Learning Actions from Events Using Agent Motions

Nikhil Krishnaswamy, Tuan Do, and James Pustejovsky

nkrishna@brandeis.edu • tuandn@brandeis.edu • jamesp@brandeis.edu

Overview

Work in event visualization from natural language (Coyne and Sproat, 2001; Siskind, 2001; Chang et al., 2015) often struggles with using specified parameters in events. These parameters may be inherent to the event itself (e.g., speed, direction, etc.), or properties of the object undergoing the event (e.g., mass, size, etc.). A computational visualization system uses an inappropriate value for one of these parameters, it may generate a visualization for a given event that does not comport with a human viewer’s understanding of what that event is. Event recognition provides a venue to explore “learning from observation,” and as a domain has achieved recent relevance in human communication with robotic agents (Yang et al., 2010; Paul et al., 2017). Learning can abstract away the parameters that vary across instances of the same motion class, making those parameters underspecified as in the aforementioned visualization problem. For an embodied agent to interact with objects, the agent must use its hands, and the hand motions effect forces upon the object. Thus, we expect that the same parameter abstraction approach can be used for the agent’s hand motions, creating a path toward action recognition.

Causal events are composed of an object model, which captures the change an object is undergoing over time, and an action model, which characterizes the activity that inheres in the causing agent (Pustejovsky and Krishnaswamy, 2016). We present results from an event visualization system using multimodal simulations and methodology from an event learning and composition system to introduce a framework for learning action recognition from the movements of the agent rather than the object. We expect such a framework may be useful for recognizing and evaluating the actions denoted by agent motions enacted without attached objects, e.g., by gestures.

Extracting Actions from Events

Where captured instances contain multiple object configurations or permutations under the same label (for example, building a wall by putting two objects near each other in various orientations), the LSTM learns event progress by changes in object relations, such as the number and relative orientation of EC or “touching” relations between objects. Since it allows the REINFORCE algorithm to generalize a concept (e.g., row) to set of common relations across, all objects captured or simulated instances without a set number of blocks. This makes the parameters that vary across the captured instances underspecified.

As we have shown that underspecified motion features appear to be strong signals of event class for objects moving in isolation, we expect the same principle holds for objects being manipulated by an agent, especially as one of the goals of our reinforcement learning pipeline is to abstract away those parameters whose values vary across the performed or simulated example actions.

For instance, let us return to the semantics of “slide” presented in Figure 1. One of the requirements is that at all times the moving object is kept EC (externally connected) with the supporting surface. Since in a 3D world, motions eventually break down into a series of translations and rotations, all relations between objects can be represented between translations and rotations, as in the reinforcement learning trials. Thus, if “sliding” motions of various speeds and moving in various directions all re- transform roughly as equal reverse motions, then as long as the object is the supporting surface (as the LSTM should produce high values of events of a sort), all these motions given enough performed examples, the REINFORCE algorithm should be able to generate an event sequence centered on the concept of sliding. We have also presented a framework for action learning that relies on abstracting away those motion parameter values that may vary across individual instances and performances of events. These two avenues naturally combine to create a pipeline for action recognition by a computational agent using information from visual and linguistic modalities (cf. (Yang et al., 2014; Yang et al., 2015), and for using action performance and gestural representations of actions as a learnable communicative modality between humans and computers.

Event Classification

Using the VoxSim simulation environment (Krishnaswamy and Pustejovsky, 2016; Krishnaswamy, 2017), we generated three video datasets for input sentences of the imperative form VERB x (or VERB x RELATION y). Amazon Mechanical Turk workers were shown a single animated movie of an event and asked them to provide three heuristically-generated captions (one of which was the original input sentence, the best one).

Sample VoxSim capture for “move the block”

Values assigned to the verb’s underspecified features (e.g., hand motion may take a wide variety of values: uses of speed and direction but always maintains a constant or near-constant vertical offset with the surface (representing the height of the object being moved)), these motions may be interpreted as representing a “slide,” regardless of whether or not any actual object is being moved. If no object is moved along with the hand, this “action model” becomes a “minne” or gestural representation of the action in question.

Conclusion & Future Directions

We have argued and presented evidence that underspecified parameters associated with motion events can serve as reliable indicators of a particular event class. We have also presented a framework for action learning that relies on abstracting away those motion parameter values that may vary across individual instances and performances of events. These two avenues naturally combine to create a pipeline for action recognition by a computational agent using information from visual and linguistic modalities (cf. (Yang et al., 2014; Yang et al., 2015), and for using action performance and gestural representations of actions as a learnable communicative modality between humans and computers.

Learning Complex Events

Even a simple event such as put(x, nearby) requires a series of translations that can be difficult for a computer to distinguish from other types of motions involving changing relations between two objects. For this is a sequential learning problem, we turn to LSTM (Hochreiter and Schmidhuber, 1997) to learn the sequence of primitive events that comprise a complex event. If the sequence can be effectively learned, it should be able to be reproduced by a virtual embodied agent, whose objective is to produce a sequence of actions that resembles movement of objects in the training data.

A sequence of feature vectors, \( f \), which represent the qualitative spatial relations between the objects in the action captures, is fed to an LSTM network along with a frame number and an event. The network outputs \( f(S, i, e) = 0 \) if \( e \) is not significant at frame \( i \). The virtual agent’s objective is then to manipulate the objects in sequence, for a increasing reward as the generated sequence more closely approximates the movement of objects in the training data. We use the REINFORCE algorithm with a Gaussian distribution policy \( \mu(\theta, x) = \text{Gaussian}(\mu, \alpha) \), where \( \mu(\alpha) \) is the degree of freedom in position (2 dimensions) and \( \alpha(\alpha) \) is the degree of freedom in orientation (1 dimension).

Planning is parameterized by \( u : q_k \rightarrow m(\theta, q_k) \), where \( q_k \) is motion performed at step \( k \) and \( x_k \) is current set of relations between objects. After each atomic object manipulation \( u \) we use the LSTM network to estimate how fully \( u \) completes the event in question, then calculate the immediate reward as \( f(S, k, e) = f(S, k, e) \).

The result is a sequence that can be executed by a virtual agent within the VoxSim environment.

Selected References


Teaching virtual agents to perform complex spatial-temporal activities.


Selected References

