Automated backgammon player: learning through self-play

**1. Abstract**

Neural Network used widely in the modern society, especially in machine learning. Through Artificial Neural Network (ANN) could make machines think like a human and behave like a human. In this project, I would use ANN to train a model that would learn to play backgammon like a player through self-play not only depends on the powerful computer calculations.

**2. Introduction**

**Background:**

In 1992, Gerald Tesauro developed a computer backgammon program “TD-Gammon” using artificial neural networks (ANNs). The performance of TD-gammon was at a level not far below that of the best human players of the time, and the strategies explored by TD-gammon led to advances in the theory of backgammon play.

In 1997, Pollack and Blair demonstrated that a coevolutionary self-play approach to developing backgammon strategies could be successful using ANNs and hill climbing (no reinforcement learning necessary). The result demonstrated that the dynamics of backgammon are particularly suited to coevolutionary learning through self-play (unlike many other games, such as chess)(Pollack, J.B., & Blair, D., 1998). After this paper public post, Tesauro write a short comment to against that hillclimbing are worthy of further exploration, to determine if it really is incapable of learning nonlinear functions.

Recent developments in ANNs (in particular deep learning methods) have generated international headlines through successful game playing implementations such as AlphaGo, capable of beating the best humans. In addition, based on pervious professional research on backgammon through self-learning, we have multiple data to train an automated backgammon player.

**Aim:**

This project will attempt to replicate an backgammon player that had been developed by Pollack and Blair, successfully using an ANN trained through self-learning, beginning with a simplified version of backgammon, and moving on to a more realistic version of the game. As an extension, if time permits, deep learning approaches will be compared.

**Objective:**

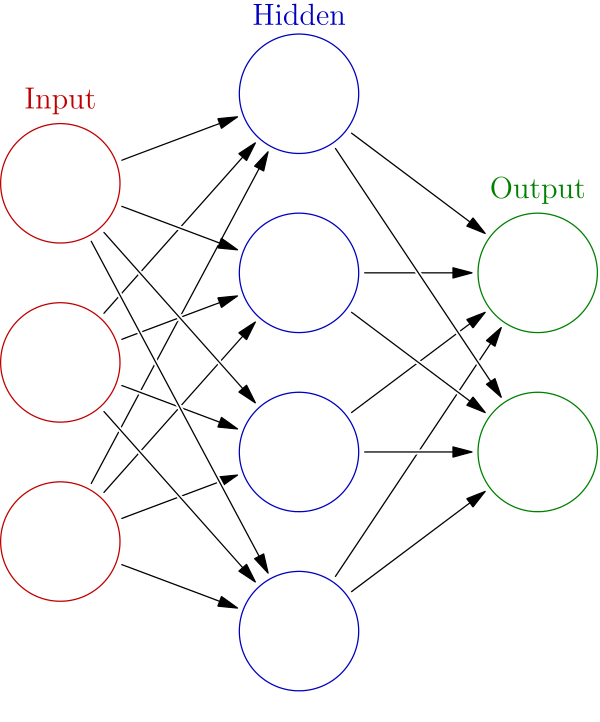
* Collect relevant paper and do research on the development
* Train a demo in Java to train
* Start self-play learning
* Evaluate the trained model through tests

**3. Proposed approach**

As for the language, after scanning some papers, most of they use Java to write the UI and not mention which language they used. Then they would use Matlab to figure out the data algorithms. Consequently, I also could use Java to write most part and then use other tools to help analyze. ANN would be the main method to train the model and may use other deep learning approaches would be compared.

**ANN**

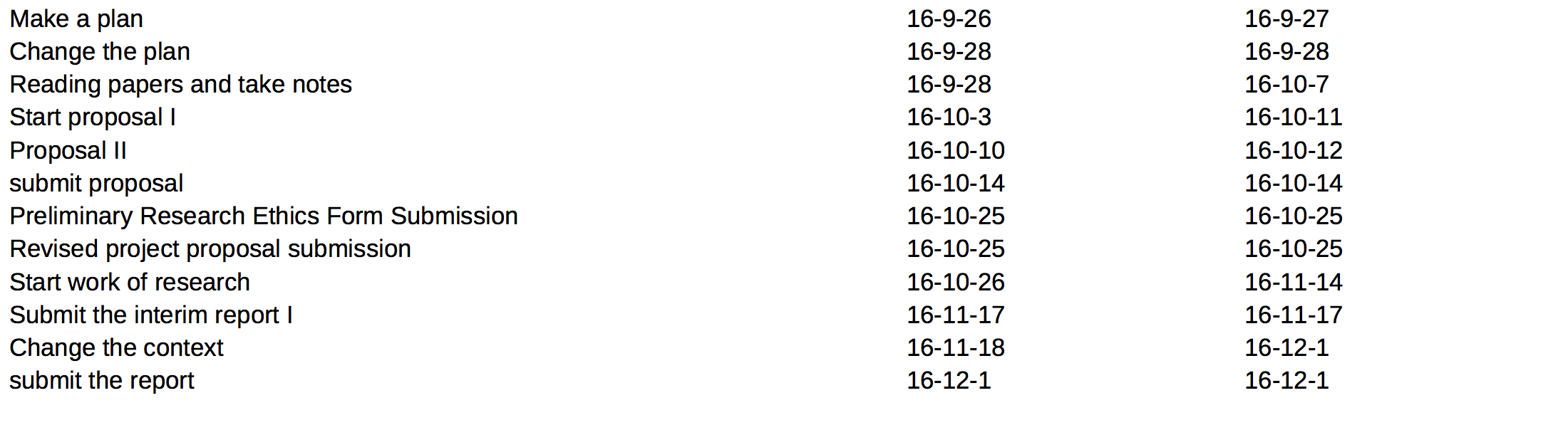
ANN contain multiply hidden layers with dynamic weights and then get a output from its learning. The weights would be modified after each train or test so only if the accuracy would drop from a high level to a lower level that this ANN model is useful.

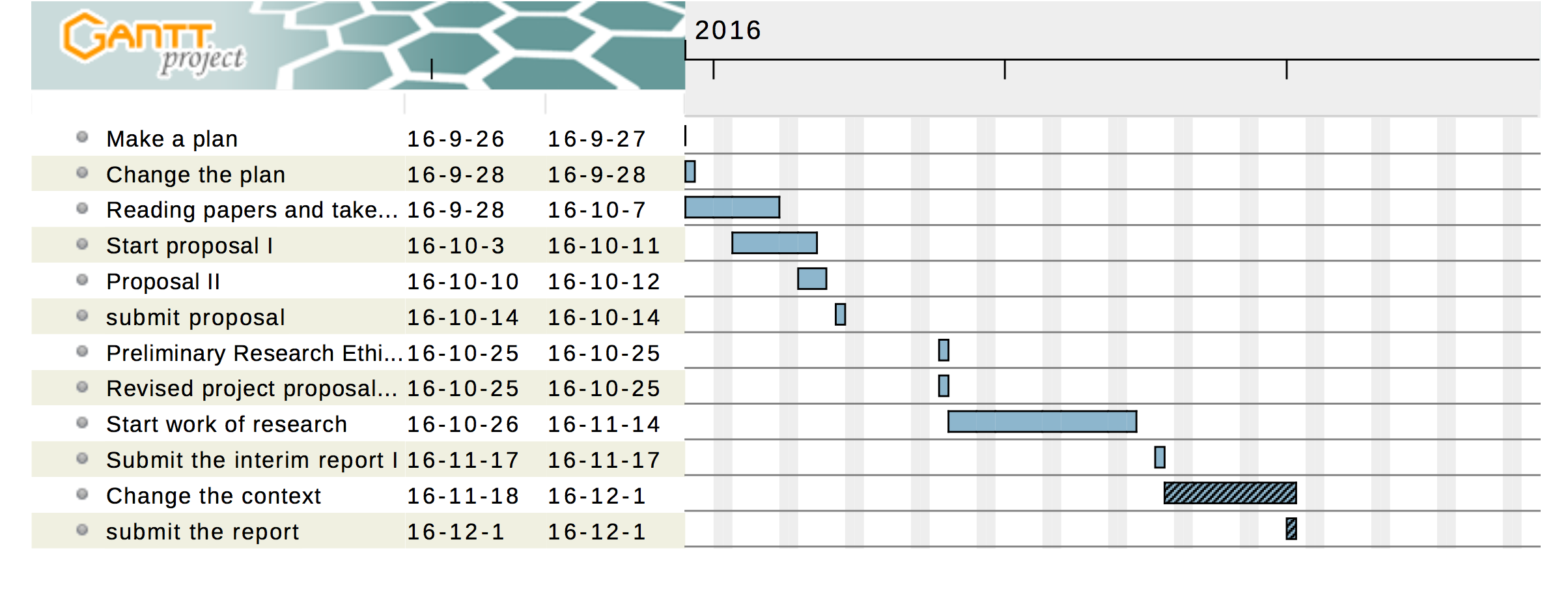


**Co-evolution**

Co-evolutionary algorithms are a class of algorithms used for generating [artificial life](https://en.wikipedia.org/wiki/Artificial_life) as well as for optimization, game learning and [machine learning](https://en.wikipedia.org/wiki/Machine_learning). It is widely used in self-learning that help machines to operated like a human.

**4. Project plan**

****

****

**References**

1. Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, *8*, 257–277.

2. Pollack, J.B., & Blair, D. (1998). Co-evolution in the successful learning of backgammon strategy. *Machine Learning*, 32.

3. Tesauro, G. (1998). Comments on Co-Evolution in the Successful Learning of Backgammon Strategy. *Machine Learning*, *32*, 241–243

.