Supplement to Chapter 10: Neuro Symbolic Reasoning and Sequential Decision Making

Supplment to Chapter 10 from Neuro Symbolic Reasoning and Learning - Current Advances and Future Directions

Deep Symbolic Regression

In this supplement, we provide a diagramthat outlines the overall approach to DSR in Figure 0.1. An example expression tree for the expression $\frac{9}{2} - e^4$ is shown in Figure 0.2. we briefly describe a very recent transformer-based approach to this problem introduced by Meta AI at NeurIPS [4].

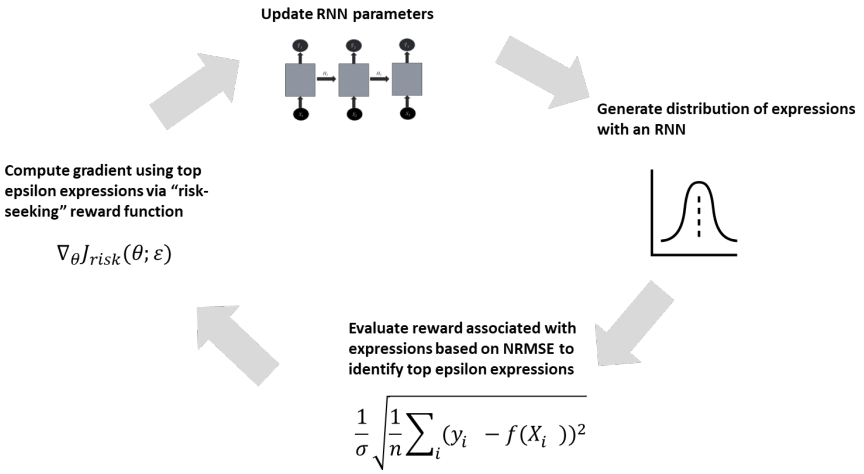

Fig. 0.1 Overview of deep symbolic regression.

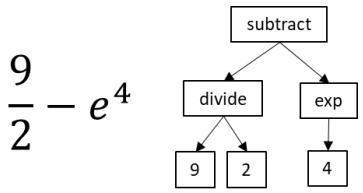

Fig. 0.2 Example expression tree.

Symbolic Regression with Transformers:
An Alternative to Deep Symbolic Regression.
At the time of this writing (2022) the transformer architecture [20] has largely supplanted RNN's in many applications which may lead some to wonder if DSR can be improved upon by leveraging this powerful architecture. Meta AI has recently introduced such an approach [4] and it is worth noting, at least at a high level some of the design decisions. Perhaps the key difference is that [4] uses a more traditional supervised paradigm, however a "sample" consisting to individual symbolic regression problems. In other words, their model is trained on three million (X, y) pairs where X is an $d \times n$ matrix. As this leads to each sample being of size $O(nd)$, an embedding is used to reduce the input size. The transformer uses 16 attention heads, an embedding dimension of 512 and 86 million parameters to produce a model. The model is trained with generated data, relying on established techniques in symbolic regression (resulting in three million training samples). The result is a model that accepts an input matrix X as described earlier and returns an expression - functioning much like a language model. This approach also is reported to produce state-of-the-art performance and has the benefit of high speed after training.

Deep Symbolic Policy Learning

In Figure 0.3 we illustrate this in an example on the Lunar Lander.
Other Considerations and Empirical Evaluation. As with DSR, DSP uses a constant optimization process as an additional refinement step. Empirical studies in [9] examined results both with and without constant optimization. An additional hierarchical entropy regularization term was also shown to improve results (note this regularization was also used in DSR). Another empirical improvement was the utilization of a soft prior for the length of an expression, as it was found in experiments that when a maximum length was used on the expression length, nearly all expressions had a length equal to the threshold. Empirical results on a suite of

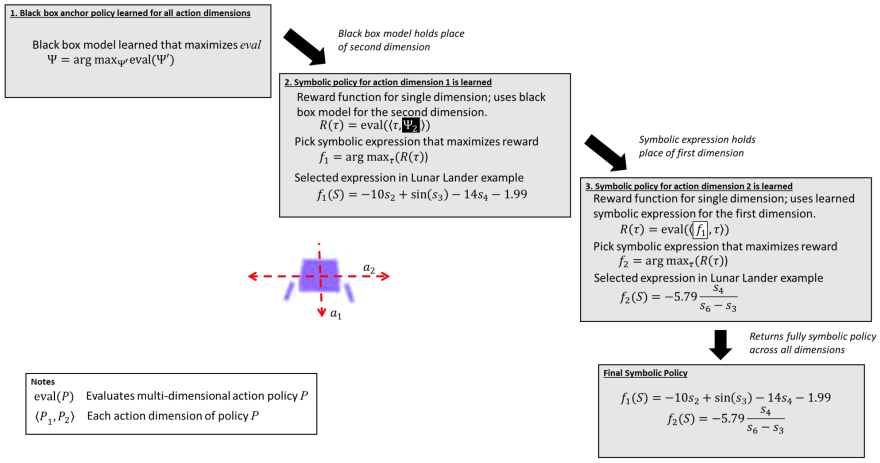

Fig. 0.3 DSP Learning using a black box model to support multiple action dimensions.

tests from OpenAI Gym (including the Lunar Lander used in the figures of this chapter) showed that DSP consistently provided near state-of-the-art results while providing for a more compact and explainable model.

Verifying Neural-based Models

Signal Temporal Logic (STL): A Brief Summary.
Signal Temporal Logic (STL) assumes an underlying set of signals associated with a set of m signals denoted X that underlie the predicates of the logic. A "signal predicate," μ is an atomic statement that is true when $f(x) \leq 0$ for some $x \in X$ and function x. A formula is defined based on the following grammar.

$$\varphi ::= \mu \mid \neg\varphi \mid \varphi' \wedge \varphi'' \mid \varphi' \vee \varphi'' \mid \Diamond_{(a,b)}\varphi \mid \Box_{(a,b)}\varphi \mid \varphi \mathbf{U}_{(a,b)}\varphi \quad (0.1)$$

Aside from the temporal operators ($\Diamond, \Box, \mathbf{U}$), we can view the remainder of the operations as being treated with the same semantics as propositional logic (see Chapter **??**) with the addition that μ is defined in terms of the a function associated with the value of an underlying signal. Note that we use the variant of STL described iin [10] which assumes that each signal predicate is associated with a single signal of X where other treatments more resemble LTN's grounding of symbols (Chapter **??**) where the symbol is defined in terms of a function over all signals. We can view the semantic structure as consisting

of a sequence of worlds over many timesteps. Likewise, each signal $x \in X$ has a value associated with each timestep. As such, the temporal operators "eventually," "always," and "until" are temporal operators ($\Diamond, \Box, \mathbf{U}$) allow for specifying formulas over multiple periods of time.

STL allows for the expression of various properties. We list several below. These can be used to express safety properties, enforce resource constraints, or account for physical laws.

- *Reasonable range.* Values of certain signals should always be within a specified range.
- *Consecutive change.* The change in a value within a certain time period should be withing a certain threshold.
- *Temporal Correlation.* Certain correlations known a-priori can be captured by such a constraint.
- *Existence.* Enforces that at a signal will eventually satisfy a certain property.

0.0.1 LNN Shielding and Guiding

The intuition behind the trace generation in STLNet can be viewed as a type of refinement if neural output. We have seen similar ideas in DSP learning (e.g., constant optimization, replacing black-box action dimensions). A similar type of refinement based on logical constraints for a reinforcement learning trained agent has been described in [7] where they enforce constraints using logical neural networks (LNN's). In that work, they introduce the concepts of "LNN Shielding" and "LNN Guiding." Here, a black box LSTM neural model is used to train a reinforcement learning agent and its actions are modified based on the deduction process on a logic program resulintg from a trained LNN [1]

We note that the idea of shielding and guarding will likely work with other paradigms other than LNN - as the authors are essentially assuming the existence of a logic program. The work does use the fact that LNN's rely on a logic that uses intervals over reals. However other paradigms such as annotated logic [6, 17], probabilistic logic [14] and other variants [18, 5, 16] also allow for this and could be suitable for this task.

Intuitively, the logic program, Π, represents external knowledge. The trained LNN adds the annotations - the $[\ell, u]$ bounds - to each propositional sentence in Π. In [7], the current state s can also be represented as a formula and a given action a can be represented as an atomic proposition. With *shielding* we can say action a is "safe" if the following is consistent.

$$\Pi \cup \{s, a\} \tag{0.2}$$

If it is inconsistent, we can measure the inconsistency by examining the difference $\ell - u$ for all sentences associated containing a and take the maximum (this is

[1] See the discussion on the "upward-downard" algorithm in Chapter **??**.

denoted $ctrd(a, s)$). In [7] the authors relax consistency requirements by considering a threshold value α which allows for some small level of inconsistency to be tolerated. The idea of guiding extends this notion by examining the interval values associated with a and subtracting $ctrd(a, s)$. They use this to compute the probability $P(a|s)$ as follows.

$$P(a|s) = \frac{e^{v(a,s)}}{\sum_{a' \in A} e^{v(a',s)}} \tag{0.3}$$

$$v(a, s) = \frac{ell_a + u_a}{2} - ctrd(a, s) \tag{0.4}$$

The agent selects an action at time t (a_t) based on the following criteria.

$$a_t = \begin{cases} \mathsf{argmax}_{a \in A} P(a|s)Q(s, a) & C > \epsilon \\ \mathsf{random}_{a \in A} P(a|s) & otherwise \end{cases} \tag{0.5}$$

Here C is a random value that determines if the action takes a random action for epsilon-greedy action selection. These methods have been shown to provide greater rewards for an agent navigating through a text-based agent simulation in [7] and represent an interesting initial step in this line of work.

References

1. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: Openai gym (2016)
2. Corso, A., Moss, R., Koren, M., Lee, R., Kochenderfer, M.: A survey of algorithms for black-box safety validation of cyber-physical systems. J. Artif. Int. Res. **72**, 377–428 (2022). DOI 10.1613/jair.1.12716. URL https://doi.org/10.1613/jair.1.12716
3. Hekmatnejad, M., Hoxha, B., Deshmukh, J.V., Yang, Y., Fainekos, G.: Formalizing and evaluating requirements of perception systems for automated vehicles using spatio-temporal perception logic (2022). DOI 10.48550/ARXIV.2206.14372. URL https://arxiv.org/abs/2206.14372
4. Kamienny, P.A., d'Ascoli, S., Lample, G., Charton, F.: End-to-end symbolic regression with transformers. In: A.H. Oh, A. Agarwal, D. Belgrave, K. Cho (eds.) Advances in Neural Information Processing Systems (2022)
5. Khuller, S., Martinez, M.V., Nau, D.S., Sliva, A., Simari, G.I., Subrahmanian, V.S.: Computing most probable worlds of action probabilistic logic programs: scalable estimation for $10^{30,000}$ worlds. Ann. Math. Artif. Intell. **51**(2-4), 295–331 (2007). DOI 10.1007/s10472-008-9089-2. URL https://doi.org/10.1007/s10472-008-9089-2
6. Kifer, M., Subrahmanian, V.: Theory of generalized annotated logic programming and its applications. J. Log. Program. **12**(3&4), 335–367 (1992)
7. Kimura, D., Chaudhury, S., Wachi, A., Kohita, R., Munawar, A., Tatsubori, M., Gray, A.: Reinforcement learning with external knowledge by using logical neural networks. CoRR **abs/2103.02363** (2021). URL https://arxiv.org/abs/2103.02363
8. Lamport, L.: Sometime' is sometimes 'not never'. In: Proceedings of the Seventh ACM Symposium on Principles of Programming Languages, ACM SIGACT-SIGPLAN (1980). URL https://www.microsoft.com/en-us/research/publication/sometime-sometimes-not-never/

9. Landajuela, M., Petersen, B.K., Kim, S., Santiago, C.P., Glatt, R., Mundhenk, N., Pettit, J.F., Faissol, D.: Discovering symbolic policies with deep reinforcement learning. In: International Conference on Machine Learning, pp. 5979–5989. PMLR (2021)
10. Ma, M., Gao, J., Feng, L., Stankovic, J.A.: Stlnet: Signal temporal logic enforced multivariate recurrent neural networks. 34th Conference on Neural Information Processing Systems (NeurIPS 2020) URL https://par.nsf.gov/biblio/10231392
11. Marcus, G.: Deep learning: A critical appraisal. CoRR **abs/1801.00631** (2018). URL http://arxiv.org/abs/1801.00631
12. Miltersen, P.B., Radhakrishnan, J., Wegener, I.: On converting cnf to dnf. Theoretical Computer Science **347**(1), 325–335 (2005). DOI https://doi.org/10.1016/j.tcs.2005.07.029. URL https://www.sciencedirect.com/science/article/pii/S0304397505004688
13. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015). URL http://dx.doi.org/10.1038/nature14236
14. Nilsson, N.J.: Probabilistic logic. Artificial Intelligence **28**(1), 71–87 (1986). DOI https://doi.org/10.1016/0004-3702(86)90031-7. URL https://www.sciencedirect.com/science/article/pii/0004370286900317
15. Petersen, B.K., Larma, M.L., Mundhenk, T.N., Santiago, C.P., Kim, S.K., Kim, J.T.: Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. arXiv preprint arXiv:1912.04871 (2019)
16. Shakarian, P., Parker, A., Simari, G., Subrahmanian, V.V.S.: Annotated probabilistic temporal logic. ACM Trans. Comput. Logic **12**(2) (2011). DOI 10.1145/1877714.1877720. URL https://doi.org/10.1145/1877714.1877720
17. Shakarian, P., Simari, G.: Extensions to generalized annotated logic and an equivalent neural architecture. In: IEEE TransAI. IEEE (2022)
18. Shakarian, P., Simari, G.I., Schroeder, R.: Mancalog: a logic for multi-attribute network cascades. In: M.L. Gini, O. Shehory, T. Ito, C.M. Jonker (eds.) International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013, pp. 1175–1176. IFAAMAS (2013). URL http://dl.acm.org/citation.cfm?id=2485129
19. Tamar, A., Glassner, Y., Mannor, S.: Optimizing the cvar via sampling (2014). DOI 10.48550/ARXIV.1404.3862. URL https://arxiv.org/abs/1404.3862
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc. (2017). URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf