EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning

Kunal Menda1  Katherine Driggs-Campbell2  Mykel J. Kochenderfer1

Abstract

Imitation learning for robotics is risky, as it frequently suffers from data mismatch and compounding errors. DAgger is an iterative algorithm that addresses these issues by aggregating training data from both the expert and novice policies, but does not consider the impact of safety. We present a probabilistic extension to DAgger, which attempts to quantify the confidence of the novice policy as a proxy for safety. Our method, EnsembleDAgger, approximates a Gaussian Process using an ensemble of neural networks. Using the variance as a measure of confidence, we compute a decision rule that captures how much we doubt the novice, thus determining when it is safe to allow the novice to act. With this approach, we aim to maximize the novice’s share of actions, while constraining the probability of failure. We demonstrate improved safety and learning performance compared to other DAgger variants in the MuJoCo HalfCheetah environment.

1. Introduction

To be truly intelligent, robotic systems must have the ability to learn by exploring their environment and state space in a safe way (Amodei et al., 2016). One method to guide exploration is to learn from expert demonstrations (Price & Boutilier, 2003; Schaal, 1997; Kober & Peters, 2010). In contrast with reinforcement learning, where an explicit reward function must be defined, imitation learning guides exploration through expert supervision, allowing a robot to effectively learn from direct experience (Argall et al., 2009). However, such supervised approaches are often suboptimal or fail when the policy that is being trained (referred to as the novice policy) encounters situations that are not adequately represented in the dataset provided by the expert (Daumé et al., 2009; Ross & Bagnell, 2010). While failures may be insignificant in simulation, safe learning is important when acting in the real world (Amodei et al., 2016).

There are several methods for guided policy search in imitation learning settings (Levine & Koltun, 2013). One example is DAgger, which improves the training dataset by aggregating new data from both the expert and novice policies (Ross & Bagnell, 2010). DAgger has many desirable properties, including online functionality and theoretical guarantees. This approach, however, does not guarantee safety. Recent work extended DAgger to address some inherent drawbacks (Kim & Pineau, 2013; Laskey et al., 2016). In particular, SafeDAgger augments DAgger with a decision rule policy to provide safe exploration while minimizing queries to the expert (Zhang & Cho, 2016).

The shared goal of these methods is to efficiently train the novice to control the system while minimizing expert intervention. These algorithms assume that by allowing the novice to act, the system will likely deviate from the expert trajectory set and sample a new state. There is a chance, however, that the state visited is unsafe, or is a failure state. If the expert acts instead, we assume that the system will move along a safe trajectory, which is likely through states similar to those previously observed. The goal of this paper is to present an algorithm that maximizes the novice’s share of actions, while constraining the probability of failure.

Ideally, the proximity to a failure state (measured as an $l_2$-distance or likelihood of encountering the state under some operating condition) is known, and a safety envelope can be computed to guarantee safety (Akametalu et al., 2014). In the case of model-free learning, such guarantees are much more difficult to make. If we consider the novice action to be a perturbed form of the expert action, then we hypothesize that for many systems, the magnitude of permissible perturbation to expert actions is related to the distance from unsafe regions. Further, in a model-free case where expert demonstrations are available, we hypothesize that there is an inverse relationship between a state’s similarity to those in expert trajectories and allowed perturbations. We visualize this intuition in Figure 1. In the left panel, we see that the maximum permissible deviation from an expert action should be low as the system approaches a wall, which is considered a dangerous state. In such settings, experts will likely prefer trajectories that maintain some margin of distance from unsafe states. Under this assumption, it follows that in unfamiliar states, the system is likely at higher risk of entering failure states, and thus it is safer to allow the expert to act. While in familiar regions, it is permissible for the novice to act with large deviation from expert action.
We demonstrate how our method out-performs existing DAgger variants in an imitation learning setting. This paper makes two key contributions: (1) we present EnsembleDAgger, a Bayesian extension to DAgger, which introduces a probabilistic notion of safety to minimize expert intervention while constraining the probability of failure; and (2) we demonstrate the utility of this approach with improved performance and safety in the MuJoCo HalfCheetah domain.

2. Background

2.1. DAgger and SafeDAgger

The DAgger framework extends traditional imitation learning by simultaneously running both an expert policy that we wish to clone and a novice policy we wish to train (Ross et al., 2011). By aggregating new data from the expert, the underlying model and reward structure are uncovered.

Using supervised learning, we train an initial novice policy \( \pi_{\text{nov,0}} \) on some initial training set \( D_0 \) generated by the expert policy \( \pi_{\text{exp}} \). With this initialization, DAgger iteratively collects additional training examples from a mixture of the expert and novice policy. During a given episode, the combined expert and novice system interacts with the environment under the supervision of a decision rule. The decision rule decides at every time-step \( t \) whether to use the action from the novice or expert to interact with the environment (Figure 2). The observations \( o_t \) received during each epoch and the expert’s choice of corresponding actions make up a new dataset called \( D_t \). The new dataset of training examples is combined with the previous sets: \( D \leftarrow D \cup D_t \), and the novice policy is then re-trained on \( D \).

By allowing the novice to act, the combined system explores parts of the state space further from the nominal trajectories of the expert. In querying the expert in these regions, the novice is able to learn a more robust policy. However, allowing the novice to always act risks the possibility of encountering an unsafe state, which can be costly in real-world experiments. The VanillaDAgger algorithm and SafeDAgger balance this trade-off by their choice of decision rules.

Under VanillaDAgger, the expert’s action is chosen with probability \( \beta_i \in [0, 1] \), where \( i \) denotes the DAgger epoch. If \( \beta_i = \lambda \beta_{i-1} \) for some \( \lambda \in (0, 1) \), then the novice takes increasingly more actions each epoch. As the novice is given more training labels from previous epochs, it is allowed greater autonomy in exploring the state space. The VanillaDAgger decision-rule does not consider any similarity measure between the novice and expert actions. Hence, even if the novice suggests a highly unsafe action, VanillaDAgger allows the novice to act with probability \( (1 - \beta_i) \).

The “optimal” decision-rule approximated by SafeDAgger, denoted SafeDAgger*, computes the discrepancy between the expert and novice and allows the novice to act if the distance between the actions is less than some chosen threshold \( \tau \) (Zhang & Cho, 2016). Though this decision rule is claimed to be optimal, we argue that it has a shortcoming.

An ideal decision rule would allow the novice to act if there is a sufficiently low probability that the system can transition to an unsafe state. If the combined system is currently near an unsafe state, the tolerable perturbation from the expert’s choice of action is smaller than when the system is far from unsafe states. Hence, in practice, the single threshold \( \tau \) employed in SafeDAgger* is either too conservative when the system is far from unsafe states.

To reduce the number of expert queries, SafeDAgger approximates the SafeDAgger* decision rule using a deep policy that determines whether or not the novice policy is likely to deviate from the reference policy. Unlike SafeDAgger, we are not concerned with minimizing expert queries during a given episode. Hence, we compare to the SafeDAgger* decision rule directly, as opposed to the approximation.

Figure 1. Visualization of the tradeoffs between familiarity and risk. (left) Example scenarios of where perturbations are (not) permissible due to low (high) risk. Red trajectories illustrate expert corrections and the blue trajectory illustrate novice actions. (right) Plots visualizing the ideal tradeoff between distance to failure state and allowed deviations and the approximate of this tradeoff using similarity to expert demonstrations and deviations.

Figure 2. Flowchart for DAgger variants, where the decision rule differs between approaches.
We restate the assumptions made to explain why the this

We present the EnsembleDAgger decision rule, in which

we use Bayesian deep learning.

Also, it is assumed that the expert policy is primarily uni-

-modal, as is commonly assumed in most imitation learning

settings. Further, using a neural network based dissimilarity

measure is useful for imitation learning as neural networks

scale more gracefully to high-dimensional input spaces and

large datasets than most non-parametric measures.

Given that we have a measure of doubt via the variance on

novice actions, we ideally would like to specify the bound

on discrepancy as a monotonically increasing function of

doubt. To meet this end, we have experimented with the idea of

making the discrepancy bound proportional to the inverse

of doubt. However, the parameters specifying an arbitrary

function mapping doubt to a discrepancy bound must be

considered hyperparameters to the algorithm and tuned by

the practitioner. We opt for the low-order approximation to

the ideal functional mapping, shown in Figure 3, because

the two hyperparameters, $\chi$ and $\tau$, are easy to interpret.

By appropriately choosing the hyper-parameters $\tau$ and $\chi$,

we satisfy the dual objectives of allowing the novice to act

only if it is sufficiently confident in its action and close to

the expert. As $\chi \to \infty$, the decision rule converges to that

of SafeDAgger*. As $\tau \to \infty$, the decision rule ignores

discrepancy, and allows the novice to act if it is confident

without comparison to what the expert action is. However,

since the novice is only confident in states similar to those

in $\mathcal{D}$, it is likely that the novice having low doubt causes

its action to also have low discrepancy, implying that the

algorithm is less sensitive to an arbitrary increase in $\tau$ than

to an arbitrary increase in $\chi$. This statement is qualified in

the next section by showing that using the doubt rule alone

(by setting $\tau = \infty$) leads to better performance than using

the discrepancy rule alone (by setting $\chi = \infty$).

4. Experiments

In this section, we present experimental validation for the

following claims we have made:

- Using the doubt rule alone, as opposed to the discrepancy
rule alone, trains a better performing novice policy
for the same compromise to the combined (expert and novice) system’s safety.

- Combining the two decision rules in EnsembleDAgger
improves the trained novice policy performance while
making the combined system strictly safer.

We justify these two claims on the MuJoCo HalfCheetah
OpenAI Gym environment.

4.1. MuJoCo HalfCheetah domain

The MuJoCo HalfCheetah-v1 domain is an OpenAI Gym
environment with observations in $\mathbb{R}^{18}$ and actions in
$\mathbb{R}^5$ (Brockman et al., 2016). We train an expert policy

or too relaxed when near them. To approximate the ideal
decision rule in a model-free manner, we propose not just
considering the distance between the novice’s and expert’s
actions, but also the uncertainty in the novice policy at a
given state. To estimate the uncertainty of the novice policy,
we use Bayesian deep learning.

3. EnsembleDAgger

We present the EnsembleDAgger decision rule, in which
the discrepancy between the expert’s and the novice’s mean
action, as well as the novice’s doubt, which is variance of
the novice’s action, are used to decide whether to choose the
novice action. According to the EnsembleDAgger decision
rule, the novice must satisfy two conditions in order to act. The first is that the discrepancy between the novice and expert’s action, i.e. $||a_{nov,t} - a_{exp,t}||^2$, must be less than some threshold $\tau$. This is the SafeDAgger* decision rule, but will henceforth be referred to as the discrepancy rule. Assuming the novice policy outputs a variance on its predicted action $\sigma_{nov,t}^2$, as an ensemble of neural networks would, then the second condition is that $\sigma_{nov,t}^2$ is less than some threshold $\chi$. We refer this condition as the doubt rule. As shown in Figure 3, in order for the novice to act according to the EnsembleDAgger decision rule, it must satisfy both the discrepancy rule and the double rule. The algorithm is parameterized by the values $\tau$ and $\chi$.

We restate the assumptions made to explain why the this
decision rule is able to better guarantee the system’s safety:

1. The expert prefers trajectories that avoid failure states, and rarely visits near failure states, implying that states dissimilar to those in expert trajectories (or states unfamiliar to the novice) are likely to be in closer proximity to failure states.

2. Following from (1), and by capturing epistemic uncertainty, or lack of familiarity with states in the training dataset, the novice’s doubt provides a model-free proxy for proximity to failure states.

3. In order to constrain the probability of encountering a failure state, the discrepancy between the action taken and the expert’s action is less than some bound.

4. The ideal bounds should be state-dependent, such that the bound is tighter in close proximity to failure states.

5. Following from (2, 4), the bound on discrepancy should decrease as the novice’s doubt increases.

Also, it is assumed that the expert policy is primarily unimodal, as is commonly assumed in most imitation learning settings. Further, using a neural network based dissimilarity measure is useful for imitation learning as neural networks scale more gracefully to high-dimensional input spaces and large datasets than most non-parametric measures.

Given that we have a measure of doubt via the variance on

novice actions, we ideally would like to specify the bound

on discrepancy as a monotonically increasing function of

Figure 3. The EnsembleDAgger decision rule is parametrized by doubt ($\chi$) and discrepancy ($\tau$) bounds, and is a low-order, model-free approximation to the ‘ideal’ decision rule, shown in green.

$\mathcal{D}$, it is likely that the novice having low doubt causes its action to also have low discrepancy, implying that the algorithm is less sensitive to an arbitrary increase in $\tau$ than to an arbitrary increase in $\chi$. This statement is qualified in the next section by showing that using the doubt rule alone (by setting $\tau = \infty$) leads to better performance than using the discrepancy rule alone (by setting $\chi = \infty$).
on this domain using the TRPO algorithm from the rlalab codebase (Duan et al., 2016). The goal is to learn a stable gait, with a reward for the distance from the origin reached. The purpose of this experiment is to compare the doubt rule, discrepancy rule, and a combination of the rules, in their ability to safely learn a policy that matches the expert score. In this experiment, we use an ensemble of five neural networks as the novice policy being trained. The system is trained for seven epochs. Each epoch samples one additional trajectory of interaction with the environment, followed by re-training the novice policy on the aggregated dataset. When training, the score over a trajectory of the lone novice and the system combined under the experiment’s decision rule are queried at each epoch. Queries average the score of the policy being tested over 20 trajectories.

We define the performance of an instance of a given decision rule by the performance of the lone novice and the performance of the combined system, which are defined as follows. The performance of the lone novice is the average score of the novice, trained under a given decision rule instance, summed over the seven epochs of training. A better performing lone novice implies that the decision rule instance is able to quickly bring the novice to expert-level scores. Similarly, the performance of the combined system is the average score of the expert and novice, combined under a given decision rule instance, summed over the seven epochs that train the novice.

Though we have no strict notion of safety in this domain, a better performing combined system implies that trajectories perturbed under the decision rule instance are still high-scoring, are thus compromising states that one may consider to be failure states are being better avoided. We compare decision rules based on their ability to maximize the novice performance while compromising the performance of the combined system as little as possible. We sample the performance of each instance of a decision rule 100 times, presenting their mean and standard errors.

We test the doubt rule using values of $\chi = [0.02, 0.05, 0.1, 0.2, 0.5]$, the discrepancy rule using values of $\tau = [0.2, 0.5, 1.0, 2.0, 5.0]$, and the full EnsembleDAgger decision rule at $(\tau, \chi) = [(0.2, 0.02), (0.5, 0.05), (1.0, 0.1), (2.0, 0.2), (5.0, 0.5)]$. Though we sample $(\tau, \chi)$ only on a line in the positive quadrant of $\mathbb{R}^2$, this ratio between $\tau$ and $\chi$ is chosen so the two rules are approximately equally responsible for preventing the novice action in the first training epoch.

The performance of the various decision rules for the parameters stated are shown in Figure 4 in the form of Pareto frontiers (since varying the hyperparameter trades-off between performance of the combined system and of the lone novice). We see that the doubt rule Pareto dominates the discrepancy rule, as the rule’s frontier achieves better novice performance for the same compromise on the combined system’s performance.

The fact that the doubt rule Pareto dominates the discrepancy rule in terms of performance is consistent with the trends observed in the inverted pendulum experiment—the doubt rule constrains the novice to act only when the state is familiar. Consequently, the perturbation from an expert action caused by choosing the novice’s action is unlikely to compromise the score of the overall trajectory, though it will likely carry the system into marginally more unfamiliar territory, thereby allowing the novice to learn a more robust policy. The doubt rule, however, allows an arbitrarily large perturbation in sufficiently familiar states, and thereby can still lead to unsafe states. There exist settings of $\chi$ and $\tau$ that can make the EnsembleDAgger decision rule safe in all states, bounding the maximum perturbation from the expert action even in very familiar states. We only sample values of $\tau$ and $\chi$ along a line in $\mathbb{R}^2$, and hence do not find that points along this line show strict improvement over the independent decision rules in all cases, but see slight improvement in novice performance over the doubt rule for the case of $(\tau, \chi) = (0.5, 0.05)$. Additionally, we find the EnsembleDAgger decision to be strictly more conservative than either of the component decision rules and thus always improves the combined system performance, as expected.

5. Conclusion

In this work, we presented an extension to the DAgger algorithm that considers the safety of the novice-expert system that provides the trajectories from which the novice learns. We assume an action proposed by the novice is safe if two conditions are met: that the action is both close to the expert’s choice of action, and that the novice has little doubt in its choice. We demonstrated that the use of doubt as a decision rule alone allows for both safe and efficient imitation of an expert policy, and can have improved performance when combined with a decision rule on discrepancy.

![Figure 4. Performance of various instances of the doubt rule, discrepancy rule, and combined EnsembleDAgger decision rules on the HalfCheetah domain. Consistent markers indicate the instances of the doubt and discrepancy ruled in an instance of the EnsembleDAgger decision rule.](image-url)
References

Akametalu, A. K., Fisac, J. F., Gillula, J. H., Kaynama, S., Zeilinger, M. N., and Tomlin, C. J. Reachability-based safe learning with Gaussian processes. In IEEE Conference on Decision and Control (CDC), pp. 1424–1431, 2014.

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in AI safety. Available on arXiv:1606.06565, 2016.

Argall, B. D., Chernova, S., Veloso, M., and Browning, B. A survey of robot learning from demonstration. Robotics and Autonomous Systems, 57(5):469–483, 2009. ISSN 0921-8890.

Berkenkamp, F., Moriconi, R., Schoellig, A. P., and Krause, A. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In IEEE Conference on Decision and Control (CDC), pp. 4661–4666, 2016.

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. Safe model-based reinforcement learning with stability guarantees. In Advances in Neural Information Processing Systems (NIPS), pp. 908–919, 2017.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI Gym. Available on arXiv:1606.01540, 2016.

Daumé, H., Langford, J., and Marcu, D. Search-based structured prediction. Machine Learning, 75(3):297–325, 2009.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In International Conference on Machine Learning (ICML), pp. 1329–1338, 2016.

Khalil, H. K. Feedback linearization. In Nonlinear Systems, chapter 13, pp. 505–508. Prentice Hall, Upper Saddle River, N.J, 2002.

Kim, B. and Pineau, J. Maximum mean discrepancy imitation learning. In Robotics: Science and Systems, 2013.

Kober, J. and Peters, J. Imitation and reinforcement learning. IEEE Robotics Automation Magazine, 17(2):55–62, June 2010. ISSN 1070-9932.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In Advances in Neural Information Processing Systems (NIPS), pp. 6405–6416, 2017.

Laskey, M., Staszałk, S., Hsieh, W. Y.-S., Mahler, J., et al. Shiv: Reducing supervisor burden in DAgger using support vectors for efficient learning from demonstrations in high dimensional state spaces. In IEEE International Conference on Robotics and Automation (ICRA), pp. 462–469, 2016.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as Gaussian processes. Available on arXiv:1711.00165, 2017.

Levine, S. and Koltun, V. Guided policy search. In International Conference on Machine Learning (ICML), pp. 1–9, 2013.

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

Price, B. and Boutlier, C. Accelerating reinforcement learning through implicit imitation. Journal of Artificial Intelligence Research, 19:569–629, 2003.

Rasmussen, C. E. and Williams, C. K. I. Gaussian Processes for Machine Learning. MIT Press, 2005.

Ross, S. and Bagnell, D. Efficient reductions for imitation learning. In International Conference on Artificial Intelligence and Statistics, pp. 661–668, 2010.

Ross, S., Gordon, G. J., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In International Conference on Artificial Intelligence and Statistics, pp. 627–635, 2011.

Schaal, S. Learning from demonstration. In Advances in Neural Information Processing Systems (NIPS), pp. 1040–1046, 1997.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1):1929–1958, 2014.

Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. Available on arXiv:1506.02142, 2015.

Zhang, J. and Cho, K. Query-efficient imitation learning for end-to-end autonomous driving. Available on arXiv:1605.06450, 2016.

Zhou, Z.-H., Wu, J., and Tang, W. Ensembling neural networks: many could be better than all. Artificial Intelligence, 137(1-2):239–263, 2002.
A. Appendix

A.1. Bayesian Approximation Methods

Recent research has focused on approximating GPs with neural networks (Lee et al., 2017). While GPs alone have shown great success in modeling uncertainty and approximating safety (Berkenkamp et al., 2016), traditional GP approaches are computationally expensive for high-dimensional feature spaces and large datasets (Rasmussen & Williams, 2005). Advances in deep learning have shown great success in handling these complexities. Two prominent methods for approximating GPs with deep neural networks are ensemble methods (Lakshminarayanan et al., 2017) and Monte-Carlo dropout (Y. Gal and Z. Ghahramani, 2015).

In this work, we chose to use the ensemble method, which is a technique for training a collection of neural networks to execute the same task and then combining the output into a single prediction. This approach has shown to significantly improve performance in practice (Zhou et al., 2002). There is a work that employed an ensemble of neural networks to approximate GPs and demonstrated that this is a more straightforward approach to estimate predictive uncertainty (PU) (Lakshminarayanan et al., 2017). Typically, neural networks predict point estimates of the output that are optimized to minimize the mean squared error on the training set. The authors claim that this approach does not capture irreducible, or aleatoric uncertainty, but only epistemic uncertainty. They propose using a proper scoring rule, like negative log-likelihood, as a loss function to train an ensemble in which each network predicts a mean and a variance of a Gaussian distribution over the output. They postulate that such loss functions provide a better measure of the quality of predictive uncertainty and thus reward better calibrated predictions.

A.2. Bayesian Approximation Techniques

We examine two approximations of GPs: ensemble methods and Monte-Carlo dropout.

Gal et al. propose approximating Bayesian models with neural networks trained with dropout (Y. Gal and Z. Ghahramani, 2015). By applying dropout at every weight layer in a network, an approximation of a Gaussian process is obtained. Given a policy trained with dropout, the network can be queried $N$ times per input observation to obtain a distribution over actions, using randomly sampled dropout masks (Y. Gal and Z. Ghahramani, 2015; Srivastava et al., 2014).

The ensemble method is a technique for training a collection of neural networks to execute the same task and then combining the output into a single prediction. This approach has shown to significantly improve performance in practice (Zhou et al., 2002). (Lakshminarayanan et al., 2017) employed an ensemble of neural networks to approximate GPs and demonstrated that this is a more straightforward approach to estimate predictive uncertainty (PU). Typically, neural networks predict point estimates of the output are optimized to minimize the mean squared error on the training set. The authors claim that this does not capture aleatoric uncertainty, but only epistemic uncertainty. They propose using a proper scoring rule, like negative log-likelihood, as a loss function to train an ensemble in which each network predicts a mean and a variance of a Gaussian distribution over the output. They postulate that such loss functions provide a better measure of the quality of predictive uncertainty and thus reward better calibrated predictions.

A.2.1. Empirical Evaluation

To determine which approximation approach is most effective in practice, we evaluate the ability of four different methods to learn the function $f(x) = \sin(\pi x) + 0.2\sin(4\pi x)$ with only eight samples. The codebase used for the evaluation (as well as the proposed algorithms) is provided in the supplementary material.

First, we fit the function with a traditional Gaussian process to act as a baseline. We use a squared-exponential kernel with a length-scale of 10. The kernel parameters chosen are the best of nine optimizer restarts (Pedregosa et al., 2011). Then, we fit a ten network ensemble trained with Mean-Square-Error (MSE) loss for 300 epochs, referred to as the Vanilla Ensemble. The same ensemble is trained with Negative-Log-Likelihood (NLL) loss for 2400 epochs, so each network directly predicts uncertainty. Finally, a single network is trained using Monte-Carlo Dropout with MSE loss and a keep-probability of 75%, also trained for 2400 epochs.

All neural network models have hidden layers of size $[128, 64, 64, 64]$, respectively. ADAM with a learning rate of $10^{-3}$ is used to optimize the vanilla ensemble and MC-dropout, while a learning rate of $10^{-4}$ is used for the ensemble with predictive uncertainty. No weight regularization or batch normalization are used. We use a batch size of 4.

Each model is queried for a mean and standard deviation for its estimate of $f(x)$ for $x \in [-1.5, 1.5]$. The standard deviations of each model are scaled such that their sum matches that of the GP, for ease of visual comparison. The results of each method are shown in Figure 9.

We see that the vanilla ensemble of models and ensemble with predictive uncertainty have the most visual similarity to a GP. We also note that the vanilla ensemble achieves this performance in a small fraction of the number of epochs with which the latter two models are trained. Therefore, in our experiments, the novice neural network architecture takes the form of an ensemble of neural networks, and we train the neural network with MSE loss.

It should be noted that the implementations did not utilize adversarial training, as recommended by Lakshminarayanan, et al. (Lakshminarayanan et al., 2017). Additionally, the training data is noiseless, which does not highlight the poten-
initial conditions are sampled uniformly during the successive epochs of DAgger. The dynamics and control law are provided in the supplementary material.

The neural network model representing the novice policy is an ensemble of ten multi-layer perceptrons, each with four hidden layers of size \([64, 64, 32, 32]\) respectively. At each DAgger epoch, the ten networks are each trained for 200 training epochs with a learning rate of \(10^{-3}\), \(l_2\)-weight regularization of \(10^{-5}\), and a mini-batch size of 16. The maximum length of any trajectory is 100. No dropout or batch normalization is used. Since the data labeled by the deterministic expert is noise-free, the networks do not individually predict variance and are trained with MSE loss.

In order to compare the two decision rules, we are interested in analyzing the regions of the state space in which they allow the novice to act. We define the permitted set for some decision rule, given some novice and expert policies, to be the set of states in which the decision rule chooses the novice action. In Figure 16, states in the permitted set are shown as black circles. Similarly, we define the permitted set volume to be the fraction of states grid-sampled in \(\theta \in [-\pi, \pi], \dot{\theta} \in [-5, 5]\) that are in the permitted set of a given decision rule, given some novice and expert policies. Additionally, we define the novice basin of attraction to be the set of states \(\mathcal{X}_0\) from which, if the novice is initialized in \(\mathcal{X}_0\) and allowed to act alone (without the help of the expert), the novice converges to the origin.

In order to make an apples-to-apples comparison between the two decision rules, we provide a budget, and analyze how the two decisions utilize this budget. The budget chosen is a fixed volume for the permitted set. At each epoch, since the novice has learned from more data, we linearly grow the permitted set volume budget. Prior to each episode, we solve for the value of \(\chi\) and \(\tau\) that will make the doubt and discrepancy rules respectively yield permitted sets with the desired volume. These values are found using bisection search.

The goal of this experiment is to show that, for some fixed volume permitted set, the doubt rule allocates that volume in the neighborhood of states represented in \(\mathcal{D}\), justifying the claim that the novice’s output variance is a good measure of dissimilarity between the query state and familiar states. Additionally, we show that the discrepancy rule haphazardly allocates volume to regions of the state space in which the novice and expert agree by chance, indicating that it is wastefully conservative in some regions of the state space, while not conservative enough in others.

In an additional experiment that follows this, we compare the decision rules in a manner meaningful to a practitioner, by fixing the hyperparameters \(a\ priori\) and keeping them fixed over all epochs. For both experiments on this domain, we control the random seed specifying the initial condition for each epoch such that it varies across epoch but is the same regardless of decision rule. The trajectory followed from that initial condition will, of course, depend on the decision rule. In all experiments, as in all variants of DAgger,
we initialize $D$ with a zeroth epoch where only the expert is queried for the action, and the decision rule is used from the first epoch onward (Ross et al., 2011).

Figure 16 shows the evolution of the permitted set under the doubt rule and the discrepancy rule for the first three epochs of an experiment. Under the doubt rule, the permitted set is concentrated in the neighborhood of the labeled states in $D$. This is because the variance of the function fitting $D$ grows as we move away from labeled states, so the permitted set is constrained to be within some neighborhood of labeled states under the doubt rule. On the other hand, under the discrepancy rule, the permitted set is more haphazardly distributed over the state space with a smaller portion of the allotted volume being in the neighborhood of labeled states. We observe this because there exist arbitrary regions of the state space in which the function fitting the $D$ happens to intersect the true control law purely by chance, leading to low discrepancy in these, often dangerous, regions.

We can see in Figure 16 that the trajectories resulting under the doubt rule carry the system to the edge of a familiar region of the state space, after which the expert is handed control to navigate unfamiliar regions. This behavior leads to a novice basin of attraction that is much larger than under the discrepancy rule, while no trajectories enter dangerous territory. However, under the discrepancy rule we see that the novice is rarely allowed to carry the system away from an expert trajectory, thereby aggregating a dataset that is not much more likely to be informative than behavior cloning. This observation qualitatively suggests that the doubt rule can train a better performing novice policy for the same level of compromise to the combined (expert and novice) system’s safety. This claim will be justified in the next experiment.

A.3.1. Controller Details

The inverted pendulum has a two-dimensional state space of $[\theta, \dot{\theta}]$ and a one-dimensional action space of $u$, as shown in Figure 14.

The inverted pendulum is guided by the following dynamical equation:

$$\dot{\theta} = \frac{1}{a} \sin \theta - b \dot{\theta} + cu \tag{1}$$

The system can be driven to the equilibrium at $[0, 0]$ by feedback linearization, using the following control law:

$$u = -\frac{a}{c} \sin \theta - \frac{1}{c} \begin{bmatrix} K_1 & K_2 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{2}$$

With feedback linearization, the gain vector $K$ is computed to stabilize the linear system specified by the new dynamics:

$$\begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -b \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{3}$$

where $v_1$ and $v_2$ are the residual linear terms. The resulting controller is deterministic but sufficiently non-linear to pose an interesting learning problem. The dynamics of this
The permitted set, i.e. states at which the novice is allowed to act, denoted by black circles, of the doubt rule is concentrated in the neighborhood of states represented in \( D \), where as the permitted set of the discrepancy rule is distributed more haphazardly across the state space. States in which the novice alone is able to converge to the origin are indicated in pink.

We consider the problem instance in which \( a = 10, b = 2, c = 10 \). Additionally, the control \( u \) is saturated to lie within \([-1, 1]\). The gains \( K \) are found to be \([0.316, 0.175]\) by using a linear quadratic regulator with cost function \( J = \int_0^T \theta^2 + \dot{\theta}^2 + 10u^2 \). Due to the control saturation, the controller does not converge to the desired fixed point from an arbitrary initial condition, but has the basin of attraction shown in Figure 15. Figure 15 also shows the region of the state space from which initial conditions are sampled uniformly during the successive epochs of DAgger.

### A.3.2. Selecting Hyperparameters

In the inverted pendulum experiment, we solve for the hyperparameters \( \chi \) and \( \tau \) at every DAgger epoch such that the corresponding decision rules create a permitted set of some desired volume. Though useful for visually comparing the two decision rules, solving for the hyperparameters in this manner is not tractable in more complex problems with higher dimensional state-action spaces or with a non-deterministic expert. Hence, in our second experiment, we compare the behavior of the two decision rules in a manner more useful to a practitioner—in which the hyperparameters \( \chi \) and \( \tau \) are chosen a priori.

We introduce additional metrics of performance:

- **Learning Performance**: The fraction of states grid-sampled from \( \theta \in [-\pi, \pi], \dot{\theta} \in [-5, 5] \), in both the expert and novice basin of attractions.

- **Failure Rate**: The fraction of repetitions of a given experiment in which the trajectory acquired at a given epoch, for a given decision rule and expert policy, leaves the expert’s basin of attraction.

For a given choice of hyperparameters, the instances of the decision rules are compared over six epochs, and results are averaged over 30 repetitions of the experiment. During each epoch, we track the learning performance, failure rate, and permitted set volume.

Figure 13 shows the results for an instance of the doubt rule with \( \chi = 10^{-3} \), the discrepancy rule with \( \tau = 10^{-1} \) and the discrepancy rule with \( \tau = 5 \cdot 10^{-2} \). As shown in Figure 10, this instance of the doubt rule demonstrates superior learning performance to either instance of the discrepancy rule. In addition to demonstrating more rapid learning, this instance exhibits no failures in any of the six epochs in any of the repeated experiments, as shown in Figure 11. Neither instance of the discrepancy rule is failure-free, and choosing the more conservative \( \tau \) reduces the failure rate at the expense of learning performance.

It is also interesting to note the evolution of the permitted
set volume over the six epochs for fixed hyperparameter choices. It appears that in this domain, the permitted set volumes grow monotonically, which matches expectations. However, we can see that the permitted set volumes for the discrepancy rule are many times larger than that of the doubt rule. This confirms the observations made in the previous experiment: the doubt rule is less permissive in allowing the novice to act, but is nonetheless able to generative more informative trajectories.

The results of this experiment confirm the observations made in the first inverted pendulum quantitatively. As a consequence of the discrepancy rule being too conservative in states familiar to the novice, the system is prevented from entering states that are informative to the novice’s learning, and hence we see poor learning performance. Furthermore, since the discrepancy rule is not conservative enough in risky states, the discrepancy rule encounters failures more frequently than the doubt rule.