EasyChair Preprint

№ 5465

Towards Explainable Agency in Multi-Agents Systems Using Inductive Logic Programming and Answer Set Programming

Minal Suresh Patil and Kary Främling

May 4, 2021

Towards Explainable Agency in Multi-Agent Systems Using Inductive Logic Programming and Answer Set Programming

Minal Suresh Patil[1][0000−0003−0026−5503] and Kary Främling[1][0000−0002−8078−5172]

Computer Science Department, Umeå universitet, Sweden
{minal.patil,kary.framling}@umu.se

Abstract. Logical reasoning is a fundamental aspect of human behaviour, and this is an important criteria to build human-like reasoning in intelligent autonomous multi-agent systems. So far, the field of knowledge representation and reasoning has employed logic-based symbolic techniques to mimic the challenging task of incorporating human-like reasoning in multi-agent systems. However, the field of machine learning has shown increasing interest to take on this challenge. In this research, we describe a methodology which is based on Inductive Logic Programming and Answer Set Programming that enables autonomous agents to generate explanations and logic-based reasoning in a form of hypothesis from a rich knowledge base (ontologies). Whilst this preliminary work addresses key limitations such as scalability and adaptability, we strongly emphasise the need for logic-based reasoning in multi-agent systems for interpretability and transparency in their behaviour.

Keywords: Logic-based reasoning · Symbolic AI · Inductive Logic Programming · Answer Set Programming · Explainability · Multi-agent Systems

1 Introduction

1.1 Motivation

Approaches in Artificial Intelligence (AI) based on machine learning, and in particular those employing artificial neural networks, differ fundamentally from approaches that perform logical deduction and reasoning on knowledge bases. The first are connectionist or sub-symbolic techniques that are able to solve complex problems over unstructured data using supervised or unsupervised learning, including problems which cannot reasonably be understood by humans. Sub-symbolic methods are generally robust against noise in training data and with the rise of deep learning, they have shown to exceed human performance in tasks involving images, videos, sound and natural language. Symbolic systems, on the other hand, perform well in tasks that use highly structured data, including

multi-agent planning, constraint optimisation problems, data management, integration and querying, and other traditional application areas of expert systems and formal semantics. Classical rule-based systems, ontologies, and knowledge graphs that power search and information retrieval across the Web are a type of symbolic AI systems. Symbolic and sub-symbolic systems are almost always compatible to one and other. For example, the key strengths of sub-symbolic techniques are weaknesses of symbolic ones, and contrariwise, the strengths of symbolic techniques are weaknesses of sub-symbolic methods. Symbolic systems are fragile and to an extent inflexible; in the sense, they are susceptible to the noise present in the data whilst performing logical encoding of a problem, which stands in contrast to the robustness and flexibility of sub-symbolic approaches. But one of the major drawbacks of sub-symbolic systems are that they are essentially black-box systems, in other words, that the systems cannot be inspected in ways that provide insight into their actions and decisions (though some recent progress has been made) while symbolic knowledge bases can in theory be interpretable on how a system has arrived to a decision or an action it has taken all the way from input data. Most importantly, symbolic and sub-symbolic systems are complementary in the types of problems and data that they are designed to solve. For instance, semantic segmentation from visual data appears to be a problem that lies generally outside the capacity of symbolic systems, whilst complex logic-based planning problems appear to be outside the purview of current sub-symbolic techniques.

2 Related Work

2.1 Agents Reasoning in Multi-Agent Systems

One of the major goals of explainable AI techniques is the efficient mapping between explainability, interpretability and causal inference, which plays an important role in effective human-AI and multi-agent interaction. Explainability is the system's ability to explain itself in natural language to a human user by being able, for instance, to communicate: *This is the reason why you are seeing this output*. To be able to articulate the cause for an action taken or decision made by the system is essential to transparency in design of multi-agent systems (MAS). MAS are the richest providers of abstractions, technologies, and methodologies for complex intelligent systems. Agent interaction within MAS at its most fundamental level typically exploits agent communication languages (ACL), which are one of the oldest standard in the MAS field [12], and are shaped around Searle's theory [23] of human communication based on speech acts. ACL aims to provide inter-agent operability, by providing communicating agents with shared syntax and ontologies.

Logic-based programming, such as Prolog [2] and other related paradigms like Answer Set Programming [8] has been used by many researchers to implement machine reasoning. Logic-based approaches have a potential capability to model explainability and transparency in machines, in particular through non-monotonic logics [10]. This is due to the fact that ethicists represent ethical theo-

ries and predicaments in a declarative format. Ethicists employ formal and informal logic-based techniques to arrive at a possible solution. Since non-monotonic reasoning deals with the problem of deriving plausible conclusions, but not infallible, from a knowledge base (a set of formulas),this puts forward non-monotonic logic, which simulate commonsense reasoning, as appropriate techniques to formalise generating a set of possible explanation and reasons for MAS.

Inductive Logic Programming (ILP) algorithms [13] are a subclass of machine learning algorithms aimed at learning logic-based programs. ILP usually requires an optimal amount of training or input examples as opposed to connectionist techniques which require massive amounts of input examples. ILP is capable of producing interpretable and explainable results, in other words, it produces a set of rules which can be analysed and adjusted if necessary by humans. Thus, ILP appears to be a suitable and favourable technique for executing and accomplishing explainability and transparency in MAS. Furthermore, in MAS, there is usually a dearth of training example and transparency of their interaction with humans and each other is imperative. ILP produces human-understandable logic-based rules (also known as inference rules) where the **head** of the rule *also known as conclusion* is inferred from the **body** of the rule, particularly, the body or *premises* causes the head *conclusion*. Interpretability and transparency of logic-based representations and reasoning is in fact one of their most acknowledged advantage.

3 Background

3.1 Answer Set Programming

Answer Set Programming (ASP) [1] is a declarative programming paradigm that can represent recursive definitions, defaults, causal relations,special forms of self-reference, and other language constructs that occur frequently in various non-mathematical domains, and are difficult to express in classical logic formalisms [5].ASP is based on the stable model(i.e., answer set) semantics of logic programs [3], and has roots in the research on non-monotonic logic theory. This research introduces key concepts and syntax in ASP that are fundamental to inter-agent explainability and transparency.

Answer Set Semantics

Definition 1 (Normal Logic Program). *A normal logic program is a set of rules (often called extended definite clauses) of the form:*

$$A \leftarrow L_1, L_2, ..., L_m (m \geq 0) \tag{1}$$

where A is an *atom* and each L_i is a *literal*. A literal can be positive or negative (in the form of *not* B) where B is an atom. A rule with no occurrences of *not* is known as *definite clause or Definite Logic Program.*

[1] Answer Set Programming is also known as Answer Set Prolog

Definition 2 (Herbrand Universe). *The Herbrand Universe for a normal logic program Π, represented as U_P, is a set of all ground terms constructed from functors and constants that appear in Π.*

Definition 3 (Herbrand Base). *A Herbrand Base for a normal logic program Π, represented as B_P, is a set of all ground atoms that are formed by the symbolic predicates in Π and terms in U_P.*

Definition 4 (Herbrand Interpretation). *A Herbrand Interpretation for a logic program Π, represented as HI_P, is a subset of B_P and all ground atoms in HI_P are true. [18]*

Definition 5 (Herbrand Model). *A Herbrand Model for a logic program Π is a Herbrand Interpretation HI_P that satisfies all the clauses in Π.*

Definition 6 (Minimal Herbrand Model). *For a Herbrand Model (H) and a logic program Pi,there exists a Minimal Herbrand model iff there exists no $H' \subset H$ which is also a Herbrand model of Π. [18]*

Definition 7 (Least Herbrand Model). *For definite logic program there exists a unique Minimal Herbrand Model, called the Least Herbrand Model, denoted by H_P.*

Definition 8 (Grounding). *The grounding of a logic program Π is the set of all ground rules r_θ where r is in Π and θ is the mapping from variables to ground terms in U_P. [24]*

Definition 9 (Reduct). *For a logic program Π and interpretation Σ, the reduct of Π^Σ can be constructed in the following way- remove all rules that has in the body the negation of an atom in Σ and then remove all negative literals from the body of the remaining rules. Thus, the reduct of Π is a ground definite program and has a single minimal model. [18]*

Definition 10 (Cautious Entailment). *An atom a is cautiously entailed by a normal logic program Π, denoted as $\Pi \models_c a$ if it is in all the answers sets of Π. [22]*

Definition 11 (Brave Entailment). *An atom a is bravely entailed by a normal logic program Π, denoted as $\Pi \models_b a$ if it is in at least one answer set of Π. [22]*

Answer Set Programming (ASP) Syntax An answer set of normal logic program P is a stable model defined by a set of rules, where each rule consists of literals, which are made up with an atom p or its default negation not p (negation as failure). Answer Set Programming (ASP) is a normal logic program with extensions: constraints, choice rules and optimisation statements. An answer set program is a collection of rules of the form:

$$H \leftarrow A_1, A_2, ..., A_m, not A_{m+1}, ..., A_n \tag{2}$$

A_i denotes a literal in logic theory. H is known as the *head* and $A_1, A_2, ..., A_m$, $notA_{m+1}, ..., A_n$ is known as the *body*. The literals of the body cannot be true at the same time. A *constraint* is when a rule has an empty head. Hence, the logic program can have many answer sets or no answer set at all. In equation (2) if $A_1, ..., Am$ is true and if $A_{m+1}, ..., An$ is false, then H will be true. A program Π representing a problem can have answer sets that constitute all the possible solutions. An *optimal set* represents an answer sets in the order of preferences in the form:

$$maximise[A_1 = W_1, ..., A_1 = W_n] \tag{3}$$

$$minimise[A_1 = W_1, ..., A_1 = W_n] \tag{4}$$

$A_1, ..., A_n$ and $W_1, ..., W_n$ represent literals and weights, respectively. This can be achieved using a ASP solver such as Clyngor. [2]

3.2 Inductive Logic Programming

Inductive Logic Programming (ILP) is a sub field of machine learning aimed at supervised inductive concept learning, and is the intersection between machine learning and logic programming [10]. The purpose of ILP is to inductively derive a hypothesis H that is a solution of a learning task, which covers all of the positive examples and none of the negative examples, given a hypothesis language for search space and cover relation [11]. ILP is based on learning from entailment, as shown:

$$B \wedge H \vdash E \tag{5}$$

where B represents background knowledge (ontologies), H represents a hypothesis space and E contains all of the positive examples, denoted E^+, and none of the negative examples, denoted E^-. The possible hypothesis space is constrained via a language bias that is specified by a series of mode declarations M. A mode declaration can be classified into head or body declaration is represented as *modeh(s)* and *modeh(b)* where s is known as *schema*. A schema is a ground literal that contains placemarkers. A *placemaker* is either $+type, -type, \#type$ which represent input, output and ground, respectively.

An advantage of ILP over statistical machine learning is that the hypothesis that an agent learns can be easily understood by a human, as it is expressed in first-order logic, making the learning process explainable. ILP has received a growing interest over the last two decades. ILP has many advantages over statistical machine learning approaches: the learned hypotheses can be easily expressed in human understandable language and explained to a human user, and it is possible to reason with the learned knowledge. By contrast, a limitation of ILP is scalability. There are usually thousands or more examples in many real-world examples. Scaling ILP tasks to cope with large examples is a challenging task [12].

[2] Python wrapper built around Clingo/Answer Set Programming

3.3 Logic Programming under Answer Set Programming Semantics

This research article introduces two ILP semantics that will be primarily used for reasoning and explainability in MAS- *Cautious Induction* and *Brave Induction [15].*

Definition 12 (Cautious Induction). *A cautious induction task (ILP_c) is a tuple $\langle B, S_M, E^+, E^- \rangle$, where B represents the background knowledge (ontologies, commonsense reasoning etc), S_M is a set of ASP rules, E^+ is a set of positive examples and E^- is a set of negative examples. A cautious inductive hypothesis $H \in \langle B, S_M, E^+, E^- \rangle$ exists for an answer set A iff $AS(B \cup H) \neq \emptyset$ and $\forall A \in AS(B \cup H), E^+ \subseteq A$ and $E^- \cap A = \emptyset$.*

Note: One needs to be aware that positive examples must be true for all answer sets and negative examples must not be present in any of the answer sets. This constraint can sometimes be too *strict* since positive examples can be true for a few answer sets but not all answer sets. A *fuzzy* cautious induction is a research area to be explored.

Definition 13 (Brave Induction). *A brave induction task (ILP_b) is a tuple $\langle B, S_M, E^+, E^- \rangle$, where B represents the background knowledge (ontologies, commonsense etc), S_M is a set of ASP rules, E^+ is a set of positive examples and E^- is a set of negative examples. A brave inductive hypothesis $H \in \langle B, S_M, E^+, E^- \rangle$ exists for an answer set A iff $\exists A \in AS(B \cup H)$ such that $E^+ \subseteq A$ and $E^- \cap A = \emptyset$.*

Note: One drawback of brave induction is it can only reason about what is true in at least one answer set of a logic program and is incapable of learning constraints. In other words, a brave induction solution for a particular logic problem that includes a constraint will still be a solution to the program if the constraint is eliminated. Thus, the brave induction technique excludes exploring for constraints when learning an explanation i.e. a solution.

4 Multi-Agent Reasoning and Explainability With Inductive Learning of Answer Set Programs (ILASP)

In order to learn more complex tasks, Learning from Answer Sets (LAS) [17] was developed in which neither Cautious Induction nor Brave Induction could learn constraints. The motive is to classify the class of ASP programs that a framework is capable of learning, if given sufficient input examples. Language biases tend, in general, to impose their own restrictions on the classes of program that can be learned and are usually used to aid the computational performance, rather than to capture intrinsic properties of a learning framework. We formally define key concepts we will employ in our design of explainable and reasoning agency for MAS: *Partial Interpretation [20]*, *Learning from Answer Sets [14]* and *Context Dependent Learning. [16]*

Definition 14 (Partial Interpretation). *A partial interpretation PI_e is a pair of atoms $\langle e^{inc}, e^{exc} \rangle$, where e^{inc} and e^{exc} are known as inclusion and exclusion respectively. An interpretation Σ is said to extend PI_e iff $e^{inc} \subseteq \Sigma$ and $e^{exc} \cap \Sigma = \emptyset$.*

Definition 15 (Learning from Answer Sets (LAS)). *A Learning from Answer Sets (LAS) is a tuple $T = \langle B, S_M, E^+, E^- \rangle$ where B is the background knowledge (ontologies, commonsense etc), S_M is the search space defined by a language bias M, E^+ and E^- are sets of partial positive examples interpretations and partial negative examples interpretation,respectively. An hypothesis H is an inductive solution of T (denoted: $H \in ILP_{LAS}(T)$) iff $H \subseteq S_M$, $\forall e^+ \in E^+ \exists AS(B \cup H)$ such as A extends e^+ and $\forall e^- \in E^- \exists AS(B \cup H)$ such as A extends e^-.*

Note: In the above definition, the positive examples has to be bravely entailed and the negation of each negative example must be cautiously entailed.

Definition 16 (Context-dependent Learning from Answer Sets). *Context-dependent examples (denoted as $ILP_{LAS}^{Context}$) are examples allow each example to have it's own extra background knowledge from the environment, which applies only to each specific example known as context. This way the background knowledge is more structured rather than one fixed background knowledge that is applied to all examples in the system.*
A context-dependent partial interpretation is defined as $\langle PI_e, C \rangle$ where PI_e and C is the partial interpretation and Context, respectively. An $ILP_{LAS}^{Context}$ task is a tuple $T = \langle B, S_M, E^+, E^- \rangle$ where B is the background knowledge (ontologies, commonsense etc), S_M is the search space defined by a language bias M, E^+ and E^- are sets of context-dependent partial positive interpretation and negative context-dependent partial negative interpretation examples,respectively. An hypothesis H is an inductive solution of T iff:
1. $H \subseteq S_M$
2. $\forall \langle PI_e, C \rangle \in E^+, \exists A \in AS(B \cup C \cup H)$ *such that* A *extends* PI_e
3. $\forall \langle PI_e, C \rangle \in E^-, \nexists A \in AS(B \cup C \cup H)$ *such that* A *extends* PI_e

Note: Context dependent examples aims to provide a more rich and structured background knowledge in order to improve the efficacy of the learning algorithm.

5 Discussion

Since this is an on-going research, we aim to demonstrate our methodology for explainability and reasoning in multi-agent systems as a workshop using Inductive Logic Programming, Answer Set Programming and particularly Inductive Learning of Answer Set Program and Context-dependent Learning from Answer Sets. Logical interpretations are interpretable to the human user since we employ logic-reasoning in our day-to-day lives to make decisions. On the other

hand, black-box models have many drawbacks: poor generalisability, opaque in nature, which makes the evaluation of trust-worthiness of inter-agent behaviour a major issue. ILP and ASP based explainability for MAS is advantageous due to the following:

- ILP and ASP can inherently learn complex relational theories due to the expressiveness of logic programs and by utilising a rich knowledge-base of based on domain-specific ontologies.
- It utilises the concept of learning bias from the rich knowledge base and ontological representation of the data even when the input data is scarce.
- The hypothesis formed using ILP and ASP are interpretable since they are logic-based hypothesis.
- Transfer Learning is possible in ILP and ASP since it learns state transition in the form of hypothesis, which can be applied to similar but different domain.

Finally, a major challenge to all these methods is how to increase the scalability of existing state of the art systems with respect to large hypothesis spaces. One possible direction is to provide mechanisms for constraining the hypothesis space using domain-specific knowledge. Some preliminary results have been proposed, where the notion of constraint-bias has been proposed and formalised as an additional input to a non-monotonic brave induction task [7]. Furthermore, agents can learn to provide preferred explanations to humans using preference learning. Preference learning is a machine learning research area that aids in the process of exploiting a set of specific features of an individual in an attempt to predict an individual's preferences [25]. Essentially, we aim to employ the agent's ranking of explanations of it's decisions and actions during the time of uncertainty [6].

6 State and Future Direction

A first prototype called Inductive Learning of Answer Set Programs for Multi-Agent Systems (ILASP-MAS) is implemented using the rule-based programming language Clingo and a Python wrapper Clyngor which shows promising results. The prototype implements a Reinforcement Learning (RL) setup of multi agents interacting with each other and the environment. RL algorithms do not make use of high-level abstraction reasoning, such as interpreting symbolic representations or causality. Furthermore, we aim to employ transfer learning to understand how agents generates explanations in a different by similar environment. An online available prototype is being planned and developed. We like to conclude by stating logic-based programming for explainability is here to stay since they are by default interpretable which can be used to develop explainable agency for MAS.

7 Acknowledgment

The work is partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artif. Intell. 101(1), 285–297 (1998)
2. Bratko I. Prolog programming for artificial intelligence. Pearson education; 2001.
3. Brewka G, Eiter T, Truszczyński M. Answer set programming at a glance. Communications of the ACM. 2011 Dec 1;54(12):92-103.
4. Dastani, M., Jacobs, N., Jonker, C.M., Treur, J.: Modeling user preferences and mediating agents in electronic commerce. In: Dignum, F., Sierra, C. (eds.) Agent Mediated Electronic Commerce. LNCS (LNAI), vol. 1991, pp. 163–193. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44682-6 10
5. Eiter T, Ianni G, Krennwallner T. Answer set programming: A primer. In Reasoning Web International Summer School 2009 Aug 30 (pp. 40-110). Springer, Berlin, Heidelberg.
6. Fürnkranz, J., Hüllermeier, E.: Pairwise preference learning and ranking. In: Lavrač, N., Gamberger, D., Blockeel, H., Todorovski, L. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 145–156. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39857-8 15
7. Furnkranz, J., Hullermeier, E.: D. athakravi, d. alrajeh, k. broda, a. russo, k. satoh:.In: Proceedings of ILP. pp. 16–32 (2014)
8. Gebser, M., Kaminski, R., Kaufmann, B. and Schaub, T., 2012. Answer set solving in practice. Synthesis lectures on artificial intelligence and machine learning, 6(3), pp.1-238.
9. Geisler, B., Ha, V., Haddawy, P.: Modeling user preferences via theory refinement. In: Proceedings of the 6th International Conference on Intelligent User Interfaces, pp. 87–90. ACM (2001)
10. Ganascia JG. Ethical system formalization using non-monotonic logics. InProceedings of the Annual Meeting of the Cognitive Science Society 2007 (Vol. 29, No. 29).
11. Horváth, T.: A model of user preference learning for content-based recommender systems. Comput. Inform. 28(4), 453–481 (2012)
12. Labrou Y, Finin T. Semantics for an agent communication language. In International Workshop on Agent Theories, Architectures, and Languages 1997 Jul 24 (pp. 209-214). Springer, Berlin, Heidelberg.
13. Lavrac N, Dzeroski S. Inductive Logic Programming. InWLP 1994 (pp. 146-160).
14. Law M, Russo A, Broda K. Inductive learning of answer set programs. In European Workshop on Logics in Artificial Intelligence 2014 Sep 24 (pp. 311-325). Springer, Cham.
15. M. Law, A. Russo, and K. Broda, "Inductive learning of answer set programs," European Conference on Logics in Artificial Intelligence (JELIA), vol. 2, no. Ray 2009, pp. 311–325, 2014.
16. Law M, Russo A, Broda K. Iterative learning of answer set programs from context dependent examples. arXiv preprint arXiv:1608.01946. 2016 Aug 5.

17. Law M, Russo A, Broda K. Inductive learning of answer set programs from noisy examples. arXiv preprint arXiv:1808.08441. 2018 Aug 25.
18. M. Gelfond and V. Lifschitz, "The stable model semantics for logic programming," 5th International Conf. of Symp. on Logic Programming, no. December 2014, pp. 1070– 1080, 1988
19. M. Sergot, "Minimal models and fixpoint semantics for definite logic programs," Lecture Notes: Knowledge Representation (C491), Department of Computing, Imperial College London, 2005.
20. Otero RP. Induction of stable models. InInternational Conference on Inductive Logic Programming 2001 Sep 9 (pp. 193-205). Springer, Berlin, Heidelberg.
21. R. Duncan Luce and Howard Raiffa.Games and Decisions: Introduction and Critical Survey. Wiley, New York, NY, 1957
22. Sakama, C., Inoue, K.: Brave induction: a logical framework for learning from incomplete information. Mach. Learn. 76(1), 3–35 (2009)
23. Searle JR, Kiefer F, Bierwisch M, editors. Speech act theory and pragmatics. Dordrecht: D. Reidel; 1980 Mar 31
24. V. Lifschitz, "What Is Answer Set Programming?.," AAAI, vol. 8, pp. 1594–1597, 2008.
25. Yannakakis GN, Maragoudakis M, Hallam J. Preference learning for cognitive modeling: a case study on entertainment preferences. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans. 2009 Sep 29;39(6):1165-75.