", but in very different manners. A "ball jumping" scene shows a ball bouncing off the ground, while a "car jumping" scene shows a car with hydraulic lifts.

Certainly on paper, we could see this dataset as satisfying the object-class-behavior association requirements. However, because we would like the model to predict behavior in an unsupervised

manner, and without time-history, in order to demonstrate semi-symbolic class-specific behavioral association, as opposed to pixel-wise appearance and motion modeling, the specific behavior that the class will perform should be accurately predictable from a single still frame. However, given a ball on the ground in one frame, we cannot say if in the next frame it is rolling away, or bouncing upwards. A future work could allow a few frames with poor prediction quality while the model identifies the action being performed, and then snap into much more certain predictions to possibly still prove OCSBA-RL. In the end, the dataset meets some of the requirements but offers more complexity than is desired for a simple proof-by-demonstration video dataset.

3 METHODS

In this section we will detail the requirements for a dataset that attempts proof-by-demonstration of object, class, state, and behavior association representation-learning (OCSBA-RL) and futures prediction within a structured environment (aka rudimentary world-structure learning) for unsupervised video-prediction algorithms. We will then justify why each requirement is necessary. Finally, we will describe the SSM dataset we have created in order to satisfy these requirements and allow proof-of-claim for OCSBA-RL in PredNet.

3.1 DATASET REQUIREMENTS

We establish the following as a minimum set of requirements for a video dataset to meet in order to properly test for object, class, state, and behavior association representation learning and futures prediction within a structured environment for unsupervised video-prediction algorithms:

1. **There should be multiple object classes.** This requirement allows the model to prove it can associate specific behaviors with specific object classes.
2. **Class-membership should be visually discernible from a single timestep (no labels).** This is required to isolate class-specific futures predictions as separate from appearance and motion modeling based on time-history.
3. **Class-membership should be general (there are all sorts of cars).** This is required to be sure that class recognition occurs for any representative of the class. Arguably this is a nice-to-have - we can define classes to be more specific (e.g. we could consider only red crosses and white ellipses of constant size, rotation, and position) but a human understanding of classes represents multiple possible variants that qualify as class members. We consider cars to be a class, and a Honda Civic to be a class-member of cars. We could also consider Honda Civics to be a smaller class, and Honda Civics with VIN# 123123123 to be a smaller class, still, with a single member. Keeping in the spirit of a class containing many such variants, however, we choose to enforce this requirement that class-membership be generalized for many such parameters (color, size, rotation, edge thickness, etc) while requiring a distinct feature subset to be present for each class member.
4. **Class-membership recognition should be robust to occlusions.** It is an well-known human ability to recognize classes subject to partial occlusion, e.g. when a car is half-occluded by a building, we still recognize it as a car. This requirement ensures that no single portion of the class-specific features are being relied upon for recognition. We can recognize an elephant by its trunk, by the texture and color of its skin, by its enormous feet, by the location where it seeks water, etc.
5. **Class-specific behaviors should be unique and consistent.** We aim to show association of behaviors to specific classes, and thus we avoid ambiguity for a simplest-case baseline test. We offer class- and world-conditioned class-specific behaviors as a step towards more-complicated and less-consistent class-specific behaviors.
6. **Behaviors should be visually discernible.** This is necessary for an unsupervised video-prediction algorithm to identify the behaviors.
7. **Behaviors should be simple.** By finding the simplest demonstrations that prove the claim, we aim to ensure that some models tested on the dataset succeed.

8. **Behaviors should be visually distinct.** This requirement is critical for performance assessment. When the model makes a prediction for how the perceived class-object will behave, evaluation should be obvious and binary; right or wrong.
9. **World structure should be visually distinct from object structure (unique textures or distinct borders).** We are looking for simplest-case tests that prove the claim. In a 2D image without depth cues, overlapping shapes of the same texture without distinct borders would be a more difficult task. Considering real-world object textures are often very distinct from their surroundings, this is a reasonable requirement.
10. **World structure should include visual occlusions.** This requirement positions the model to use a class-based understanding to in-fill occluded parts of the object, and to perform figure-ground organization between object and world in order to further demonstrate the model effectively understands which parts of the image belong to the object and which belong to the world structure.
11. **Video-prediction and object-class-behavior recognition and association should be qualitatively clear and/or quantitatively measurable.** The proof-by-demonstration results should be unambiguous, with distinct correct and incorrect predictions, where correct predictions explicitly rely on the correct object-class-behavior associations. Where feasible and necessary, quantitative metrics should be provided assessing performance and further justifying a successful prediction.

3.2 Dataset Creation

Here we will discuss the design of our video SSM dataset that meets our dataset requirements in order to allow proof-by-demonstration of OCSBA-RL in PredNet. We will discuss the animation software used to produce the videos and the specific tasks encoded into the videos that allow testing for OCSBA-RL.

All of the animated task videos were produced using the Processing programming language (Pro). Separate scripts were prepared to produce each animation. The language allowed manipulation of all parameters as dictated by the dataset requirements. Every animated task video was produced in 50 x 50 pixel resolution with three RGB color channels.

Each task video positions a video-prediction algorithm such as PredNet to predict the next frame. The task videos portray animations consisting of a world consisting of a solid-color background, fixed occlusions, and objects. The objects belong to one of two shape-classes; either an ellipse or a cross shape. Each class has a unique class-specific behavior; a motion. Specifically, crosses always move at a constant speed towards the bottom side of the image, while ellipses always move at the same constant speed as crosses, but to the right side of the image. For task videos 1 and 2, instances of each class have various parameters set randomly. These parameters include color, size, aspect ratio, rotation, and position, where position refers to a fixed positional coordinate for each class (x-pos for crosses moving downwards, y-pos for ellipses moving to the right, in a typical Cartesian-coordinate system). Thus, a member of each class is recognized due to its shape characteristics alone, and there are an arbitrary number of different instances of each class, generalized by these parameters. For task videos 3 and 4, in order to allow object color to indicate object state and for simplicity, only the parameters other than color were set randomly. Finally, generalization of class membership is shown further by training and testing the model on instances of different sizes and aspect ratios. More specifically, generalization interpolation and extrapolation were performed for testing. This means that when the animations were generated, the size and aspect ratios were randomly sampled from two ranges, each with an internal gap range blocked out. See Figure 5 for a visualization. Training shapes were sampled outside of this gap, while testing shapes were sampled only within this gap (interpolation), or beyond the ranges (extrapolation). Therefore, the model is shown to recognize and predict behavior for novel class instances at test time.

There are four task videos, altogether, as described in the next sections.

3.2.1 Single Instance Prediction

This task video displays animations of shapes moving across the screen per the class-specific motions. Only one shape is visible on the screen at any time. The fixed coordinate for each class is

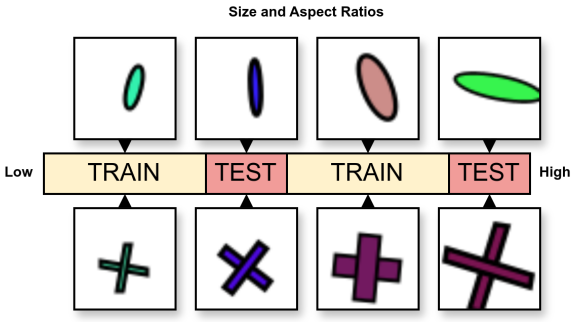


Figure 5: Generalized Train and Test Size and Aspect Ratios

randomly sampled within the ranges (width*[0.1, 0.9], height*[0,1, 0,9]) for (cross, ellipse), respectively. A series of bars form occlusions that cover portions of the shapes as they pass by. During training, animations are shown for both class types, and the model is asked to predict the position and occlusion of the presented shape in the next frame. Successful prediction relies crucially on learning the two classes, the two behaviors, and the associations between class and behavior. At prediction time, the model must recognize the shape-class, isolate the portions of the image that belong to the shape-object and those that belong to the world, apply the proper class-specific motion to the isolated shape-object, and fill in the world over and around the object.

3.2.2 Multiple Instances Prediction

This task video is identical to the Single Instance prediction case, except that there are up to two shapes visible on the screen at one time. Succeeding in this task provides further evidence that the model really knows which aspects of the scene constitute each object. In the single instance case, there is concern that the background, with the object, together are shifted down or to the right, but in this case, we see the objects correctly predicted to move in different directions.

3.2.3 Class State Conditional Prediction

This task video displays animations of shapes moving across the screen per the class-specific motions, conditioned on the "state" of the presented class-instance. The state of the class-instance is designated through shape color. Each class-instance is either red (#FF0000) or black (#000000) during all portions of the animations, and thus color is not randomly selected for each class-instance as in the previous two task videos. When the class-instance state is red, the object will show movement in the next frame, and when the state is black, the object will *not* show motion in the next frame. Thus, the model is asked to predict the position of the presented shape in the next frame, and successful prediction relies on the model having learned the association between class-state and subsequent motion, in addition to the associations between class and class-specific motion.

3.2.4 World State Conditional Prediction

Similar to Class-Conditional Prediction, this task video displays animations of shapes moving across the screen per the class-specific motions, conditioned on a "state", but this time it regards the state of the world. This time, all shapes are a constant black (#000000). World state is indicated by a cross pattern of squares that switch from red (#FF0000) to green (#00FF00). When the world state is red, the presented shapes will not show motion in the next frame, but when the world state is green, the presented shapes *will* show motion in the next frame. Thus, the model is asked to predict the position of the presented shape in the next frame, and successful prediction relies on the model having learned the association between the world-state and subsequent motion of the presented class-instance, in addition to the associations between class and class-specific motion.

Name	# of Layers	# of Output Channels per Layer	# of Time-Steps	Layer Weights
Baseline	4	[3, 48, 96, 192]	10	[1, 0.1, 0.1, 0.1]

Table 1: PredNet Model Details

4 EXPERIMENTS

In this section we will describe and visualize the experiments performed, document the results obtained, and detail any conclusions that can be drawn with reasonable certainty.

4.1 MODEL SETUP

We implemented the Baseline PredNet (with L_{all} layer weighting) in Keras 3 per specifications given by Lotter et al. (2017). See their paper for a description of the model and Figure 6 for an overview visualization. Model details are provided in Table 1. The code is publicly available at `https://github.com/ofSingularMind/parallel_prednet`.

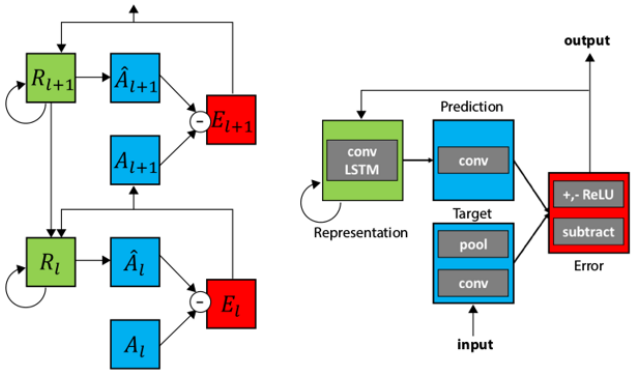


Figure 6: PredNet Architecture (Reproduced from Lotter et al. (2017))

4.2 EXPERIMENTAL SETUP AND TRAINING

For each of the four tasks, training videos comprised of 80k images were produced via the Processing animation language. PredNet is then trained via next-frame prediction-error minimization. One important note is that PredNet struggled to predict the objects and occlusions in the task videos, over the white background, without assistance. To provide this assistance, the first 40k images (1st stage) were produced with a random RGB colored background, by pixel, and occlusions, by shape. The colors shifted each frame. See Figure 7 for an example. Following these first 40k images, PredNet then completed training on the white background and grey occlusions images (2nd stage).

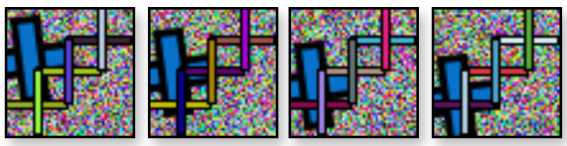


Figure 7: Random RGB Background and Occlusions Used During 1st-Stage Training

4.3 EVALUATION

After the model has been trained on the videos pertaining to each task, with the training shape sizes and aspect ratios, the model is then evaluated on videos from the SSM testing datasets, with testing

shape sizes and aspect ratios, as discussed in Section 3.2. There are two testing regimes, one for a qualitative performance analysis, and one for a quantitative performance analysis.

The qualitative performance analysis involves preparing a testing video dataset for each task, and presenting the trained PredNet model with a starting frame in the video, and asking PredNet to predict the next frame. Qualitative performance is deemed good and passing if the next-frame predictions from the starting frame display the correct object and its properties (color, shape, border width, size, aspect ratio), and if they display a transformation of the object presented in the starting frame, according to the object's class-specific behavior (crosses move down while ellipses move to the right). If the predictions display ambiguity about where the object will be in the next frame, or how it will look (aside from minor rendering quality variances), then the task is failed. After each presented starting frame, model tensor activations are reset. This erases any time-history in the model's ConvLSTM, requiring the model to decide predictions based on the presented shape in the single frame alone.

The quantitative performance analysis involves preparing a testing video dataset with 1000 image pairs for each task. At test time, we present the first image in the pair to a trained PredNet model and ask it to produce a prediction for the next frame. We then quantitatively compare the ground-truth and predicted next frames by calculating mean-squared-error (MSE) between the frames. Each image pair shows the same shape(s) in the two frames, but in the second frame, the shapes have moved per the dataset task design (crosses down, ellipses to the right, respecting the object and world states). By designing the test this way, we remove unpredictability between the first and second frame. If we instead tested on a raw animation video dataset like we trained on, there would be a number of frames where shapes randomly pop into the scene. Then, when calculating the MSE, we would be accumulating error due to the model's inability to predict these shapes popping in. We instead only want to accumulate error for the predictable changes between frames, and thus we construct our test to evaluate the model on controlled image pairs.

For Tasks 3 and 4, there is additional unpredictability related to the state transitions that needs to be removed. For Task 3, the object's color, indicating object state, will change randomly from one frame to the next. We remove this unpredictability by applying post-processing to both the ground-truth and predicted frames. This post-processing adjusts each frame to a binary black and white, where black is applied wherever the ground-truth and predicted frames display an object or prediction artifact (technically any non-white pixel becomes black). This effectively sets the quantitative analysis here to look only at object shape and pose prediction error, while the rendering quality of the shape interior and shape border are ignored. Similarly, for Task 4, the world-state indicators (the five red/green squares) will randomly change from red to green or from green to red, from one frame to another. In this case, to remove the unpredictable element from the MSE calculations, both the ground-truth and predicted frames are post-processed to set the world-state indicators to a shared solid color (we used blue). This allows the MSE to focus only on the predicted shape properties and pose.

4.4 Results

Here we present the results of the four trained baseline PredNet models against the qualitative and quantitative tasks described in the preceding section. The qualitative results are presented first in Section 4.4.1, while the quantitative results are next, in Section 4.4.2.

4.4.1 Qualitative Results

See Figures 8 through 15 for the qualitative results for PredNet on the SSM Dataset, and Section 4.3 for a description of the test methods.

4.4.2 Quantitative Results

See Table 2 for the quantitative results for PredNet on the SSM Dataset, and Section 4.3 for a description of the test methods.

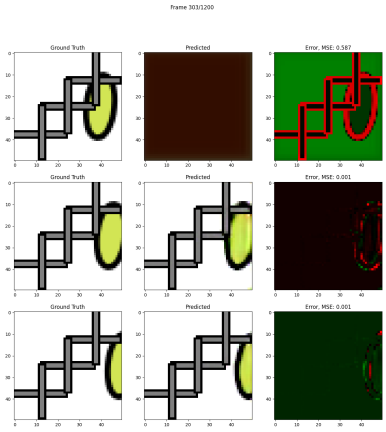


Figure 8: Qualitative Results for Task 1 - Single Shape Prediction, Example 1: Success.

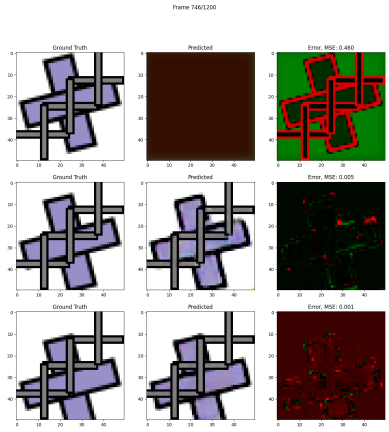


Figure 9: Qualitative Results for Task 1 - Single Shape Prediction, Example 2: Success.

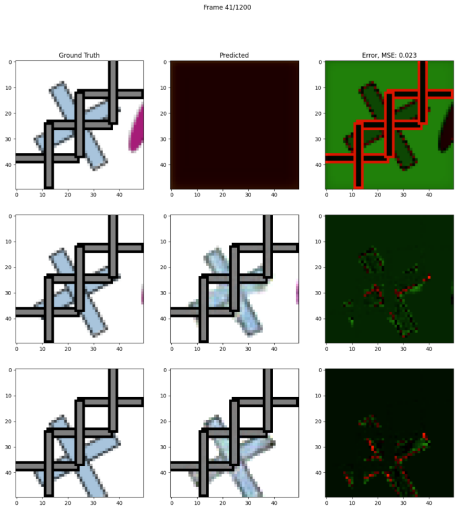


Figure 10: Qualitative Results for Task 2 - Multiple Shape Prediction, Example 1: Success.

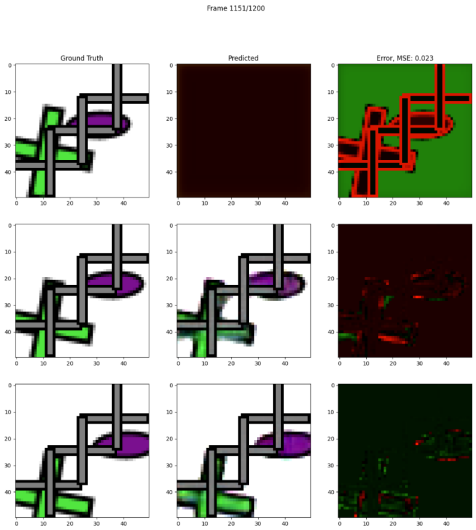


Figure 11: Qualitative Results for Task 2 - Multiple Shape Prediction, Example 2: Success.

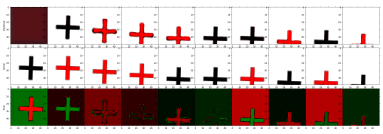


Figure 12: Qualitative Results for Task 3 - Class-State Conditional Prediction, Example 1: Success.

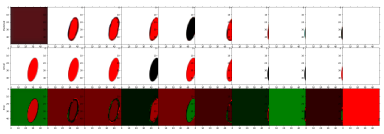


Figure 13: Qualitative Results for Task 3 - Class-State Conditional Prediction, Example 2: Success.

Prediction Source	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Previous-Frame as Prediction (MSE)	0.06487	0.10255	0.06345	0.05006
Baseline PredNet Prediction (MSE)	0.00519	0.02325	0.00549	0.00502
(PredNet Improvement over Previous-Frame Prediction)	92.00%	77.33%	91.35%	89.97%

Table 2: Quantitative Results by PredNet on the SSM Dataset

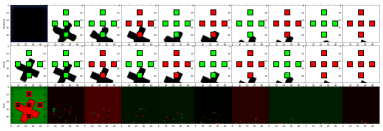


Figure 14: Qualitative Results for Task 4 - World-State Conditional Prediction, Example 1: Success.

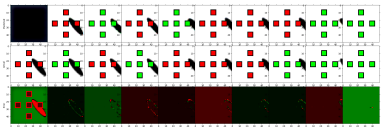


Figure 15: Qualitative Results for Task 4 - World-State Conditional Prediction, Example 2: Success.

5 Discussion

5.1 Proof of claim of OCSBA-RL in PredNet

We conclude that PredNet is capable of performing structure learning of objects, classes, and class-specific behaviors, including those conditioned on both class state and world state, at a rudimentary level. This is evidenced by successful performance on a set of "proof-by-demonstration" video-prediction tasks in the SSM dataset. Through performance on these task videos, we see that PredNet learns to predict the class-specific behavior of distinct, randomly arranged, and partially occluded class-instance objects. Performance on these task videos is qualitatively evaluated per the success criteria outlined in Section 1.2, summarized below:

- The model should perform figure-ground organization to isolate perceived objects from the rest of the scene.
- The model should perform reasonable generative modeling to in-fill previously-occluded and unseen portions of the perceived objects.
- The model should produce next-frame predictions that correctly update any general and class-specific changing-properties.
- The model should recognize in-domain class members and generalize to a reasonable domain extension.
- The model should produce predicted futures that obey the effect of perceived object- and world-states.
- The model should apply behavioral transformations according to the classes of perceived objects.
- The model should produce predictions without the aid of recent time-histories.

5.2 Further evidence for predictive coding

Neuroscience offers a compelling theory in predictive coding for how the human mind learns about objects, classes, their properties and states, and associates behaviors to specific classes. By successfully evaluating a machine learning algorithm based on this theory against these human abilities, we claim further support for this theory.

5.3 Areas where PredNet struggles

Despite the impressive performance seen from PredNet in demonstration through the SSM dataset tasks, there are a couple of areas noted where PredNet struggles to demonstrate a complete understanding of the attributes of OCSBA-RL. These include (1) generating estimates for previously occluded portions of objects, both in in the single- and multiple-shape tasks, and (2) disambiguation when shapes overlap in the multiple-shape task, Task 2.

In the first area where PredNet struggles, despite a rather high number of examples of the cross and ellipse shapes which should enable estimation of true shape form for previously occluded portions of shapes, PredNet tends to in-fill estimates that are somewhat imprecise. In Figure 16, we can see how the green ellipse is predicted to move in the correct manner, but the black border is not well-rendered. This may be due to the fact that actually, due to the low image resolution, there is

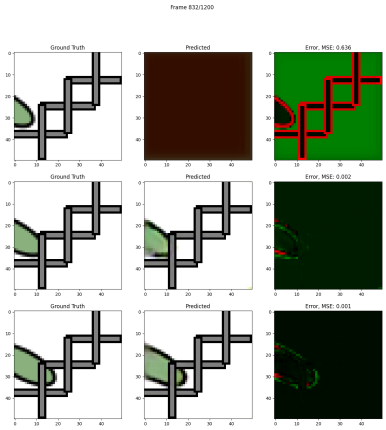


Figure 16: Infill Difficulty Example #1

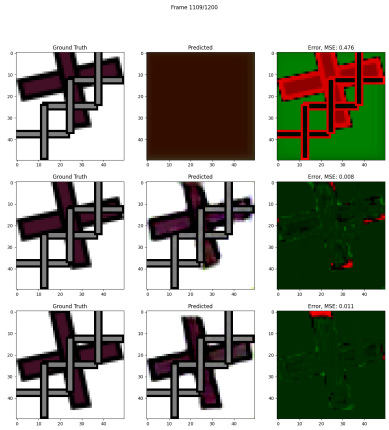


Figure 17: Infill Difficulty Example #2

a number of ellipse shapes greater than one that could appear as shown in the first time-step in the first row. Then, if this is true, PredNet is properly modelling uncertainty by making the predicted border a bit blurry. This is in-line with how the MSE loss function that PredNet utilizes operates. Namely, the mean of all possible outcomes (or shapes) often serves as the best estimate under MSE, as it minimizes the sum of squared deviations from all those outcomes (Oprea et al., 2022). Finally, then, training on higher resolution images may allow PredNet to disambiguate better what the true form of the shape is as there all fewer possible fitting shapes. On the other hand, in Figure 17, we can see an example that should have no ambiguity about the proper form and position of the lower border of the horizontal arm of the cross. In the first frame, we can see sufficiently the two edges that meet to form the lower right-hand corner of the horizontal arm. This provides the information about how to draw this lower edge of the horizontal arm, but PredNet does not form a perfect prediction. Additionally, the bottom-most edge of the vertical arm is seen in both the first and second frames, but PredNet does not draw properly this known edge.

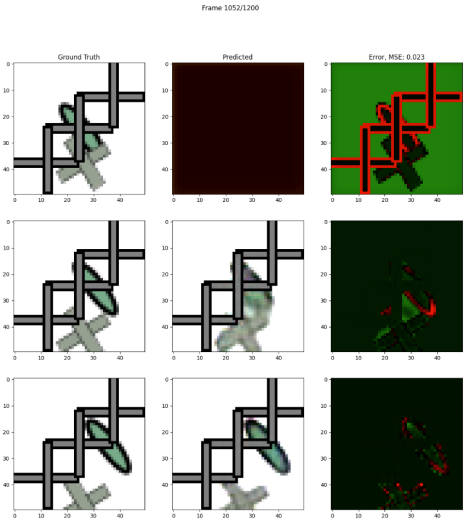


Figure 18: Shape-Overlap Difficulty Example #1

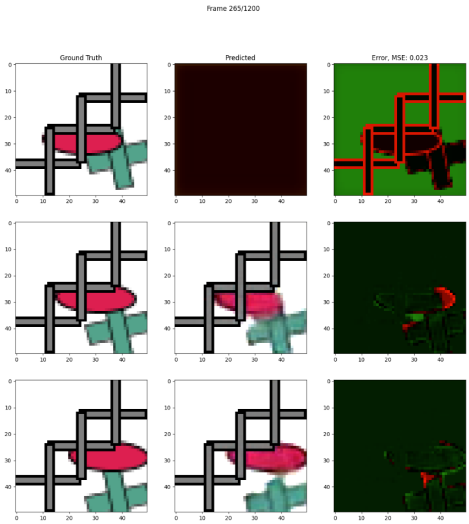


Figure 19: Shape-Overlap Difficulty Example #2

In the second area where PredNet struggles, overlapping shapes, especially of similar colors, are difficult for PredNet. It demonstrates a difficulty in determining which lines in the image belong to which shape, which leads to poor shape rendering, although the predicted motions are generally

correct. In Figure 18, we see that the ellipse and cross have been moved to the correct locations, but that the previously overlapping areas show quite poor segmented shape-rendering. In the next frame, with some accumulated time-history and with the shapes no longer overlapping, we see the model again accurately predicting form and position. In Figure 19, the overlap and subsequent rendering-quality reduction are more severe. For the regions of the shapes furthest from the region of overlap, the predictions for form and position are decent, but for the region of previous overlap, we see the shape of the red ellipse is truncated on the right side, and the left arm of the green cross (which was previously overlapping) is improperly estimated. Perhaps the largest error made, however, is in the next frame, where now most of the cross has been revealed. Here, the red ellipse looks okay, but the top of the vertical arm of the green cross is truncated. In the second frame, we note that this portion is still overlapping, and so we can explain this error as being a result of that, but the issue is that never in all of the training data has there been a cross with a shorter vertical arm than the horizontal arm. This rather embarrassing mistake on behalf of PredNet demonstrates only an incomplete or fragile representation of the objects and classes it has learned to model. When shapes overlap, this representation proves insufficient to correctly isolate the shapes every time.

5.4 Utilizing Object-Centric Processing in an Attempt to Improve PredNet's Structure Learning

In this section, we will review the efforts made to augment PredNet with object-centric processing capabilities in order to improve PredNet in the areas where it struggles. We will discuss the justification for this effort, the design details and their justification, and the results of the effort along with a discussion about reasons believed to be holding back an implementation like this from offering more aid to PredNet.

1. Justification for why it was necessary to improve PredNet
2. Introduction to the proposal for improving PredNet in an object-centric manner
3. Justification for why improving PredNet should be done in an object-centric manner
4. Introduction and Justification for the high-level design of Object-Centric PredNet
5. Introduction and Justification for the low-level design of Object-Centric PredNet
6. Discussion of results
7. Discussion of possible reasons why the approach is not out-performing baseline
8. Discussion of the possibility that unrestricted PredNet may be superior
9. Discussion of possible areas for model improvement

5.4.1 Justification for why it was necessary to improve PredNet

Baseline PredNet showed an excellent ability to associate a behavior with a class object (pick which direction to go) and to estimate the degree to which to apply the behavior. However, Baseline PredNet showed only an okay ability to predict the exact shape of the object in the next frame and to predict infilling for previously occluded portions of the object in the next frame. And, lastly, when two objects overlapped, Baseline PredNet showed a poor ability to disentangle the portions of the image belonging to each shape, and subsequently, again, to in-fill the previously overlapping portions of each shape. Altogether, these flaws suggest that Baseline PredNet can detect class objects and associate class behaviors with those objects well, but that Baseline PredNet struggled to remember and apply the vast experience with the different, highly-regular shapes that PredNet had observed. For example, when the corner of a cross shape was occluded in one frame, Baseline PredNet would tend to struggle to infill the pointed corner as it was revealed from the occlusions in the next frame. This, despite never having seen an un-occluded cross shape with rounded corners. What we would prefer instead, is to recognize that the partially-occluded shape and the revealed shape in the next frame are the same object, and have the same shape, despite part of it being covered up. If PredNet could recognize this, then we would expect better shape in-filling as the object moved away from the occlusions. Humans tend to *imagine* the unoccluded shape in full, which allows us to be less surprised when the shape is revealed to have the form we imagined (Bower, 2021). In summary, we hoped to improve PredNet's ability to recognize (A) class-object shape-continuity, (B) the regularities of a object-class' true form (all crosses have sharp corners, for example), and

lastly, (C) to apply this knowledge to perform better at the challenging task of forming predictions for overlapping objects.

5.4.2 Introduction to the proposal for improving PredNet in an object-centric manner

Taking note of the three areas of improvement we are focusing on, we see that each is particularly focused on the understanding of what it means to be one object class versus another. This leads us to suggest the high-level strategy of making PredNet more "object-centric". This means that PredNet will both learn about and predict for each of the objects individually in the scene, as opposed to learning about and predicting for the whole scene together. By learning, we mean that attention within the scene, and to the received prediction-error signals will be focused on individual objects in the scene, and then, for predicting, we mean that PredNet will be forming predictions for the individual objects themselves. Then, all the predictions are aggregated into a composite prediction for the scene as a whole.

5.4.3 Justification for why improving PredNet should be done in an object-centric manner

Having detailed the focus of improvement for PredNet, we now justify why moving towards a more object-centric PredNet was determined to be the best route. We note that humans, including infants, are good at tasks involving perception of shape continuity and object permanence (Pätzold & Liszkowski, 2020) and that humans selectively attend to the individual objects in a scene (Lindsay, 2020). Then, we hypothesize that recognizing the prediction errors associated with individual objects may help to lock in the true forms of the object classes. Lastly, we hypothesize that forming predictions for and receiving error signals for individual objects in the scene will allow PredNet to better focus on how each object in the scene is behaving, and changing from one frame to the next.

There are other justifications for why one would encourage an object-centric focus for PredNet. First, as is detailed in Section A.1, model investigations did not reveal specifically where in the model the learned aspects of specific class objects were being stored. Moving towards a more object-centric approach, depending on the implementation, may lead to a higher degree of model explainability. As we will see later on, our proposed implementation sought to make this aspect of what has been learned about certain class objects, and where it was stored, explicit.

5.4.4 Introduction and Justification for the high-level design of Object-Centric PredNet

In order to allow PredNet to improve at the mentioned goal abilities, we have suggested we encourage PredNet to focus more on, and predict for, the individual objects in the scene. If successful, then we argue that PredNet is operating in a more object-centric manner. In order to allow PredNet to focus on and predict for the individual objects in the scene, we suggest the following high-level modifications. In the next section, we will discuss how each modification is implemented.

1. Scene decomposition - Decompose the scene, which is an image of several overlapping objects, in general, into several images, one for each object, in which single isolated objects from the scene are displayed over black backgrounds. From time-step to time-step, keep track of which objects are in which decomposed frame to ensure that temporal regularities are being recognized for the same objects over time.
2. Classification - For each decomposed frame displaying only a single object in the scene, perform a classification task to associate a class label with each frame.
3. Short-term memory - Based on the class label assigned, store in a sliding window short-term memory, the last two frames observed for each object in the scene.
4. Long-term memory - For each class label, and for each sequential pair of frames, form long term memories by compressing each two-frame sequence into a low-dimensional vector encoding the spatial and temporal properties for an observation of that object class. Maintain a set of these vector memories as a form of stored knowledge about an object class, formed through PredNet's observations of that class.

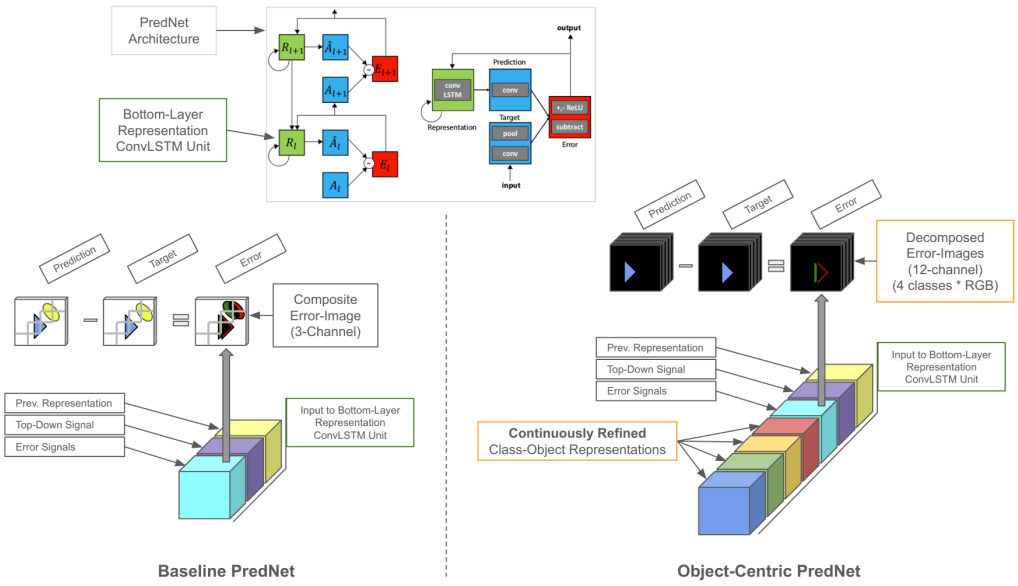


Figure 20: High-Level Object-Centric PredNet Implementation - We change the data structure PredNet predicts for from a composite image to a stack of decomposed images, one for each class-object, and we include learned class-specific object representations as additional inputs.

5. Class-Object representations - At each time-step, and for each class, consolidate the set of stored long-term memories along with context from the current scene in order to form a single information structure (tensor) representing what PredNet knows about the object classes in general, and how this knowledge should be applied to the current scene in order to inform next-frame predictions for each object in the scene.
6. Decomposed predictions - At each time step, and for each decomposed frame, aligned based on the specific identity of the objects in the previous frame, form predictions for each of the decomposed frames, considering the recent past states of the object in the current and past frames, as well as the context-aware object-class representations provided to PredNet's representation units. Predictions take into account the error signals for the other objects in the scene.
7. Decomposed learning - At each timestep, and for each decomposed frame, make adjustments to model calculation parameters based on the error signals generated for each object's prediction imperfections.

In summary, PredNet should decompose the scene into the individual objects present, identify the class to which each object belongs, and continuously create short- and long-term memories related to the observations of each object class. These long-term memories are then used to form a representation of how a particular class typically looks and behaves. This class representation is applied to the current frame context to create a consolidated understanding of each class. This consolidated representation helps inform PredNet about the likely future states of each class object in the frame, based on both past observations and the current behavior of the object. PredNet then uses this information to make predictions for each isolated object in the scene, taking into account the states of all objects in the scene. Finally, it adjusts model parameters based on the error signals generated for each object, improving its future predictions for similar objects in the scene. Note that all of steps 2 through 4 occur within a standalone ObjectRepresentations layer augmented within PredNet's bottom predictive-coding layer, while step 5 occurs in each predictive-coding layer, creating object representations individually tailored for that layer's predictions and error signals. Additionally, steps 6 and 7 only occur in the bottom predictive-coding layer. This is because only the bottom layer produces representations and predictions concretely aligned on a channel-basis to the scene elements. The upper layers, having vastly more representation and prediction channels, tend to produce more abstract representations and predictions that the bottom layer then learns to interpret.

Step 1 is performed in the dataset loading phase during training and testing. More detail will follow in the next section.

5.4.5 Introduction and Justification for the low-level design of Object-Centric PredNet

Scene decomposition A large part of the inspiration for the object-centric PredNet and the decision to decompose the scene into isolated objects was the Deepmind model, MONet Burgess et al. (2019). This model is able to learn to decompose a scene into a set of textured masks, one for each object in the scene. An additional benefit was that the textured masks formed would, in some cases, show proper in-filling for the partially-occluded portions of of the objects; so, each mask displayed what the object looks like on its own. We felt this would be particularly useful for learning about the true form of the objects in the scene as they moved behind obstacles. In the end, however, three flaws prevented the direct use of MONet in our final implementation to improve PredNet. See Figure 22 for a visualization of all three. First, the in-filling only showed up for objects in 3D scenes. This was true in Deepmind's published results as well as in our testing of MONet on the SSM dataset. This flaw, by itself, did not prevent use of MONet. However, in addition to the lack of in-filling, we observed the two other flaws to be an inconsistency in object-ordering for the created masks, and imperfections in the created masks. By inconsistent ordering, we mean that, from one time-step to the next, MONet would assign object A to slot 1, and object B to slot 2. Then, in the next frame, MONet would sometimes swap the arrangement of objects to slots, for example placing object B in slot 1, and so on. This is problematic when we consider how PredNet is expected to learn from these decomposed frames, that is, by use of a convolutional LSTM (ConvLSTM) which expects coherent sequences of images in order to produce a representation of that sequence. If the sequences fed into the ConvLSTM are misaligned, sometimes showing object A, and other times showing object B, then the representations formed would try to represent both objects in the sequence together, which is not in alignment with our object-centric approach, where we would like to form representations of sequences of isolated class-objects. Then, by imperfections in the created masks, we mean that while, in the complete scene, the objects are distinct, and should be able to be unambiguously decomposed, sometimes MONet would form a pair of textured masks that combined the features of the two objects. In reality, MONet performed quite well at the isolation task, and only rarely mixed together the two shapes. Confronted with these three imperfections, we decided that, because MONet very-nearly did what we required of it, we felt that it would be a reasonable stretch to perform the scene decomposition (and frame-to-frame object alignment) ourselves, manually, considering the very simple shape motion dataset. The result of this decision is that PredNet is now augmented with an artificial ability to decompose a scene into its isolated objects which is representative of where we feel unsupervised scene decomposition capabilities will be in a couple of months or years. Note that no manual object in-filling was performed. While MONet did demonstrate this ability, for 3D objects, due to time constraints, we leave this as future work to further improve PredNet. Finally, we discuss where the scene decomposition occurs. The intent was to place MONet within the Target units of the bottom layer for Baseline PredNet. This means that as each new scene frame was observed by PredNet, it would immediately be decomposed into a stack of class-object-sorted images before passing it on to PredNet for standard processing. Then, as we moved away from MONet, we decided to perform the scene decomposition directly within the dataset loading sequence. A dataset dataloader would pull out a batch of sequences of images, then pre-process each image into a stack of decomposed frames, then align each frame within the stacks according to the class-object positions within the stacks for the previous frame, ensuring that the top position in each stack, from time-step to time-step, would display the same object moving around, for example. The images in each stack were concatenated in the channel dimension, producing an H x W x 12-channel tensor from the H x W x 3-channel input scene image. These twelve channels (from four 3-channel RGB images) correspond to the up to four class objects present in the scene at a time, per the SSM dataset. Those classes include the background, the criss-cross occlusions, and the crosses and ellipses. See Figure 23 for a visualization.

In summary, while Baseline PredNet would receive a sequence of single images of the scene, Object-Centric PredNet receives sequences of class-object-aligned stacks of images.

Classification In order to allow storage of class-specific short- and long-term memory, as well as creations of class-specific object representations, as classification network is used to sort the

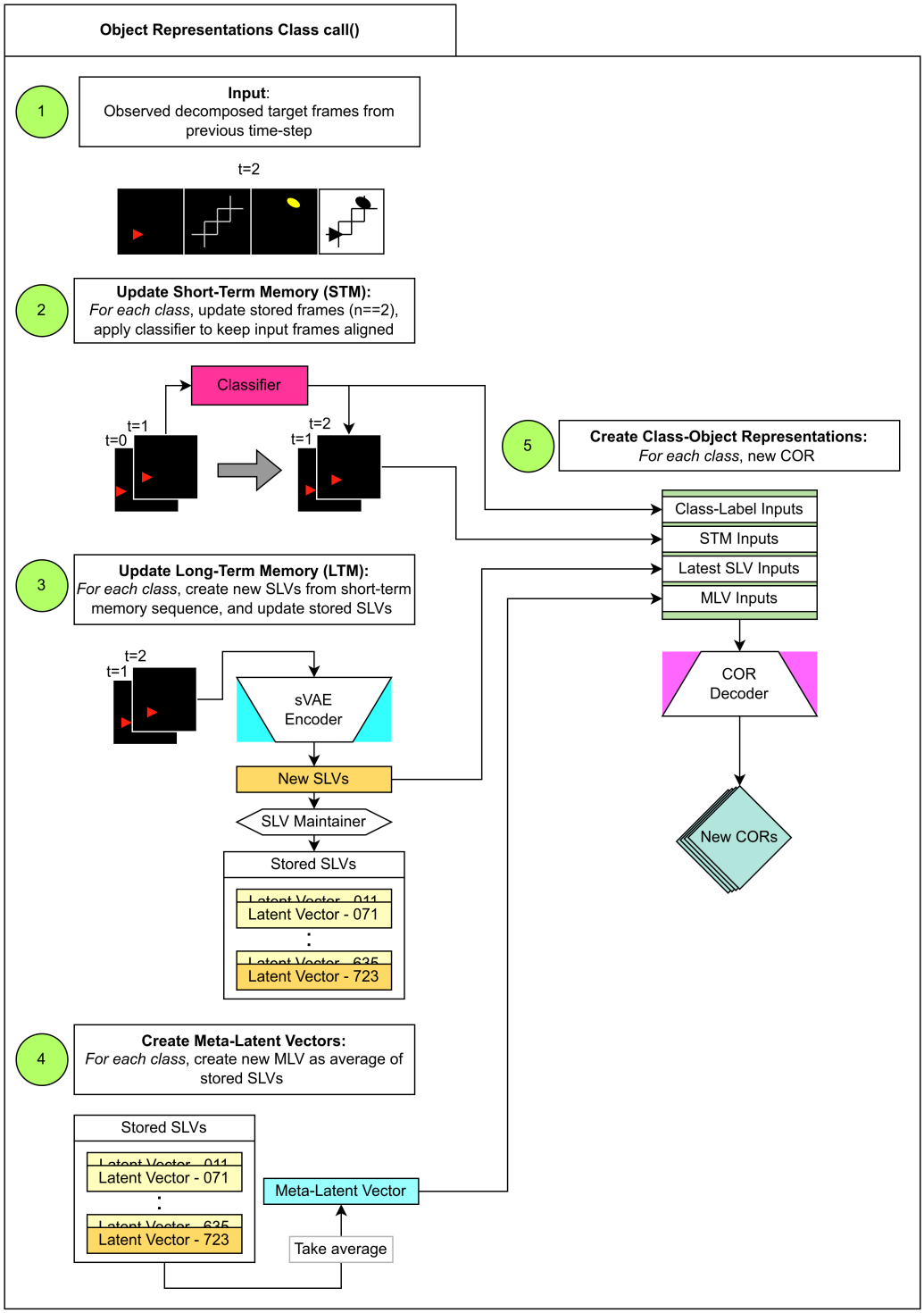


Figure 21: Object-Representations Class Process Flow

observed, decomposed frames as they enter memory. As noted in Section 5.4.4, the classifier is only present in the bottom layer of PredNet. The classifier was chosen to be a simple custom CNN composed of a stack of alternating 2D convolutional layers and max-pooling layers, ending with a pair of dense layers that produce a final set of logits for each frame in a batch of decomposed frames

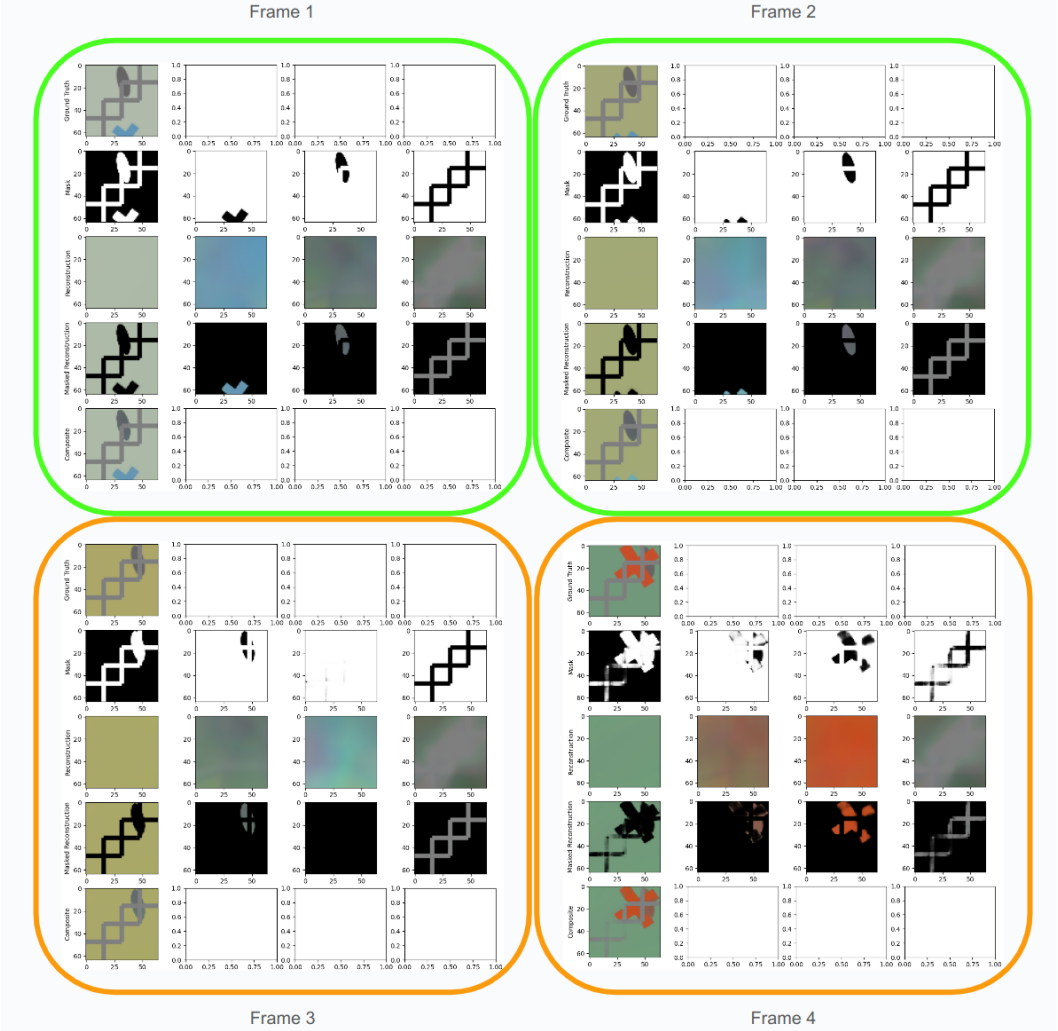


Figure 22: Mixed Performance from MONet. The first two frames show MONet decomposing and aligning the images by class well. The third frame makes a class-alignment mistake, and the fourth frame makes a quality mistake. In all frames, in-filling is absent.

presented to the classifier. The logits represent a degree of confidence associated with each of the candidate classes. The logits exit the classifier in raw form, and are converted into a probability distribution over the four classes during the short-term memory storage stage. Note that we pre-train the classifier on samples from a dataset similar to the SSM dataset except that during the creation of this alternate dataset, the colors of the four object classes are locked to four distinct colors. This allows us to decompose the frames, and assign a class label based on the color of the object, automatically. See Figure 24 for the architecture details.

In summary, the classifier produces a set of predictions corresponding to the most-likely class for each decomposed frame in the 12-channel stack, at each time-step.

Short-Term Memory The short-term memory storage stage is simple, and occurs only in the bottom-layer of PredNet. There are two steps. At each time-step, a batch of 12-channel decomposed-frame tensors arrives. The order of the images in the stack, however may not align with the classification class-IDs. So, we take the logits from the classifier associated with this stack of frames, use softmax(logits * beta), where beta is a large number (e.g. 1e6) to convert it into an approximate one-hot encoding, and then re-order the images in the stack so that the image classified 'Class 0' is in position '0' within the stack, the image classified 'Class 1' is in position '1' within the stack, and so

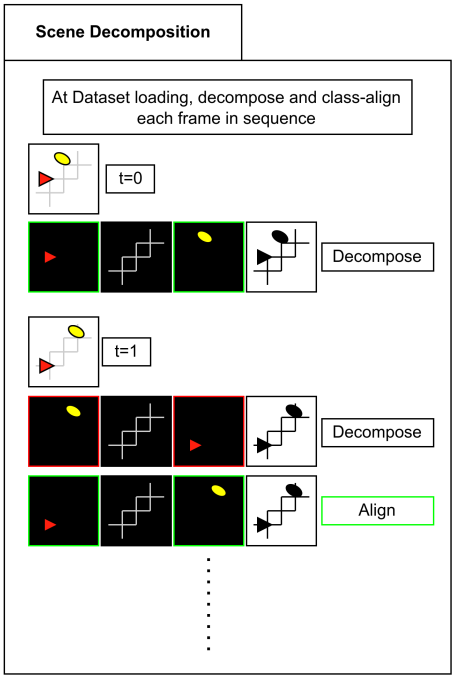


Figure 23: Dataset Loading Scene Decomposition - Note: The SSM uses crosses instead of triangles which are only shown for clarity.

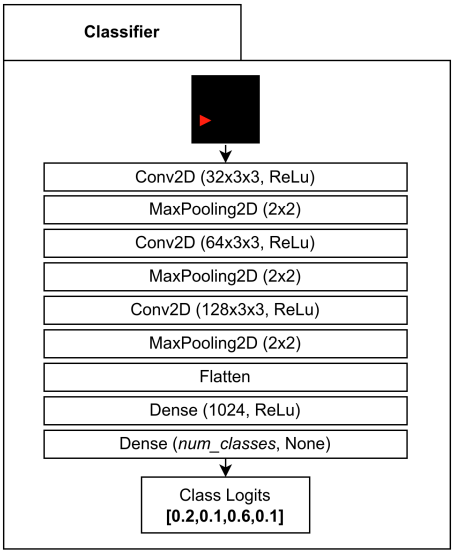


Figure 24: Decomposed-Frame Classifier Architecture

on. Next, with this reordered stack of images, we update our short-term memory buffer. This buffer stores the most recent frames for each class according to a specified hyperparameter, which dictates the number of frames to retain in each sequence—for example, the last two frames. For simplicity, if we consider a batch size of one, the buffer maintains the last two frames for each class as a form of short-term memory. These frames remain uncompressed to preserve the full information content in the short term.

There is one consideration to be aware of. Recall that the SSM dataset consists of shapes moving down and right. In can happen that one or both shapes have moved 'off frame' and then one or two of the frames received at the short-term memory storage stage are empty frames. In this case, the classifier will give consistent but ambiguous logits corresponding to which class is present in the frame. We handle this by first nullifying the logits associated with empty frames, then apply a softmax over the class-prediction axis (again with a high beta value to ensure nearly one-hot predictions) to 'lock-in' the predictions for the non-empty frames. Finally we add a little positive-value noise to the nullified logits to make them nearly zero but unequal. Finally, we apply another softmax, but this time along the image stack axis. The previous softmax identified the most likely class for images with non-nullified logits. This second softmax along the image-stack direction then assigns a probability of 1.0 to the as-yet unassigned classes for the empty frames. In the end, this ensures that each class has one frame assigned to it, with blank frames being randomly assigned to the classes not actually present in the frame in that time-step.

Long-Term Memory At each time-step, the long-term memory storage step converts the latest short-term memories into a compressed and structured vector that is memory- and information-efficient for long-term storage and later usage. Each vector encodes both spatial and temporal properties of the input short-term memory sequence, on a class-by-class basis. The key to the long-term memory formation is our custom Sequence-Variational-Autoencoder (sVAE), see Figure 25. A typical VAE (Kingma & Welling, 2013) is extended by including a convolutional LSTM (ConvLSTM) at the head of the encoder. This ConvLSTM takes the short-term memory sequence as input, and outputs a single tensor representing the whole sequence. This output is then fed into a typical conditional VAE encoder (Kingma et al., 2014) comprised of a set of 2D convolutional layers followed by a dense layer where the class ID of the current sequence is encoded in, and then another dense layer to produce the final class-specific "sequence latent vector". This sequence latent vector then represents a single long-term memory. We then maintain a set of such sequence latent vectors for each class using a Sequence Latent Maintainer, see Figure 26. This set, then, represents the long-term memory for what PredNet has observed it to mean to be that class, including spatial and temporal properties, and over several instances observed. This sequence VAE is trained to encode compressed and informative sequence latent vectors by pre-training the unit on a sequence reconstruction task over the SSM dataset. The sequence reconstruction task is performed by also training a decoder which converts this newly compressed sequence latent vector back into the input sequence, as close as it can. This process is optimized by minimizing reconstruction error and the KL divergence over the latent variables. Note that while the short-term memory sequences are of 3-channel RGB images, these images are converted to binary masks before being passed into the encoder. This is mirrored on the decoder side, where the decoder produces a matching sequence of binary masks. The intent here was to ignore color-information and allow better reconstructions from the single channel image.

The rationale for the Sequence-VAE design is as follows. First, a ConvLSTM is designed to transform an input sequence of images into to out sequence of tensors representing the spatiotemporal features at each timestep (SHI et al., 2015). However, a ConvLSTM can also be used to transform a continuous sequence of images into a single tensor that represents the whole sequence. This is performed by processing the sequence and taking only the final output tensor as the sequence representation. The specifics of the form out the output tensor from a ConvLSTM will depend on the loss function used to train it. In this context, because the loss function is connected to the later-decoded sequence reconstruction loss, the ConvLSTM is simply trained to provide a useful representation of the sequence for that task. Then, with this useful representation of the sequence in hand, we use a VAE to compress these sequences into a single, structured object, in this case, a small vector (dimensionality 32). A VAE compresses high-dimensional inputs into a set of latent dimensions that tend to display structural properties. This means that, ideally, each element of the compressed-form vector, or, latent vector, tends to encode a specific aspect or property of the input data. For example, if the input data is an image of various simulated 3D objects, adjusting just one of the latent vector elements will tend to adjust, for example, the color or position or shape or size of the 3D object. This is shown when the latent vectors are decoded back into their source images. Then, when we adjust single elements of the latent vectors, we can then decode those adjusted vectors and see how the reconstructed images have changed. For example, adjusting one latent vector element might show reconstructed images of the same shape, but changing colors. This structural encoding is useful in our case, because we want to store specific spatiotemporal memories of observed class objects. Later on, these stored memories can be combined in a number of fashions to produce another vector

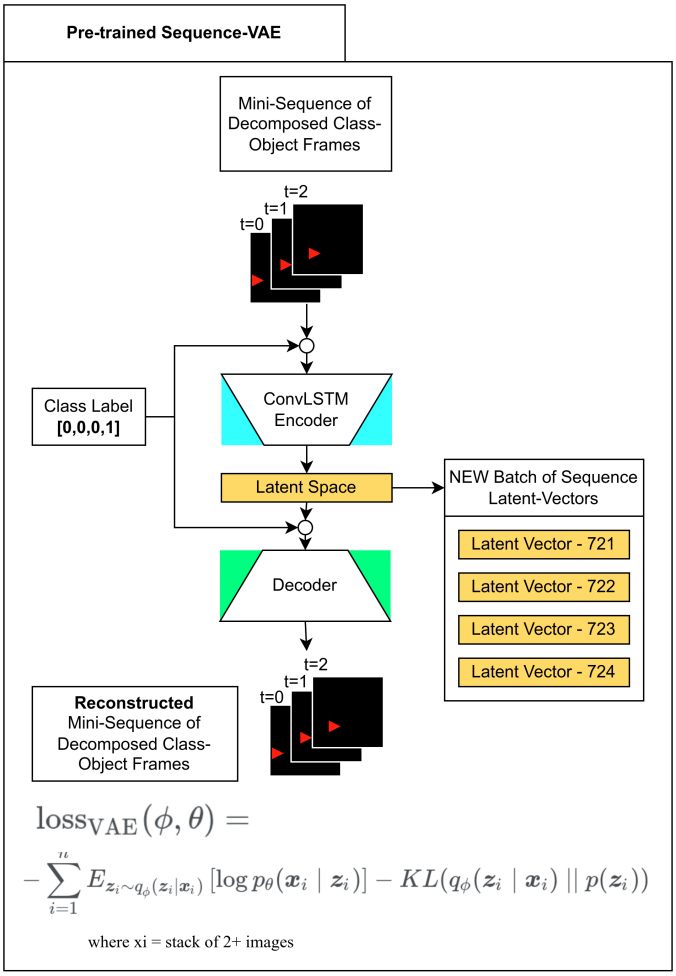


Figure 25: Sequence Variational Autoencoder (sVAE)

that represents all of these spatiotemporal properties as seen over many observations. This combination of spatiotemporally-encoded latent vectors (sequence latent vectors) we refer to as a meta-latent vector, as it represents the properties observed for a class object, in general.

At each time-step, we produce a new sequence latent vector for each object class, but we only keep a limited number of them in long-term memory. We hypothesized that it would be advantageous to store an "ideal" set of these sequence latent vectors. Several methods were attempted in order to maintain this set of ideal vectors. We felt that maximizing the diversity among the stored experiences would be a strong approach, attempting to ensure two things. First, it would ensure that the meta-latent vectors formed from the combination of the set would represent a wide range of observations from these class objects. Second, it would ensure that, over time, the set of stored vectors would stabilize to an ideal set. This occurs, because, over time, as we experience more and more observations of a class object, we tend to eventually have seen all of the various things that that object tends to do, and all the different forms / appearances / positions that the class object will appear in. Having seen "all aspects" of the class object, and having picked from all of those observations, the most diverse set, we then see that new observations tend to be less novel than the existing experiences, and thus do not overwrite them. As this happens, we argue, then, that we have "learned what there is to learn" about the class object. Quantifying diversity was approached with a combined loss metric seeking to maximize the sum of pairwise distances between the stored vectors, in summation with the total volume spanned by the set of vectors as computed by the determinant (or log-determinant for numerical stability) of the matrix, G, formed by the matrix multiplication

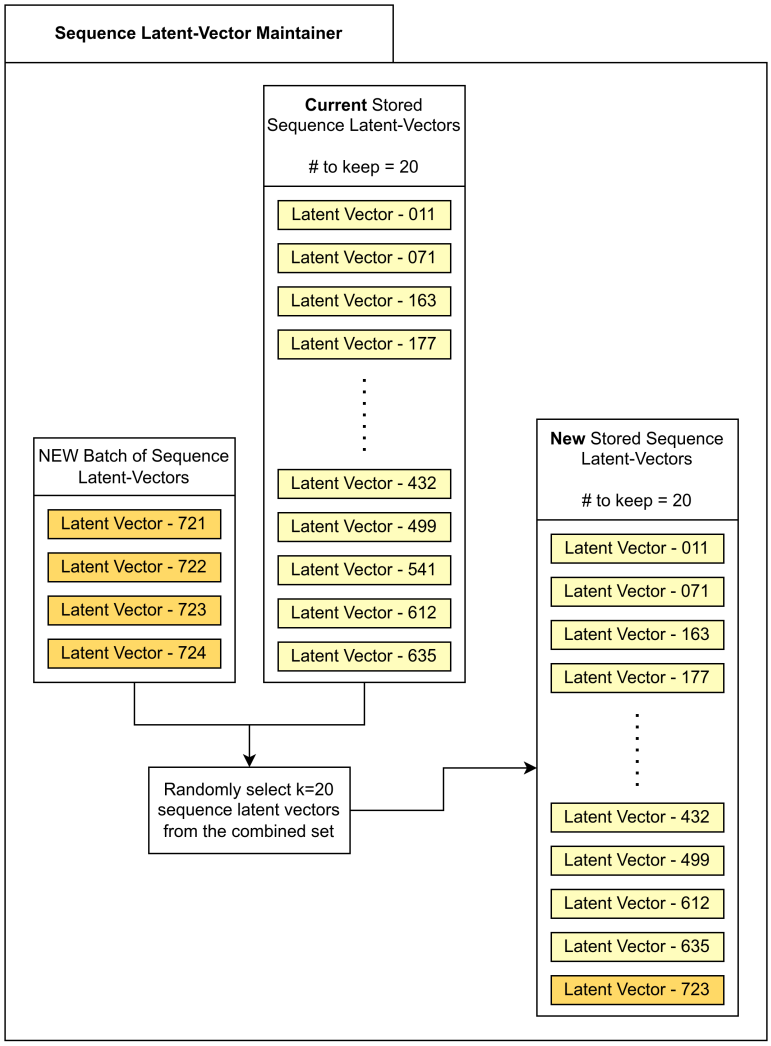


Figure 26: Sequence Latent Vector Maintainer

of the tensor of vectors with dimensions (number of stored vectors, vector dimensionality) and the transpose of this tensor of vectors (Gover & Krikorian, 2010). $[volume = logdet(V * V^T)]$. This is known as the Gram method, named after Jørgen Pedersen Gram. Unfortunately, however, when considering an existing, full set of vectors (full in the sense that the desired number of vectors to be stored in total has been reached) and a new vector (or batch of vectors), we must make a "selection" from this combined set of candidate vectors. In general, "hard" selection means to pick exact copies from the combined set, and "soft" selection means to form a new set of vectors as the weighted sum of vectors from the combined set. We felt that in order to maintain the integrity of the stored set of vectors, we needed to be performing a hard selection, or at-least, a very close approximation. In deep-learning, however, this is difficult, and extensive research into the problem did not reveal a solution. We, however, devised two such methods. In the end, however, due to time constraints, and numerical stability and/or gradient propagation issues, that may or may not prevent the actual use of these two methods, we considered two simpler approaches. The first approach requires us to simply maintain a sliding window of experiences. This means that, as each new frame is fed through PredNet, a new sequence latent vector experience is formed. Then, we update our long-term memory by removing the oldest experience, and adding in our newest experience. The second approach, instead, randomly selects sequence latent vectors from the combined set of stored and new vectors. Both of these approaches have no ability to maximize diversity as discussed, but they do offer simplicity and

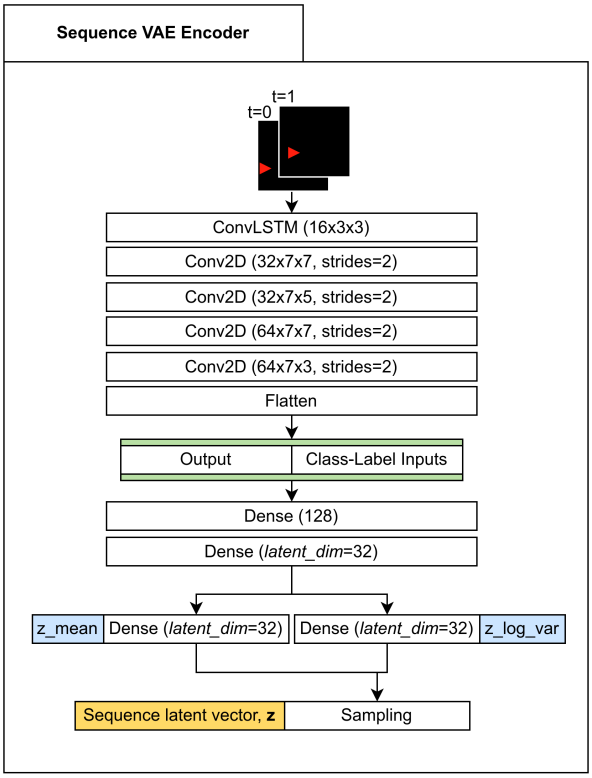


Figure 27: Sequence Variational Autoencoder - Encoder

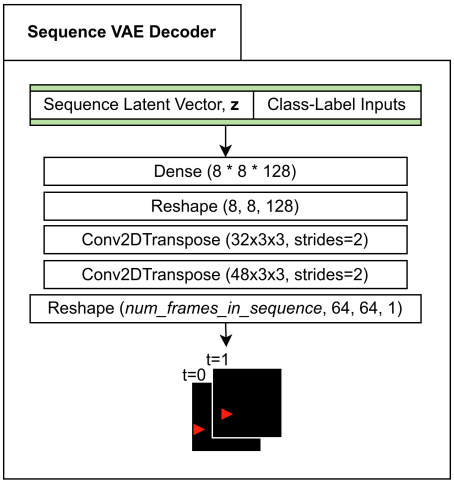


Figure 28: Sequence Variational Autoencoder - Decoder

computational efficiency. The published results in this paper utilize the random selection approach. Over the sliding window approach, the random selection approach offers a broader set of long-term memories which should increase vector diversity. Finally, we store the 20 last such spatiotemporal sequence latent vector experiences for each class, and we feel certain that, for our dataset, due to the lack of extreme variability in class object form and behavior, this maintained set is still fairly representative of the class-object's properties overall.

In summary, for each class, 20 randomly-selected stored sequence latent vectors represent our object-centric long-term memory storage.

Class-Object Representations The class-object representations are then formed from this class-specific long-term memory storage. Having formed a stored set of sequence latent vector experiences, as just discussed, we want to then combine this set into a single form representing the class-object's properties in general; the meta-latent vector. Again, we considered a few options. One was to pass the flattened set of sequence latent vectors through a series of dense layers to then produce a single vector of specified size. This may have been a fine approach, but for reasons explained later, we went with another, third approach. Second, we considered to utilize a second VAE that would perform further compression of the set of vectors. The VAE would be trained to reconstruct the original set of latent vectors with a regularized latent space. We found, however, that reconstruction error remained high despite efforts to make the encoder and decoder units robust and capable. We hypothesize that, because each of the original sequence latent vectors are of a compressed, and thus highly information-rich, form, by attempting to perform further compression (we chose a meta-latent dimensionality only twice that of the original vectors), there is simply too much loss of information to reconstruct the input set of vectors well. Due to time-constraints, proof of this hypothesis is left for future work. The third option we considered was the simple average of the input set of sequence-latent vectors. The rationale here is that, if each element of the input sequence latent vectors already encodes some aspect of the class object's form and behavior, then the average should be the average of these aspects. Consider that the first element controls to a high degree the x-positional coordinate of the shape, and the second element controls strongly the color of the shape, and the third element controls strongly the rotation of the shape, and the fourth element controls the direction of movement from one frame in the sequence to the next frame, and the fifth the speed, etc., then, for a single class object, averaging this set of vectors will result in a sequence latent vector representing the average of these properties. So, the average size, average direction of motion (constant in our dataset, dependent only on the object's class), average position, etc. Based on this rationale, we felt that this was a fairly informative means to combine the vectors for later use. Additionally, similar to the method we chose to select which vectors to store, this method is parameter-free and computationally-efficient.

Now that we have formed a single meta-latent vector, by taking the average of our long-term memory set of stored sequence latent vectors, which represents a combined set of spatiotemporal class-object experiences, we will now discuss how the vector is used in order to assist PredNet in making informed predictions about the objects it is predicting for. First, we discuss where the output of this process should go. This tells us what tensor shape we should expect our output to be in. We considered and tested three options for where in the PredNet processing stream to insert this class-object specific information. First, we considered to place the output into PredNet's bottom-layer Representation unit as additional input alongside the top-down information signal. Second, we considered to utilize the object representations directly in the Prediction units, as an additional input. Finally, and this was our chosen implementation, we utilize the first approach, but we form object representations tailored for the Representation units in each PredNet layer, instead of just the bottom layer. Tailored in the sense that they have the correct spatial dimensions for that layer.

Taking this final approach of feeding the object representations to the Representation units in each PredNet layer as our model, we then describe how the consolidated meta-latent vector for each class is turned into a form usable by these Representation units. We utilize a "class-object representation (COR) decoder" quite similar to the decoder for the sequence-VAE. There are two differences. First, the COR decoder takes four inputs: for each class, the decoder takes the meta-latent vector, the latest sequence latent vector, the latest stored short-term memory frame sequence, and the class ID encoded as a one-hot vector. The logic is thus; we want to combine the general class knowledge from the class-specific meta-latent vector with context about what that class object is doing and looking like right now in the current scene. This context is provided by the latest sequence latent vector and latest frame sequence. As a first input, the meta-latent vector and the latest sequence latent vector are concatenated along with the class ID vector. These get expanded into an image-shaped tensor the same size as the Representation unit in that layer of PredNet. This tensor is then concatenated with the image sequence, and then convolved into a final class-object representation tensor combining general class features and local context in order to assist each PredNet layer to produce informative Representation tensors for subsequent predictions and error-minimization. See

Figure 29 for details. And second, the COR decoder is trained via PredNet's overall prediction-error minimization scheme, instead of being pre-trained.

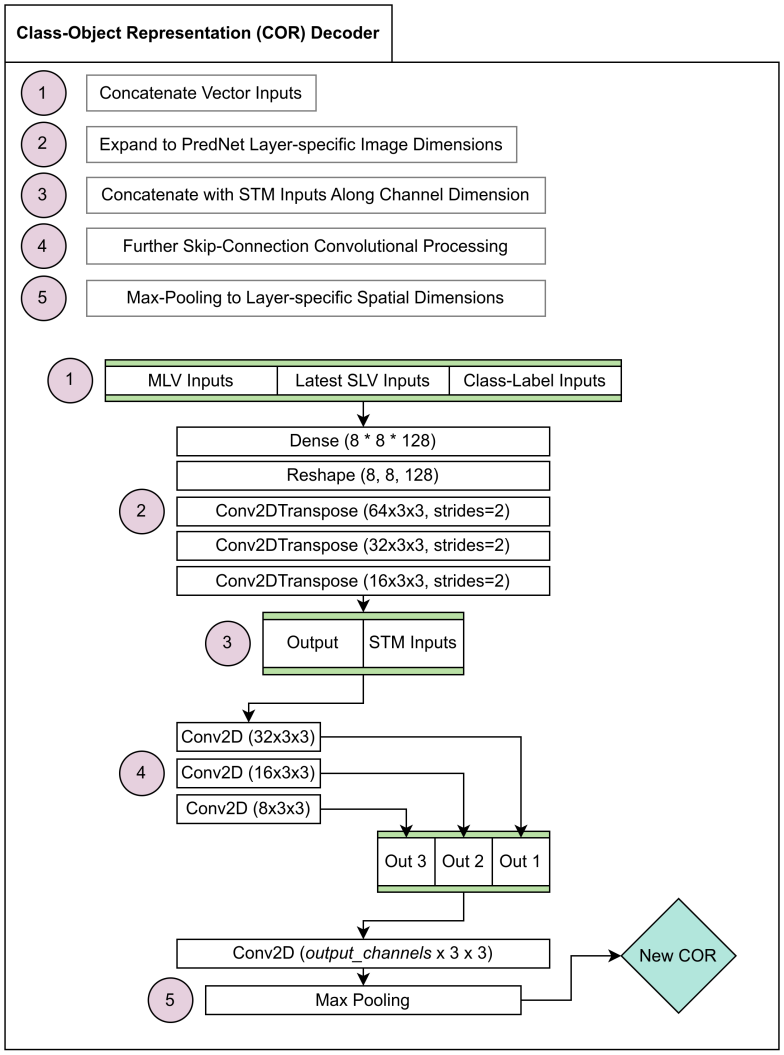


Figure 29: Class-Object Representation Decoder

In summary, steps 2 through 5 store short-term and long-term class-specific spatio-temporal memories, and convert these memories into a local-time-contextualized tensor representing what it means to be that class, right now. These are then passed to each PredNet layer's Representation units as additional inputs to inform predictive-coding representation and prediction formation.

Decomposed Predictions and Decomposed Learning So, now that each PredNet layer has been informed with class-object representations, we now look at the final steps which follow the standard predictive-coding prediction-error minimization scheme. The only difference now is that the Representation units at the bottom layer produce a representation tensor that is expected to be used by the bottom-layer prediction unit to produce a 12-channel tensor, instead of a 3-channel one. Again, this 12-channel tensor is just the predicted next-frames for the four, distinct class-objects in the frame. Then, as discussed earlier, our target images are also a 12-channel stack of frames. With these predictions and targets in hand, we can then calculate prediction error for each class-object, and pass that back to the representation units, and up the hierarchy, to start the cycle all over again, aiming to minimize class-specific prediction errors over time. See Figure 20 for an example of training progress for the Object-Centric PredNet. Note that in the figure, every three rows is structured as

[Predictions, Targets, Raw Error] for the four class-objects, except for the last two rows, which show the predicted and target composite frames.

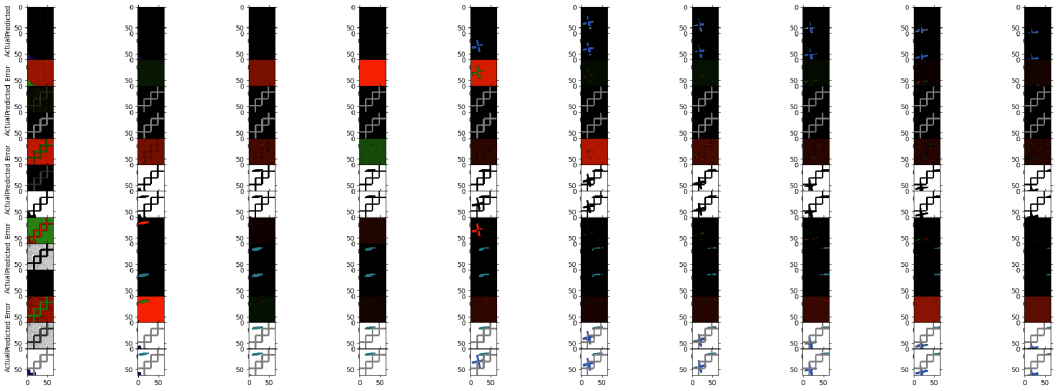


Figure 30: Training Progress for Object-Centric PredNet

In reality, we have to consider how to form the loss function for the decomposed frames. We have a few options. We could (A) measure error only for the decomposed frames, or (B), measure error for both the decomposed frames and the composite reconstructed frame, or (C), only measure error for the composite reconstructed frame. We experimented with all three, and results were somewhat similar across the board and time did not allow putting a finer point on which loss function was superior. The final test results published in the next section use option (B) with the following weighting: total_prediction_errors = 0.1 * reconstructed_frame_error + decomposed_frame_errors

5.4.6 Discussion of Results

Here we will discuss the results from our Object-Centric PredNet in comparison to the Baseline PredNet. In short, the object-centric model did not perform as well as the smaller and much-quicker baseline PredNet model. This was disappointing. It is unclear whether the results are due to an imperfection in the implementation as described, or due to a fault of logic for how these information flows are being managed. This is discussed further in the next section. In Figures 31, 32, and 33, we see the comparative visual performance between the Baseline and Object-Centric models. This test was performed as a set of 3000 image-pair prediction tasks. Note that the Object-Centric PredNet utilizes manual scene decomposition, and so the image borders needed to be removed to allow fully solid-colored objects. And in Table 3, we can see the quantitative comparison for parameter counts and average prediction MSE as compared to using the previous frames as predictions.

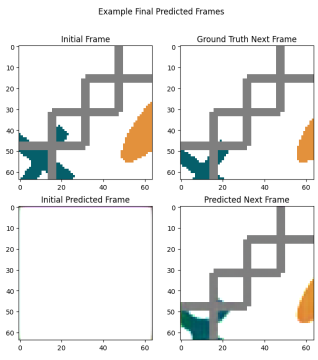


Figure 31: Baseline Results

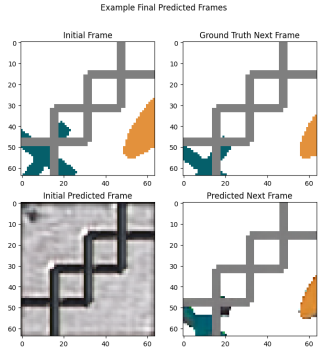


Figure 32: Object-Centric Results

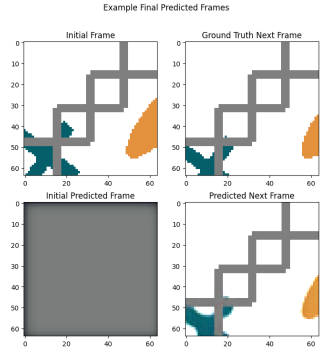


Figure 33: Double Baseline Results

ID #	# of Layers	# of Channels	# of Parameters	Average MSE
PF	-	-	-	0.03456
B	4	3, 48, 96, 192	6,915,948	0.01018
OC	4	3, 48, 96, 192	22,587,873	0.01281
DB	2	3, 96, 192, 384	35,382,804	**0.00992**

Table 3: PredNet Improvement Evaluation Results: (PF) - Previous Frame as Prediction, (B) - Baseline PredNet, (OC) - Object-Centric PredNet, (DB) - Double-Channel Counts for Baseline PredNet

5.4.7 Discussion of possible reasons why the approach is not out-performing baseline

We feel strongly that the sVAE-encoded sequence latent vectors are an efficient and informative structure to encode a spatial and temporal observation-memory of a particular object-class over time. However, upon investigation into the structure of the encoded sequence latent vectors, we see that evidence for "disentangled" structure is scant. According to Higgins et al. (2017), setting a hyper-parameter for a constant factor, beta, applied to the KL-divergence loss term can help control to what degree the latent dimensions are disentangled. We trained our sequence-VAE with a beta value of 0.25, in order to produce more accurate reconstructions. The authors note that beta values above 1.0 will tend to encode a more-disentangled latent space, while beta values below 1.0 will allow the VAE to utilize more of the latent dimensions to encode the various aspects of the input sequence, potentially leading to over-fitting and inferior latent space disentanglement. In a future work, we would like to try training with a higher beta value, above 1.0, and possibly also a lower dimensionality of the latent space, to see if we cannot encourage a more-discrete encoding of the input sequence characteristics into the latent dimensions. See Figure 34 to see how varying the latent dimensions independently changes the decoded output sequence. Note that the decoder produces a pair of images, pertaining to the input sequence, and that these are displayed both between the red bars. The base vector used to produce the plot is displayed down the middle column. Then, to the left and right, we see how varying the individual latent dimensions of this base vector somewhat changes the decoded output sequence. Admittedly, it is not the most informative plot at a glance, but if we look at latent dimension 20, for example, we see how varying this dimension alone tends to make the cross thinner or thicker, which is a valid aspect of the input sequences. Curiously, position and rotation seem completely absent from the plot, which implies that a combination of latent dimensions work in tandem to control these aspects. Recall that the sequence-VAE operates over binary masks, and so color is not encoded or decoded.

We feel the random-selection from stored and latest sequence latent vectors approach to maintaining a long-term memory may be non-ideal but perhaps not vastly inferior to an explicit diversity-maximizing approach. Explicit diversity maximization may be too extreme anyhow, storing sequence latent vectors associated with shapes at the edge of the screen, etc. It is left as future-work to determine the superior method.

We feel that forming our meta-latent vector for class-object representation decoding as the average of the long-term memory stored sequence latent vectors is a reasonable approach considering the hypothetical consistent encoded structure outputted from variational autoencoders. However, as noted, the actual structural encoding from our sequence-VAE is rather weak, and so it may be that only when the sequence latent vector structural encoding is improved that we can then expect average sequence latent vectors to be informative. For example, in a weakly-structured, or "entangled", space, the latent dimensions will tend to affect multiple aspects of the input sequence simultaneously. When we then average these sequence latent vectors, we can get unpredictable results because the combined effects of multiple entangled dimensions are not linear or simple to interpret. The averaged vector may not represent a meaningful "average" of the features but rather a mix of entangled and possibly conflicting influences, leading to distorted or uninterpretable decoded sequences. This is noted as a major implementation flaw, possibly being the single issue preventing better-than-baseline performance.

We also feel that the object-representation decoder is a likely bottleneck in the process. While the sequence-VAE is pre-trained to produce informative sequence latent vectors, the object-

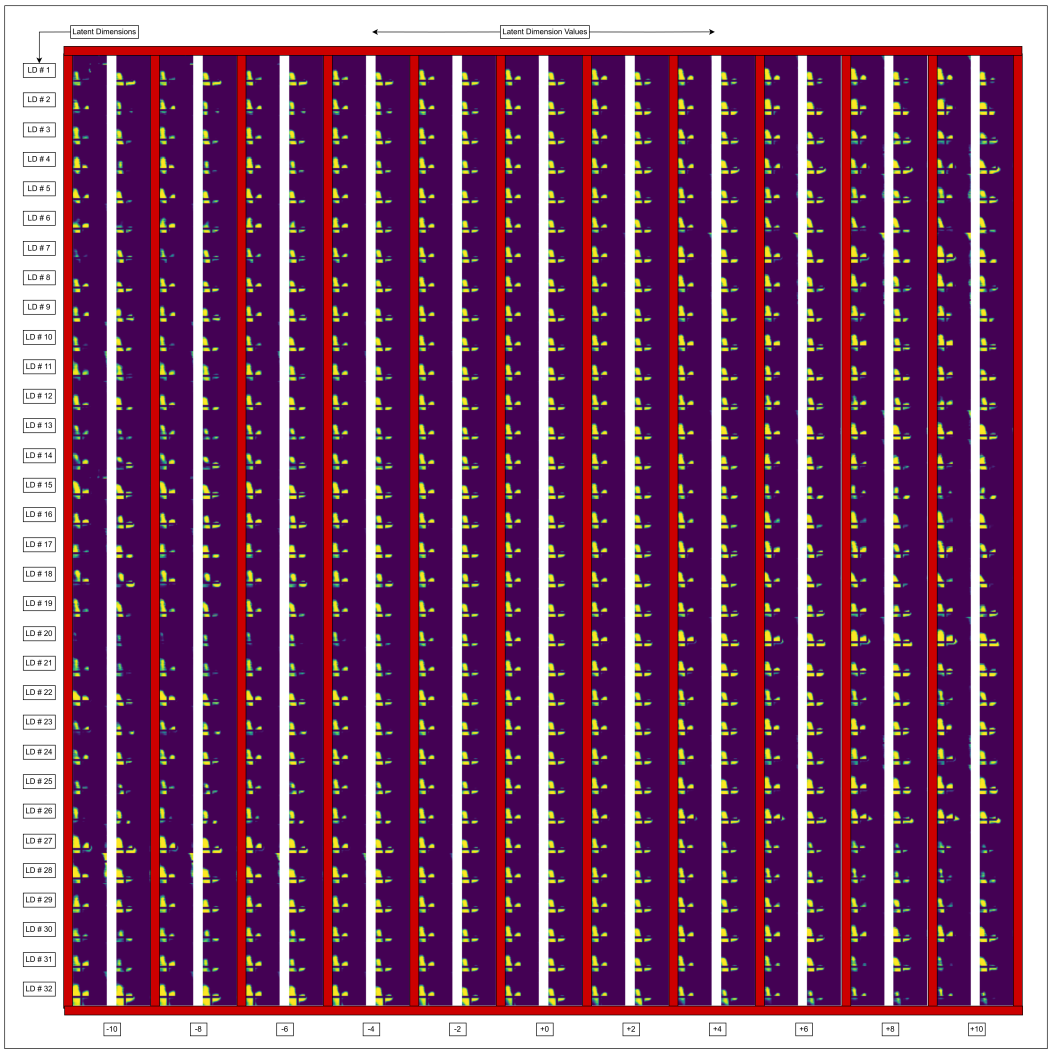


Figure 34: Sequence VAE Latent-Space Traversal

representation decoder is only trained via gradients backpropagated from the PredNet predictive-coding prediction-error-minimization scheme. Through this scheme, we hope that the object-representation decoder learns to produce informative and contextualized class-object representations in a format interpretable to the PredNet Represenation units. Had the results from our modified PredNet been superior to baseline, a number of investigations could have been performed to prove that the object representations were being used and positively contributing to the superior prediction results. For example, we could look to the weights of the PredNet-layer Representation units' convolutional layer kernels that pull in the information from the passed-in class-object representations. If these weights are of equal or greater magnitude to the weights applied to the other Representation-unit inputs, then we could confirm that the class-object representations are contributing to the final representation tensors for prediction. Additionally, we could have zeroed-out these class-object representations at each PredNet-layer, and show degraded performance. Finally, we must note that no aspect of the input tensors to the Representation units are abstract; the ConvLSTM internal to each Representation unit is simply a weighted sum of singular 2D convolutional layers applied to the inputs and previous representation tensors. Then, this tensor passes one more Prediction 2D convolutional layer to be transformed into the final prediction images. A 2D convolutional layer has no means to transform abstract data into an image, rather a 2D convolutional layer act simply as a filter over the input "image", regardless of how many input channels there are. All of this is to say that, the class-object representations formed should essentially be images of the object-classes,

generalized over long-term observations, and contextualized to the current class object's form and position. And so we should be able, at the bottom-layer, to plot the class-object tensors and see that the representations visually resemble the subsequent predictions made. The results from this final investigation at shown in Figure 35. Unfortunately, they show only a weak connection to the cross and ellipse shapes from the SSM dataset the COR decoder was trained on.

5.4.8 Discussion of the possibility that unrestricted PredNet may be superior

Having noted that the baseline PredNet is out-performing our object-centric PredNet, we would like to discuss one possible interpretation of this result. The baseline PredNet is largely unconstrained and general. It applies predictive coding over spatial tensors to produce tensors similar to the incoming tensors. There is little explicit manipulation of information except to connect the hierarchical layers, and to feed prediction error and the previous representation tensor back into the Representation unit to enable recurrence and produce the next representation tensor. This general approach works remarkably well and brings into question that, if improved performance is all we are after, and we are willing to increase parameter count (our object-centric PredNet has over 3x as many parameters as the baseline PredNet), then perhaps simply making the baseline PredNet larger is the correct way to go. Is there some emergent performance expected as we allow PredNet to filter out more and more information from the error signals (via increasing channel counts), or by allowing PredNet to form ever more abstract representations of the overall frame sequences (by increasing layer count)?

This was an interesting question and we ran a test to test the baseline with double channel counts. We chose not to test increasing the number of layers, because the spatial resolution at the top layer was already only 8x8 considering our 64x64 input images. So, further layers would be 4x4, 2x2, and finally 1x1. While this could be interesting, due to time-constraints we only looked at increased channel count. See Table 3 for the results. As is clear, the baseline PredNet may already be operating near the limits of performance, at least with respect to our dataset. There is only so much information to be filtered out of an image with solid-colored 2D shapes, so this seems reasonable. Perhaps with real-world images, the double-baseline results would show a stronger margin over baseline.

5.4.9 Discussion of possible areas for model improvement

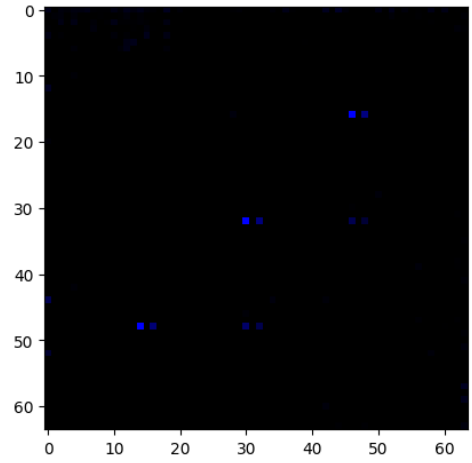
It would be advantageous to improve the performance of the sequence VAE. During development, we struggled to implement a sequence VAE that produced quality reconstructions with a disentangled latent space. Possibly the Sequence VAE developed by Zhu et al. (2020), albeit more complicated, would produce better sequence latent vectors.

6 Conclusion

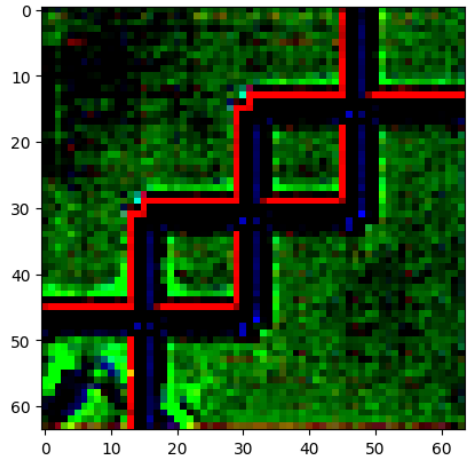
This paper and its underlying research aimed to develop a method for testing unsupervised video-prediction models for human-like abilities. Specifically, it focused on evaluating the models' capacity for structure learning related to the perception of objects, classes, and states, the association of behaviors with specific classes, and the formation of predictions based on these perceptions and associations.

Despite a thorough literature review of existing datasets and testing methodologies, an existing means to test for these abilities in an unsupervised manner could not be found. As a result, a new Simple Shape Motion (SSM) dataset and corresponding set of success criteria was created that allow an unsupervised video-prediction model to prove by demonstration that it can perform these abilities.

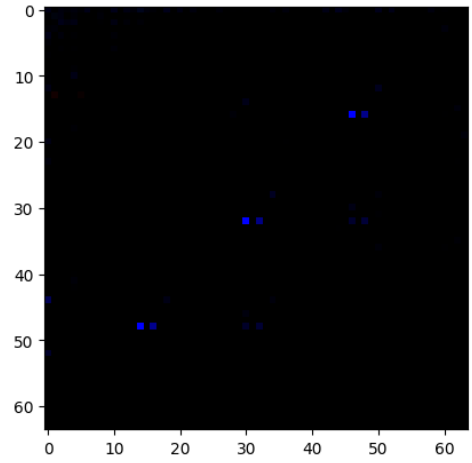
This general and widely-applicable dataset was then applied to the well-known PredNet model based on the leading neuroscience theory for how the mind learns structure in the world, called predictive coding. Based on meeting the defined success criteria, we conclude that PredNet is capable of performing structure learning of objects, classes, and class-specific behaviors, including those conditioned on both class-state and world-state, at a rudimentary level. This conclusion offers additional support for the neuroscience theory of predictive coding. In addition to this main conclusion for PredNet, we also present ablation studies that reveal that both the depth of hierarchy, and the number of representational filters, contribute significantly to model performance, which serves to



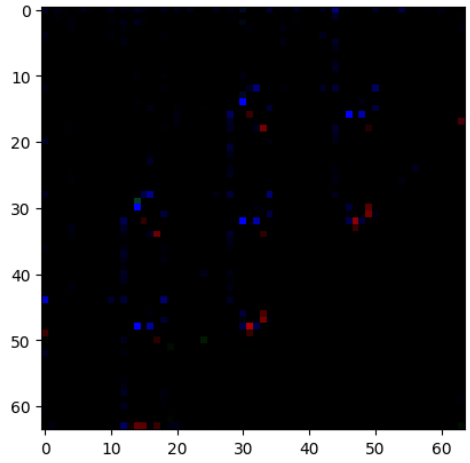
(a) Object Representation Tensor in Bottom-Layer, Class 0



(b) Object Representation Tensor in Bottom-Layer, Class 1



(c) Object Representation Tensor in Bottom-Layer, Class 2



(d) Object Representation Tensor in Bottom-Layer, Class 3

Figure 35: Plotting Bottom-Layer Class-Object Representation Tensors

guide future predictive coding model improvements. We also present model investigations based on convolutional neural networks that fail to decisively reveal class-specific recognition or behavioral associations, further justifying the need for proof-by-demonstration tasks via the SSM dataset or similar.

Lastly, we attempted to improve PredNet by fundamentally changing the data over which it learns to identify structure in, to include object-centric representations but the results were inferior to baseline. A major noted implementation flaw may be the single culprit prevent superior performance.

ACKNOWLEDGMENTS

Thanks to TU Delft and Martijn Wisse

REFERENCES

Processing. https://processing.org/. Accessed: May 23, 2024.

Louis Annabi, Alexandre Pitti, and Mathias Quoy. On the relationship between variational inference and auto-associative memory. (arXiv:2210.08013), October 2022. doi: 10.48550/arXiv.2210.08013. URL http://arxiv.org/abs/2210.08013. arXiv:2210.08013 [cs].

Matt E. M. Bower. Do we visually experience objects' occluded parts? *Canadian Journal of Philosophy*, 51(4):239–255, May 2021. ISSN 0045-5091, 1911-0820. doi: 10.1017/can.2021.22.

Christopher P. Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. (arXiv:1901.11390), January 2019. URL http://arxiv.org/abs/1901.11390. arXiv:1901.11390 [cs, stat].

Chang Chen, Fei Deng, and Sungjin Ahn. Roots: Object-centric representation and rendering of 3d scenes. (arXiv:2006.06130), July 2021. doi: 10.48550/arXiv.2006.06130. URL http://arxiv.org/abs/2006.06130. arXiv:2006.06130 [cs, stat].

Sebastien Ehrhardt, Aron Monszpart, Niloy Mitra, and Andrea Vedaldi. Unsupervised intuitive physics from visual observations. (arXiv:1805.05086), March 2019. doi: 10.48550/arXiv.1805.05086. URL http://arxiv.org/abs/1805.05086. arXiv:1805.05086 [cs].

Patrick Emami, Pan He, Sanjay Ranka, and Anand Rangarajan. Efficient iterative amortized inference for learning symmetric and disentangled multi-object representations. (arXiv:2106.03630), June 2021. URL http://arxiv.org/abs/2106.03630. arXiv:2106.03630 [cs].

S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. (arXiv:1603.08575), August 2016. doi: 10.48550/arXiv.1603.08575. URL http://arxiv.org/abs/1603.08575. arXiv:1603.08575 [cs].

S. M. Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, David P. Reichert, Lars Buesing, Theophane Weber, Oriol Vinyals, Dan Rosenbaum, Neil Rabinowitz, Helen King, Chloe Hillier, Matt Botvinick, Daan Wierstra, Koray Kavukcuoglu, and Demis Hassabis. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, June 2018. doi: 10.1126/science.aar6170.

Teresa Farroni, Antonio M. Chiarelli, Sarah Lloyd-Fox, Stefano Massaccesi, Arcangelo Merla, Valentina Di Gangi, Tania Mattarello, Dino Faraguna, and Mark H. Johnson. Infant cortex responds to other humans from shortly after birth. *Scientific Reports*, 3(11):2851, October 2013. ISSN 2045-2322. doi: 10.1038/srep02851.

Maria Laura Filippetti, Mark H. Johnson, Sarah Lloyd-Fox, Danica Dragovic, and Teresa Farroni. Body perception in newborns. *Current Biology*, 23(23):2413–2416, December 2013. ISSN 0960-9822. doi: 10.1016/j.cub.2013.10.017.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. (arXiv:1605.07157), October 2016. URL http://arxiv.org/abs/1605.07157. arXiv:1605.07157 [cs].

Peter A. Frensch and Dennis Rünger. Implicit learning. *Current Directions in Psychological Science*, 12(1):13–18, February 2003. ISSN 0963-7214. doi: 10.1111/1467-8721.01213.

Mingqi Gao, Feng Zheng, James J. Q. Yu, Caifeng Shan, Guiguang Ding, and Jungong Han. Deep learning for video object segmentation: a review. *Artificial Intelligence Review*, 56(1):457–531, January 2023. ISSN 1573-7462. doi: 10.1007/s10462-022-10176-7.

Eugene Gover and Nishan Krikorian. Determinants and the volumes of parallelotopes and zonotopes. *Linear Algebra and its Applications*, 433(1):28–40, July 2010. ISSN 0024-3795. doi: 10.1016/j.laa.2010.01.031.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. -vae: Learning basic visual concepts with a constrained variational framework. 2017.

Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng Chau. Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. (arXiv:1904.02323), September 2019. doi: 10.48550/arXiv.1904.02323. URL http://arxiv.org/abs/1904.02323. arXiv:1904.02323 [cs].

Michael Hoy, Zhigang Tu, Kang Dang, and Justin Dauwels. Learning to predict pedestrian intention via variational tracking networks. pp. 3132–3137, November 2018. doi: 10.1109/ITSC.2018.8569641.

Christopher R. Hoyt and Art B. Owen. Probing neural networks with t-sne, class-specific projections and a guided tour. (arXiv:2107.12547), July 2021. URL http://arxiv.org/abs/2107.12547. arXiv:2107.12547 [cs].

Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Niebles. Learning to decompose and disentangle representations for video prediction. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/496e05e1aea0a9c4655800e8a7b9ea28-Abstract.html.

Yanping Huang and Rajesh P. N. Rao. Predictive coding. *WIREs Cognitive Science*, 2(5):580–593, 2011. ISSN 1939-5086. doi: 10.1002/wcs.142.

Miguel Jaques, Timothy Hospedales, and Michael Burke. Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. 2020.

Beibei Jin, Yu Hu, Qiankun Tang, Jingyu Niu, Zhiping Shi, Yinhe Han, and Xiaowei Li. Exploring spatial-temporal multi-frequency analysis for high-fidelity and temporal-consistency video prediction. (arXiv:2002.09905), May 2020. URL http://arxiv.org/abs/2002.09905.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes, December 2013. URL https://arxiv.org/abs/1312.6114v11.

Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models. (arXiv:1406.5298), October 2014. doi: 10.48550/arXiv.1406.5298. URL http://arxiv.org/abs/1406.5298.

Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. (arXiv:1911.12247), January 2020. doi: 10.48550/arXiv.1911.12247. URL http://arxiv.org/abs/1911.12247. arXiv:1911.12247 [cs, stat].

Adam R. Kosiorek, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh. Sequential attend, infer, repeat: Generative modelling of moving objects. (arXiv:1806.01794), November 2018. URL http://arxiv.org/abs/1806.01794. arXiv:1806.01794 [cs, stat].

Ananya Kumar, S. M. Ali Eslami, Danilo J. Rezende, Marta Garnelo, Fabio Viola, Edward Lockhart, and Murray Shanahan. Consistent generative query networks. (arXiv:1807.02033), April 2019. URL http://arxiv.org/abs/1807.02033. arXiv:1807.02033 [cs, stat].

Marcus Lewis, Scott Purdy, Subutai Ahmad, and Jeff Hawkins. Locations in the neocortex: A theory of sensorimotor object recognition using cortical grid cells. *Frontiers in Neural Circuits*, 13:22, April 2019. ISSN 1662-5110. doi: 10.3389/fncir.2019.00022.

Zhixuan Lin, Yi-Fu Wu, Skand Peri, Bofeng Fu, Jindong Jiang, and Sungjin Ahn. Improving generative imagination in object-centric world models. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 6140–6149. PMLR, November 2020. URL https://proceedings.mlr.press/v119/lin20f.html.

Grace W. Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in Computational Neuroscience*, 14, April 2020. ISSN 1662-5188. doi: 10.3389/fncom.2020.00029. URL https://www.frontiersin.org/journals/computational-neuroscience/articles/10.3389/fncom.2020.00029/full.

William Lotter, Gabriel Kreiman, and David Cox. Unsupervised learning of visual structure using predictive generative networks. (arXiv:1511.06380), January 2016. URL http://arxiv.org/abs/1511.06380. arXiv:1511.06380 [cs, q-bio].

William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. (arXiv:1605.08104), February 2017. doi: 10.48550/arXiv.1605.08104. URL http://arxiv.org/abs/1605.08104. arXiv:1605.08104 [cs, q-bio].

Wei-Chiu Ma, De-An Huang, Namhoon Lee, and Kris M. Kitani. Forecasting interactive dynamics of pedestrians with fictitious play. (arXiv:1604.01431), March 2017. doi: 10.48550/arXiv.1604.01431. URL http://arxiv.org/abs/1604.01431. arXiv:1604.01431 [cs].

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, February 2018. URL https://arxiv.org/abs/1802.03426v3.

Beren Millidge, Anil Seth, and Christopher L. Buckley. Predictive coding: a theoretical and experimental review. (arXiv:2107.12979), July 2022. URL http://arxiv.org/abs/2107.12979. arXiv:2107.12979 [cs, q-bio].

Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin P Murphy, and Honglak Lee. Unsupervised learning of object structure and dynamics from videos. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/d82c8d1619ad8176d665453cfb2e55f0-Abstract.html.

Li Nanbo, Cian Eastwood, and Robert Fisher. Learning object-centric representations of multi-object scenes from multiple views. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5656–5666. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3d9dabe52805a1ea21864b09f3397593-Abstract.html.

Li Nanbo, Muhammad Ahmed Raza, Hu Wenbin, Zhaole Sun, and Robert B. Fisher. Object-centric representation learning with generative spatial-temporal factorization. (arXiv:2111.05393), November 2021. doi: 10.48550/arXiv.2111.05393. URL http://arxiv.org/abs/2111.05393. arXiv:2111.05393 [cs].

Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John Alejandro Castro-Vargas, Sergio Orts-Escolano, Jose Garcia-Rodriguez, and Antonis Argyros. A review on deep learning techniques for video prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):2806–2826, June 2022. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2020.3045007.

Wiebke Pätzold and Ulf Liszkowski. Pupillometric voe paradigm reveals that 18- but not 10-month-olds spontaneously represent occluded objects (but not empty sets). *PLOS ONE*, 15(4):e0230913, April 2020. ISSN 1932-6203. doi: 10.1371/journal.pone.0230913.

Rajesh P. N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, January 1999. ISSN 1097-6256, 1546-1726. doi: 10.1038/4580.

Amir Rasouli. Deep learning for vision-based prediction: A survey. (arXiv:2007.00095), July 2020. URL http://arxiv.org/abs/2007.00095. arXiv:2007.00095 [cs].

Sebastian Ruder. An overview of multi-task learning in deep neural networks. (arXiv:1706.05098), June 2017. doi: 10.48550/arXiv.1706.05098. URL http://arxiv.org/abs/1706.05098. arXiv:1706.05098 [cs, stat].

Changjiang Shi, Zhijie Zhang, Wanchang Zhang, Chuanrong Zhang, and Qiang Xu. Learning multiscale temporal–spatial–spectral features via a multipath convolutional lstm neural network for change detection with hyperspectral images. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–16, 2022. ISSN 0196-2892, 1558-0644. doi: 10.1109/TGRS.2022.3176642.

Xingjian SHI, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun WOO. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. (arXiv:1409.1556), April 2015. URL http://arxiv.org/abs/1409.1556.

Gautam Singh, Jaesik Yoon, Youngsung Son, and Sungjin Ahn. Sequential neural processes. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/110209d8fae7417509ba71ad97c17639-Abstract.html.

Giulia Slavic, Abrham Shiferaw Alemaw, Lucio Marcenaro, and Carlo Regazzoni. Learning of linear video prediction models in a multi-modal framework for anomaly detection. In *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 1569–1573, September 2021. doi: 10.1109/ICIP42928.2021.9506049. URL https://ieeexplore-ieee-org.tudelft.idm.oclc.org/abstract/document/9506049.

Nagabhushan Somraj, Manoj Surya Kashi, S. P. Arun, and Rajiv Soundararajan. Understanding the perceived quality of video predictions. *Signal Processing: Image Communication*, 102:116626, March 2022. ISSN 09235965. doi: 10.1016/j.image.2021.116626. arXiv:2005.00356 [cs, eess].

Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. (arXiv:1502.04681), January 2016. URL http://arxiv.org/abs/1502.04681.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. (arXiv:1605.07736), October 2016. doi: 10.48550/arXiv.1605.07736. URL http://arxiv.org/abs/1605.07736. arXiv:1605.07736 [cs].

Michael E Tipping and Christopher M Bishop. Mixtures of probabilistic principal component analysers.

David C. Van Essen and John H. R. Maunsell. Hierarchical organization and functional streams in the visual cortex. 6:370–375, January 1983. ISSN 0166-2236. doi: 10.1016/0166-2236(83)90167-4.

Sjoerd van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. (arXiv:1802.10353), February 2018. URL http://arxiv.org/abs/1802.10353. arXiv:1802.10353 [cs].

Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. (arXiv:1910.03151), April 2020. URL http://arxiv.org/abs/1910.03151. arXiv:1910.03151 [cs].

Nicholas Watters, Loic Matthey, Matko Bosnjak, Christopher P. Burgess, and Alexander Lerchner. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. (arXiv:1905.09275), August 2019. URL http://arxiv.org/abs/1905.09275. arXiv:1905.09275 [cs].

Chenliang Xu, Shao-Hang Hsieh, Caiming Xiong, and Jason J. Corso. Can humans fly? action understanding with multiple classes of actors. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2264–2273, Boston, MA, USA, June 2015. IEEE. ISBN 978-1-4673-6964-0. doi: 10.1109/CVPR.2015.7298839. URL http://ieeexplore.ieee.org/document/7298839/.

Zhenjia Xu, Zhijian Liu, Chen Sun, Kevin Murphy, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Unsupervised discovery of parts, structure, and dynamics. (arXiv:1903.05136), March 2019. doi: 10.48550/arXiv.1903.05136. URL http://arxiv.org/abs/1903.05136. arXiv:1903.05136 [cs].

Wilson Yan, Danijar Hafner, Stephen James, and Pieter Abbeel. Temporally consistent transformers for video generation. (arXiv:2210.02396), May 2023. doi: 10.48550/arXiv.2210.02396. URL http://arxiv.org/abs/2210.02396. arXiv:2210.02396 [cs].

Rui Yao, Guosheng Lin, Shixiong Xia, Jiaqi Zhao, and Yong Zhou. Video object segmentation and tracking: A survey. *ACM Transactions on Intelligent Systems and Technology*, 11(4):36:1–36:47, May 2020. ISSN 2157-6904. doi: 10.1145/3391743.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. (arXiv:1311.2901), November 2013. doi: 10.48550/arXiv.1311.2901. URL http://arxiv.org/abs/1311.2901. arXiv:1311.2901 [cs].

Yizhe Zhu, Martin Renqiang Min, Asim Kadav, and Hans Peter Graf. S3vae: Self-supervised sequential vae for representation disentanglement and data generation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6537–6546, Seattle, WA, USA, June 2020. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.00657. URL https://ieeexplore.ieee.org/document/9157480/.

A Appendix

A.1 Model Investigations

The following model investigations were performed in order to better understand how PredNet is forming object, class, and associated class-specific behavior representations. Inspiration for these model investigations comes from the literature for investigating convolutional neural networks. Each subsection pertaining to the below list of investigations will introduce the methodology and source.

1. Section A.1.1 - Top-activated channels by mean and STD
2. Section A.1.2 - Top-activated channels by aggregation
3. Section A.1.3 - Predictions per restricted top-down influence
4. Section A.1.4 - Dimensionality reduction for convolutional filter weights
5. Section A.1.5 - Dimensionality reduction for convolutional feature maps
6. Section A.1.6 - Images to maximally-activate convolutional filters via optimization

A.1.1 TOP-ACTIVATED CHANNELS BY MEAN AND STD

In this investigation, the mean and standard deviation for the global-max-pooled activation for each representation and prediction channel, in each PredNet layer, are computed over 5000 instances each of cross and ellipse class-objects presented to the trained model from Task 1, introduced in Section 3.2.1. The intent was to identify a subset of channels within each layer that primarily activate for either one or the other class. For example, if only 15 of the 192 channels in the top layer representation tensor activate strongly for crosses, while a different 15 channels activate strongly for ellipses then this would indicate a learned, fairly sparse, distributed representation activating for each class, aka semi-symbolic learning as discussed in the introduction. As shown in Figure 36, at the highest layer, there does appear to be some class-specific channel activations. However, as shown in Figure 39, forming predictions based on these class-specific channels alone does not produce images that clearly portray either a cross or an ellipse. As a result, we conclude that while semi-symbolic learning does occur for class recognition, a number of as-yet unidentified channel activations shared between classes also play a part in the semi-symbolic activation and predictions in recognition of a presented object's class.

Inspiration for this investigation came from both Numenta and their work regarding sparse-distributed representations for object recognition, (Lewis et al., 2019), and from SUMMIT, where a similar approach is applied to identify which channels in a layer most activate and represent each class in a model (Hohman et al., 2019). The approach used by SUMMIT is discussed further in Section A.1.2 below.

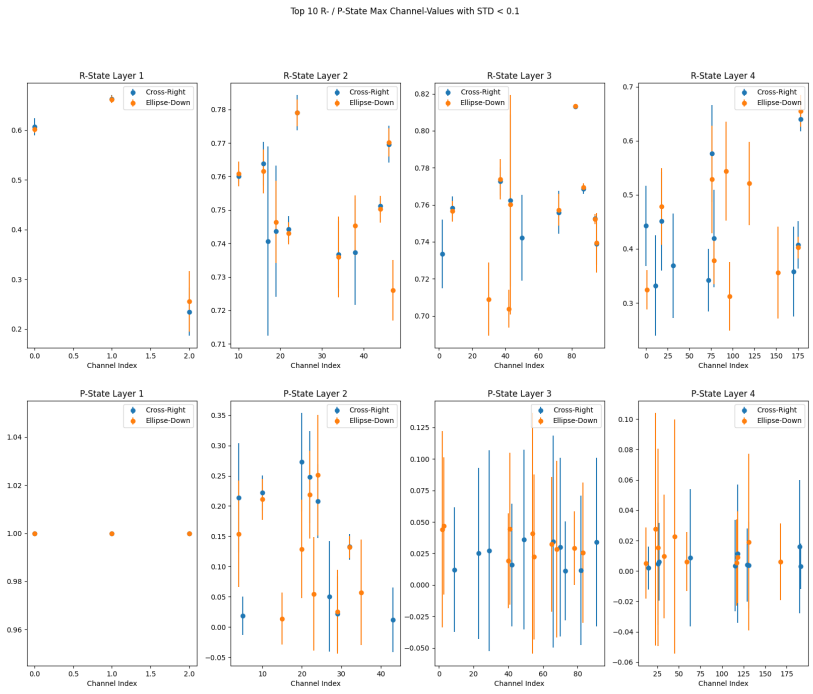


Figure 36: Top 10 Representation and Prediction Conv. Filters by Activation Mean and STD

A.1.2 TOP-ACTIVATED CHANNELS BY AGGREGATION

This investigation is similar to that of Section A.1.1 presented above, however, instead of calculating the mean and standard deviation of activations for each channel in response to presented cross and ellipse class-objects, we evaluate the channel influence on class recognition by means of an aggregated ranking. This aggregated ranking counts the number of times each channel was a top contributor by activation weight to the total activation response of all channels for each class. We

considered channels contributing to the top 10% ($k_{M2} = 0.1$) as aggregation winners. We refer to Section 6.1, Aggregation Method 2 in Hohman et al. (2019) for further details. The same 5000 instances of cross and ellipse class-objects each from Section A.1.1 above are presented to the trained model from Task 1. The resulting aggregation winner counts are displayed in Figure 37. We note that there is significant class-overlap again between the top channels activating in response to the two shape-classes. Therefore, in order to show more clearly the difference in channel activations per class, we also present the aggregation winner count *differences* in Figure 38.

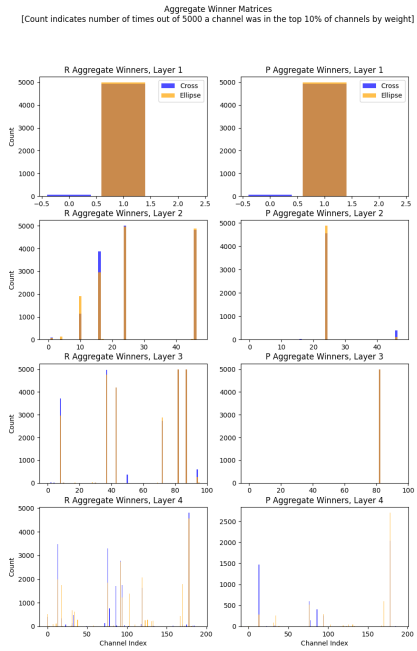


Figure 37: Aggregation Winners

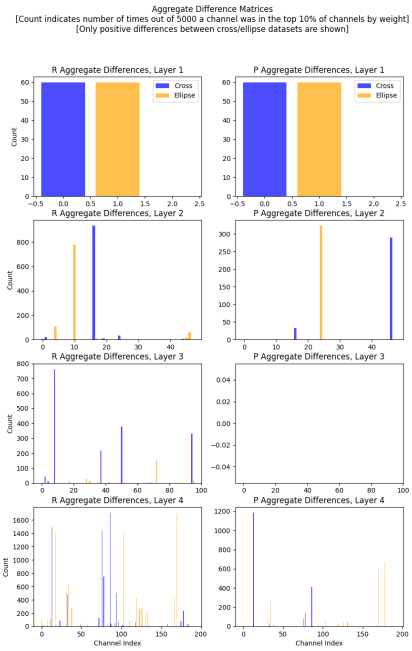


Figure 38: Aggregation Winner Differences

A.1.3 PREDICTIONS PER RESTRICTED TOP-DOWN INFLUENCE

This investigation involved three sub-investigations. First, we wanted to know how the activations from any single layer influenced the final predicted images. This is shown in the leftmost portion of Figure 39. Second, we wanted to know how the activations from any restricted subset of upper layers influenced the final predicted images. This is shown in the middle portion of Figure 39. Third, we investigate with the same goal as the middle portion of the image, except now we perform channel filtering on the top-layer channel activations. This means that the top 10 channels per class were identified from the investigation in Section A.1.1, and then at prediction time, prior to top-down influence, activations for all channels besides these top channels are reset to zero. The intent here is to determine if the top-channel-activations in the top layer for a specific class are sufficient to produce quality predictions. We focused on ellipse-shape activations only. This third sub-investigation is shown in the rightmost portion of Figure 39.

All results for this investigation are presented in Figure 39. Some explanation of the figure is required. In each of the three portions of the figure (left, middle, and top), we see paired rows, where the top and bottom row in each pair displays the average filter activations for the *Representation* and *Prediction* tensors in each layer, respectively. Then, within each row, the four image-squares show the representation and prediction tensors pertaining to the four PredNet layers, with the bottom layer on the left, up to the top layer on the right. Thus, in the green boxes, with all layers activated, we see a sharp prediction for the next frame in the bottom row, leftmost image-square, considering the input provided in the upper left. Please note that this investigation was carried out on a model trained on a previous version of the SSM dataset in which ellipses move downwards while crosses move to the right. Taking this into consideration, we see that the predicted image, then, is correct. Then, for

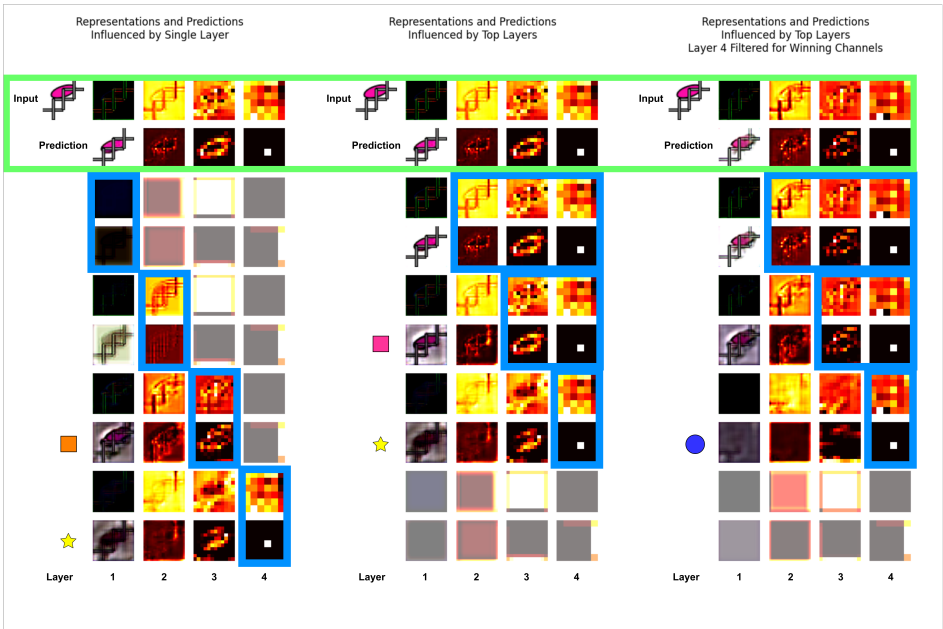


Figure 39: Representations and Predictions from Top-Down Influences

each of the three portions of the image, pertaining to the three sub-investigations mentioned above, the blue boxes indicate which layers are contributing to the lowest-layer's predictions. For example, in the leftmost portion of the image, where we see an orange square symbol, the layer 3 tensors are boxed in blue. This indicates that, after presentation of the input image, the representation and prediction tensors for all other layers besides layer 3 are cleared, and then top-down predictions are flowed down from layer 3 to produce the final predicted image. Finally, the faded image-squares represent upper layers that were reset to zero and not subsequently populated via top-down influence.

From sub-investigation one, we can conclude that the only single-layer influence able to produce a viable prediction is that from layer 4. This result is indicated by the yellow star symbol. While the prediction quality is certainly degraded, the prediction does appear to show a pink ellipse in the correct, downwards position. Considering this layer contains the most representation channels to activate and pass down as a top-down influence, this result is not altogether surprising. Single-layer influence from layers 1 and 2, on the other hand, appear to produce predictions that leave the input largely unchanged, albeit in significantly degraded image quality. Finally, single-layer influence from layers 3 (orange square symbol) is interesting as it seems to apply behavioral transformations somewhat between those for crosses and ellipses; the predicted ellipse shape appears stretched both rightwards and downwards.

From sub-investigation two, we can conclude that resetting activations for lower levels will degrade prediction image quality, with more degradation seen for the resetting of more lower levels. The final case, where only layer 4 is left to influence the predicted image, shows to lowest prediction image quality. We also note that this result is identical to that found in sub-investigation one, and so the same yellow star symbol is shown.

From sub-investigation three, in the green box, where only layer 4 has been filtered and no other layers have been reset, we see that predicted image quality has been severely degraded. Not only is image quality lower, but the discrete behavioral prediction appears to be mixed between downwards and rightwards motion as discussed above regarding the orange-square-symbol result for top-down influence only from layer 3. As we then progressively reset the lower layers, moving down the rightmost portion of the image towards the blue square, we see the prediction *concept* remain the same while image quality is further reduced.

In conclusion for this investigation, only the top layer, layer 4, appears to contain sufficient information to produce a viable prediction through top-down influence on its own. Based on the single-layer

influence results for layer 3, we see some ambiguity for which behavioral transformation to apply. We could speculate that perhaps layer 3 is performing some specialization for behavioral transformations, but without indication from layer 4, it cannot conclude which transformation to apply. However, this is quite speculative, and instead, based on the formed representations displayed for each layer, we conclude more reasonably that each layer is focused on the same goal of representing and predicting the same scene, albeit from a different perspective. These different perspectives are inherent to how predictive coding operates. Based on the inputs to each layer, we note that the bottom layer is seeking to model and predict for the ground-truth external reality, while the next layer up is seeking to model and predict for the prediction-error signal produced by the layer below. We see that these two input signals, ground-truth reality and prediction-error are fundamentally different signals in that information has been added (via bottom layer predictions) and removed (by taking the difference between ground-truth and predictions) to the ground-truth reality signal in order to form the prediction-error signal for the second layer. This prediction error signal then is a novel informational perspective describing an interaction between layer 1 and the environment, and so the second layer is really modeling how the bottom layer is interacting with some inferred environment. As we traverse the hierarchy up further, the abstraction continues, where layer 3 is now modeling how layer 2 behaves in response to layer 1 predictions and an unseen environment signal, and so on. Each of these perspectives, then, are learned to be passed down and interpreted to form an accurate prediction at the bottom layer.

Inspiration for this investigation came from our own intuition upon reflection of the presence of *some* class-specific channel activations as demonstrated in Section A.1.1.

A.1.4 Dimensionality reduction for convolutional filter weights

In this investigation, we aimed to perform dimensionality reduction on the convolutional filter weights to determine if the filters form clusters that can be clearly divided by class recognition. We utilized Principal Component Analysis (PCA) (Tipping & Bishop) and Uniform Manifold Approximation and Projection (UMAP). (McInnes et al., 2018) to reveal linear and non-linear clustering relationships, respectively. Using a trained PredNet model, the trained weights within each layer's Representation ConvLSTM convolutional filter gates, and within the Prediction convolutional filters, are gathered and flattened into vectors, one for each filter. A separate dimensionality reducer is fit to the filter vectors for each layer. Finally, the filter vectors are reduced via this fitted reducer. The results are shown in Figures 40 and 41 for PCA and UMAP, respectively. As is clearly evident, the filters, as defined as points in a high-dimensional space over its spatial and channel dimensions, do not form clear clusters based on the features each filter is specialized to respond to.

Inspiration for this investigation came from our own intuition upon reflection of the desire to determine if the model was producing class-specific specialized filters. Validation of this approach is found in the literature where Hoyt & Owen (2021) seek to investigate how neural networks separate classes in the outputs of various layers in popular convolutional neural networks.

A.1.5 Dimensionality reduction for convolutional feature maps

Similar to the dimensionality reduction investigation applied to the convolutional filter weights in Section A.1.4, in this investigation, we also attempt to reveal class-specific clustering behavior but this time from within the Representation and Prediction activated feature maps. We also only investigate the final Representation ConvLSTM output hidden state feature maps, instead of those for each convolutional gate as in Section A.1.4. These feature maps are produced by presenting 5000 examples each of cross and ellipse shapes to a trained PredNet model, producing a model activation (feature map) for each example shape. Thus, we are left with 10,000 total feature maps. Each feature map is a spatial tensor of shape ($height\ x\ width\ x\ channels$), where $height$, $width$, and $channels$ vary based on layer number. We then flatten each of these feature maps into a vector and fit a dimensionality reducer to the combined set of cross and ellipse feature maps. With the trained reducer, we then separately reduce the cross and ellipse feature maps so that we can color them differently in the plot and identify any clustering between classes. The result is shown in Figures 42 and 43 for PCA and UMAP, respectively. As the figures clearly show, the reduction from thousands of features in each map to just two does not produce visible clustering, preventing us from determining that any subset of convolutional filters in each layer responds more to one class over another. Potentially t-SNE or use of an autoencoder for dimensionality reduction could be more effective in

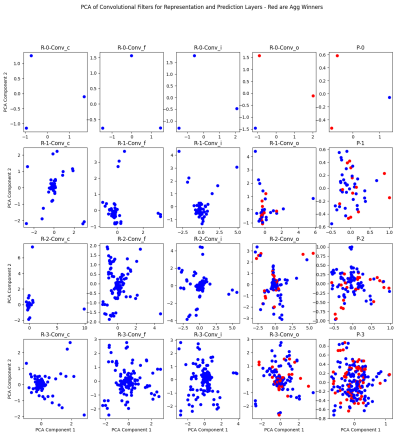


Figure 40: PCA Applied to Conv. Filters

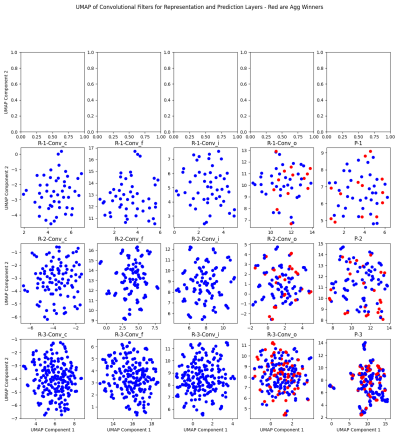


Figure 41: UMAP Applied to Conv. Filters

revealing non-linear clustering behavior between activated feature maps based on class, but this is left as future work.

Here as well, inspiration for this investigation came from our own intuition upon reflection of the desire to determine if the model was producing class-specific specialized filters, and validation of the approach is found in Hoyt & Owen (2021).

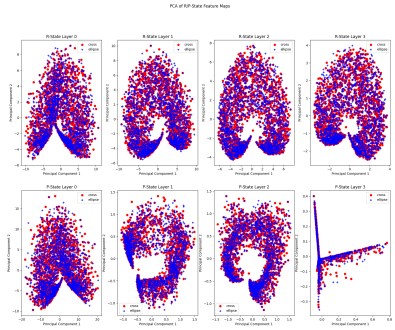


Figure 42: PCA Applied to Feature Maps

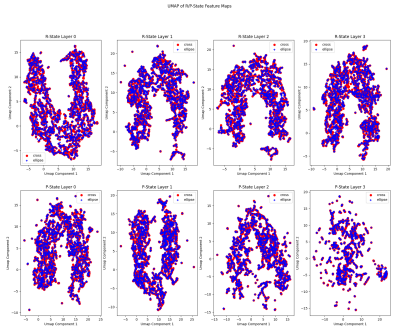


Figure 43: UMAP Applied to Feature Maps

A.1.6 IMAGES TO MAXIMALLY-ACTIVATE CONVOLUTIONAL FILTERS VIA OPTIMIZATION

In this investigation we seek to generate images that maximally activate the convolutional filters within a trained PredNet model. The generated images are formed through gradient ascent over each filter's activation magnitude. While we performed this execerise for each Representation and Prediction convolutional filter, we show only a single set of filters, for Representation ConvLSTM output filters in Layer 3. These are presented in Figure 44. Unfortunately, the images do not display any meaningful structures from which to derive explanations for the learned representations. This is also the case for the filters in the other layers.

Inspiration for this investigation comes from the literature regarding convolutional neural network interpretability investigations (Zeiler & Fergus, 2013).

A.2 ABLATION STUDY

In this ablation study, several PredNet ablation variants were compared to the baseline PredNet model to determine which parts of PredNet are crucial for its prediction accuracy. The study in-

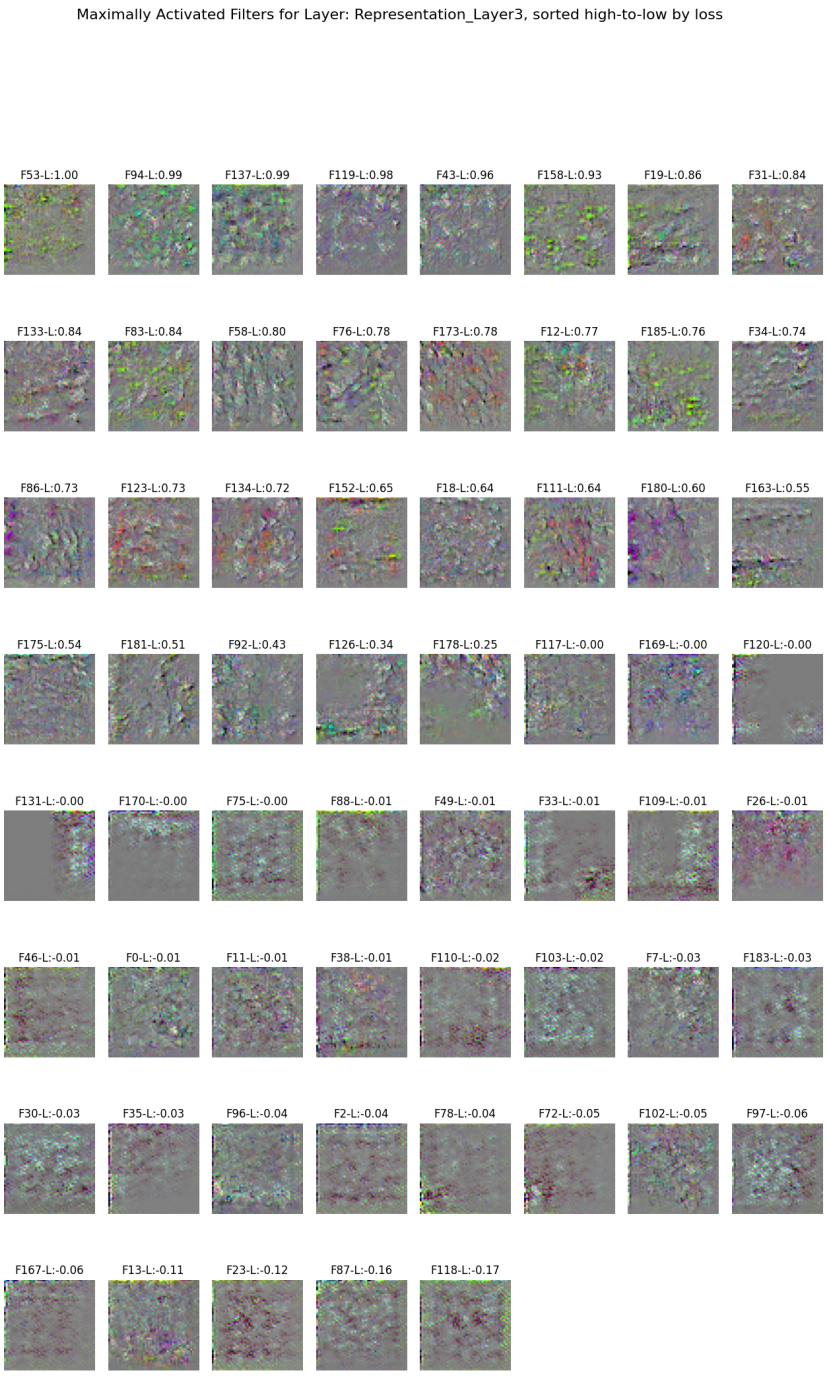


Figure 44: Input Images that Maximally Activate Layer 4 Conv. Filters

ID #	# of Layers	# of Channels	# R-CLSTM w/ ECA	# of Parameters	Average MSE
PF	-	-	-	-	0.10255
B	4	3, 48, 96, 192	-	6,915,948	**0.02325**
3	4	3, 48, 96, 192	2	11,673,799	0.03566
4	2	3, 336	-	13,266,348	0.04182
5	4	3, 12, 24, 48	-	435,084	0.05369
6	4	3, 12, 24, 48	3	1,033,398	0.06081
7	1	3	-	1,068	0.09705
8	1	339	-	4,220,892	0.09972

Table 4: Ablation Study Results: (PF) - Previous Frame as Prediction, (B) - Baseline Model

volved running each ablation-variant against the same multi-object quantitative-analysis experiment as described in Section 4.3. This produces a MSE value for each variant, representing the next-frame prediction accuracy in a series of single-step predictions.

To define the ablation-variants, we examined altering the number of layers and the number of channels in each layer. Additionally, per Shi et al. (2022), inspiration was taken to attempt to improve baseline performance by stacking multiple ConvLSTM units within the Representation layers. The motivation for this approach is to find patterns of multiple scales within each set of input data (sensory input images for the bottom layer, prediction errors for the layer below for the upper layers). So, in this approach, while the base ConvLSTM will seek to find patterns in and to model the input data for that layer, the ConvLSTM units stacked above will attempt to find patterns in and to model the hidden representations formed by the ConvLSTM unit below. The outputs from each ConvLSTM are concatenated channel-wise and then modulated via Efficient Channel Attention (ECA) per Wang et al. (2020). ECA is intended to weight the outputted representation channels per a learned 1-dimensional convolution of kernel length 3 applied through the channels at each spatial position. The approach seemed promising but either due to insufficient or improper training, or to the approach being ill-fitting or overkill for the task, the results were less accurate than the baseline.

The results for each ablation variant are displayed in Table 4. They are ordered (past Baseline) per ascending MSE. Note that the 'PF' and 'B' under ID # in the table refer to (PF) Previous-Frame as Prediction and (B) Baseline PredNet. Also note that the number of channels altogether in the Baseline PredNet is equal to $3 + 48 + 96 + 192 = 339$, and that is where the 336 and 339 channel counts come from, in an attempt to isolate the effect of only reducing layer count and not channel count.

From the ablation study results, it is evident that a hierarchical structure is necessary for optimal performance. A deeper hierarchy outperforms a shallower one, even when both have an equal number of representation channels, and when we note that the average spatial dimensions for the representation channels are higher for the shallower hierarchy. Additionally, increasing the number of channels while keeping the layer count constant also enhances performance. Finally, without further modifications to PredNet, or a different dataset, implementing hierarchical ConvLSTM's in the representation unit of each layer worsens performance while significantly increasing the number of model parameters, thus making this an undesirable model change.

A.3 Future Work

First, there are a number of other human-like abilities that could be tested:

1. **Instance recognition** Here we would like to evaluate how well a model can learn to distinguish and predict for unique instances of a common class. Thus, the model is expected to associate class-specific behaviors while also learning any unique aspect of the class-instances the model has encountered. See Figure 45 for an illustration. In this figure, instances of a rolling-circle class are indicated by a unique color and size combination. A model should learn the class-specific behavior for rolling-circles (that they roll along surfaces) while noting the different rolling speeds of the encountered instances. A human might say, "I have seen many rolling circles. I have also noticed that these rolling circles often roll at different speeds. Now, I see a new rolling circle, and I note the speed at which

it rolls. Next time I see this particular rolling circle, I will attempt to recall that it rolls at this speed." This human ability is useful to avoid over-generalization and stereotyping.

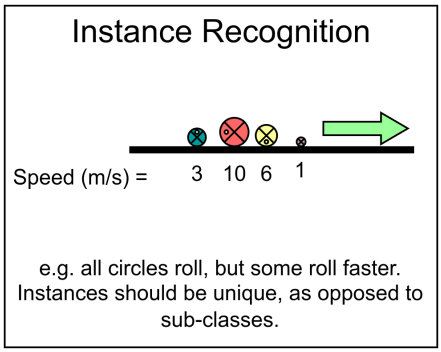


Figure 45: Instance Recognition

2. **Inter-class behaviors** This ability describes how humans can learn how different classes of objects interact. See Figure 46 for an illustration. The figure describes one possible test implementation where the inter-class behavior is shared for all classes, namely members of the same class repel each other while members of different classes attract. Alternatively, tests could implement unique inter-class behaviors for different pairs of objects. For example, a magnet and an iron nail will interact attractively while a magnet and a plastic ball will not interact at a distance. This human ability is useful to test for because the nature of object interactions is fundamental aspect of the structure of our world.

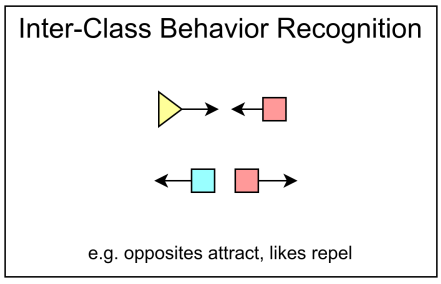


Figure 46: Inter-Class Behavior Recognition

3. **Global behaviors** Here we include behaviors shared by all classes in addition to the class-specific behaviors. See Figure 47 for an illustration where two examples are provided, namely, the response of all objects to gravity, and that of object permanence. To some degree the SSM dataset does test for object permanence in that previously-occluded portions of the objects do receive generative infilling, but never are the tracked objects fully occluded.

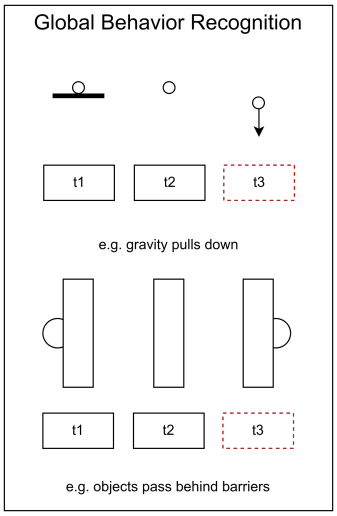


Figure 47: Global Behavior Recognition

4. **Long-term associations** Ideally, as humans navigate the world, we should store permanent knowledge for facts that do not generally change. For example, gravity is a fairly constant aspect of our experience in the world and thus we should always recall that letting go of an object in mid-air will result in the object falling away from my hand. See Figure 48 for an illustration of an example where a specific inter-class behavior is expected to be maintained over the duration of the operation of the model. This can be imagined as the interaction between a sharp pin and a balloon. Regardless of the history before these two objects collide, when they do collide, the pair initiates an event, that of a balloon popping.

   Associations need not pertain to a pair of objects. We can also remember values for properties of a single object at arbitrary periods in the future. I know that my bicycle is painted red and should be able to recall that fact when needed.

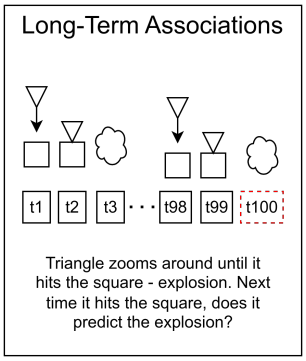


Figure 48: Long-Term Associations

5. **n-Shot learning** Here, we would like to evaluate how quickly a model can learn new classes or class-instances, and their associated behaviors. See Figure 49 for an example illustration related to the rolling motion of various polygons. For simple behaviors and objects, humans can perform this learning from a single viewing. From experience training PredNet on the SSM dataset, however, it was clear that roughly 40,000 training sequences were required to reach the predictive performance demonstrated in Section 4.4. Further evaluation in a future work is required to determine if PredNet, now trained to perform OCSBA-RL, can learn new class/behavior combinations more quickly.

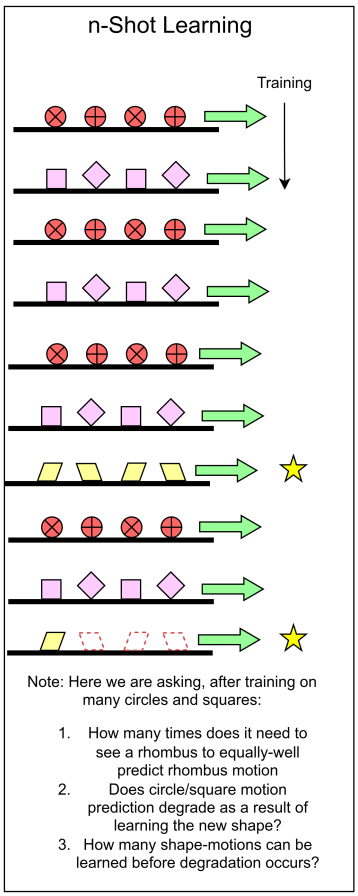


Figure 49: n-Shot Learning

6. **Adaption to change in environment** This human ability relates to learning environment-specific structure. For example, Checkers and Chess are two board games that use a square checkered board and pieces that move in specific ways. However, we recognize that the choice of game determines how the pieces move. The example is imperfect because the game pieces are not identical, but in a model test implementation, the structure can be made solely environment-specific. See Figure 50 for an illustration. In this figure, in the left portion labeled 'Env 1', circles are known to translate to the right based on a binary world state. On the other hand, in 'Env 2', we can establish a scenario where circles roll, instead of translate, and that they do so always, without a world state indicator. This is a bit of an obscure human ability, but per the mentioned examples we can see its usefulness.

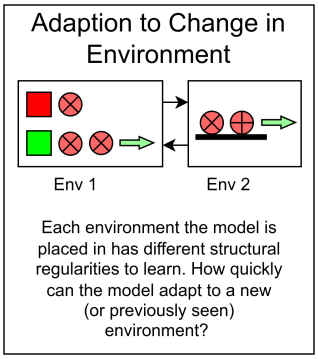


Figure 50: Adaption to Change in Environment

Second, PredNet forms a representation that can evaluated for use in downstream tasks, one of which is next-frame video prediction reliant on rudimentary structure learning comprising objects, classes, and associated class-specific behaviors. We have seen how PredNet performs in this task via the SSM dataset. Other downstream tasks could be those used to benchmark CNNs such as image classification, object detection, localization, action classification, semantic segmentation, etc., all of which, however, require training supervision (Simonyan & Zisserman, 2015). We have seen already in Lotter et al. (2017) how the PredNet representations formed can be useful to extract semantic information about the present scene, for example car steering wheel angle. Based on the experimental results presented in this paper regarding structure learning, it is noted that future work evaluating PredNet's representation against supervised data associated with the learned structure (such as the number of objects in the scene, or a one-hot encoded representation of the direction the object is moving etc.) would be a worthwhile endeavor. As noted in regards to the "Composite Learning PredNet" model improvement described in Section A.4, this supervised evaluation can be included during training for a potential representation-learning enhancement.

A.4 Other Model Improvements

Due to the inspiring but imperfect performance of the trained baseline PredNet models on the four created SSM dataset tasks, the following concepts may show promise to improve representational learning power and predictive performance. These improvements can each be implemented alongside the main next-steps for model improvement described in 5.4.

1. **Upgrade MSE loss function** Oprea et al. (2022) and Somraj et al. (2022) both review the loss functions used in video prediction models (see Section 2.4). They note that the chosen loss function can have an effect on the types of errors made by the prediction model. The errors noted for pixel-wise loss functions such as MSE (as is used in PredNet) include blurry predictions while those for adversarial loss functions include object-related issues such as distorted object-shapes, and the sudden appearance or disappearance of objects in predicted frames. Despite the issues noted with each loss function, some researchers seek to combine MSE and adversarial losses for a best-of-both-worlds result. This may also be promising for PredNet.

2. **Incorporate Multi-Frequency Pre-Processing** Taking inspiration from Jin et al. (2020) and the human visual system, target video frames can be decomposed into a spread of directional spatial-frequency channels. These channels can then be passed through a CNN and fed into the input to PredNet's Representation ConvLSTM unit, similar to as described for object-centric data in Section 5.4. By providing this additional context to PredNet, the model can be expected to maintain an accurate understanding of the overall scene per low-frequency components, while high-frequency components focus on local details and allow PredNet to preserve these details in its subsequent predictions.

3. **Multi-modal PredNet** - In addition to raw image data, environmental regularities can be made more obvious by means of data extension and enrichment. By including more diverse perspectives on the same data, the model is likely to have an easier time learning to predict according to these regularities. These perspectives can include image disparity for 3D scenes, optical flow, and image segmentation. These different modalities can be concatenated channel-wise for direct use with the baseline PredNet architecture, or several PredNet models can work in parallel, each on their own modality, and contributing top-down predictions to the other modality PredNet models. See Figure 51 for an illustration. Existing research certainly exist in this strain. For example, Slavic et al. (2021) look at combining ground-truth odometry data alongside video data for improved video prediction performance. It is important to note that while 3D disparity data can be deterministically calculated, optical-flow and image-segmentation can only be provided to the model as learned estimates of the underlying ground-truth. Therefore, any inconsistencies in the flow or segmentation estimates is encountered as noise by the model, and should be minimized.

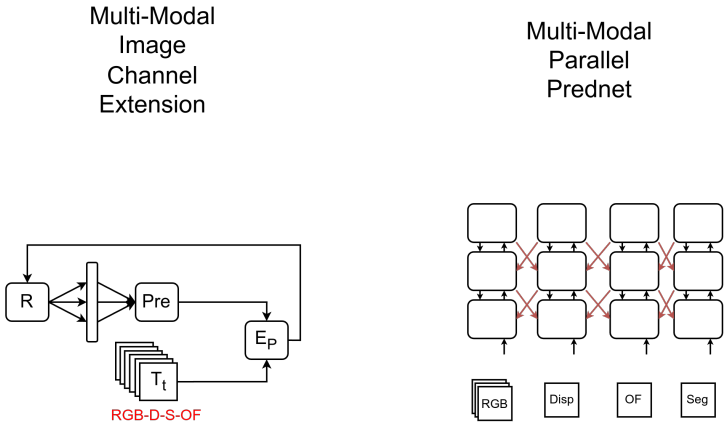


Figure 51: Multi-Modal PredNet Variants

4. **Composite-Learning PredNet** - Taking inspiration from the research direction of multi-task learning within neural network training, we present here how such a concept could be implemented with PredNet (Ruder, 2017). See Figure 52 for an illustration. In the figure, we show "Pre" - predictions being formed for targets at time t, while "Re" - reconstructions are formed for the previous time-step. Similar work with ConvLSTMs has proven beneficial (Srivastava et al., 2016). Additionally, derived metrics such as numbers of each class present in the scene can also tasked, however these metrics pose a supervised learning task instead of the self-supervision the model is currently employing. For a synthetic dataset, these metrics could be formed easily alongside the videos, but in a real-world continual-learning setting, such metrics would not be immediately available.

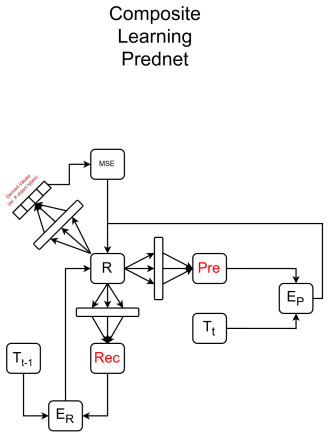


Figure 52: Composite Learning PredNet

5. **Memory-Augmented PredNet** - Though not yet explicitly tested, the task envisioned in Figure 48 is likely to be facilitated by augmenting PredNet with a long-term memory. Instead of storing discrete learned facts, this improvement envisions connecting an auto-associative memory module to one or more of the representation units in the layers of PredNet. See Figure 53 for an illustration for where the memory module has been connected to the top layer's representation unit. Taking inspiration from Annabi et al. (2022), we propose a possible VAE-based AA module implementation as follows. A VAE with ConvLSTM encoder and decoder is trained to complete incomplete input sequences (for the past *n* time steps with the current time-step masked to a placeholder value) of the top-layer's Representation unit's activated tensor. So, we are proposing to encode sequences of the top-layer's representation, and to retrieve a sequence that includes the expected next representation. This final representation in the sequence can then be included as input to the Representation unit's ConvLSTM for generating the next representation from which to

form the layer's prediction. Annabi et al. (2022) propose a memory-dependent prior distribution, where the memory is a set of stored encoded representations. However, as these authors note, over-parameterized VAE's already implement auto-associative memory. Furthermore, we feel the restriction to a set of stored, discrete memories may be memory and computationally limiting. For these reasons, the VAE with ConvLSTM encoder and decoder would be our initial choice.

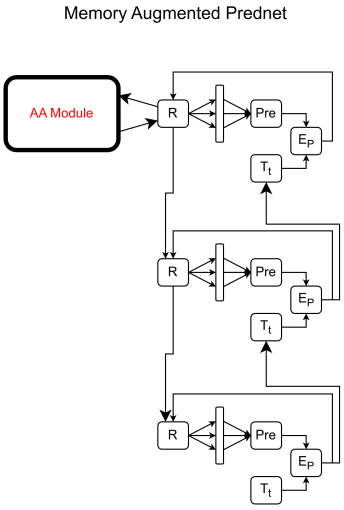


Figure 53: Memory-Augmented PredNet

6. **Pan-Hierarchical PredNet** Inspiration for this model improvement comes from the highly interconnected nature of the human visual system, where there is information transfer between all levels of the hierarchy (Van Essen & Maunsell, 1983). The Pan-Hierarchical PredNet, then, seeks to evaluate the representations formed at every level, and to provide back its own meta-representation to facilitate a form of communication between the layers in a simple manner. - See Figure 54 for an illustration.

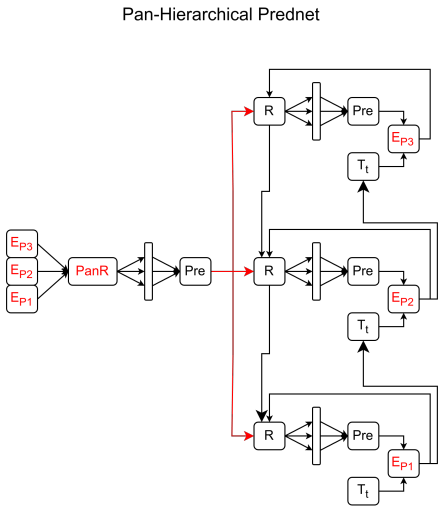


Figure 54: Pan-Hierarchical PredNet