Learning to Reason: Predict-then-Optimize vs Predict-and-Optimize

Marianne Defresne[1,2], Thomas Schiex[2], Sophie Barbe[1]

[1] TBI, Université de Toulouse, CNRS, INRAE, INSA, ANITI, 31077 Toulouse, France; firstname.name@insa-toulouse.fr

[2] Université Fédérale de Toulouse, ANITI, INRAE, UR 875 31326 Toulouse, France; firstname.name@inrae.fr

Abstract

Many real-life decision making problems involve reasoning or optimizing over discrete variables on ill-defined problems, where exact constraints or parameters are unknown and only indirect correlated variables are observed together with solutions. In these situations, one may want to machine learn how to predict a solution directly from the observed variables, learning effectively "how to reason" or "how to optimize". A promising research direction involves the combination of Machine Learning (ML) and discrete reasoning (DR) [1]: the problem is formulated as the discrete optimization/reasoning problem whose parameters are predicted from data. Because of the various types of observed variables and of the complexity of their relationship to hidden constraints and parameters, Deep Learning (DL) is often considered as a suitable learning tool here. However, DL requires differentiable loss functions, which are either zero or unavailable for discrete reasoning [2].

Two approaches have emerged to tackle such problems. On the one hand, the *Predict-then-optimize* approach learns how to predict the discrete problem to solve and then solves it. The basic components are well understood, learning can be efficient and exact DR can be performed during inference. But the effect of errors on parameters on the final solution are not accounted for [3] which makes the approach sub-optimal in low-data regimes.

For this reason, much of the attention has been focused on the direct integration of optimization in the loss function, an approach that is often denoted as *Predict-and-optimize* or *Decision-focused learning* [4]. The challenges here is to provide efficient end-to-end differentiable training. Some approaches rely on efficient differentiable optimization layers that capture the DR problem through a continuous relaxation [5, 6, 7]. They can tackle complex NP-hard problems at the cost of approximate solving during inference: even optimally parameterized relaxations will fail to offer exact solutions. Another family of approaches extracts meaningful gradients out of exact DR solvers during training [2, 8, 9], enabling exact solving during inference. This however limits their application to very small instances of NP-hard DR problems (e.g., scheduling with 3 tasks only [3]) because of the tremendous optimization cost during training, a cost which is worsened by the essentially randomly parameterized nature of predicted problems in the first training epochs.

We compare a *Predict-then-optimize* and a *Predict-and-optimize* approach on a classical Constraint Programming (CP) benchmark, the NP-complete sudoku problem. Our goal is to learn how to solve new grids of sudoku from examples of solved sudoku grids (as we already did using ML technology [10]), described here as a sequence of 81 numbers, each with an associated grid coordinate. To relax the Boolean nature of constraints (satisfied or violated), we learn the parameters of a weighted CP model (a binary Cost Function Network or CFN [11]) using the CFN solver toulbar2 [12] during inference and training (in Predict-and-optimize mode).

For *Predict-then-optimize*, we introduce the Gangster-PLL loss, a variant of the well-established Pseudo-loglikelihood loss [13] and use the Hinge Loss [9] for *Predict-and-optimize*, both with regularization on the CFN parameters learned. We observe that the *Predict-then-optimize* approach combines efficient learning with exact reasoning during inference, providing totally accurate solutions even in relatively low regime situations. The far more expensive *Predict-and-optimize* approach struggles at learning how to solve a sudoku grid, requiring more tuning of the exact

solver. Contrarily to the Gangster-PLL approach, it tends to learn minimal CFN models, where redundant constraints are absent [14].

We conclude that in many situations, the *Predict-then-optimize* approach has numerous advantages. It is possibly the only usable approach when it comes to learning how reason from huge solutions. We are currently applying it with success on a challenging NP-hard problem with examples having several thousands of variables, a situation that would be clearly out of reach of *Predict-then-optimize* approaches relying on exact DR solvers.
Overall, we estimate this presentation would take between 15 and 30 minutes.

References

[1] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[2] M. Vlastelica, A. Paulus, V. Musil, G. Martius, and M. Rolínek, "Differentiation of blackbox combinatorial solvers," 2019.

[3] J. Mandi, V. Bucarey, M. M. K. Tchomba, and T. Guns, "Decision-focused learning: Through the lens of learning to rank," in *Proceedings of the 39th International Conference on Machine Learning* (K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, eds.), vol. 162 of *Proceedings of Machine Learning Research*, pp. 14935–14947, PMLR, 17–23 Jul 2022.

[4] B. Wilder, B. Dilkina, and M. Tambe, "Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1658–1665, Jul. 2019.

[5] P.-W. Wang, P. Donti, B. Wilder, and Z. Kolter, "SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6545–6554, PMLR, 09–15 Jun 2019.

[6] A. N. Elmachtoub and P. Grigas, "Smart "predict, then optimize"," *Management Science*, vol. 68, no. 1, pp. 9–26, 2022.

[7] J. Mandi and T. Guns, "Interior point solving for lp-based prediction+ optimisation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7272–7282, 2020.

[8] S. S. Sahoo, A. Paulus, M. Vlastelica, V. Musil, V. Kuleshov, and G. Martius, "Backpropagation through combinatorial algorithms: Identity with projection works," 2022.

[9] Y. Altun, D. McAllester, and M. Belkin, "Maximum margin semi-supervised learning for structured variables," *Advances in neural information processing systems*, vol. 18, 2005.

[10] C. Brouard, S. d. Givry, and T. Schiex, "Pushing data into CP models using graphical model learning and solving," in *International Conference on Principles and Practice of Constraint Programming*, pp. 811–827, Springer, 2020.

[11] T. Schiex, H. Fargier, G. Verfaillie, *et al.*, "Valued constraint satisfaction problems: Hard and easy problems," *IJCAI (1)*, vol. 95, pp. 631–639, 1995.

[12] D. Allouche, S. d. Givry, G. Katsirelos, T. Schiex, and M. Zytnicki, "Anytime hybrid best-first search with tree decomposition for weighted csp," in *International Conference on Principles and Practice of Constraint Programming*, pp. 12–29, Springer, 2015.

[13] J. Besag, "Statistical analysis of non-lattice data," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 24, no. 3, pp. 179–195, 1975.

[14] B. Demoen and M. G. de la Banda, "Redundant sudoku rules," *Theory and Practice of Logic Programming*, vol. 14, no. 3, pp. 363–377, 2014.