AI Agent optimized by Artificial Bee Colony (ABC) Algorithm to play Cartpole Game

Ashwini Devendra Prabhu
Syracuse University
adprabhu@syr.edu

Abstract— ***The traditional Artificial Bee Colony Algorithm is used to train the neural network. The training of neural network involves a task of finding optimal weights and bias values. The Artificial Bee Colony (ABC) Algorithm increases the rewards gained over different evolution of the network policy parameters.***

The ABC algorithm-based policy network is then compared with Deep Q-Network (DQN) based agent to see the performance comparison of both the optimization methods. The DQN uses policy gradient approach to find the optimal policy network parameters while Evolutionary strategy approach uses the Artificial Bee Colony (ABC) algorithm.

Keywords— Reinforcement learning, Artificial Neural Network, DQN (Deep Q-Network), ABC (Artificial Bee Colony) algorithm

I. Introduction

A neural network's success mainly depends on adaptability, learning during its training by examples and to solve problems efficiently during its testing. All these factors make it difficult to optimize the parameters of an artificial neural network. The optimization process of artificial neural networks involves finding optimal weights and bias values so as to minimize the output error value.

The study of different insect behaviors, animal colonies and swarms has led to the introduction of many nature inspired optimization algorithms. These algorithms have a very consistence performance on neural network training. The Artificial Bee Colony (ABC) algorithm is one such swarm intelligence algorithm for global optimization based on the foraging behavior of the bees. This algorithm was proposed by Dervis Karboga.

This project explores the performance of the ABC algorithm for optimizing the connection weights of feed-forward neural networks for playing cartpole game and presents a comparison with the neural network implemented using DQN.

Problem Scenario

Cart-Pole (also known as Inverted Pendulum) is a classic game where pole is attached to a cart which moves along a track. The goal is to prevent the pole from falling over when the cart moves. The cart can move either to the left or right along the track.

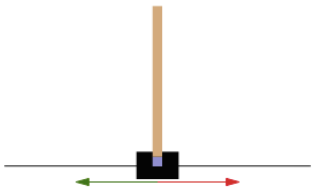

Fig 1: Cart pole Schematic Diagram [4]

Dataset

This project is based on a reinforcement learning problem. Hence, we do not have a dataset. For the agent's training purpose, it is allowed to interact with the environment so that it can retrieve episodes of interactions. Hence, we obtain the states, actions and rewards of each step. The reward is 1 at each time step when the pole is upright.

For evolutionary strategy, we ignore the presence of neural networks, environment, AI agent or any of its interactions. Using evolutionary strategy, we find the best policy parameters that helps the AI agent to get the best reward. The evolutionary strategy used here is the Artificial Bee Colony (ABC) algorithm, a swarm-based optimization algorithm that imitates the foraging behavior of honey bees. The fitness of the algorithm is the final reward obtained during an episode.

In the DQN based policy gradient approach we train the Neural Network using back propagation to find the optimal Q-table for the current game. Both the Neural Networks are different in sense that the DQN approach is optimized for the best actions while the ES based Neural Network directly optimizes the policy parameters for the game play.

II. Training Feed-Forward Neural Network

Reinforcement learning is a subset of Machine Learning where an AI agent has to learn how to complete a task by interacting with its environment. It involves Markov's Decision Process which is a decision-making loop. Markov's Decision process starts with the environment's initial state, predicts some best action for that current state and executes it on the environment. It then returns the reward and the environment's next state. Reinforcement learning problems can be solved using Deep Q learning method which aims at choosing the best action for the state.

A neural network mainly consists of an input layer, some hidden layers and an output layer. Each layer consists of a set of nodes or neurons which are interconnected. When the input is passed, it is multiplied with weight values. The summation of these weighted inputs is then passed on to the next layer. When we apply an evolutionary strategy to an artificial neural network, it means that we are trying to update the weights values of each neural network.

We initially create cartpole game's environment. This is provided by Open-AI Gym. Gym is a collection of environments used for developing and testing reinforcement learning algorithms. We then create an AI Agent with state and action values. The actions include pushing the cart either to the left or to right. The state includes factors like the position of the cart, angle of the pole, velocity of the cart and velocity of the pole. The neural network model consists of an input layer, one hidden layer and one output layer. Since our

agent needs to remember its previous actions for better performance, we implement experience replay. The results obtained from implementing DQN is shown in the Results section.

III. ABC Algorithm

The Original Idea -

A honeybee swarm consists of employed bees and unemployed bees. These bees search for food sources which are the flowers with nectar. Bees communicate through a waggle dance about the food. It dances in a figure of eight shape that contains the following information:

- The **Direction** of flower (angle between the sun and the flower).
- The **Distance** from the bee-hive (duration of the dance)
- The **Quality** rating (fitness) (frequency of the dance)

The Employed bees carry information about the food source and communicate the information with other bees waiting at the hive through the waggle dance. Unemployed bees are classified as onlooker bees and scout bee [2]. The Onlooker Bees will patrol the employees to verify when a specific food source is not worth it anymore. The Scout Bees will be the ones looking for new food sources locations. Employed and onlooker bees perform the **exploitation** search. Scouts carry out the **exploration** search.

Implementation

In each cycle of ABC algorithm, both global and local probabilistic search is implemented. Each cycle has a number of tasks which are performed by different bees. [1] The employed bees continuously try to locate better food sources in the neighborhood of their current food source by changing a randomly chosen dimension of their food source. The onlooker bees then probabilistically locate better food resource with the information provided by employee bees. Scout bees carry out global exploration of the search space by randomly choosing new food sources to initialize the algorithm, and to replace food sources that have been deemed exhausted when they have failed too many times to lead to improvements. [1].

The phases in Bee algorithm are as follows:

1. Initialization Phase

Control Parameters are set here. The following definition is used for initialization purposes

$$x_{mi} = l_i + rand\ (0,1) * (u_i - l_i)$$

x_m is the vectors of the population of food sources and is initialized as m = 1, 2, ... SN where SN is population size. Each x_m vector holds n variables, (x_{mi} where i=1...n), which are to be optimized so as to minimize the objective function. l_i and u_i are the lower and upper bound of the parameter x_{mi} [5].

2. Employed Bees Phase

Employed bees generate new food sources v_m having more nectar within the neighborhood of the food source x_m in their memory. They find a neighbor food source and then evaluate its profitability (fitness). Neighbor food source v_m using the formula given by equation:

$$v_{mi} = x_{mi} + \phi_{mi}\ (x_{mi} - x_{ki})$$

where x_k is a randomly selected food source, i is a randomly chosen parameter index
ϕ_{mi} is a random number within the range [−a, a]

After producing the new food source v_m, its fitness is calculated, and a greedy selection is applied between v_m and x_m. Fitness can be calculated using the formula given by equation:

$$fit_m(x_m) = \begin{cases} \frac{1}{1+f_m(x_m)} & if\ (fm(xm)=>0) \\ 1 + abs\ f_m(x_m) & if\ fm(xm) < 0 \end{cases}$$

If better fitness values are found, the new solution replaces the old one in the memory of that employed bee [5].

3. Onlooker Bees Phase

Employed bees share their food source information with onlooker bees waiting in the hive and then onlooker bees probabilistically choose their food sources depending on this information.

The probability values are calculated using the fitness values provided by employed bees. Probability value p_m with which x_m is chosen by an onlooker bee can be calculated by using the expression:

$$p_m = \frac{fit_m(x_m)}{\sum_{m=1}^{SN} fit_m(x_m)}$$

4. Scout Bees Phase

The scout bees search for food sources randomly. Employed bees whose solutions cannot be improved through a predetermined number of trials will be converted to the scout bees. The solution will be abandoned and is said to be "limited" or "abandonment criteria" [5]. The employed or the onlooker bees try generating new solutions around an old solution whereas the scout bee randomly searches for new solution in the x_{mi} vector again.

Ending Criteria

From the above description it is clear that the ABC algorithm has three control parameters that need to be set appropriately for the given problem: The bee colony size, the abandoning limit and the maximum number of search cycles.

Relating ABC algorithm to our Problem

The food resource to the ABC algorithm can be considered as our AI Agent parameters. Each agent is a neural network with weights and biases. It consists of n values. In the initialization phase we will be creating m number of random food resources i.e., m = 1, 2 … SN. Therefore, we have SN number of agents. In employee bee phase, we will be calculating the fitness of all the population. We will be then generating new agents (food sources) v_m with updated weights and biases. The fitness of v_m and x_m are compared and the one with higher fitness values are chosen. In the onlooker bee phase, a new solution x_m is chosen probabilistically. In the scout bee phase, the employee bee which cannot find an optimal food resource (which meets the abandonment criteria) is randomly initialized to a new food resource. The detailed ABC algorithm pseudocode [3] is as follows:

ABC Algorithm:

1: Generate the initial population x_m, m = 1 . . . *SN*
2: Evaluate the fitness (fit$_m$ (x_m))
3: set cycle to 1
4: **repeat**
5: **FOR** each employed bee {
Produce new solution v_m and calculate the value fit$_m$
Select food through Greedy Process}
6: Calculate the probability values p_m for the solutions (x_m)
7: **FOR** each onlooker bee {
Select a solution in x_m depending on p_m
Produce new solution v_m
Calculate the value fit$_m$
Apply greedy selection process}
8: **If** there is an abandoned solution,
then replace it with a new solution
9: Memorize the best solution so far
10: cycle=cycle+1
11: **until** cycle=maximum cycle number

IV. Results

Both the DQN and ABC algorithms are run for 100 iterations. The hyper-parameter values used during implementation are as in the table in Fig. 2. The Fitness Vs. Iterations graph plot obtained from implementation of Cartpole game through DQN method is shown in Fig. 3.

Parameters	***Values***
Learning rate	0.001
Exploration rate	1.0
Exploration Decay	0.995
Exploration Minimum value	0.1
Gamma value	0.95
Batch Size	20

Fig 2: Hyper Parameters used in DQN implementation

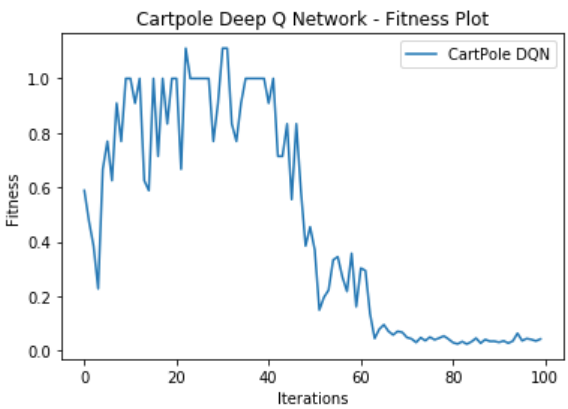

Fig 3: Deep Q Network Fitness Plot

The Fitness Vs. Iterations graph plot obtained after applying ABC algorithm to the feed-forward neural network is as shown in the Fig.4.

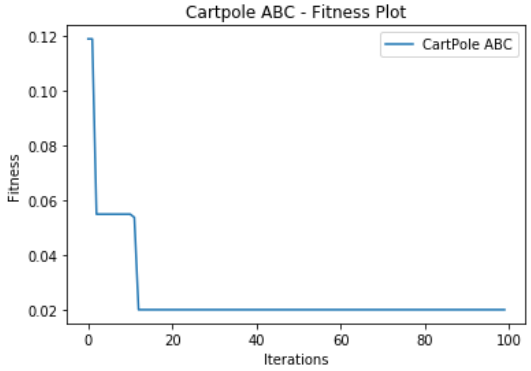

Fig 4: ABC algorithm Fitness Plot

Comparing both the plots, we see in Fig 5. that ABC Algorithm converges faster than the Neural Network model using backpropagation. Also, ABC Algorithm Neural Network Model gets better score than the DQN Neural Network model.

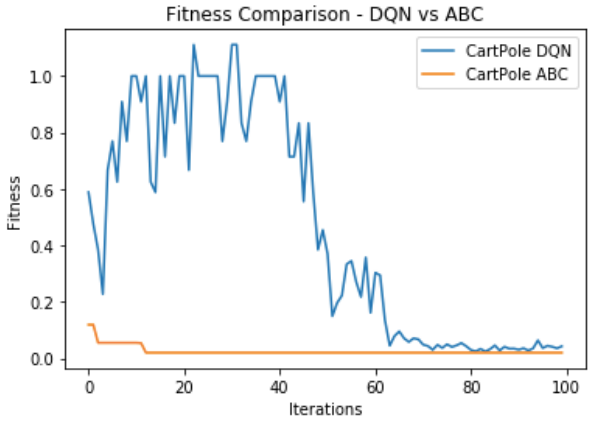

Fig 5: Comparison of DQN and ABC algorithm Fitness Plots

V. Conclusion

Recently, there have been a lot of improvements in evolutionary algorithms. A variety of nature inspired algorithms are available and also many hybridized algorithms have emerged which perform better compared to the traditional ones. In this work, the traditional artificial bee colony algorithm, which is simple and robust optimization algorithm, is used to train feed-forward artificial neural network for the cartpole game. The performance of the cartpole agent is certainly better when ABC algorithm is applied rather than just the DQN algorithm. However, the traditional algorithms tend to get stuck in the local minima and the global search techniques might capture the global minima very late. These issues are overcome by the hybrid algorithms which are introduced to apply to neural networks. As a future work, I would like to implement one of these hybridized algorithms and compare the performances.

References

[1] John A Bullinaria and Khulood AlYahya, Artificial Bee Colony Training of neural networks.
https://link.springer.com/chapter/10.1007/978-3-319-01692-4_15

[2] Behzad Nozohour Leilabady, Babak Fazelabdolabadi, “On the application of Artificial Bee colony (ABC) algorithm for optimization of well placements in fractured reservoirs; efficiency comparison with particle swarm optimization (PSO) methodology” https://www.sciencedirect.com/science/article/pii/S2405656115000723

[3] Ozturk, Celal, and Dervis Karaboga. "Hybrid artificial bee colony algorithm for neural network training." 2011 IEEE congress of evolutionary computation (CEC). IEEE, 2011. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5949602&tag=1

[4] Greg Surma, “Cartpole: Introduction to Reinforcement Learning” https://towardsdatascience.com/cartpole-introduction-to-reinforcement-learning-ed0eb5b58288

[5] Wikipedia, “Artificial Bee Colony Algorithm” http://www.scholarpedia.org/article/Artificial_bee_colony_algorithm#Eq-5