Learning & Reasoning for Logic-Based Complex Event Recognition

Alexander Artikis[2,1] and Nikos Katzouris[1]

[1] Institute of Informatics, National Center for Scientific Research (NCSR) "Demokritos", Athens, Greece
[2] University of Pireaus, Pireaus, Greece
{nkatz,a.artikis}@iit.demokritos.gr

Abstract. Complex Event Recognition (CER) refers to the detection of special events of interest in a multitude of large, heterogeneous and interdependent data streams. In this short paper we discuss several aspects of CER research, such as efficient detection of patterns on event streams, handling uncertainty and noise in streams, and machine learning techniques for inferring interesting patterns. We focus on logic-based methods for CER, which are of special interest thanks to their formal semantics and their direct connections to uncertainty handling and machine learning via Statistical Relational Learning and Inductive Logic Programming.

1 Introduction

The proliferation of devices that work in real-time, constantly producing data streams, has generated new challenges for systems working with massive amounts of data. Such systems aim to extract actionable knowledge from continuously evolving data streams, without storing every bit of the incoming information for processing at a later time, which is infeasible. This is achieved by defining a set of queries/patterns, continuously applied to the data streams. Each such pattern includes a set of temporal constraints and, possibly, a set of spatial constraints, expressing a composite or complex event of special significance for a given application. The system must then be efficient enough so that instances of pattern satisfaction can be reported to a user with minimal latency. Such systems are called Complex Event Recognition (CER) systems [8, 9, 1].

CER systems are widely adopted in contemporary applications, including, among others, security, infrastructure monitoring & predictive maintainance, transportation & logistics, assisted living, fraud detection and medicine. Moreover, Big Data frameworks, such as Apache Storm, Spark Streaming and Flink, have been extending their stream processing functionality by including implementations for CER.

A CER system faces multiple challenges. One issue is the requirement for minimal latency, leading to the need for highly efficient reasoning mechanisms, scalable to high-velocity streams. As such streams are typically noisy and domain knowledge often insufficient or incomplete, noise resilience and uncertainty handling are also important. Moreover, complex event patterns may not always be available and even if they are, they frequently need to be revised/updated to adapt to change in the dynamic input.

This poses an additional challenge of using machine learning in order to extract patterns from streams and/or revise existing ones from new data.

In this short paper we discuss several aspects of reasoning, uncertainty handling and machine learning for CER from the perspective of logic-based CER systems, i.e. systems that adopt a logical representation of event patterns. Altough numerous CER systems have been proposed in the literature [8, 9], logic-based CER systems are of particular interest because they exhibit a formal, declarative semantics [1], they facilitate the use of domain knowledge in the CER process, and allow reasoning with complex relations between entities. Moreover, they have direct connections to uncertainty handling and machine learning techniques via Statistical Relational Learning (SRL), and Inductive Logic Programming (ILP).

2 Reasoning & Uncertainty Handling

Temporal domains are dynamic in nature and often characterized by commosense phenomena, which should be taken into account in CER applications. Building temporal reasoning engines on top of action formalisms is a way to address that. RTEC (Event Calculus for Run-Time reasoning) [3] is such an engine, relying on an efficient dialect of the Event Calculus, which is a logic programming formalism for representing and reasoning about events and their effects [17]. Complex event (CE) patterns in RTEC identify the conditions in which a CE is initiated and terminated. Then, according to the law of inertia, a CE holds at a time-point T if it has been initiated at some time-point earlier than T, and has not been terminated in the meantime.

RTEC has been optimised for CER, in order to be scalable to high-velocity data streams. A form of caching stores the results of subcomputations in the computer memory to avoid unnecessary recomputations. A set of interval manipulation constructs simplify CE patterns and improve reasoning efficiency. A simple indexing mechanism makes RTEC robust to events that are irrelevant to the patterns we want to match and so RTEC can operate without data filtering modules. Finally, a 'windowing' mechanism supports real-time CER. One main motivation for RTEC is that it should remain efficient and scalable in applications where events arrive with a (variable) delay from, or are revised by, the underlying sensors: RTEC can update the intervals of the already recognised CEs, and recognise new CEs, when data arrive with a delay or following revision.

RTEC has been analysed theoretically, through a complexity analysis, and assessed experimentally in several application domains, including city transport and traffic management [4], activity recognition on video feeds [3], and maritime monitoring [22]. In all of these applications, RTEC has proven capable of performing real-time CER, scaling to large data streams and highly complex event patterns.

CER applications exhibit various types of uncertainty, ranging from incomplete and erroneous data streams to imperfect CE patterns [1]. A line of research for handling uncertainty in CER extends the Event Calculus with probabilistic reasoning. Prob-EC [25] is a logic programming implementation of the Event Calculus using the ProbLog engine [16], that incorporates probabilistic semantics into logic programming. Prob-EC is the first Event Calculus dialect able to deal with uncertainty in the input data streams.

For example, Prob-EC is more resilient to spurious data than the standard (crisp) Event Calculus.

MLN-EC [24] is an Event Calculus implementation based on Markov Logic Networks (MLN). CE patterns may be associated with weight values, indicating our confidence in them. Inference can then be performed regarding the time intervals during which CEs of interest hold. Like Prob-EC, MLN-EC increases the probability of a CE every time its initiating conditions are satisfied, and decreases this probability whenever its terminating conditions are satisfied. Moreover, in MLN-EC the domain-independent Event Calculus rules, expressing the law of inertia, may be associated with weight values, introducing probabilistic inertia. This way, the model is highly customisable, by tuning appropriately the weight values with the use of machine learning techniques, and thus achieves high predictive accuracy in a wide range applications.

3 Machine Learning

The manual authoring of CE patterns is a tedious and error-prone process. Consequently, the automated construction of such patterns from data is highly desirable and is attracting attention in the event processing community [19]. However, existing techniques have several limitations [11]. They often resort to ad-hoc learning techniques, tailored towards learning ad-hoc event pattern specification languages, which are hard to evaluate in more generic learning settings; They have limited support for using background knowledge and handling uncertainty, while they all assume a batch learning setting, which is inadequate in stream-handling CER applications. We present some techniques for logic-based machine learning for CER, which are able to overcome most of the aforementioned limitations. We mainly focus on online, single-pass algorithms, capable of dealing with streaming data.

As mentioned earlier, we are dealing with CE patterns in the form of temporal rules in the Event Calculus. Although ILP and SRL offer a variety of techniques for learning logical representations, learning with the Event Calculus is challenging for most existing relational learning approaches. The main reason for that is the non-monotonicity of the Negation as Failure operator that the Event Calculus uses for representing inertia, which makes the divide-and-conquer-based search of most ILP algorithms inappropriate. Non-monotonic ILP algorithms can handle the task [23, 7], but they scale poorly, as they learn whole theories from the entirety of the training data, while improvements to such algorithms that allow some form of incremental processing to enhance efficiency [5, 18] cannot handle data arriving over time.

Closely related to non-monotonic ILP algorithms is ILED [12] (Incremental Learning of Event Definitions), which learns Event Calculus theories incrementally, using theory revision techniques. It is a full-memory system, meaning that revisions account for a growing historical memory of accumulated data. However, its core operators are applicable in an online setting, where the data are presented in mini-batches (each mini-batch containing e.g. data from a recent period of time) and the theory at hand is minimally revised to account for the data in each new batch. The main drawback of such an approach is that the heurstic value of the revisions is constrained within the most recent data batch, which may result to sub-optimal revisions. In principle, larger data batches

lead to better revisions at the expense of longer training times, and the tradeoff is not straightforward to assess.

An alternative approach is put forth by OLED (Online Learning of Event Definitions) [13], which gives-up ILED's expensive, theory-level search and treats initiation and termination conditions of CEs as two different concepts, which are learnt in parallel in a rule-by-rule fashion. Rules' quality is assessed differently for each type of rule (initiation or termination). This is based on estimating from the training data each rule's utility in a "virtual theory", where it would be used with the Event Calculus in the backgound knowledge. OLED uses a standard hill-climbing approach, gradually adding literals to rules' bodies based on whether they improve the current version of the rule. The hill-climbing process is made online via Hoeffding bounds, which allow to assess the quality of candidate specializations on a subset of the input, with high probability.

OLED has been evaluated on several applications, including activity recognition [13], maritime monitoring [14], credit card fraud detection [2] and community evolution prediction in social networks [6]. On some of these applications it was shown able to compete with several batch ILP and SRL approaches in terms of the quality of its learnt theories, while yielding speed-ups of several orders of magnitude. Recently, it has been extended to a parallel version [14], which splits training over a data stream accross multiple processors, yielding significant speed-ups over the serial algorithm.

In addition to crisp ILP learners, a number of SRL learners tailored to the ML-for-CER task have also been proposed. OSLa [21] is such a learner for Markov Logic Networks (MLNs), which extends previous work [10] by exploiting the background knowledge in order to significantly constrain the search for patterns. OSLa works by constantly updating the rules in an MLN in the face of new data that stream-in, by adding new rules and updating the weights of existing ones. The addition of new rules takes place in response to a "missed" (false negative) CE instance and is based on a relational pathfinding process. First, "paths" of atoms connected via domain constants are identified. For each such path, a new rule is generated, having the missed CE instance in the head and the conjunction of the path atoms in the body. Finally, a lifted version of each new rule is generated by replacing constants by variables. The new lifted rules are then added to the MLN theory. AdaGrad-based weight learning is responsible for eventually pushing the weights of irrelevant rules to zero. OLED has also been combined with MLN [15], interleaving its single-pass hill-climbing structure learning process with AdaGrad-based weight learning.

Since labelled data in real-world CER applications are often scarce, semi-supervised techniques that take advantage of small quantities of available labelled data towards inferring missing labels are of particular interest. One such approach is SPLICE [20]. It relies on graph-cut minimisation, a technique that derives labels for unlabelled data, based on their distance to their labelled counterparts. The technique is adapted to first order logic via a suitable structural distance function for measuring the distance between sets of logical atoms. The labelling process in SPLICE's technique is done online (single-pass), by means of a caching mechanism that stores previously seen labels for future usage. Experiments on activity recognition and maritime monitoring applications showed that SPLICE can assist an online structure learner (OSLa and OLED

where used) to construct high-quality theories using as few as 20% of the original supervision on some occasions.

4 Conclusions

We presented an overview of techniques for logic-based reasoning, uncertainty handling and learning in temporal domains, towards efficient and robust CER. More information on logic-based CER, including papers, code and datasets for the work discussed here are freely available from the we page of the CER group of NCSR "Demokritos"[3].

References

1. Elias Alevizos, Anastasios Skarlatidis, Alexander Artikis, and Georgios Paliouras. Probabilistic complex event recognition: A survey. *ACM Comput. Surv.*, 50(5):71:1–71:31, 2017.
2. Alexander Artikis, Nikos Katzouris, Ivo Correia, Chris Baber, Natan Morar, Inna Skarbovsky, Fabiana Fournier, and Georgios Paliouras. A prototype for credit card fraud management: Industry paper. In *DEBS*, pages 249–260, 2017.
3. Alexander Artikis, Marek J. Sergot, and Georgios Paliouras. An event calculus for event recognition. *IEEE TKDE*, 27(4):895–908, 2015.
4. Alexander Artikis, Matthias Weidlich, François Schnitzler, Ioannis Boutsis, Thomas Liebig, Nico Piatkowski, Christian Bockermann, Katharina Morik, Vana Kalogeraki, Jakub Marecek, Avigdor Gal, Shie Mannor, Dimitrios Gunopulos, and Dermot Kinane. Heterogeneous stream processing and crowdsourcing for urban traffic management. In *EDBT*, pages 712–723, 2014.
5. D. Athakravi, D. Corapi, K. Broda, and A. Russo. Learning through hypothesis refinement using answer set programming. In *ILP-2013*, pages 31–46. Springer, 2013.
6. George Athanasopoulos, George Paliouras, Dimitrios Vogiatzis, Grigorios Tzortzis, and Nikos Katzouris. Predicting the evolution of communities with online inductive logic programming. In *25th International Symposium on Temporal Representation and Reasoning, TIME 2018, Warsaw, Poland, October 15-17, 2018*, pages 4:1–4:20, 2018.
7. D. Corapi, A. Russo, and E. Lupu. Inductive logic programming as abductive search. In *ICLP-2010*, pages 54–63, 2010.
8. Gianpaolo Cugola and Alessandro Margara. Processing flows of information: From data stream to complex event processing. *ACM Comput. Surv.*, 44(3):15:1–15:62, 2012.
9. Nikos Giatrakos, Alexander Artikis, Antonios Deligiannakis, and Minos N. Garofalakis. Complex event recognition in the big data era. *PVLDB*, 10(12):1996–1999, 2017.
10. T. N. Huynh and R. J. Mooney. Online Structure Learning for Markov Logic Networks. In *ECML*, pages 81–96, 2011.
11. N. Katzouris. Scalable relational learning for event recognition. *PhD Thesis, University of Athens*, http://users.iit.demokritos.gr/ nkatz/papers/nkatz-phd.pdf, 2017.
12. Nikos Katzouris, Alexander Artikis, and Georgios Paliouras. Incremental learning of event definitions with inductive logic programming. *Machine Learning*, 100(2-3):555–585, 2015.
13. Nikos Katzouris, Alexander Artikis, and Georgios Paliouras. Online learning of event definitions. *TPLP*, 16(5-6):817–833, 2016.
14. Nikos Katzouris, Alexander Artikis, and Georgios Paliouras. Parallel online event calculus learning for complex event recognition. *Future Generation Comp. Syst.*, 94:468–478, 2019.

[3] `http://cer.iit.demokritos.gr`

15. Nikos Katzouris, Evangelos Michelioudakis, Alexander Artikis, and Georgios Paliouras. Online learning of weighted relational rules for complex event recognition. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part II*, pages 396–413, 2018.
16. Angelika Kimmig, Bart Demoen, Luc De Raedt, Vítor Costa, and Ricardo Rocha. On the implementation of the probabilistic logic programming language ProbLog. *TPLP*, 11(2-3):235–262, 2011.
17. Robert A. Kowalski and Marek J. Sergot. A logic-based calculus of events. *New Generation Comput.*, 4(1):67–95, 1986.
18. M. Law, A. Russo, and K. Broda. Iterative learning of answer set programs from context dependent examples. *Theory and Practice of Logic Programming*, 16(5-6):834–848, 2016.
19. Alessandro Margara, Gianpaolo Cugola, and Giordano Tamburrelli. Learning from the past: automated rule generation for complex event processing. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, pages 47–58. ACM, 2014.
20. E. Michelioudakis, A. Artikis, and Paliouras G. Semi-supervised online structure learning for composite event recognition. *Machine Learning*, to appear, 2019.
21. E. Michelioudakis, A. Skarlatidis, G. Paliouras, and A. Artikis. Online structure learning using background knowledge axiomatization. In *ECML*, 2016.
22. Kostas Patroumpas, Elias Alevizos, Alexander Artikis, Marios Vodas, Nikos Pelekis, and Yannis Theodoridis. Online event recognition from moving vessel trajectories. *GeoInformatica*, 21(2), 2017.
23. Oliver Ray. Nonmonotonic abductive inductive learning. *JAL*, 7(3):329–340, 2009.
24. A. Skarlatidis, G. Paliouras, A. Artikis, and G. A. Vouros. Probabilistic Event Calculus for Event Recognition. *ACM Transactions on Computational Logic*, 16(2):11:1–11:37, 2015.
25. Anastasios Skarlatidis, Alexander Artikis, Jason Filipou, and Georgios Paliouras. A probabilistic logic programming event calculus. *TPLP*, 15(2):213–245, 2015.