Learning Logics By Imitation: A Proposal

Cody Rosevear[1] [0000−0003−0886−9475]

University Of Alberta, Edmonton AB T6G 2E8, Canada rosevear@ualberta.ca

Abstract. Recent work in machine learning (ML) and reinforcement learning (RL) has yielded a number of breakthroughs in artificial intelligence (AI) that have led to significant improvements to the state-of-the-art in a number of application domains. However, there are strong arguments that despite these impressive achievements machine learning as generally practiced is insufficient to achieve artificial general intelligence in the long term, so long as it fails to incorporate the capacity for reasoning and abstraction. The following work articulates the arguments in favour of focusing on replicating reasoning and abstraction within machines, surveys current attempts to combine aspects of logic and machine learning within the context of interactive theorem proving and SAT solving, and proposes applying the technique of imitation learning to the problem of learning formal logics, in an effort to address the issue of reasoning and abstraction in machines.

Keywords: interactive theorem proving · imitation learning · machine learning · reinforcement learning · logic · SAT

1 Introduction

The field of AI has, since its inception, been fragmented into various competing paradigms concerning how to achieve the goal of general intelligence. The early days of the enterprise saw a division along the lines of those characterized as 'neats' versus 'scruffies' [13], where the former placed a strong emphasis on the importance of declarative knowledge representation and provably correct formal methods, while the latter preferred a more open ended and methodologically diverse approach that did not shy away from ad-hoc solutions and empirical demonstrations as the measure of success.

While this explicit characterization of the competing camps has in recent years been largely rendered obsolete, there nevertheless remains a degree of division concerning whether current methods that yield success with respect to specific applications can be scaled up to tackle the problem of general intelligence. In particular, the sub-fields of ML and RL have enjoyed a number of older theoretical breakthroughs which, when combined with the exponential increases in data and computing power that have occured in the last half-century, have yielded significant improvements to the state-of-the-art in such diverse application areas as facial recognition [17], autonomous aerial control [1], speech recognition [6], game playing (Poker, Go) [15], [11], and dynamic random access memory control, [12] among others.

Nevertheless, despite these successes, the question remains as to whether or not the statistical methodology of machine learning, broadly construed, can account for the capacities of reasoning and abstraction that are inarguably central to the concept of general intelligence. Indeed, there are in fact a number of arguments in computational psychology that suggest some type of logic that operates on structured, declarative knowledge representations is necessary to truly account for human level intelligence.

After providing a short summary of supervised machine learning and reinforcement learning in section 2 to provide the appropriate background, the following work articulates the arguments in favour of the importance of logics for cognition in section 3. This is followed by a small survey of related work that seeks to apply either ML or RL to the problems of interactive theorem proving and SAT solving. The notion of learning logics via the abstract framework of Sakama and Inoue [14] is then discussed in section 4, and the viability of applying the formalism of imitation learning as an implementation of the framework is analyzed. Finally, section 5 concludes the analysis and proposes future work.

2 Background

2.1 Supervised Machine Learning

The traditional supervised learning problem is essentially one of function approximation within the context of limited labeled data. Given a data set composed of inputs x and corresponding target outputs y, the goal is to infer a function $f(x)$ from the data, such that when provided with new, unlabeled data points $\bar{x}$ the corresponding outputs $\bar{y}$ will be predicted, with minimal error.

The source inputs x are encoded as vectors with values that represent the relevant 'features' upon which the prediction will be based. These features are weighted according to some initialization scheme, and then these weights, denoted by θ, are iteratively tuned throughout the learning process to yield the final functional approximation. The learning process itself is framed as an optimization problem, where a loss function $J(\theta)$ quantifies the discrepancy between the outputs predicted by the model and the actual expected outputs for a given input. This loss is minimzed by finding the optimal set of weights with respect to the loss function via gradient descent (or some variation thereof).

2.2 Reinforcement Learning

A traditional RL problem as described in Sutton & Barto [16] is composed of two parts: an agent, and an environment. The environment defines a set of states over which the agent can be said to inhabit at a given time step t, and for each such state there is a set of actions the agent can perform. Each state action pair (s_t, a_t) yields a corresponding real valued scalar reward, r_t, while transitioning to a new state s_{t+1} according to a (usually unknown) state transition function, which can be either deterministic or stochastic. The agent takes a new action

a_{t+1}, yielding a new reward r_{t+1}, and the cycle continues, either until termination in the case of an episodic task with terminal states, or indefinitely, in the case of a continuing task.

The agent chooses actions according to a policy $\pi(a \mid s)$, which defines the probability of selecting action a given state s. The return, denoted G_t, is defined as the sum of the discounted rewards since time t. That is, $G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_t$, where $0 \leq \gamma \leq 1$ is the discount factor and T is the final time step. T may be a natural number or, in the case of continuing tasks, be assigned the value of ∞, in which case γ must be < 1 to ensure that no returns diverge. The discount factor is used to specify how far-sighted the agent is, where 0 indicates a myopic agent concerned with maximizing the immediate reward and 1 specifies an agent that takes into account the full value of future states when deciding its actions.

The state-action value function $Q(s, a)$ is defined with respect to a particular policy π, such that it is equal to the expected return if one were to take action a in state s and follow policy π therafter. The goal of the agent is, roughly, to find a policy that maximizes the agent's expected return within the current environment. See figure 1 for a depiction of the agent-environment loop for reinforcement leanring.

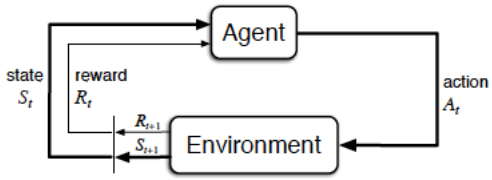

Fig. 1. The agent-environment loop for reinforcement learning. Source: [16]

3 The Problem Of Reasoning

Previous work in computational psychology has argued against the notion that machine learing and associated neural network models are sufficient to replicate the full spectrum of cognitive abilities as displayed by human agents [5]. In particular, they highlight a number of features of human cognition that seem to require structured mental representations which allow for formal reasoning over symbolic expressions, rather than simply the mapping of inputs to outputs predicated on tuning weights to pick up on statistical regularities in vast amounts of data. These features include the productivity and systematicity of thought, each of which are detailed below.

3.1 The Productivity Of Thought

It is clear that human agents are finite beings, with finite capacities for memory and computation. As such, they are bounded with respect to what types of computations they can perform, including those computations involved in using natural language. However, for any given particular thought or linguistic utterance, there is no reason to suppose that an individual will not be able to understand or produce that thought, provided they possess the prerequiste vocabulary of terms.

For example, consider the following sentence: "Galactic muffins are poisonous for purple giraffes with tiny hats who live on the other side of Mars". It is likely that there are some individuals for whom this sentence has never been said to before; however, any competent English speaker will be able to understand the sentence quite easily. This is also the case with respect to an infinite number of other sentences or thoughts, even though any individual will only ever speak or have spoken to them a subset of those thoughts/sentences that are possible. This capacity to understand an infinte number of sentences/thoughts is termed the *productivity* of thought, and it needs to be accounted for given that humans are finite beings, with finite resources, who are nevertheless seemingly capable of comprehending an infinite number of thoughts.

The only known way to understand an infite number of sentences using finite means is through the use of recursive rules defined over primitive elements. Indeed, formal logic routinely makes use of such rules in order to define infinite sets of well-formed-formulas with respect to a particular logical system; and this is done in only a few lines, requiring little in the way of memory or computation when implemented on a computer.

This suggests that in order to account for such productivity truly artificial general intelligence will require at least some type of recursively defined formal logic as part of its architecture.

3.2 The Systematicity Of Thought

Human cognition also displays the property of *systematicity*. This refers to the capactiy that humans intrinsically understand a given thought/sentence assuming that they understand a suitably related thought/sentence. For instance, the capacity to understand the phrase "Mary loves John" would seem to entail the capacity to understand the phrase "John loves Mary", in virtue of the symmetry of the 'loves' relationship [2].

This type of systematicity is relatively straightforward to account for, even if the details remain elusive, under the assumption that humans are performing operations over symbolic expressions of some sort when reasoning/understanding sentences. For in that case, the systemic capacity to understand both sentences is explicable as the result of postulating underlying combinatorial rules that define how to compose constituent tokens into larger expressions, irrespective of the specific values of the tokens, in much the same way that logical inference rules

in formal logics are defined over the shape of expressions without respect to their specific contents.

This is in stark contrast to neural networks, which do not rely on combinatorial rules to model the semantic and syntactic relationships between words and phrases. Instead, the network is trained using vast data sets to pick up on statistical regularities and define semantic notions in terms of the co-occurence of words and phrases with other words and phrases [10].

Learning semantic relations in this way, however, can yield cases where systematicity in the learning agent is violated, such that the agent can only understand one direction of the phrase "Mary loves John" and not the other. This can occur simply because of the nature of the data set and how it ends up tuning the network's weights. Although it may be the case that for a given data set systematicity may appear to be present with respect to certain words or phrases, this is, in some sense, a lucky accident due the properties of that data set, rather than something that intrinsically follows from the learning procedure, and which is guaranteed to generalize to new words/phrases. However, since humans seem never to have this problem of understanding one phrase and not understadning its symmetric partner, it seems unlikely that neural networks alone will be able to account for the systematicity of cognition. They are, in effect, governed by the training data too strongly and specifically to prevent such lapses in systematicity from occurrring.

4 Related Work

In light of the necessity of reasoning and abstraction for artificial general intelligence, it is worth considering to what extent previous work has attempted to combine the methodologies of machine learning with problem domains more normally regarded as the purview of symbolic AI.

Some [8] have attempted to use various convolutional and recurrent neural network architectures to assist interactive theorems provers with the problem of determining the usefullness of certain statements for proving a given conjecture within the domain of higher-order logic. Although the baseline models are better than random, they fail to properly take account of the syntactic structure of the statements and conjectures in question, leaving significant room for the possibility of performance improvements should an architecture that can leverage such structure be discovered.

Others have attempted to apply ML to the simpler case of first-order logic [3]. Specifically, they address the problem of selecting the best pre-specified parameter setting (called a heuristic) of an interactive theorem prover on a per-conjecture basis. Results show a significant improvement over manually selecting a specific heuristic to be used across all conjectures, both in terms of the number of conjectures proved and the total amount of proving time requireed; however, the author's also note the importance of finding optimal features sets from amidst the combinatorially explosive complete feature space, as this likely a bottleneck with respect to overall performance on the task..

Similarly, still others [9] have taken to applying reinforcement learning to the boolean satisfiability problem, with the goal of improving the well known DPLL algorithm [4] by learning how to contextually select, based on the current SAT problem subinstance, the best DPLL branching rule from among a pre-specified set of such rules. For some specific problem instances encoded as SAT problems, results using RL were either competitive or superior to the approach of selecting a single branching. However, the authors note that there is still significant room for improvement with respect to how the current state of the problem is represented, which may allow for superior results across a wider range of problem instances in the future.

5 Learning Logics

5.1 Framework

In all of the related work discussed above, ML/RL is used essentially for the development of specialized heuristics to assist the theorem prover/solver in making decisions in some way; the focus is not on learning the axiomatic system itself. However, the question of how to learn a logic, rather than how to simply assist in deciding which inference to make within that logic, is, arguably, just as important a goal to achieve, at least with respect to laying the foundation for an approach to general intelligence.

The work of Sakama and Inoue [14] seeks to address this problem more explicitly. In particular, they propose an abstract framework for learning logics. This framework includes two agents: a teacher agent A that contains a deduction system L which takes in as input a set of sentences S in the target logic and outputs $T \subseteq Th(S)$, where $Th(S)$ is the set of logical consequences of S. The student agent M is equipped with a learning algorithm L, such that it takes in as input the set of sentences S and consequences T, and outputs the corresponding deductive system K, which can account for the inferential relationship between S and T (See figure 2 for a graphical depiction of the framework). The current proposal is that imitation inspired reinforcement learning [7] is one possible way of implementing this abstract framework.

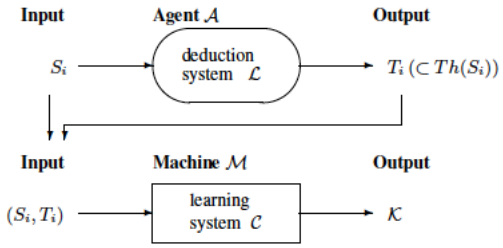

Fig. 2. Abstract Logic Learning Framework. Source: [14]

5.2 Learning By Imitation

The proposed problem formulation is as follows: the learning task is considered episodic, such that the set of sentences S, drawn uniformly at random from a meta-training set, $S - Meta$, of sets of sentences, comprise the start state of the episode. S is assigned to both the deductive system L and the learning system C so that each can engage in an independent derivation of the set T. The set T is itself computed prior to the start of the episode, and the presence of all of the elements of this set when in the current derivation of both the deductive system and the learning agent is treated as a terminal goal state, while all other states are non-terminal.

The action space is finite and discrete, and consists of the capacity to transform the current statement via the application of any of a number of pre-specified well-formed-formulas to the data structure that represents the current derivation being attempted by the learning agent C. Some of these transformations will be valid inference rules; others will not. While this introduces a degree of bias into the learning agent, in the sense that it is not being allowed to choose 'inferences' at a given time step from an infinite array of possibilites, so long as the pre-specified set is large relative to the number of valid inferences, of which there will only be so many that are non-trivial, there is still plenty of room for the agent to learn, and it serves as a good starting point implementationally to explore the notion of learning logics. This simplification can be removed in future work.

With respect to rewards, the agent is rewarded at each time step with a +1 if its most recent transformation is one that has already been performed by the deduction system at some point, and -1 otherwise. Essentialy, the notion is that the agent will travel through the space of possible statements, trying to derive the set T based on selecting actions according to a changing policy. This policy will be modified in virtue of how it leads to the agent being penalized to the degree that the agent's actions do not conform to the deductive system, and rewarded to the degree that they do, independently at each time step.

For example, consider the set of sentences and the corresponding inference below:

$$\begin{aligned} P \\ P \rightarrow Q \\ \therefore Q \end{aligned}$$

This is a simple example of modus ponens. In terms of the proposed problem formulation, the first two lines would constitute a state in the current episode, and the correct action would be to 'infer' Q, thus bringing the learning agent into the terminal state. The deduction system, when provided the first two lines as input, may in fact do just that, and terminate this episode quite quickly; but the learning agent will be free to select from among a much larger pre-specified set of atoms or expressions in an attempt to 'guess' what the consequence of the first two lines should be. If in fact it outputs Q, it will be rewarded with a

+1 and transition to the terminal state, ending the episode, while if it outputs anything else, it would be given a -1, even if it is a different valid inference (such as disjunction introduction), since it did not imitate what action the deduction system performed. The learning system would in this case fail to transition to the next terminal state, and, after performing an update to the cost function for that state (ee below), simply return back to the starting state composed of the first two statements. This provides the agent with the opportunity to try other actions, that is, make other guesses, and see what rewards they yield, either until the agent reaches the maximum number of time steps for the episode or correctly infers Q.

Over time, the learning system C is attempting to minimize the loss function defined by equation 1. It does so via a parameterized version of Q-learning [18], where *theta* denotes the weights of an artificial neural network that takes in the specified state and action, and outputs the value of that state-action pair, as described in section 3. Actions are selected using an ϵ-greedy policy, which selects the action with the highest value for the current state with probability $1 - \epsilon$, and selects randomly from among all the possible actions for the current state with probability ϵ. ϵ should be set to decay to 0 slowly over time, so that as the agent learns fewer of its choices will be based on chance, and more based on the value of the best action in the current state.

$$L_t = (r_t + \gamma \max_a Q(s_{t+1}, a, \theta_t) - Q(s_t, a_t, \theta_t))^2 \tag{1}$$

Ultimately, the goal is that the inference rules of the axiomatic system will become implicitly encoded within the agent's policy, at least to the degree that the learning agent's policy ends up reflecting the capacities of the deduction system.

6 Conclusion And Future Work

The approach proposed above is simply a sketch of the general notion of how to incorporate output from another agent, the deductive system L, to influence the learning agent C, in such a way so as to push C towards the behaviour of L, via imitation. A fully functioning implementation is needed in order to test out the validity of this manner of framing the problem.

Moreover, the problem as formulated above is among the most straightforward ways of doing so, and is done primarily with the notion fo implementational simplicity in mind, and it would likely be beneficial to modify the formulation in light of initial results. In particular, it seems probable that the state representation of the current set of sentences will play a large role in the quality of the learned agent, just as it did so in much of the related work articualted earlier. It is, therefore, likely that proper experimentation with respect to different ways of encoding the current state should be tested.

It may also be worth considering how to create state-dependent actions, wherein the action space of a given state is a function of the features of the

current state, rather than a pre-specified set, as this could possibly allow, when combined with a type of state aggregation based abstraction, for the learning of inference rules for non-atomic statements as below:

$$\begin{aligned} P \wedge Q \\ (P \wedge Q) \rightarrow R \\ \therefore R \end{aligned}$$

In addition, it seems likely that the specific deduction system used will itself have a profound effect on the capacity of the learning system to train well; different deductive systems are liable to deduce the same theorem in different ways depending on their implementation, and just as in real life some pupils may find particular instructors more congenial than others, it would be worth exploring to what degree the deductive system used impacts the capacity of the learning system to train well.

Finally, once a suitably rich state representation of logical statements is discovered, a profitable learning-deduction system pair is found, and the initial results are promising, in the sense of being better than random with a degree of statistical significance, it would be worthwhile to extend the scope of the problem beyond that of simply identifying correct, single step inference rules to the case where the deduction system infers multiple consequences of a given set of statements, and the agent must learn how to parse which inference rule is happening at each step, and see to what extent it can individuate between these different inference rules when more than one of them is used on a given set of sentences S.

7 Acknowledgments

The author would like to thank Jia You for their guidance and support concerning the present work.

References

1. Abbeel, P., Coates, A., Ng, A.Y.: Autonomous helicopter aerobatics through apprenticeship learning. The International Journal of Robotics Research **29**(13), 1608–1639 (2010)
2. Aydede, M.: The language of thought hypothesis. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University, fall 2015 edn. (2015)
3. Bridge, J.P., Holden, S.B., Paulson, L.C.: Machine learning for first-order theorem proving. Journal of automated reasoning **53**(2), 141–172 (2014)
4. Davis, M., Logemann, G., Loveland, D.: A machine program for theorem-proving. Commun. ACM **5**(7), 394–397 (Jul 1962). https://doi.org/10.1145/368273.368557, http://doi.acm.org/10.1145/368273.368557

5. Fodor, J.A., Pylyshyn, Z.W.: Connectionism and cognitive architecture: A critical analysis. Cognition **28**(1-2), 3–71 (1988)
6. Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks. In: Acoustics, speech and signal processing (icassp), 2013 ieee international conference on. pp. 6645–6649. IEEE (2013)
7. Hussein, A., Gaber, M.M., Elyan, E., Jayne, C.: Imitation learning: A survey of learning methods. ACM Computing Surveys (CSUR) **50**(2), 21 (2017)
8. Kaliszyk, C., Chollet, F., Szegedy, C.: Holstep: A machine learning dataset for higher-order logic theorem proving. CoRR **abs/1703.00426** (2017), http://arxiv.org/abs/1703.00426
9. Lagoudakis, M.G., Littman, M.L.: Learning to select branching rules in the dpll procedure for satisfiability. Electronic Notes in Discrete Mathematics **9**, 344–359 (2001)
10. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
11. Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., Bowling, M.: Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. Science **356**(6337), 508–513 (2017)
12. Mukundan, J., Martinez, J.F.: Morse: Multi-objective reconfigurable self-optimizing memory scheduler. In: High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on. pp. 1–12. IEEE (2012)
13. Russel, S., Norwig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall (2003)
14. Sakama, C., Inoue, K.: Can machines learn logics? In: International Conference on Artificial General Intelligence. pp. 341–351. Springer (2015)
15. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. nature **529**(7587), 484–489 (2016)
16. Sutton, R.S., Barto, A.G.: Reinforcement Learning : An Introduction. MIT Press (2017)
17. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1701–1708 (2014)
18. Watkins, C.J.C.H.: Learning from delayed rewards. Ph.D. thesis, King's College, Cambridge (1989)