La PLL-Emmental pour l'apprentissage neuro-symbolique efficace de contraintes & fonction objectif

Marianne Defresne[1], Sophie Barbe[2], Thomas Schiex[3]

[1] LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse
marianne.defresne@laas.fr
[2] TBI, Université de Toulouse, CNRS, INRAE, INSA, ANITI, France
[3] Université Fédérale de Toulouse, ANITI, INRAE, UR 875, France

Mots-clés : *Constraint programming, Machine Learning, Optimisation combinatoire*

La programmation par contraintes a été décrite comme le Saint Graal de la programmation : "l'utilisateur décrit le problème, l'ordinateur le résout" [5]. En pratique, modéliser une situation réelle en un ensemble de variables, de contraintes et éventuellement une fonction objectif est une tâche difficile requérant des connaissances expertes du domaine d'application. De plus, le caractère formel des modèles de programmation par contraintes ou de recherche opérationnelle rend difficile la modélisation de problèmes réels (incertains, dynamiques, contextuels, ...). Une approche pour simplifier la modélisation et la rendre plus flexible est d'apprendre les paramètres du modèle à partir de données. Ici, nous présentons une méthode pour l'apprentissage de contraintes souples, permettant de représenter des données incertaines, à partir d'exemples de solutions (proche-)optimales. Un exemple simple illustrant ce cadre est d'apprendre à jouer au Sudoku à partir d'images de grilles résolues sans jamais connaître les règles.

Formalisation. Le problème est modélisé par un réseau de fonction de coûts (CFN pour *cost function network*) [2], une extension pondérée de la programmation par contraintes, offrant une modélisation plus flexible adaptée notamment aux données incertaines. Les CFNs permettent de modélisation des problèmes de satisfaction de contraintes pondérées (WCSP pour *Weighted Constraint Satisfaction Problem*). Le WSCP généralise le problème de satisfaction de contraintes (CSP) en attribuant un coût à chaque affectation d'un sous-ensemble de variables. Par rapport à un CSP classique, les fonctions de coût capturent à la fois les contraintes (associées à un coût maximal) et la fonction objectif.

L'apprentissage automatique d'un CFN revient à apprendre toutes les fonctions de coûts (donc simultanément les contraintes et objectifs). La difficulté principale est de combiner ce modèle, dont les variables de décision sont discrètes, avec l'optimisation continue de l'apprentissage. Concrètement, pour une structure d'entrée (par exemple une grille de Sudoku), un réseau de neurones prédit l'ensemble des fonctions de coût (modélisant les règles du jeu), qui est ensuite minimisé pour obtenir une solution (grille complétée). Les règles apprises peuvent être évaluées en comparant la solution prédite à celle observée. Toutefois, le nombre d'erreurs ne peut pas directement être utilisé comme fonction de perte pour entraîner le réseau de neurones car il est discret et ses gradients sont donc non-informatifs. Les méthodes existantes abordent ce problème soit en construisant une fonction de perte différentiable [6] ou via une relaxation continue du problème discret [7]. Les premières souffrent d'un temps d'entraînement long dû aux résolutions répétées de problèmes NP-difficiles, rendant tout passage à l'échelle prohibitif, tandis que les secondes ne permettent que des résolutions approximatives, même à l'inférence. Sur le problème du Sudoku, elles sont limitées respectivement par un entraînement de plusieurs jours et par l'incapacité à fournir des solutions pour la totalité des grilles de test.

Méthode. Notre méthode consiste à découpler l'apprentissage de l'optimisation afin de permettre un meilleur passage à l'échelle (Fig 1) . La difficulté est alors d'évaluer la qualité des

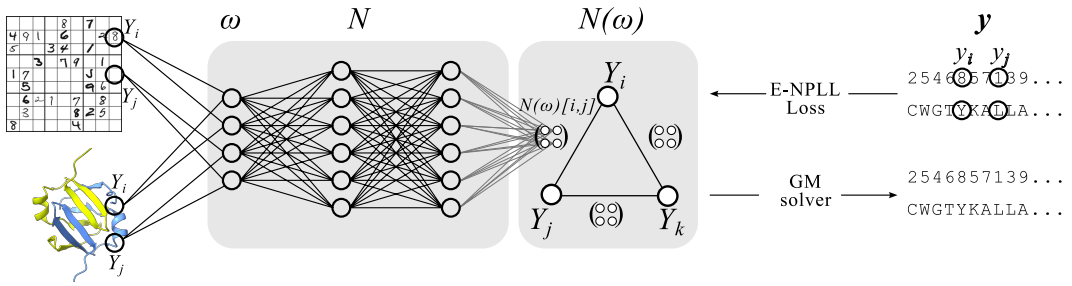

FIG. 1 – Architecture hybride proposée, composée d'un réseau de neurones et d'un solveur discret. Le réseau de neurones N prend en entrée un objet structuré (comme une grille de Sudoku ou une structure de protéines) et prédit les paramètres d'un problème combinatoire, qui, une fois optimisée par la solveur, permet de retrouver les solutions observées. Le réseau est entraîné sans le solveur grâce à la loss E-PLL.

fonctions de coûts prédites par le réseau de neurones sans résoudre le problème combinatoire qu'elles définissent (les "vrais" coûts n'étant pas observés). Pour ce faire, ces fonctions de coûts prédites sont interprétées en termes de probabilités — un coût élevé correspondant à une probabilité faible — en considérant le CFN comme un champ de Markov, une version stochastique des CFN. Les champs de Markov peuvent être appris à partir de données en minimisant une pseudo-vraisemblance (PLL) [1], mais la PLL échoue à apprendre des règles logiques, telles que celles du Sudoku. Intuitivement, le comportement de la PLL est lié à la redondance partielle entre les contraintes à apprendre. Nous avons donc proposé une variante, la PLL Emmental (ou E-PLL) [3], qui supprime cette redondance en masquant aléatoirement quelques variables.

Résultats. Sur le benchmark d'apprentissage du Sudoku, l'E-PLL permet d'apprendre les contraintes correctes (et donc de résoudre toutes les grilles de test) 10 fois plus vite et avec 50 fois moins de données que les méthodes existantes. C'est à notre connaissance la première méthode intégrant un solveur exact capable de passer à l'échelle. Nous avons étendu notre architecture hybride s'étend à d'autres problèmes de la communauté CP/ML, dont le Decision-Focused-Learning (DFL), qui vise à prédire les paramètres d'une fonction objectif linéaire de manière à optimiser la décision finale [4]. Enfin, l'E-PLL s'applique sur des problèmes réels de grande taille, comme celui de designer des séquences protéines optimisée pour une structure 3D d'entrée. La méthode a été validée expérimentalement sur deux designs de protéines.

References

[1] Julian Besag. "Statistical analysis of non-lattice data". In: *Journal of the Royal Statistical Society: Series D (The Statistician)* 24.3 (1975), pp. 179–195.

[2] Martin Cooper, Simon de Givry, and Thomas Schiex. "Graphical models: queries, complexity, algorithms". In: *Leibniz International Proceedings in Informatics* 154 (2020).

[3] Marianne Defresne, Sophie Barbe, and Thomas Schiex. "Scalable Coupling of Deep Learning with Logical Reasoning". In: *Proc. of IJCAI-23*. Aug. 2023, pp. 3615–3623.

[4] Marianne Defresne et al. "Efficient Neuro-Symbolic Learning of Constraints and Objective". In: *arXiv preprint arXiv:2508.20978* (2025).

[5] Eugene Freuder. "In pursuit of the holy grail". In: *ACM Computing Surveys (CSUR)* 28.4es (1996), 63–es.

[6] Marin Vlastelica Pogančić et al. "Differentiation of Blackbox Combinatorial Solvers". In: *International Conference on Learning Representations*. 2020.

[7] Po-Wei Wang et al. "SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver". In: *Proc. of ICML24*. Vol. 97. PMLR, pp. 6545–6554.