Scientific Research Publishing

American Journal of Computational Mathematics, 2022, 12, 209-215
https://www.scirp.org/journal/ajcm
ISSN Online: 2161-1211
ISSN Print: 2161-1203

Analysis of Soft Decision Trees for Passive-Expert Reinforcement Learning

Jonathan Martini, Daniel J. Fonseca

Department of Mechanical Engineering, The University of Alabama, Tuscaloosa, USA
Email: dfonseca@ua.edu

How to cite this paper: Martini, J. and Fonseca, D.J. (2022) Analysis of Soft Decision Trees for Passive-Expert Reinforcement Learning. *American Journal of Computational Mathematics*, **12**, 209-215.
https://doi.org/10.4236/ajcm.2022.122012

Received: March 8, 2022
Accepted: May 28, 2022
Published: May 31, 2022

Abstract

This paper explores the use of soft decision trees [1] in basic reinforcement applications to examine the efficacy of using passive-expert like networks for optimal Q-Value learning on Artificial Neural Networks (ANN). The soft decision tree networks were built using the PyTorch machine learning and the OpenAi's Gym environment frameworks. The conducted research study aimed at assessing the performance of soft decision tree networks on Cartpole as provided in the OpenAi Gym software package. The baseline performance metric that the soft decision tree networks were compared against was a simple Deep Neural Network using several linear layers with ReLU and Softmax activation functions for the input and output layers, respectively. All networks were trained using the Backpropagation algorithm provided generically by PyTorch'sAutograd module.

Keywords

Deep Learning, Soft Decision Trees, Passive Reinforcement Learning, Recurrent Neural Networks

1. Introduction

Artificial neural networks (ANNs) have historically been hard to dissect, and thus, they are conventionally considered as "black boxes" in terms of their internal workings [2]. This has barred ANNs from being widely adopted into various industries due to the lack of "transparency" on how they converge to a final recommendation or conclusion [3] [4]. Their black box stigma persists since ANNs are generally an approximation of available data to create a trend-based function to represent various cause and effect relationships in data regardless of the application [5]. This leads to uncertainty in possible predictions which could have unexpected negative effects when used in commercial applications [6].

Even though complete certainty is unachievable, there exist various methods to observe the decision-making process in ANNs. One such method is to dichotomize image recognition model filters to give context on how these networks process and transform data to make inferences. The application of such a method to reinforcement learning problems is not any different. Instead of sorting categories, actions are classified within a discrete action space [7] [8]. A problematic issue is that most networks used for Deep Q-Learning [9], a subset of reinforcement learning, generally share a lumped-layer like network model where extracting individual weights to determine learned inferences is not feasible. Soft Decision Trees effectively solve this problem as they have a filter at each node in a binary tree that uses softmax or sigmoid activation to determine the next filter to use in the network until the leaf nodes are reached. The leaf node returned contains the probability distribution for the classification of the analyzed data. Therefore, a soft decision tree can take the form of an 2^n different possible paths where n is the number of layers in the tree network. At any one time, a soft decision tree can mimic 2^n different ANNs reflecting a case-by-case analysis of the data being processed [10] [11] [12]. Applying these passive-expert like networks to various control problems can allow for analysis of the various inferences determined by the model to prevent unexpected negative consequences of bad predictions from an ANN that would be detrimental to commercial applications.

2. Soft Decision Tree Neural Networks

Soft decision trees are essentially a binary tree of nodes along a linear layer with an activation function that generates a probability distribution, specifically sigmoid or softmax. At inner nodes, this output is limited to 2, at leaf nodes it is limited to the desired output space. This is shown in **Figure 1**. The network uses a loss function, $L(x)$, that minimizes the cross entropy between each leaf, weighted by its path probability and the target distribution (Equation (1)) [1]. To prevent stagnation of the network and a dependency on only one path, a "penalty" is introduced which promotes using various paths to evaluate data in the network. This "penalty" is the cross entropy within the looked-for average distribution 0.5, 0.5 for the two sub-trees and the actual average distribution α, $(1-\alpha)$ (Equation (2) and (3)) [1]. Next, the sum of all the penalties is taken and multiplied by λ, a hyper parameter of the model, which determines the strength of the penalty applied to each layer. This hyper parameter decays proportionally to 2^{-N} for each layer $N \in n$. This technique was used to achieve better accuracy results [1].

$$L(x) = -\log\left(\sum_{\ell \in \text{Leaf Nodes}} P^{\ell}(x) \sum_{k} T_k \log Q_k^{\ell}\right) \tag{1}$$

$$C = -\lambda \sum_{i \in \text{Inner Nodes}} 0.5\log(\alpha_i) + 0.5\log(1-\alpha_i) \tag{2}$$

where,

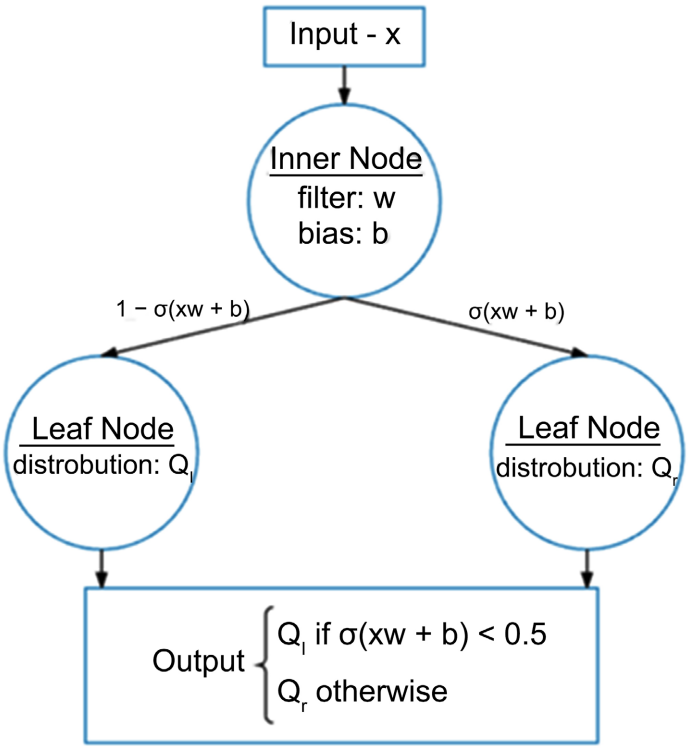

Figure 1. An overview of the structure of a soft decision tree [1].

$$\alpha_i = \sum_x P^i(x)\, p_{i(x)} \Big/ \sum_x P^i(x) \tag{3}$$

and $P^i(x)$ is the path probability from root the node 1.

3. Deep Q-Learning Reinforcement Learning

In this study, only deep Q-Learning methods were evaluated for soft decision trees on the various classical control problems implemented in the OpenAi's Gym package. Deep Q-learning uses an approximation of the value function $Q(s, a)$ through summing the reward at the current state with a discounted reward estimated by evaluating the next state in the environment via the model, and applying the discounting parameter γ (Equation (4)) [9] [13].

$$r_{n+1} = r_n + \gamma \max\left(Q\left(s_{n+1}, a_{n+1}\right)\right) \tag{4}$$

To fit the network to the highest rewarded state-action pairs, a replay buffer is created that randomly samples a queue of n length to generate a dataset with a state and a reward-weighted action output value. Each episode, which is collection of the state, action, reward, and next state, is record into the replay buffer. In this implementation, only the highest rewarded actions were stored in the replay buffer. After sampling and generating a state-rewarded action pair minibatch, the network was fitted to the minibatch. This was done repeatedly until the network converged. This process is demonstrated graphically in **Figure 2**. This attribute was arbitrary for the various control problems analyzed in this paper.

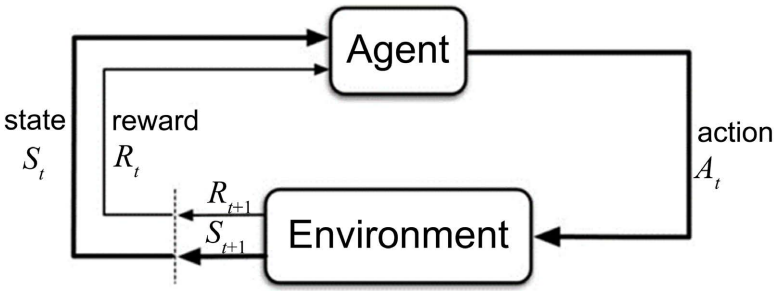

Figure 2. Reinforcement learning flowchart diagram.

4. Evaluated Models

A basic DQN (**Figure 3**) was evaluated as a baseline. It consisted of ReLU activations after every layer besides the last one, which was a Softmax activation. Only layers 3 and 6 had dropout layers which both were set at 0.2. The input space matched the state space of the cartpole environment which was 4. The output space matched the action space of the cartpole environment which was 2. The loss function used for the DQN network was the Mean Squared Error (MSE) Loss.

The soft decision tree configured for reinforcement learning was built using 6 layers and a penalty of 5.00E−4 (**Figure 4** and **Figure 5**). The loss function used for the soft decision tree is depicted in Equation (1).

5. Computational Implementation and Results

To help the models converge quicker, a variable reward function (Equation (5)) was used specifically for cartpole.

$$r_n = \left(\sum_{i=0}^{n} r_i - \left(\frac{x_n^2}{11.52} - \frac{\Theta_n^2}{288} \right) \right) \Big/ 200 \tag{5}$$

The variable reward function presented is a demonstration of the various preprocessing methods used to make pattern recognition in neural networks easier. Specifically, the function takes the accumulated rewards of the current episode and subtracts the square position and theta divided by their episode termination thresholds because those are the maximum values that trigger an environment reset. A target network was not used due to the difficulty of swapping weights with soft decision trees. Therefore, the convergence of the implemented Q-Learning algorithm is sporadic and random given a random seed, 1.

Usually a target and actor network is used to handle the randomness of the various states and constant updating of weights in deep Q-Learning. This is illustrated in **Figure 6**, where there are significant jumps in rewards between episodes which demonstrate how unstable and unpredictable this algorithm is for reinforcement learning without certain safe guards such as a target-actor network training scheme.

Overall, the results attained during this study are promising. The authors were able to show that the use of soft decision trees in basic reinforcement of DQN

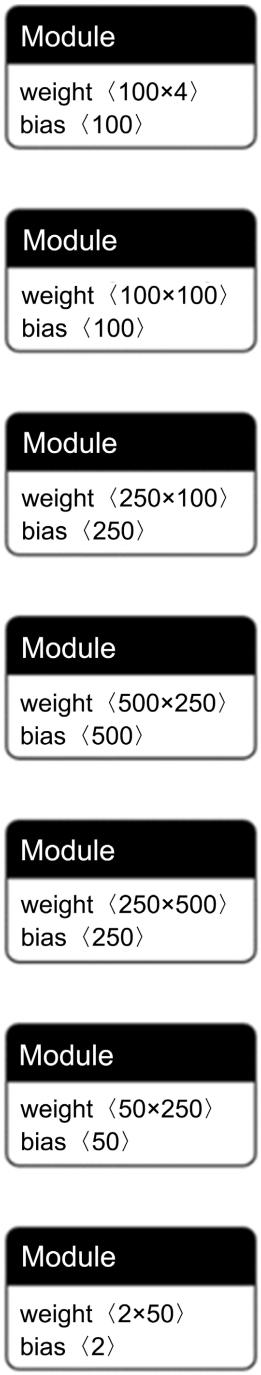

Figure 3. DQN network diagram.

SDT Param	Value
Depth	6
Input Dim	4
Output Dim	2
Penalty	1.00E-03
Weight Decay	5.00E-04

Figure 4. Soft decision tree configuration for deep Q-learning.

Q Params	Value
Discount γ	0.004
L-Rate	0.0001
Dropout	0.2
Replay Size	500
Mini Batch	12
Epsilon	100

Figure 5. Parameters for determining siscounted rewards, replay, and model training.

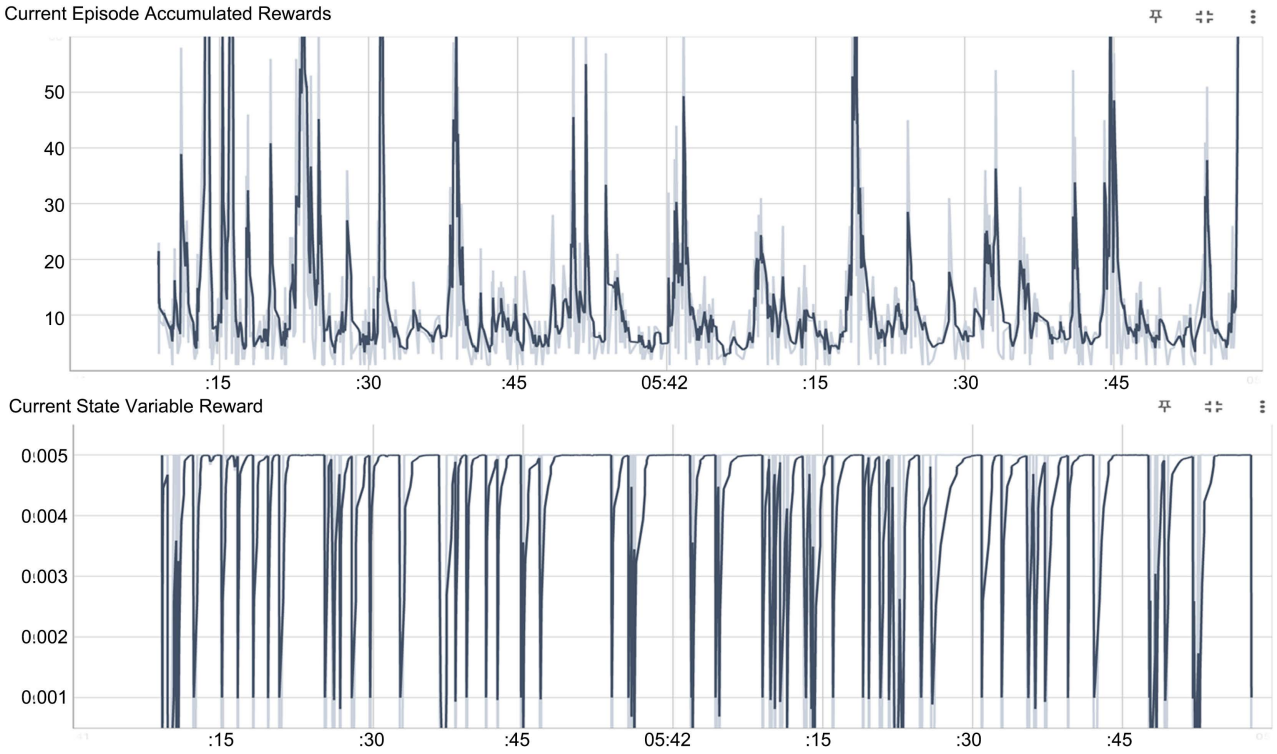

Figure 6. Pytorch tensorboard summary writer output graph for the baseline DQN Model reporting the accumulated and current state rewards versus time until convergence.

networks can help in understanding the inner workings of optimal Q-Value learning on the ANN models. However, further runs with the addition of carefully designed safe-guarding paradigms are needed to fully assess the efficacy of such an approach.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Frosst, N. and Hinton, G.E. (2017) Distilling a Neural Network into a Soft Decision Tree. ArXiv abs /1711.09784.

[2] Olden, J.D. and Jackson, D.A. (2002) Illuminating the "Black Box": A Randomization Approach for Understanding Variable Contributions in Artificial Neural Networks. *Ecological Modelling*, **154**, 135-150. https://doi.org/10.1016/S0304-3800(02)00064-9

[3] Fahimeh, G., Mehridehnavi, A., Pérez-Garrido, A. and Pérez-Sánchez, H. (2018) Neural Network and Deep-Learning Algorithms Used in QSAR Studies: Merits and Drawbacks. *Drug Discovery Today*, **23**, 1784-1790. https://doi.org/10.1016/j.drudis.2018.06.016

[4] Shahid, N., Rappon, T. and Berta, W. (2019) Applications of Artificial Neural Networks in Health Care Organizational Decision-Making: A Scoping Review. *PLOS ONE*, **14**, 1-22. https://doi.org/10.1371/journal.pone.0212356

[5] Castelvecchi, D. (2016) Can We Open the Black Box of AI? *Nature*, **538**, 20-23. https://doi.org/10.1038/538020a

[6] Dayhoff, J.E. and DeLeo, J.M. (2001) Artificial Neural Networks: Opening the Black Box. *Cancer*, **91**, 1615-1635. https://doi.org/10.1002/1097-0142(20010415)91:8+<1615::AID-CNCR1175>3.0.CO;2-L

[7] Zdolsek, G., Chen, Y., Bögl, H., Wang, C., Woisetschläger, M. and Schilcher, J. (2001) Deep Networks with Promising Diagnostic Accuracy for the Classification of Atypical Femoral Fractures. *Acta Orthopaedica*, **92**, 1-7. https://doi.org/10.1080/17453674.2021.1891512

[8] Fernandes, K. and Cardoso, J.S. (2018) Ordinal Image Segmentation Using Deep Neural Networks. 2018 *International Joint Conference on Neural Networks* (*IJCNN*), Rio de Janeiro, 8-13 July 2018, 1-7. https://doi.org/10.1109/IJCNN.2018.8489527

[9] Mnih, V. (2013) Playing Atari with Deep Reinforcement Learning. ArXiv abs/1312.5602.

[10] İrsoy, O., Yıldız, T. and Alpaydın, E. (2012) Soft Decision Trees. *Proceedings of the 21st International Conference on Pattern Recognition*, 1819-1822.

[11] Hua, Y., Ge, S., Li, C., Luo, Z. and Jin, X. (2018) Distilling Deep Neural Networks for Robust Classification with Soft Decision Trees. 2018 *14th IEEE International Conference on Signal Processing* (*ICSP*), Beijing, 12-16 August 2018, 1128-1132. https://doi.org/10.1109/ICSP.2018.8652478

[12] Hayashi, H. (2013) Neural Network Rule Extraction by a New Ensemble Concept and Its Theoretical and Historical Background: A Review. *International Journal of Computational Intelligence and Applications*, **12**, 1340006. https://doi.org/10.1142/S1469026813400063

[13] Mnih, V., Kavukcuoglu, K. and Silver, D. (2015) Human-Level Control through Deep Reinforcement Learning. *Nature*, **518**, 529-533. https://doi.org/10.1038/nature14236