Southwestern University

Evolution of Artificial Intelligence in Tetris Play Using Raw Screen Inputs

Gabriela Gonzalez and Lauren Gillespie

Introduction

Tetris is a popular and challenging video game released in that has been frequently used in artificial intelligence research. Artificial agents have been taught to play Tetris with controllers that pick the best placement for each falling piece with excellent results[1]. However, these controllers evaluate possible piece placements by focusing on the most relevant information about the game state, hand-picked by humans. In contrast, a human player simply uses raw data from the screen to decide which action to take next. This research attempts to learn computer agents to play Tetris solely with raw screen inputs, much as a human would, using the evolutionary algorithm HyperNEAT (Hypercube-based NeuroEvolution of Augmenting Topologies [2]).

Tetris

Tetris is an interesting domain for AI researchers due to its non-deterministic nature. Tetris is played on a 10 by 20 grid and players arrange and stack 7 distinct 4-block pieces known as tetrominoes. These pieces appear randomly one at a time then fall slowly from the top of the screen. The goal of the game is to completely fill horizontal rows with these pieces. Filled rows will disappear and increase the player's score. The more rows cleared at a time, the higher the score increases. This continues until the player is unable to keep the blocks below the top of the screen, at which point the player loses.

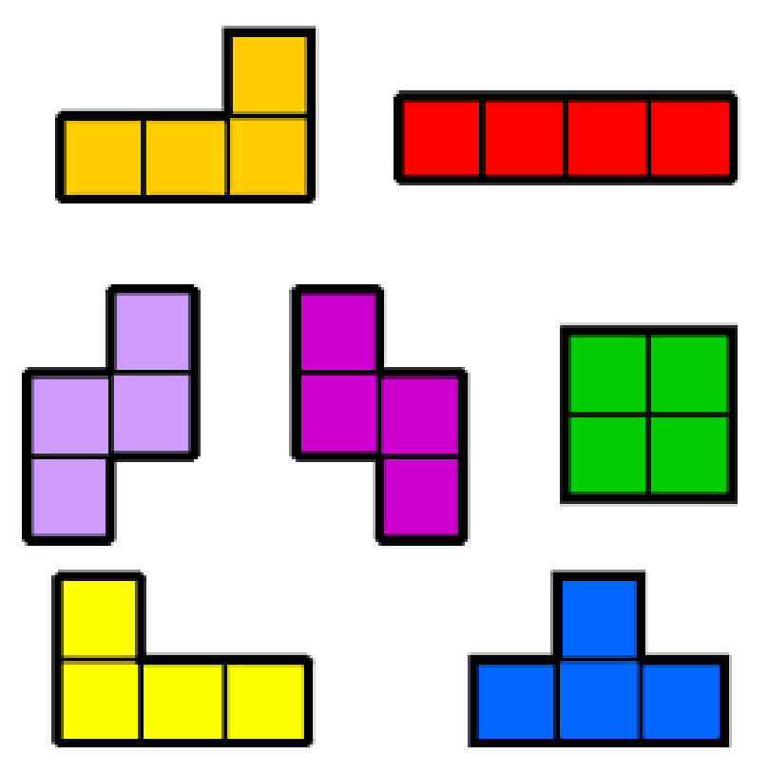


Figure 1. tetrominoes used in Tetris

HyperNEAT

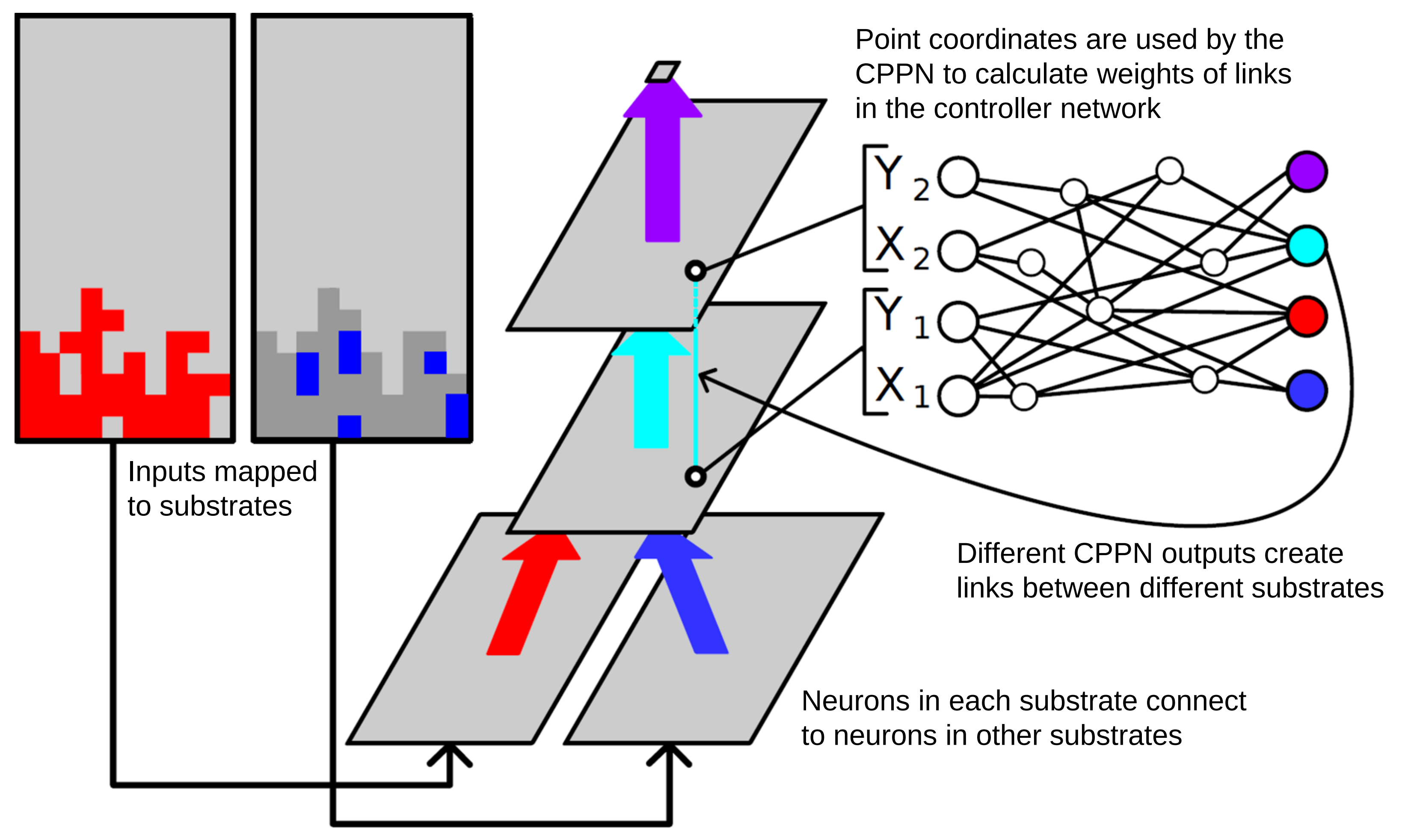
HyperNEAT is an extension of NEAT (NeuroEvolution of Augmenting Topologies), a popular algorithm for evolving artificial neural networks to act as controllers for AI agents [3]. HyperNEAT builds upon NEAT by evolving Compositional Pattern-Producing Networks which then create the actual controller neural network. The neurons of the controller network are organized into layers, called substrates, which define the potential structure of the network, giving the network a geometry unique to the domain. The CPPN uses the locations of these neurons in each substrate to create links between them, resulting in a controller network. In order for the networks to use raw screen inputs, the geometry of the substrates mirrors the geometry of the game itself and therefore the controller network is effectively visualizing the game screen, and is able to learn effectively.

Experiments

Learning approaches to Tetris typically use inputs that are human-designed to rate available moves. However, this research avoids heavy human guidance to allow an AI to learn the task on its own. This is accomplished by using HyperNEAT and only the raw screen inputs of a Tetris board.

The raw inputs consisted of two game screens; the first screen represented all blank spaces as 0's and placed blocks as 1's, and the second screen represented blank space under placed blocks from the first screen as -1's. This way, the agents know that "holes" exist, but must learn what they are.

Agents were evolved using both NEAT and HyperNEAT to play Tetris using only these raw inputs. Even though their controller networks used the same inputs, HyperNEAT's controllers are aware of the 2D layout of the inputs of the game. When a piece would appear on the board, the agent would find every ending placement, or after-state, for the piece. The agent would then use each resulting after-state to create a set of inputs for the neural network which would score the after-state to compare it to other after-states. The agent would choose the after-state that results in the highest utility score, and this process would continue until the agent lost or reached the time limit.


Point coordinates are used by the CPPN to calculate weights of links in the controller network
Y_2
X_2
Y_1
X_1
Inputs mapped to substrates
Different CPPN outputs create links between different substrates
Neurons in each substrate connect to neurons in other substrates
Figure 2. Application of HyperNEAT to Tetris domain

Results/Discussion

There were 30 runs of both NEAT and HyperNEAT in this experiment. Each run went up to 300 generations with a population size of 100 agents per generation. To get a reliable assessment of the skill of each Tetris player, each agent played the game 5 times. Average scores and average trial durations were used to select the best individuals to survive from one generation to the next. In total, each completed evaluation for either HyperNEAT or NEAT played 4,500,000 games of Tetris. Out of a combined total of 9,000,000 games of Tetris, we can see that HyperNEAT performs significantly better than NEAT using these raw inputs.

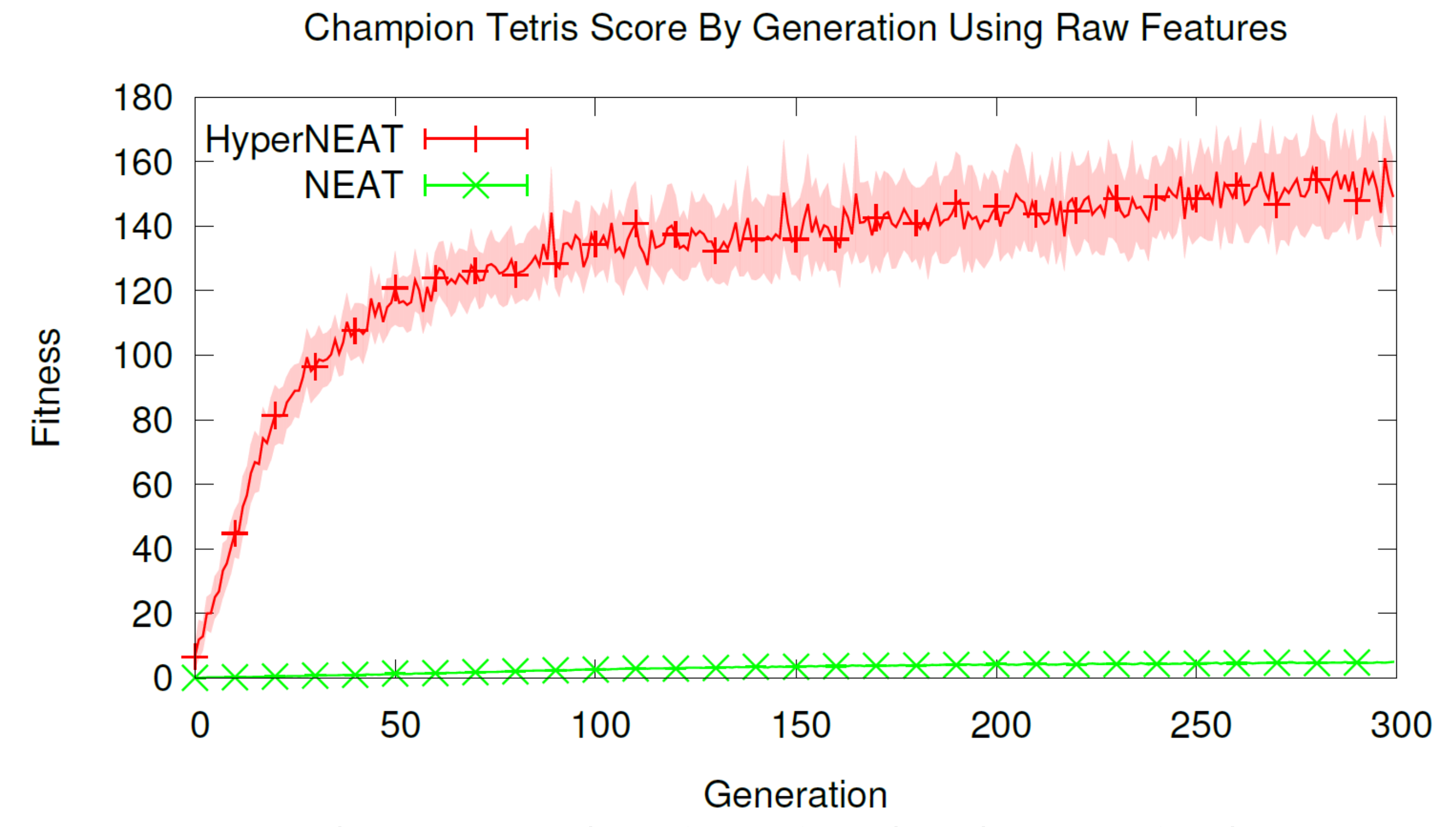


Figure 3. Comparison of score from champion agent of each generation from HyperNEAT and NEAT with raw features

While using raw inputs with HyperNEAT gets decent results, much better performance is possible with hand-designed feature inputs, be it with NEAT or other forms of learning. However, HyperNEAT can learn how to play Tetris competently with very little guidance. Artificial agents learning with raw inputs will be important for fully general intelligent agents in the future and learning with HyperNEAT is a step in that direction.

References

[1] C. Thiery, and B. Scherrer. Building Controllers for Tetris. In International Computer Games Association 32(1), 2009.

[2] K. O. Stanley, D. D'Ambrosio, and J. Gauci. A Hypercube-Based Indirect Encoding For Evolving Large-Scale Neural Networks. In Artificial Life Journal 15(2) MIT Press, 2009.

[3] K. O. Stanley and R. Miikkulainen. Evolving Neural Networks through Augmenting Topologies. In Evolutionary Computation Journal 10(2) MIT Press, 2002.

hhmi Howard Hughes Medical Institute