Exploiting Learned Policies and Learned Heuristics in Bounded-Suboptimal Search

Matias Greco
Departamento de Ciencia de la Computación
Pontificia Universidad Católica de Chile
Vicuña Mackena 4860, Macul, Santiago, Chile
mogreco@uc.cl

Abstract

Machine Learning (ML) has made significant progress to perform different tasks, such as image classification, speech recognition, and natural language processing, mainly driven by deep learning. Also, ML algorithms, through learning policies or heuristics estimates, have demonstrated potential for solving deterministic problems that would usually be solved using search techniques. Nevertheless, in solving a search problem with purely learning techniques, it is not possible to deliver guarantees regarding the quality of the solution. This research explores how a learned policy or heuristic can be integrated with a bounded-suboptimal search algorithm using Focal Search, sorting the FOCAL list using the concept of discrepancies to speed up the search. On the experimental side, we train a simple neural network as a learned policy and the DeepCubeA as a learned heuristic for the 15-puzzle domain. The results show that a learned policy or heuristic can reduce, at least, one order of magnitude, the expansions than WA* with the same bound and deliver better solution quality.

Introduction

In recent years, Machine Learning algorithms have demonstrated potential for solving problems that would normally be solved using search techniques: for example, TSP (Vinyals, Fortunato, and Jaitly 2015), Sokoban games (Groshev et al. 2018), Real-Time Search (Muñoz et al. 2018), Moving blocks puzzles (Graves et al. 2016), Rubix Cubes (Agostinelli et al. 2019), etc. All these examples use a neural network configuration specifically designed and trained to perform a specific task.

Despite the fact that these models may show great performance, it is expected that the neural network will make mistakes, which may affect solution quality or lead into dead ends. By and large, exploiting learned (and imperfect) policies and heuristics to speed up search with a suboptimality bound is an open question.

In deterministic problems, a typical way to integrate a learned policy in search is to use the recommended action as preferred operator (Helmert 2006). Yoon, Fern, and Givan (2007) proposes to use Limited Discrepancy Search (LDS) with a learned policy and Orseau et al. (2018) proposes to use the policy and the deep of the solution as a score to choose an action, and delivering guarantees regarding the number of expanded nodes. However, none of these approaches deliver suboptimality guarantees and may deliver low solution quality. On the other hand, a typical way to integrate a learned heuristic is to use it in a Greedy-BFS or WA* (Agostinelli et al. 2019) scheme. However, because the learned heuristic is inadmissible, it is not possible to provide suboptimality guarantees.

This research proposes to use a learned policy and learned heuristic in a Focal Search (FS) procedure. We assume that policies are neural network classifiers that use a softmax activation in its output layer and deliver, for every action, the probability that the resulting state be part of an optimal path. On the other hand, for learned heuristics, neural networks deliver an inadmissible cost-to-go estimates.

FS is a well-known bounded-suboptimal algorithm that uses inadmissible heuristic functions, which may not be cost-to-go estimates. FS uses two lists: OPEN, which is the search frontier sorted in ascending order by $f = g + h$, and FOCAL, which is sorted arbitrarily. We propose different ways to exploit a learned policy or learned heuristic in the FOCAL list. The method consists of sorting the FOCAL list by an evaluation function (called $h_{\texttt{FOCAL}}$), which depends on the learned policy or heuristic. We present two methods to define $h_{\texttt{FOCAL}}$: score-based, which take advantage of the probability delivered by the neural network; and discrepancy-based, which use the discrepancy of a path, defined by how many times the path choose the recommended action (or the action which best cost-to-go).

On the experimental side, we train a simple neural network as a policy to solve 15-puzzle and also use a pre-trained model of DeepCubeA as learned heuristic. Our results show, in both schemes, that discrepancy-based FS yields the best results and outperforms the bounded-suboptimal algorithm Weighted A* (WA*) (Pohl 1970), both in terms of expansions and solution quality.

Learned Policies and Heuristics in FS

We assume that a *learned policy* is a function $\pi : A, S \rightarrow [0, 1]$ that maps each state-action pair to a probability, and therefore $\pi(a, s)$ is such that $\sum_{a \in A} \pi(a, s) = 1$, for every state s; and a *learned heuristic* is a function $h : S \rightarrow \mathbb{R}^{0,+}$, which maps a state s to a prediction of its cost-to-go.

Now, we define the two families of $h_{\texttt{FOCAL}}$, which are defined in terms of the learned policy or heuristic.

Score-Based $h_{\texttt{FOCAL}}$

For a learned policy, given a newly generated state s, which is about to be inserted into FOCAL, and its path is given by the parent relation from s_{start} to s is $\sigma(s) = s_1, s_2, s_3, \ldots, s_n$ where $s_1 = s_{start}$ and $s_n = s$. Hence, we define the following secondary heuristic as $h_{\texttt{FOCAL}}$:

$$h_{score-1}(s) = -\prod_{i=1}^{n-1} \pi(\lambda(s_i, s_{i+1}), s_i), \qquad \text{(Score-1)}$$

Where $\lambda : E \rightarrow A$ is a labeling function that associates each arc of the search graph with an action. Use Score-1 to sort FOCAL results in an algorithm that expands first those nodes maximizing the likelihood of the path generated by π, where the negative sign is used because FOCAL is sorted in ascending order.

An alternative is simply evaluate the last edge of the path $\sigma(s)$, namely (s_{n-1}, s_n), and consider the likelihood that the policy generates s_n from its parent, s_{n-1}. We define the following secondary heuristic, which can be interpreted as focusing just on the last arc and not take into account the likelihood in the path:

$$h_{score-2}(s) = -\pi(\lambda(s_{n-1}, s_n), s_{n-1}), \qquad \text{(Score-2)}$$

Discrepancy-Based $h_{\texttt{FOCAL}}$

For a learned policy, we define the discrepancy of a state as the number of times along the path $\sigma(s)$ in which the action recommended by the policy (i.e. the action with higher probability) was not taken. Thus, We define h_{disc} as:

$$h_{\text{disc}}(s) = \sum_{i=1}^{n-1} [a_i \neq \text{argmax}_{a \in A}(\pi(a, s_i))], \qquad \text{(Disc)}$$

$h_{\text{disc}}(s)$ is derived by the concept used in the Limited Discrepancy Search algorithm (Harvey and Ginsberg 1995). As originally conceived, the notion of discrepancy is defined for a path of states $\sigma = s_1, \ldots, s_n$. To compute σ's discrepancy we initialize our discrepancy counter to zero, iterate an index i from 2 to n, and increment the counter when there is a successor of s_{i-1}, different from s_i, which has an h-value lower than $h(s_i)$.

For a learned heuristic, we use the original definition of discrepancy, i.e. we define h_{disc} as the number of times along a path $\sigma(s)$ in which the state with best heuristic value was not taken. Thus, for a learned heuristic, we define $h_{\text{disc}} = \sum_{i=1}^{n-1} [s_{i+1} \neq \text{argmin}_{s' \in succ(s)}\{h(s')\}]$, where $[A] = 1$ if Boolean expression A evaluates to true, and $[A] = 0$ otherwise).

Empirical Evaluation

We test our algorithms on the 15-puzzle domain and compare against WA*. For a learned policy, we trained a simple neural network with 1.5 million examples extracted from 30 thousand optimal traces. The neural network has three hidden layers and a 4-dimension output layer with softmax as activation function, where each one represents an action. For the learned heuristic, we use the pre-trained model by DeepCubeA (Agostinelli et al. 2019). To evaluate the algorithms' performance, we use Korf's 100 search tasks (Korf 1985).

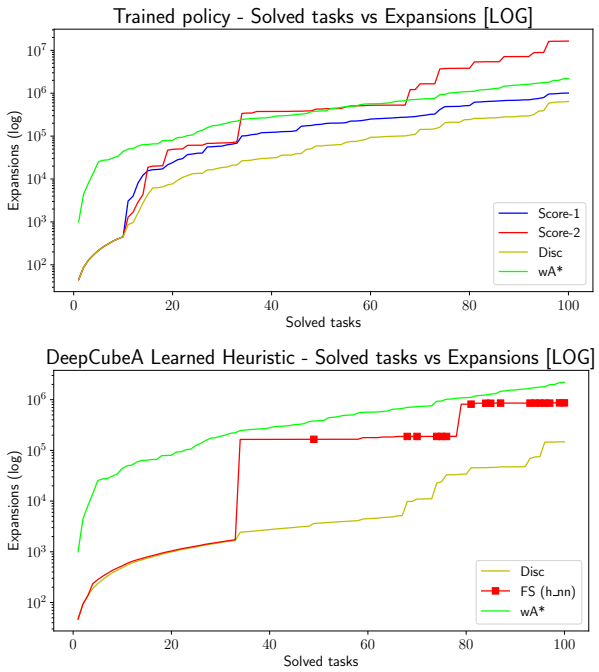

Figure 1: Results in 15-puzzle over the 100 korf's instances with a suboptimality bound $w = 1.5$

Figure 1 shows the results—in terms of expansions—using the learned policy and the learned heuristic with the suboptimality bound set to 1.5. Using the learned policy, the results show that *Score-1* and *Disc* outperform WA* by one order of magnitude with respect to the number of expansions. Using the learned heuristic, the results shows that use the learned heuristic to sort the focal fail to solve 16 of the 100 problems (marked with a red square) and *Disc* outperform WA* by almost two order of magnitude with respect to the number of expansions. Because the learned policy and the learned heuristic were trained using different techniques, the results can not compare between them.

Conclusions

This research presented two families of heuristics applicable to Focal Search when a learned policy or heuristic is available. These methods allow to exploit the learned policy or heuristic and provide suboptimality guarantees. The score-based $h_{\texttt{FOCAL}}$ seeks to maximize the probabilities provided by the network, instead discrepancy-based $h_{\texttt{FOCAL}}$ maximize the probability that its path it is a prefix of an optimal path. The results show that using FS with a learned policy or heuristic improves the solution quality and provides suboptimality guarantees.

References

Agostinelli, F.; McAleer, S.; Shmakov, A.; and Baldi, P. 2019. Solving the Rubik's cube with deep reinforcement learning and search. *Nature Machine Intelligence* 1(8): 356–363.

Graves, A.; Wayne, G.; Reynolds, M.; Harley, T.; Danihelka, I.; Grabska-Barwinska, A.; Colmenarejo, S. G.; Grefenstette, E.; Ramalho, T.; Agapiou, J. P.; Badia, A. P.; Hermann, K. M.; Zwols, Y.; Ostrovski, G.; Cain, A.; King, H.; Summerfield, C.; Blunsom, P.; Kavukcuoglu, K.; and Hassabis, D. 2016. Hybrid computing using a neural network with dynamic external memory. *Nat.* 538(7626): 471–476. doi:10.1038/nature20101. URL https://doi.org/10.1038/nature20101.

Groshev, E.; Goldstein, M.; Tamar, A.; Srivastava, S.; and Abbeel, P. 2018. Learning Generalized Reactive Policies Using Deep Neural Networks. In de Weerdt, M.; Koenig, S.; Röger, G.; and Spaan, M. T. J., eds., *Proceedings of the 28th International Conference on Automated Planning and Scheduling (ICAPS)*, 408–416. AAAI Press.

Harvey, W. D.; and Ginsberg, M. L. 1995. Limited discrepancy search. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research* 26: 191–246.

Korf, R. E. 1985. Depth-First Iterative-Deepening: An Optimal Admissible Tree Search. *Artificial Intelligence* 27(1): 97–109.

Muñoz, F.; Fadic, M.; Hernández, C.; and Baier, J. A. 2018. A Neural Network for Decision Making in Real-Time Heuristic Search. In Bulitko, V.; and Storandt, S., eds., *Proceedings of the 11th Symposium on Combinatorial Search (SoCS)*, 173–177. AAAI Press.

Orseau, L.; Lelis, L.; Lattimore, T.; and Weber, T. 2018. Single-Agent Policy Tree Search With Guarantees. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 3205–3215. URL https://proceedings.neurips.cc/paper/2018/hash/52c5189391854c93e8a0e1326e56c14f-Abstract.html.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, 2692–2700.

Yoon, S. W.; Fern, A.; and Givan, R. 2007. Using Learned Policies in Heuristic-Search Planning. In Veloso, M. M., ed., *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2047–2053.