Events and Machine Learning

Augustus Hebblewhite[1], Jakob Hohwy[2], and Tom Drummond[1]

[1]Department of Electrical and Computer Systems Engineering, Monash University, Melbourne, Australia
[2]Cognition & Philosophy Lab, Department of Philosophy, Monash University, Melbourne, Australia
{augustus.hebblewhite, jakob.hohwy, tom.drummond}@monash.edu

Abstract

An important question to understanding events is how they arise from unstructured sensory information. (Baldwin & Kosie, 2019) and (Shin & DuBrow, 2019) point out some problems with the 'Event Segmentation Theory' account of how events are formed, and offer plausible extensions or alternatives. We outline one interesting way that 'events' might arise in machine learning contexts, where we have some ability to look under the hood of the cognitive machinery, and consider how this relates to the other accounts given in this issue.

In exploring some possible conditions for event segmentation, the articles in this special issue invite discussion on several important high-level questions. This commentary focuses on the two articles, (Baldwin & Kosie, 2019) and (Shin & DuBrow, 2019), which give accounts of how cognitive agents might segment event boundaries. These papers highlight the role of organising structure, especially hierarchy and abstraction, in information processing. Our own research engages with these ideas in the context of creating machine learning agents that make sensible decisions in complicated environments, and we see a great deal of value in fields outside of computer science as we converge on some of the same questions.

The account of event segmentation from Baldwin and Kosie is closest to Event Segmentation Theory (EST; (Zacks, Speer, Swallow, Braver, & Reynolds, 2007)), especially – in their words – the feature that "it is when transient prediction error – 'surprisal' – spikes that one experiences the culmination of an event, just as it transitions into the next". Baldwin and Kosie observe however that while there is good evidence that surprisal is correlated with event boundaries, surprisal is too limited to be the sole quantity whose variation defines event boundaries. Shin and DuBrow make a similar objection to EST, observing that change in surprisal is neither a sufficient nor necessary condition for event boundaries. This shortfall in EST motivates the alternate accounts in both papers.

Baldwin and Kosie draw upon the way higher level brain processes appear to modulate attentional sampling. They propose an additional mechanism based in an agent's predictions about their own level of surprisal. Shin and DuBrow favour an idea familiar to those in computer science, in which different events are clusters in a non-parametric latent space of state representations. Event segmentation (through time) occurs when observations are placed in different clusters in the latent space, and experience of new events gives rise to new clusters.

This commentary seeks to briefly compare these mechanisms to how event segmentation could happen in machine learning agents, especially those designed to perform reinforcement learning (RL). RL is a sub-field of machine learning (ML) that aims to design artificial decision makers that observe their environment, take actions, and receive rewards based on their performance.

The popular machine learning algorithms of today might seem to have little to say about events. For better or worse, many of the high profile recent successes within machine learning and RL have been in creating agents that are effective at performing particular tasks, but where these agents do not appear to engage in the sort of state or action representation that 'events' come out of. Some examples; the deep Q learning agents that beat Lee Sedol at Go (Silver et al., 2016) and can solve many of the Atari environments (Mnih et al., 2015) don't have any explicit representation of time; they learn to associate states with expected reward values, but don't pay attention to anything that came before their current state[1]. This is also true of actor-critic based algorithms like Proximal Policy Optimization (PPO) (Schulman, Wolski, Dhariwal, Radford, & Klimov, 2017). In other contexts, algorithms like those used by DeepMind to play StarCraft 2 (Vinyals et al., 2019) utilise a recurrent neural network (Hochreiter & Schmidhuber, 1997) which can represent some temporal information. Parts of the network system can read their own internal states from the time step before, allowing for information of the past to be retained within the network activations. However the recurrent neural network simply incorporates the past information into its current activations; it is not doing much to organise the information temporally.

There are at least two things that one needs from an account of event segmentation in order to actually segment events in some system. Events are related to the variation of some property of the system (e.g. 'surprisal' in EST), and the first thing an account needs is to specify which property is relevant. (Baldwin & Kosie, 2019) and (Shin & DuBrow, 2019) offer candidates here. The second thing an account needs though is to say where exactly the boundaries occur, for any given 'level' of event. We say 'level' here to indicate that events may be hierarchical - there may be 'high level' events like "baking a cake" that are composed of 'low[er] level' events like "put the flour into the bowl". EST, as well as the papers we focus on, leave this second question open. What particular absolute/relative amount of change in surprisal constitutes a new event? How different should the latent space representation be from one time-step to another to count as a new event? These questions are difficult, and arise in different forms in many domains of ontology. To put these ideas into practice in AI however, we might need to come up with answers.

How might ML provide clues for a way forward in this specific problem of picking the boundaries? One approach is to train a machine learning system in a domain where some 'events' happen, to see how well it can adapt to the changes. In (Butz, Bilkey, Humaidan, Knott, & Otte, 2019) a system is trained to control 3 different types of vehicle with different control dynamics. The authors find an architecture that can use observations about its control inputs and motion outputs to help it understand which vehicle mode it is in, and observe that changes in vehicle mode correspond to increases in prediction error which the system can use as a signal that it needs to adapt. Another avenue could be through reinforcement learning. We suspect there are several possible ways RL could inform event segmentation, but for this brief commentary we will focus on 'hierarchical reinforcement learning' (HRL), which is one interesting candidate. HRL is a branch of RL research that explores agents that are designed to explicitly interact with/interpret the environment at different levels of abstraction. A simple HLR model might consist of one module that plans at a high level and another

[1] Strictly speaking the Atari algorithm is fed 4 frames, and so can be thought of as having "memory" of the last 3 frames plus the present one, for example.

that is responsible for low level execution. One fundamental archetype of HRL enriches classical RL agents to be able to call not only actions but also 'options' (Sutton, Precup, & Singh, 1999), that is, sub-policies that generate their own actions given the environment and then pass control back to the top level. For example, an agent may decide it needs to go into the next room, for which it calls the 'go through the door' option and executes all the actions recommended by this option. Once the option terminates, it makes another choice.

This particular framework was chosen from among many interesting HRL structures, but in it we can see one elegant candidate for low level event boundaries; the junction between options. That is to say, we could think of options themselves as low level events[2]. (Sutton et al., 1999) see scope for expanding this into deeper hierarchies; options calling options and so on, with each level of the hierarchy grouping options into higher level 'events', in much the same way as we humans might group muscle contractions into 'reaching for a cup' into 'drinking a cup of tea'. This also speaks to the generalisability needed for robust machine learning systems, and the recombinability of events highlighted in many of the papers of the special issue. It is particularly reminiscent of (although much more simplistic than) the account of hierarchical event representations in (Knott & Takac, 2019).

What we hope to point out with these ML approaches is that they can help us choose actual event boundaries - the second feature of a full account of event segmentation, in our view. This comes out of asking the system to give us something else; performance on predictive tasks, reward maximisation etc., such that event boundaries are chosen by the system to facilitate this task.

How may this play out in practice? Say we have a HRL agent that is trying to learn to make cups of tea. It gets positive feedback (reward) if it makes a good cup of tea or negative feedback (negative reward) if it messes up. As the agent tries to maximise the expected reward it might be the case that this is better done by breaking up the task into separate sub-sequences. If a particular choice of sub-sequence (e.g. 'fill the kettle') is useful, the network will tend to delineate it. If it is not useful, the network will tend to find boundaries elsewhere (e.g., to break up 'fill up kettle' into something different; perhaps 'pick up kettle', 'move kettle under tap', ...). The algorithm works this out because the parameters that define options - both the actions that constitute the option and the termination conditions - can be designed to be differentiable (Bacon, Harb, & Precup, 2017). That is to say, the algorithm can work out how changing these parameters changes the overall expected reward, and if the optimisation goes smoothly they will tend to settle on values that maximise reward. This goal-oriented process of identifying boundaries is likely to correlate with peaks in surprisal, but we might expect it to deviate in some situations, for example if there is some surprising dynamic that is unimportant to our goal.

The ability of current HRL approaches to scale up to rich multi-level hierarchical events, such as making tea, remains to be seen; there are numerous practical challenges. What we hope to have pointed out is one theoretical way for RL to be used in constructing ontologies - an inventory of the "items" in an environment - that include something like events. We see similarity here with our own ongoing experiments, which explore how RL can give rise to ontologies in the domain of 'image segmentation' (labelling which objects lie where in an image). Both research projects are concerned with building high level representations for these ontologies, which we consider an important part of the puzzle of AI generalisation and efficient learning. In confronting these challenges, much

[2]This is not a necessary condition for events in general, as this definition only captures events which consist of sequences of actions

inspiration can be found in the articles in this issue, and the varied scientific disciplines they draw upon.

References

Bacon, P.-L., Harb, J., & Precup, D. (2017). The option-critic architecture. In *Thirty-first AAAI conference on artificial intelligence.*

Baldwin, D. A., & Kosie, J. E. (2019). How does the mind render streaming experience as events? *Topics in Cognitive Science*, *This Volume*.

Butz, M. V., Bilkey, D., Humaidan, D., Knott, A., & Otte, S. (2019). Learning, planning, and control in a monolithic neural event inference architecture. *Neural Networks*, *117*, 135–144.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Knott, A., & Takac, M. (2019). Roles for event representation is sensorimotor experience, memory formation and language processing. *Topics in Cognitive Science*, *This Volume*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... others (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shin, Y. S., & DuBrow, S. (2019). Structuring memory through inference-based event segmentation. *Topics in Cognitive Science*, *This Volume*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... others (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*(1-2), 181–211.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., ... others (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, *575*(7782), 350–354.

Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S., & Reynolds, J. R. (2007). Event perception: a mind-brain perspective. *Psychological Bulletin*, *133*(2), 273.