Combining Inadmissible Heuristics with a Neural Network

Herschel Darko

Motivation

A* is an algorithm that uses a heuristic to search for a solution to problems, When the heuristic is *admissible*, A* tries to find the cheapest or optimal solution but this can be expensive depending on how complex the problem is [1].

Real-world problems generally require practical solutions [2] and A* using an inadmissible heuristic will produce these solutions but how can we determine which heuristic would provide the nearest to optimal solution.

Proposal: Use a Neural Network to combine multiple inadmissible heuristics instead of trying to pick one, creating a new heuristic in the process.

Background

Admissible Heuristic vs. Inadmissible Heuristic

An heuristic is a function. It accepts a puzzle state and estimates the cost to reach a goal puzzle state. The lower the cost the closer the current state is to the goal state.

- *Admissibility* means this cost is never overestimated
- *Inadmissibility* means that the cost can be overestimated.

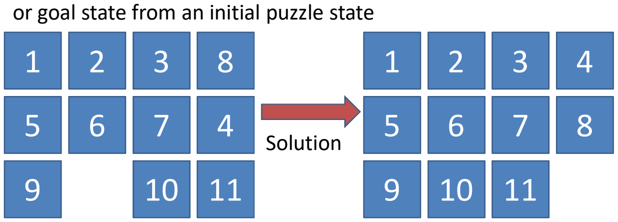
Neural Network:

Neural Networks are classifiers. They predict the output of the input data they are given, with this output either being a range of continuous values or discrete classes. Neural Networks learn by example. This means we train them with samples of input data along with the expected output.

For the paper, the output will be the moves that can occur in a sliding tile puzzle. Moving the blank Left, Right, Up or Down and the input will be the values the inadmissible heuristics output for a puzzle configuration or puzzle state.

11-Sliding Tile Puzzle

This problem consists of 11 numbered square tiles and one empty (blank) position. Any tile horizontally or vertically adjacent to the blank can be moved into the blank's position. The goal is to shift all the tiles into an ordered goal configuration or goal state from an initial puzzle state



Random Initial Puzzle State

Goal State

Methods

We will create an algorithm that uses our neural network heuristic and use it to solve a series of puzzles. The resources used and the solutions created will be compared to A* using other inadmissible heuristics .

A* will solve each puzzle three times and each time it shall use a different heuristic.

The solutions from our algorithm and A* will be compared in terms of cost and the amount of resources (memory) required to find the solutions.

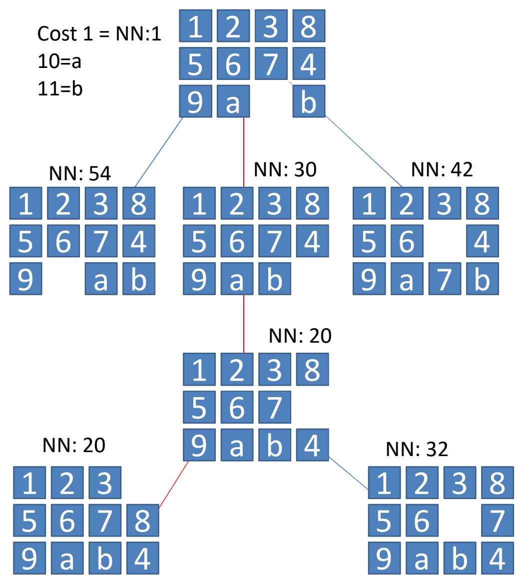
The inadmissible heuristics used by A* will be also be the ones combined by the neural network, this will show if neural network can outperform the individual heuristics used to train it.

Neural Network Heuristic

The neural network heuristic has 3 steps:

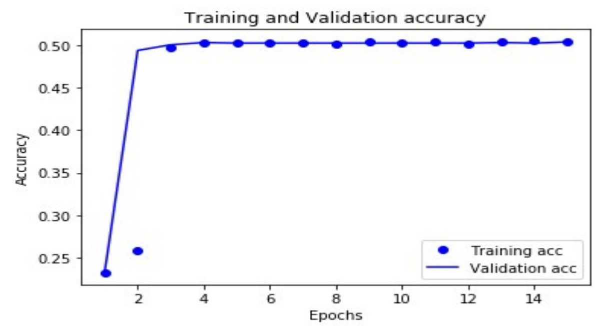
1. The network is given a vector of heuristic costs for a specific puzzle state.
2. The neural network then assigns a cost to the moves Left, Right, Up, Down.
3. The move it predicts the vector is most likely to map to will have the lowest cost with the other moves assigned ever increasing costs.

The algorithm that uses this heuristic will pick the legal move with the lowest cost.



Results

After training, the neural network could predict the accurate expected move approximately, 50% of the time when tested on individual puzzle states. When the experimental heuristic search algorithm was compared with A* , it was found that A* was superior with lower costs and lower resources needed with the individual heuristics.



Next Steps

We aim to improve the neural network's accuracy. Methods we are considering include trying a different problem domain and increasing the number of inadmissible heuristics used by the neural network

References

[1] - Malte Helmert and Gabriele Röger. How good is almost perfect? In AAAI, volume 8, pages 944–949, 2008.

[2] - Wheeler Ruml Ethan Burns and Minh Binh Do. Heuristic hearch when time matters. Journal of Artificial Intelligence Research, 47:697–740, 2013

Acknowledgements

This research was supported by Earlham College and its faculty.