

An XAI Framework for Automated Allocation and Exploitation of Resources

Diego Marcia

Mathematics and Computer Science Department at Università degli Studi di Cagliari (UniCA), Italy

Abstract

I propose an explainable-AI framework to join high-level system specifications with the probabilistic output of Machine Learning algorithms via a formal reasoning tool. The main application guiding the planning of this research activity is an Intelligent System to manage a smart building environment while aiming at minimizing energy consumption. However, it can be generalised to allow the management of any system with well-defined sets of constraints and rules expressed in a formal language, which has to take in consideration transient user requirements expressed in Natural Language.

Keywords

Logic programming, probabilistic programming, machine learning, natural language interfaces

1. Introduction

Controlling a *smart building* environment requires having both a set of solid default behaviours, as well as the ability to adapt to contingent changes from the former, in line with the temporary needs of the users. Aim of my project is providing a two-level programming framework for such systems. At the lower level, the framework should provide planners with a concise language to define the expected components of the system, together with an objective function the system should try to optimise. Such optimisation might be subjected to both general constraints defined at planning time, and to transient constraints defined at run-time by the users. The handling of the latter constitutes the higher level part of the framework, where users with no technical expertise should be able to define short- to long-term behaviours the system is expected to abide to, by using a Natural Language Interface (NLI). Being subject to general constraints, the optimisation of the objective function might require planning the availability of resources (in a smart building, these constraints can express concepts like “hot water should always be available”). For this reason, the system is expected to be able to schedule several actions on its components, together with forecasting the conditions it will most likely operate in the short term. This requires the use of Machine Learning (ML) algorithms and techniques, which allow to anticipate both the use of the available resources by the users, and the external (environmental) conditions of operation. While Neural Networks are a current hot topic in similar tasks carried under such uncertain conditions, they tend to act as an opaque layer between users and the

SEBD 2022: The 30th Italian Symposium on Advanced Database Systems, June 19-22, 2022, Tirrenia (PI), Italy

✉ diego.marcia@unica.it (D. Marcia)

🌐 diegomarcia.github.io (D. Marcia)

🆔 0000-0003-2576-8836 (D. Marcia)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

governance of the system. For this reason, one of the objectives of this project, is avoiding the use of Neural Networks to carry the planning and managing of resources, in favour of highly explainable inferences. One of the options currently being taken into consideration for the architecture of this explainable-AI framework, is the use of Logical Programming for governing the behaviour of the system.

2. Overview of the framework

In its most general formulation, the framework operates on two levels: on the lower level, the framework is geared toward *Intelligent Systems* (IS) developers and architects, while the higher level is intended to make such ISs usable by end-users. However, it should be noted that domain experts taking part in the initial planning of the system's architecture might lack the expertise required for the use of developer tools, so the model of interaction exposed to end users might be re-purposed for use by such actors in the system definition phase.

The two levels have different focus. The components that make up the real-world system to be governed, most likely sensors and actuators in an automation project, will be defined in the lower level. A subset of these components will be used in defining an *Objective Function* (OF), which the IS will try to optimise. The definition of the Objective Function will most likely allow the use of constraints. To make a use-case example, the system architects of a Building Automation scenario would be interested in reducing the electric energy consumption, while ensuring hot water will always be available at the faucets. End users will most likely have little to no knowledge of the rules governing the lower layer, and will exclusively interact with the upper layer: to them, the IS exposes uniquely a NLI, or a similarly user-friendly interface. The users would require the system to deviate from the baseline behaviour, to satisfy their needs. Such requirements would be expressed in two forms: immediate commands, and scenarios. The former are commands whose execution require an immediate and often temporary change in the system configuration ("*Open the kitchen window*"), while the scenarios are commands to be repeated with a certain periodicity, or more generally, which are not bound in the immediate time ("*Turn the AC on on afternoons*").

It is possible that transient user requirements conflict with the satisfaction of the OF and its constraints. Two follow-ups are possible for handling this situation: either giving priority to the user needs, in other words taking the user input as an additional constraint, or initiate a NLI conversational session with the user instead, trying to understand if the conflict arose from a flawed interpretation of their input, or if they wish to deviate from the default behaviour, and to what extent.

Since the real-world system being managed has to deal with user preferences and expectations, as well as meet some pre-defined performance criteria, end users and system maintainers would both benefit from some degree of introspection by the IS. For this reason, the tools and techniques falling in the domain of explainable-AI will most likely be preferred in the IS development. Although deep neural networks can be used for the purpose of predicting user needs in home automation scenarios [1] [2], they usually lack the aforementioned introspection capabilities this project aims at achieving. One of the main objectives is indeed giving the users a full track of the steps that led to a certain autonomous decision from the IS, starting

from the end-user natural language input and the system architect optimisation objectives and constraints.

3. Planned techniques for the lower level

The core idea of the project, is the integration of Machine Learning techniques with logic programming in an hybrid XAI framework, with an emphasis geared toward the latter. With this setting in mind, the developers will be provided with a Logic Programming (LP) interface (most likely, a Prolog dialect) to express the OF and its constraints. Not only that, but the framework itself will be centred around a reasoning engine to forecast the future state of the components involved in the OF calculation (sensors, in a Building Automation scenario), whose role will be the intelligent allocation of resources to satisfy the requirements. The role of the ML techniques would be providing predictions on future states of the system, while all the reasoning based on these values would be carried via LP techniques and tools. The developers should in this phase provide for each component of the system, be it active or passive, the code lambdas to interact with their low-level primitives for input/output.

It should be noted that the topic of intelligent scheduling of appliances use has been already studied in recent years, and fully explainable methods for it have been proposed, even taking into consideration the negative cost of energy given by photovoltaic panels [3] or in the absence of absolute certainty over the user preferences [4], but there still is room for possible improvements, like the integration with predictive models of cost estimation and with the detection of user preferences, or the the improvement of scheduling flexibility.

Regarding the integration between ML techniques, or more generally probabilistic predictions of future conditions with a LP framework, one interesting setting worth exploring is DeepProbLog [5], which allows combining a probabilistic variant of ProLog [6] with the probabilistic outputs of tools such as neural networks.

The LP interface is not the only possible tool for developers and architects of the IS. the first of two alternative techniques being taken into consideration is the use of a structured DDL, which might allow an unambiguous enumeration and description of the system components without requiring the knowledge of a programming language. This DDL, still to be fixed, might also help in the concise expression of the OF constraints. Another alternative is represented by the use of an NLI, such as in the upper layer. The System architects, who might be expert in the problem domain, are not expected to also be experts in software developing techniques: their description of the system might therefore come in the form of semi-structured text written in a Natural Language. From this, the framework might extract key understandings of the components and the rules governing the planned IS, and generate code stubs for the developers. Such a pre-processing step might also be useful in identifying inconsistencies and contradictions in the requirements description from the architects, requiring their active disambiguation and/or emendation, or providing developers with attention flags.

4. Planned techniques for the upper (NLI) level

The upper level provides an interface to non-technical users of the IS. Therefore, it cannot provide for formal programming languages, and should focus on more natural interaction models. The most obvious choice would be a Natural Language Interface, possibly but not necessarily in the form of a speech recognition module, which gives —especially in its *conversational* interaction mode— a good compromise between the user’s spontaneous expression of their needs and the guidance required for compliance with the IS definition. Models that try to find a compromise between precision and user engagement have already been studied [7]. I expect the accepted language to be a rich form of Controlled Natural Language, which will not be designed with minimality in mind, so as to allow the user to express their inputs in more than one way.

This latter aspect is of particular interest. The IS components are enumerated and described in a formal language, with names and actions which are bounded to specific code lambdas. Therefore, it is a requirement that the framework’s reasoning engine operates on these specific names. However the users, not being aware of the low-level specification, might use different and ambiguous terms to refer to the System components. This requires the disambiguation of their input. A natural choice for this activity would be the use of word embeddings to find the most likely match between what the user asked to operate on and what is available to the inference engine. This has already been applied to different domains [8]. The conversational interaction would be effective in determining whether the user simply called the right object with the wrong name, or asked for an action the IS is not capable of carrying. The same applies to user requests which might conflict with the general behaviour defined by the System developers. It should be noted that these same techniques might be used for providing the System architects with a user-friendly interface for defining and amending their high-level specification [9].

5. Considerations on the expected milestones

At the time of writing, the PhD is in the fifth month of year one. The first implementation step would be exploring the feasibility of the NLI, in particular with respect to the definition of a first form of CNL and the use of word embeddings. The output of this layer would be an intermediate form between a Natural Language and the language to be used by the inference engine. I expect this phase to be over before the second half of Y2.

With the start of Y2, it should already be clear with language will be used in the inference phase, together with the specification language to be used by system developers. Therefore, by the first half of Y3, the lower level of the framework should be complete. The second half of Y3 would be devoted to integrating the two layers.

Acknowledgments

Diego Marcia’s PhD is funded via a NOP Research and Innovation 2014-2020 grant, Action IV.5 – PhD programmes on green topics <http://www.ponricerca.gov.it/opportunita/react-eu-phd-programmes-on-innovation-and-green-topics/>.

References

- [1] D. Popa, F. Pop, C. Serbanescu, A. Castiglione, Deep learning model for home automation and energy reduction in a smart home environment platform, *Neural Computing and Applications* 31 (2019) 1317 – 1337. doi:10.1007/s00521-018-3724-6.
- [2] M. Nasir, K. Muhammad, A. Ullah, J. Ahmad, S. Wook Baik, M. Sajjad, Enabling automation and edge intelligence over resource constraint iot devices for smart home, *Neurocomputing* 491 (2022) 494–506. doi:<https://doi.org/10.1016/j.neucom.2021.04.138>.
- [3] F. A. Qayyum, M. Naeem, A. S. Khwaja, A. Anpalagan, L. Guan, B. Venkatesh, Appliance scheduling optimization in smart home networks, *IEEE Access* 3 (2015) 2176–2190. doi:10.1109/ACCESS.2015.2496117.
- [4] V. Nguyen, W. Yeoh, T. C. Son, V. Kreinovich, T. Le, A scheduler for smart homes with probabilistic user preferences, in: M. Baldoni, M. Dastani, B. Liao, Y. Sakurai, R. Zalila Wenkstern (Eds.), *PRIMA 2019: Principles and Practice of Multi-Agent Systems*, Springer International Publishing, 2019, pp. 138–152. doi:10.1007/978-3-030-33792-6_9.
- [5] R. Manhaeve, S. Dumančić, A. Kimmig, T. Demeester, L. De Raedt, Neural probabilistic logic programming in deepproblog, *Artificial Intelligence* 298 (2021) 103504. URL: <https://www.sciencedirect.com/science/article/pii/S0004370221000552>. doi:<https://doi.org/10.1016/j.artint.2021.103504>.
- [6] L. De Raedt, A. Kimmig, Probabilistic (logic) programming concepts, *Machine Learning* 100 (2015) 5 – 47. doi:<https://doi.org/10.1007/s10994-015-5494-z>.
- [7] N. C. Truong, T. Baarslag, S. Ramchurn, L. Tran-Thanh, Interactive scheduling of appliance usage in the home, in: *25th International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2016, p. 7. URL: <http://eprints.soton.ac.uk/id/eprint/396670>.
- [8] A. Morales-Garzón, J. Gómez-Romero, M. J. Martín-Bautista, A word embedding-based method for unsupervised adaptation of cooking recipes, *IEEE Access* 9 (2021) 27389 – 27404. doi:10.1109/ACCESS.2021.3058559.
- [9] S. Mishra, A. Sharma, Automatic word embeddings-based glossary term extraction from large-sized software requirements, in: N. Madhavji, L. Pasquale, A. Ferrari, S. Gnesi (Eds.), *Requirements Engineering: Foundation for Software Quality*, Springer International Publishing, 2020, pp. 203 – 218. doi:10.1007/978-3-030-44429-7_15.