Error-based updating of event representations enables prediction of human activity at human scale

Matthew Bezdek[1]*, Tan Nguyen[1]*, Samuel J. Gershman[2], Aaron Bobick[3], Todd Braver[1], Jeffrey Zacks[1]**

[1]Department of Psychological and Brain Sciences, Washington University in St. Louis, St. Louis, MO, USA

[2]Department of Psychology and Center for Brain Science, Harvard University, Cambridge, MA, USA

[3]Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO, USA

*These authors contributed equally to this work.

**Corresponding author:

Email: jzacks@wustl.edu

Abstract

Humans form sequences of *event models*—representations of the immediate situation—to predict how activity will unfold. Two challenging questions about event models are "How is activity segmented into events?" and "How is knowledge about different event types learned and represented?" We constructed a computational model combining a recurrent neural network for short-term dynamics with Bayesian inference over event types for event-to-event transitions. This architecture learns schemas representing knowledge about event types and uses them, along with observed perceptual information, to construct a series of event models, updating them when predictions are violated. The architecture learned to predict activity dynamics from one pass through an 18-hour corpus of naturalistic human activity. When tested on another 3.5 hours of activities, it switched schemas at times corresponding with human segmentation and formed human-like event categories—despite being given no feedback about segmentation or categorization. These results establish that event schemas and updating based on prediction error can ground learning and can naturally reproduce two important features of human comprehension.

Keywords

Event cognition, segmentation, action perception, computational modeling

Introduction

To act effectively in a dynamic, complex environment, humans and other species use memory and knowledge to predict how activity will unfold over time[1]. Knowledge about event types can be encoded in structured representations called *event schemas*[2–4]. One effective means to predict the near future is to use event schemas and perceptual information to construct and maintain an internal model of the current situation (an *event model*). For example, if one is serving a cup of coffee, an event model can help to predict that the person receiving the coffee will pick it up, and how they will move to do so. However, for such a model to be effective it needs to transition when one thing ends and other begins—once the coffee has been taken up, a coffee-serving model is unhelpful. One proposed mechanism is to monitor prediction error and transition when prediction error spikes[5]. Alternatives include proposals that the updating of event models is regulated by monitoring changes in states of the world[6] or by learning about community structure—a network of relations between potential states of the world[7]. All these proposals share the premise that predictive mechanisms are guided by stable representations that encode learned environmental regularities.

Models of comprehension in language[8] and visual perception[9,10] simulate predictive processing by presenting the model with a sequence of states and training it to predict the next state on each timestep. In naturalistic event comprehension, the relevant states often have covariance relationships amongst features and sequential relationships over time. One study[11] trained a gated recurrent neural network using this one-step prediction task on a simplified sequence of human motion tracking data, and found that gating information into the network's event representation improved prediction performance. Another study[7] trained a feedforward neural network to predict sequences of discrete states, where the training environment was

constructed such that possible sequences were generated from a network of 15 states that formed 3 “communities.” Transitions between states within a community were common, whereas transitions that crossed communities were rare. This sort of structure can be viewed as an abstraction of situations in which larger tasks such as serving a cup of coffee, cleaning up, and taking customers’ payments consist of collections of sub-actions, and transitions amongst sub-actions are more frequent than transitions between larger tasks. The network learned about this structure, so that its hidden units represented which community the current state was drawn from. Studies such as these demonstrate that machine learning systems can learn to form stable event representations from sequences of states. Using a corpus of hand-coded representations of entities, actions, and contexts, another study showed that a recurrent neural network could learn to predict the dynamics of how these sequences and features interact and to generalize those dynamics to new classes of events[12].

The Structured Event Memory (SEM)[13] framework extends neural network models of event cognition by encoding different event schemas with separate weight vectors. Approximate Bayesian inference is used to determine which event schema is currently active based on observed scene transitions. In SEM, when the currently active event model has low posterior probability (a prediction error), the algorithm resets the active event model by selecting the most probable event schema for the new event model. The set of RNN weights that are learned during training are *event schemas*, and each activation of an event schema constitutes the construction of an *event model*. The SEM framework has been used to model how activities are segmented into events, how working memory is updated at event boundaries, how a category of event generalizes to a new instance, and how event structure organizes long-term memory[13].

However, all of the previous modeling work has exhibited significant limitations in the ability to simulate event comprehension at the structure and scale of human performance. There are two key reasons for this. First, the representations of each scene are abstracted to the point that they cannot be compared meaningfully with human perceptual representations. Previous simulations have used arbitrary localist codings[7], highly simplified pose representations[11], or pixelwise video representations[13]. Second, the activities used for training and evaluation have been too brief or simplistic to capture the naturalistic structure of human action performance and comprehension. In particular, toy datasets may lack the structure and variability that enables new learning trajectories to emerge that better capture human comprehension. Moreover, model assumptions that may work in toy datasets may fail in a large, naturalistic dataset; in fact, this was a key challenge that needed to be solved in the work reported here, as described below. These two limitations have precluded answering critical questions about the modeling of event comprehension. First, can a cognitive system that transitions event models based on prediction error develop event representations that allow it to learn the dynamics of naturalistic activity? Second, does such an algorithm lead to human-like event segmentation and categorization?

To address the challenges of scaling up modeling of human event comprehension, we adapted the SEM architecture so that it could be trained and evaluated on rich representations of naturalistic human goal-directed activity, which were recorded and processed such that the model’s output scene vectors and internal activations could be directly compared to human performance on a moment-by-moment basis. The model was trained on over 18 hours of recordings of actors completing extended naturalistic activities, which are almost identical to input data that humans experience to learn event structure and an order of magnitude larger than previous toy datasets. Each activity was recorded with three video cameras and an infrared

time-of-flight depth sensor that captured the three-dimensional pose of the body. The identities and positions of objects were tracked over time using semi-automated object tracking, and semantic information about objects was incorporated using a large-scale language model[14]. These recordings were subjected to a dimension reduction process that preserved interpretable information about the dynamics of body movement and the semantics of the objects with which the actor interacted. The computational model was trained to take in the running sequence of these reduced representations and to predict the next timepoint in the sequence, 1/3 of a second later. Here, we report how the model learned and how it segmented activity. We compare the model's segmentation and categorization to human judgments on 3 hours 39 minutes of validation activities.

Results

Models of event comprehension were trained and tested on the Multi-angle Extended Three-dimensional Activities (META) stimulus set, a corpus of naturalistic activities[15]. In each activity, an actor performed a series of 6 to 7 scripted actions in a realistic environment. Because the visual and semantic features processed in mid-level human vision may be the building blocks from which event representations are constructed[16], we used a combination of human and computer vision methods to generate a rich set of these features. From three-dimensional joint position recordings, we calculated features of body pose, velocity and acceleration, as well as inter-hand distance, velocity, and acceleration. To represent the semantic meanings of interactive objects in the activities, we annotated bounding boxes that tracked the positions of objects, then used a language model (GloVe[14]) trained on a large text corpus[17] and translated the name of each object to a vector embedding. We then computed a weighted vector representation of the objects closest to the actor's right hand, and the mean vector representation of all objects currently present in the scene. Principal component analysis reduced a set of 253 input features (object appearances, object disappearances, mean frame-to-frame change in pixel luminance values, skeletal motion features, object semantic features) to a set of 30 features that we presented as *input scene vectors* to the event prediction model.

The core architecture of the model, depicted in Fig. 1, was modified from the Structured Event Memory (SEM) model[13]; we will refer to this as SEM-2.0. This model uses fully-connected recurrent neural networks (RNNs) to represent event schemas, and an approximate Bayesian inference (clustering) process to assign incoming scene vectors to event schemas. On each time step (3 Hz), a currently active RNN is presented with an input scene vector and predicts the next scene vector. The clustering process then compares the posterior probability of the active RNN's prediction, relative to that from all other models, and then either 1) retains the current event model, 2) activates a different event schema from the library, 3) retains the same schema, but resets its activation values, or 4) spawns a new event schema. We will refer to 2-4 as *switching* event models. SEM-2.0 includes hyper-parameters for stickiness, the tendency to keep the active model (to ensure temporal coherence in events), concentration, the tendency to spawn new models, and learning rate to update RNNs. The hyper-parameters used for our experiments are shown in Table 1, and hyper-parameter tuning is described in Supplementary Information.

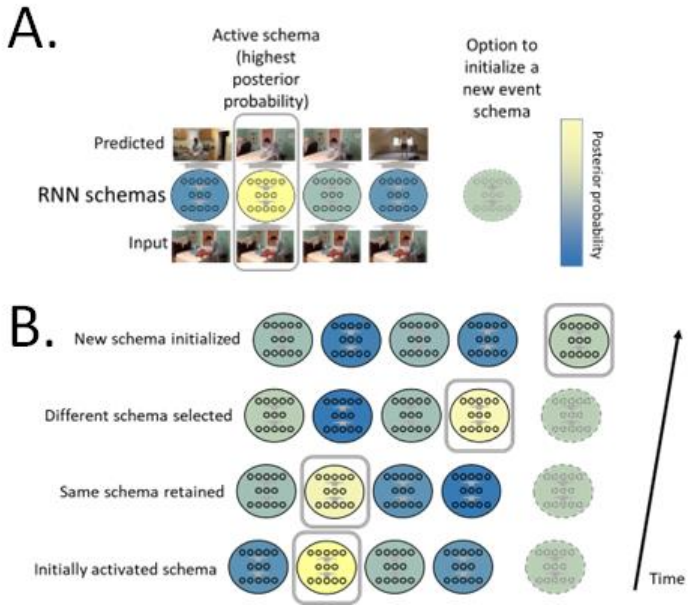


Fig. 1. Overview of SEM architecture. (A) In this hypothetical example, SEM is in the process of training and has generated four event schemas (RNNs). At each time step, the event schemas predict the current input scene vector from the previous scene vector. Based on the posterior probability, SEM keeps the active event schema active, switches to another schema in its library, or spawns a new event schema (depicted as an RNN with a dotted outline). In this case, the active schema is retained. The resulting active event schema updates its weights by backpropagating its prediction error. **(B)** A potential sequence of outcomes. On the first two timesteps, the currently active schema is retained. On the third timestep, SEM switches to a different previously-learned schema. On the fourth timestep, SEM initializes a new schema. (Not shown: SEM separately evaluates the probability of the current schema based on the current RNN hidden unit values and based on re-initializing the RNN. If resetting the hidden unit values is found to be more valuable, they are reset. This allows SEM to model, for example, washing a plate and then immediately washing a second plate.)

Learning Rate	Stickiness	Concentration	Input scene vector dimensionality	Number of RNN hidden units
1e-3	1e7	1e-1	30	16

Table 1. Model's hyper-parameters.

The original SEM implementation (SEM-1.0)[13] was able to account for structure in several small datasets, and it deployed new event schemas effectively, such that event schemas were

frequently reused. However, when we applied SEM-1.0 to the corpus dataset, we observed that the model used a small number of event schemas to account for most of 22 hours of activities. There were three modeling assumptions that led to this ineffective use of resources. First, newly spawned event schemas were initialized to random weights; this disadvantages new event schemas for the learning of naturalistic activities that afford rich “general knowledge” about feature co-occurrence and dynamics. In SEM-2.0, we initialized newly spawned event schemas with weights from a single RNN that was trained on all scene vectors up to that point in time. This can be understood as an approximate form of “empirical Bayes” estimation. Second, the process SEM-1.0 used to assign prior probabilities to schemas was the sticky Chinese Restaurant Process[18] (a type of Dirichlet process). The Dirichlet process is a commonly-used prior distribution that has a “rich-get-richer" property[19]—a small number of large clusters accounts for most observations. In SEM 1.0, this property caused most timepoints to be assigned to only a small number of event schemas. Although “rich-get-richer” might be appropriate to some clustering applications, this property might not be desirable in applications where a more balanced prior distribution is desired[19,20]. In SEM-2.0, we instead used a uniform prior distribution, while retaining stickiness and concentration parameters. Third, in SEM-1.0 active schemas made predictions about the current scene from the current scene vector, but inactive schemas were asked to make predictions from a random vector. This approach is computationally efficient, but it puts inactive schemas at a disadvantage. SEM-2.0 provides all schemas with input scene vectors from previous timepoints (see Methods for details) to make predictions.

From 128 activities (total duration: 21 h 43 m, range: 5 m 35 s to 19 m 16 s, mean: 10 m 11 s), activities were randomly split into a training set of 108 activities (18 h 4 m) and a validation set of 20 activities (3 h 39 m). In contrast to the common practice with deep learning models of interleaving learning with repeated presentation of stimuli[21], SEM-2.0 encountered and learned each training activity only once, watching the whole activity before moving to the next activity. This training regime approximates the structure of naturalistic training, in which each event is experienced only once, but similar events and event sequences recur. After training on each activity, the validation set was tested with learning turned off.

SEM-2.0 learns to predict naturalistic scene dynamics and outperforms comparison models

The time course of prediction error for the trained SEM-2.0 model contains spikes of high prediction error punctuating periods of stable predictions (Fig. 2A). SEM-2.0’s mean prediction error for validation activities decreased over the course of training (Fig. 2B, top), as SEM-2.0 partitioned event knowledge into discrete recurrent neural networks and used a Bayesian updating process to select the active event schema. To assess the impact of event knowledge partitioning, we created a *generic model*, composed of a single RNN that predicted the incoming scene vector from the last input scene vectors. This model used the same parameters as SEM-2.0 for its one event schema, but could not switch or spawn new event schemas. The generic model had higher mean validation prediction error across 16 simulations (SEM: 0.575, 95% CI (0.571, 0.579); generic: 0.636, 95% CI (0.622, 0.651); also see Fig. 2B and 2C, top panels, for 4 simulations).

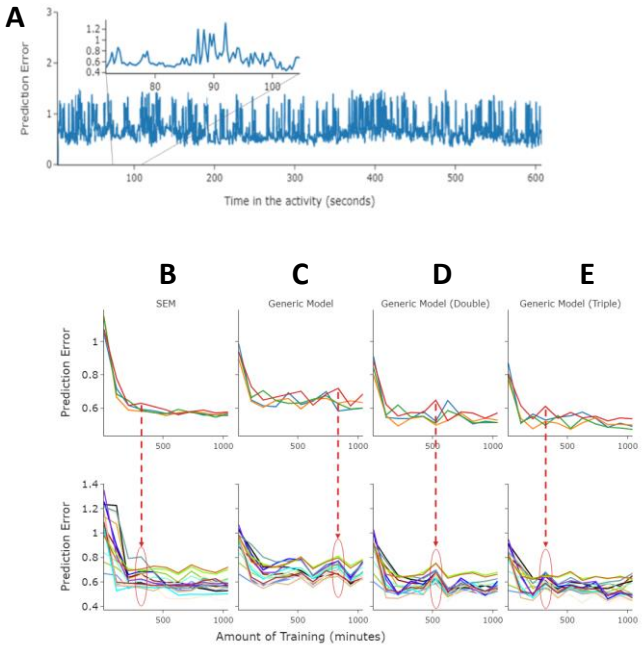


Fig. 2. SEM-2.0 shows intermittent spikes in prediction error and an interference-resistant learning trajectory. (A) Prediction error of SEM-2.0 shows regular spiking. Illustrated here is prediction error for one cleaning activity in the validation set after training on approximately 450 minutes of other activities in the training set. The inset above shows the prediction error for a smaller time window of the larger activity. **(B-E)** Prediction errors for SEM-2.0 and generic models over the course of training. Each point is the mean prediction error for all validation activities at each evaluation time over the course of training. **(B-E)** Top: Mean prediction error for all validation activities across training for SEM-2.0, generic model with the same, double, and triple the number of hidden units. Four colors indicate four different simulations with different random weight initializations and different orders of training activities. Using discrete event schemas and a Bayesian updating process, SEM-2.0 reduces prediction error over the course of training. The generic model reduces prediction error over the course of training, to a level of prediction error slightly higher than SEM-2.0. Generic models with double and triple the number of hidden units can reduce errors lower than SEM-2.0; however, all generic models show greater interference from new learning (mean prediction error fluctuates across training). Bottom: Prediction errors for all validation activities across training for SEM-2.0, generic model with the same, double, and triple the number of hidden units, for the “red” simulation. SEM-2.0 shows interference around minute 300-th, with prediction errors for a couple of validation activities increasing. However, the generic model shows catastrophic interference around

minute 800-th, with almost all validation activities' prediction errors increasing. The same pattern can be observed in the generic models with double and triple number of hidden units.

Compared to the generic model, SEM allocates more storage to representing the results of learning because it accumulates a library of event schemas. For comparison, we considered expanded generic models had double (2x) or triple (3x) the number of units in the hidden layer of the model. Notably, these models experienced much more weight updating than the generic model or SEM; whereas only the active model in SEM is able to update its weights, the original and expanded generic models update all their weights on each time step, which increases the amount of weight updating 2.5 times and 4 times, respectively. As shown in Fig. 2D-E (top panels), expanding the size of the hidden layer in the generic model also reduced the mean validation prediction error across 16 simulations (2x: 0.545, 95% CI (0.537, 0.554); 3x: 0.511, 95% CI (0.505, 0.518)). Thus, whereas adding the ability to switch event schemas reduces prediction error, this set of simulations demonstrate that it is also possible to reduce prediction error by increasing the hidden layer size. However, there are differences in the weight updating processes between SEM and the generic models: Whereas the generic models must update its weights with each new input scene vector, the schema weight updating mechanism in SEM is able to silo data from a newly-encountered event without compromising the integrity of the other event schemas in its library. This reduced interference: As shown in Fig. 2C-E (top panels), the generic models' average prediction error for validation activities sometimes increased as they saw more training activities, suggesting that the new learning from recent training activities had interfered with previously acquired learning that was beneficial to predict validation activities. Fig. 2C-E (bottom panels) shows prediction errors for all validation activities across training for the "red" simulation. The generic model shows catastrophic interference at many points across training (e.g. around minute 510-th or minute 800-th), with almost all validation activities' prediction errors increase. The same pattern can be observed in the generic models with double and triple number of hidden units.

To quantify the benefit of modeling scene dynamics with an RNN, we created a pair of very simple comparison models: The *last scene model* simply used the last scene vector as its prediction for the current scene, instead of generating a prediction as the output of an RNN. The *recent scene model* used a moving average of the previous three scene vectors as its prediction for the current scene vector. Both models performed poorly compared to SEM (mean prediction errors of 1.99, 95% CI (1.93, 2.05) and 2.23, 95% CI (2.17, 2.28), respectively).

SEM-2.0 segments activities in a human-like fashion without being reinforced for segmentation

On each timestep, SEM selects an RNN to remain active or become active; this can be interpreted as an *event label* categorizing that timestep. For each event label e_n, SEM assumes the event schema e_n is active and generates a predicted scene vector conditioned on e_n. The probability of assigning event label e_n to the input scene vector monotonically decreases with the difference between the input scene vector and the predicted scene vector generated by event schema e_n. Moreover, for the active event label (the event label assigned to the previous scene vector), SEM calculates two probabilities: the probability of observing the input scene vector if the active event continues, and the probability of observing the input scene vector if the active event restarts. An *event boundary* is inferred when event labels for subsequent scene

vectors are different, or when the probability of restarting the active event is higher than the probability of continuing. *Events* are the intervals between event boundaries.

A key test of the model is to determine whether it generates human-like event boundaries for naturalistic stimuli. To compare SEM-2.0 to human performance on event updating and understanding, we used data from the META stimulus set[15]. Normative event boundaries were collected from an online sample of participants. Participants were instructed to watch a randomly-selected sequence and to press a button each time one meaningful unit of activity ended and another began. Each participant was assigned a grain of *coarse*, defined as the largest meaningful units of activity, or *fine*, defined as the smallest meaningful units of activity, and instructed appropriately. Participants could segment multiple videos. We collected 30 segmentations per grain per activity.

To quantify model-to-human and human-to-human segmentation agreement, we calculated the proportion of human raters who segmented during each timestep, and computed the point-biserial correlation between that normative human segmentation time series and (a) each individual human rater, and (b) SEM-2.0 models' segmentation. The possible range of this correlation depends on the number of event boundaries; thus, comparing correlations for two segmenters who identify different numbers of event boundaries can be misleading. Therefore, we scaled the correlation[22] based on its minimum possible and maximum possible values, given the number of boundaries observed. We also assessed how likely the result would occur by chance by permutation testing: shuffling event boundaries while preserving event lengths (see Methods). As shown in Fig. 3A, SEM-2.0's point-biserial correlations across training were much larger than would be expected by chance, and are within the lower end of the distribution of human-to-human segmentation agreement.

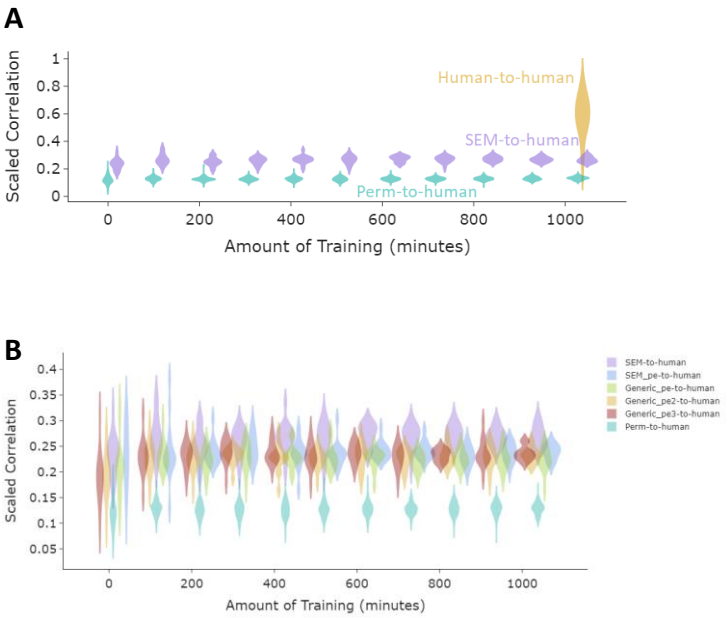


Fig. 3. The model's segmentation agrees with that of human observers, though SEM-2.0 was never given feedback on segmentation. (A) Scaled point-biserial correlations across all validation activities for SEM-2.0 simulations, humans, and permutations across training. Each purple violin plot is a distribution of point-biserial correlation for different initializations of SEM-2.0. Each light sea-green violin plot is a null distribution generated by shuffling SEM-2.0's boundaries while preserving SEM-2.0's event lengths. The goldenrod violin plot is a distribution of scaled point-biserial correlation for different human subjects (which doesn't change over the course of training). SEM-2.0's segmentation agreement with human segmenters is bigger than expected by chance, and falls within the lower end of human segmenters. ***(B)*** Comparison of agreement with human segmentation between SEM-2.0 and generic models. SEM-to-human denotes SEM-2.0's boundaries derived from Bayesian inference (event label switches). SEM_pe-to-human denotes SEM-2.0's boundaries derived from its prediction error. Generic_pe-to-human, Generic_pe2 -to-human, and Generic_pe3 -to-human denote boundaries derived from prediction errors in the generic model, the generic model with double and triple the number of hidden units. Each violin plot is a distribution of scaled point-biserial correlation for different initializations and training orders. Segmentation agreement between generic models and humans is bigger than expected by chance, though it is smaller than segmentation agreement between SEM-2.0 and humans.

SEM-2.0's segmentation agreement out-performs generic models

Because the generic model has only one event schema, it doesn't intrinsically produce event boundaries. To estimate event boundaries for the generic models, we identified timesteps where its prediction errors were highest. For a fair comparison between the generic models and SEM-2.0, we also applied the same algorithm to SEM-2.0's prediction errors to obtain event boundaries. We calculated point-biserial correlations for generic model's boundaries and SEM-2.0's boundaries. While SEM-2.0's and generic models' event boundaries derived from prediction errors align with human segmentation significantly more than chance, SEM-2.0's event boundaries derived from the Bayesian inference process (event label switches) had higher agreement with human segmentation than these PE-derived boundaries (Fig. 3B). This result suggests that, although increasing the number of hidden units in the generic model reduces prediction error (Fig. 2C), it does not lead to more human-like event segmentation.

SEM-2.0 produces flurries of updating at some event boundaries

Examining the time course of SEM-2.0's updating reveals that there are moments when SEM-2.0 makes a flurry of rapid switches within a relatively short time window. Figs. 4A and 4B show SEM-2.0's boundaries for two example validation activities. Fig. 4C shows the distribution of elapsed durations between consecutive boundaries. The distribution is heavily right-skewed, and approximately 42% of consecutive boundaries have durations below 2 seconds, showing that SEM-2.0 makes flurries of rapid switches within a short time. Even though we know that humans agree where event boundaries are, we don't know if it is the case that the brain experiences one boundary or a series of boundaries. Thus, SEM-2.0 makes the novel prediction that the brain might sometimes experience a series of switches before settling into a new stable event model.

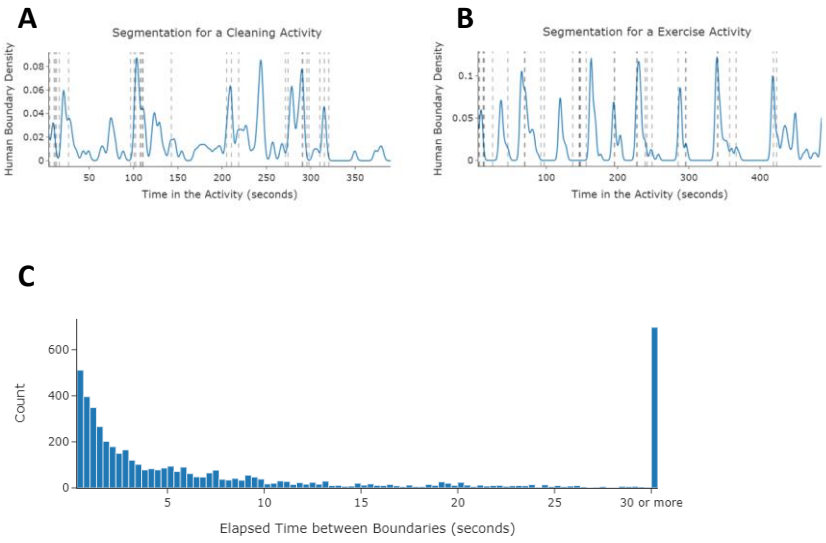
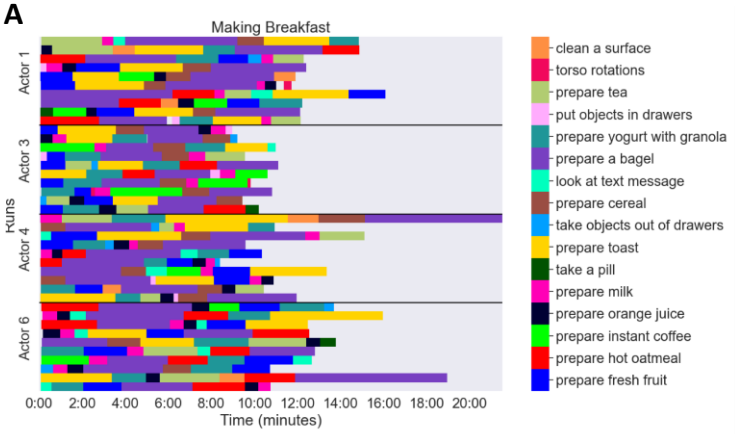


Fig. 4. The trained model shows occasional flurries of updating. Examples of SEM-2.0's boundaries for **(A)** one "cleaning room" activity and **(B)** one "exercise" activity. For the cleaning activity, SEM-2.0 made flurries of switches around the 10^{th} and 105^{th} seconds. For the exercise activity, SEM-2.0 made flurries of switches around the 5^{th} and 150^{th} seconds. **(C)** Distribution of elapsed time between SEM-2.0's consecutive boundaries. Durations longer than 30 seconds were collapsed together. Most of the pairs of consecutive boundaries have small durations, indicating that SEM-2.0 made rapid switches within short intervals.

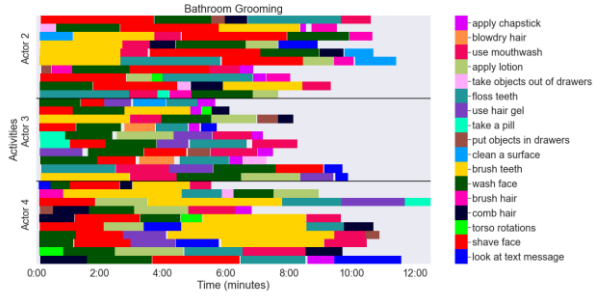
SEM-2.0 forms schemas that correspond with judges' action categories, and generalizes across actors and environments without being reinforced for categorization

To comprehend an activity, one needs to not only capture its boundaries but also to relate the current activity to previous knowledge. SEM-2.0's event labels model the act of classifying a current moment as an instance of a previously-learned activity. To evaluate SEM-2.0's ability to classify, we used the *script action labels* that were provided to the actors before recording of each activity. We had two human raters watch videos of the activities and identify the beginning and ending of each of the 6-7 scripted actions per activity. Agreement between raters was high, with a median discrepancy of 1.40 s between raters. Discrepancies were resolved by computing the mean of the time annotations. We compared human-rated action labels and SEM-2.0's event labels. Fig. 5A shows examples for these script action labels, and Fig. 5B shows SEM-2.0's event labels for one activity.

A


Making Breakfast
Actor 1
Actor 3
Actor 4
Actor 6
Runs
0:00 2:00 4:00 6:00 8:00 10:00 12:00 14:00 16:00 18:00 20:00
Time (minutes)
clean a surface
torso rotations
prepare tea
put objects in drawers
prepare yogurt with granola
prepare a bagel
look at text message
prepare cereal
take objects out of drawers
prepare toast
take a pill
prepare milk
prepare orange juice
prepare instant coffee
prepare hot oatmeal
prepare fresh fruit

B


Bathroom Grooming
Actor 2
Actor 3
Actor 4
Activities
0:00 2:00 4:00 6:00 8:00 10:00 12:00
Time (minutes)
apply chapstick
blowdry hair
use mouthwash
apply lotion
take objects out of drawers
floss teeth
use hair gel
take a pill
put objects in drawers
clean a surface
brush teeth
wash face
brush hair
comb hair
torso rotations
shave face
look at text message

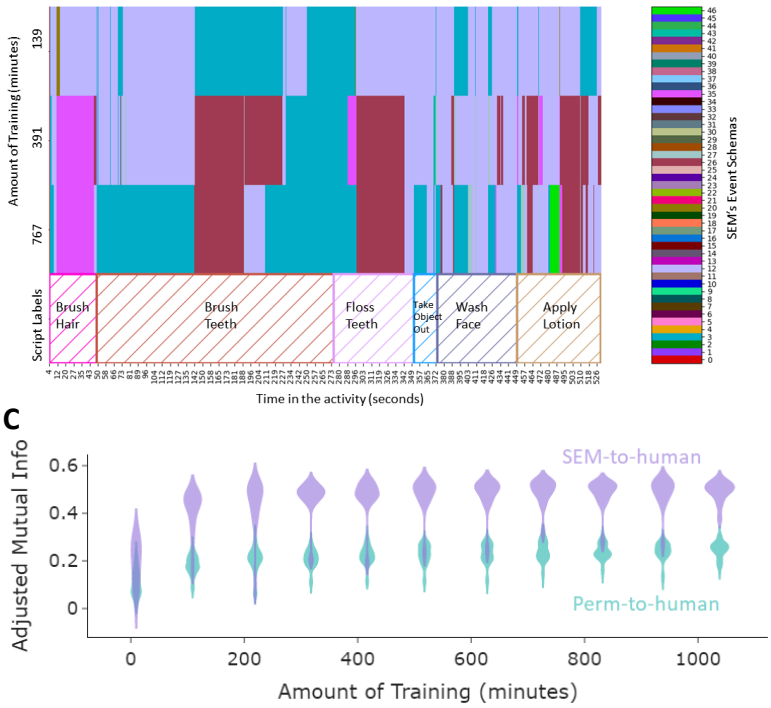


Fig. 5. The model's event categorization agrees with script action labels, despite never receiving feedback on categorization. (A) Examples of script action labels for "bathroom grooming" activities. Each row is an activity, and each group on the y-axis indicates the actor performing these action sequences. Each color represents an action label. X-axis indicates the length of the activity. Notably, actions varied in their order and durations across activities. **(B)** An example of SEM-2.0's active event schemas for one bathroom grooming activity at three different points (top three rows) in training (139 minutes, 391 minutes, and 767 minutes) and script action labels (last row). As training progressed, the SEM-2.0's assignments of event schemas to timepoints more closely corresponded to human raters' assignments of script action labels to timepoints (e.g. the trained SEM-2.0 activated one schema for most of the "brush hair" script action label). **(C)** Categorization agreement between SEM-2.0 event labels and human action labels. Each purple violin plot is a distribution of adjusted mutual information scores between different simulations of SEM-2.0. Light sea-green violins indicate distributions of adjusted mutual information scores for permutations: each permutation is a shuffle of SEM-2.0's boundaries while preserving event lengths. SEM-2.0's adjusted mutual information scores increase across training, and remains significantly bigger than chance.

To quantify SEM-2.0's agreement with human action categories, we calculated the *adjusted mutual information* between SEM-2.0's event labels and the scripted action labels. Mutual information quantifies the information shared by the two partitioning (clustering) algorithms (both SEM-2.0 and humans partition input scene vectors into clusters) and thus can be employed as a

categorization similarity measure. If SEM-2.0 categorizes input scene vectors in a human-like way, and SEM-2.0's event schemas generalize across instances of the same action, mutual information between SEM-2.0 and script action labels will be high (see Methods). The adjusted mutual information score corrects for the chance level of expected mutual information between two partitions. To test the significance of the adjusted mutual information between SEM-2.0's event schemas and script action labels, we performed a permutation test by randomly shuffling the order of SEM-2.0's events 100 times and computing the adjusted mutual information between SEM-2.0's event schemas and action labels (see Methods). As shown in Fig. 5C, the adjusted mutual information between SEM-2.0's event schemas and script action labels was significantly higher than the permuted null distributions, indicating that SEM-2.0 can form event schemas that generalize across actions performed by different actors in different environments. Examination of the correspondence between SEM-2.0 categories ad script action labels revealed that SEM's categories generalized across actors and environments: The same schema was often activated for the same script action performed by different actors in different environments (see Supplementary information.)

Note that perfect adjusted mutual information is possible only if two partitions have the same number of categories; if two partitions differ in the number of categories (i.e., if one is finer than the other), the best possible adjusted mutual information score is lower. Over training, SEM-2.0 develops finer-grained event schemas and finer-grained segmentation: By the end of training, its mean event length was 29.47 s, suggesting that it was identifying sub-events of the scripted actions (mean length of 79.90 s). This lowers the adjusted mutual information score. To better understand how finer-grained segmentation affects the relationship between SEM-2.0's evolving partitioning and human categories, we used two complementary categorization metrics, purity and coverage. This analysis confirmed that SEM-2.0's categorization at the end of training captured sub-units of the action script labels (see Supplementary Information).

The rate of event schema formation slows over learning

A notable aspect of SEM-2.0's behavior is the rate at which it forms new event schemas over the course of its training. Fig. 6 shows the number of SEM-2.0's event schemas over the course of training. Early in training, when SEM-2.0 hasn't learned much, it keeps creating new event schemas to capture the statistics of the stimuli. In the middle and late of training, SEM-2.0 starts to reuse its event schemas to accommodate novel stimuli while using these novel stimuli to update the weights of its existing event schemas. Note that the total number of event schemas at the end of training differ across simulations, ranging from 20 to 59 event schemas. Because simulations have different training orders of activities, this result suggests that the model's event schemas depend on training history.

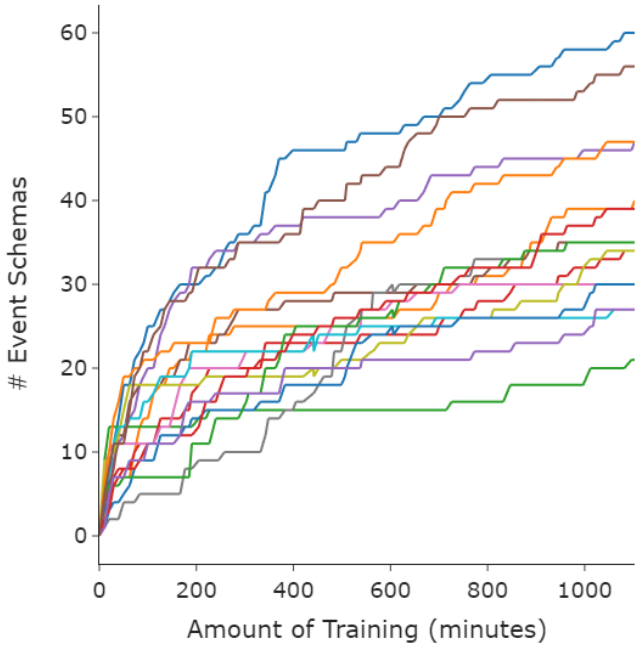


Fig. 6. SEM-2.0's rate of schema generation levels off across training. Color lines are simulations of SEM-2.0 with different weight initializations and training orders. SEM-2.0 created many new event schemas early in training and reused these event schemas in the middle and late of training.

Comparison with input-deprived models

To explore the contribution of motion features and semantic features on SEM-2.0's prediction, segmentation, and categorization, we created two input-deprived SEM-2.0 versions, semantics-deprived and motion-deprived models (see Methods) and compared them to SEM-2.0. Deprived models have higher prediction error than SEM-2.0, with large contribution coming from respective deprived features. Full SEM-2.0 and deprived models had comparable segmentation correlation with humans and categorization agreement with humans (see Supplementary Information).

Discussion

Event models facilitate effective real-time responding to dynamic activity, planning for the future, and encoding for long-term memory[16,23–25]. The present simulations, conducted with a large-scale naturalistic corpus of everyday activity, tested an architecture for implementing event models based on two distinctive features: First, event knowledge is encapsulated in schemas, with schema selection governed by inferring latent structure in the dynamics of activity. Second, the metric for evaluating schema fit, and thus triggering updating, is prediction error. The results show conclusively that encapsulating schema knowledge aids prediction, leads to human-like action categories, and produces event segmentation that aligns with the segmentation perceived by human observers.

Prediction

The model was trained to predict 1/3 of a second into the future based on a sequence of scene representations. The nature of these representations is significant: Rather than using arbitrary sequences or simple toy examples, we utilized densely annotated naturalistic recordings of everyday activity. The simulation input and output scene vectors were constructed to capture key properties of object and action processing in the primate visual system[26]; we view these as a rough summary of the results of early- and mid-level visual processing. This strategy allowed for direct comparison of human and model event segmentation and categorization, enabled the model to generalize its schemas across actors and environments, and should facilitate generalization to other everyday activities as well as comparison to neurophysiological data.

The model's ability to learn is grounded in its hybrid architecture—a gated recurrent network to learn short-run event dynamics joined with inference over event schema process to model transitions from event to event. These results extend previous findings that recurrent neural networks are effective for sequential learning[8,21,27], and that gating enables learning longer-duration time dependencies[27]. Comparison models that did not incorporate event schema inference were able to achieve reasonable performance on the problem, particularly if they were endowed with larger numbers of hidden units; however, they were less able to protect previous learning from interference[28,29]. Such interference can be circumvented with interleaved training[30,31], but interleaving is not characteristic of natural experience. SEM's segregation of event knowledge into multiple schemas distinguishes it from previous models of event cognition[7,11,32], in which event dynamics was learned by a single neural network instead of a library of networks.

Segmentation

The model's segmentation corresponded with human segmentation of extended naturalistic activity. In SEM, segmentation reflects the updating of the model's current event representation in response to prediction error. This instantiates the core principle of Event Segmentation Theory[5]. Crucially, the model was never provided with information about human segmentation; its segmentation corresponds with human segmentation as a natural side-effect of implementing the algorithm. The results are consistent with a small-scale demonstration using SEM-1.0[13]. (Under reasonable linking assumptions, the generic RNN also approximated human-like event segmentation[11].) Going beyond previous demonstrations, SEM-2.0 produced the first evidence of unreinforced human-like segmentation with large-scale naturalistic activities.

Categorization and Generalization

SEM-2.0's event schemas matched human action categories, again without being provided with information about human categories. It generalized event schemas to performances by different actors in different environments, indicating that it could learn underlying event dynamics while smoothing surface features. In previous work with SEM-1.0[13], a componential representational format that explicitly marked actors, object, and actions was able to generalize across thematic roles in a toy example: When pretrained on sequences such as "Ask(Tom, Charan)" followed by "Answer (Charan, Tom)," SEM-1.0 generalized to "Ask(Dick, Bruce)" followed by "Answer(Bruce, Dick)." The current results show generalization in a much more naturalistic context, with large numbers of examples, and no explicit marking of thematic roles.

Conclusion and Extensions

Despite the importance of event comprehension for understanding human cognition and action, previous modeling attempts have been conducted on stimuli which were highly abstract or simplified, and too brief to capture the variability and structure of naturalistic events[7,11,13]. The META dataset[15] provides a more rigorous test for computational models of event comprehension by allowing for a model training and testing regime that resembles human learning experience. From the original SEM architecture, several cognitive assumptions were adapted, and the resulting architecture captures key properties of human event comprehension when trained and evaluated on rich representations of naturalistic human goal-directed activity.

One important problem for future research is to account for the learning of hierarchical structure in activity[33]. A sequence of small-scale events such as "browsing the menu," "call the waiter/waitress," and "order meals" may occur reliably as part of a larger event such as "order food at a restaurant." Behavioral[34,35] and neuroimaging[36,37] data provide evidence that human event comprehension represents such temporal hierarchy. There are at least two promising approaches to modeling hierarchical event structure[13]. One is to extend the schema selection process to learn transition dynamics between events, effectively grouping sub-events belonging to a larger event together. Another strategy is to allow SEM to learn events on multiple timescales simultaneously. Scaling the noise variance parameter for schemas (RNNs) can change sensitivity to prediction errors, which modulates granularity.

Another central problem to be explored is the reuse of previously-learned schemas to create a new one. For example, an event "selling stock at a coffee shop" could include elements of event "selling stock" and elements of event "have coffee at a coffee shop." One approach to such reuse is to represent events compositionally, decomposing components into elements and combining these components in a rule-like manner[38]. Notably, Elman and McRae[12] demonstrated that a single, large neural network could combine elements learned in different events. The authors trained a neural network model on two different events: a person cuts food in a restaurant with a knife and a person cuts themselves with a knife and bleeds. The network was tested with a new event: a person was in the restaurant and cut themself; and it correctly inferred that the person bleeds, combining elements from the two learned events. A challenge for both rule-like and neural network approaches to knowledge reuse is to capture the flexible and graded nature of human performance. The use of a large-scale corpus and a hybrid computational architecture offers promise to develop mechanistic explanations of how humans and other animals solve these challenging cognitive problems, and others.

Methods

Model architecture and implementation

The core architecture of the Structured Event Memory (SEM) model has two main components: a library of recurrent neural network (RNN) schemas, and a generative model of event labels[13]. The specific RNN architecture was a four-layer, fully-connected neural network with gated recurrent units (GRU), a leaky rectified linear activation function (leaky ReLU), and 50 percent dropout for regularization. The generative model clusters each incoming scene vector to an event schema by inferring which event schema generated the scene vector, using local maximum a posterior (MAP) estimation (Fig.1). For each incoming vector, SEM computes likelihoods that the vector belongs to each event schema by comparing event schemas' predictions with the scene vector, with higher similarity indicating higher likelihood. SEM-2.0 assigns prior probabilities based on a sticky "uniform process" (sUP) prior, explained further below[19]. Priors and likelihoods are combined to compute posterior probabilities for all event schemas, and the incoming scene vector is assigned to the event schema with the highest posterior probability. Consequently, in this architecture, event boundaries are defined as selecting a new schema (or selecting to re-initialize the currently active schema). This inference mechanism over event schemas by estimating local-MAP approximates Bayesian inference; exact Bayesian inference would require computations for all past clustering outcomes. However, comparisons between local-MAP and more exact forms of Bayesian inference have shown their performance on certain problems to be highly similar[39].

In SEM-1.0[13], there were three sources of bias (modeling assumptions) that created an imbalance in the relative activation of event schemas. One source of bias was that newly spawned event schemas were initialized to random weights. This initialization disadvantages new event schemas for the learning of naturalistic activities like the META corpus, because the environment is rich with general features and dynamics, such as where objects are typically found and how bodies can move, as well as event-specific information. To address this imbalance, we initialized newly spawned-event schemas with weights from an RNN that was trained on all scene vectors up to that point in time. Second, the process used to assign priors to event schemas was the sticky Chinese restaurant process, which assigns higher prior probabilities to event schemas that have more frequently been activated in the past. This led to the activation of a small number of event schemas for most time points and rarely activated newly-spawned event schemas; therefore, in SEM-2.0, we replaced this process with the sUP, in which large and small clusters have equal probability. As in the sCRP, the sUP has a hyperparameter called stickiness that controls the tendency to remain in the currently active event, and a hyperparameter called concentration that controls the likelihood of spawning new event schemas. Removing this "rich-get-richer" property helped SEM-2.0 to use event schemas more evenly. Third, SEM-1.0 asked active schemas to make predictions about the current scene by feeding them scene vectors from previous timepoints while asking inactive schemas to make predictions by feeding them a random vector. This approach helped the authors circumvent a computational challenge because predictions from inactive schemas could be cached (because the random vector was constant, predictions were also constant) and used to compute likelihoods for the inactive event schemas. However, that approach placed inactive event schemas at a disadvantage because the input scene vectors to these event schemas were not informative to predict the current scene vector, while the input scene vectors to the

active event schemas was the scene vector from previous timesteps. Consequently, inactive event schemas were less likely to be selected and update their weights (since only the active event schema updates its weights at a specific timestep), resulting in only some initial event schemas activating and updating their weights most of the time. Relatedly, because event schemas in SEM-1.0 were trained to predict current scene vectors from either previous scene vectors or random vectors, their predictive power was compromised. We therefore modified SEM-2.0 so that both active and inactive schemas were provided with the previous scene vectors as input. To retain efficient processing, we parallelized the calculation of predictions from active and inactive schemas. These changes led to more even use of event schemas and reduced prediction error (see Supplementary Information).

All code was implemented in Python, using Tensorflow library for neural network implementation: https://github.com/mbezdek/extended-event-modeling. Hyperparameters were chosen by performing a grid-search across several potential values and selecting the configuration of values that minimized prediction error and most closely matched the mean number of human event boundaries (see Supplementary Information).

Materials

SEM-2.0 was trained on the Multi-angle Extended Three-dimensional Activities (META) stimulus set[15]. This stimulus set contains over 25 hours of performances of everyday activities of about 10 minutes each, performed in realistic environments (see SI for more information). Performances were captured with a Kinect V2 device, which includes a video camera and a time-of-flight depth sensor[40], and two other video cameras. The Microsoft Kinect for Windows SDK 2.0 was used to infer the three-dimensional positions of the actors' skeletal joints from the recorded depth image stream. The three-dimensional joint positions for each frame were translated to place the mid-spine joint at the origin (0,0,0). Joint coordinates were then rotated about the Y-axis to align the left and right shoulder joints on a common plane in the Z-axis. Raw features were smoothed with a rolling mean of seven frames (three frames before and three frames after). From the joint position data, we calculated joint velocity and acceleration, as well as the inter-hand distance, velocity, and acceleration.

Semantic information about objects that the actors touched was captured using the following method. For a subset of video frames at ten seconds intervals, human annotators marked the positions and identities of objects with bounding boxes. Then, using the subset of labeled frames, a computer vision tracking model was used to track the positions of objects both forward and backward in time between the labeled frames (Siam Region Proposal Network[41]). Each tracker was dropped when the model's confidence fell below a threshold and the Hungarian algorithm was used to match forward and backward tracks. Object appearance and disappearance features were binary, taking values of ones for frames in which at least one object begins to be tracked and frames in which at least one object is no longer tracked, respectively. For each frame, the name of each object present in the scene was converted to a 50-dimensional vector using the GloVe language model[14] trained on the large collection of Internet encyclopedia entries and news articles of the Wikipedia 2014 and GigaWord 5[17] corpora. A 50-dimensional feature set was created as an average of the embeddings of all objects present in the current frame. In addition to these features representing the semantic meanings of objects in the environment, we created features of the semantic meaning of objects nearest to the actor's right hand. The three-dimensional Euclidean distance between the actor's right hand joint and the average depth in Z axis of the pixels in the object's bounding box was

calculated. Then, a weighted average of the object vectors was calculated, scaled by the inverse of the distance to the actor's hand. In total there were 102 object-related features, comprising object appearances and disappearances, 50 features for the average embedding of all the objects present in the scene, and 50 features of the nearest objects to the actor's hand.

We performed principal component analysis to reduce the dimensionality of the feature vectors. Dimensionality reduction was performed separately on body motion and semantic features, to allow for modality-specific calculations of predictions and errors. The resulting set of features contained 30 dimensions (14 body motion dimensions, 13 semantic dimensions, 2 dimensions for object appearances and disappearances, and 1 dimension for the correlation of pixel luminance between successive video frames). This dimensionality reduction preserved 76% of the original variance of the full feature set.

Training and Testing regimen

The Kinect body tracking produced large-scale errors for some of the activities (for example, in activities where the actor's lower body was occluded, the Kinect mistook the actor's upper body as the whole actor). To prevent activities with large tracking errors from having an undue influence on the model's performance, we developed a filtering algorithm to select high-quality activities for training and validation. First, we calculated the range of all possible values for each derived body motion feature. The algorithm has two thresholds to: 1) mark bad features and 2) filter bad activities. For each feature in each activity, the feature was considered good if y% of the values for that feature fell inside of the ninetieth percentile of the possible range, otherwise it was marked. If more than x% of all features of the activity were marked, the activity was filtered. A grid-search was used to determine values for x and y. See Fig. 7 for the proportion of activities that are left after applying each pair of x and y thresholds. The x-axis and y-axis in the plot correspond to threshold x and threshold y respectively, and the annotated values indicate the proportion of activities left. Because SEM-2.0 showed learning saturation after watching approximately 70 to 90 activities, we can afford to apply stringent thresholds to maintain high quality without worrying about depleting the training dataset. As a result, we ended up with 128 activities out of the 149 activities in the META stimulus set (86%); an activity is selected if more than 80% if its features are considered good, and a feature is considered good if more than 80% of its value fall within the ninetieth percentile of the possible range.

Filtering thresholds for activity inclusion

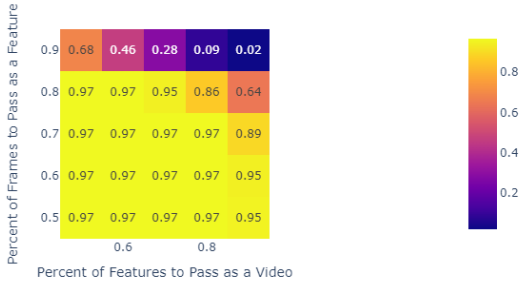


Fig. 7. The proportion of activities passing the filtering algorithm with different combinations of two thresholds. Y-axis represents the proportion of frames falling within the ninetieth percentile so that a feature can be marked as good. X-axis represents the proportion of good features so that the video can be used for training and testing. Each annotated number is the proportion of activities passing the filtering algorithm with each combination of threshold.

We split the 128 activities into a training set (108 activities) and validation set (20 activities). SEM-2.0 watched activities from the training set, and at each time step the weights in the active RNNs in SEM-2.0 were updated by back-propagating the squared error between SEM-2.0's predictions and the input scene vectors. In this way, SEM-2.0 was only permitted to learn from each training activity once, to match the uniqueness of experience of humans. We evaluated SEM-2.0 after it watched each training activity by presenting SEM-2.0 with all validation activities, while freezing SEM-2.0's parameters to prevent weight updating.

Generic model architecture

The SEM architecture builds a library of RNN event schemas and chooses between them using a Bayesian inference process over event schemas. For comparison with SEM-2.0, we created a generic model consisting of a single RNN and removed the Bayesian inference mechanism. In this way, we tested a system that applies a single RNN event schema to predict the next time step against the full SEM-2.0 model which selects between separate RNNs to predict the next time step. Apart from this key difference, the parameters were matched between SEM-2.0 and the generic model. SEM-2.0's library of RNNs adds a larger number of parameters above the generic model's but the same amount of weight updating (learning) on each time step; the exact number of weights depends on the number of event schemas created. To address this issue, we also created variations of the generic model with double and triple the number of units in the hidden layer. This increases the number of weights to be more comparable with the number of weights in SEM-2.0—but gives the generic model a large advantage in the amount of weight updating it experiences.

Evaluation Metrics

Whole prediction error

In order to measure how a model learns to predict over the course of training, we calculated its prediction error for each pass through the validation set. The prediction error for each timestep is the Euclidean distance between the active model's prediction and the input scene vector at that timestep, and the summarized prediction error for the validation set is the average of the prediction errors calculated at all timesteps for all activities.

$$PE = \frac{1}{|activities|} \sum_{a \in activities} \frac{1}{|timesteps|} \sum_{t \in timesteps_a} \sqrt[2]{\left(v(t)_{predicted} - v(t)_{input}\right)^2}; \; v \in R^{30} \quad (1)$$

Parcellated prediction error

Because we performed dimensionality reduction separately on body motion features, object semantics, object appearance or disappearance, and optical features, we could calculate

predictions and prediction errors specifically for each of those. The resulting set of features contained 30 dimensions (14 body motion dimensions, 13 semantic dimensions, 2 dimensions for object appearances and disappearances, and 1 dimension for the correlation of pixel luminance between successive video frames). Prediction error for a specific modality at each timestep is the Euclidean distance between the model's prediction for that modality and the input scene vector for that modality. For example, prediction error at each timestep for body motion features is the Euclidean distance between a 14-dimensional prediction vector, which is a subset of the model's prediction, and a 14-dimensional input scene vector, which is a subset of the input scene vector.

Segmentation

An event boundary for SEM occurs when it switches from one event schema to another event schema, or when it restarts the current event schema. The generic model only has one event schema, so this definition of event boundaries is not applicable. To simulate event boundaries for the generic model, we identified peaks in prediction error. Peaks were local maxima selected in descending order of height. The number of peaks was matched to SEM-2.0's number of event boundaries for that validation activity. To have a fairer comparison between the generic model and SEM-2.0, we applied the same algorithm to SEM-2.0's prediction errors to derive event boundaries. The generic model's boundaries, SEM-2.0's actual boundaries, and SEM-2.0's derived boundaries were compared against human boundaries to calculate point-biserial correlations.

Mutual Information

To quantify SEM-2.0's categorization agreement with human categorization, and how SEM-2.0's event schemas generalize to actions performed by the same actor in different instances and by different actors at different locations, we computed the mutual information score (with adjustment to account for chance[42]) between the model event labels and script action labels every time the model completes watching a training activity. We treated script action labels as one clustering of input scene vectors, and model's event labels as another clustering of input scene vectors. Formally, given a stimulus set, *Corpus*, with N input scene vectors (the total number of input scene vectors across all validation activities), and two partitions of Corpus, namely *Schemas* (SEM-2.0's categorization) and Actions (humans' categorization):

$$Corpus = \{v_1, v_2, \ldots, v_n\};\ v_i \in R^{30} \quad (2)$$

$$Schemas = \{schema_1, schema_2, \ldots, schema_n\};\ schema_i = \{v_i, v_j, \ldots, v_k\} \quad (3)$$

$$Actions = \{action_1, action_2, \ldots, action_m\};\ action_i = \{v_{i'}, v_{j'}, \ldots, v_{k'}\} \quad (4)$$

The mutual information score between Schemas and Actions is computed as:

$$I(Schemas;\ Actions) = \sum_{y \in Schemas} \sum_{x \in Actions} P(schema,\ action) \log\left(\frac{P(schema,\ action)}{P(schema)P(action)}\right) \quad (5)$$

Concretely, equation 5 is a summation over all schema-action pairs. An example of a schema-action pair is shown in Fig. 8, and the calculation for that pair is below:

$$P(Event\ 4;\ Jumping\ Jacks)\log\left(\frac{P(Event\ 4;\ Jumping\ Jacks)}{P(Event\ 4)P(Jumping\ Jacks)}\right) \quad (6)$$

Equation 6 was repeated and summed for all pairs of SEM-2.0 event schemas and scripted action labels to derive mutual information score (Eq. 5).

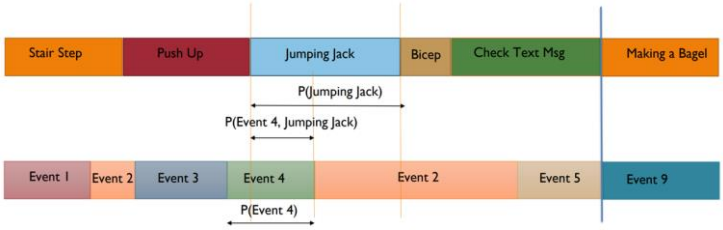


Fig. 8: Conceptual example of the calculation of mutual information. Top: a sequence of human actions. Orange vertical line indicates the start of one action and the end of another action. Blue vertical line indicates the transition between two activities (since all validation activities were concatenated to compute mutual information score). P(Jumping Jack) and P(Event 4) are marginal probabilities of jumping jack scenes and scenes that were assigned to SEM's event 4. P(Event 4, Jumping Jack) is the joint probability of scenes assigned to event 4 and jumping jack. In this illustration, the result of equation 6 will be high since there is a high correspondence between event 4 and jumping jack.

For each pair of $Schema_i$ (e.g. SEM-2.0 event schema "4," meaning the 4th schema created by SEM-2.0 during training) and $Action_j$ (e.g. script action label "jumping jack"), P(event 4, jumping jack) is the probability that an input scene vector belongs to both clusters event 4 and jumping jack. If cluster event 4 corresponds with cluster jumping jack (a large number of timesteps are labeled as both SEM-2.0 event schema 4 and script action label jumping jack), the ratio within the logarithmic function will be high and the term for this pair will also be high. The mutual information score can also inform us about SEM-2.0's ability to generalize across actors and activities. A given action (e.g. jumping jack) can be performed by different actors in different environments. If SEM-2.0 is able to generalize across actors and environments, it should assign the same event label (e.g. event 4) to those input scene vectors. In that case, the adjusted mutual information score will be high. In contrast, if SEM-2.0 assigns different event labels to the same action performed by a different actor in a different room (event 4 to actor A jumping jack and event 20 to actor B jumping jack), the score will be low. To the degree that SEM-2.0 categorizes input scene vectors in a human-like way, and its event schemas generalize to instances of the same action, adjusted mutual information between two partitioning algorithms will be high.

Permutation testing

We assessed how likely the results would occur by chance via permutation testing. SEM-2.0's event labels for all validation activities were first concatenated. A permutation is generated by

shuffling runs of event labels, thus preserving event lengths in the resulted permutation. The shuffling not only changes SEM-2.0's event labels for particular input scene vectors but also changes SEM-2.0's event boundaries. As a result, the shuffling procedure can be used for both segmentation and categorization tests. When we concatenate two validation activities, there is an interval between the onset of SEM-2.0's last event in the first activity and the onset of SEM-2.0's first event in the second activity. Because human event boundaries are less likely to fall into these intervals, and SEM-2.0 never placed boundaries in these intervals, we made sure permutations did not have boundaries within these intervals so that SEM-2.0 would not have an unfair advantage over permutations. Permutations were repeated 100 times and scaled point-biserial correlation, adjusted mutual information, purity, and coverage were computed for each permutation.

Training and testing input-deprived models

To test the ability of the model to learn representations with limited input, we trained two versions of SEM-2.0, each version with eight random initializations, that generated predictions of the full scene vector from deprived input features: we withheld either the body motion features or the semantic features. The input scene vector is a concatenation of multiple PCA-ed vectors: a 14-dimensional vector for body motion features, a 13-dimensional vector for semantic feature, a 2-dimensional vector for object appearance/disappearance feature, a 1-dimensional vector for optical flow feature. For the semantics-deprived model, we set the 13-dimensional vector for semantic features to zeros before feeding into the model. For the motion-deprived model, we set the 14-dimensional vector for body motion features to zeros before feeding into the model. The two models still had to make predictions for all features: 30-dimensional output scene vectors. We evaluated deprived models on the aforementioned metrics: prediction error, scaled point-biserial correlation, and adjusted mutual information; we used permutation testing to assess the statistical significance of segmentation and categorization metrics.

Data Availability

The META stimulus set is publicly available through its Open Science Foundation repository: https://osf.io/3embr/

Code Availability

All processing scripts required to run analyses are available at: https://github.com/mbezdek/extended-event-modeling.

Acknowledgements

This research was supported by a grant from the Office of Naval Research (N00014-17-1-2961). We would like to thank Vi Nguyen and Matt Steinhaus for providing annotations, and Maverick Smith for helpful comments.

Author Contributions

Competing Interests

The authors declare no competing interests.

References

1. Clark, A. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behav. Brain Sci.* **36**, 181–204 (2013).

2. Graesser, A. C. & Nakamura, G. V. The Impact of a Schema on Comprehension and Memory. in *Psychology of Learning and Motivation* vol. 16 59–109 (Elsevier, 1982).

3. Anderson, R. C. Schema-Directed Processes in Language Comprehension. in *Cognitive Psychology and Instruction* (eds. Lesgold, A. M., Pellegrino, J. W., Fokkema, S. D. & Glaser, R.) 67–82 (Springer US, 1978). doi:10.1007/978-1-4684-2535-2_8.

4. Bartlett, F. C. *Remembering: A study in experimental and social psychology*. (Cambridge university press, 1932).

5. Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S. & Reynolds, J. R. Event perception: A mind/brain perspective. *Psychol. Bull.* **133**, 273–293 (2007).

6. Zwaan, R. A., Langston, M. C. & Graesser, A. C. The construction of situation models in narrative comprehension: An event-indexing model. *Psychol. Sci.* **6**, 292–297 (1995).

7. Schapiro, A. C., Rogers, T. T., Cordova, N. I., Turk-Browne, N. B. & Botvinick, M. M. Neural representations of events arise from temporal community structure. *Nat. Neurosci.* **16**, 486–492 (2013).

8. Elman, J. L. Finding structure in time. *Cogn. Sci.* **14**, 179–211 (1990).

9. Smith, K. *et al.* Modeling expectation violation in intuitive physics with coarse probabilistic object representations. *Adv. Neural Inf. Process. Syst.* **32**, (2019).

10. Rao, R. P. N. An optimal estimation approach to visual perception and learning. *Vision Res.* **39**, 1963–1989 (1999).

11. Reynolds, J. R., Zacks, J. M. & Braver, T. S. A computational model of event segmentation from perceptual prediction. *Cogn. Sci.* **31**, 613–643 (2007).

12. Elman, J. L. & McRae, K. A model of event knowledge. *Psychol. Rev.* **126**, 252–291 (2019).

13. Franklin, N. T., Norman, K. A., Ranganath, C., Zacks, J. M. & Gershman, S. J. Structured event memory: A neuro-symbolic model of event cognition. *Psychol. Rev.* **127**, 327–361 (2020).

14. Pennington, J., Socher, R. & Manning, C. Glove: Global Vectors for Word Representation. in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 1532–1543 (Association for Computational Linguistics, 2014). doi:10.3115/v1/D14-1162.

15. Bezdek, M. *et al. The Multi-angle Extended Three-dimensional Activities (META) stimulus set: A tool for studying event cognition*. https://osf.io/r5tju (2022) doi:10.31234/osf.io/r5tju.

16. Richmond, L. L. & Zacks, J. M. Constructing Experience: Event Models from Perception to Action. *Trends Cogn. Sci.* **21**, 962–980 (2017).

17. Parker, Robert, Graff, David, Kong, Junbo, Chen, Ke & Maeda, Kazuaki. English Gigaword Fifth Edition. 9542041 KB (2011) doi:10.35111/WK4F-QT80.

18. Fox, E. B., Sudderth, E. B., Jordan, M. I. & Willsky, A. S. A sticky HDP-HMM with application to speaker diarization. *Ann. Appl. Stat.* **5**, (2011).

19. Wallach, H., Jensen, S., Dicker, L. & Heller, K. An Alternative Prior Process for Nonparametric Bayesian Clustering. in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* 892–899 (JMLR Workshop and Conference Proceedings, 2010).

20. Welling, M. Flexible Priors for Infinite Mixture Models. 8.

21. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).

22. Kurby, C. A. & Zacks, J. M. Starting from scratch and building brick by brick in comprehension. *Mem. Cognit.* **40**, 812–826 (2012).

23. Hommel, B., Müsseler, J., Aschersleben, G. & Prinz, W. The Theory of Event Coding (TEC): A framework for perception and action planning. *Behav. Brain Sci.* **24**, 849–878 (2001).

24. Szpunar, K. K., Chan, J. C. K. & McDermott, K. B. Contextual Processing in Episodic Future Thought. *Cereb. Cortex* **19**, 1539–1548 (2009).

25. Kurby, C. A. & Zacks, J. M. Segmentation in the perception and memory of events. *Trends Cogn. Sci.* **12**, 72–79 (2008).

26. Goodale, M. A., Króliczak, G. & Westwood, D. A. Dual routes to action: contributions of the dorsal and ventral streams to adaptive behavior. in *Progress in Brain Research* vol. 149 269–283 (Elsevier, 2005).

27. Hochreiter, S. & Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **9**, 1735–1780 (1997).

28. Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A. & Bengio, Y. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. (2013) doi:10.48550/ARXIV.1312.6211.

29. McCloskey, M. & Cohen, N. J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. in *Psychology of Learning and Motivation* vol. 24 109–165 (Elsevier, 1989).

30. McClelland, J. L., McNaughton, B. L. & Lampinen, A. K. Integration of new information in memory: new insights from a complementary learning systems perspective. *Philos. Trans. R. Soc. B Biol. Sci.* **375**, 20190637 (2020).

31. McClelland, J. L., McNaughton, B. L. & O'Reilly, R. C. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychol. Rev.* **102**, 419–457 (1995).

32. Elman, J. L. & McRae, K. A model of event knowledge. *Psychol. Rev.* **126**, 252–291 (2019).

33. Dickman, H. R. The Perception of Behavioral Units. in *The stream of behavior: Explorations of its structure & content.* (ed. Barker, R. G.) 23–41 (Appleton-Century-Crofts, 1963). doi:10.1037/11177-002.

34. Hard, B. M., Tversky, B. & Lang, D. S. Making sense of abstract events: Building event schemas. *Mem. Cognit.* **34**, 1221–1235 (2006).

35. Zacks, J. M., Tversky, B. & Iyer, G. Perceiving, remembering, and communicating structure in events. *J. Exp. Psychol. Gen.* **130**, 29–58 (2001).

36. Hasson, U., Yang, E., Vallines, I., Heeger, D. J. & Rubin, N. A Hierarchy of Temporal Receptive Windows in Human Cortex. *J. Neurosci.* **28**, 2539–2550 (2008).

37. Baldassano, C. *et al.* Discovering Event Structure in Continuous Narrative Perception and Memory. *Neuron* **95**, 709-721.e5 (2017).

38. Minsky, M. A framework for representing knowledge. in *The Psychology of Computer Vision* (ed. Winston, P.) (McGraw Hill).

39. Wang, L. & Dunson, D. B. Fast Bayesian Inference in Dirichlet Process Mixture Models. *J. Comput. Graph. Stat.* **20**, 196–216 (2011).

40. Corti, A., Giancola, S., Mainetti, G. & Sala, R. A metrological characterization of the Kinect V2 time-of-flight camera. *Robot. Auton. Syst.* **75**, 584–594 (2016).

41. Li, B., Yan, J., Wu, W., Zhu, Z. & Hu, X. High Performance Visual Tracking with Siamese Region Proposal Network. in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* 8971–8980 (IEEE, 2018). doi:10.1109/CVPR.2018.00935.

42. Vinh, N. X., Epps, J. & Bailey, J. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. 18.

Supplementary Information

Visualization of input features and SEM's prediction

We rendered input features, SEM-2.0's prediction and segmentation for SEM versions with full input features, without semantic features, and without motion features under directory "output_activities" stored in OSF: https://osf.io/39qwz/?view_only=d186e6c784fa47f4a2da01ffcf08d850. Description of these videos can be found in the same directory.

Structure of the META Stimulus Set

The META stimulus set is comprised of over 25 hours of humans performing activities in naturalistic settings. Each activity was one of four different types of activities (making breakfast, exercising, cleaning a room, or bathroom grooming) and was performed by one of five different actors in a unique realistic room for each activity type, for a total of 15 environments. An algorithm generated scripts of six to seven actions for each activity, five to six actions sampled from a set of nine to 13 actions of a specific activity type and one action sampled from a set of six actions that could appear in any activity type (see Fig. 4A for the sequences of actions for one activity type). The algorithm randomly selected actions from sets of sub-activity actions, with rules for excluding generated scripts with unrealistic combinations or orders of specific actions (for example, applying lotion before washing or shaving one's face). The process was repeated so that per actor per activity type, there were ten activity scripts with unique sequences of actions to perform. In total there were 150 activities. Actors memorized and performed the actions for each activity consecutively with no breaks in the recording, and there was no dialogue or other actor present in any of the activities. Each performance was filmed with three synchronous color cameras and a time-of-flight depth sensor. For the set of activities used in this study there was an average duration of 10 minutes 22 seconds per activity. For more detailed information on the META stimulus set, please see the cited paper[15].

Modifications from original SEM implementation

Inspecting the performance of the original SEM implementation (SEM-1.0)[13] on the META stimulus set[15] revealed that SEM spawned a lot of event schemas but used only a couple of them to assign to input scene vectors. There were three reasons that led SEM-1.0 to this behavior: 1) The sCRP preferred highly frequent events, which could lead to a self-perpetuating circle, 2) new event schemas were initialized to random weights without any knowledge of the dataset, and 3) inactive RNNs were at a disadvantage because they had to make predictions from random input scene vectors.

To evaluate how evenly SEM uses its event schemas, we computed entropy for the distribution of schema activations on either training activities or validation activities. Because the range of entropy depends on the number of elements (or the number of schemas), comparing entropy between two SEM versions with different number of event schemas could be misleading.

Entropy was normalized based on the maximum possible entropy (uniform distribution) for each particular number of schemas.

In the sCRP, there is a term in the prior over schemas that biases SEM towards reusing event schemas that were activated frequently in the past, facilitating generalization across events. However, this rich-get-richer property led SEM to create a small number of event schemas; the more these schemas were used, the more likely they would be used again since they were highly frequent events. Using the sticky uniform process (sUP) steered SEM towards activating a larger set of event schemas, and activations were distributed more evenly for both training and validation activities (SEM-1.0 versus SEM-1.2-add_generic in Fig. S2).

SEM-1.0 initialized random weights to newly spawned event schemas, reducing the predictive power of these event schemas and consequently making these schemas less likely to be activated. Suppose SEM has one previously learned event schema for making a toast and one newly spawned event schema for making apple juice. Now, suppose the input scene vector is one where the actor is making tea. Because the previously learned event schema has general knowledge about what objects should be around, and general forms and movements of actors, even if its prediction is not perfect, it'll still be closer to the input scene vector than the prediction from newly spawned event schema (an RNN with random weights). Thus, previously learned event schemas are more likely to be activated than newly spawned event schemas due to random weight initializations. In this version of SEM, a generic model was trained alongside with SEM and new event schemas were initialized to the generic model's weights. The addition of the generic model and the removal of advantage for highly frequent event schemas helped SEM use a much larger set of event schemas, and activate them much more evenly for both training and validation activities (SEM-1.0 versus SEM-1.3-generic_sUP in Fig. S2). Fig. S3 shows examples of distribution of schema activations for different versions of SEM.

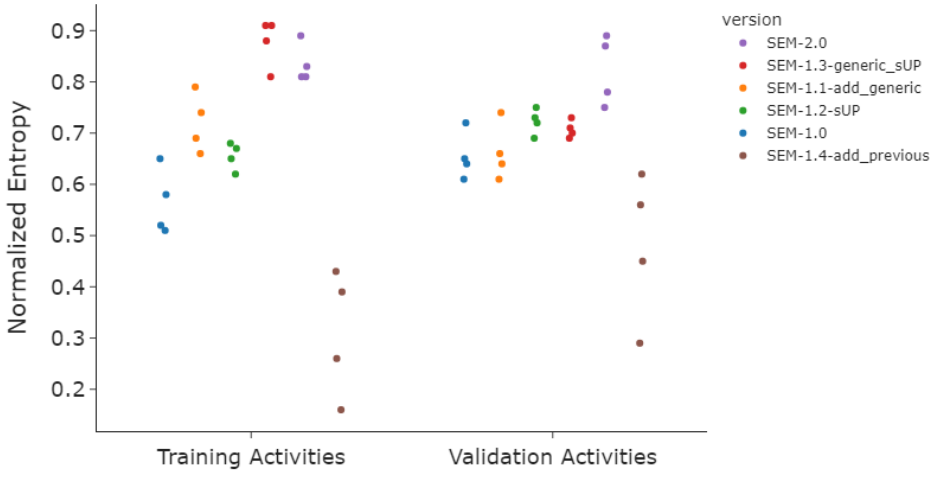


Fig. S2. Effects of modifications made in SEM-2.0 on the entropy of the distribution of event schema activations. Each SEM version was simulated with four different random initializations. SEM-1.0 is original SEM. SEM-1.1-add_generic is the original SEM with one modification: initializing new RNNs to the generic model's weights. SEM-1.2-sUP is the original SEM with one modification: using sUP instead of sCRP. SEM-1.3-generic-sUP is the original SEM with the two modifications above. SEM-1.4-add_previous is the original with one modification: feeding the previous scene vectors to all RNNs instead of only active RNNs. SEM-2.0 is the version with three modifications. Initializing new RNNs to the generic model's weights alone improved schema activations for training activities, and using the sUP alone improved schema activations for both training and validation activities. In contrast, feeding previous scene vectors alone hurts schema activations. The combination of three modifications improved schema activations by a large degree on both training and validation activities.

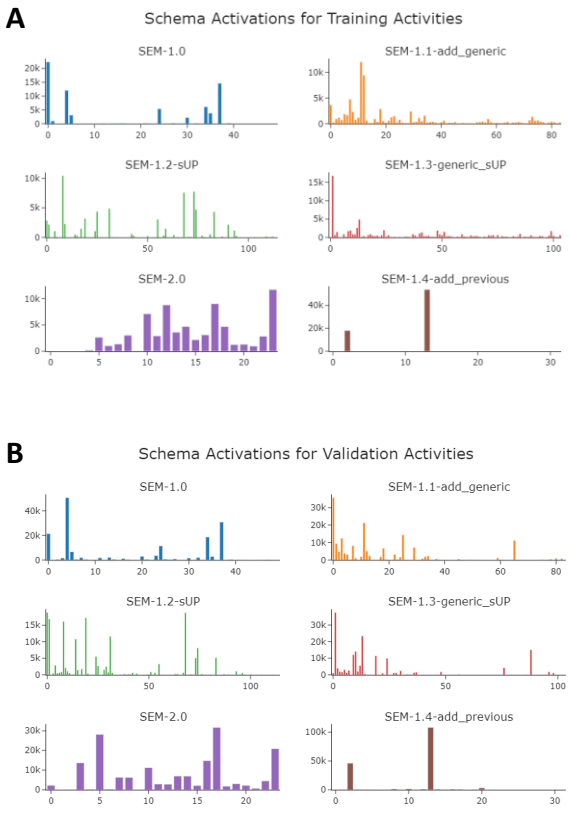


Fig. S3. Examples of schema activations for different versions of SEM for training (A) and validation (B) activities. All versions were simulated with the same random initialization. X-axis indicates event schema, and y-axis indicates schema activation. For this particular

random initialization, adding the generic model or using sUP both made SEM activate schemas more evenly. In contrast, feeding previous scene vectors alone made SEM use some select schemas. The combination of three modifications made SEM create a smaller number of schemas and activate schemas much more evenly for both training and validation activities.

At each timestep, SEM computes the likelihood for each RNN given the input scene vector. In principle, each RNN should be fed scene vectors from previous timesteps and asked to make a prediction for the current scene vector, and the predicted scene vector would be compared to the input scene vector to compute the likelihood. However, this approach is computationally expensive as the number of event schemas in SEM grows. In SEM-1.0[13], the authors circumvented this challenge by feeding random vectors to RNNs instead of scene vectors from previous timesteps and asking RNNs to make predictions from these random vectors. These predictions were cached and used to compute likelihoods for RNNs that were not active at the previous timestep. That approach reduced computations drastically and made the training much faster. However, this approach placed inactive RNNs at a disadvantage because the input scene vectors to these RNNs were not informative to predict the current scene vector. Consequently, inactive RNNs were less likely to be selected and less likely to update their weights (as only the active RNN updates its weights at a specific timestep). As a result, some RNNs that are activated initially snowball and dominate other RNNs, getting activated most of the time. Furthermore, because these RNNs must be able to predict the next scene vector from either previous scene vectors or random vectors, their predictive power is compromised compared to the situation where they only need to predict the next scene vector from previous scene vectors. In SEM-2.0, in addition to the aforementioned changes, we fed previous scene vectors to all RNNs and parallelized RNNs' predictions to alleviate the computational challenge. SEM-2.0 created a smaller number of RNNs to capture the data and more importantly activated RNNs more evenly for both training and validation activities (SEM-1.0 versus SEM-2.0 in Fig. S2). In addition, prediction error was reduced (SEM-1.0 versus SEM-1.4-add_previous and SEM-2.0 in Fig. S4).

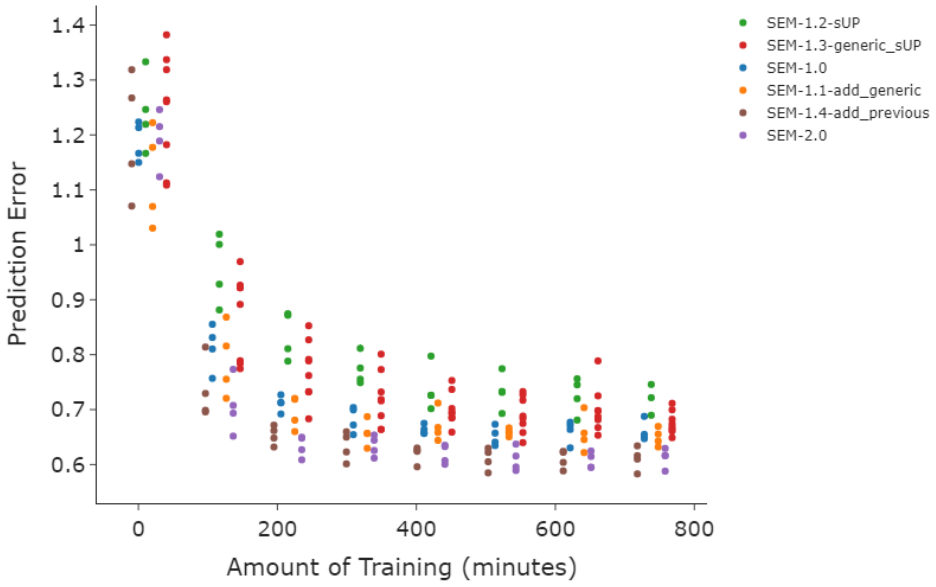


Fig. S4. The effect of feeding previous scene vectors to RNNs on prediction error. Using the sUP (SEM-1.2-sUP) increased prediction error while adding the generic model (SEM-1.1-add_generic) decreased prediction error a little bit. Adding previous scene vectors to all RNNs reduced prediction error (SEM-1.4-add_previous and SEM-2.0).

Generalization across actors and environments

We qualitatively evaluated how SEM-2.0's event schemas generalize across actors in this dataset. For many of the ground-truth action labels, there were multiple action instances performed by the same actor in a different activity instance or performed by a different actor in a different environment. Thus, we were able to test whether across instances, actors, and environments, SEM-2.0 would assign the same event label to the same script action label. Fig. S5A shows a confusion matrix of the proportion of timepoints from each scripted action label instance in the validation set assigned to each SEM-2.0 schema for one trained instantiation of SEM-2.0. Several of SEM-2.0's event schemas generalize across actors. For example, in this instantiation of SEM-2.0, schema 18 is the most-frequently activated schema for the scripted action 'wash face' across instances, whereas schema 14 is the most frequently activated schema for 'perform bicep curls' across multiple actors. In general, SEM-2.0 generates separate sets of schemas for separate activity types and often over-generalizes to the activity type level.

Fig. S5B shows the correspondence between schemas and action labels after shuffling the assignments of schemas to timepoints.

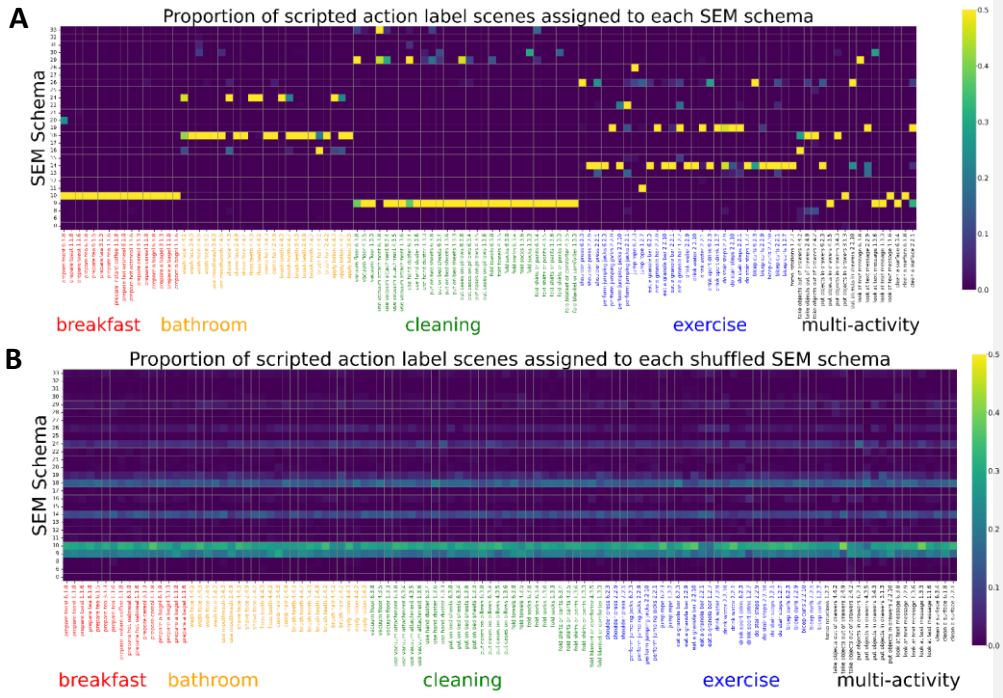


Fig. S5. Confusion matrix of the proportion of timepoints from each scripted action label instance in the validation set assigned to each SEM-2.0 schema, for one initialization of SEM-2.0 (top) and the confusion matrix after randomly shuffling the assignment of SEM-2.0 schemas to timepoints (bottom). Scripted action labels are ordered by the type of activity they appeared in, with breakfast activity actions in red, bathroom activity actions in orange, cleaning activity actions in green, exercise activity actions in blue, and multi-activity actions in black. SEM-2.0 schemas are arranged in the order that they were spawned. Timepoints without a scripted action label and scripted action labels with just a single instance in the validation set are omitted. Several schemas generalize across instances, actors, and environments and some schemas over-generalize to entire activity types (e.g., schema 10 is recruited for almost all breakfast actions). After shuffling the SEM-2.0 schemas, there are visible differences in the relative frequency of schemas but no clustering according to action or activity type.

Purity and coverage comparing event schemas and judges' action categories

To quantify SEM's agreement with humans in terms of clustering, we calculated purity and coverage for all validation activities. Across validation activities, we concatenated SEM's event labels to form a single vector of event labels, we also concatenated script action labels to form a single vector of action labels. Purity and coverage were computed based on the two label vectors. Purity is the degree to which a SEM event schema belongs to a single script action label. The formula for purity, with |x| denoting the total length of x, is:

$$Purity = \sum_{schema \in Schemas} \frac{\max_{action \in Actions} |schema \cap action|}{|schema|} \quad (S1)$$

Coverage is the degree to which a script action label is covered by a single SEM event schema, and the formula for coverage is:

$$Coverage = \sum_{action \in Actions} \frac{\max_{schema \in Schemas} |action \cap schema|}{|action|} \quad (S2)$$

Purity and coverage range from 0 to 1 (1 is perfect purity or coverage). A schematic illustration of purity and coverage is shown in Fig. S6.

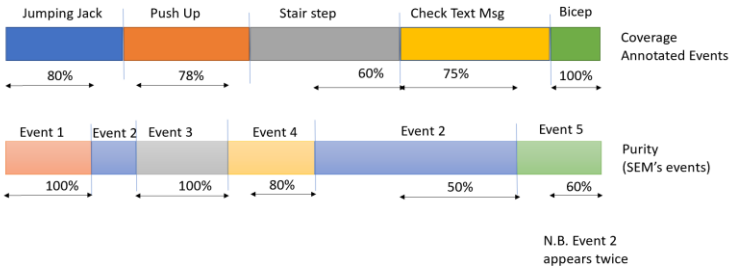


Fig. S6. A schematic illustration of purity and coverage for SEM's event schemas and script actions. Percentages under action labels are their coverage, how much of that action is covered by a single SEM's event schema. Percentages under SEM's event schemas are their purity, the largest portion of the event schema matched with a single action label.

If SEM assigns unique event labels to each input scene vector, its purity will be 1, but its coverage will be close to 0. If SEM assigns a single event label to all input scene vectors, its coverage will be 1, but its purity will suffer a lot (close to 0). Thus, purity and coverage represent a tradeoff, and if SEM's classification of timesteps resembles script action labels, SEM will perform well on both purity and coverage. We assessed how likely the results would occur by

chance by permutation testing: for each activity, we shuffled SEM's events 100 times while preserving event length and calculated purity and average to derive null distributions. Fig S7 below shows permutation tests for all validation activities across training.

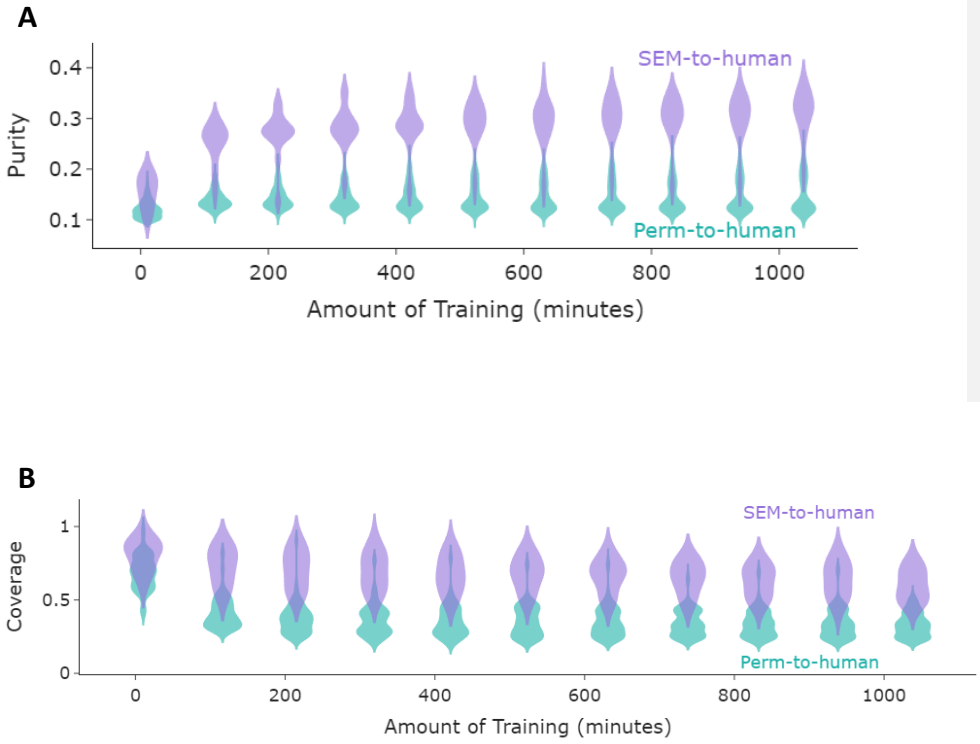


Fig. S7. Purity (A) and coverage (B) for multiple simulations of SEM, compared against permutations across training. Permutations are performed such that event lengths are preserved.

As shown in Fig. S7, SEM's coverage decreases slightly across training, and SEM's purity increases across training. Early in training, SEM has a relatively small number of generalist event schemas to accommodate the stimuli. Later in training, SEM's event schemas become more specialized and correspond more to script action labels (see Fig. 4A).

Hyper-parameter search

To select appropriate hyperparameters for SEM, we performed a grid search with stickiness (1e-1, 1e0), concentration (1e4, 1e5), and learning rate (1e-2, 1e-3, 1e-4). For each combination of the three hyperparameters, six simulations with different random seeds were run. Those simulations from 12 sets (2x2x3) of hyperparameters were pit against each other to see which set of hyperparameters reduce prediction error the most. To evaluate the influence of each hyperparameter on prediction error, for each hyperparameter, simulations with the same value for that hyperparameter were collapsed and compared against simulations with a different value for that hyperparameter. Adjusting the learning rate had a large effect on prediction error, whereas adjusting stickiness and concentration did not noticeably affect prediction error (Fig. S6). After selecting the appropriate learning rate, stickiness and concentration were then adjusted so that the resulting SEM simulations would have a similar number of boundaries to that of human fine-grain segmentation, whose median number of boundaries per validation activity was 22 (Fig. S8); this was done to facilitate comparison between SEM's segmentation and human segmentation. Note that because SEM showed flurries of boundaries whereas human segmenters did not, comparing the raw number of boundaries between SEM and humans could be misleading. To factor in these flurries, we performed the following adjustment to SEM's event boundaries before counting them: if two boundaries were closer than two seconds, only one boundary was retained. Stickiness was searched in (1e-1, 1e0, 1e1) and concentration was searched in (1e5, 1e6, 1e7). The nine combinations resulted in SEM simulations with similar prediction errors. We picked the configuration with stickiness of 1e7 and concentration of 1e-1 since its average adjusted number of boundaries for validation activities was 23, closest to human's median number of boundaries (fig S9).

SEM-2.0 has a number of other hyperparameters that are not of theoretical interest. We set hyperparameters that control SEM's sensitivity to error according to stimulus statistics, as suggested in SEM-1.0[13]; see table S1. We retained hyperparameters for Adam optimization algorithm; see Table S2.

Noise variance prior (ν)	Degree of freedom for noise variance prior (τ)
0.06	10

Table S1. Hyper-parameters that control SEM's sensitivity to error, noise variance prior and degree of freedom.

β_1	β_2	ε	Decay	n_epochs
0.9	0.999	1e-08	0	10

Table S2. Hyper-parameter values for the Adam optimization algorithm used to train the recurrent neural network. All simulations used the same optimization hyper-parameters.

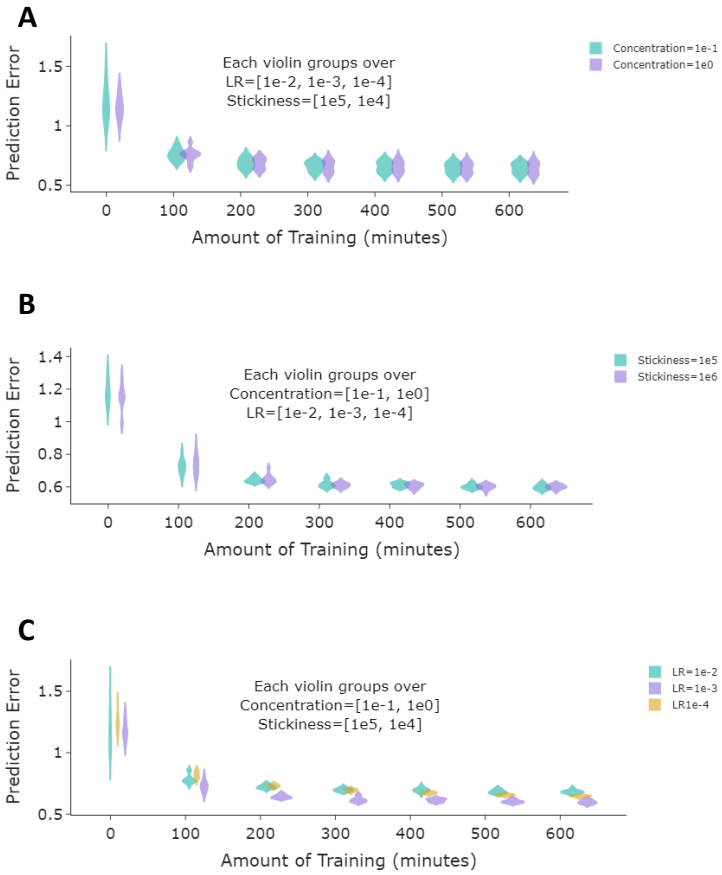


Fig. S8. **The relative influence of hyperparameters on prediction error.** A violin plot is shown for all simulations at each level of the given hyper-parameter (grouping all levels of the other two hyper-parameters). Changing stickiness or concentration didn't affect prediction error much (A and B panels). However, changing learning rate affected prediction error significantly (C panel), and simulations with learning rate of 1e-3 performed best.

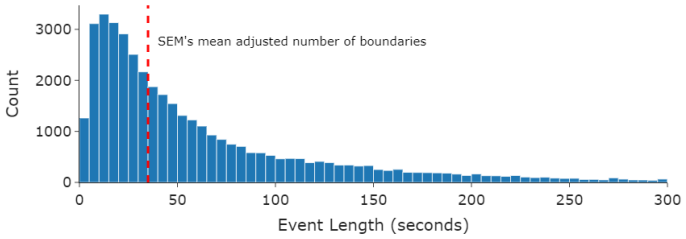


Fig. S9. Histogram of the number of boundaries for each validation activity from human fine-grain segmentation. Red line is the mean of SEM's adjusted number of boundaries for each validation activity.

Models deprived of some input features produce higher prediction errors

Scene vectors consisted of features related to biological motion and to the semantic meaning of interactive objects in the environment. To test the ability of SEM-2.0 to learn predictive representations with limited input, we trained versions of SEM-2.0 that generated predictions of the full scene vector but withheld either the body movement features (*motion-deprived model*) or the semantic features (*semantics-deprived model*) from the input. For both the motion-deprived model and the semantics-deprived model, with a subset of input features, SEM-2.0 was able to learn to predict scene vectors and to reduce prediction error over time, though prediction error remained higher than in the model that received the full set of input features (Fig. S10). As expected, a large contribution to prediction error in both deprived models was from the deprived features.

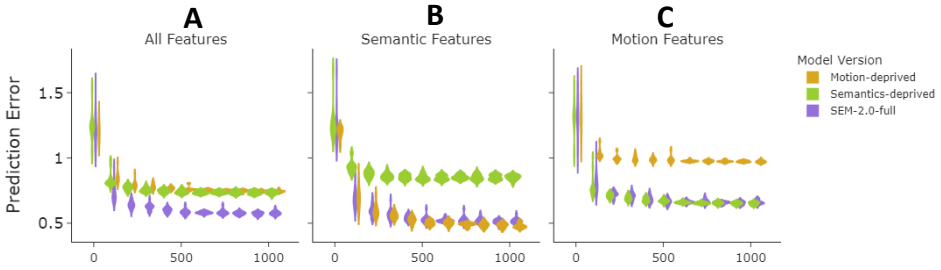


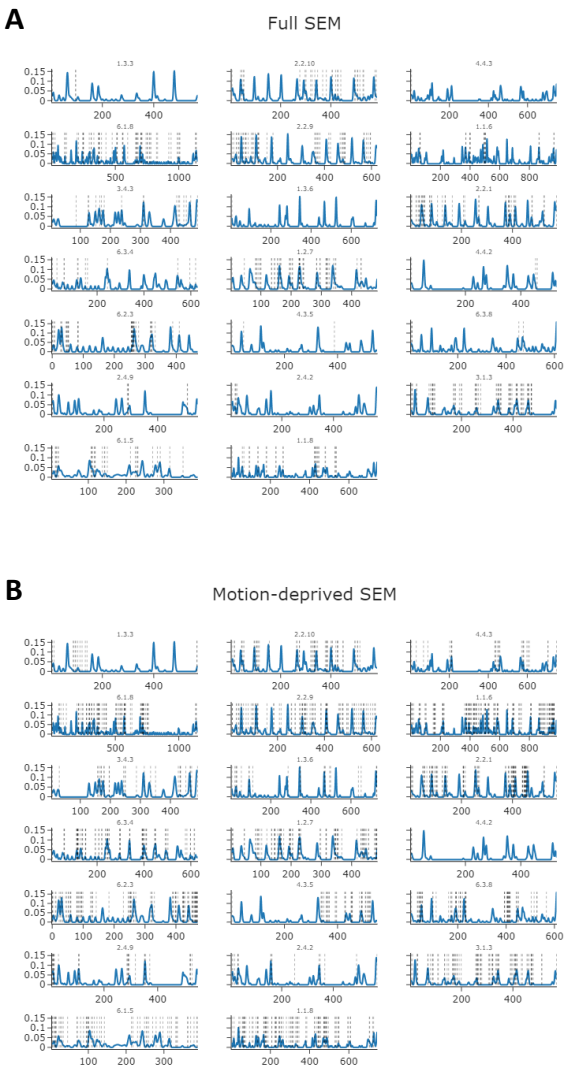
Fig. S10: **Total and parcellated prediction error for three versions of SEM-2.0 (full SEM-2.0, semantics-deprived SEM-2.0, motion-deprived SEM-2.0).** For each SEM-2.0 version, there were multiple simulations with different weight initializations and training orders. Each violin plot is a distribution of prediction errors for these simulations. A-C (left to right): full prediction error, prediction error only for motion features, and prediction error only for semantic features. Training SEM-2.0 without body motion or semantic features results in an increase in total prediction error (left). Training SEM-2.0 without body motion features or semantic features

results in higher prediction error for motion features or semantic features (middle and right), respectively, and prediction errors for these features decrease over time.

Segmentation agreement is similar for SEM-2.0 and deprived models

We also compared segmentation agreement between models and humans for the motion-deprived model and the semantics-deprived model. We again used a permutation test to generate a null distribution for each type of model, and combined these null distributions into a single null distribution. Examples of SEM-2.0 segmentation and deprived models' segmentation are in Fig. S11: SEM-2.0 with full input features and deprived models generated event boundaries that align with human segmentation. Comparing these models using the scaled biserial correlation, the score was similar for all models (Fig. S12). This result suggests that prediction errors from either semantic information or motion information is sufficient to elicit

event boundaries that achieve the degree of agreement with event boundaries perceived by humans as full SEM-2.0 did.



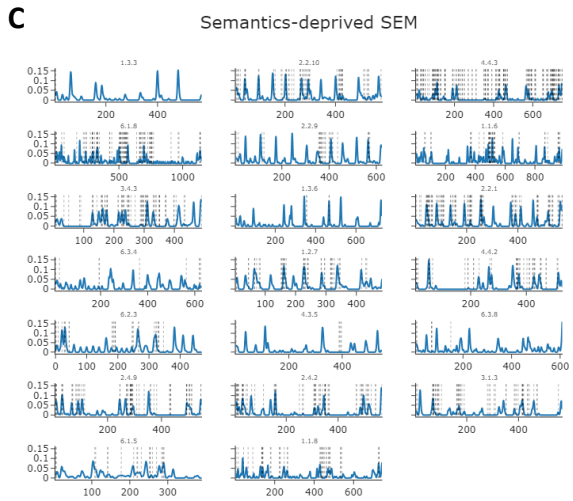


Fig. S11: Comparison of event boundaries for all validation activities between SEM-2.0 and the deprived models. A-C: event boundaries from SEM-2.0, motion-deprived model, and semantic-deprived model. All models generated event boundaries (dashed vertical lines) that align with human segmentation (blue line).

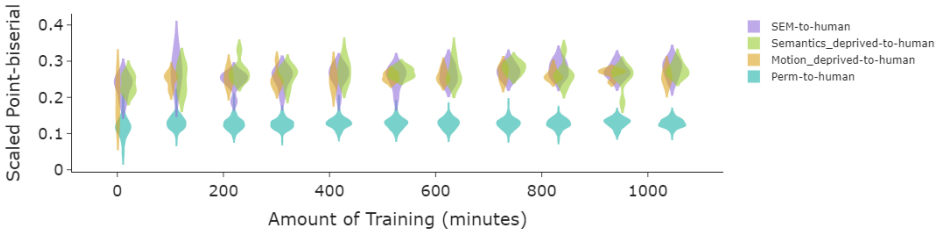


Fig. S12: Comparison of agreement with human segmentation between SEM-2.0 and deprived models. SEM-to-human denotes boundaries from SEM-2.0 with full input features, Semantics_deprived-to-human denotes boundaries from semantics-deprived model, and Motion_deprived-to-human denotes boundaries from motion-deprived model. Perm-to-human denotes permutated boundaries from all models. Each violin plot is a distribution of scaled point-biserial correlation for different initializations and training orders. Segmentation agreement between deprived models and humans exceeds chance performance, and it is comparable to full SEM-2.0's segmentation agreement with humans.

Deprived model schemas show correspondence with human action categories

To test if the schemas formed by deprived models align with human action categories, we computed adjusted mutual information between human action labels and the motion-deprived and semantics-deprived models. Adjusted mutual information scores for the motion-deprived model and the semantics-deprived models were a bit lower than the score for SEM with full input features, though all models performed better than chance (Fig. S13). Thus, whether the model has access to only body motion information, only object information, or both, it learns human-like event categories.

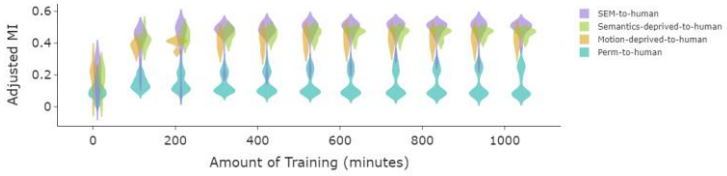


Fig. S13: Categorization agreement with humans for SEM-2.0 and deprived models. Each violin plot is a distribution of adjusted mutual information scores for different initializations and training orders. Categorization agreement between deprived models and humans is greater than expected by chance, and it is a bit lower compared to full SEM-2.0's categorization agreement with humans.