Neurosymbolic Integration of Linear Temporal Logic in Non Symbolic Domains*

Elena Umili

Sapienza University of Rome
umili@diag.uniroma1.it

Abstract. Linear Temporal Logic (LTL) is widely used to specify temporal relationships and dynamic constraints for autonomous agents. However, in order to be used in practice in real-world domains, this high-level knowledge must be *grounded* in the task domain and integrated with perception and learning modules that are intrinsically continuous and subsymbolic. In this short paper, I describe many ways to integrate formal symbolic knowledge in LTL in non-symbolic domains using deep-learning modules and neuro-symbolic techniques, and I discuss the results obtained in different kinds of applications, ranging from classification of complex data to DFA induction to non-Markovian Reinforcement Learning.

Keywords: Neurosymbolic AI · Linear Temporal Logic · Deep Learning.

1 Introduction

Linear Temporal Logic (LTL) [10] is a modal logic widely used in different domains, such as Robotics [7] and Business Process Management [5], for specifying temporal relationships, dynamic constraints, and performing automated reasoning. However, exploiting LTL knowledge in real-world applications can be difficult due to the knowledge's symbolic "crispy" nature. This short paper explores different techniques to relax the knowledge to make it applicable in continuous domains where symbols are grounded through Deep Learning modules and the symbol grounding function and/or the symbolic temporal specification can be unknown or partially known. In particular, we propose two different techniques: (i) one based on Logic Tensor Networks [2] [13] and (ii) one based on Probabilistic Finite Automaton [12]. We apply the first approach to classifying sequences of images, and we show that our approach requires less data and is less prone to overfitting than purely deep-learning-based methods. We use the second approach to learn DFA specifications from traces with gradient-based optimization, showing that it can learn larger automata and is more resilient to noise in the dataset than prior work. Finally, we propose an extension of our second approach [11] that we apply to non-Markovian Deep Reinforcement Learning problems [1]. This third contribution has shown to be more sample efficient of methods based on Recurrent Neural Networks (RNN), and, at the same time, it requires less prior knowledge than methods based on LTL, such as Reward Machines [3] and Restraining Bolts [4].

* This work is partially supported by the ERC Advanced Grant WhiteMech (No. 834228), by the EU ICT-48 2020 project TAILOR (No. 952215), by the PRIN project RIPER (No. 20203FFYLK) and by the PNRR MUR project PE0000013-FAIR.

2 Problem formulation

We consider the problem of integrating some symbolic background knowledge expressed as an LTLf formula ϕ in a non-symbolic environment producing at each run a sequence of images $I = i_0, i_1, ..i_{l-1}$ and some high-level label over the sequence. Each image in the sequence is the 'rendering' of a symbolic interpretation over the formula alphabet P. This means that there exists a function $sg : \mathcal{I} \rightarrow 2^P$, where $\mathcal{I}$ is the space of images, that maps each image into the truth values of symbols in P, we call this function *symbol grounding function*. We aim to exploit deep learning perception and symbolic reasoning in our system to leverage both subsymbolic data and symbolic knowledge.

3 Models

3.1 Recurrent Logic Tensor Networks

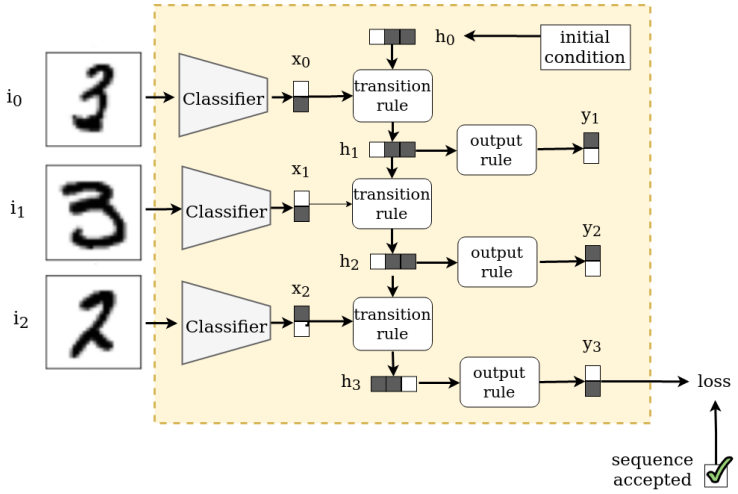

Fig. 1: Design of the Recurrent Logic Tensor Network used in [13].

Logic Tensor Networks (LTN) [2] are a neuro-symbolic framework that can reason and learn by exploiting both structured symbolic knowledge and raw data. It implements Real Logic, which is fuzzy relaxation of First Order Logic (FOL). Thanks to continuous logic, neural networks can co-exist in the logic framework and actually implement logic elements, grounding every atom in a real tensor.

LTN can be used for querying, reasoning, and learning: here we focus on learning. LTN can learn from both data and symbolic knowledge by imposing the knowledge available, and searching for the groundings that maximize the satisfiability of that

knowledge. This is done by defining a loss objective inverse to the given formula's satisfaction level and optimizing the system's trainable weights by back-propagation.

In our prior work [13], we use the same concept of *learning by best satisfiability*, but we apply it to the DFA generated by the LTLf formula. The neural computational graph implementing the automaton has therefore a *recurrent* structure, like a Long short-term memory (LSTM) neural network, and can be applied to sequences of any length. This feature is missing in the current implementation of LTN, and it is very convenient for imposing logic specifications that are extended in the time dimension.

Our framework is based on three fuzzy predicates: $Symbol$, $State$, and $Output$. The predicate $Symbol(p, t)$ denotes whether the t-th image in the sequence belongs to class p. We ground this predicate with a convolutional neural network, as shown in Figure 1. At any time t we are in a state q_k of the automaton, we encode this information with another fuzzy predicate $State$, where $State(q_k, t)$ is true if we are in state q_k at time t. Finally, the fuzzy predicate $Output$ represents the machine's output in a given time, denoting with $Output(o_i, t)$ whether the machine gives output o_i at time t. In particular, the output can be a symbol in the binary alphabet $\{Acc, Rej\}$ if our temporal specification is a DFA, or a symbol in the alphabet $\{o_0, o_1, ..., o_{N_O-1}\}$ in case the temporal specification is a Moore Machine.

We use these predicates to define a knowledge base (KB) composed of three axioms: (i) the initial condition, (ii) the transition rule, and (iii) the output rule. In particular, the initial condition only specifies the initial state and does not depend on the classifier predictions. The transition rule calculates the next state given the current automaton state and the symbol prediction over the current image. The output rule calculates the current output given the current state. These two rules are applied recursively as many times as many images compose the sequence.

By applying the rules in the KB, we can monitor the satisfaction of the formula ϕ during time, and, if we know some labels specifying which image sequences are accepted by the formula and which are not, we can impose this information defining a loss on the fuzzy automaton output.

3.2 Probabilistic relaxation of DFA: DeepDFA

In another prior work [12], we propose a different neural architecture based on Probabilistic Finite Automata (PFA). PFAs are easier to integrate with neural networks since we can calculate the probability that a sequence is accepted by applying matrix multiplications. In particular, we represent a PFA in matrix form as a *transition matrix* M_t, an *input vector* v_i and an *output vector* v_o. Given a string $x = x[0]x[1]...x[l-1]$, the probability that the string is accepted is calculated as follows.

$$v_i \times M_t[x[0]] \times M_t[x[1]] \times ... \times M_t[x[l-1]] \times v_o \tag{1}$$

In our work, we have designed a recurrent neural network with parameters including a transition matrix and an output vector, resembling the working of a PFA, that we call DeepDFA [12]. Since DFAs can also be represented in the same matrix form, the architecture can impose as background knowledge both DFA and PFA specifications. Differently from the framework presented in Section 3.1, this model can only be applied

to tasks where the symbols are assumed to be *mutually exclusive*, i.e., at each time step one and only one symbol is true, and the others are false. Another important difference between Recurrent LTN and DeepDFA is that the latter can also be employed to *learn* the DFA specification from traces, and not only to impose it as outside background knowledge. In particular, to learn a DFA from traces with DeepDFA, we have used a specific activation function that smoothly approximates one-hot vectors to drive the PFA to be a DFA during training while maintaining the differentiability of the model.

3.3 DeepDFA with probabilistic grounded symbols

Finally, we propose a slightly different model in [11] that extends DeepDFA to probabilistic grounded symbols. The latter adds to DeepDFA the calculation of the expectation value over the next DFA state using the symbol's probabilities at each time step. It is a more general framework applicable in non-symbolic environments, that we have texted in the context of non-Markovian Reinforcement Learning domains.

4 Applications

4.1 Exploiting LTL knowledge in image sequence classification

In prior work [13], we used the recurrent LTN architecture explained in Section 3.1 to increase the performance of a sequence classifier in visual tasks. In particular, we considered the task of classifying a sequence of images as compliant or not with a given formula, by exploiting the formula knowledge and a set of sequence-level labels expressing if the sequence of images is compliant or not with the formula. Note that we do not assume *any* knowledge of the symbol grounding function. Symbols are grounded in the images implicitly by our framework while it tends to maximize the conformance of the predicted DFA outputs with the sequence labels we have in the dataset. Compared with a purely deep-learning-based approach that cannot exploit the formula knowledge, our approach reaches higher accuracy, even if we decrease the number of samples in the dataset, showing that our way of embedding logical knowledge in the network is very effective.

4.2 Neural DFA induction from traces

In another work [12], we tested DeepDFA in learning DFA specifications from labeled sequences of images. Our approach has shown to be very effective in learning compact DFA from data by minimizing the binary cross-entropy loss between the model predictions and the labels. In particular, we compared DeepDFA with a classical combinatorial algorithm for DFA induction based on SAT [15], and we found that our framework can maintain high performances even with large target DFA and with a small percentage of errors in the training data, while the SAT-based approach performs very poorly in these cases. We also compared DeepDFA with another kind of *hybrid* method between RNNs and DFAs: L* extraction [14]. The latter consists of training an RNN on the same task and extracting an equivalent DFA from the RNN. We found that applying this method to

some complex languages can be tricky, since it can require training many different RNN architectures before finding the best one for the specific language. Instead, our method has only one hyperparameter, resulting in similar performances and a very much lighter fine-tuning.

4.3 Application to Reinforcement Learning: Visual Reward Machines

Non-markovian Reinforcement Learning (RL) tasks are arduous, because intelligent agents must consider the entire history of state-action pairs to act rationally in the environment. A common approach to this kind of task uses RNNs to preprocess experience data sequences and automatically extract a state representation for the RL algorithm [8] [6] [9]. However, there are no theoretical guarantees the resulting state representation will be Markovian. Another kind of approach, such as Reward Machines (RM) [3], uses LTL to specify the temporally-extended tasks and compose a Markovian state representation [4]. However, this approach requires prior knowledge of both the symbol grounding function mapping the environment observations in the specification's symbols and the temporal property. This limits the applicability of this approach in real-world domains. In a previous work [11], we defined Visual Reward Machines (VRM) as a neurosymbolic framework based on the model described in Section 3.3. VRMs compose the state representation as RMs, so as to have the same theoretical guarantees in the limit, and they are equivalent to RMs in case of complete knowledge of the task. However, VRMs are still applicable in the case of *missing knowledge* because they can integrate the available prior knowledge with the data they observe in the environment to learn the missing modules (the symbol grounding function and/or the DFA). Compared with methods based on RNNs, our approach reaches higher values of cumulative discounted rewards in visual non-symbolic tasks where RMs cannot be applied.

5 Conclusions

In conclusion, I described many prior works on integrating Linear Temporal Logic in non-symbolic (visual) domains, showing the advantage of relying on both prior structured knowledge and unstructured data acquired in the environment. We remark that future artificial systems should be able to acquire and integrate both these two sources of knowledge from human users and/or the environment, since this is a fundamental milestone of AI systems to achieve complex tasks, and this is the main objective behind our current and future research. In particular, many improvements on the described systems are still possible, such as, for example, integrating richer temporal formalisms such as Alternating-Time Temporal Logic and Signal Temporal Logic, which we let as future research.

References

1. Bacchus, F., Boutilier, C., Grove, A.: Rewarding behaviors. pp. 1160–1167. Portland, OR (1996), behaviors.pdf

2. Badreddine, S., d'Avila Garcez, A., Serafini, L., Spranger, M.: Logic tensor networks. Artificial Intelligence **303**, 103649 (2022). https://doi.org/https://doi.org/10.1016/j.artint.2021.103649, https://www.sciencedirect.com/science/article/pii/S0004370221002009
3. Camacho, A., Toro Icarte, R., Klassen, T.Q., Valenzano, R., McIlraith, S.A.: Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. pp. 6065–6073. International Joint Conferences on Artificial Intelligence Organization (7 2019). https://doi.org/10.24963/ijcai.2019/840, https://doi.org/10.24963/ijcai.2019/840
4. Giacomo, G.D., Iocchi, L., Favorito, M., Patrizi, F.: Foundations for restraining bolts: Reinforcement learning with ltlf/ldlf restraining specifications (2019)
5. Giacomo, G.D., Masellis, R.D., Grasso, M., Maggi, F.M., Montali, M.: Monitoring business metaconstraints based on ltl and ldl for finite traces. In: BPM (2014)
6. Ha, D., Schmidhuber, J.: Recurrent world models facilitate policy evolution. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf
7. He, K., Wells, A.M., Kavraki, L.E., Vardi, M.Y.: Efficient symbolic reactive synthesis for finite-horizon tasks. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 8993–8999 (2019). https://doi.org/10.1109/ICRA.2019.8794170
8. Heess, N., Hunt, J.J., Lillicrap, T.P., Silver, D.: Memory-based control with recurrent neural networks. CoRR **abs/1512.04455** (2015), http://arxiv.org/abs/1512.04455
9. Kapturowski, S., Ostrovski, G., Dabney, W., Quan, J., Munos, R.: Recurrent experience replay in distributed reinforcement learning. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=r1lyTjAqYX
10. Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977. pp. 46–57. IEEE Computer Society (1977). https://doi.org/10.1109/SFCS.1977.32, https://doi.org/10.1109/SFCS.1977.32
11. Umili, E., Argenziano, F., Barbin, A., Capobianco, R.: Visual reward machines. In: Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, La Certosa di Pontignano, Siena, Italy, July 3-5, 2023. pp. 255–267 (2023), https://ceur-ws.org/Vol-3432/paper23.pdf
12. Umili, E., Capobianco, R.: Deepdfa: a transparent neural network design for dfa induction (2023). https://doi.org/10.13140/RG.2.2.25449.98401
13. Umili, E., Capobianco, R., Giacomo, G.D.: Grounding ltlf specifications in images. In: Proceedings of the 16th International Workshop on Neural-Symbolic Learning and Reasoning as part of the 2nd International Joint Conference on Learning & Reasoning (IJCLR 2022), Cumberland Lodge, Windsor Great Park, UK, September 28-30, 2022. pp. 45–63 (2022), http://ceur-ws.org/Vol-3212/paper4.pdf
14. Weiss, G., Goldberg, Y., Yahav, E.: Extracting automata from recurrent neural networks using queries and counterexamples. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 5247–5256. PMLR (10–15 Jul 2018), https://proceedings.mlr.press/v80/weiss18a.html
15. Zakirzyanov, I., Morgado, A., Ignatiev, A., Ulyantsev, V.I., Marques-Silva, J.: Efficient symmetry breaking for sat-based minimum dfa inference. In: LATA (2019)