nature machine intelligence

SUPPLEMENTARY INFORMATION

https://doi.org/10.1038/s42256-019-0070-z

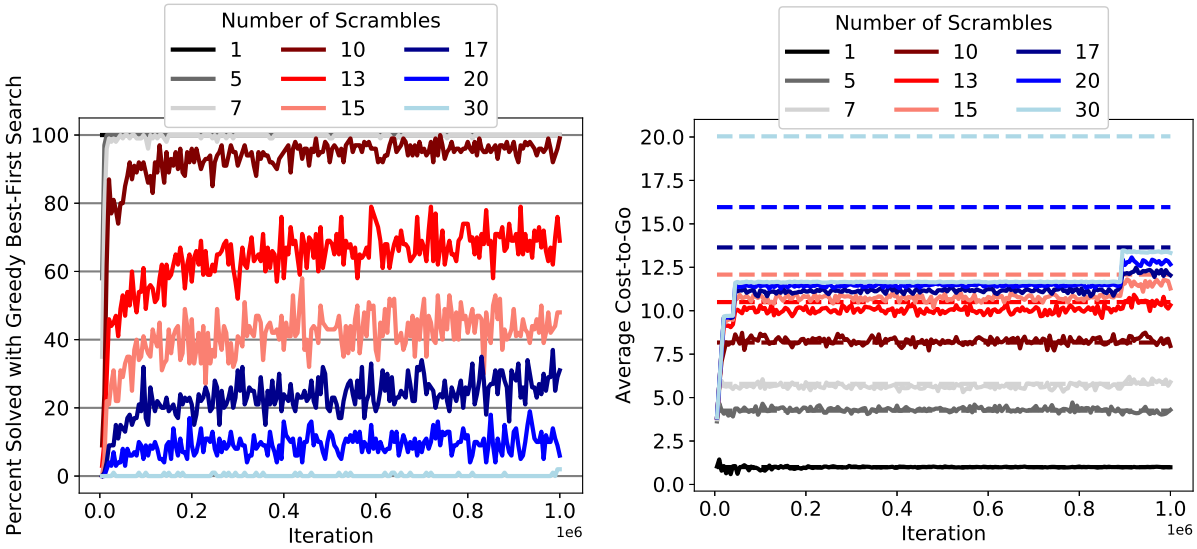
In the format provided by the authors and unedited.

Solving the Rubik's cube with deep reinforcement learning and search

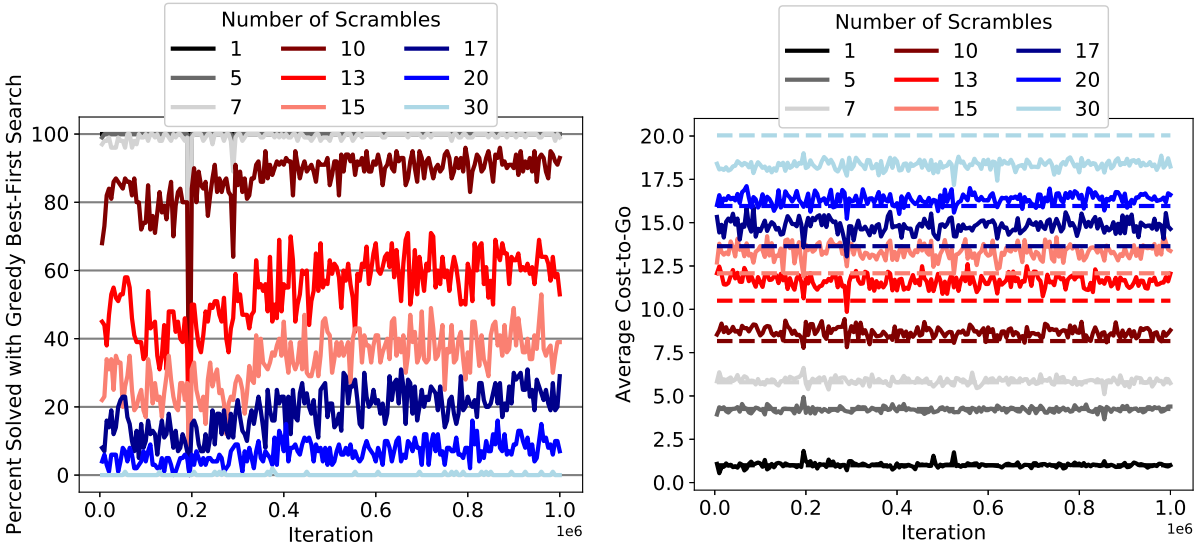
Forest Agostinelli[1,3], Stephen McAleer[2,3], Alexander Shmakov[1,3] and Pierre Baldi[1,2]*

[1]Department of Computer Science, University of California Irvine, Irvine, CA, USA. [2]Department of Statistics, University of California Irvine, Irvine, CA, USA. [3]These authors contributed equally: Forest Agostinelli, Stephen McAleer, Alexander Shmakov. *e-mail: pfbaldi@uci.edu

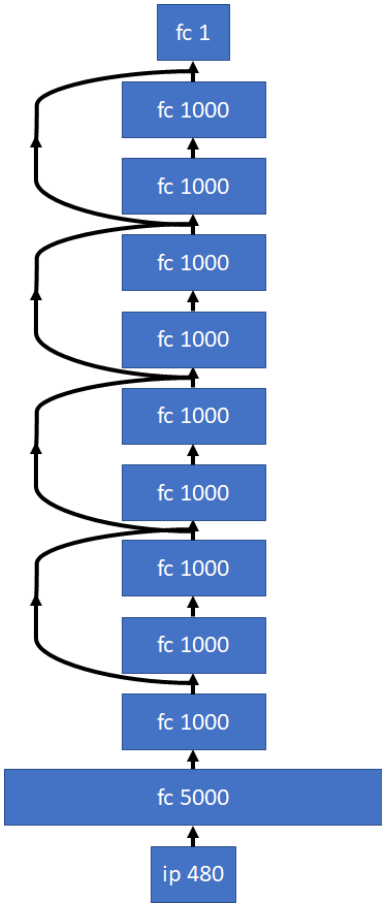
Supplementary Figures



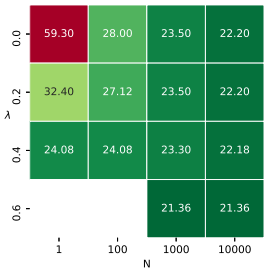
Supplementary Figure 1: The performance of DeepCubeA using a depth-2 breadth-first search. While a depth-2 breadth first search uses more information than a depth-1 breadth-first search, the plots show that the performance of a depth-2 breadth-first search is the same as DeepCubeA. However, it is considerably more expensive computationally. The dashed lines represent the true average cost-to-go.



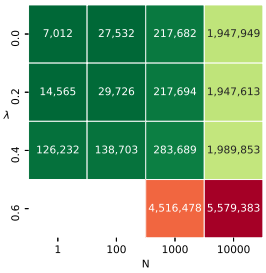
Supplementary Figure 2: The performance of imitating the optimal cost-to-go function. Attempting to imitate the optimal value function leads to, at best, similar performance to DeepCubeA. The dashed lines represent the true average cost-to-go.



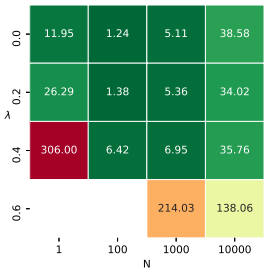
Supplementary Figure 3: The deep neural network architecture used by DeepCubeA. The cube state is entered in the input layer (ip) and activities are propagated forward through a series of fully connected layers (fc) to produce the cost-to-go estimate at the output layer.



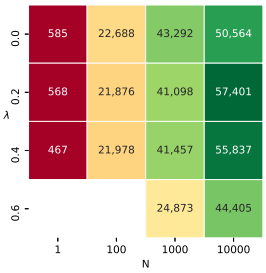
(a) Solution Length



(b) Nodes Generated

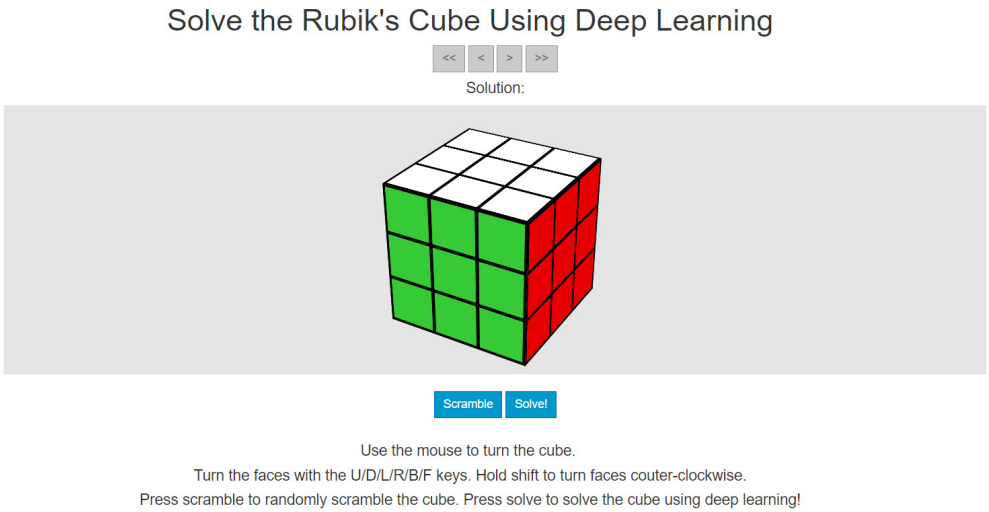


(c) Solve Time

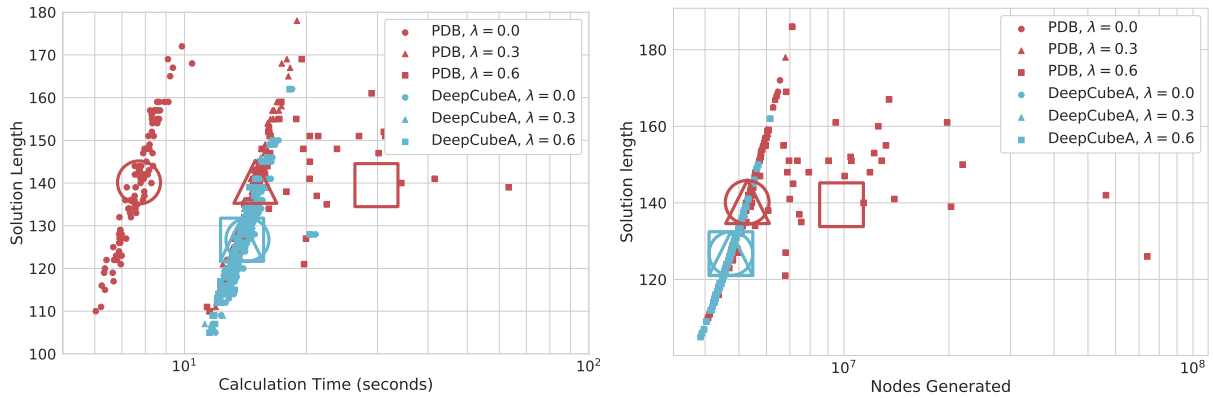


(d) Nodes/Second

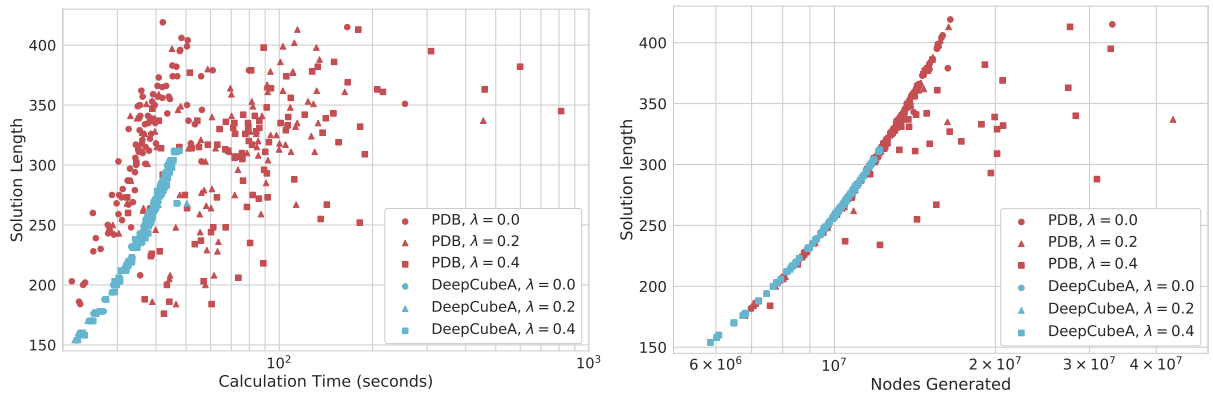
Supplementary Figure 4: Effects of BWAS hyperparameters on Rubik's Cube solving performance.



Supplementary Figure 5: A visualization of the DeepCubeA web server (http://deepcube.igb.uci.edu/).



Supplementary Figure 6: The performance of DeepCubeA vs pattern databases (PDBs) when solving the 35-puzzle with BWAS. $\mathrm{N} = 10,000$ and $\lambda = 0.0, 0.3, 0.6$. Each dot represents the result on a single state. DeepCubeA is almost always faster than PDBs and always produces shorter solutions. The large shapes represent the average of the respective run. The results show that DeepCubeA, on average, always produces shorter solutions and, on average, is faster than PDBs for two of the three assignments of λ.



Supplementary Figure 7: The performance of DeepCubeA vs pattern databases (PDBs) when solving the 48-puzzle with BWAS. $\mathrm{N} = 10,000$ and λ is either 0.0, 0.2, or 0.4. Each dot represents the result on a single state. DeepCubeA is generally faster than PDBs and generally produces shorter solutions than PDBs.