ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

Fine-grained Event Learning of Human-Object Interaction with LSTM-CRF

Tuan Do and James Pustejovsky [*]

Brandeis University - Department of Computer Science
Waltham, Massachusetts - United States of America

Abstract.
Event learning is one of the most important problems in AI. However, notwithstanding significant research efforts, it is still a very complex task, especially when the events involve the interaction of humans or agents with other objects, as it requires modeling human kinematics and object movements. This study proposes a methodology for learning complex human-object interaction (HOI) events, involving the recording, annotation and classification of event interactions. For annotation, we allow multiple interpretations of a motion capture by slicing over its temporal span; for classification, we use Long-Short Term Memory (LSTM) sequential models with Conditional Randon Field (CRF) for constraints of outputs. Using a setup involving captures of human-object interaction as three dimensional inputs, we argue that this approach could be used for event types involving complex spatio-temporal dynamics.

1 Introduction

The study of events has long involved many disciplines, including philosophy, cognitive psychology, linguistics, computer science, and AI. The Gestalt school of philosophy characterized events as whole processes that emerge from the relations between their components. Cognitive psychologists, such as Tulving [1], recognized the importance of events by postulating a separate cognitive process called *episodic memory*. The representation of events in natural language has been studied from many different approaches, from formal logic and AI [2], and frames [3], to computational linguistics [4]. Combining perspectives from computer science, logic, and linguistics, some recent work suggests that events can be effectively modeled as programs within a dynamic logic (DITL) [5], enabling computer simulations of linguistic expressions [6].

In computer science, there is little consensus about how events should be modeled for learning. They can be represented atomically, i.e., entire events are predicted in a classification manner [7], or as combinations of more primitive actions [8], i.e., complex event types are learned based on recognition of combined primitive actions. For the former type of event representation, there are quantitative approaches based on low-level pixel features such as in [9] and qualitative approaches such as induction from relational states among event participants [10]. For the latter approach, systems such as [11], use state transition graphical models such as Dynamic Bayesian Networks (DBN).

[*]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. We would like to thank Nikhil Krishnaswamy and Keigh Rim for their discussion and input on this topic. All errors and mistakes are, of course, the responsibilities of the authors..

While learning events as a whole works best for human motion signatures such as *running*, *sitting* etc., it poses a problem for event types that require distinctions in spatio-temporal relationships between objects. As pointed in [10], it is also difficult to model events as strict orderings of subevents, especially when there are *overlapping* or *during* relations between them. Moreover, if the purpose of event learning is to facilitate communication and interaction between human and computational agents, such as robots, to achieve some common goals, these agents need to keep track of multiple events at the same time, involving themselves, other humans, as well as the surrounding environment. From a practical point of view, this calls for a finer-grained treatment of event modeling.

It is also the case that a fine-grained analysis of events is strongly supported from a theoretical point of view. For example, it has long been known that event classification needs to take into account what is called *extra-verbal factors*. Event types should not be semantically defined only by a base verbal expression, such as *running* or *walking*, but need to incorporate other components of the expression compositionally, such as objects and adjuncts, which can change the event type of the overall verb phrase or sentence [12].

Motivated by those arguments, we suggest a different approach to event learning. Instead of treating events as whole, or as programs of subevents, we allow multiple interpretations of a motion capture by slicing over its temporal span and give a separate annotation for each slice.

In particular, we use an event capture and annotation tool called ECAT [13], which employs Microsoft Kinect® to capture sessions of performers interacting with two types of objects, a cube (which can be slid on a flat surface) and a cylinder (which can be rolled). Objects are tracked using markers fixed to their sides facing the camera. They are then projected into three dimensional space using depth of field. Performers are tracked using the Kinect API, which provides three dimensional inputs of a performer's joint points (e.g., wrist, palm, shoulder). Sessions are first sliced, and each slice is annotated with a textual description using our event language. Our sequence learning algorithms (LSTM-CRF) will input sequences of feature vectors and output a representation of an event.

The main contributions of our study are twofolds. Firstly we created a framework for event recording and annotation that takes into account their temporal dynamics, i.e., different interpretations of events on different temporal spans. Applying a flavor of the popular sequential learning method LSTM that accommodates to output constraints, we achieved good performance in our human-object interaction setup.

2 Learning Framework

We used three dimensional coordinates of bodies tracked by Kinect SDK to model human kinematics. Only joint points on the upper body of performers (13 joint points) are kept and concatenated, because their lower parts are occluded when they interact with objects on top of a table. In addition, for each marker detected (using Glyph detection algorithm[14]), we generate 12 features (4 3-D corners). Features of objects are concatenated into a vector of a fixed size (there is always one performer and two objects tracked). A sample in the dataset consists of a sequence of fixed length of feature vectors. Its label is mapped from the textual annotation into an output structure (Subject, Object, Locative, Verb, Preposition). We will call this output structure $(Y_1, \ldots, Y_5)$.

2.1 LSTM

LSTM is a flavor of deep Recursive neural network (RNN) that has generally solved the problem of "vanishing gradients" in traditional RNN learning [15, 16] and has found their application in a wide range of problems involving sequential learning, such as hand written recognition, speech recognition, gesture recognition, etc.

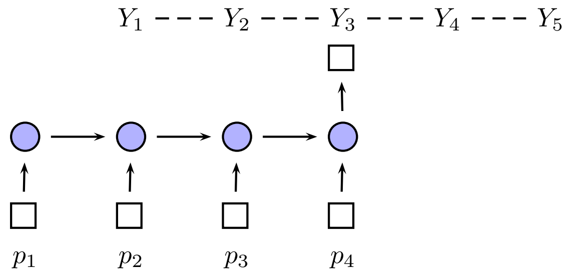

Fig. 1: LSTM model with possible constraints of outputs with CRF. CRF layer is represented as dashed links among predicted labels.

We will not describe in detail here our LSTM implementation, as we provide online access to the code and approach [1]. Briefly, however, the model passes each feature vector through a linear layer before feeding each sequence into an LSTM. Each label Y_i requires a separate LSTM cell, X_i. Depending on whether we predict each label independently and combine them for final prediction, or we predict on the basis of the sum of outputs from the last layer, two variants are considered, correspondingly LSTM-I and LSTM-W.

2.2 CRF

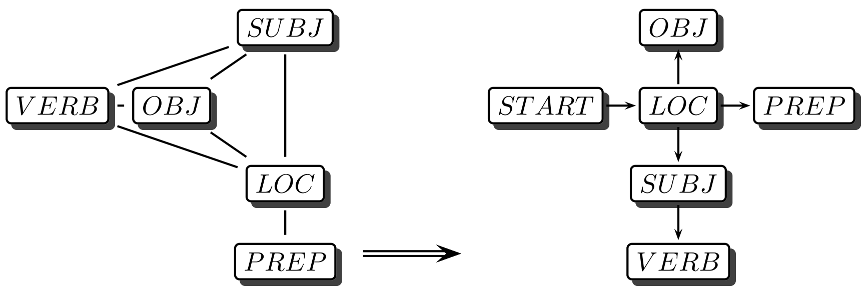

Fig. 2: Reformation from general **CRF** (left) to **Tree-CRF** (right)

CRF has been used extensively to learn structured output as it allows specification of constraints of output labels[17]. In this model we wish to constrain the

[1]https://github.com/tuandnvn/ecat_learning

outputs so that: one object (Performer or the other objects) is not allowed to fill two different syntactic slots; when there is no verb, all the other slots should be None; locative and preposition are dependent, because if locative is None, preposition must also be None and vice versa. The edges between nodes on the left side of Figure 2 show the dependencies on output labels that we wish to model.

However, training and classifying using a full CRF model would be more difficult, especially when implemented with a neural network architecture. We modified the model into a tree-CRF structure (right side of Figure 2) to make the model learnable using dynamic programming. The complexity of the algorithm reduced from $O(n^5)$ to $O(n^2) * 5)$ where n is the size of our vocabulary. The learning problem is thereby changed to learning the weights along the edges on the tree-CRF, for example, *P_locative_preposition* (together with parameters of LSTM). The directionality of edges is the forward direction of the message passing algorithm used for learning (and in reverse, for testing using the backward direction).

2.3 LSTM-CRF

LSTM-CRF is a natural extension of LSTM applied for constrained outputs. For instance it is used for named entities recognition task to model constraints on BIO labels[18]. To put CRF learning on top of LSTM-W, we modify the term t (the term before softmax) produced by outputs of LSTM as followings.

$$\begin{aligned} t(l,s,o,p,v) &= t_l + t_s + t_o + t_p + t_v \quad \textbf{Original LSTM-W} \\ t(l,s,o,p,v) &= t_l + t_s + t_o + t_p + t_v \quad \textbf{Modified} \\ &\quad + P_{start\_l} + P_{ls} + P_{lo} + P_{lp} + P_{sv} \end{aligned}$$

where l, s, o, p, v stand for Locative, Subject, Object, Preposition and Verb respectively.

In training, $softmax$ is calculated for a predicted label combination, namely (l', s', o', p', v') as below. We can calculate the log of sum using message passing over the tree nodes of the CRF tree. We use cross entropy between predicted distribution and correct output as the $cost$ in training.

$$\begin{aligned} softmax &= exp[t(l',s',o',p',v') - log[\sum_l \sum_s \sum_o \sum_p \sum_v exp(t(l,s,o,p,v))]] \\ &= exp[t(l',s',o',p',v') - log[\sum_l exp(t_l + P_{start\_l})[\sum_s exp(t_s + P_{ls}) \\ &\sum_v exp(t_v + P_{sv})][\sum_o exp(t_o + P_{lo})][\sum_p exp(t_p + P_{lp})]] \end{aligned}$$

In evaluation, a similar message passing algorithm is used, but instead of *log_sum*, we use max to calculate the probabilities and $argmax$ to keep track of the best combination.

3 Experiments

3.1 Event Capture and Annotation

To demonstrate our model's capability to learn the *spatio-temporal* dynamics of object interactions in events, we use a collection of four action types: *push*, *pull*, *slide*, and *roll*, along with three different *spatial* prepositions used for space configurations between objects, namely *toward* (when the trajectory of a moving object is straightly lined up with a destination static object and makes it closer to that target), *away from* (makes it further from that object) and *past* (moving object getting closer to static object then further again).

Afterwards, for each session, we sliced the events into short segments of 20 frames. Two annotators were assigned to watch and annotate them (segments can be played back). To speed up annotation, only event types related to original captured types are shown for selection. For instance, if the event type of the captured session is "The performer pushes A toward B", other available event types are "The performer pushes A", "A slides toward B" or "None".

3.2 Classification Results

Our LSTM models have one hidden layer of 200 features. Two methods are used to combat over-fitting: (i) dropout in LSTM cell with probability of 0.8, and (ii) gradient clipping by a global norm. The network is trained with mini-batch gradient descent optimization for 200 epochs on the Tensorflow library.

Most frequent label tagging is used as the baseline for this study: that is, we simply predict any sample with the most frequent tuple seen in the training corpus.

Captured sessions are split into training and testing sets on the proportion of 60/40, i.e., 18 sessions of training and 12 sessions for testing for each event type. That gives a total of 2680 training samples and 1840 testing samples. Precisions reported are averaged over 5 runs (each run is obtained with a random initialization). Breakdowns of precision for each label show that verb precision is the lowest, with high confusion between the following pairs: *push* vs *roll*, *pull* vs *roll*, and *slide* vs *roll*. It is likely because of poor tracking result when objects are rolled. In fact, the capture tool could not recognize objects in many frames when objects roll fast, and it compensated by using interpolation. Improvement on tracking of objects, however, is not the target of our study.

Model	Precision
Baseline	6%
LSTM-I	38%
LSTM-W	39%
LSTM-CRF	43%

Table 1: Evaluation

Label	Precision
Subject	86%
Object	87%
Locative	73%
Verb	68%
Preposition	72%

Table 2: Label precision breakdown

We observed significant improvement of learning using LSTM over baseline, especially when it is coupled with CRF. Moreover, we also observed a reduction of invalid outputs from 20% to 3% when CRF is used. We consider these results to be quite good, particularly since the sequential learning model we used is both simple and fast, and we did not employ any feature engineering method.

4 Conclusion and Future Directions

In this paper, we have demonstrated a methodology that provides reasonable learning results for a set of complex event types. We hope that our study will provide a starting point for further investigations into fine-grained event learning. Currently our learning method requires a fix number of objects in inputs, which could be overcome by incrementally adding object features into a fix size feature vector, possibly by using a recursive neural network. Regarding our annotation framework, a natural extension is that spans of different lengths could be annotated with appropriate re-sampling methods. We leave these as some of our future research topics.

We are currently applying this learning pipeline (simple interval annotation + sequential learning with constraint outputs) on a large movie dataset that have event annotations. We are looking to publish our results in the near future.

References

[1] Endel Tulving. *Elements of Episodic Memory.* Oxford University Press, 1983.

[2] James Allen. Towards a general theory of action and time. *Arificial Intelligence*, 23:123–154, 1984.

[3] C. Fillmore. *Santa Cruz Lectures on Deixis.* Indiana University Linguistics Club. Bloomington, IN, 1975.

[4] James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. Iso-timeml: A standard for annotating temporal information in language. In *LREC*, 2011.

[5] James Pustejovsky and Jessica L Moszkowicz. The qualitative spatial dynamics of motion in language. *Spatial Cognition & Computation*, 11(1):15–44, 2011.

[6] James Pustejovsky and Nikhil Krishnaswamy. Generating simulations of motion events from verbal descriptions. *Lexical and Computational Semantics (* SEM 2014)*, page 99, 2014.

[7] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. *arXiv preprint arXiv:1604.02808*, 2016.

[8] Harini Veeraraghavan, Nikolaos Papanikolopoulos, and Paul Schrater. Learning dynamic event descriptions in image sequences. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE, 2007.

[9] Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3361–3368. IEEE, 2011.

[10] Krishna Sandeep Reddy Dubba, Anthony G Cohn, David C Hogg, Mehul Bhatt, and Frank Dylla. Learning relational event models from video. *Journal of Artificial Intelligence Research*, 53:41–90, 2015.

[11] Anthony Hoogs and AG Amitha Perera. Video activity recognition in the real world. In *AAAI*, pages 1551–1554, 2008.

[12] James Pustejovsky. *The Generative Lexicon.* MIT Press, Cambridge, MA, 1995.

[13] Tuan Do, Nikhil Krishnaswamy, and James Pustejovsky. Ecat: Event capture annotation tool. *Proceedings of ISA-12: International Workshop on Semantic Annotation*, 2016.

[14] Andrew Kirillov. From glyph recognition to augmented reality. `https://www.codeproject.com/articles/258856/from-glyph-recognition-to-augmented-reality`. Accessed: 2016-03-15.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[17] Charles Sutton and Andrew McCallum. *An introduction to conditional random fields for relational learning*, volume 2. Introduction to statistical relational learning. MIT Press, 2006.

[18] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.