An Adaptive Semantic Stream Reasoning Framework for Deep Neural Networks

Danh Le-Phuoc[a], Thomas Eiter[b]

[a] Technical University Berlin
[b] Technical University Vienna

Abstract

Driven by deep neural networks (DNN), the recent development of computer vision makes visual sensors such as stereo cameras and Lidars ubiquitous in autonomous cars, robotics and traffic monitoring. However, due to operational constraints, a processing pipeline like object tracking has to hard-wire an engineered set of DNN models to a fixed processing logic. To overcome this, we propose a novel semantic reasoning approach that uses stream reasoning programs for in-cooperating DNN models with commonsense and domain knowledge using Answer Set Programming (ASP). This approach enables us to realize a reasoning framework that can adaptively reconfigure the reasoning plan in each execution step of incoming stream data.

Keywords

Semantic Reasoning, Neural-symoblic, Stream Reasoning, Stream Processing

1. Motivation

The recent development of computer vision (CV) driven by deep neural networks (DNN) makes visual sensors such as stereo cameras and Lidars ubiquitous in autonomous cars, robotics and traffic monitoring. In particular, many DNN models for object detection [1] and tracking [2] are available. However making reliably working in a real-life online processing pipeline such as an Automated Driving System (ADS) or a traffic surveillance query engine is still very challenging. For example, [2] reports that the most accurate DNN-driven multi-object tracking (MOT) pipelines can process only 4-5 frames/second. To make such a system work online e.g. for ADS, where processing delay must be less than 100ms [3], one has to hard-wire a fixed sets of DNN models with some sacrifices on accuracy and robustness as the design constraints of an ADS limit how much hardware can be put into a system [3]. For instance, an additional 400 W power consumption translates to a 3.23% reduction in miles per gallon for a 2017 Audi A4 sedan or similarly, the additional power consumption will reduce the total driving range of electric vehicles.

Such a design-time trade-off often leads to unpredictable fatal errors in a real deployment. For example, the recent report of Uber's accident in Arizona [4] says *"...The ADS detected the pedestrian 5.6 seconds before impact. Although the ADS continued to track the pedestrian until the crash, it never accurately classified her as a pedestrian or predicted her path. By the time the ADS determined that a collision was imminent (1 second before impact), the situation exceeded the response specifications of the ADS braking system..."*. This accident could have not happened if the ADS could reconfigure the object tracking pipeline on the fly, e.g changing DNN models or using alternative sensor sources to improve the accuracy on detection and tracking.

This motivates us to propose an approach of combining stream reasoning with probabilistic inference to continuously configure such processing pipelines based in semantic information representing commonsense and domain knowledge. The use of semantic information together with DNNs has proved to be useful and led to better accuracy in image understanding [5] and in object tracking [6]. Similar to ours, these approaches use declarative approaches to represent the processing pipelines of visual data. However, none of them have considered how to deal with the aforementioned operational constraints in the context of stream processing. Our approach represents such constraints in an extension of Answer Set Programming (ASP). This extension is proposed by leveraging LARS formulas [7] for expressing stream reasoning programmes to incorporate uncertainty of probabilistic inference operations under weighted rules similar to LP^{MLN} [8], called *semantic reasoning rules*. As a result, we will be able to dynamically express a visual sensor fusion pipeline, e.g. MOT over multiple cameras, by seman-

Proceedings of the CIKM 2020 Workshops, October 19-20, Galway, Ireland.
EMAIL: danh.lephuoc@tu-berlin.de (D. Le-Phuoc); thomas.eiter@tuwien.ac.at (T. Eiter)
ORCID: 0000-0003-2480-9261 (D. Le-Phuoc); 0000-0001-6003-6345 (T. Eiter)

tic reasoning rules to fuse probabilistic inference operations with ASP-based solving processes. Moreover, the expressive power of our approach also enables us to express operational constraints together with optimisation goals as a probabilistic planning programme similar to pBC^+ [9] so that our reasoning framework can reconfigure the reasoning plan adaptively in each execution step of incoming stream data.

2. Formalization of Semantic Stream Reasoning with DNN Models

Semantically, a multi-sensor fusion data pipeline will consume the data that is observed by a *Sensor* as a stream of observations (represented as an *Observation*) following standardized W3C/OGC Semantic Sensor Network Ontology (SSN) [10] . For instance, the example in Figure 1 shows 3 image frames are observed by a traffic camera. These observations will then be fed into a probabilistic inference process such as a DNN model or a CV algorithm (represented as a *Procedure*) to provide derived stream elements which then are representing *Sampling* instances. In this example, we have $det(yl, b_1, car)$ representing for the output bounding box b_1 from the YOLO detector $yl \in DT$ where DT (short for *Detector*) represents for the set of detectors supported. Similarly, $trk(39, b_2)$ represents for tracking bounding box b_2 generated by a tracking algorithm (a *Tracker*) which associates b_2 with the tracklet 39 via the popular object tracking algorithm SORT [11].

The symbol *FeatureOfInterest* (*FoI*) is used to represent the domain of physical objects which are subjects for the sensor observations, e.g. tracking objects and field of views (FoV) of the camera. The relationship between a *Result* generated by probabilistic inference algorithms (e.g. YOLO detection model or Kalman filter algorithm) to such object is represented by the predicate *isSampleOf* (denoted as *iSO*). As such algorithms have output with uncertainty, we will use an abduction reasoning process to search for explainable evidences for *iSO* via rules driven commonsense and domain knowledge similar to [6].

To formalise the reasoning process with such a semantic representation of stream data, we need a temporal model that allows us to reason about the properties and features of objects from streams of sensor observations. This model must account for the laws of the physical world movement and in particular be able to fill gaps of incomplete information (e.g., if we do not see objects appearing in observations, or camera reads are missing), based on commonsense principles. We thus use the LARS framework [7] to represent a reasoning programme over our semantic stream data which can evaluated using an ASP solver.

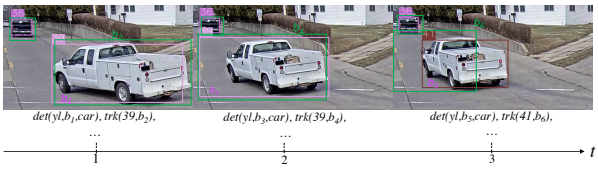

Figure 1: A Semantic Visual Stream Snapshot

The LARS framework provides formulas with Boolean connectives and temporal operators $@_\tau\alpha$, $\Box\alpha$, and $\Diamond\alpha$ to evaluate a formula α at a time point τ (resp. every, some time point) in the current stream S; window operators $\boxplus^w\alpha$ take for evaluating α a data snapshot (substream) S' from S by applying a function w on S. For example, the formula $\boxplus^{+5}\Diamond det(yl, B, car)$ states that at evaluation time τ, $det(yl, B, car)$ holds at some time point τ' in the window $[\tau - 5, .., \tau]$ selected by $w = +5$; in our representation, $det(yl, B, car)$ is the matching condition for *"a car was detected in bounding box B by the YOLO detector"*. The formula $@_\tau\alpha$ is aligned with a fluent F resp. an event E in Event Calculus (EC) [12], such that $@_\tau F \equiv holdsAt(F, \tau)$ and $@_\tau E \equiv happens(E, \tau)$. This will help us to employ ASP-based EC axioms for common sense reasoning rules as proposed in [6].

To deal with uncertainty, we extend LARS with weighted rules under LP^{MLN} semantics as in [8]. In LP^{MLN}, facts generated from feature extractors (DNN models or OpenCV algorithms) as well as abduction rules on top can be annotated with uncertainty information given by a weight, which allows reasoning under certain levels of uncertainty.

For our concerns, a *semantic reasoning program* Π is a set of *weighted rules* r of the form

$$\omega : \alpha \longleftarrow \beta \tag{1}$$

where α, β are LARS formulas and $\omega \in \mathbb{R} \cup \{x\}$ is the *weight* of the rule. If $\omega = x$, then r is a *hard rule*, otherwise a *soft rule*; by $\Pi^{\mathbb{H}}$ and $\Pi^{\$}$ we denote the sets of hard and soft rules of Π, respectively. The semantics of Π is given by the answer streams [7] S of the LARS program Π_S obtained from Π by dropping the weights and each r where S violates $\alpha \longleftarrow \beta$; each such S gets a probability $Pr_\Pi(S)$ assigned calculated from the weights of the rules retained for Π_S; for more information, we refer to [8]. In Section 3 we will address how to translate restricted programs Π into ASP

programs to be fed into an ASP solver, and how the weights ω can be learned from training data.

To demonstrate how to build a semantic reasoning program, we will emulate DeepSORT tracking algorithm [13] via soft rules that can search for supporting evidences to *re-identify* objects associated with tracklets created by Kalman filter above, by using visual appearance associations. DeepSORT extends SORT with a DNN that is trained to discriminate targeted objects (e.g. pedestrians or vehicles) on a labelled re-identification dataset.

Hence, we will search for pairs of bounding boxes of two similar tracklets w.r.t. visual appearance. Due to a large search space of possible matches, we will limit the search space by filtering the candidates based on their temporal and spatial properties. Therefore, we use rules with windows to reason about two disconnected tracklets that have two bounding boxes matched within a time window of δ^M time points which are aligned with the DeepSORT's gallery of associated appearance descriptors for each track. Based on this gallery of previous tracked boxes, the cosine distance is computed to compare appearance information that are particularly useful to recover identities after long term occlusions, when motion is less discriminative. Hence, for merging two adjacent tracklets that have visual appearance matches, we use the parametrized soft rule (15) below. The pair of parameters (δ^M, vM^j) has to be specified for each reasoning step via the probabilistic planning component of our dynamic reasoning framework in Section 3; vM^j is one of the available visual matching models that represent the association metrics to discriminate comparing bounding boxes.

$$\begin{aligned}\omega_1^j : iSO(B_1, O) \leftarrow\ & @_\tau trk(T_1, B_1), col(B_1, T_2), trklet(T_2, O),\\ & @_{\tau_e} ends(T_2), @_{\tau_e} \boxplus^{+\delta^M} \Diamond trk(T_2, B_2),\\ & \tau < \tau_e + 3, vMatch(vM^j, B_1, B_2) \qquad (2)\end{aligned}$$

Similarly, we can define rules to trigger the object matching search based on visual appearances of two tracklets from two adjacent cameras. We use $@_\tau dist(B_1, C, d^t)$ to specify the time difference d^t from the candidate camera C at time point τ to start the search for the matches via the auxiliary predicate $dist$. Also, C is filtered by the auxiliary predicate $next$ stating C is adjacent to the camera where B_1 was generated. $leftFoV(O, C)$ represents for "object O left the FoV of camera C".

$$\begin{aligned}\omega_2^j : iSO(B_1, O) \leftarrow\ & @_\tau trk(T_1, B_1), next(B_1, C), \qquad (3)\\ & @_\tau dist(B_1, C, d^t), @_{\tau+d^t} \boxplus^{+\delta^M} \Diamond leftFoV(O, C),\\ & trklet(T_2, O), trk(T_2, B_2), vMatch(vM^j, B_1, B_2)\end{aligned}$$

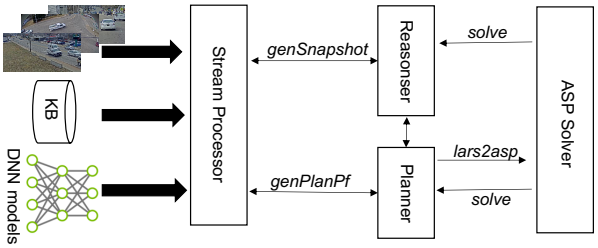

Figure 2: Stream Reasoning Framework

3. Dynamic Reasoning Framework

To the realize our reasoning approach in Secion 2, we proposes a dynamic reasoning framework illustrated in Figure 2. The key components *Reasoner* and *Planner* of the framework are built on top an *ASP Solver* and a *Stream Processor* which are pluggable and generic modules. The control logic of the framework is governed by Algorithm 1.

While any ASP Solver supporting weak constraints can be used in our framework, existing stream processors such as relational or graph stream processing engines need to be extended with some prerequisite features to connect with the rest of the framework. For instance, in our under-development prototype, we extend CQELS [14] to enable DNN inference on GPUs as built-in functions for CQELS-QL, the graph-based continuous query language of CQELS. Via CQELS-QL, the auxiliary predicates such as col, $next$ and $dist$ in Section 2 are expressed as continuous queries in order to delegate the processing to the stream processor. This mechanism also helps us to avoid grounding overhead in continuous solving via materialised views similar to the over-grounding approach in [15]. In particular, we leverage continuous multiway-joins with windows of CQELS to delegate the processing of LARS formulas that do not occur in rule heads.

Moreover, the visual stream data with different formats (e.g. RGB videos and Lidar PointCloud) together with the knowledge base (KB) have to be normalized to the data model supported by the stream processor. For example, ontologies and metadata and extracted symbols as outputs of DNN inference processes are represented as temporal graphs of CQELS. With these features, the stream processor is able to generate data snapshots and planning profiles in ASP readable format via two methods $genPlanPf$ and $genSnapshot$ respectively.

The *Planner* calls the method $genPlanPf$ to prepare the input for the first reasoning step of each time point

Algorithm 1 Semantic_Reasoning(S, t)

Input: Semantic Stream S, new observation $\mathbb{O}$, time point t
Output: Optimal answer set I^*

1: $i \leftarrow 0, \hat{\Pi^{\$}} \leftarrow \emptyset$
2: **for** $\{\omega : \alpha \leftarrow \beta\} \in \Pi^{\$}$ **do**
3: $i \leftarrow i + 1$
4: $\hat{\Pi^{\$}} \leftarrow \hat{\Pi^{\$}} \cup \{unsat(i) \leftarrow \beta, not\ \alpha\}$
5: $\hat{\Pi^{\$}} \leftarrow \hat{\Pi^{\$}} \cup \{\alpha \leftarrow \beta, not\ unsat(i)\}$
6: $\hat{\Pi^{\$}} \leftarrow \hat{\Pi^{\$}} \cup \{:\sim unsat(i)\ [\omega@0]\}$
7: **end for**
8: $\Pi^p \leftarrow genPlanPf(S, \mathbb{O}, \hat{\Pi^{\$}}, t-1)$
9: $\hat{\Pi^p} \leftarrow lars2asp(\hat{\Pi^{\$}} \cup \Pi^{\mathrm{H}} \cup \Pi^p, t)$
10: $I^p \leftarrow \mathrm{Solve}(\hat{\Pi^p})$
11: $\Pi^D \leftarrow genSnapshot(S, I^p, t)$
12: $\hat{\Pi} \leftarrow lars2asp(\hat{\Pi^{\$}} \cup \Pi^{\mathrm{H}} \cup \Pi^D, t)$
13: $I^* \leftarrow \mathrm{Solve}(\hat{\Pi})$
14: **return** I^*

τ which finds the optimal reasoning plan to achieve a certain goal Π^G under an operational constraint Π^p, following a probabilistic planning approach from [9] in lines (8) to (10) of Algorithm 1. The reasoning problem formalised in following is to find a configuration (dt^i, vM^j, δ^M) to result the highest probability of our tracking goal. For example, we can specify the goal of being able to track the objects that were tracked in the previous time point $\tau - 1$ as

$$\Pi^G = \bigwedge_{O \in @_{\tau-1} trklet(O,\_)} @_\tau trklet(O, \_)$$

Similarly, an operational constraint Π^p can be expressed ASP rules. For instance, the example rule below represents the constraint to limit executable plans $plan(D, V, \delta)$ at time point τ based on the estimations the execution time of a candidate detection model D and visual a matching model V together a candidate window parameter δ^M. The auxiliary predicates *est* and *nObj* provide the time estimation for corresponding DNN operations and the number of objects tracked in camera C.

$$\begin{aligned} :\sim\ & @_\tau est(D, \tau_D), @_\tau est(V, \tau_V), @_\tau plan(D, V, \delta), \\ & @_{\tau-1} nObj(C, N), \tau_D + N * \delta^M * \tau_V > maxTime \end{aligned} \quad (4)$$

From Π^p, lines (8) to (10) carry out solving the reasoning problem formalised by formula (4). To generate the LARS program from the soft rules $\Pi^{\$}$, the algorithm rewrites $\Pi^{\$}$ into LARS formulas with weak constraints as $\Pi^{\$}$ in lines (1) to (7) by extending the similar algorithm for LP^{MLN} in [8]. Then, we use the incremental ASP encoding algorithm of Ticker [16] for rewriting LARS formulas to an ASP program using the method *lars2asp*, whose optimal models (answers sets) I^p correspond to the solutions of (5).

$$\underset{I^p : \hat{\Pi^p}(\tau) \Vdash I^p}{\mathrm{argmax}} \ Pr_{\hat{\Pi^p}}(\Pi^G | S, \tau, \Pi^p) \quad (5)$$

With a chosen plan embedded in some I^p, the *Reasoner* calls the method *genSnapshot* to generate the input data of the reasoning program for the next step in line (11) to carry out the second reasoning step from line (12) to (14) to generate the output of the whole pipeline as the optimal model I^*. To specify the weights of the soft rules, we use the weight learning approach of [17] which fits the weights using the training data via gradient ascent. Training is done offline but uses Algorithm 1 to compute an optimal stable model in each step of updating weights in the gradient method.

4. Conclusion

This position paper presented a novel semantic reasoning approach that enables probabilistic planning to adaptively optimize the sensor fusion pipeline under operational constraints expressed in ASP. The approach is realised with a dynamic reasoning mechanism that can integrate the uncertainy of DNN inference with semantic information, e.g common sense and domain knowledge in conjunction with runtime information as inputs for operational constraints.

We are currently implementing an open sourced prototype of the proposed reasoning framework in Java to exploit the code bases of CQELS and Ticker. We use the Java native interface to wrap C/C++ libaries of Clingo 5.4.0 as the ASP Solver and NVidia CUDA 10.2 as the DNN Inference engine. The solving and inference tasks are coordinated in an asynchronous multithreading fashion to exploit the massive parallel capabilities of CPUs and GPUs.

Acknowledgments

This work was funded in part by the German Ministry for Education and Research as BIFOLD - Berlin Institute for the Foundations of Learning and Data (refs 01IS18025A and 01IS18037A) and supported by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under the program ICT of the Future (FFG-PNr.: 861263, project DynaCon).

References

[1] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, M. Pietikäinen, Deep learning for generic object detection: A survey, IJCV (2019).

[2] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, F. Herrera, Deep learning in video multi-object tracking: A survey, Neurocomputing (2019). URL: http://www.sciencedirect.com/science/article/pii/S0925231219315966. doi:https://doi.org/10.1016/j.neucom.2019.11.023.

[3] S.-C. Lin, Y. Zhang, C.-H. Hsu, M. Skach, M. E. Haque, L. Tang, J. Mars, The architectural implications of autonomous driving: Constraints and acceleration, in: ASPLOS '18, 2018.

[4] NTSB, Collision between vehicle controlled by developmental automated driving system and pedestrian in Tempe, Arizona, https://www.ntsb.gov/news/events/Documents/2019-HWY18MH010-BMG-abstract.pdf, 2019. Accessed: 2020-01-15.

[5] S. Aditya, Y. Yang, C. Baral, Integrating knowledge and reasoning in image understanding, in: IJCAI, 2019. URL: https://doi.org/10.24963/ijcai.2019/873. doi:10.24963/ijcai.2019/873.

[6] J. Suchan, M. Bhatt, S. Varadarajan, Out of sight but not out of mind: An answer set programming based online abduction framework for visual sensemaking in autonomous driving, in: IJCAI-19, 2019. URL: https://doi.org/10.24963/ijcai.2019/260. doi:10.24963/ijcai.2019/260.

[7] H. Beck, M. Dao-Tran, T. Eiter, LARS: A logic-based framework for analytic reasoning over streams, Artif. Intell. 261 (2018) 16–70. URL: https://doi.org/10.1016/j.artint.2018.04.003. doi:10.1016/j.artint.2018.04.003.

[8] J. Lee, Z. Yang, LPMLN, Weak Constraints, and P-log, in: AAAI, 2017.

[9] J. Lee, Y. Wang, A probabilistic extension of action language BC$^+$, TPLP 18 (2018) 607–622. URL: https://doi.org/10.1017/S1471068418000303. doi:10.1017/S1471068418000303.

[10] K. Janowicz, A. Haller, S. J. D. Cox, D. L. Phuoc, M. Lefrançois, SOSA: A lightweight ontology for sensors, observations, samples, and actuators, J. Web Semant. 56 (2019) 1–10.

[11] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: ICIP, 2016, pp. 3464–3468.

[12] E. T. Mueller, Commonsense Reasoning: An Event Calculus Based Approach, 2 ed., 2015.

[13] N. Wojke, A. Bewley, D. Paulus, Simple online and realtime tracking with a deep association metric, in: ICIP, 2017.

[14] D. Le-Phuoc, M. Dao-Tran, J. X. Parreira, M. Hauswirth, A native and adaptive approach for unified processing of linked streams and linked data, in: ISWC, 2011, pp. 370–388.

[15] F. Calimeri, G. Ianni, F. Pacenza, S. Perri, J. Zangari, Incremental answer set programming with overgrounding, TPLP 19 (2019).

[16] H. Beck, T. Eiter, C. Folie, Ticker: A system for incremental asp-based stream reasoning, TPLP (2017). URL: https://doi.org/10.1017/S1471068417000370. doi:10.1017/S1471068417000370.

[17] J. Lee, Y. Wang, Weight learning in a probabilistic extension of answer set programs, in: KR, 2018, pp. 22–31.