Deep Neural Networks for Constraint Satisfaction Problems: an Initial Investigation for the N-Queens

Andrea Galassi, Michele Lombardi, Paola Mello, Michela Milano

Department of Computer Science and Engineering (DISI), University of Bologna
{a.galassi,michele.lombardi2,paola.mello,michela.milano}@unibo.it

Deep Neural Networks (DNNs) have recently been very successful in number of fields, such as image and text classification, natural language processing, or image reconstruction [1].

In this work, we run a first investigation to probe whether a DNN can learn how to construct solutions of a Constraint Satisfaction Problem (CSP), without any explicit, symbolic, information about the constraints themselves. Specifically, we train a DNN to extend a partial solution by making a feasible "move", i.e. a variable assignment. The term "feasible" refers here to global consistency, meaning that the move is guaranteed to lead to a complete feasible assignment. The network is trained on a subset of solutions, which are decomposed to yield a number of partial assignments.

The approach is also of practical interest: the trained DNN could then be used either to generate a solution autonomously, or as a heuristic to guide a more traditional search process. The latter approach allows to combine the network with a classical declarative model, thus obtaining a hybrid where (possibly different) constraints are enforced by the DNNs or by a combinatorial approach (e.g. Constraint Programming, or Satisfiability Modulo Theory). This setup has a chance to speed-up search, and allows to use the DNN to encode constraints that are either implicit in the past solutions or hard to model by a classical, declarative, approach (similarly to e.g. Empirical Model Learning [2]). The training set could be constructed based on historical solutions.

Given that our investigation is in its early stages, however, our interest at this point is chiefly scientific. In particular, we are interested in some fundamental questions that are challenging to address. We wish to determine whether a sub-symbolic approach such as a DNN can learn something of the abstract, symbolic, structure of a combinatorial problem, and to what extent a DNN can get creative and generate solutions not necessarily similar to those employed to build the training set.

As a benchmark we consider a classical CSP, i.e. the n-queen completion problem [3]. However, since we encode solutions as bit vectors without problem-specific assumptions, our design can be readily applied to other problems. Our final DNNs can generate solutions that (on average) violate 0.67 constraints out of 84, and can be repaired by moving 0.37 queens. As a term of comparison, a random generator (a proxy for what can be done without access to problem knowledge) violates more than 11 constraints and requires moving more than 4.7 queens. Even more interestingly, the DNNs are three times more likely than random to generate a *globally consistent* move, when fed with a partial assignment *that does not come from a solution in the training set*. This is intriguing, as it may signal that the network actually managed to learn something of the overall problem structure.

Our setup is reminiscent of Reinforcement Learning with DNNs [4]. As a main difference, we rely entirely on off-line training: Reinforcement Learning systems eventually get to learn by actually making decisions and receiving feedback in a simulated environment.

This process may be too time consuming for a combinatorial problem, and it also requires a complete knowledge of the target system. Conversely, our training is performed entirely off-line, and relies only on the information available from a limited number of positive samples (i.e. the solutions used to build the training set). In [5] and [6], the authors propose a method to design a Hopfield network that encodes a known set of constraints for a fixed set of variables. The network state can then provably converge to a feasible solution. However, such method requires full knowledge of the problem constraints, rather than just a set of solutions.

We are aware that this line of research is at its early steps, and a number of important research questions remain open. For example, despite our approach is problem-agnostic, it remains to be proved whether DNNs can successfully learn something about more complex combinatorial problems. Moreover, our design requires problems of well-defined size (i.e. number of variables):

this may be a serious issue in practice, although the method could still be viable for combinatorial problems defined over the same physical system.

In general, tests on practical applications and comparisons with state of the art solution approaches are top priorities, and the same holds for the definition of methods to integrate our DNNs in an actual, search-based, solution approach. Finally, we need to refine our analysis techniques to asses how "abstract" the knowledge of the DNNs can be, and whether the way the solutions are constructed resembles any known search heuristic.

References

1. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org
2. Lombardi M., Milano M., Bartolini A.: Empirical decision model learning. Artif. Intell. 244: 343-367 (2017)
3. Gent I. P., Jefferson C., Nightingale P.: Complexity of n-Queens Completion. Journal of Artificial Intelligence Research, 59, 815-848 (2017)
4. Sutton R. S., Barto A. G.: Reinforcement learning: An introduction. MIT press (1998)
5. Adorf H. M., Johnston M. D.: A discrete stochastic neural network algorithm for constraint satisfaction problems. IJCNN International Joint Conference on Neural Networks, San Diego, 3, 917-924 (1990)
6. Wang C.J., Tsang E.: Solving constraint satisfaction problems using neural-networks. Proceedings of IEE Second International Conference on Artificial Neural Networks: 295-299 (1991)