Neural Network Heuristics for Numeric Planning: A Preliminary Study

Valerio Borelli[1,2,*], Alfonso Emilio Gerevini[1], Enrico Scala[1] and Ivan Serina[1]

[1]University of Brescia, Italy
[2]Sapienza University of Rome, Italy

Abstract

Heuristic search is a key technique in almost all types of automated planning approaches. Various works have shown that black-box approaches, such as neural networks and deep neural networks, can be used to learn an heuristic competitive with the state of the heuristics for classical planning problems [1, 2, 3]. However little to no work has been done regarding numeric planning problems. In our work we are investigating if similar methods can also be applied to numeric planning problems, and how they can be improved in a numeric planning context.

Keywords

Heuristic Search, Automated Planning, Mixed Discrete and Continuous Domains

1. Starting point

The starting point for our work has been how the model defined in [1] can be applied to a numeric planning context.
The model takes as an input a state, and return a heuristic value as output, such a value can be used as a heuristic function for a specific planning problem.
We will now discuss briefly about all the components that must be defined, and what the results seen up to this point are.

1.1. Planning Framework

A numeric planning problem is a tuple $\Pi =< s_0, A, G, V >$ where V is the set of variables, which can be either propositional or numeric, $s0$ is the initial state of the problem, A is a set of actions, either numeric or propositional, and G is the goal of the problem, which consists in a set of propositional and numeric goal conditions.
An action $a \in A$ is a pair $< Pre(a), Eff(a) >$, in which $Pre(a)$ represents the set of preconditions that must be verified in a state s before executing the action, and $Eff(a)$ represents the set of effects of the action, that will be applied to the actual state to reach the next state.

AIxIA 2023: 22nd International Conference of the Italian Association for Artificial Intelligence,November 6-9, 2023, Rome, Italy

*Corresponding author.

valerio.borelli@unibs.it (V. Borelli); alfonso.gerevini@unibs.it (A. E. Gerevini); enrico.scala@unibs.it (E. Scala); ivan.serina@unibs.it (I. Serina)
0009-0007-3833-4654 (V. Borelli); 0000-0001-9008-6386 (A. E. Gerevini); 0000-0003-2274-875X (E. Scala); 0000-0002-7785-9492 (I. Serina)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

A plan $\pi =< a_1, ..., a_n >$ is a sequence of actions, and is considered a solution for the planning problem if applying the sequence of actions, starting from the initial state of the problem, leads to a final state in which all the goals of the problem are satisfied.

1.2. Model of the network

The starting model for the neural network, as previously said, is the one defined in [1].
The model is a multi-label classification network [4], in which the inputs of the network are states generated extracted from a valid plan. The outputs are represented with a unary encoding, which basically consists in a multi-label approach with ordered labels [5], and represents heuristic values.
Regarding hyperparameters, the model has three hidden layers, with a number of neurons that scales in equal size steps from the input layer size to output layer size. Therefore, the number of trainable weights is strictly correlated to the number of non-static facts of the planning problem and the maximum heuristic value that we want to have as output.
All the layers in the network use sigmoid activation functions, without regularization.

1.3. Creation of the samples

In order to generate enough samples to train the neural network, for each task we use a Random Walk to generate new initial states for that task. Then, for each initial state generated in this way, the teacher search, which in our case is ENHSP [6] using *GBFS* as search algorithm and h^{add} [7] as heuristic function, solves the task. For each plan found this way, one state is selected randomly and used as a sample for the neural network.

Greedy best-first search (*GBFS*) [Doran and Michie, 1996] is a suboptimal search algorithm. It takes a greedy approach to explore the state-space, always prioritizing the node with the lowest heuristic value.
h^{add} [8] is an inadmissible heuristic, that considers all the subgoals of a problem independent. In our case we use a version of h^{add} generalized for numeric planning problems [7].

1.4. Preliminary Results

Preliminary results on the previously described models show that, compared to classical planning, the heuristics defined this way works much worse with respect to traditional numeric planning heuristics.
In particular, while in classical planning the generated heuristics, despite being slower than traditional ones, still manages to expand a smaller number of nodes, in the numerical case this difference does not exist with the current model.

Preliminary experiments have been conducted on two numerical domains: block grouping and counters [9], the first consists of grouping blocks of the same colour in the same cell, while the latter aims to find a specific numeric value for each defined counter, with the purpose of satisfy all the numeric constraints between the counters.
The results, as said before, show that the described model cannot learn a competitive heuristic.

In particular, in the case of block grouping, the heuristics found works worse than *hadd* both in terms of time and in terms of expanded nodes. In the case of counters the results are even worse: the model fails to apprehend and predict the heuristic value methodically, and therefore, using the resulting heuristics the planner often can't find a plan.

2. Future Work

In order to assess if what was done for classical planning can also be achieved for numeric planning, there are two possible directions, complementary to each other.
The first direction is to understand if the model can be improved, first and foremost via hyperparameter optimization.
Given that the starting model was built to work with boolean values, it is natural to think that a similar model with numerical values could have very different best configurations.
Furthermore, we have no guarantees that a multi-label classification network is still the better solution. Different approaches may work better in this case, such as a regression model, or a multi-classification model with a one-hot encoding to interpret the output.
Lastly, different approaches to generate the samples might be taken into consideration, such as taking as samples all the states of the generated plans.

The latter is to find numeric problems in which traditional heuristics struggles to guide the planner correctly. For example one domain that is particularly challenging for traditional heuristics, is called Settlers [10]. This domain revolves around the management of resources, with the objective to construct a variety of structures in specific locations. Its difficulty lies in the fact that actions often have complex interactions and dependencies, which traditional heuristics often fail to detect, due to the approximations on which they are based.

References

[1] P. Ferber, M. Helmert, J. Hoffmann, Neural network heuristics for classical planning: A study of hyperparameter space, in: ECAI 2020, IOS Press, 2020, pp. 2346–2353.

[2] R. Karia, S. Srivastava, Learning generalized relational heuristic networks for model-agnostic planning, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 8064–8073.

[3] W. Shen, F. Trevizan, S. Thiébaux, Learning domain-independent planning heuristics with hypergraph networks, in: Proceedings of the International Conference on Automated Planning and Scheduling, volume 30, 2020, pp. 574–584.

[4] G. Tsoumakas, I. Katakis, Multi-label classification: An overview, International Journal of Data Warehousing and Mining (IJDWM) 3 (2007) 1–13.

[5] J. Cheng, Z. Wang, G. Pollastri, A neural network approach to ordinal regression, in: 2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence), IEEE, 2008, pp. 1279–1284.

[6] E. Scala, P. Haslum, S. Thiébaux, M. Ramirez, Interval-based relaxation for general numeric planning, in: ECAI 2016, IOS Press, 2016, pp. 655–663.

[7] E. Scala, P. Haslum, S. Thiébaux, et al., Heuristics for numeric planning via subgoaling (2016).

[8] P. Haslum, H. Geffner, Admissible heuristics for optimal planning., in: AIPS, Citeseer, 2000, pp. 140–149.

[9] G. Frances, H. Geffner, Modeling and computation in planning: Better heuristics from more expressive languages, in: Proceedings of the International Conference on Automated Planning and Scheduling, volume 25, 2015, pp. 70–78.

[10] D. Long, M. Fox, The 3rd international planning competition: Results and analysis, Journal of Artificial Intelligence Research 20 (2003) 1–59.