Interpretability and performance comparisons of decision tree surrogate models produced by AGGREVATE

Jonathan Simon Wols
Supervisor(s): Anna Lukina

EEMCS, Delft University of Technology, The Netherlands
j.s.wols@student.tudelft.nl, A.Lukina@tudelft.nl

Abstract

Imitation learning algorithms, such as AGGREVATE, have proven successful in solving many challenging tasks accurately and efficiently. In practice, however, they have not been applied quite as much. Black box policies produced by imitation learning algorithms can not ensure the safety needed for real-world applications. This paper extends this field by outputting a decision tree surrogate model from AGGREVATE and comparing it to other imitation learning algorithms (Behavioral cloning, GAIL, DAGGER, VIPER) in terms of interpretability as well as performance. A modification to AGGREVATE is proposed to train decision tree policies that can be used to explain individual decision-making of the model. Three simple environments of open AI Gym have been used to compare the multiple different imitation learning algorithms. The experiments reveal that on performance, AGGREVATE overall performs better than the baseline behavioral cloning but slightly worse than GAIL, DAGGER and VIPER. AGGREVATE performs slightly better in terms of interpretability on these simple environments. Both of these conclusions could be explained by the fewer data points used by AGGREVATE. Further study can be done into the subjective interpretability of AGGREVATE as well as more difficult environments where the extra exploring of AGGREVATE should help with finding the best solution.

1 Introduction

Explainable reinforcement learning is the research field dedicated to making artificial intelligence processes, in this case specifically reinforcement learning, more interpretable and transparent [19]. The reinforcement learning models are often black boxes with no clear way of explaining what the thought process of the AI is [10]. The field of explainable reinforcement learning tries to solve this by making a surrogate model, such as a decision tree model or a rule based model, of the decisions made by the AI [23]. Human engineers can check if the AI makes the correct steps and conclusions and can modify the process to improve the overall quality of the AI processes.

There are multiple examples of machine learning systems which are used to make critical life-changing decisions. For example, checking steps would be very beneficial to the natural science field that has seen an increase in usage of machine learning but has not been able to use all of its scientific conclusions because of the lack of explainability in the black box process [20]. There are also applications in healthcare, the military and major engineering breakthroughs, like for example self-driving cars [7].

This lack of insight raises both ethical and judicial questions, leading to an increasing lack of trust in the black box system [7]. The AIs can not give the desired worst-case guarantees needed in those systems because their internal decision-making can not be checked by human engineers.

Explainability in the reinforcement learning field is a problem worth solving. Many complex reinforcement learning models are so called 'black boxes'. These models only return the result, they however do not explain why a decision was made and on what features of the data set the decision was based. Making the black box systems interpretable will increase the trust in the system for all parties involved.

Imitation learning is training a policy with the use of an expert. The policy will try to imitate the expert to get a reward as high as possible. Imitation learning is used when demonstrations are easier to find than to formulate difficult behavior. Some imitation learning algorithms use interactive expert access to question the expert while training, while other algorithms base their policy on samples provided without interacting with the expert.

In this project, the focus will be on a specific imitation learning method called AGGREVATE[21] and its interpretability in comparison to a Behavioral Cloning baseline and three other methods investigated by fellow students. These methods are *GAIL* [11], DAGGER [22] and VIPER [4].

AGGREVATE is an algorithm based on DAGGER with an extension focused on extra random steps to explore all possible paths using the expert behavior. The paper introducing this algorithm[21] focuses mainly on the implementation of the algorithm and its analysis. There is however barely any research into interpretability for this algorithm and its predecessor.

This project will help fill the interpretability gap by comparing it to different imitation learning algorithms. The algorithm has been recreated and further modified to produce a surrogate decision tree model. The model can then be compared on performance and interpretability to a behavioral cloning baseline and aforementioned other imitation learning algorithms.

The main research question is: *How do decision tree surrogate models produced by the* AggreVaTe *algorithm compare to a behavioral cloning baseline and other imitation learning algorithms in terms of interpretability and performance?* The following sub-questions are used to answer this question:

Which parameters is the AggreVaTe algorithm sensitive to? This will provide better insight into how the algorithm builds up its surrogate model and how the different parameters influence its outcome in terms of performance and interpretability.

On which tasks does AggreVaTe perform better than the other imitation learning algorithms? This will help to structure the comparison by checking specific tasks and therefore getting concrete comparison results.

We present a modification to AggreVaTe to train a decision tree surrogate model that can be compared in terms of interpretability and performance in comparison to other decision tree policies. The differences of the decision tree policies produced by different algorithms were investigated and reported. This research concluded that, on the given parameters, AggreVaTe in terms of performance performs slightly worse on simple environments than the other imitation learning algorithms but slightly better in comparison to behavioral cloning. In terms of interpretability, all imitation learning algorithms performed fairly similar but AggreVaTe overall has smaller decision trees due to the smaller number of data points. There were still distinct differences in the decision trees presented from the different algorithms that were discussed in this paper.

Related work

The paper introducing DAgger[22] proposes this new algorithm and verifies it theoretically as well as practically by comparing its performance to two other iterative machine learning algorithms. The extension on DAgger, utilizing cost of actions while learning, was introduced in the work of Ross and Bagnell (2014) as AggreVaTe, short for 'Aggregate Values to Imitate'. In that paper AggreVaTe is presented and theoretically analyzed. There is, however, no practical comparison to its predecessor or any other comparable imitation learning algorithm. The works on DAgger and AggreVaTe do not consider the interpretability of their algorithms in any way, not theoretical nor practical. There has been work on another imitation learning algorithm based on DAgger that does learn verifiable policies called Viper[4]. Viper outputs decision trees that are verified on correctness and stability but does not compare them to any other policies in terms of interpretability. There has been no practical comparison on AggreVaTe in terms of performance, nor has it been researched in terms of interpretability.

The next section focuses on the preliminaries concerning reinforcement learning, imitation learning and the difference between explainability and interpretability. After that, there will be a formal problem definition in section 3. Using that definition, the methodology will be explained in section 4. Section 5 explains the experimental setup and section 6 displays the corresponding results. Using these results section 7 will be used to discuss them and in section 8 the conclusions will be drawn, and future work will be discussed. Section 9 will reflect on the ethical aspects and methods used in this research.

2 Preliminaries

In this section all terms and variable definitions are given and also all decisions that have been made are explained. The section starts off with the preliminaries required to understand the algorithm. Then the base algorithm DAgger and its extension AggreVaTe are explained. After that, there is an explanation on what a surrogate model is and how one specific one was chosen. The final subsection explains the comparison metrics and the decisions on the baselines.

2.1 Variable definition

Before we continue on to the algorithm and its interpretability, it is useful to formally define the used variables first. In this research, only finite horizon control problems are considered. They are modeled as Markov Decision Problems, hereafter mentioned as MDP, with states s and actions a.

Denote $\prod$ as the class of all policies mapping states to actions considered by the learner. Define the expert as π*, the learner as π and the trained learner as $\hat{\pi}$. We assume the policy class $\prod$ contains good policies and that the expert π* is a good policy.

The cost function C(s,a) returns the immediate cost of performing action a in state s. The future cost-to-go function for executing action a in state s and then predicting the rest of the trajectory, t-1 steps, using some policy π is $Q_t^\pi(s,a)$.

The state distribution received by executing policy π in the MDP at time t is denoted as d_π^t. Denote the average state distribution over a time horizon of T as $d_\pi = \frac{1}{T}\sum_{t=1}^{T} d_\pi^t$. Combining the cost-to-go function and the state distribution denotes the total cost of executing policy π for T time steps as $J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{s\sim d_\pi^t}[C(s,\pi(s))]$, this can be used as the overall performance metric for cost comparison.

2.2 Algorithm

AggreVaTe is an extension on the algorithm of DAgger[22]. First the base algorithm of DAgger is explained, after which the idea and implementation of the extension will be denoted.

DAgger starts of with a data set D of the form [(s, π*(s)], this is initially empty. Then in the very first iteration of the algorithm the expert performs a certain number of trajectories and adds these to the empty data set D, after which the first

	Advantages	Disadvantages
Decision trees	Graphical model Subset of relevant attributes	Prone to overfitting Might include irrelevant values
Classification rules	Textual structure shows implications Better individual analyzing	Textual structure shows no importance distinction Worse full picture analyzing Possible conflicting classes
Decision table	Table can fill in instances with most frequent class matching the table cell	Requires a lot of space
Nearest neighbors	Explains classification of new instances	Not a model but an explanation
Bayesian networks	Graphical model	Even small models might be confusing

Table 1: Advantages and disadvantages of classification models

policy $\hat{\pi}$ gets trained which represents the expert as closely as possible. Then for N iterations use the last trained policy $\hat{\pi}_n$ to obtain new trajectories, after which the expert is queried for all states in these new trajectories to obtain new data D_i = (s, π*(s)) which get aggregated into data set D to train the new policy $\hat{\pi}_{n+1}$.

AGGREVATE improves on this algorithm by letting the policy take a certain number of steps after which a random exploratory action is taken. The expert will continue the trajectory for all possible actions giving a cost-to-go vector corresponding to all actions. With this data point added to the data set, the action with the minimal cost-to-go can be chosen to train the next policy.

This approach explores more trajectories that would otherwise not be found, leading to a more diverse data set and a potential to obtain a route that would not have been found by an imitation learning algorithm such as DAGGER. The cost-to-go given by the expert also gives the policy a better idea of which actions to take. Having a high cost-to-go for an action means that that trajectory will probably not recover from there, this action must be avoided at all cost if the policy still wants to recover.

For the full pseudocode of the base algorithm of AGGREVATE[21] see Algorithm 2 in the appendix, the underlined lines of code are added or updated in comparison to the DAGGER pseudocode.

The implementation of AGGREVATE consists of a similar iterative approach but differs to obtain samples. The execution of AGGREVATE starts, just like DAGGER, by initializing the empty data set D. The difference however is that the tuples entered into the data set are fairly different. DAGGER only had the state and its expert his prediction while AGGREVATE has the state, time step, action and a cost-to-go vector for all possible actions afterwards. In the very first iteration, AGGREVATE also queries the expert for trajectories to obtain data. This is where the random exploration actions start to come into play. In the trajectory at a random time t, in a state s a random action will be taken and the cost-to-go of the expert will be observed for the rest of the trajectory. From this data, the first minimized cost-to-go policy $\hat{\pi}$ will be trained. Now for N iterations the last trained policy $\hat{\pi}_n$ will be used to obtain a trajectory till a certain uniformly obtained time step t. From there a random action will be taken after which the cost-to-go is observed by the expert for all possible actions at that point. This generates a cost weighted data-point $D_i = \{(s, t, a, \hat{Q})\}$ which gets aggravated to D. This complete data set D will be the new basis for training the next cost-sensitive classifier $\hat{\pi}_{n+1}$ on.

2.3 Surrogate model

A surrogate model is a more simplistic model that mimics the input/output behaviour of the model that it represents. We use imitation learning to extract an explainable surrogate model from a black box expert model to give insight into the decisions made by the policy. We can use these surrogate models to compare on interpretability.

Different surrogate classification models found in a paper on comprehensible models[9] were considered. These models are decision trees, classification rules, decision tables, nearest neighbors and Bayesian network. The advantages and disadvantages of these models[9] are summarized in table 1. Nearest neighbors is not taken into account, since nearest neighbors explains classifications of new instances but does not provide any model function for interpretability.

From this table in combination with surveys comparing decision trees, classification rules and decision tables among different users[24; 1] the conclusion can be drawn that a decision tree is the best option for the surrogate classification model to compare in terms of interpretability.

2.4 Comparison metrics

The metrics for comparing different imitation learning algorithms can be split up in two parts, the more objective performance and the more subjective interpretability comparison.

Performance To obtain the Performance of an algorithm, we let the trained policy perform the trajectories a certain number of times. From these trajectories we obtain their rewards, which we can see as our data set. From this data set we can measure the average reward and how spread out this data set is. The average reward shows how the algorithm performs, and the standard deviation shows how consistent the algorithm is.

Interpretability Interpretability is inherently more difficult to compare on since interpretability has no, so called, mathematical rigour[14], this means that getting objective comparisons done is more difficult. Another difficulty is that all algorithms that are being compared on also output decision trees, which means that other metrics like local explanation, visualization and parameter comparison are very similar.

The most straight-forward objective metric for interpretability is the number of nodes, overall less nodes means less rules to go through to understand the tree, which leads to a more interpretable tree. There has also been a survey[8] that noticed the opposite while questioning several users, these users sometimes preferred larger decision trees if that meant that more informative attributes would be shown. They also disliked too small decision trees for complex scenarios, because they felt like there was critical information left out. For the experiments done in this research, with a small number of parameters and a simple environment, we choose to define fewer nodes as more interpretable.

Another metric that relates to total number of nodes is the max depth that it corresponds to, this parameter is set when the decision tree is initialized and does give more information on the former metric.

In this paper, only the interpretability metrics that can be objectively compared will be used. Subjective metrics are more useful to understand the trust in the algorithm that is required to understand if users would be comfortable using the imitation learning processes. Subjective comparisons, made using a survey, are however really time-consuming and hard to draw conclusions from due to less direct comparisons among users. The limitation with purely objective comparisons is that less nodes might still be less interpretable due to confusing predicates. Therefore, we will manually observe the decision trees and compare their performance in terms of interpretability.

2.5 Baselines

Before the imitation learning algorithms can be compared among each other, there should first be a baseline to compare to. Having this baseline allows more solid conclusions. The most basic imitation learning algorithm is behavioral cloning[2].

The way behavioral cloning works is as follows, an expert is used to obtain demonstrations consisting of state-action pairs. These pairs are used to train a policy using supervised learning by minimizing the loss function $L(a*, \pi(s))$. This can be achieved by using a classifier or regressor to replicate the expert[25]. Behavioral cloning is a simple approach that can work excellently in several situations.

Behavioral cloning has a very distinctive flaw. Once the policy makes a mistake and deviates from the expert trajectories, it might add up to new mistakes leading to a state that might not be in the data set obtained from the expert.

All imitation learning algorithms compared in this paper improve on this initial most simplified version in one way or the other, therefore it works perfectly as a baseline to compare improvements on.

Another baseline is necessary for the comparison of learning performance. For this, the expert Q-learning[26] is used. Q-learning was chosen since it is simple, reliable and the experiments performed are in a discrete action and state space. The Q-learning baseline is the expert that all imitation learning algorithms learn from, and is therefore interesting to compare to. Knowing how well the expert performed and then comparing to the algorithms and knowing whether the algorithms are far behind or maybe even ahead of the experts in terms of performance should provide valuable information.

To clarify, the first baseline compares the imitation learning performance while the second baseline compares the learning performance itself.

3 Problem definition

How do the imitation learning algorithms AGGREVATE, *GAIL,* DAGGER *and* VIPER, *who have been modified to produce decision tree policies, compare to a Behavioral Cloning baseline and each other in terms of interpretability and performance? Can we explain the difference in performance based on the decision trees that produce the results?*
AGGREVATE has been theoretically analyzed in the paper it was introduced in[21], there was, however, no practical comparison for this algorithm. DAGGER and AGGREVATE both were only researched as outputting black box policies that can not be interpreted by humans. Interpretability is a large gap in imitation learning research and is a problem worth solving since it can improve the trust in the algorithms significantly. The VIPER[4] algorithm already started improving in terms of interpretability by producing a verifiable policy in the form of a decision tree. To determine how AGGREVATE performs in terms of performance and interpretability, we must compare it to other imitation learning algorithms, leading to the following problems being investigated.

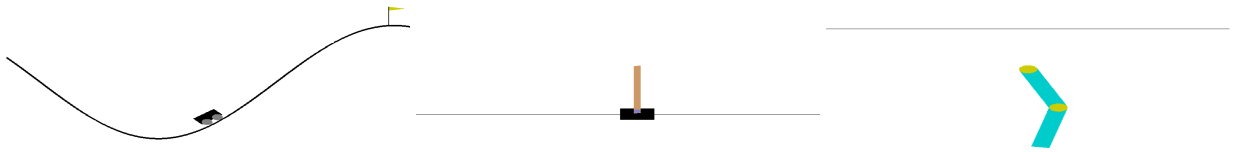
4 Methodology

Originally AGGREVATE does not specify what kind of policy model is outputted. To be able to compare in the best possible way on an interpretability scale, we had decided on outputting a decision tree as mentioned in section 2. Therefore, the code was modified to train a decision tree, for the new algorithm pseudocode see algorithm 2. This has been done using inspiration from the VIPER[4] paper, which also outputs a decision tree. To obtain a decision tree policy, we used a classification tree from scikit-learn[18] to train (state, action) tuples on. To get (state, action) tuples from the (state, time step, action, cost-to-go) data set we had to get the minimal cost-to-go action for every state and time step. We achieved this by resampling the dataset using cost-sensitive weighted classification[5] minimizing the expected cost and returning the corresponding state action pairs.

To compare the decision tree policies outputted by AGGREVATE to the other imitation learning algorithms, we use the metrics defined in section 2 as well as compare the decision trees on a more individual level. Looking at decision trees and their predicates can already explain a lot in terms of interpretability as well as performance. Knowing that one decision tree is performing better in terms of reward and standard deviation than the other can be explained by looking at the different decisions made at different locations in the trees.

5 Experimental Setup

For the experiments we use the openAI Gym toolkit[6]. This toolkit allows our comparisons to be on a universal



(a) Mountain car

(b) Cartpole

(c) Acrobot

Figure 1: Snapshots of the environments

Algorithm 1 AGGREVATE algorithm training a decision tree

```
Initialize $D \leftarrow \emptyset, \hat{\pi}_1$ to any policy in $\prod$.
for $i = 1$ to N do
    Collect m data points as follows:
    for $j = 1$ to m do
        Sample uniformly t $\in \{1, 2, ..., T\}$
        Start new trajectory in some initial state
        Execute current policy $\pi_i$ up to time $t - 1$
        Take exploration action $a_t$ in state $s_t$ at time t
        Execute expert from time $t + 1$ to T and observe estimate of cost-to-go $\hat{Q}$ starting at time t
    end for
    Get dataset $D_i = \{(s, t, a, \hat{Q})\}$
    Aggregate datasets: $D \leftarrow D \cup D_i$.
    Resample dataset D' $\leftarrow \{(s, a) \sim p((s, a)) \propto [C(s, a) \in D]\}$
    Train decision tree $\hat{\pi}_{i+1}$ on D'
end for
Return best $\hat{\pi}_i$ on validation
```

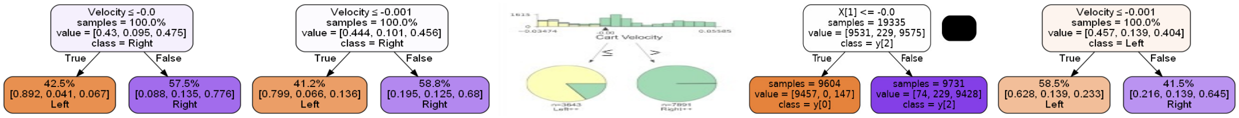
benchmark. The openAI Gym benchmark environments all return the variables needed to perform the algorithms described in section 2 and 4. The environments chosen can be seen in figure 1.

The first experiment and most simple experiment has been conducted on the mountain car environment, see figure 1a. The **goal** of the mountain car problem[16] is for the car at the bottom of the canyon to scale the right mountain to obtain the flag. The issue is that the engine of the car is not powerful enough to drive up the mountain from the lowest point of the ravine in one go, the solution is to drive back and forth building up momentum to have enough to get to the top. The **states** in this environment are defined as (x-position, velocity) tuples, where the x-position is -0.5 at the lowest point of the canyon and negative to the left and positive to the right of this lowest point. The velocity is positive if the car is moving to the right and negative when moving to the left. The **actions** possible are moving left, doing nothing and moving right. The **reward** obtained from the action is -1 for every time step taken when not arriving at the finish flag. The **time horizon** T is 200 time steps and therefore not arriving at the finish flag gives a total reward of -200, a decent run arrives at the flag in 130 time steps and therefore has a total reward of -130.

The second experiment, similar in difficulty, is on the cartpole environment[13]. The environment consists of a cart with a pole on top of it, see figure 1b. The **goal** of this environment is to balance the pole by moving the cart left and right to keep it from falling over for as long as possible. The **states** in this environment are larger than the mountain car problem, they consist of cart position, cart velocity, pole angle and pole angular velocity. This gives a four dimensional state space instead of two in the former experiment. The number of **actions** available however are smaller, there are only two actions available, either pushing left or pushing right. The **reward** obtained is different in comparison to the other experiments described in this section. For every time step taken, where the pole does not fall over, the reward goes up by one. The **time horizon** T is 500 time steps, this horizon gets hit when the policy is trained to never get out of balance.

The third, more difficult, experiment is on the acrobot environment[17]. The acrobot is modeled after a gymnast on a parallel bar, see figure 1c. The upper link corresponds to the upper body and the lower one to the legs. The upper joints can be seen as the wrists and the joint in the middle as the waist. The **goal** of this problem is to get the lower link to a certain height. Initially, the links are positioned straight down and therefore the environment requires the links and joints to work together to swing and pivot left and right to get to the required height. The **states** in this environment are even larger than the cartpole experiment, they consist of the cosine and sine of both rotational joints and the velocity of the angular joints. This gives a six dimensional state space. The **actions** available are similar to mountain car, there are three actions available applying -1, 0 or 1 torque on the joint between the two links. The **reward** obtained is also identical to mountain car, leading to -1 reward for not achieving the height per step and 0 once achieved. The **time horizon** T is 500 time steps and therefore the lowest reward, for not achieving the height, is -500. A good run hits the threshold between -80 to -100.

The parameters used in the algorithm all need to be similar for the comparisons to hold up. For the AGGREVATE algorithm, the number of iterations or the number of rollouts need to be high since it only obtains a single data point per rollout. Max depth has been established to be 1 for mountain car, 3 for cartpole and 2 for the acrobot. Increasing the max depth for all 3 leads to barely any improvement while drastically decreasing interpretability. The decision trees are further improved using cost complexity pruning[18], which removes the weakest link nodes to a certain extent. The value of this parameter was found using scikit build in functions and is applied to



(a) Behavioral cloning (b) DAGGER (c) GAIL (d) VIPER (e) AGGREVATE

Figure 2: Decision trees on mountain car

MC	BC	DAG.	GAIL	VIPER	AGG.
Reward	-120.8	-120.3	**-119.2**	-119.6	-120.4
SD	4.0	**3.6**	3.8	3.7	4.5
Nodes	3	3	3	3	3

Table 2: Results for mountain car with depth 1

Cartpole	BC	DAG.	GAIL	VIPER	AGG.
Reward	207.2	499.8	498.2	**500**	488.4
SD	44.2	0.7	14.12	**0**	22.0
Nodes	9	7	11	15(9)	**5**

Table 3: Results for cartpole with depth 3

Behavioral Cloning, DAGGER and AGGREVATE.

6 Results

The performance results and decision trees for GAIL and VIPER have been obtained from our colleagues Caspar Meijer and Otto Kaaij respectively. For more information on those results and how they were obtained see the papers on the interpretability of GAIL[15] and VIPER[12]. These results were obtained using the same openAI Gym environments. DAGGER, VIPER and AGGREVATE are based on the same kind of algorithm, they use the same parameters to obtain a comparison as reliable as possible. There is one discrepancy between the performance tests, namely the experts used to train the policies differ in quality between results from this experiment and those from our colleagues, the expert mentioned in the environment results are used for behavioral cloning, DAGGER and AGGREVATE. Another interesting fact is that the VIPER implementation does not prune its decision trees algorithmically, but decides to do this manually afterwards. The tables will show, in brackets, the manually pruned number of nodes if possible.

An explanation on how to read the decision trees can be found in appendix A.2, in that appendix all full size decision trees can be found as well.

Mountain car The parameters used to obtain these results were as follows. Number of samples per iteration was 100, there were 10 iterations and the ccp alpha used for pruning was found irrelevant since there was only one layer that would never be pruned.

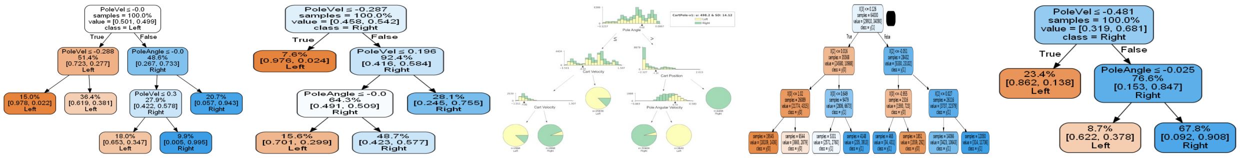
Table 2 shows the results obtained on mountain car to compare the objective metrics between the baseline Behavioral Cloning, DAGGER, GAIL, VIPER and AGGREVATE. The corresponding decision trees can be seen in figure 2. The expert had an average reward of -125 with a standard deviation of 13. From the table we can conclude that on this most simple environment there is not a big difference between all five imitation learning algorithms in terms of average reward, they all average around -120 score with limited standard deviation.

Looking at the objective interpretability there is no difference in the number of nodes, this is to be expected since the minimum as well as maximum of depth one is three nodes. Looking at the decision trees themselves, all five perform exactly the same operation to determine their outcome, they check if the speed is negative and if that is the case they will continue to go left to build more momentum. The moment the car's speed returns to zero because of gravity, all trees policies will determine to go right which should either get to the finish or do the entire process again to get more momentum.

Cartpole The parameters used to obtain these results were as follows, number of samples per iteration was 250, there were 20 iterations and the ccp alpha used for pruning was found to be 0.013 for cartpole.

Table 3 shows the results obtained on cartpole. The expert had an average reward of 475.5 with a standard deviation of 59.8. From this table we can conclude that all algorithms except for behavioral cloning performed really well. Behavioral Cloning performed way worse than the others, probably because that algorithm only learns the exact paths from the expert, but once it arrives in a state yet unknown, then it will not know the exact recovery action.

The objective interpretability of the algorithms on cartpole is very interesting to compare since the number of nodes differ a lot, see figure 3 for all decision trees. Behavioral cloning has the worst performance, it uses nine nodes while performing under half the time that the other algorithms perform. DAGGER performs almost perfect and has a medium number of nodes, with seven. Trying to understand the decision tree produced by DAGGER is already easier than Behavioral Cloning but is still a bit confusing, it checks on pole velocity twice right after each other before continuing on to check the angle. GAIL has the most amount of nodes and confusingly checks on cart velocity twice in a row. VIPER has a bad interpretability performance even when manually pruned. It performs perfectly, however, it is also quite difficult to understand. VIPER and GAIL are the only two algorithms producing a decision tree checking on the cart position and velocity, the other three algorithms did not use that part of the observation space and based their decision solely on the pole. AGGREVATE has a score slightly worse than DAGGER but objectively the most interpretable tree, it pruned the tree to only use a depth of two in comparison to three that the other decision trees used leading also to a smaller amount of nodes in general. The tree also makes sense to the human reader by checking on both pole parameters only once.



(a) Behavioral cloning (b) DAGGER (c) GAIL (d) VIPER (e) AGGREVATE

Figure 3: Decision trees on cartpole

Acrobot	BC	DAG.	GAIL	VIPER	AGG.
Reward	-94.5	-83.5	**-83.0**	-84.4	-85.5
SD	37.3	**10.8**	14.9	21.0	24.5
Nodes	5	5	7	7	**3**

Table 4: Results for acrobot with depth 2

Acrobot The parameters used to obtain these results were almost equal to cartpole, number of samples per iteration was also 250, there were 20 iterations just like cartpole only the ccp alpha used for pruning was found to be different at a value of 0.004.

Table 4 shows the results obtained on acrobot. The expert had an average reward of -180.5 with a standard deviation of 75. From this we can conclude that the expert is not trained consistently since it has a very high standard deviation. The imitation learning algorithms all function a lot better than the expert in terms of performance, this is probably because of the high standard deviation of the expert leading to the imitation learning algorithms learning from the best expert paths as long as the number of samples and iterations is high enough.

The objective interpretability of the algorithms on acrobot is quite similar, with only a small difference in nodes between the worst in VIPER and GAIL with seven nodes, the standard in Behavioral Cloning and DAGGER with five nodes and AGGREVATE with only three. Taking a closer look at the decision trees produced by these algorithms in figure 4, we can see that both five node trees can be easily reduced to three node trees since the two leaves corresponding to going right could be placed in a single leaf one layer up, without the check since the outcome will be the same. Another interesting comparison is that Behavioral cloning compares on the velocity of joint 2 while both DAGGER and AGGREVATE check on the velocity of joint 1.

7 Discussion

The simple experiments from section 6 are performed with parameters specifically found for AGGREVATE due its slow data collection. This leads to a large amount of rollouts which benefits AGGREVATE but is a disadvantage for the other algorithms. It should be noted that the experts of the GAIL and VIPER experiments were different leading to the comparisons being less conclusive.

The experiments show that AGGREVATE performs better than behavioral cloning and slightly worse in comparison to the other imitation learning algorithms in terms of performance. It performs slightly better than all the other imitation learning algorithms in terms of interpretability. The performance decrease could be explained through the fewer numbers of data points usable by AGGREVATE. This probably also leads to nodes being discarded more quickly while pruning developing a more interpretable tree less prone to overfitting.

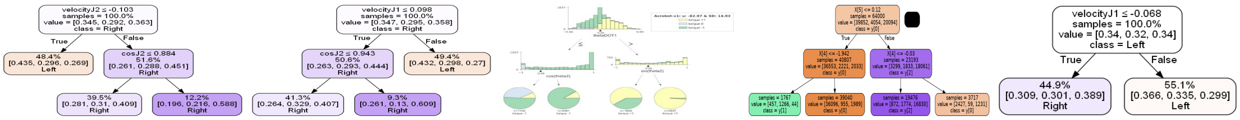
On mountain car there was only a relatively small difference between Behavioral Cloning and the four other imitation learning algorithms. All algorithms use the same number of nodes, since the depth only allows three nodes.

Cartpole showed that AGGREVATE performed a bit worse than the others, this might be because of the limited number of data points. Cartpole with its four state observations can quite easily get to a very specific combination of these observations that might not have been explored with the limited data points of AGGREVATE. This may lead to the policy not knowing what to do and failing near the end. They all function a lot better than Behavioral Cloning in terms of performance. AGGREVATE has the most clear decision tree buildup because it only checks on both pole parameters once and requires one less depth. The base VIPER tree is very hard to interpret and requires manual pruning to get to the same level of interpretability as behavioral cloning. GAIL is even more difficult to understand and confusing to read.

In the acrobot environment, all algorithms got a bit closer together again, in all comparison metrics. The most notable difference is that AGGREVATE pruned to a more clear 3 node tree, while some other algorithms stayed in a 5 node tree. VIPER and GAIL even went so far as to use all 7 nodes possible with depth 2. Both behavioral cloning and DAGGER could be reduced to a 3 node tree as well when using manual pruning. Performance wise, the values were not different enough to draw a strong conclusion.

The interpretability of the decision trees seems to be fairly similar on these simple environments. Mountain car is very interpretable for humans and explains very clearly the thought process with these parameters, namely that only the velocity matters to get to the end goal. Acrobot is slightly harder since these trees can be of depth two, leading to more predicates as well as the observations being very mathematical concerning angles and angular velocities in comparison to velocity and distance of a car(t). Cartpole is the least interpretable decision tree because of the number of different paths and due to it sometimes checking the same predicates twice, leading to possible confusion.

Looking forward at more difficult benchmarks, we expect AGGREVATE to start to perform better than DAGGER when multiple paths to the solution are available. This because we think that once there are multiple correct solutions with



(a) Behavioral cloning (b) DAGGER (c) GAIL (d) VIPER (e) AGGREVATE

Figure 4: Decision trees on acrobot

different levels of reward the exploratory actions performed by AGGREVATE should come to the correct conclusion on the best possible solution in comparison to DAGGER that might be stuck on one suboptimal solution.

The most significant limitation of AGGREVATE is how much time it takes to obtain a valuable data set to train on. Every sample only brings in one data point in comparison to time horizon T data points for DAGGER and other imitation learning algorithms per iteration. All these extra steps do entail to more exploration and possible better paths. So in short DAGGER is faster and more efficient, but AGGREVATE provides more specific information and eventual improvements. When combining the strongest elements of both algorithms, we can get an algorithm that produces initial efficient students using DAGGER that can then be trained on by AGGREVATE to gain the improvements of AGGREVATE with fewer of the time intensive trajectories that AGGREVATE is known for to get the best possible students.

Another interesting approach to machine learning is using online decision tree learning instead of batch learning. Using online learning the decision tree will be trained whenever a new data point arrives leading to the new data point already being integrated into the model before the next trajectory is started. This leads to all upcoming data points in that iteration to already be updated with the previous data points in that iteration.

Online learning makes the model more adaptable and data-efficient. A disadvantage of this approach is that the model will have to be trained every single data point, in practice this is much harder to implement and manage than batch training. Online learning is normally faster than batch learning, which could be an improvement to the current slow single sample algorithm. The problem with online learning is keeping track of new and old samples, balancing them and maintaining accuracy while training and continuously adapting the model.

8 Conclusions and Future Work

In this paper the imitation learning algorithm AGGREVATE is compared to a baseline Behavioral Cloning and other imitation learning algorithms GAIL, DAGGER and VIPER, in terms of interpretability and performance. To compare AGGREVATE in terms of interpretability, the algorithm has been modified to output a surrogate model in the form of a decision tree which can be evaluated on interpretability. The decision tree paths were compared using their performance as a way of establishing the different thought processes of the policies.

It could be concluded that in these three environments using the parameters defined AGGREVATE prunes slightly more than Behavioral Cloning and the other imitation learning algorithms leading to slightly more interpretable decision trees. In terms of performance, AGGREVATE performs better than behavioral cloning but slightly worse than the other algorithms. Both of the conclusions drawn here can be explained due to the smaller number of data points obtained by AGGREVATE. Fewer data points lead to more possible failing paths that have not yet been explored, but it also leads to a decision tree less prone to overfitting.

The parameters AGGREVATE is sensitive to are the number of iterations N and the number of samples M, since the AGGREVATE algorithm only obtains a single data-point every sample and having a larger sized data set improves performance substantially. The problem with this large number of iterations and samples is that every test requires a substantial amount of time to fully run.

The AGGREVATE algorithm performs better on tasks with optimal paths not instantly found by the policy, this is because of the random actions performed in the samples. With the random actions performed at a random time step t, it leads to more paths being explored which might lead to the optimal path not yet discovered. These kinds of environments have not been tested, and this conclusion is therefore purely theoretical.

Not everything could be researched in this paper due to time constraints. The current comparisons are on simple environments but where AGGREVATE improves on is random steps and their cost to go, therefore for future work more difficult environments can be fascinating to compare on.

Another compelling future research possibility is in the subjective interpretability comparison metrics, to fairly compare decision trees and their interpretability there could be a survey among users of a specific process and ask them whether the decision tree surrogate model is interpretable to them and if it increases their trust in the system. The metrics for this research could be accuracy of understanding, response time and confidence with which the survey participants answered questions about the decision trees.

A final interesting future possibility is the implementation of the combination of DAGGER and AGGREVATE into a single algorithm to improve performance in the early stages but still having the extensive searches of AGGREVATE. This does not directly improve the interpretability of the algorithm, however, it should increase accuracy and performance by a large margin.

9 Responsible Research

Ethical aspects

The fields of both reinforcement and imitation learning have a dangerous responsibility for the greater good of society. As mentioned in the introduction, examples of self-driving cars, automated job application processes and artificial intelligence in the military or police all lead to the question on how ethically responsible the algorithms and their training data are. These black box processes explain no inner workings of the model and can therefore lead to ethical and judicial problems.

The following examples show that the lack of insight in black box processes can be dangerous for society and therefore increases the lack of trust in these systems[7]. Self-driving cars might seem to be working as intended, but when faced with a situation unknown in the training data it might still make the wrong decision because it does not make the correct correlation that human drivers do. Bias in the training data of job application software might choose someone who is more fitting to previous employees instead of picking someone who is the best for the job. Automated military or police devices might wrongly identify someone, leading to innocent lives being ruined.

In this research, the emphasis is on making the black box processes interpretable to increase this aforementioned trust in the system and show explicitly the inner workings of the models. If interpretable models can become the norm in the future, we can understand why certain decisions are made and therefore counteract the possible bias in training data and guarantee the worst-case scenarios required to function in society.

Reproducibility

Reproducibility is a large problem in the field of multiple different sciences, as shown in a paper on the reproducibility crisis[3]. That paper states that in the physics and engineering field, almost 70% of reproduced experiments failed. From those shockingly high numbers, we can conclude that we should look critically at all papers used to obtain information and make sure the data and given information makes sense. All results in this paper are obtained through the same code and should be readily available once all code is complete to make this paper as transparent as possible and give further researchers easily reproducible data to check for themselves.

Research Integrity

The 'Netherlands Code of Conduct for Research Integrity (2018)' states there to be five main principles to guide research integrity: honesty, scrupulousness, transparency, independence and responsibility. We will briefly reflect on all these principles.

Honesty

This research has a lot of variables the output is dependent on, we have tried to report every important decision and result in this paper as accurately as possible. The conclusions drawn in this paper are based on the results obtained using a specific set of parameters, for other parameters these conclusions could maybe not hold due to the different nature of the imitation learning algorithms, we have tried to sufficiently make this clear while drawing conclusions.

Scrupulousness

We have tried to create the algorithm as close to the author's theoretical creation as possible. We are however no experts in the field of imitation learning algorithms. With the help of our supervisor and peer reviews, we believe the results are justified and scientific. We have tried to report all obtained data and results as clear as possible.

Transparency

To be as transparent as possible, all parameters used in all experiments were summarized, and all data was ordered as neatly as possible. All code should be available once everything is completed, leading to the most transparent possible results. All the reasoning and explanation has been attempted to be written as clear as possible, to make every step of the thought process as transparent as possible.

Independence

In the writing of this report, there has been no commercial or political nature involved. The paper is however not completely impartial since this paper has been written as the final product of a course. This means that the paper had to be shortened, leading to some observations or images being removed that would otherwise be published. Another aspect worth mentioning is the time frame we are set in, we only had 10 weeks to perform all research, experiments and write the entire report. There are still many questions that unfortunately could not be answered due to time constraints.

Responsibility

The imitation learning field, as mentioned before, has a lot of potential when looking at possible influence on society. We therefore understand the responsibility for delivering scientifically and societally relevant research. We think making imitation learning algorithms interpretable is of legitimate interests to most humans in society.

References

[1] Hiva Allahyari and Niklas Lavesson. User-oriented assessment of classification model understandability. In *11th scandinavian conference on Artificial intelligence*. IOS Press, 2011.

[2] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[3] Monya Baker. Reproducibility crisis. *Nature*, 533(26):353–66, 2016.

[4] Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction. *arXiv preprint arXiv:1805.08328*, 2018.

[5] Alina Beygelzimer, John Langford, and Bianca Zadrozny. Machine learning techniques—reductions between prediction quality metrics. In *Performance Modeling and Engineering*, pages 3–28. Springer, 2008.

[6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.

[7] Arun Das and Paul Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*, 2020.

[8] Tapio Elomaa. In defense of c4. 5: Notes on learning one-level decision trees. In *Machine Learning Proceedings 1994*, pages 62–69. Elsevier, 1994.

[9] Alex A Freitas. Comprehensible classification models: a position paper. *ACM SIGKDD explorations newsletter*, 15(1):1–10, 2014.

[10] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[11] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

[12] Otto Kaaij and Anna Lukina. Interpretability and performance of surrogate decision trees produced by viper. *TU Delft course CSE3000*, 2022.

[13] Swagat Kumar. Balancing a cartpole system with reinforcement learning–a tutorial. *arXiv preprint arXiv:2006.04938*, 2020.

[14] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[15] Caspar Meijer and Anna Lukina. Using decision trees together with generative adversarial imitation learning to give insight into black box reinforcement learning models. *TU Delft course CSE3000*, 2022.

[16] Andrew Moore. Knowledge of knowledge and intelligent experimentation for learning control. 1991.

[17] Richard M Murray and John Edmond Hauser. *A case study in approximate linearization: The acrobat example*. Electronics Research Laboratory, College of Engineering, University of . . . , 1991.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] Erika Puiutta and Eric M. S. P. Veith. Explainable reinforcement learning: A survey. *CoRR*, abs/2005.06247, 2020.

[20] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *Ieee Access*, 8:42200–42216, 2020.

[21] Stephane Ross and J Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*, 2014.

[22] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[23] Alexander Sieusahai and Matthew Guzdial. Explaining deep reinforcement learning agents in the atari domain through a surrogate model. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 17, pages 82–90, 2021.

[24] Girish H Subramanian, John Nosek, Sankaran P Raghunathan, and Santosh S Kanitkar. A comparison of the decision table and tree. *Communications of the ACM*, 35(1):89–94, 1992.

[25] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation, 2018.

[26] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

A Appendix

A.1 DAGGER algorithm

Algorithm 2 Base algorithm of AGGREVATE

```
Initialize $D \leftarrow \emptyset, \hat{\pi}_1$ to any policy in $\prod$.
for $i = 1$ to N do
    Collect m data points as follows:
    for $j = 1$ to m do
        Sample uniformly t $\in \{1, 2, ..., T\}$
        Start new trajectory in some initial state
        Execute current policy $\pi_i$ up to time $t - 1$
        Take exploration action $a_t$ in state $s_t$ at time t
        Execute expert from time $t + 1$ to T and observe
        estimate of cost-to-go $\hat{Q}$ starting at time t
    end for
    Get dataset $D_i = \{(s, t, a, \hat{Q})\}$
    Aggregate datasets: $D \leftarrow D \cup D_i$.
    Train cost-sensitive classifier $\hat{\pi}_{i+1}$ on D
end for
Return best $\hat{\pi}_i$ on validation
```

A.2 Decision trees

The following information can be retrieved from a decision tree policy like in figure 5. The decision tree is trained with max depth 2 on the mountain car environment. From this decision tree we can conclude what action will be taken when certain conditions are met. Every node consists of four lines giving information about the model. The first line, the predicate, checks whether the velocity of the car is negative or positive, if the predicate is satisfied for the state in which the model is in then the model will check the next rule in the node following the 'True' arrow. The second line shows the percentage of how many of the samples are in the current node, in the first node this will always be 100% but from the second layer onward you can see how often the model gets into that state. The third line shows the probability of going left, doing nothing or going right in that specific node and the fourth line shows the action corresponding to the highest probability of line three.

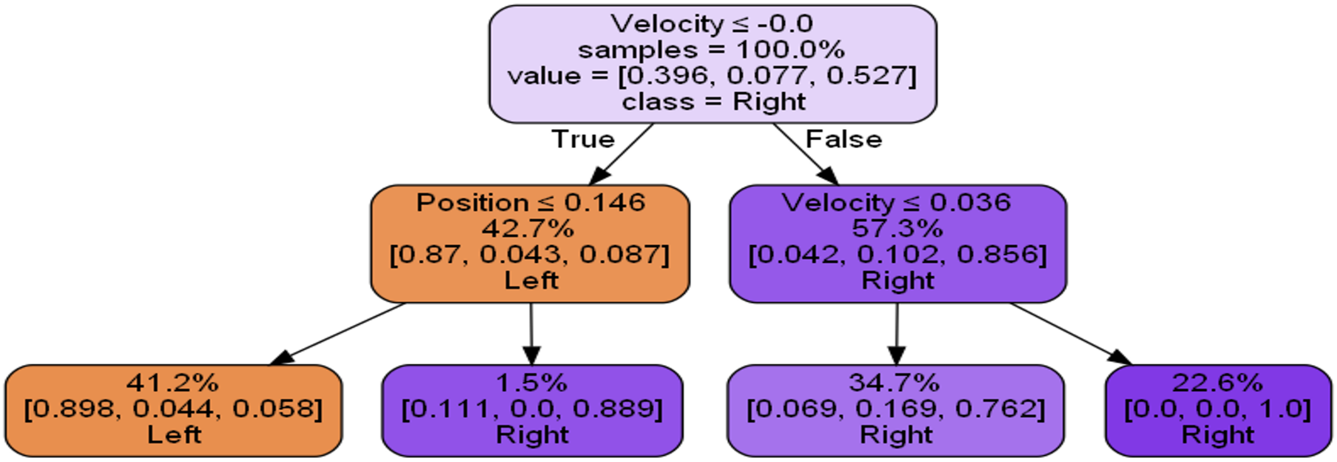
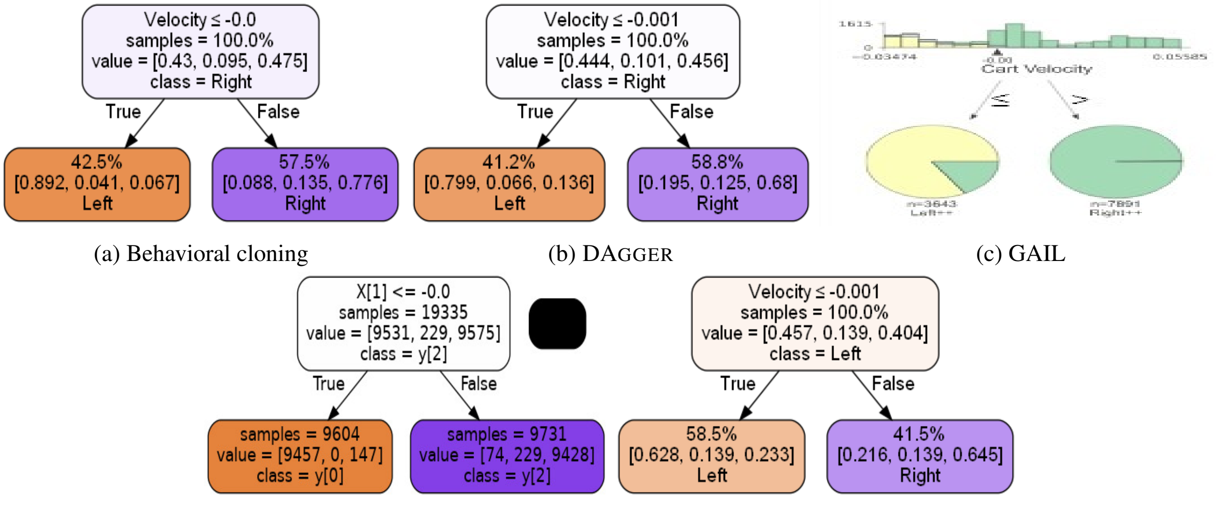


Figure 5: Decision tree corresponding to the mountain car environment



(a) Behavioral cloning

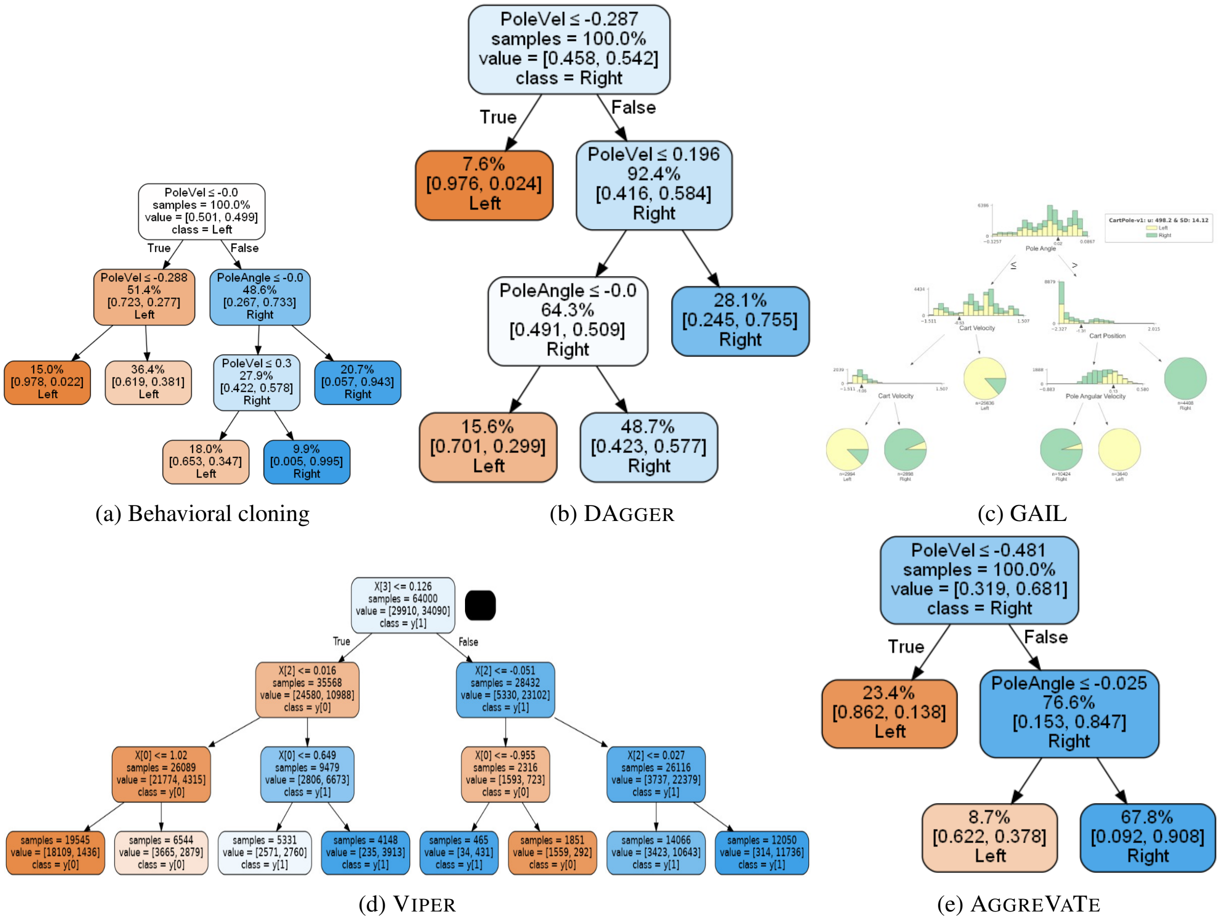
(b) DAGGER

(c) GAIL

(d) VIPER

(e) AGGREVATE

Figure 6: Decision trees on mountain car



(a) Behavioral cloning

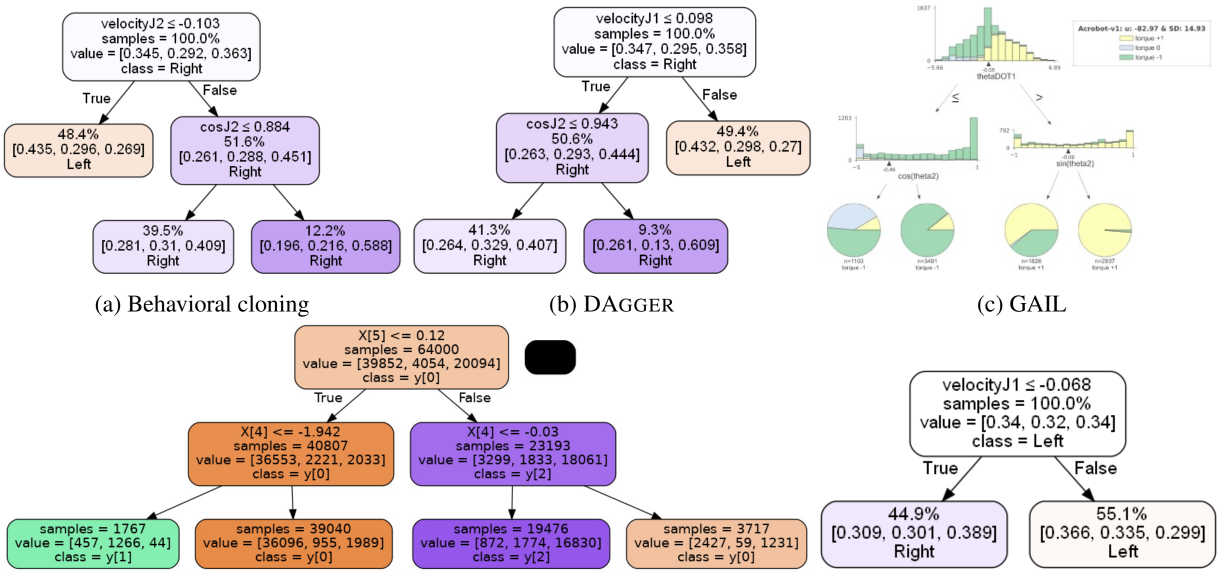
(b) DAGGER

(c) GAIL

(d) VIPER

(e) AGGREVATE

Figure 7: Decision trees on cartpole



(a) Behavioral cloning

(b) DAGGER

(c) GAIL

(d) VIPER

(e) AGGREVATE

Figure 8: Decision trees on acrobot