Decision Focused Learning for Prediction + Optimisation Problems

Maxime Mulamba[1, 2], **Emilio Gamba**[1, 2], **Tias Guns**[1, 2]

[1]Vrije Universiteit Brussel, Belgium,
[2]KU Leuven, Belgium,
{maxime.mulamba, emilio.gamba}@vub.be, tias.guns@kuleuven.be

Abstract

Increasingly, constraint programming problems can not be fully specified as facts, but part of the problem input is predicted using machine learning; for example demand and price in energy scheduling, (truck)load in vehicle transport, or even visual or natural language input. This talk has two purposes. First, we will highlight two settings in which a tighter integration between the machine learning and the model-then-solve leads to improved results. Next, we introduce the CPMpy constraint programming library, which was made to be highly compatible with machine learning libraries, in order to use the output of one in the other and reversely.

Motivation

Machine Learning (ML) and *Constraint Solving* (including CP, SAT, MIP) are central paradigms for respectively learning and reasoning in AI. Traditionally, Machine Learning techniques are developed for prediction tasks in a wide range of application domains such as visual question answering (Wu et al. 2017), medical diagnosis (Garg and Mago 2021), and many more (Shinde and Shah 2018). Whereas, constraint-solving methods involve formulating and solving complex constraint satisfaction and optimization problems. Highly efficient solvers exist for the various paradigms in constraint solving including Constraint Programming (CP), Boolean satisfaction problems (SAT), Pseudo-Boolean optimization problems, (mixed) integer linear programming problems (MILP) as well as knowledge compilation approaches.

Many real-world applications can be modeled as constraint-solving tasks. In practice, given a decision-making problem, domain experts interact with an optimization expert to model the problem as a constraint-solving task, based on stakeholders' requirements. Position papers (Canoy and Guns 2019) argue that this process is static and too rigid. A modern constraint solver should be able to learn from implicit user preferences or from the environment, both of which are uncertain and must be inferred from available features or raw sensor input. This is why in reality, practitioners rely on a combination of ML and Constraint solving.

Consider applications such as Energy-cost-Aware Scheduling, where a set of tasks are scheduled on a set of machines over a given period divided into time slots. The goal is to efficiently allocate machine usage, where the energy cost at each time slot is not known and must be estimated (Simonis et al. 2014). In such a case, a practitioner will first train an ML model on the prediction task as in standard supervised learning. Then the trained ML model is used to make a points estimate of the uncertain parameter, which serves as input to the constraint solver. However, such an approach does not take into account the interplay of constraint solving with prediction errors nor the impact it has on solution quality (Mandi et al. 2020).

We propose an overview of approaches to solving constraint programming problems where part of the problem input is predicted using machine learning, for example, demand and price in energy scheduling, (truck)load in vehicle transport or even visual or natural language input involved in a more complex reasoning task.

First, we highlight two settings in which a tighter integration between the machine learning and the model-then-solve leads to improved results. Next, we introduce the CPMpy constraint programming library, which was made to be highly compatible with machine learning libraries, so as to use the output of one in the other and reversely.

Prediction + Optimisation

Recent years have witnessed an increasing interest in hybrid systems that leverage machine learning for constraint solving problems (Bengio, Lodi, and Prouvost 2021; Cappart et al. 2021; Kotary et al. 2021). We specifically focus on problems under the *Predict-and-Optimize* category, wherein a subset of coefficients in the objective function of a discrete optimization task must be inferred from noisy input data (Elmachtoub and Grigas 2022).

Two-Staged Predict-and-Optimize

Predict-and-optimize problems are usually solved in a *two-stage* process: first, prediction of the coefficients in objective function from features; then solving of the constraint reasoning task. In the case of the prediction step being a classification task, a wrong prediction can lead to a reasoning task with no feasible solutions. As a pedagogical example, consider an AI system developed to interpret and solve a pen-and-paper Sudoku puzzle scanned with a smartphone (Mulamba et al. 2020). After a picture of a Sudoku puzzle is

taken, the picture is segmented into 81 cells. A pre-trained convolutional neural network (CNN) is then applied to each cell in order to make a digit prediction. For each cell, the output of the CNN is a probability distribution over the 10 possible values: digits 1-9 and the empty cell value. Even a CNN with an accuracy of 99% would lead to a correct solution only approximately $0.99^{81} = 44\%$ of the time. In this tutorial, we will explore ways to mitigate this issue by smarter modeling of the reasoning task to account for uncertainty. The *visual* Sudoku also illustrates how this framework is also suitable for Constraint Satisfaction Problems.

Decision-Focused Learning

Predict-and-Optimize problems can also involve a regression task. In our case, the practitioner cares more about the solution quality in the downstream reasoning task than the accuracy on the prediction task. Therefore, a ML model trained within that context should aim to minimize a loss function which reflects that solution quality. Namely, the learning algorithm should be *decision*-focused instead of prediction-focused. There is a growing amount of work on decision-focused learning (DFL) methods in the past few years (Niepert, Minervini, and Franceschi 2021; Wilder, Dilkina, and Tambe 2019; Mandi and Guns 2020; Pogancic et al. 2020) .

The main challenge most of them aim to overcome is that a loss function that reflects the solution quality for a discrete optimization task is discrete. Hence, its gradient does not provide meaningful weights update for a gradient-based learning algorithm, such as commonly used to train Deep Neural Network (DNN).

A second challenge is scalability. Indeed, each forward pass through the DNN while training requires solving the downstream reasoning task, which can be computationally expensive. Exploiting the incrementality capabilities of solvers, as done in CPMpy, is also key here.

We will introduce a family of DFL methods that work with CP solvers (Mandi et al. 2021) and that address both challenges (Mandi et al. 2022; Mulamba et al. 2021).

CPMpy

CPMpy is an open-source python CP Modelling library CPMpy (Guns 2019) for easy interfacing with current constraint solvers and lower-level libraries: OR-tools, MiniZinc in CP, Z3 in SMT, PySAT in SAT, Gurobi in MIP. On the language modeling side, CPMpy uses the widely popular n-dimensional arrays of NumPy (*ndarray*) as a building block for constants, and decisions variables thus allowing for easy integration with scientific packages such as SciPy, the general convex optimization framework CVXpy and popular deep learning (ML) frameworks such as Tensorflow and PyTorch. Furthermore, CPMpy eases incremental solving for solvers that support it, especially for predict-and-optimize methods that require solving the same constrained problem with different predictions.

Detailed, Point-Form Outline of the Tutorial

1. Prediction + Optimization
   (a) Perception-based constraint solving
      - State-of-the-art techniques to combine CP with ML
      - Demonstration: Use-case of the Sudoku Assistant
   (b) Decision Focused learning
      - Challenges
      - Surrogate Losses
      - Solution Cache
2. Quick introduction to CP using CPMpy:
   - Motivation
   - Examples demonstrating the language constructs
   - Integrating CP with ML predictions: An example from perception-based constraint solving
3. Conclusion, Outlooks, and Questions

Acknowledgements

This research received partial funding from the Flemish Government (AI Research Program); and funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant No. 101002802, CHAT-Opt).

References

Bengio, Y.; Lodi, A.; and Prouvost, A. 2021. Machine learning for combinatorial optimization: A methodological tour d'horizon. *Eur. J. Oper. Res.*, 290(2): 405–421.

Canoy, R.; and Guns, T. 2019. Vehicle Routing by Learning from Historical Solutions. In *CP*, volume 11802 of *Lecture Notes in Computer Science*, 54–70. Springer.

Cappart, Q.; Chételat, D.; Khalil, E. B.; Lodi, A.; Morris, C.; and Velickovic, P. 2021. Combinatorial Optimization and Reasoning with Graph Neural Networks. In *IJCAI*, 4348–4355. ijcai.org.

Elmachtoub, A. N.; and Grigas, P. 2022. Smart "Predict, then Optimize". *Manag. Sci.*, 68(1): 9–26.

Garg, A.; and Mago, V. 2021. Role of machine learning in medical research: A survey. *Computer Science Review*, 40: 100370.

Guns, T. 2019. Increasing modeling language convenience with a universal n-dimensional array,CPpy as python-embedded example. In *The 18th workshop on Constraint Modelling and Reformulation (ModRef 2019)*.

Kotary, J.; Fioretto, F.; Hentenryck, P. V.; and Wilder, B. 2021. End-to-End Constrained Optimization Learning: A Survey. In *IJCAI*, 4475–4482. ijcai.org.

Mandi, J.; Bucarey, V.; Tchomba, M. M. K.; and Guns, T. 2022. Decision-Focused Learning: Through the Lens of Learning to Rank. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, 14935–14947. PMLR.

Mandi, J.; Canoy, R.; Bucarey, V.; and Guns, T. 2021. Data Driven VRP: A Neural Network Model to Learn Hidden Preferences for VRP. In *CP*, volume 210 of *LIPIcs*, 42:1–42:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Mandi, J.; Demirovic, E.; Stuckey, P. J.; and Guns, T. 2020. Smart Predict-and-Optimize for Hard Combinatorial Optimization Problems. In *AAAI*, 1603–1610. AAAI Press.

Mandi, J.; and Guns, T. 2020. Interior Point Solving for LP-based prediction+optimisation. In *NeurIPS*.

Mulamba, M.; Mandi, J.; Canoy, R.; and Guns, T. 2020. Hybrid classification and reasoning for image-based constraint solving. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 364–380. Springer.

Mulamba, M.; Mandi, J.; Diligenti, M.; Lombardi, M.; Bucarey, V.; and Guns, T. 2021. Contrastive Losses and Solution Caching for Predict-and-Optimize. In *IJCAI*, 2833–2840. ijcai.org.

Niepert, M.; Minervini, P.; and Franceschi, L. 2021. Implicit MLE: Backpropagating Through Discrete Exponential Family Distributions. In *NeurIPS*, 14567–14579.

Pogancic, M. V.; Paulus, A.; Musil, V.; Martius, G.; and Rolínek, M. 2020. Differentiation of Blackbox Combinatorial Solvers. In *ICLR*. OpenReview.net.

Shinde, P. P.; and Shah, S. 2018. A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, 1–6. IEEE.

Simonis, H.; O'Sullivan, B.; Mehta, D.; Hurley, B.; and Cauwer, M. D. 2014. CSPLib Problem 059: Energy-Cost Aware Scheduling. http://www.csplib.org/Problems/prob059.

Wilder, B.; Dilkina, B.; and Tambe, M. 2019. Melding the Data-Decisions Pipeline: Decision-Focused Learning for Combinatorial Optimization. In *AAAI*, 1658–1665. AAAI Press.

Wu, Q.; Teney, D.; Wang, P.; Shen, C.; Dick, A.; and Van Den Hengel, A. 2017. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163: 21–40.