



BIROn - Birkbeck Institutional Research Online

Blakeman, Sam and Mareschal, Denis (2022) Selective particle attention: rapidly and flexibly selecting features for deep reinforcement learning. *Neural Networks* 150 , pp. 408-421. ISSN 0893-6080.

Downloaded from: <https://eprints.bbk.ac.uk/id/eprint/47766/>

Usage Guidelines:

Please refer to usage guidelines at <https://eprints.bbk.ac.uk/policies.html>
contact lib-eprints@bbk.ac.uk.

or alternatively

Selective Particle Attention: Rapidly and Flexibly Selecting Features for Deep Reinforcement Learning

Sam Blakeman^{a,b,*}, Denis Mareschal^b

^a*Sony AI
Wiesenstrasse 5, 8952
Schlieren
Switzerland*

^b*Centre for Brain and Cognitive Development
Department of Psychological Sciences
Birkbeck, University of London
Malet Street, WC1E 7HX
United Kingdom*

Abstract

Deep Reinforcement Learning (RL) is often criticized for being data inefficient and inflexible to changes in task structure. Part of the reason for these issues is that Deep RL typically learns end-to-end using backpropagation, which results in task-specific representations. One approach for circumventing these problems is to apply Deep RL to existing representations that have been learned in a more task-agnostic fashion. However, this only partially solves the problem as the Deep RL algorithm learns a function of all pre-existing representations and is therefore still susceptible to data inefficiency and a lack of flexibility. Biological agents appear to solve this problem by forming internal representations over many tasks and only selecting a subset of these features for decision-making based on the task at hand; a process commonly referred to as selective attention. We take inspiration from selective attention in biological agents and propose a novel algorithm called Selective Particle Attention (SPA), which selects subsets of existing representations for Deep RL. Crucially, these subsets are not learned through backpropagation, which is slow and prone to overfitting, but instead via a particle filter that rapidly and flexibly identifies key subsets of features using only reward feedback. We evaluate SPA on two tasks that involve raw pixel input and dynamic changes to the task structure, and show that it greatly increases the efficiency and flexibility of downstream Deep RL algorithms.

Keywords: Selective Attention, Visual Features, Reinforcement Learning, Particle Filter, Neural Networks

*Corresponding author

Email addresses: samrobertallan.blakeman@sony.com (Sam Blakeman^a), d.mareschal@bbk.ac.uk (Denis Mareschal)

1. Introduction

Deep Reinforcement Learning (RL) has emerged as a core framework for training agents to solve complex decision-making tasks, such as playing video games from raw pixel values (Mnih et al., 2015, 2016). Despite these successes, Deep RL is often criticized for being data inefficient and inflexible to changes in task (Lake et al., 2017). This is largely due to the fact that representations are most commonly learned in an end-to-end manner via backpropagation. The learned function therefore depends on all the input values and the intermediate representations are highly tuned to a specific task, which reduces both efficiency and flexibility. One way to overcome these issues is to apply Deep RL to representations that have been learned from the raw input in a more task-agnostic manner, such as via unsupervised learning (Bengio, 2012) or additional training objectives (Jaderberg et al., 2016). This helps to improve efficiency because the dimensionality of the pre-learned representations is often lower than the original input. Flexibility is also improved because the pre-learned representations are not tuned to a specific task and can therefore support a range of different tasks. These improvements are limited however, because all pre-learned features are provided as input to the Deep RL algorithm. The scale of the dimensionality reduction is restricted by the number of pre-learned representations and changes to the task will require backpropagating errors across all of the pre-learned features, thereby over-writing any previously learned function.

We propose that these limitations of using pre-learned representations can be overcome by applying a selective attention mechanism that is inspired by biological agents. Selective attention in biological agents acts as a gating mechanism, whereby the firing of neurons that encode an attended feature increase, while the firing rates of neurons encoding a non-attended feature decrease (Treue and Trujillo, 1999; Saenz et al., 2002). This gating mechanism helps to improve the efficiency and flexibility of the learning algorithm in several ways. Firstly, it serves to further reduce the dimensionality of the learning problem (Jones and Canas, 2010; Niv et al., 2015; Wilson and Niv, 2012), which increases the efficiency of learning and reduces sensitivity to changes in the environment. Secondly, if the task or goal changes then a new subset of features can be chosen quickly and flexibly without over-writing the newly learned function thereby alleviating catastrophic forgetting and interference. Finally, selective attention can allow for learning over different time-scales. For example, the underlying representations can be learned slowly (Schyns et al., 1998) over many tasks while the subset of underlying features that are important for the current task can be identified and mapped to actions on a much shorter time-scale i.e. within a single task.

In recent years, Deep RL researchers have shown a lot of interest in attention mechanisms. Most notably, self-attention and the transformer architecture have been applied in Deep RL to push the state-of-the-art further (Zambaldi et al., 2018; Manchin et al., 2019; Shen et al., 2019). At its core, self-attention allows an agent to learn how to attend to different internal representations based on the task at hand by mapping query and key-pair vectors to an output vector (Vaswani et al., 2017). Crucially, this process relies on learning the query, key and value representations via backpropagation in an end-to-end manner. Unfortunately, this reliance on backpropagation for the learning of representations means that self-attention still suffers from the classic criticisms of Deep RL; poor data efficiency and a lack of flexibility, the very issues that attention in biological agents attempts to solve.

In order to harness the benefits of selective attention in biological agents we believe that an attention mechanism should be able to rapidly and flexibly select subsets of features without relying on learning new representations via backpropagation. However, such a mechanism faces several key challenges; (i) the number of feature combinations could be extremely large, (ii) reward feedback may be sparse, and (iii) changes in the environment may occur without warning. To address these challenges we present a novel algorithm termed *Selective Particle Attention (SPA)* (Figure 1A), which uses a particle to filter to attend to subsets of features produced by a pre-trained neural network. These features are then passed to a down-stream Deep RL algorithm that maps them to actions using a deep neural network. This leads to a substantial improvement in the data efficiency and flexibility of the Deep RL algorithm. Most importantly, this process does not rely on backpropagation and is therefore able to attend to key subsets of pre-existing features using only reward feedback in a rapid and flexible manner.

We evaluate *SPA* on two key tasks; a multiple choice task and a 2D video game based on collecting objects. Both tasks involve processing observations from raw pixel input and dealing with unannounced changes in task structure. In both cases the selective attention mechanism of *SPA* leads to improved performance in terms of the onset, speed, robustness and flexibility of learning compared to approaches that attend to

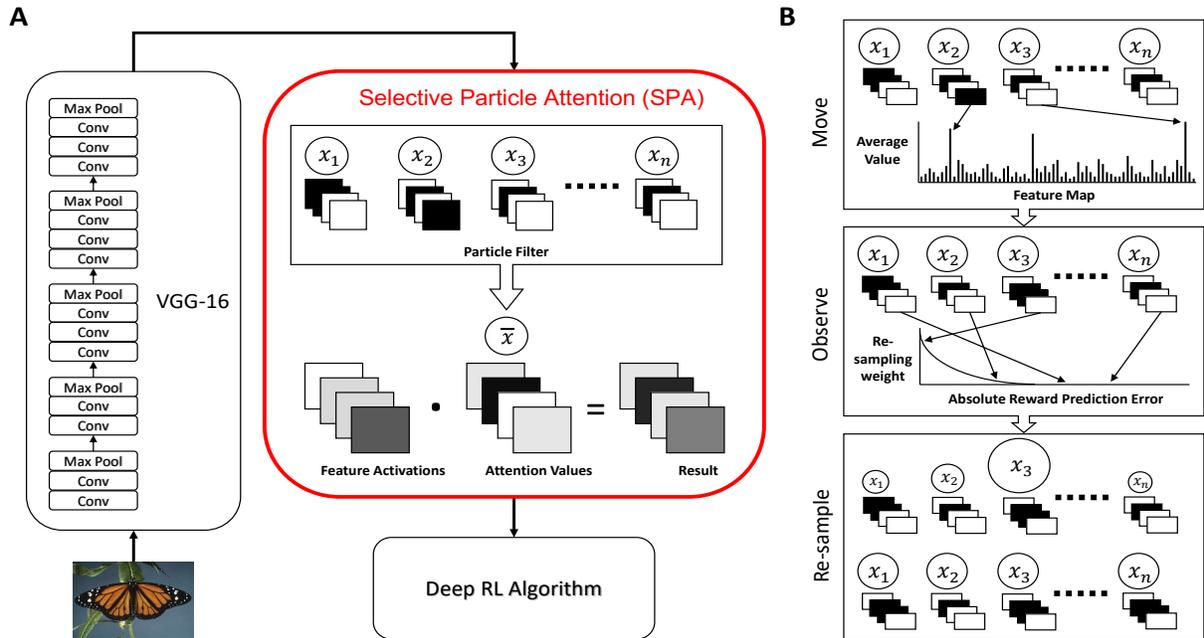


Figure 1: Graphical depiction of Selective Particle Attention (SPA). (A) We use a pre-trained deep convolutional neural network (VGG-16) to extract 2D feature maps from an image. The feature maps are then fed to SPA, which uses a particle filter to generate attention values. Each particle (x_i) in the particle filter uses a binary vector to represent a hypothesis about which feature maps are useful for the current task. The attention values are calculated as the mean over all particle states, normalised to sum to one (\bar{x}). Each of the feature maps is multiplied by its corresponding attention value in an element-wise fashion to produce the output of SPA, which is passed to a Deep Reinforcement Learning (RL) algorithm that maps the features to actions using a deep neural network. (B) Updating of the particle filter. In the first (movement) step, a small proportion of particle states (x_i) are updated to reflect the most active features given the current input. In the second (observe) step, particles are assigned re-sampling weights based on how accurately they predict reward. In the final (re-sample) step, the re-sampling weights are normalised to produce a probability distribution and the particles are re-sampled with replacement.

all features, a random subset of features, or employ self-attention. We also show that these findings occur independently of the RL algorithm used, making it applicable to a variety of problems. The remainder of this paper is as divided as follows. In Section 3.2 we provide a detailed description of *SPA*. In Section 4.1 we assess *SPA* on a multiple choice task that has a changing reward structure and involves making single decisions. In Section 4.2 we assess *SPA* on a 2D video game that can change either in reward structure or state space and that requires the agent to make sequential decisions. Finally, in Section 5 we discuss the implications of our findings and highlight promising avenues for future research.

2. Related Work

2.1. Latent Representations for Deep Reinforcement Learning

In end-to-end Reinforcement Learning (RL) representations of complex high dimensional inputs, such as pixel values, are learned using a sparse reward signal. This leads to low sample efficiency and representations that are highly specialized for a single task. One way to circumvent these problems is to learn low dimensional representations using other objective functions that have a denser training signal. These representations can then be passed to a Deep RL algorithm, which reduces the complexity of the problem faced by the Deep RL algorithm and leaves the underlying representations unperturbed for future tasks.

In line with this theory, several previous works have used autoencoders to learn low dimensional representations of high dimensional inputs. These low dimensional representations are then used to form the state space of an RL algorithm (Lange and Riedmiller, 2010; Finn et al., 2015; Nair et al., 2018; Higgins et al., 2017; Yarats et al., 2019). Crucially, the authors find that this approach leads to an improvement in the efficiency and robustness of the learned policy compared to a policy learned end-to-end. Similarly, other types of learning such as self-supervised learning can be used to extract features for RL. For example, contrastive learning has been used to learn latent representations for Deep RL algorithms to act upon (Dwibedi et al., 2018; Srinivas et al., 2020). This again substantially improves the efficiency of the Deep RL algorithm.

While applying Deep RL algorithms to latent representations has seen many successes in recent years, we believe this process can be improved further by incorporating a *fast* and *flexible* method for attending to subsets of latent features. By only attending to a subset of features, those not important for the current task can be ignored by the Deep RL algorithm. This further improves efficiency because the dimensionality of the problem is reduced and also further improves flexibility because if the task changes then a new subset of features can be attended to without entirely overwriting the previously learned function.

2.2. Attention in Deep Reinforcement Learning

In Deep Reinforcement Learning (RL) attention to subsets of features is typically achieved by multiplying each feature by a value between 0 and 1. A value closer to 1 means that more 'attention' is paid to that feature. While this simple filtering process is effective, the difficulty lies in inferring the attention values for a given task. This is particularly difficult when the only feedback is in the form of a sparse, delayed reward signal. The most popular approach for learning a vector of attention values is known as self-attention (Vaswani et al., 2017). Self-attention works by learning three separate representations from the provided input; queries, keys and values. The attention vector is calculated as the dot-product between the queries and keys followed by a softmax function. The output is then the weighted sum of the values, where the weights are provided by the attention vector. This mechanism has been widely used in Deep RL (Zambaldi et al., 2018; Mott et al., 2019; Shen et al., 2019; Manchin et al., 2019; Iqbal and Sha, 2019; Parisotto et al., 2020; Wang et al., 2020; Bramlage and Cortese, 2021) and has been involved in achieving start-of-the-art results (Vinyals et al., 2019).

Despite these successes we believe that self-attention is not an ideal mechanism for selecting subsets of features to be used by a Deep RL algorithm. The primary benefit of self-attention is its ability to relate different parts of the input regardless of how far apart they are. This is particularly useful when trying to solve problems where the input is a sequence (Vaswani et al., 2017) or when abstract relationships within the input need to be represented (Zambaldi et al., 2018). Self-attention's primary goal is not to identify subsets of features in a rapid and dynamic manner, which is evident from the fact that backpropagation is used to learn the query, key and value representations. This reliance on backpropagation means that the underlying attention mechanism is slow to train and inflexible; these are the exact issues that attending to subsets of features should help to alleviate.

105 Attempts to learn an attention vector without backpropagation have been limited. Most notably, Yuezhang et al. (2018) explored hand-crafting attention values or inferring them with optical flow. Hand-crafting attention values is effective but circumvents the problem of learning the values, which is the focus of this work. In comparison, optical flow can only be applied to problems involving visual movement and is fixed regardless of reward structure. This means that it captures bottom-up attention, where attention is influenced by the
110 inherent properties of the input, but not top-down attention, where attention is guided by current goals and reward structure.

2.3. Particle Filters

Filtering-based approaches have been widely used to deal with uncertainty and non-linear dynamics in Markov processes (Zhang et al., 2021b,a; Rudenko, 2019). In the present work we propose that particle filters
115 in particular can be used to rapidly and dynamically learn which features to attend to by combining both bottom-up and top-down attention. Particle filters rely on approximate inference and importance sampling to infer the values of a hidden variable given sequential observations (Doucet et al., 2009). As a result, they have been used in Deep Reinforcement Learning (RL) to deal with the problem of partially observable Markov decision processes (POMDPs), where the true Markovian state of the environment is unknown and must be
120 inferred from past observations (Besse, 2009; Igl et al., 2018; Ma et al., 2020).

In the present work we suggest that particle filters have several properties that also make them a good candidate for rapidly and dynamically attending to subsets of pre-learned features. Firstly, they use approximate inference, which is important because the number of feature combinations to consider can be prohibitively large. Secondly, particle filters are able to deal with non-linear dynamics and non-gaussian
125 distributions, which are typical of RL problems. Finally, and most importantly, particle filters rely on sequential importance sampling rather than backpropagation. This means that they can be updated rapidly and dynamically in comparison to approaches such as self-attention.

3. Model Implementation

3.1. Bio-inspired Features of Selective Particle Attention (SPA)

Our approach is heavily inspired by selective attention in biological agents. Selective attention is the process of selecting subsets of pre-existing features for decision-making and is thought to be the responsibility of the Pre-Frontal Cortex (PFC) (Desimone and Duncan, 1995; Miller and Cohen, 2001; Bichot et al., 2015; Paneri and Gregoriou, 2017). For example, Selective Particle Attention (SPA) applies attention by multiplying each feature by a value between 0 and 1, which mimics the heightening and dampening of
130 responses often associated with biological attention (Treue and Trujillo, 1999; Saenz et al., 2002). These values are learned via a particle filter, which has been proposed to be a plausible computational model of biological attention (Radulescu et al., 2019a) that can model shifts in attention better than gradual error-driven learning (Radulescu et al., 2019b). Each particle of the filter is used to represent a hypothesis about which features are important for the current task. This reflects the way the PFC represents competing hypotheses about
140 which features of the visual input to attend to given the current goal (Miller and Cohen, 2001; Radulescu et al., 2019a). It also consistent with Bayesian theories of the brain that suggest flexible behaviour is based on approximate inference and hypothesis testing (Koechlin, 2014; Jiang et al., 2014; Donoso et al., 2014).

The states of each particle are updated based on (1) how accurately the combination of features predicts future reward and (2) the magnitude of activation of individual features. The first step reflects top-down
145 attention and is based on a theoretical proposal from neuroscience that humans learn to attend to the features that are most predictive of reward (Mackintosh, 1975). For example, people are able to switch between an object-based or a feature-based state representation based on which one is the best predictor of reward (Farashahi et al., 2017). The second step reflects bottom-up attention and is inspired by the fact that individual features with inherent saliency play an important role in the biasing of biological attention
150 (Desimone and Duncan, 1995; Treue, 2003). Once the particle states have been updated via these two steps, the attention values are calculated by averaging the particle states and normalising the result to sum to one. This is in line with computational neuroscience models of visual attention, which highlight normalisation as a key step for capturing competition between features in the visual cortex (Reynolds and Heeger, 2009).

3.2. Algorithmic Details

3.2.1. Attention Vector

In order to filter the pre-learned representations for the Deep RL algorithm, Selective Particle Attention (SPA) uses an *attention vector* to perform soft-attention. The attention vector has K dimensions, where each entry can take on any real valued number between 0 and 1 inclusive:

$$\mathbf{A} \in [0, 1]^K \quad (1)$$

Where K is the number of pre-learned features, which in our case was 512. This attention vector is applied to the pre-learned features using element-wise multiplication between \mathbf{A} and the values of each feature. In the case of VGG-16, each entry in \mathbf{A} is replicated to match the dimensions of a single feature map, with the same attentional value applied across all spatial locations. This process re-weights the feature map activations, amplifying feature maps with a large attentional weight while maintaining all spatial information. The output is then reshaped into a single vector and passed on to a Deep Reinforcement Learning (RL) algorithm that maps it to a chosen action.

3.2.2. Particle Filter

Rapidly and flexibly learning the values of the attention vector in an RL setting is difficult because feedback is a sparse scalar reward and actions have temporal consequences. To solve this problem, we use a particle filter to learn the values of the attention vector given the current task. Particle filters are typically used to estimate the value of a latent variable (X) given noisy samples of an observed variable (O) when the number of potential values is large. The overall goal of a particle filter is to use a set of ‘particles’ to represent a posterior distribution over the latent variable. Each particle represents a belief or hypothesis about the value of the latent variable and the density of the particles can be used to approximate the posterior distribution.

In our case the latent variable X represents the configuration of features that are useful for the current task. We use a value of 1 to denote a feature as useful and a value of 0 to denote a feature as not useful. This means that x can be any binary vector of size K :

$$x \in \{0, 1\}^K \quad (2)$$

Where K is the number of features, which in our case was 512 corresponding to the number of feature maps of VGG-16. As the number of possible values that X can take is 2^K , we use N particles to approximate the posterior distribution over X where $N \ll 2^K$. The state x^i of the i^{th} particle therefore corresponds to a binary vector of length K and provides a hypothesis about which features it deems relevant for the current task.

A particle filter consists of two main steps; a movement step and an observation step. In the movement step, the particles are updated based on some known transition probability for the latent variable:

$$x' \sim P(X'|x) \quad (3)$$

This process is often used to represent the passing of time. In our approach we introduce the notion of bottom-up attention during the movement step. Let \bar{f}_t^k denote the average value over all the units in feature map k from VGG-16 at time t . At each time-step t a particle is updated as follows with some probability ϕ :

$$v_t^k = \frac{\bar{f}_t^k}{\sum_{j=1}^K \bar{f}_t^j} \quad (4)$$

$$p^k = \frac{\exp(v_t^k * \tau_{BU})}{\max_j \exp(v_t^j * \tau_{BU})} \quad (5)$$

$$P(x'_k = n) = \begin{cases} p^k & \text{for } n = 1 \\ 1 - p^k & \text{for } n = 0 \end{cases} \quad (6)$$

First the mean activation values for each feature map of VGG-16 are normalised to sum to 1. This normalisation accounts for differences in overall activation values between time steps and preserves relative

180 differences between activation values. The activation values are then exponentiated and normalised by the maximum value across all feature maps. This ensures that the most active feature will receive a value of 1. Finally this is used as the probability that the k^{th} entry of the particle state will be equal to one, as described by a Bernoulli distribution. In this way, a proportion of the particles are updated to represent the most active features given the current input. This is akin to bottom-up attention, whereby highly salient perceptual features capture one’s attention in an involuntary manner. In our agent this serves to introduce a prior to attend to the highly active features of the current task. The free parameter τ_{BU} controls the strength of the bottom up attention, a higher value of τ_{BU} leads to a higher probability of the most active features being attended to and the least active features not being attended to.

185 In the observation step of a particle filter, particles are weighted based on the likelihood of the observed variable O given the value of the latent variable X represented by a particle. These weights are then used to re-sample the particles and update the posterior distribution ready for the next time step. In our approach, we introduce top-down attention during this step. We take our observed variable O to be the return from a given state R_t . We calculate the likelihood of this return by using the particle state as the attention vector and calculating the error between the predicted state value and the return. Let x^i denote the state of the i^{th} particle, the likelihood for x^i is calculated as follows:

$$A_k^i = \frac{x_k^i}{\sum_{j=1}^K x_j^i} \quad (7)$$

$$\delta^i = (R_t - V(s_t; \mathbf{A}^i))^2 \quad (8)$$

$$P(R_t|x^i) \propto \exp(-(\delta^i - \min_j \delta^j) * \tau_{TD}) \quad (9)$$

190 Where R_t is the return from state s_t and $V(s_t; \mathbf{A}^i)$ is the predicted state value calculated by using the normalised particle state x^i as the attention vector. The normalisation step in Equation 7 accounts for the different numbers of features that are attended to by different particles. Equation 8 calculates the squared error between the return and predicted state value. The likelihood of the return R_t is then proportional to this error value. This process can be seen as evaluating the accuracy of a particle’s hypothesis over which features are relevant for the given task. If the particle’s hypothesis is good then it will more accurately predict the target return and so will produce a larger likelihood. In this way we capture the effect of top-down attention, whereby a set of hypotheses are evaluated and the most accurate ones are considered for the next time step. τ_{TD} controls the strength of this top down attention; a larger value will more strongly penalise hypotheses that are inaccurate.

195 Once the likelihoods have been calculated they are normalised to form a probability distribution and the particles are re-sampled with replacement:

$$P(x') = \frac{\sum_{i=1}^N P(R_t|x^i)\mathbb{I}(x^i = x')}{\sum_{i=1}^N P(R_t|x^i)} \quad (10)$$

$$x' \sim P(x') \quad (11)$$

Once the re-sampling has been performed, the final step is to reset the value of the attention vector. This is done by setting the attention vector to be the mean of the particle states and then normalising the vector to sum to one:

$$\bar{x}_k = \frac{1}{N} \sum_{n=1}^N x_k^n \quad (12)$$

$$A_k = \frac{\bar{x}_k}{\sum_{j=1}^K \bar{x}_j} \quad (13)$$

Where N is the number of particles and K is the number of features. The full algorithm used to update the attention vector can be seen in Algorithm 1 in the appendix.

200 3.3. Modular Design Choices

In order to assess how well Selective Particle Attention (*SPA*) can rapidly and flexibly attend to subsets of pre-learned features, we have to decide on a method for extracting the features and on a Deep RL algorithm

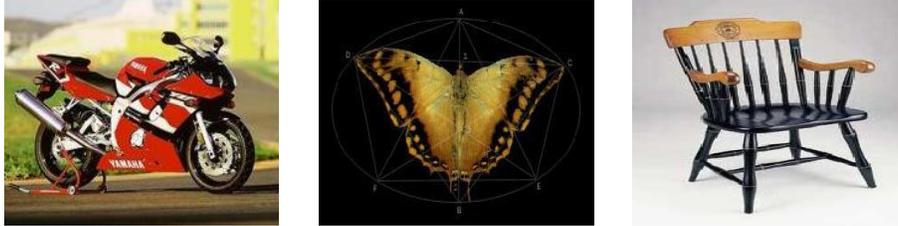


Figure 2: *Example images from the Caltech 101 data set, which consists of 101 object categories with approximately 40 to 800 images per category. Left image is from the 'motorbikes' category, the middle image is from the 'butterfly' category and the right image is from the 'chair' category. On each trial the agent was presented with three separate images each selected randomly from three different categories. For a given block, only one image category was associated with a reward of +1 and the rest were associated with a reward of 0. The rewarded category was chosen at random for each block of trials.*

for learning a control policy from these features. A vast array of approaches exist for feature extraction that are compatible with *SPA*. We chose to use the pre-trained Deep Convolutional Neural Network (DCNN) VGG-16 (Simonyan and Zisserman, 2014) because it can be applied to raw pixel values, it is widespread in the machine learning literature and it has a good correspondence with representations found in the visual stream of biological agents (Schrimpf et al., 2018). Similarly, there are many Deep RL algorithms that can be applied to pre-learned features. Often this decision is influenced by the nature of the problem you are trying to solve, for example whether it involves a continuous or discrete action space. In our case the only constraint is that the Deep RL algorithm must estimate a value function. This is critical because it allows *SPA* to decide whether features should be attended to or not based on how well they predict future reward (see Section 3.2). For the first experiment (multiple choice), we used a Deep Neural Network to estimate the value of each image and chose the image with the highest value in an ϵ -greedy manner. The network was trained to minimise the difference between the predicted value of the chosen image and the actual reward obtained, which is equivalent to performing a one-step Monte Carlo backup (Sutton and Barto, 1998). For the second experiment (object collector), we chose to use Advantage Actor-Critic (A2C) (Mnih et al., 2016), which has been widely used for solving video games.

More details of these algorithms can be found in the Appendix, including all of the hyper-parameter values used (Table 1). The majority of these values are standard for Deep RL based on the types of problems that we are exploring. However, the number of particles, the frequency of attention updates, and the strength of the bottom-up and top-down attention are unique to our approach (*SPA*). We therefore conducted an informal parameter search over these parameters and found that the results were largely consistent. We chose the final values based on their order of magnitude.

4. Experiments

4.1. Making Single-Step Decisions in the Face of a Changing Environment

In order to test the effectiveness of *SPA* we designed two tasks that require an agent to select actions based on raw pixel inputs and to switch attention without warning in response to changing task demands. In both tasks the score used to evaluate the agent was the achieved reward because this was the value the algorithm was tasked with optimizing. The first task consisted of multiple choice questions using images from the entire Caltech 101 data set (Fei-Fei et al., 2006), which consists of 101 object categories with approximately 40 to 800 images per category. At the start of the task, three object categories were randomly chosen from the Caltech 101 data set and one of them was randomly designated as the 'target' category. Each question then consisted of presenting one image randomly sampled from each of the three categories (Figure 2). The image from the 'target' category was associated with a positive reward of +1 while the images from the other categories were associated with no reward. Crucially, after every 50 questions, the 'target' category was changed at random.

The purpose for this task design was two-fold. Firstly, the agent has to work out on each trial which image is associated with a reward based on the features of the 'target' category. For example, if the target category is 'soccer ball' then the agent has to learn to attend to features that are indicative of round shapes in order to obtain reward. Secondly, because the 'target' category changes every 50 trials, the agent also has

to adapt to this change in reward structure by dynamically shifting its attention to the features of another category using only reward feedback.

To ensure that the agent did not learn to remember specific exemplars in the data set we had training and test phases. During the training phase the agent was allowed to update the parameters of the Deep RL network in response to reward feedback but during the test phase it was not. Therefore the only parameters the agent was allowed to change during the test phase was the attention values using the particle filter. Importantly, the test phase used images from the chosen object categories that were not presented during training. In total, the training phase consisted of 200 blocks of 50 trials and the test phase consisted of 10 blocks of 50 trials.

As a baseline to measure the effectiveness of our proposed attention mechanism we compared the performance of *SPA* to a version of *SPA* where each entry of the attention vector was set to a fixed value of $1/K$. This corresponds to attending to all features of VGG-16 equally and we refer to this approach as *SPA_{ALL}*. We also ran an ideal observer model on the multiple choice task to provide a measure of ceiling performance. The ideal observer model selects the image corresponding to the last rewarded object category. All models select a random action with probability ϵ in order to encourage exploration.

Figure 3A shows the performance of *SPA*, *SPA_{ALL}* and the ideal observer during training on the multiple choice task for one random combination of object categories over 5 random seeds. In this example *SPA* performs close to optimal as it shows a similar learning trajectory to the ideal observer. In comparison, *SPA_{ALL}* performs poorly and is substantially worse than both *SPA* and the ideal observer. To test the robustness of these findings we ran all three approaches over 5 random seeds on 20 different combinations of object categories. Figure 3C shows the results of these simulations. *SPA* out-performed *SPA_{ALL}* during training for every combination of categories that we tested.

While the dynamic attention mechanism of *SPA* appeared to provide a substantial benefit during training, we also wanted to test whether this benefit generalized to unseen images. Figure 3B shows the results of the three approaches on the test phase after the training seen in Figure 3A. Again the rewarded image category was changed every 50 trials. Importantly the test blocks used images that were not used during training and all the weights of the Deep RL algorithm were frozen so that only the attention vector could change in the case of *SPA*. Again *SPA* exhibited performance similar to that of the ideal observer, while *SPA_{ALL}* showed significantly worse performance. Figure 3D shows the test results over 5 random seeds for all 20 of the different category combinations. As with training, *SPA* out-performed *SPA_{ALL}* for all of the category combinations. These results suggest that the benefit of the attention mechanism of *SPA* generalizes well to unseen images.

Both the training and test results suggest that *SPA* is able to cope with changes in the reward function by dynamically re-configuring existing representations for the purpose of state evaluation. Figure 4 shows an example of the attention vector during the test phase. The attention vector reliably changes when the target image category changes. This confirms that the attention mechanism of *SPA* is able to use changes in the reward function to re-evaluate the features that need to be attended to. Interestingly, the attention vector is not the same every time a given image category is made the target. This is likely due to the fact that *SPA* will be biased towards attending to features that are present in the first few images, which will be different for each block. In addition, there is likely to be a contextual effect of the rewarded image category in the previous block. For example, if in the previous block the category 'soccer ball' was rewarded and in the current block 'faces' are rewarded, then this might bias the selection of features that correspond to 'round' in the current block.

The results shown in Figures 3B and 3D indicate that the performance of *SPA* can vary depending on the combination of image categories that are chosen. This suggests that *SPA* finds it easier to discriminate between certain image categories compared to others based on their features. Figures 5A and 5B show the mean feature map values of VGG-16 for the image categories that *SPA* performed best (leopards, faces and soccer balls) and worst (cups, chandeliers and cellphones) on. In the best case scenario, the feature maps contain several features that are substantially more active than others. This likely provides a good substrate for bottom-up attention because there are a handful of features that are reliably more active than the others, which corresponds to a strong prior over hypotheses. In comparison, in the worse case scenario, the features take on a more uniform distribution of activation values and so the prior over hypotheses is weaker.

Bottom-up attention aside, having a few highly active features is only useful for the multiple choice task if they help to discriminate between the different image categories. As seen in Figures 5A and 5B, each image

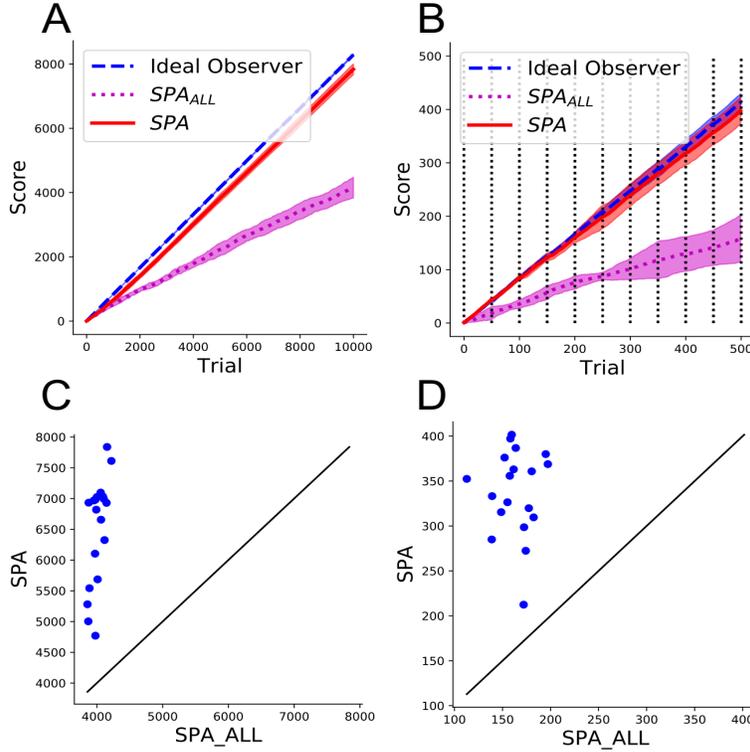


Figure 3: Performance of Selective Particle Attention (SPA) on the multiple choice task. (A-B) Training (A) and test (B) reward over 5 random seeds for the leopards, faces and soccer balls image categories. The rewarded image category was changed randomly every 50 trials and is represented by a vertical black line in B. Error bars represent one standard deviation. SPA used selective attention to attend to features that it deemed useful for the current task. SPA_{ALL} did not use selective attention and instead attended to every feature of VGG-16. The Ideal observer model represents ceiling performance. (C-D) Comparison of SPA and SPA_{ALL} in terms of total training (C) and test (D) reward for 20 different image category combinations. Each point represents the mean performance over 5 random seeds for an image category combination. Solid line represents equal performance between SPA and SPA_{ALL}

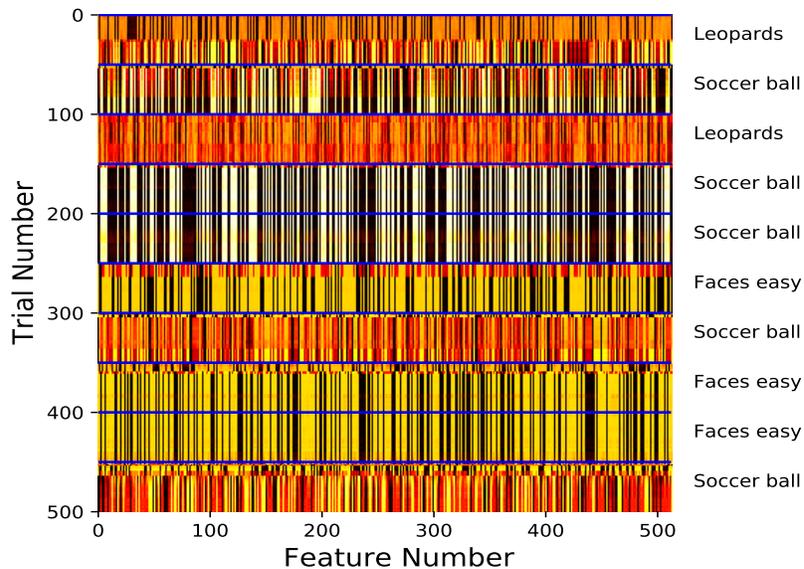


Figure 4: Example of the attention vector values for the leopards, faces and soccer balls image categories. Black represents a value of 0 and white represents a value of 1. The solid blue horizontal line represents a random change in the rewarded image category. The rewarded image category for a given block is presented on the right.

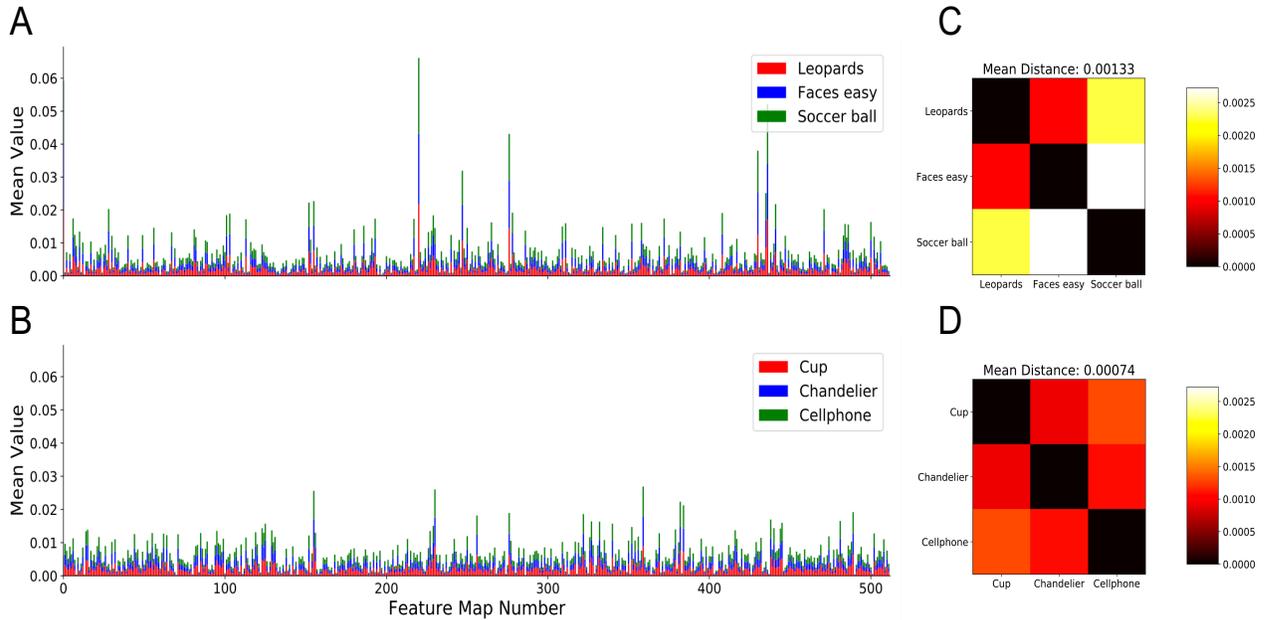


Figure 5: Analysis of the feature map values produced by VGG-16 during the multiple choice task. (A-B) Average feature map values of VGG-16 for the combination of image categories that lead to the best (A) and worst (B) training performance of SPA. (C-D) Euclidean distances between the average feature map values of VGG-16 for the combination of image categories that lead to the best (C) and worst (D) training performance of SPA.

category produces 512 average feature values, which can then be expressed as a vector in Euclidean space. Figures 5C and 5D show the Euclidean distance between these vectors for the best and worst case scenarios respectively. In the best case scenario, the Euclidean distance between the majority of the image categories is larger than in the worst case scenario. In addition the mean Euclidean distance over all pairwise comparisons in the best case scenario is nearly double that of the worst case scenario. This suggests that the categories in the best case scenario are easier to discriminate between because they are further apart in Euclidean space. This is likely to help the top-down attention of SPA because each particle will produce very different value estimates depending on the image category being attended to.

4.2. Making Multi-Step Decisions in the Face of a Changing Environment

Having demonstrated the effectiveness of SPA in the multiple choice task, we next wanted to investigate whether this translated to more complex tasks involving sequential decision-making. To this end we designed a 2D video game, which we refer to as the object collection game, using Pygame (www.pygame.org). In the game, the agent controlled a grey block at the bottom of the screen and could move it either left or right at each time step. An object, which could vary in shape and colour, was generated every second at the top of the screen and moved downwards to the bottom of the screen in a straight line. The goal of the agent was to collect objects by colliding with them as they reached the bottom of the screen. Importantly, the agent was only rewarded for collecting objects that had a specific colour and/ or shape, which encourages the agent to attend to specific features that are indicative of reward. Agents were trained for 2000 episodes, with each episode lasting 60 seconds.

We explored two different variations of this basic game configuration. In the first variation, the agent was given a reward of +1 for collecting objects of a certain shape and a reward of 0 for all other shapes (Figure 6A). This tested the agent’s ability to focus on particular features of the environment and ignore others; the agent had to attend to the target shape while ignoring colour and all other shapes. As we saw in the multiple choice task, one benefit of an effective selective attention mechanism is that attention can be altered in response to changes in the reward structure. To test whether this was also the case in the object collection game, we randomly selected a new shape to be rewarded after 1000 episodes. This is akin to changing the

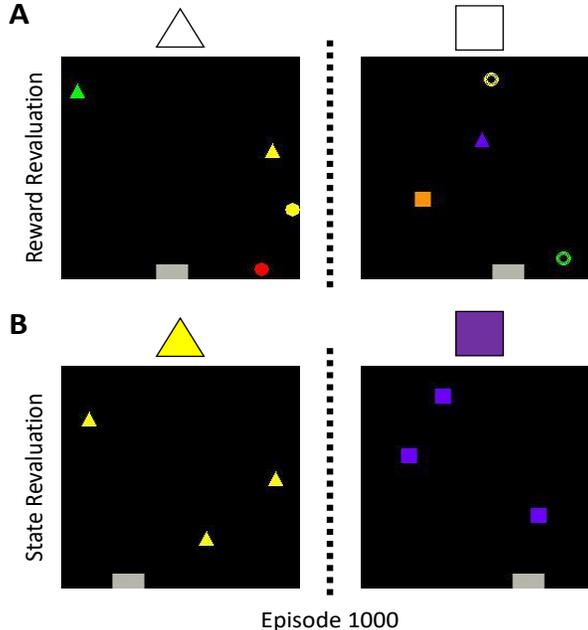


Figure 6: Example screenshots from the two variations of the object collection game. The agent is in control of the grey rectangle and can move it either left or right to catch objects that move from the top of the screen to the bottom. The rewarded object is depicted above each screenshot. (A) The reward revaluation task. A specific shape is associated with a reward of +1 regardless of its colour. All other shapes are associated with a reward of 0. The rewarded shape is then randomly changed after 1000 episodes, which corresponds to a change in the reward function of the environment. (B) The state revaluation task. All objects are the same shape and colour, and are associated with a reward of +1. The shape and colour of these objects is then randomly changed after 1000 episodes, which corresponds to a change in the state space of the environment.

reward function of the environment and requires the agent to re-evaluate its policy. We refer to this variation as the reward revaluation task.

In addition to the reward function, other aspects of the environment can change such as the set of perceptual observations experienced by the agent. In RL this is commonly referred to as a change in state space. The second variation of the object collection game therefore explored the ability of *SPA* to deal with changes in state space (Figure 6B). For the first 1000 episodes, all the objects were the same shape and colour and were associated with a reward of +1. After 1000 episodes the shape and colour of the objects were changed to a different shape and colour. This corresponds to a change in the perceptual input of the agent because the objects present in the first half of training are different to the ones present in the second half of training. We refer to this variation as the state revaluation task.

As with the multiple choice task, we compared *SPA* to a baseline approach that attended to all features of VGG-16, which we refer to as *SPA_{ALL}*. In addition to *SPA_{ALL}*, we also included two other conditions for comparison. The first condition set the attention vector to a random binary vector, which was then normalised to sum to 1. This condition was included to account for the fact that random feature reduction may lead to improved performance and we refer to it as *SPA_{RANDOM}*. The second conditioned used self-attention rather than *SPA* to attend to the features of VGG-16. We chose to compare *SPA* to self-attention because it is a standard state-of-the-art attention mechanism in Deep RL and relies upon backpropagation rather than approximate inference. Self-attention was implemented as follows; the features from VGG-16 were used to learn a query vector and K key vectors, where K is the number of feature maps of VGG-16. The standard self attention update (Vaswani et al., 2017) was then applied to obtain an attention vector with K dimensions that summed to one, which mirrors the attention vector of *SPA*. The attention vector was then applied to the features of VGG-16 in the same way as *SPA*.

Figure 7A shows the results of *SPA*, *SPA_{ALL}*, *SPA_{RANDOM}* and self-attention on the reward revaluation task. Focusing on the first 1000 episodes before the change in reward function, *SPA* significantly out-performed *SPA_{ALL}*, *SPA_{RANDOM}* and self-attention over the course of learning. *SPA_{ALL}* saw the worst performance showing little evidence of learning over the first 1000 episodes. This suggests that naively learning over all

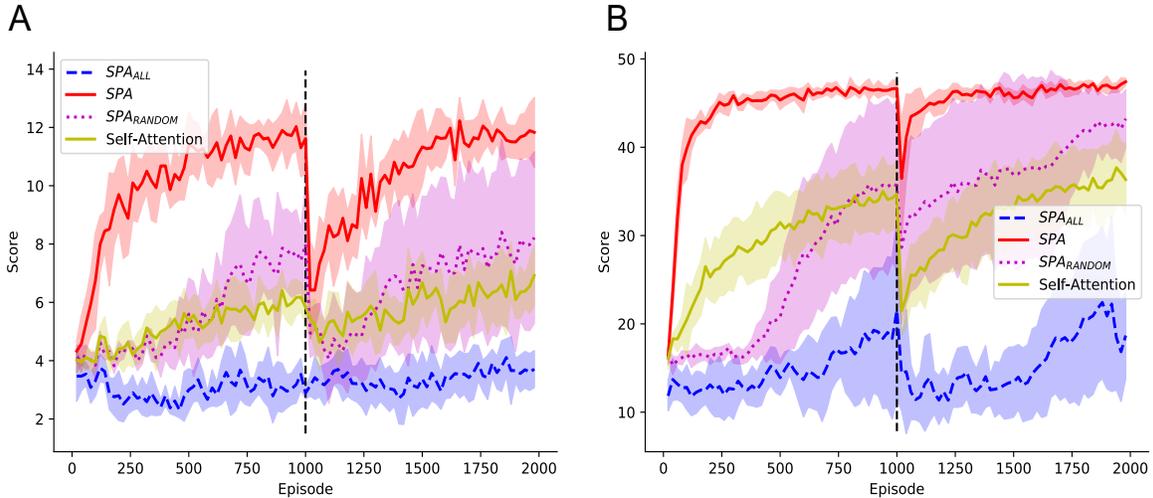


Figure 7: Performance of SPA , SPA_{ALL} , SPA_{RANDOM} and self-attention on the object collection game. Each point is the average reward over the last 20 episodes. Error bars represent one standard deviation. Each approach was run over 5 different random seeds and the error bars represent one standard deviation. SPA used selective attention to attend to features that it deemed useful for the current task. SPA_{ALL} did not use selective attention and instead attended to every feature of VGG-16. SPA_{RANDOM} attended to a random subset of features, which changed with each random seed (A) The vertical dashed line indicates where the rewarded object was changed without warning. (B) All objects were the same colour and shape. The vertical dashed line indicates where the shape and colour of these objects was changed without warning.

features is a highly ineffective strategy given a limited amount of experience. In contrast, SPA_{RANDOM} and self-attention showed evidence of gradual learning after around 500 episodes. The learning of SPA_{RANDOM} was highly variable as would be expected given that the features were randomly selected for on each random seed. Nevertheless this demonstrates that simply reducing the number of features at random is sufficient to provide a learning benefit. SPA began improving its reward almost immediately after a handful of episodes. This onset of learning is noticeably earlier than the other approaches. Not only did learning occur earlier but it was also much faster, as indicated by the sharper increase in reward compared to the other approaches. As a result, by episode 1000 SPA was able to achieve a higher reward than the other approaches. Importantly, SPA dramatically out-performed self-attention highlighting the benefit of using approximate inference for attention as opposed to backpropagation. Overall, these results suggest that the particle filter mechanism of SPA is able to identify features useful for value computation and that this translates into a substantial learning benefit given the current task.

Upon the change in reward function after 1000 episodes, SPA_{ALL} continued to demonstrate no evidence of learning. In comparison, SPA_{RANDOM} and self-attention showed a decrease in performance as soon as the reward function changed and they were slow to recover performance. SPA also showed a marked decrease in performance immediately after the change in reward function. This is to be expected as the agent had no prior warning that the reward function was about to change. However, the performance of SPA dropped to a level that was still above SPA_{RANDOM} and self-attention. The performance then remained significantly higher than SPA_{RANDOM} and self-attention throughout the recovery in performance. This resulted in SPA achieving a higher reward than all other approaches on the final episode. These results suggest that the particle filter mechanism of SPA is better equipped to handle changes in the environment by quickly re-evaluating which features are important for the current task.

Figure 7B shows the results of the four approaches on the change in state space task. As before, for the first 1000 episodes SPA demonstrated evidence of learning that occurred earlier, faster and more robustly than the other approaches. Upon the change in state space after 1000 episodes, SPA_{ALL} dropped back to starting levels of performance and was slow to recover to a low level of performance. In comparison, SPA showed a slight decrease in performance but was quickly able to recover to a high level of performance. Both SPA_{RANDOM} and self-attention also saw a drop in performance but the rate of recovery was slower

375 than *SPA* and only reached an intermediate level of performance. Interestingly, self-attention saw a larger
drop in performance than *SPARANDOM*. This highlights one of the key issues of using self-attention and
backpropagation for an attention mechanism; the query and key representations are learned specifically for
the first task and have to be slowly re-learned when the task changes. In comparison, the fixed attention of
380 *SPARANDOM* may already include some features that are useful for the second task and so it is less affected
by the change in task. These results further support the idea that the selective attention mechanism of *SPA*
is able to use approximate inference to quickly adapt to changes in the current task by re-orientating its
attention towards a different set of features. In addition, the consistently high performance of *SPA* and its
fast rate of recovery during the change in state space indicates that much of the knowledge learned by the
Deep RL algorithm in the first 1000 episodes was still of use in the second 100 episodes. This suggests that
385 the selective attention mechanism of *SPA* is also able to promote the transfer of knowledge between tasks.

5. Discussion

Learning latent representations for Deep Reinforcement Learning (RL) can lead to many benefits including
a lower dimensional input, access to a denser training signal and representations that are useful for multiple
tasks. Typically, when latent or pre-learned representations are used for Deep RL all of the extracted features
390 are used to learn a policy. However this naive approach can be improved upon by only passing the features
that are relevant for the current task to the Deep RL algorithm. This improves learning efficiency because the
dimensionality of the input is reduced further and also improves flexibility because there is less interference
between features when the task changes. Despite these benefits, mechanisms for attending to subsets of
features in Deep RL tend to rely on backpropagation which makes them slow to learn and inflexible to
395 changes in task. As a result, this negates the many potential benefits of being able to rapidly and dynamically
attend to subsets of features. To solve this problem we proposed a novel algorithm, termed Selective Particle
Attention (*SPA*), that uses a particle filter to attend to subsets of features using only reward feedback, and
sequential importance sampling instead of backpropagation.

5.1. Key Results

400 We evaluated *SPA* on two different tasks. The first task was a multiple choice task involving naturalistic
images. *SPA* was able to achieve close to ceiling performance on this task for several examples and dramatically
out-performed a naive version that attended to all features. Inspection of *SPA*'s vector of attention values
showed that it was able to quickly change its configuration in response to changes in the reward function.
This highlights how *SPA* can be used to quickly respond to unannounced changes in the environment and
405 improve the overall efficiency of learning.

The aforementioned multiple choice task can be viewed as a single step Markov Decision Process (MDP) or
a supervised learning problem where the network has to predict the reward associated with each image. *SPA*
can therefore theoretically be applied to a range of classification or regression problems such as MNIST (Deng,
2012) or CIFAR-10 (Krizhevsky et al., 2009). However, the purpose of this research was to explore whether
410 *SPA* could be applied to RL tasks/multi-step MDPs. This is a substantially harder problem because there
are temporal dependencies and the reward feedback is both delayed and sparse. With this in mind, we chose
the second task to be a simple 2D video game. As with the multiple choice task, *SPA* was able to perform
significantly better than the naive approach that attended to all features. This provides evidence that *SPA*
can improve the efficiency of learning on complex RL problems that require temporal dependencies. It also
415 demonstrates that *SPA* is a general approach that can be used with a variety of Deep RL algorithms so long as
they estimate a value function. In addition to out-performing the naive approach, *SPA* also performed better
than an approach that randomly selected a subset of features to attend to. This demonstrates that *SPA* is
able to attend to features in a targeted manner and that the benefits of *SPA* are not simply due to random
dimensionality reduction. As a final baseline on the 2D video game, we compared *SPA* to self-attention and
420 again *SPA* performed significantly better. This highlights the benefit of using approximate inference instead
of backpropagation to rapidly and dynamically attend to features.

Interestingly, in order to successfully apply *SPA* to the 2D object collection game the attention vector was
only updated every 1000 time-steps rather than on every time step. This was necessary because it allowed the
Deep RL algorithm time to respond to the change in attention and attempt to learn a useful function of the
425 currently attended features. This is akin to a person learning based on a single hypothesis for a fixed amount

of time before deciding whether to change to another hypothesis or not. This may explain why attentional inertia is prevalent in children and adults (Anderson et al., 1987; Burns and Anderson, 1993; Richards and Anderson, 2004; Longman et al., 2014), as the brain requires time to evaluate a given hypothesis before deciding whether to switch attention. Future work should systematically explore this apparent trade-off between the potential benefits of switching to a new hypothesis and the time needed to sufficiently evaluate a hypothesis.

Part of the reason for the design of the object collection game was that it allowed for the rewarded shape to be easily changed during learning. This change can be manipulated to correspond to a change in the reward function or state space. We found that *SPA* was significantly better equipped to deal with such changes compared to other naive approaches that either attended to all or a random subset of features. In particular, *SPA* showed an extremely fast recovery in response to a change in state space. This is often considered an extremely difficult problem as deep neural networks typically fail catastrophically when the input distribution is changed (Lake et al., 2017). Nevertheless, the small drop in performance and rapid recovery to asymptotic levels suggests that *SPA* was able to transfer much of the knowledge that it had acquired before the change in state space. These results further support the findings of the multiple choice task, which demonstrated *SPA*'s improved ability to deal with changes in the environment.

5.2. Why Selective Particle Attention is Beneficial

We believe the ability of *SPA* to focus on a subset of features based on the current task is beneficial for several reasons. Firstly, it helps to reduce the dimensionality of the problem, which reduces the impact of noisy features and the complexity of the function that needs to be learned. This improves generalization because the learned function does not fit spurious features and therefore ignores any changes to them. This benefit of dimensionality reduction can even be seen in the case of *SPARANDOM*, which out-performed *SPAALL* on the object collection game. *SPA* takes this one step further however, by providing targeted dimensionality reduction rather than selecting a random subset of features. Furthermore, the selective attention of *SPA* helps to reduce interference between tasks by guiding learning onto different sets of features. This results in the learning of different sub-networks that can be quickly identified based on the current task. This greatly improves the capacity of the Deep RL algorithm to represent different functions and to switch between them in the face of changes to the environment. Finally *SPA* uses approximate Bayesian inference, rather than backpropagation, to learn which features to attend to and this leads to learning on a much faster time scale.

5.3. The Mechanisms Behind Selective Particle Attention

The key component of *SPA* is a particle filter, which is used to infer the vector of attention values using only reward feedback. One of the primary benefits of using a particle filter is that it relies on sampling to produce an approximate value of a hidden variable. This is important because the potential number of feature combinations that need to be evaluated for a given task can be extremely large. In our case there were 2^{512} potential feature combinations and so it would be computationally infeasible to evaluate all of them. However, by using only 250 particles we were still able to converge to a satisfactory solution thanks to the iterative re-sampling procedure of the particle filter. Future work should explore how the number of particles in *SPA* affects its ability to find the best combination of features and therefore its asymptotic performance on tasks. The number of particles may be a limitation in real world applications where inference time is critical and computational resources are limited. If the number of underlying features is large then many particles may be required to approximate the posterior distribution. However, it is worth noting that, as the particles converge, the likelihood of the observed variable need only be calculated once for particles with identical states.

The movement step of the particle filter is guided by bottom-up attention, which introduces a bias towards the most active features. This is equivalent to a prior over the hypothesis space, favouring hypotheses with relatively few features. Interestingly, this is consistent with findings in neuroscience that people tend to make decisions based on individual features before reasoning about objects that involve more complex combinations of features (Farashahi et al., 2017; Choung et al., 2017). Similarly, people find it harder to perform classification tasks as the number of relevant dimensions increases (Shepard et al., 1961). The bottom-up attention captured in the movement step of the particle filter therefore seems to introduce a biologically plausible bias over the hypothesis space.

The observation step of the particle filter is guided by top-down attention. This process involves evaluating different hypotheses based on their ability to predict future reward (Mackintosh, 1975). It has been proposed

that a similar mechanism may occur in the corticostriatal circuitry of humans (Radulescu et al., 2019a). The proposal states that different pools of neurons in the PFC may represent different hypotheses about the structure of the current RL task. These pools then compete via lateral inhibition and connections to the striatum help to favour pools that lead to reward. This process parallels the observation step of *SPA*, whereby hypotheses that are more predictive of reward are more likely to be re-sampled and out-compete other hypotheses. In addition, the phenomenon of representing and testing multiple hypotheses during RL appears to be prevalent in human populations (Wilson and Niv, 2012).

5.4. Dependency on Underlying Representations

The effectiveness of *SPA* naturally depends on the nature of the features or representations that it is attending to. If in real world applications the underlying representations are not abstract enough and there is not a subset of features that are sufficient to achieve the desired task performance then the effectiveness of *SPA* will be limited. In the human brain it has been proposed that the usefulness of selective visual feature attention decreases as you move back through the visual stream (Lindsay and Miller, 2018). This is because features present later in the visual stream consist of higher-order representations that are increasingly abstract. For example, one of the object categories in the multiple choice task was faces, which are known to be represented later on in the visual stream of the brain (Grill-Spector et al., 2018). Having such a representation makes the multiple choice task easy for the brain because it only has to attend to one feature rather than a collection of low level features. This also reduces the need to consider lots of complicated hypotheses. Future work should test whether *SPA* displays similar behaviour by testing whether its performance decreases as attention is applied to earlier convolutional layers.

The updating of the particle filter in *SPA* operates on a much shorter time scale compared to the training of the deep neural network components via backpropagation. This reflects the problem faced by the brain as learning in the visual stream is slow and so the brain needs to use attention to rapidly pick from pre-existing representations in order to solve the current task. However, gradual learning does still occur in the visual stream and is thought to underlie the gradual change from novice to expert on perceptual categorization tasks (Schyns et al., 1998). From the perspective of *SPA*, gradual learning in VGG-16 could lead to the emergence of new features for the particle filter to attend to and for the Deep RL algorithm to utilise. As the learning would be slow and occur over several tasks, these new features would generalize over the tasks being repeatedly performed by the agent. Observed behaviour then becomes an interaction between two processes occurring at very different time scales: (1) the slow emergence of new features predictive of temporally stable elements in the environment, and (2) the rapid selection of existing features driven by immediate (or short term) reward. Future work should therefore explore whether slow incremental learning in VGG-16 would improve the performance of *SPA* over a continuous sequence of different tasks.

The weights of VGG-16 are the result of extensive pre-training on an image classification task known as Imagenet (Deng et al., 2009). Importantly, the data that was used to pre-train VGG-16 involved images that were not used for the multiple choice task and the object collection game. In theory, using the original Imagenet dataset that it was trained on may have lead to better results in the multiple choice task. This is because VGG-16 will have already been trained to produce feature values that distinguish between the object categories in Imagenet, making it easier for *SPA* to attend to discriminating sets of features. This hypothesis is supported by our analysis of the features produced by VGG-16 during the multiple choice task. For the combination of image categories that *SPA* performed best on, the vectors of mean feature values were further apart in Euclidean space compared to the combination of image categories that *SPA* performed worst on. This suggests that the more dissimilar the feature values are between the different image categories, the easier it is for *SPA* to discriminate between them and quickly change its focus of attention.

This dependence of *SPA* on the properties of the underlying representations that it attends to opens up several interesting avenues of future research. In particular, it would be interesting to explore whether the use of disentangled representations (Higgins et al., 2016, 2018) could further improve performance. Independent factors of variation may be easier to attend to because the informative features are separate from each other and so only simple hypotheses are required to solve the task at hand rather than complex combinations of features. Another major benefit of disentangled representations would be that the resulting attention vector would be more interpretable as each attended feature has a natural interpretation; e.g., colour or shape.

6. Concluding Remarks

530 We have presented a novel algorithm, termed Selective Particle Attention (*SPA*), that rapidly and flexibly identifies subsets of latent features for a downstream Deep Reinforcement Learning (RL) algorithm. *SPA* is a general approach that is independent of the method used to extract the latent representations or the Deep RL algorithm that is applied to them, so long as the Deep RL algorithm approximates a value function. Using two tasks as a proof-of-concept, we show that *SPA* greatly improves the efficiency and flexibility of
535 learning compared to approaches that use all of the latent features for Deep RL or that use backpropagation to attend to specific features. At its core, *SPA* uses a particle filter to infer which features are relevant for the current task using only reward feedback. Crucially, this particle filter uses approximate inference instead of backpropagation and so it can be updated rapidly and dynamically. The inference process is based on the magnitude of activation of the features and how well they predict future reward, which is inspired by
540 theories from neuroscience. Future work should explore how the nature of the latent representations affects the behaviour of *SPA*. For example, does *SPA* benefit from attending to latent representations that are increasingly abstract or disentangled, and can they be learned slowly over many tasks to complement the fast learning of *SPA*?

Acknowledgments

545 This work was funded by the UK Biotechnology and Biological Sciences Research Council (BBSRC). We thank NVIDIA for a hardware grant that provided the Graphics Processing Unit (GPU) used to run the simulations. We also thank Martin Eimer for comments on an initial draft of this manuscript.

References

- 550 Anderson, D.R., Choi, H.P., Lorch, E.P., 1987. Attentional inertia reduces distractibility during young children’s tv viewing. *Child Development* , 798–806.
- Bengio, Y., 2012. Deep learning of representations for unsupervised and transfer learning, in: *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 17–36.
- Besse, C., 2009. Particle filter-based policy gradient in pomdps .
- 555 Bichot, N.P., Heard, M.T., DeGennaro, E.M., Desimone, R., 2015. A source for feature-based attention in the prefrontal cortex. *Neuron* 88, 832–844.
- Bramlage, L., Cortese, A., 2021. Generalized attention-weighted reinforcement learning. *Neural Networks* .
- Burns, J.J., Anderson, D.R., 1993. Attentional inertia and recognition memory in adult television viewing. *Communication Research* 20, 777–799.
- 560 Choung, O.h., Lee, S.W., Jeong, Y., 2017. Exploring feature dimensions to learn a new policy in an uninformed reinforcement learning task. *Scientific reports* 7, 1–12.
- Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: *2009 IEEE conference on computer vision and pattern recognition*, Ieee. pp. 248–255.
- Deng, L., 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine* 29, 141–142.
- 565 Desimone, R., Duncan, J., 1995. Neural mechanisms of selective visual attention. *Annual review of neuroscience* 18, 193–222.
- Donoso, M., Collins, A.G., Koechlin, E., 2014. Foundations of human reasoning in the prefrontal cortex. *Science* 344, 1481–1486.
- 570 Doucet, A., Johansen, A.M., et al., 2009. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering* 12, 3.

- Dwibedi, D., Tompson, J., Lynch, C., Sermanet, P., 2018. Learning actionable representations from visual observations, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 1577–1584.
- Farashahi, S., Rowe, K., Aslami, Z., Lee, D., Soltani, A., 2017. Feature-based learning improves adaptability without compromising precision. *Nature communications* 8, 1–16.
- 575
- Fei-Fei, L., Fergus, R., Perona, P., 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence* 28, 594–611.
- Finn, C., Tan, X.Y., Duan, Y., Darrell, T., Levine, S., Abbeel, P., 2015. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *arXiv preprint arXiv:1509.06113* 25.
- 580 Grill-Spector, K., Weiner, K.S., Gomez, J., Stigliani, A., Natu, V.S., 2018. The functional neuroanatomy of face perception: from brain measurements to deep neural networks. *Interface Focus* 8, 20180013.
- Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., Lerchner, A., 2018. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230* .
- Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., Lerchner, A., 2016. beta-vae: Learning basic visual concepts with a constrained variational framework .
- 585
- Higgins, I., Pal, A., Rusu, A.A., Matthey, L., Burgess, C.P., Pritzel, A., Botvinick, M., Blundell, C., Lerchner, A., 2017. Darla: Improving zero-shot transfer in reinforcement learning. *arXiv preprint arXiv:1707.08475* .
- Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. *Neural computation* 9, 1735–1780.
- Igl, M., Zintgraf, L., Le, T.A., Wood, F., Whiteson, S., 2018. Deep variational reinforcement learning for pomdps, in: *International Conference on Machine Learning*, PMLR. pp. 2117–2126.
- 590
- Iqbal, S., Sha, F., 2019. Actor-attention-critic for multi-agent reinforcement learning, in: *International Conference on Machine Learning*, PMLR. pp. 2961–2970.
- Jaderberg, M., Mnih, V., Czarnecki, W.M., Schaul, T., Leibo, J.Z., Silver, D., Kavukcuoglu, K., 2016. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397* .
- 595
- Jiang, J., Heller, K., Egner, T., 2014. Bayesian modeling of flexible cognitive control. *Neuroscience & Biobehavioral Reviews* 46, 30–43.
- Jones, M., Canas, F., 2010. Integrating reinforcement learning with models of representation learning, in: *Proceedings of the Annual Meeting of the Cognitive Science Society*.
- Koechlin, E., 2014. An evolutionary computational theory of prefrontal executive function in decision-making. *Philosophical Transactions of the Royal Society B: Biological Sciences* 369, 20130474.
- 600
- Krizhevsky, A., Hinton, G., et al., 2009. Learning multiple layers of features from tiny images .
- Lake, B.M., Ullman, T.D., Tenenbaum, J.B., Gershman, S.J., 2017. Building machines that learn and think like people. *Behavioral and brain sciences* 40.
- Lange, S., Riedmiller, M., 2010. Deep auto-encoder neural networks in reinforcement learning, in: *The 2010 International Joint Conference on Neural Networks (IJCNN)*, IEEE. pp. 1–8.
- 605
- Lindsay, G.W., Miller, K.D., 2018. How biological attention mechanisms improve task performance in a large-scale visual system model. *ELife* 7, e38105.
- Longman, C.S., Lavric, A., Munteanu, C., Monsell, S., 2014. Attentional inertia and delayed orienting of spatial attention in task-switching. *Journal of Experimental Psychology: Human Perception and Performance* 40, 1580.
- 610
- Ma, X., Karkus, P., Hsu, D., Lee, W.S., Ye, N., 2020. Discriminative particle filter reinforcement learning for complex partial observations. *arXiv preprint arXiv:2002.09884* .

- Mackintosh, N.J., 1975. A theory of attention: Variations in the associability of stimuli with reinforcement. *Psychological review* 82, 276.
- 615 Manchin, A., Abbasnejad, E., van den Hengel, A., 2019. Reinforcement learning with attention that works: A self-supervised approach, in: *International Conference on Neural Information Processing*, Springer. pp. 223–230.
- Miller, E.K., Cohen, J.D., 2001. An integrative theory of prefrontal cortex function. *Annual review of neuroscience* 24, 167–202.
- 620 Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K., 2016. Asynchronous methods for deep reinforcement learning, in: *International conference on machine learning*, pp. 1928–1937.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. *nature* 518, 529–533.
- 625 Mott, A., Zoran, D., Chrzanowski, M., Wierstra, D., Rezende, D.J., 2019. Towards interpretable reinforcement learning using attention augmented agents, in: *Advances in Neural Information Processing Systems*, pp. 12329–12338.
- Nair, A., Pong, V., Dalal, M., Bahl, S., Lin, S., Levine, S., 2018. Visual reinforcement learning with imagined goals. *arXiv preprint arXiv:1807.04742* .
- 630 Niv, Y., Daniel, R., Geana, A., Gershman, S.J., Leong, Y.C., Radulescu, A., Wilson, R.C., 2015. Reinforcement learning in multidimensional environments relies on attention mechanisms. *Journal of Neuroscience* 35, 8145–8157.
- Paneri, S., Gregoriou, G.G., 2017. Top-down control of visual attention by the prefrontal cortex. functional specialization and long-range interactions. *Frontiers in neuroscience* 11, 545.
- 635 Parisotto, E., Song, F., Rae, J., Pascanu, R., Gulcehre, C., Jayakumar, S., Jaderberg, M., Kaufman, R.L., Clark, A., Noury, S., et al., 2020. Stabilizing transformers for reinforcement learning, in: *International Conference on Machine Learning*, PMLR. pp. 7487–7498.
- Radulescu, A., Niv, Y., Ballard, I., 2019a. Holistic reinforcement learning: the role of structure and attention. *Trends in cognitive sciences* 23, 278–292.
- 640 Radulescu, A., Niv, Y., Daw, N.D., 2019b. A particle filtering account of selective attention during learning. *2019 Conference on Cognitive Computational Neuroscience* .
- Reynolds, J.H., Heeger, D.J., 2009. The normalization model of attention. *Neuron* 61, 168–185.
- Richards, J.E., Anderson, D.R., 2004. Attentional inertia in children’s extended looking at television, in: *Advances in child development and behavior*. Elsevier. volume 32, pp. 163–212.
- 645 Rudenko, E., 2019. Algorithms and programs of suboptimal nonlinear filtering for markov processes, in: *AIP Conference Proceedings*, AIP Publishing LLC. p. 020017.
- Saenz, M., Buracas, G.T., Boynton, G.M., 2002. Global effects of feature-based attention in human visual cortex. *Nature neuroscience* 5, 631–632.
- 650 Schrimpf, M., Kubilius, J., Hong, H., Majaj, N.J., Rajalingham, R., Issa, E.B., Kar, K., Bashivan, P., Prescott-Roy, J., Schmidt, K., et al., 2018. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv* , 407007.
- Schyns, P.G., Goldstone, R.L., Thibaut, J.P., 1998. The development of features in object concepts. *Behavioral and brain Sciences* 21, 1–17.

- 655 Shen, X., Yin, C., Hou, X., 2019. Self-attention for deep reinforcement learning, in: Proceedings of the 2019 4th International Conference on Mathematics and Artificial Intelligence, Association for Computing Machinery. p. 71–75.
- Shepard, R.N., Hovland, C.I., Jenkins, H.M., 1961. Learning and memorization of classifications. *Psychological monographs: General and applied* 75, 1.
- 660 Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* .
- Srinivas, A., Laskin, M., Abbeel, P., 2020. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136* .
- Sutton, R.S., Barto, A.G., 1998. *Reinforcement learning: An introduction* .
- 665 Treue, S., 2003. Visual attention: the where, what, how and why of saliency. *Current opinion in neurobiology* 13, 428–432.
- Treue, S., Trujillo, J.C.M., 1999. Feature-based attention influences motion processing gain in macaque visual cortex. *Nature* 399, 575–579.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. *arXiv:1706.03762*.
- 670 Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al., 2019. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* 575, 350–354.
- Wang, Q., Hao, Y., Cao, J., 2020. Adrl: An attention-based deep reinforcement learning framework for knowledge graph reasoning. *Knowledge-Based Systems* 197, 105910.
- 675 Wilson, R.C., Niv, Y., 2012. Inferring relevance in a changing world. *Frontiers in human neuroscience* 5, 189.
- Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., Fergus, R., 2019. Improving sample efficiency in model-free reinforcement learning from images. *arXiv preprint arXiv:1910.01741* .
- Yuezhang, L., Zhang, R., Ballard, D.H., 2018. An initial attempt of combining visual selective attention with deep reinforcement learning. *arXiv preprint arXiv:1811.04407* .
- 680 Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D., Lillicrap, T., Lockhart, E., et al., 2018. Relational deep reinforcement learning. *arXiv preprint arXiv:1806.01830* .
- Zhang, X., He, J., Stojanović, V., Luan, X., et al., 2021a. Finite-time asynchronous dissipative filtering of conic-type nonlinear markov jump systems. *Science China Information Sciences* .
- 685 Zhang, X., Wang, H., Stojanovic, V., Cheng, P., He, S., Luan, X., Liu, F., 2021b. Asynchronous fault detection for interval type-2 fuzzy nonhomogeneous higher-level markov jump systems with uncertain transition probabilities. *IEEE Transactions on Fuzzy Systems* .

7. Appendix

7.1. Full Algorithm for Updating Attention

Algorithm 1 shows the full pseudo-code for updating the attention vector \mathbf{A} . The input consists of vectors of past normalised mean feature map values, states and estimated returns from time t to $t + Z$, as well as the movement probability ϕ . For the multiple choice experiments Z was equal to 0 and for the object collection game it was equal to 9. A value of 9 was chosen for the object collection game because it corresponded to the length of the n-step updates used in the Deep RL algorithm (see Section 7.3). By averaging over multiple time-steps in Algorithm 1, the variance of the feature probabilities and the likelihood of each particle state are reduced.

Algorithm 1 UpdateAttention($\mathbf{v}, \mathbf{s}, \mathbf{R}, \phi$)

Receive normalized mean feature map values $\mathbf{v} = \{v_t, \dots, v_{t+Z}\}$, states $\mathbf{s} = \{s_t, \dots, s_{t+Z}\}$, returns $\mathbf{R} = \{R_t, \dots, R_{t+Z}\}$ and movement probability ϕ

1. Perform Movement Step

Calculate feature probabilities

$$p^k = \frac{\frac{1}{Z+1} \sum_{z=t}^{t+Z} \exp(v_z^k * \tau_{\text{BU}})}{\max_j \frac{1}{Z+1} \sum_{z=t}^{t+Z} \exp(v_z^j * \tau_{\text{BU}})}$$

Update each particle state x^i with probability ϕ

$$P(x_k^i = n) = \begin{cases} p^k & \text{for } n = 1 \\ 1 - p^k & \text{for } n = 0 \end{cases}$$

2. Perform Observe Step

Calculate the likelihoods of each particle x^i

$$A_k^i = \frac{x_k^i}{\sum_{j=1}^K x_j^i}$$

$$\delta^i = \frac{1}{Z+1} \sum_{z=t}^{t+Z} (R_z - V(s_z; \mathbf{A}^i))^2$$

$$P(\mathbf{R}|x^i) \propto \exp(-(\delta^i - \min_j \delta^j) * \tau_{TD})$$

Re-sample particles based on calculated likelihoods

$$P(x') = \frac{\sum_{i=1}^N P(\mathbf{R}|x^i) \mathbb{I}(x^i = x')}{\sum_{i=1}^N P(\mathbf{R}|x^i)}$$

$$x^i \sim P(x')$$

3. Update Attention

Calculate mean particle state and normalise

$$\bar{x}_k = \frac{1}{N} \sum_{n=1}^N x_k^n$$

$$A_k = \frac{\bar{x}_k}{\sum_{j=1}^K \bar{x}_j}$$

7.2. Full Algorithm for the Multiple Choice Task

For the multiple choice task we passed the results of the attention layer to a fully connected three layer feed forward Deep Neural Network (DNN) (512 units, 256 units and 1 unit). The DNN was used to approximate the value of a given state, which in this case was the value of a given image. To select an action the value of each image was calculated and one of the images was chosen in an ϵ -greedy manner, with $\epsilon = .2$. As each trial was based on a single choice, the problem is equivalent to a single step Markov Decision Process (MDP). We therefore trained the DNN after each trial to minimise the squared error between the reward experienced on a trial (r_t) and the predicted value of the chosen image (s_t):

$$J(\theta) = \frac{1}{2} (r_t - V(s_t; \theta, \mathbf{A}))^2 \tag{14}$$

$$\nabla_{\theta} J(\theta) = -(r_t - V(s_t; \theta, \mathbf{A})) \nabla_{\theta} V(s_t; \theta, \mathbf{A}) \tag{15}$$

Where θ are the parameters of the DNN, \mathbf{A} is the attention vector, s_t is the image chosen at time t and r_t is the reward associated with the image chosen. We used RMSProp as an optimizer and the hyper-parameter values can be found in Table 1. For the particle filter of *SPA*, the return (R_t) in Equation 8 was simply the reward experienced at the end of each trial (r_t).

7.3. Full Algorithm for the Object Collection Game

For the object collection game we passed the results of the attention layer to a fully connected layer (1024 units) followed by a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) network (256 units), which output the state value (1 unit) and a softmax distribution over actions (2 units). The network therefore resembled an advantage actor critic (A2C) architecture (Mnih et al., 2016), whereby the predicted state value from the critic was used to calculate the advantage function. We chose this setup to explore whether *SPA* could work when the network also had to compute a policy. The agent received every 8th frame as input. The cost function of the critic was the mean squared error between the return and the predicted value:

$$J(\theta) = \frac{1}{2}(R_t - V(s_t; \theta, \mathbf{A}))^2 \tag{16}$$

$$\nabla_{\theta} J(\theta) = -(R_t - V(s_t; \theta, \mathbf{A})) \nabla_{\theta} V(s_t; \theta, \mathbf{A}) \tag{17}$$

Where θ are the parameters of the DNN, \mathbf{A} is the attention vector, s_t is the frame of the game at time t and R_t is the return from state s_t . The actor was updated using the advantage calculated from the critic:

$$A(s_t, a_t) = R_t - V(s_t; \theta, \mathbf{A}) \tag{18}$$

$$\nabla_{\theta} L(\theta) = A(s_t, a_t) \nabla_{\theta} \ln \pi(a_t | s_t; \theta, \mathbf{A}) + \beta \nabla_{\theta} H(\pi(a_t | s_t; \theta, \mathbf{A})) \tag{19}$$

Where H represents the entropy of the softmax distribution over actions and is included to encourage exploration. β is a free parameter that controls the strength of this entropy term and was set to 0.01 for all simulations. In all cases, the return R_t from state s_t was estimated using n-step Temporal Difference (TD) learning:

$$R_t^n = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n V(s_{t+n}; \theta, \mathbf{A}) \tag{20}$$

This was used for calculating the target for the critic via Equation 16 and for calculating the advantage in Equation 18. The training procedure was the same as in Mnih et al. (2016) with t_{max} set to 10. More specifically, the agent selects actions up until t_{max} or the trial ends. At this point the the gradients are calculated for each of the n-step TD-learning updates using the longest possible n-step return. For example, the update for the last state will be a one-step update whereas the update for the first state will be a t_{max} -step update. These gradients are all applied in a single update.

The returns calculated using the aforementioned procedure were also used in Equation 8 for calculating the observation step of the particle filter. However, the movement and update steps of the particle filter were not performed on every training step (when t_{max} or the trial ends). Instead they were performed every c_{max} training steps, we found that this improved both stability and reduced training time. The values used for the hyper-parameters can be found in Table 1.

Table 1: Hyper-parameter values used for the Multiple Choice task (MC) and Object Collection game (OC).

Parameter	Value (MC)	Value (OC)	Description
N	250	250	Number of particles
τ_{BU}	10	1	Strength of bottom-up attention
τ_{TD}	10	10	Strength of top-down attention
c_{max}	1	1000	Frequency of attention updates
t_{max}	1	10	Maximum length of return
ϵ	.2	-	Probability of selecting a random action
λ	.00025	-	Learning rate for RMSProp
κ	.95	-	Momentum for RMSProp
ι	.01	-	Constant for denominator in RMSProp
β	-	0.01	Exploration strength
γ	-	.99	Discount factor for future rewards
m	-	8	Number of frames skipped
α	-	0.0001	Learning rate for Adam optimizer
d	-	128	Dimension of the query and key vectors for self-attention