Learning Actions from Events Using Agent Motions

Nikhil Krishnaswamy, Tuan Do, and James Pustejovsky
Brandeis University
Waltham, MA, USA
{nkrishna, tuandn, jamesp}@brandeis.edu

Abstract

In this paper, we present results from a deep neural network event classifier that uses lexical semantic features derived from parameters that are underspecified in the event typing. These results demonstrate that the presence or absence of an underspecified feature is a strong predictor of event class, and we propose a model for extending this approach to *action recognition* (i.e., the recognition of processes enacted by an agent) by using reinforcement learning to learn complex actions from object motions, and then "factoring out" the specifics of the object to recognize an action denoted by an agent motion, such as a gesture, alone.

Keywords: actions, events, semantics, multimodal, language, gesture, recognition, classification.

1. Introduction

Work in event *visualization* from natural language description (e.g., (Coyne and Sproat, 2001; Siskind, 2001; Chang et al., 2015) among others) often struggles with the problem of underspecified parameters in events enacted over arbitrary objects. These parameters may be inherent to the event itself (e.g., speed, direction, etc.), or properties of the object argument(s) (e.g., axis of rotation, geometrical concavity, etc.). Should a computational visualization system use an inappropriate value for one of these parameters, it may generate a visualization for a given event that does not comport with a human viewer's understanding of what that event is, such as rotating a cylindrical object about its non-major axis for a "roll." Previously we explored these issues and solutions to them in (Krishnaswamy and Pustejovsky, 2016a; Krishnaswamy and Pustejovsky, 2016b; Krishnaswamy, 2017).

Event *recognition* from the perspective of visual data processing or object tracking (cf. (Yang et al., 2013)) provides a venue to explore "learning from observation," and as a domain has achieved recent relevance in human communication with robotic agents (Yang et al., 2015b; Paul et al., 2017). Captured three-dimensional sequences of labeled events performed by human actors can be classified as distinct event types. Learning can abstract away the parameters that vary across instances of the same motion class in the data, making those parameters underspecified as well, as in the visualization problem discussed above. In order for an embodied agent to interact with objects, the agent must use its hands, and the hand motions effect forces upon the object, and therefore the action undertaken with it. Thus, we expect that the same parameter abstraction approach can be used for the agent's hand motions, regardless of whether an actual object is being manipulated. This creates a path toward action recognition from hand gestures only.

We assume causal events are composed of an *object model*, which captures the change an object is undergoing over time, and an *action model*, which characterizes the activity that inheres in the causing agent (Pustejovsky and Krishnaswamy, 2016). We have been exploring event visualization through multimodal simulations using scenarios involving objects moving, and event learning and composition through observation focusing on the object position sequence rather than the agent motion. In this paper, we will present results from the former system and methodology from the latter to introduce a framework for learning action recognition from the movements of the *agent* rather than the object. We expect such a framework may be useful for recognizing and evaluating the actions denoted by agent motions enacted without attached objects, e.g., by gestures.

2. Related and Prior Research

Event detection and classification in NLP often rely on deep learning algorithms that exploit shallow lexical features and word embeddings. While these approaches are able to take advantage of big data resources for scalability, they often fail to leverage richer semantic information that situates the event in the world (Spiliopoulou et al., 2017), which is an important factor in QA and event understanding (Saurí et al., 2005).

An agent's *embodiment* might be a physical presence or merely a point of view, but it provides important knowledge about objects in the world, their situatedness, and their availability for different types of interactions. Therefore, we created *visualizations* of events in a three-dimensional visual event simulator, VoxSim (Krishnaswamy and Pustejovsky, 2016a; Krishnaswamy and Pustejovsky, 2016b), and its underlying modeling language, VoxML (Pustejovsky and Krishnaswamy, 2016), while varying the parameters that are left underspecified in the event semantics (as encoded in VoxML), and then presented the visualizations for human evaluation to determine a set of "best values" for said parameters.

Event recognition that combines language and visual data for various purposes is a subject of many models and approaches within the computer vision (Ikizler et al., 2008; Gupta et al., 2009; Cao et al., 2013; Siddharth et al., 2014; Andriluka et al., 2014) and computational linguistic (Ronchi and Perona, 2015; Gella et al., 2016) community. Our rich model of events and their participants also facilitates human communication with a computational agent (Pustejovsky et al., 2017), and so we use the annotation capabilities of VoxML to annotate and learn event representations from existing video data (Do et al., 2016; Do and Pustejovsky, 2017a). Since these two lines of research ap-

$$\begin{bmatrix} \textbf{slide} \\ \text{LEX} = \begin{bmatrix} \text{PRED} = \textbf{slide} \\ \text{TYPE} = \textbf{process} \end{bmatrix} \\ \text{TYPE} = \begin{bmatrix} \text{HEAD} = \textbf{process} \\ \text{ARGS} = \begin{bmatrix} \text{A}_1 = \textbf{x:agent} \\ \text{A}_2 = \textbf{y:physobj} \\ \text{A}_3 = \textbf{z:surface} \end{bmatrix} \\ \text{BODY} = \begin{bmatrix} \text{E}_1 = grasp(x,y) \\ \text{E}_2 = [while(hold(x,y), \\ \quad while(EC(y,z), \\ \quad move(x,y)))] \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 1: Sample VoxML semantics: [[SLIDE]]. Note the absence of speed and direction parameters, indicating they are underspecified. (EC refers to the Region Connection Calculus (Randell et al., 1992) relation "externally connected")

proach event classification and learning from the generation and the recognition sides, respectively, we aim to bridge the two to create a multimodal event representation capable of being learned from sparse data (a la (Do and Pustejovsky, 2017b; Zellers and Choi, 2017)), that can separate the object motion from the complete event model, leaving the agent's motion, or "action model."

3. Event Classification

Using VoxSim, we generated three visualizations for each input sentence of the imperative form $VERB\ x$ (or $VERB\ x\ RELATION\ y$ for those verbs requiring an adjunct). The visualizations were presented to Amazon Mechanical Turk workers for evaluation in a pair of tasks, one of which gave the Turkers a single animated movie of an event and asked them to select, out of three heuristically-generated possible captions (one of which was the original input sentence; the other two vary either the verb or the indirect object if applicable), the best one. Multiple options were allowed as was "none." Each Human Intelligence Task (HIT) was completed by 8 individual workers, for a total of 26,856 individual evaluations. This task effectively required annotators to predict which sentence was used to generate the visualization in question. As this closely resembles event classification with a discrete label set, these results (Krishnaswamy, 2017) provide a "ground truth" against which to assess machine-learning algorithms performing an analogous task.

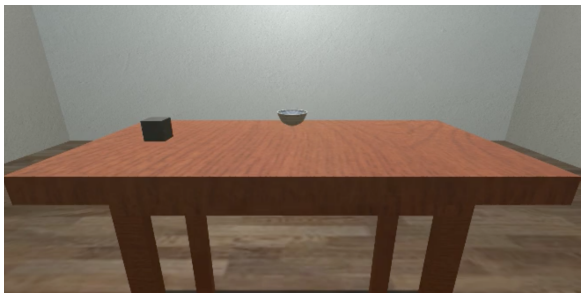

Figure 2: Sample VoxSim capture as presented to evaluators. Caption options for this video were a) "put the block touching the spoon"; b) "move the block" (the original input sentence); and c) "put the block near the bowl."

During the visualization process, we saved feature vectors containing the randomly-generated values for those parameters of each verb that were left underspecified in its semantic encoding. As certain verbs (such as "move") are highly underspecified, with most parameters left without assigned values, while others (for example, "put") may have only one or a few underspecified parameters, these feature vectors were given sparse representations as JSON dictionaries that were then "densified" with empty values for machine learning.

```
{"MotionSpeed":"12.21398",
 "MotionManner":"turn(front_cover)",
 "TranslocSpeed":"",
 "TranslocDir":"",
 "RotSpeed":"",
 "RotAngle":"104.7686",
 "RotAxis":"",
 "RotDir":"",
 "SymmetryAxis":"",
 "PlacementOrder":"",
 "RelOrientation":"",
 "RelOffset":""}
```

Figure 3: "Densified" feature vector for "open the book" action, showing list of parameters evaluated against

The task put to the classifiers trained on these feature vectors was to pick the verb of the input sentence that generated the feature vector and its associated visualization, out of either the same three choices given the human evaluators for the same question (the "restricted" choice set), or all action verbs in the test set (the "unrestricted" choice set).

move(x)	*put(x,touching(y))*	*flip(x,edge(x))*
turn(x)	*put(x,on(y))*	*flip(x,center(x))*
roll(x)	*put(x,in(y))*	*close(x)*
slide(x)	*put(x,near(y))*	*open(x)*
spin(x)	*lean(x,on(y))*	
lift(x)	*lean(x,against(y))*	

Table 1: Event predicate test set

We first established a baseline by feeding these feature vectors into a maximum entropy logistic regression classifier using generalized iterative scaling. Next we trained a multilayer neural network, consisting of four layers of 10, 20, 20, and 10 nodes, respectively, using the TensorFlow framework (Abadi et al., 2016) with a variety of variations:

1. A "vanilla" four-layer DNN
2. DNN with features weighted by IDF metric[1]
3. DNN with IDF weights on only "discrete" features (those features which are maximally specified by choosing a value assignment out of a set of categories rather than a continuous range—i.e., motion manner, rotation axis, symmetry axis, placement order, and relative orientation)

[1]The "inverse document frequency" of a feature (the "term") in a vector (the "document"). Since each feature occurs at most one time in each feature vector, tf for any feature and any vector is either 1 or 0, making TF-IDF over this dataset identical to IDF

4. DNN *excluding* feature values and including IDF-weighted binary presence or absence only
5. A combined linear-DNN classifier, using linear estimation for continuous features and DNN classification for discrete features
6. Combined linear-DNN classifier with features weighted by IDF metric
7. Combined linear-DNN classifier with IDF weights on the discrete features only
8. Combined linear-DNN classifier excluding feature values and including IDF-weighted binary presence or absence only

10-fold cross-validation was run on the baseline and all neural net classifier variations for up to 5,000 training steps, with a convergence threshold of .0001 for the MaxEnt algorithm.

Classifier	μ **Accuracy (restricted set)**	μ **Accuracy (unrestricted set)**
Baseline	0.4850	0.1662
DNN variant 1	0.9788	0.9514
DNN variant 2	0.9788	0.9547
DNN variant 3	0.9800	0.9550
DNN variant 4	**0.9895**	**0.9707**
DNN variant 5	0.9615	0.9150
DNN variant 6	0.9600	0.9144
DNN variant 7	0.9615	0.9675
DNN variant 8	0.9871	0.9150

Table 2: Mean classifier accuracy across cross-validation

All DNN variations identified the motion predicate with greater than 90% accuracy even when given a choice of all available motion predicates. Both DNN and combined Linear-DNN methods that used feature IDF weights only in place of actual feature values actually *outperformed all other methods*. In the purely deep learning network, the weights-only method (variant 4) ends up besting all the others slightly (by about 1-2%). Independent of its actual value, the presence or absence of a given underspecified feature turns out to be quite a strong predictor of motion class.

For this event classification task, we used simulated visualizations of objects moving without being affected by an agent. Since an event's exact manner of underspecification depends on which parameters are missing from the event semantics, we can intuit that, in an action performed by an agent, whether real or simulated, if those same parameters do not remain constant across multiple iterations of the same event, that should be a signal that those agent motions are also denoting an event where those same parameters are underspecified or missing.

4. Complex Event Learning

Many of the events or actions used in the task outlined in Section 3. are quite complex. For example, $lean(x, on(y))$ requires a series of rotations of x and then a movement of x so that it touches y in an appropriate configuration. Even something conceptually simple, such as $put(x, near(y))$, requires a series of translations that can be difficult for a computer to distinguish from other types of motions involving changing relations between two objects.

As this is a sequential learning problem, we turn to LSTM (Hochreiter and Schmidhuber, 1997) to learn the sequence of primitive events that comprise a complex event. LSTM has found utility in a range of problems involving sequential learning, such as speech and gesture recognition. If the sequence can be effectively learned, it should be able to be reproduced by a virtual embodied agent, whose objective is to produce a sequence of actions that resembles movement of objects in the training data. This type of parameterized reinforcement learning is best solved by using policy gradients (Gullapalli, 1990; Peters and Schaal, 2008). Here, we use the REINFORCE algorithm (Williams, 1992), for its effectiveness in policy gradient learning.

Using ECAT, an open-source event capture and annotation tool (Do et al., 2016), we capture performers interacting with objects on a table to replicate the virtual scenes generated with VoxSim, but with the presence of a real agent to manipulate the objects. For the purpose of event learning we limit the object set to only blocks. Video is captured with Microsoft Kinect® depth-sensing cameras, objects are tracked using markers fixed to their sides, and three-dimensional coordinates of performer joints are also captured and annotated. ECAT annotation provides a mapping to VoxML object and event semantics.

Figure 4: Performing an object interaction

Captured object positions are then flattened to two dimensions in order to normalize any jitter in the capture and allow for easier evaluation of object relations relative to the table surface. This simplified simulator is written in Python and allows for simulation of data that is similar to the real captured data without the graphics overhead required by VoxSim.

A sequence of feature vectors, S, which represent the qualitative spatial relations between the objects in the action captures or the simplified simulator, is fed to an LSTM network along with a frame number i and an event e. The network outputs a function $f(S, i, e) = 0 \leq q_i \leq 1$ that estimates the progress of e at frame i.

The virtual agent's objective is then to manipulate the objects in sequence, for a reward that is greater when the generated sequence more closely approximates the movement of objects in the training data. We aim to achieve this via reinforcement learning, using the REINFORCE algorithm with a Gaussian distribution policy $\pi_\theta(u|x) = Gaussian(\mu, \sigma)$, where $dim(\mu)$ is the degree of freedom

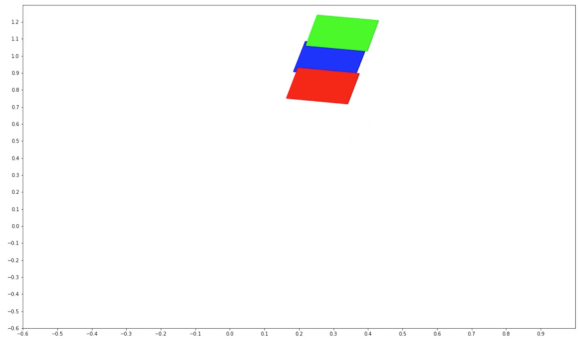

Figure 5: Simplified simulator in two dimensions

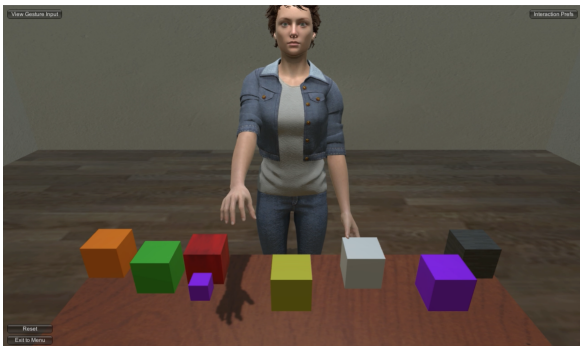

Figure 6: Frame of an agent demonstrating a gesture representing "slide"

in position (2 dimensions) and $dim(\sigma)$ is the degree of freedom in orientation (1 dimension).

Planning is parameterized by policy parameters θ : $u_k \sim \pi_\theta(u_k|x_k)$, where u_k is the motion performed by the agent at step k and x_k is current set of relations between objects. μ and σ are learned by an artificial neural network weighted by θ from the REINFORCE algorithm, which is determined by gradient descent.

We simulate each atomic object manipulation u_k, record the frame-to-frame sequential features, feed them into LSTM network to estimate how fully u_k completes the complex event in question, then calculate the immediate reward as the difference between the complex event progress at the beginning of u_k and at the end of u_k, and finally select the agent move that leads to the highest reward.

The result is a sequence that can be executed by a virtual agent within the VoxSim environment. If successful, a human judge watching this executed event should agree that it satisfies the event class of the description as in the event classification task described in Section 3.. Experiments are currently ongoing to test this model of event learning (Do et al., 2018).

5. Extracting Actions from Events

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

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

For instance, let us return to the semantics of "slide" presented in Figure 1. One of the requirements is that at all times the moving object is kept EC (externally connected) with the supporting surface. Since in a 3D environment, all motions eventually break down into a series of translations and rotations, all relations between objects can be represented as relative offsets and orientations, as in the reinforcement learning trials. Thus, if "sliding" motions of various speeds and moving in various directions all return roughly equal rewards as long as the object remains attached to the supporting surface (as the LSTM should produce high values of event progress for all these motions given enough performed examples), the REINFORCE algorithm should be able to generate an event sequence wherein many values for these parameters can be sampled from the Gaussian distribution, and the action, when performed by an agent with those values, should satisfy an observer's judgment given the "slide" label. Thus the high variance of motion speed and motion direction comport with those parameters' status as strong signals of the "slide" event class.

Since in the 3D simulated world with the agent, objects are manipulated by attaching them to the agent's "graspers" or hands, so that the motion of the hand controls the motion of a grasped object, it is the motion of the hand that dictates what class of action is being undertaken. Thus in the above example, if the hand motion may take a wide variety of values of speed and direction but always maintains a constant or near-constant vertical offset with the surface (representing the height of the object being moved), then this motion may be interpreted as representing a "slide," regardless of whether or not any actual object is being moved. If no object is moved along with the hand, this "action model" becomes a "mime" or gestural representation of the action in question.

6. Conclusion

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

tion performance and gestural representations of actions as a learnable communicative modality between humans and computers.

7. Acknowledgements

The authors would like to thank the reviewers for their helpful comments, and Alex Luu for his help in performing event training examples. This work is supported by a contract with the US Defense Advanced Research Projects Agency (DARPA), Contract W911NF-15-C-0238. Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

8. Bibliographical References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA*.

Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693.

Cao, Y., Barrett, D., Barbu, A., Narayanaswamy, S., Yu, H., Michaux, A., Lin, Y., Dickinson, S., Mark Siskind, J., and Wang, S. (2013). Recognize human activities from partially observed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2658–2665.

Chang, A., Monroe, W., Savva, M., Potts, C., and Manning, C. D. (2015). Text to 3D scene generation with rich lexical grounding. *arXiv preprint arXiv:1505.06289*.

Coyne, B. and Sproat, R. (2001). WordsEye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 487–496. ACM.

Do, T. and Pustejovsky, J. (2017a). Fine-grained event learning of human-object interaction with lstm-crf. *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*.

Do, T. and Pustejovsky, J. (2017b). Learning event representation: As sparse as possible, but not sparser. *arXiv preprint arXiv:1710.00448*.

Do, T., Krishnaswamy, N., and Pustejovsky, J. (2016). ECAT: Event capture annotation tool. *Proceedings of ISA-12: International Workshop on Semantic Annotation*.

Do, T., Krishnaswamy, N., and Pustejovsky, J. (2018). Teaching virtual agents to perform complex spatial-temporal activities. *AAAI Spring Symposium: Integrating Representation, Reasoning, Learning, and Execution for Goal Directed Autonomy*.

Gella, S., Lapata, M., and Keller, F. (2016). Unsupervised visual sense disambiguation for verbs using multimodal embeddings. *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, 182-192. San Diego.*

Gullapalli, V. (1990). A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural networks*, 3(6):671–692.

Gupta, A., Kembhavi, A., and Davis, L. S. (2009). Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1775–1789.

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

Ikizler, N., Cinbis, R. G., Pehlivan, S., and Duygulu, P. (2008). Recognizing actions from still images. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE.

Krishnaswamy, N. and Pustejovsky, J. (2016a). Multimodal semantic simulations of linguistically underspecified motion events. In *Spatial Cognition X: International Conference on Spatial Cognition*. Springer.

Krishnaswamy, N. and Pustejovsky, J. (2016b). VoxSim: A visual platform for modeling motion language. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. ACL.

Krishnaswamy, N. (2017). *Monte-Carlo Simulation Generation Through Operationalization of Spatial Primitives*. Ph.D. thesis, Brandeis University.

Paul, R., Arkin, J., Roy, N., and Howard, T. (2017). Grounding abstract spatial concepts for language interaction with robots. In *IJCAI-17 Proceedings*.

Peters, J. and Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697.

Pustejovsky, J. and Krishnaswamy, N. (2016). VoxML: A visualization modeling language. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, May. European Language Resources Association (ELRA).

Pustejovsky, J., Krishnaswamy, N., and Do, T. (2017). Object embodiment in a multimodal simulation. *AAAI Spring Symposium: Interactive Multisensory Object Perception for Embodied Agents*.

Randell, D., Cui, Z., Cohn, A., Nebel, B., Rich, C., and Swartout, W. (1992). A spatial logic based on regions and connection. In *KR'92. Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference*, pages 165–176, San Mateo. Morgan Kaufmann.

Ronchi, M. R. and Perona, P. (2015). Describing common human visual actions in images. *arXiv preprint arXiv:1506.02203*.

Saurí, R., Knippen, R., Verhagen, M., and Pustejovsky, J. (2005). Evita: a robust event recognizer for qa systems. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language*

Processing, pages 700–707. Association for Computational Linguistics.

Siddharth, N., Barbu, A., and Mark Siskind, J. (2014). Seeing what you're told: Sentence-guided activity recognition in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 732–739.

Siskind, J. M. (2001). Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *J. Artif. Intell. Res.(JAIR)*, 15:31–90.

Spiliopoulou, E., Hovy, E., and Mitamura, T. (2017). Event detection using frame-semantic parser. In *Proceedings of the Events and Stories in the News Workshop*, pages 15–20.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Yang, Y., Fermüller, C., and Aloimonos, Y. (2013). Detection of manipulation action consequences (mac). In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2563–2570. IEEE.

Yang, Y., Guha, A., Fermuller, C., and Aloimonos, Y. (2014). A cognitive system for understanding human manipulation actions. *Advances in Cognitive Sysytems*, 3:67–86.

Yang, Y., Aloimonos, Y., Fermuller, C., and Aksoy, E. E. (2015a). Learning the semantics of manipulation action. *arXiv preprint arXiv:1512.01525*.

Yang, Y., Li, Y., Fermüller, C., and Aloimonos, Y. (2015b). Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web. In *AAAI*, pages 3686–3693.

Zellers, R. and Choi, Y. (2017). Zero-shot activity recognition with verb attribute induction. *arXiv preprint arXiv:1707.09468*.